22.12.2013
Ville Lahdenvuo
Testausdokumentti

NodePieSpy

https://github.com/tuhoojabotti/NodePieSpy

1. Sisällys

2. Johdanto

NodePieSpy on työkalu, jonka avulla voi analysoida sosiaalisia verkkoja. Ohjelmisto toimii tällä hetkellä vain irssin (http://irssi.org/) lokien perusteella, mutta se on tulevaisuudessa laajennettavissa esimerkiksi Facebook- tai Github-suhteiden kanssa toimivaksi.

Tällä hetkellä ohjelmisto koostuu kahdesta komponentista. NodePieSpy on palvelinohjelma, joka hoitaa tiedon lataamisen ja käsittelyn, ja tarjoaa HTTP:n yli rajapinnan, jolla tietoja voi pyytää JSON-formaatissa. NodePieSpy-Client on HTML-sivu, joka käyttää kyseistä rajapintaa ja visualisoi tulokset. Tämä käyttöliittymä ei ole harjoitustyön osana, vaikka sitä tullaankin käyttämään projektin testauksessa ja esittelyssä.

Projektin inspiraationa toimii PieSpy: http://www.jibble.org/piespy/

3. Ominaisuudet

Tässä osiossa selitän miten mitäkin ominaisuutta on testattu.

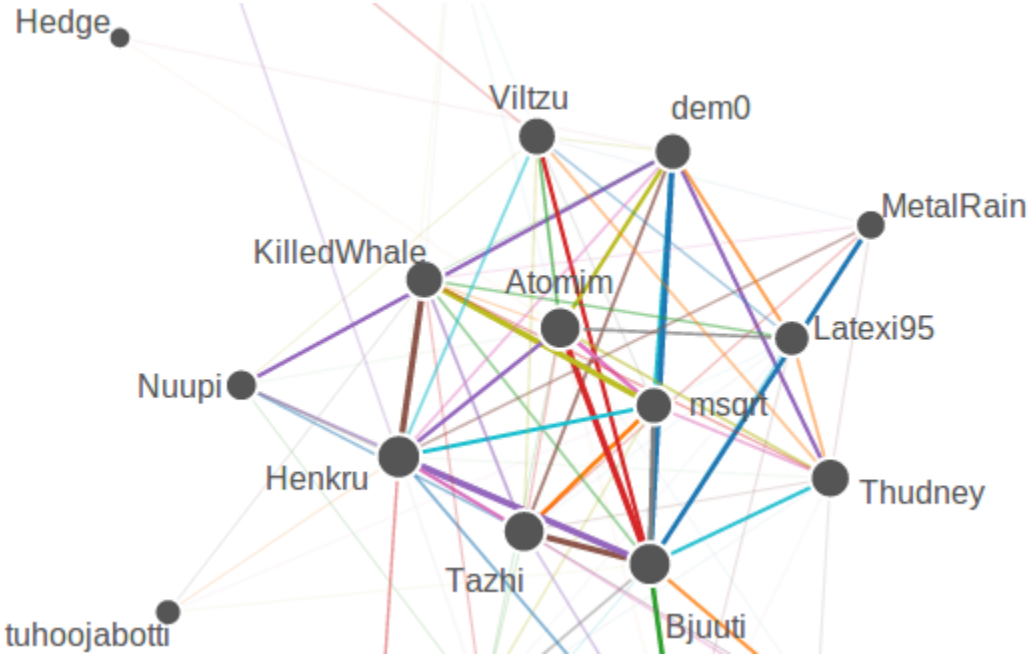
22.12.2013
Ville Lahdenvuo
Testausdokumentti

3.1. Päättelyheuristiikat

Tärkein osa koko ohjelmistossa on itse tietojen jäsentäminen sosiaaliseksi verkoksi. Tiedot tulevat Irssi IRC-asiakasohjelman lokitiedostoista.

3.1.1. Vierekkäisyys

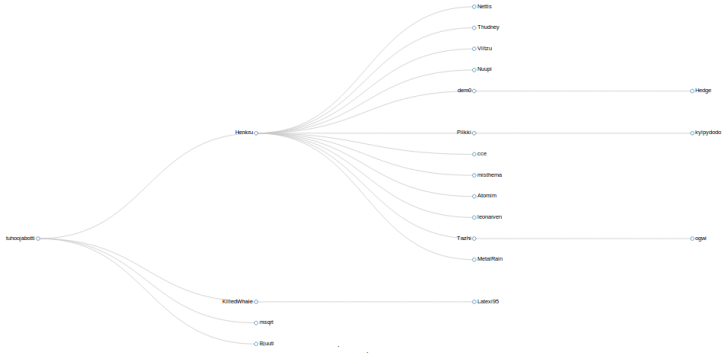
Testattu lähinnä silmämääräisesti asiakasohjelmalla ja konsolitulostuksilla.



3.5. Analysointialgoritmit

3.5.1. BFS-puu

Puun testaus on myös toteutettu silmämääräisesti ensin HTTP-vastauksia tulkitsemalla ja sitten graafisen implementaation



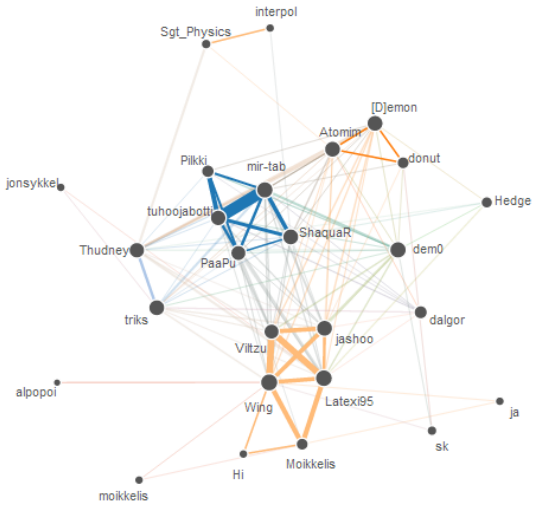
tarkastamisella.

22.12.2013
Ville Lahdenvuo
Testausdokumentti

Implementaatiosta löytyikin bugi, joka aiheutti väärän puun generoimisen. https://github.com/tuhoojabotti/NodePieSpy/blob/ff705a192fb6cee5ed465de9cd9ea4661083dbb5/lib/algorithms/bfs.js#L25 visited-taulukon arvon asetus oli alunperin virheellisesti toistorakenteen jälkeen. Tästä aiheutui se, että jo läpikäytyjä solmuja käytiin läpi useammin ja puun järjestys meni rikki.

3.5.2. Ryvästäminen

Testaaminen suoritettiin aluksi json datan silmämääräisellä tulkitsemisella. Eli tarkistettiin näyttääkö syntymä matriisi yhtään oikeanlaiselta esimerkkitulosteeseen verrattuna. Aluksi ajattelin, että syntyvää matriisia voisi ajatella perus vierusmatriisina, mutta tämä ei johtanut mihinkään. Vähän pohdiskeltuani kirjoitin koodin, joka tulkitsee matriisin ja lisää solmuille ryhmätunnisteen, jota käytetään visualisoinnissa:



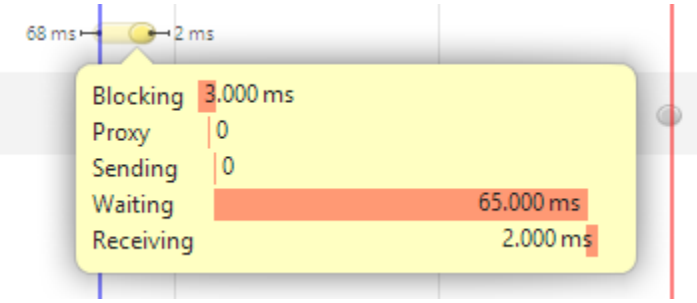
4. Suoritusteho

Ohjelman oltua päällä 2 viikkoa se oli kerännyt huomattavat määrät dataa, joka aiheutti käyttöliittymän hidastumista suuremmilla keskustelukanavilla. Kuitenkin itse backend pysyy suorituskykyisenä. Mikäli ohjelmasta tulee suositumpi, on myös mahdollista ottaa käyttöön aliprosessien käyttö, eli jokaista suoritinydintä kohden luotaisiin oma prosessi.

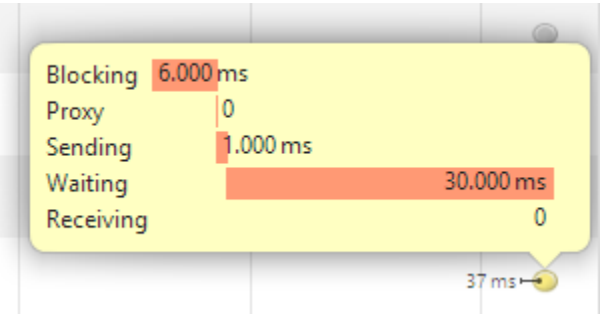
4.1. rivejä 10 000

Ladataan lokeista 10 000 riviä ja tutkitaan kuinka nopeasti palvelin vastaa:

22.12.2013
Ville Lahdenvuo
Testausdokumentti



Kuten kuvasta näkyy, verkon lataaminen palvelimelta ei kestä kovin kauaa.



Samoin BFS:n generoiminen tapahtuu silmänräpäyksessä.

4.2. ryvästämisen nopeus

En ollut tarkistanut, että algoritmi toimii oikein. Oli päässyt käymään niin, että vaikka tulos oli jo valmis, jatkoi algoritmi toistamista turhaan, sillä tarkistusfunktio ei toiminut. Pienillä verkoilla tämä ei ollut ongelma, sillä niiden kanssa 700 iteraatiotakin pyörähti silmänräpäyksessä, mutta suuremmilla verkoilla se jumitti koko palvelimen, jonka korjasin lisäämällä viisi prosessia palvelimeen ennen demotilaisuutta.

Demotilaisuuden jälkeen minulla oli aikaa pureutua testaamaan klusterointialgoritmia enemmän ja huomasin virheen, jonka jälkeen klusterointi onnistuu yleensä aina alle kymmenellä iteraatiolla ja alle sekunnissa suurellakin kanavalla: