Ctl160 Tekstikorpusten tietojenkäsittely 490160-0 Kolmas luento

Nicholas Volk

Yleisen kielitieteen laitos, Helsingin yliopisto

HELSINGIN YLIOPISTON
YLEISEN KIELITIETEEN LAITOS

Lisää kuvausvoimaa: `egrep` eli `grep -E`

- Viime kerralla tutustuimme `fgrep`-käskyyn, joka pystyi poimimaan rivejä halutun osajonon perusteella
- Kuvausvoimaa lisäämään voidaan käyttää säännöllisiä lausekkeita (regular expressions, regexp)
- Säännölliset lausekkeet ovat eri asia kuin komeontitulkin jokerimerkit (wildcards) ’*’, ’?’...
- Tietyillä, pian opittavilla merkeillä on tietty säännöllisissä lausekkeissa erikoismerkitys
- `egrep`-käsky hyödyntää säännöllisiä lausekkeita osumia etsiessään
- Niin tekee myös tavallinen `grep`, mutta sitä emme käsittele tällä kurssilla (syntaksi vain on vähän erilainen)

Mikä tahansa merkki

- Säännöllisten lausekkeiden piste tarkoittaa mitä tahansa yhtä merkkiä
- Tavallinen piste kirjoitetaan `\.`
- `tr ' ' '\n ' | egrep -x "....."` poimisi kaikkia viisikirjaimiset sanat
- (vrt. bash-komentotulkin '?'-jokerimerkki)
- Yhden tietyn merkin ja minkä tahansa merkin väliin sijoittuu merkkijoukko, jossa merkki voi olla mikä tahansa annettuun joukkoon kuuluva merkki

Merkkijoukko

- Merkkijoukko koostuu hakasulkeiden sisään kirjoitetuista merkeistä, esim. suomen vokaalit: `[aeiouyäö]`
- (Bash-komentotulkin jokerimerkit taas samat)
- Esim. lauseke `pa[dr]at` osuu, kun syötteessä on joko `padat` tai `parat`.
- Hakasulkeet tarvitaan (toisin kuin `tr`-käskyssä)
- Merkkijoukon komplementti saadaan kirjoittamalla `^` välittömästi vasemman hakasulun perään: `[^aeiouyäö]`
- Jos hakasulku ei ilmaise merkkijoukon alkua tai loppua pitää (tai ainakin kannattaa) sen eteen kirjoittaa kenoviiva: `\[` ja `\]`

- merkkijoukossa

- Kuten `tr`-käskyssä, voidaan merkkijoukossa käyttää ’-’-merkkiä merkityksessä x:stä y:hen: `[0-9]`
- Jos halutaan viitata `merkkiin`, ei sen erityismerkitykseen, kirjoitetaan se `egrep`-käskyssä merkkijoukon ensimmäiseksi tai viimeiseksi merkiksi
- Perlissä toimii kenoviivan laitto `-`:n eteen: `\-`

Määrällistäjät: optionaalisuus

- Merkin, merkkijoukon ja suluilla rajatun alueen voi tehdä valinnaiseksi kirjoittamalla sen perään kysymysmerkki `?`
- Kysymysmerkkiin merkkinä viitataan kirjoittamalla `\?`
- Lingvistisesti motivoituneita esimerkkejä ei ole (vielä) helppo keksiä:

```
egrep "Amerii?kka"
egrep "ka?akk?u"
egrep "märj?ät"
egrep "ve[dr]?et"
```

Kleenen plus ja tähti

- Jos annettua merkkiä, merkkijoukkoa tai suluilla rajattua aluetta seuraa plus-merkki, tämä voi esiintyä 1:stä äärettömään kertaa
- Esimerkiksi positiiviset kokonaisluvut voisi määrittää seuraavasti: `[0-9]+`
- Tähti tarkoittaa nollasta äärettömään kappaletta.
- Parempi määritelmä kokonaisluvuille olisi `[1-9][0-9]*`
- Merkkeihin itseensä viitataan taas kenoviivan avulla: `\+` ja `\*`
- `Aa+rgh?`

Universaalikieli

- Mikä tahansa merkkijono eli universaalikieli voidaan nyt kuvata kahdella merkillä seuraavasti: `.*`
- Eli äskeinen onnistuu aina, myös tyhjällä merkkijonolla, kun taas `.+` onnistuu kaikella muulla paitsi tyhjällä merkkijonolla
- (Lauseke `.*` korreloi `*`-jokerimerkin kanssa, mitä nyt jälkimmäinen ei löydä Unixin piilotiedostoja...)
- `egrep "ei .* eikä"`

Tarkemmat määrän rajaukset

- Halutun asian perään kirjoitettujen aaltosulkeiden avulla toistojen määrä voidaan ilmaista tarkemmin:
- Tasan viisi kappaletta: `.....` ⇔ `.{5}`
- 5-7 merkkiä: `.....?.?` ⇔ `.{5,7}`
- 5 tai enemmän: `.....+` ⇔ `.{5,}`

Viittaukset rivin alkuun ja loppuun

- `^` viittaa säännöllisissä lausekkeissa merkkijonon alkuun ja `$` merkkijonon loppuun:

  ```
  egrep "^[A-ZÅÄÖ]"
  egrep "ss[aä]$"
  ```

- Yhdessä ne vastaavat `x`-optiota:

  `egrep -x "sana"` ⇔ `egrep "^sana$"`

- Rivinvaihtomerkkiin `\n` ei `egrep`-käskyllä voi viitata

- Sarkainmerkin voi kirjoittaa `egrep`-käskylle näppäilemällä `Ctrl-v TAB`

Erityismerkityksen poistosta

- Yllä on opetettu, että kun halutaan viitata merkkiin, jolla on jokin erikoismerkitys, niin sen eteen pitää laitaa kenoviiva.
- Tämä ei ole aina aivan totta...
- Esim. merkkijoukon sisällä piste, kysymysmerkki ja kumppanit eivät tarvitse kenoviivaa eteensä, sillä merkkijoukon sisällä ko. merkit ovat semanttisesti yksiselitteisiä, eli ne voivat tarkoittaa siellä vain yhtä asiaa.
- Samaten ^ tarkoitti eri asiaa lausekkeen ja merkkijoukon alussa (mitä?)
- Jos ei ole varma kontekstin sallimasta vapaudesta, kannattaa käyttää kenoviivoja...
  ... siis minun mielestäni, muitakin näkemyksiä on...

Ryhmittely

- Sulkujen avulla voidaan osoittaa "määrällistäjille"yhtä merkkiä suurempi alue:

```
(ha)+!
(h[aei])+!
(reduplikaatio){2}
10( Australian)? dollarilla
```

Ryhmittely 2

- Piippu-merkki `|` tarjoittaa valinnaisuutta
- Itse käytän sitä vain sulkujen sisällä:

```
san(oi|a)s[st]a
K(u|a)mpi voitti
M(\.|atti) Meikälainen
(pitää|täytyy|on pakko)
```

- Välillä näkee käyettävän myös muotoja:

```
((pitää)|(täytyy)|(on pakko))
pitää|täytyy|on pakko
```

Palindromiesimerkki

- `egrep`-käskyllä on mahdollista viitata taaksepäin aiemmin ryhmiteltyihin kokonaisuuksiin
- Viittaus tapahtuu kirjoittamalla `\n`, jossa n on 1-9.
- Esim. kuusikirjaimisen palindromin tunnistaa ehto
  `egrep "^(.)(.)(.)\3\2\1$"`
- Kiva tietää, ei tarvitse osata...

less

- Käskyllä `less` selata tiedoston sisältöä (mm. page up-, page down-, nuoli ylös- ja nuoli alas-näppäimet).
- `q`:n painaminen lopettaa.
- `h` antaa listan erilaisista näppäinyhdistelmistä.

less ja säännölliset lausekkeet

- Painamalla `/`-merkkiä pääsee kirjoittamaan säännöllisen lausekkeen.
- Lausekkeen kirjoittamisen ja enterin painamisen jälkeen `less` siirtyy ensimmäisen osuman kohdalle.
- `n` siirtyy seuraavaan osumaan.
- `N` siirtyy edelliseen osumaan.

Skriptaus ja editorit

- Usein käytetyt käskyjonot voi yhdistää "skriptiksi".
- Skripit kirjoitetetaan ja niitä muokataan jollakin editorilla.
- Tällä kurssilla käytämme GNU Emacs -editoria.
- Emacs käynnistyy käskyllä (laitoksen koneissa) käskyllä `emacs`.
- Luotava/muokattava tiedosto voidaan kertoa argumenttina: `emacs foo`.

Emacs-editorin ajamisesta

- Ikkunoinnin toimiessa Emacs-editorin käynnistyskäskyn perään kannattaa lisätä &-merkki.
- Tällöin prosessi haarautuu: voit jatkaa käskyjen antamista myös komentoriviltä.
- Ikkunoimattomassa ympäristössä jätetään &-merkki pois, muuten prosessi pyörii "taustalla".
- Ikkunoimattomassa ympäristössä ei voi käyttää graafista käyttöliittymää, joten ainakin muutama perusnäppäinyhdistelmä on syytä osata.
- (Taustalla pyörivän prosessin saa esiin käskyllä `fg`.)

Keskeisiä näppäinyhdistelmiä

- Tärkeimmät käskyt löytyvät graafisista valikoista, mutta niitä ei voi aina käyttää. Siksi on tarkoituksenmukaista osata muutama näppäinyhdistelmä.
- Tiedoston talletus: `CTRL-x CTRL-s`
- Emacsin sulkeminen: `CTRL-x CTRL-c`
- Emacsista poistuminen sulkematta Emacsia: `CTRL-z` (paluu käskyllä `fg`)
- Peruutus: `CTRL-x u`
- Emacsin hyödyllisistä näppäinyhdistelmistä enemmän kurssin WWW-sivujen yleiset materiaalit -kohdassa.

Emacsin hakutoiminnot

- `fgrep`-mäinen haku eteenpäin: `CTRL-s sana`
- Sama uudelleen: `CTRL-s CTRL-s`
- `fgrep`-mäinen haku taaksepäin: `CTRL-r sana`
- Merkkijonon korvaaminen: `M-%` (lue `ALT-SHIFT-5`)
- Lausekkeen korvaaminen: `C-M-%`
  (lue `CTRL-ALT-SHIFT+5`)

Bash-skriptien teko

- Shell-skripti on komentotulkille kelpaavista käskyistä koottu tiedosto, johon on luku- ja suoritusoikeudet.
- Esim. `chmod a+rx` *`skriptin_nimi`*
- Tällöin kutsumalla tiedoston nimeä voi käyttää käskynä.
- Hyvä tiedostopääte shell-skriptille on `.sh`
- Skriptin ajaminen edellyttää ohjelman suoritusoikeuden lisäksi myös ohjelman lukuoikeuden.
- Konekielisen tiedoston ajamiseen riittää siis pelkkä suoritusoikeus. (Konekieltä ei ole tarkoitettu ihmisen tai Unixin komentotulkin luettavaksi.)

PATH

- Ympäristömuuttujassa `$PATH` on ':'-merkillä eroteltu ne hakemistot, joista komentotulkki etsii käytettäviä käskyjä.
- `echo $PATH`
- `which komento` kertoo missä päin hakemistorakennetta `komento` sijaitsee.
- Riippumatta `$PATH` muuttujan arvosta työhakemistossa olevaa oikein oikeuksin varustettua skriptiä voi kutsua käskyllä `./nimi`
- Vastaavasti muitakin suhteellisia ja absoluuttisia polkuja suoritettavaan tiedostoon voi käyttää.

Shell-skriptien teko (2)

- Shell-skriptien siirreltävyyden varmistamiseksi käytämme vanhempaa sh-komentotulkkia bash-komentotulkin sijaan.
- Käsky `which sh` kertoo missä paikassa hakemistorakennetta shell-komentotulkki sijaitsee.
- Skriptin ensimmäisellä rivillä kerrotaan mikä ohjelma tulkitsee skriptin ja siinä olevat käskyt.
- Meillä shell-skriptin ensimmäinen rivi on: `#!/bin/sh` tai `#!/bin/bash`

Moodit

- Emacs-editorissa on moodeja, jotka helpottavat ohjelmien kirjoittamista.
- Emacs päättelee joskus tiedostopäätteen tai tiedoston ensimmäisen rivin perusteella moodin.
- shell-moodin saa päälle kirjoittamalla
  `M-x sh-mode`
  jossa iso M-kirjain tarkoittaa meta-näppäinta.
- `M-x` voi kirjoittaa joko painamalla `Alt-x` tai painamalla ensin `Esc` ja hetken päästä `x`-näppäintä.
- `M-x global-font-lock-mode` kytkee värit päälle tai pois päältä.

Shell-skriptin argumentit: ongelma

- Omallekin shell-skriptille voi välittää argumentteja (optiota, tiedostonnimiä...)
- Argumentit tallettuvat muuttujiin $1, $2 ... $9 ja kollektiivisesti muuttujiin $* ja $@
- Argumentteja voi olla enemmän kuin yhdeksän, mutta näiden opettelu ei kuulu tälle kurssille (`shift`).
- Skriptien tulisi siis osata huolehtia argumenttien käsittelystä...
- Ongelma on samanlainen kuin ensimmäisellä luennolla esitelty `tr`-käskyn kykenemättömyys saada tiedostoja argumentteina

Shell-skriptit argumentit: tiedostoargumentit

- Tiedostoargumenttien osalta ongelman voi ratkaista laittamalla skriptin ensimmäiseksi käskyksi rivin
  `cat $@ |`
- Tällöin `cat`-käsky yhdistää argumenttitiedostonsa ja laittaa ne oletussyöttönä seuraavalle käskylle
- Komentotulkin kommentin aloittaa muuten risuaitamerkki ’#’. Kommentit helpottavat elämää...
- Se komentotulkista, lisää emacs-asiaa ensi kerralla