Likimääräinen hahmon etsintä bittirinnakkaisesti

Paavo Koskinen, Teppo Niinimäki

Johdanto

Likimääräisessä hahmonsovituksessa etsitään tekstistä hahmon kaltaisia esiintymiä. Samankaltaisuutta mitataan editointietäisyydellä. Käyttökohteita löytyy esimerkiksi oikeinkirjoituksen tarkastuksessa ja bioinformatiikassa.

Vertasimme kolmen eri algoritmin käytännön nopeuksia erilaisilla aineistoilla. Kaksi algoritmeista, Wu-Manber ja Myers, hyödyntävät bittirinnakkaisuutta. Kolmas vertailtava, Ukkosen algoritmi, perustuu rajoitettuun dynaamiseen ohjelmointiin.

Editointietäisyys

Kahden merkkijonon välinen editointietäisyys on pienin määrä muokkausoperaatioita, joilla ensimmäinen jono saadaan muunnettua toiseksi (tai toisin päin). Sallittuja operaatioita ovat merkin lisäys, merkin poisto sekä merkin muuttaminen toiseksi.

Ukkosen algoritmi

Ukkosen algoritmi perustuu rajoitettuun dynaamiseen ohjelmointiin. Algoritmissa täytetään $n \times k$-taulukkoa A, missä n on tekstin pituus k on suurin sallittu editointietäisyys. Kuhunkin taulukon alkioon A_{ij} lasketaan hahmon P alkuosan $P[1..i]$ ja tähän parhaiten täsmäävän tekstin T positioon j päättyvän osajonon välinen editointietäisyys. Taulukon alkion A_{ij} arvo voidaan laskea kaavalla:

$$A_{ij} = \min \begin{cases} A_{i-1,j-1}, \text{ jos } P[i] = T[j], \text{ muuten } A_{i-1,j-1}+1 \\ A_{i-1,j}+1 \\ A_{i,j-1}+1 \end{cases}$$

Taulukko voidaan laskea sarakkeittain, jolloin riittää pitää viimeisin sarakkeellinen alkioita muistissa. Sarakkeesta riittää laskea keskimäärin $O(k)$ ensimmäistä alkiota. Algoritmin keskimääräinen aikavaativuus on $O(nk)$.

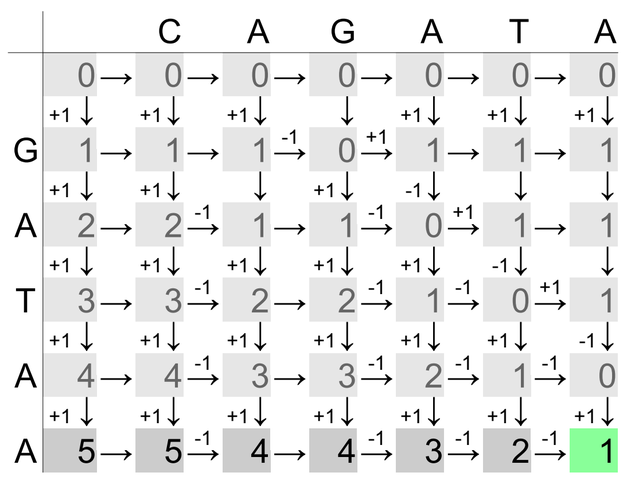
Jos sarakkeen j viimeisen alkion arvoksi tulee enintään k, löytyy tekstistä osuma, joka päättyy indeksiin j.

		C	A	G	A	T	A	A	G	A	G	A	A
	0	0	0	0	0	0	0	0	0	0	0	0	0
G	1	1	1	0	1	1	1	1	0	1	0	1	1
A	2	2	1	1	0	1	1	1	1	0	1	0	1
T				2	1	0	1	2	2	1	1	1	1
A						1	0	1	2	2	2	1	1
A							**1**	**0**	**1**	2			**1**

Tekstistä CAGATAAGAGAA haetaan hahmoa GATAA, osumat on merkitty vihreällä.

Myersin algoritmi

Ukkosen algoritmin taulukossa vierekkäisten alkioiden ero voi olla korkeintaan yksi. Myersin algoritmissa lasketaan solujen arvojen sijaan niiden vaaka- ja pystyerotuksia. Kunkin erotuksen tallentamiseen riittää kaksi bittiä. Myers on kehittänyt menetelmän laskea sarakkeen erotukset edellisestä sarakkeesta bittirinnakkaisesti. Algoritmi pitää kirjaa sarakkeen viimeisestä alkiosta ja päivittää sitä laskettujen erotusten pohjalta. Algoritmin aikavaativuus on $O(n)$, kun hahmon pituus on enintään koneen sanan pituus.



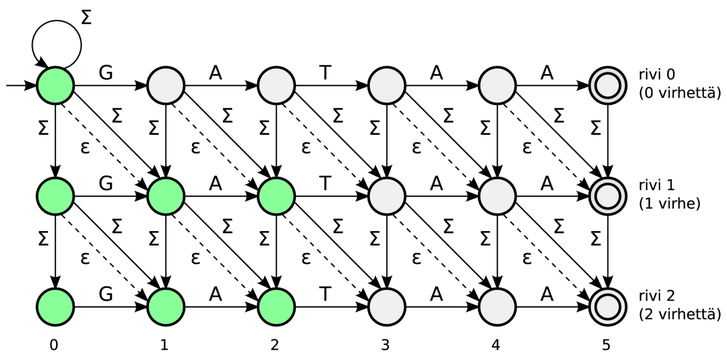
Erotukset osalle edellisen esimerkin taulukkoa.

Wun-Manberin algoritmi

Wun-Manberin algoritmi pohjautuu dynaamisen ohjelmoinnin sijaan epädeterministisen äärellisen automaatin simulointiin. Automaatti saa syötteenään tekstin T. Automaatin sarakkeen i rivin k tila vastaa tilannetta, jossa hahmon alkuosa $P[1..i]$ täsmää tekstin käsiteltävään kohtaan k:lla virheellä. Jos jokin automaatin oikean laidan tiloista on aktiivinen, löytyy tekstistä osuma, joka päättyy viimeksi luettuun merkkiin.

Siirtymä oikealle vastaa tilannetta, jossa $P[i+1]$ on syötteestä luettava merkki. Siirtymä alas vastaa luetun merkin poistamista ja ε-siirtymä alas oikealle merkin lisäämistä. Mielivaltaisen merkin korvaamista vastaa Σ-siirtymä alaviistoon.

Kunkin rivin tilojen aktiivisuutta pidetään tallessa omassa bittivektorissaan. Automaatin toimintaa voidaan näin simuloida bittirinnakkaisesti, jolloin aikavaativuus on O(nk), kunhan hahmon pituus on enintään koneen sanan pituus.

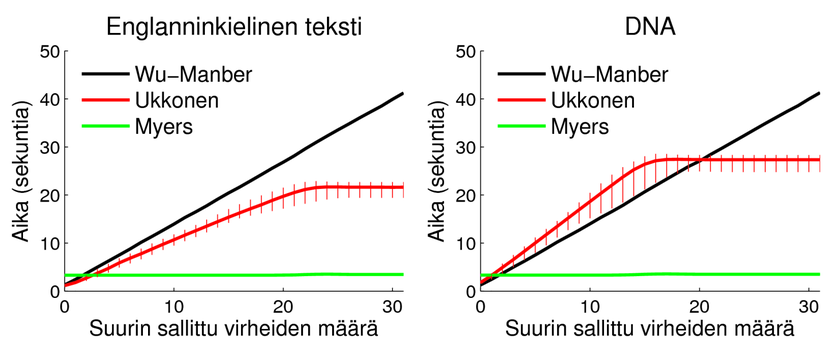


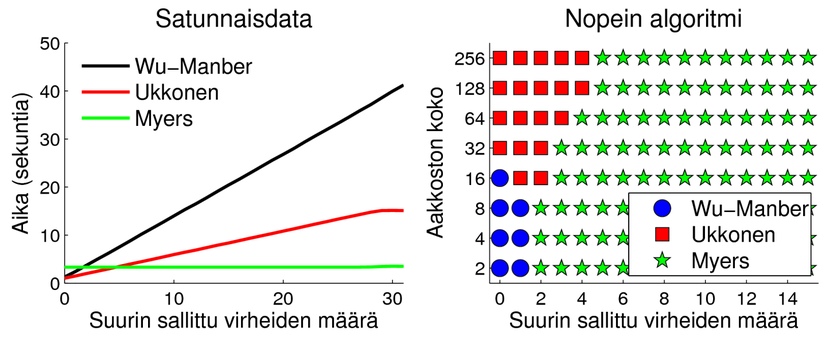
Hahmon GATAA automaatti, kun syötteestä on luettu merkit C ja A.

Testaus

Testasimme algoritmien toteutusten nopeutta kolmella eri tyyppisellä testiaineistolla: englanninkielisellä tekstillä, DNA-sekvensseillä ja satunnaisesti muodostetulla aineistolla.

Testit suoritettiin hahmon pituuksilla 1, 3, 7, 15, 31 ja 63. Hahmoiksi valittiin satunnaisia osajonoja aineistoista. Kullekin pituudelle sallittua virheiden määrää vaihdeltiin nollasta hahmon pituuteen. Tulokset eri hahmon pituuksilla olivat lähes identtiset. Kolmessa ensimmäisessä kuvassa hahmon pituus on 31 merkkiä. Satunnaisdatalla aakkoston koko on 256. Viimeisessä kuvassa algoritmeja on verrattu satunnaisdatalla eri kokoisilla aakkostoilla.





Yhteenveto

Wun-Manberin algoritmi pärjää parhaiten pienillä aakkostoilla ja virhetoleransseilla. Aakkoston koon kasvaessa Ukkosen algoritmi kirii nopeimmaksi. Mikäli virheitä sallitaan muutamaa enemmän, on Myersin algoritmi selvästi nopein. Kuvaajat noudattavat hyvin algoritmien aikavaativuusluokkia. Ukkosen algoritmin ajantarve lakkaa kasvamasta sallitun virhemäärän mukana, kun kaikki taulukon arvot joudutaan joka tapauksessa laskemaan. Lisäksi ajantarve vaihtelee hieman eri hahmoilla toisin kuin Myersin ja Wun-Manberin algoritmeilla.

Lähteet

Ukkosen algoritmi ja Myersin algoritmi:
Kärkkäinen, Merkkijonomenetelmät-kurssin luentomuistiinpanot, 2008

Wun-Manberin algoritmi:
Navarro ja Raffinot, Flexible Pattern Matching in Strings – Practical on-line search algorithms for texts and biological sequences, 2002

Helsingin yliopisto, Tietojenkäsittelytieteen laitos
Merkkijonomenetelmät, syksy 2008