TUGAS AKHIR - IS184853

ANALISIS JEJARING SOSIAL TERHADAP CALON ANGGOTA LEGISLATIF DALAM PEMILIHAN UMUM 2019 BERDASARKAN DATA KOMISI PEMILIHAN UMUM (KPU)

SOCIAL NETWORK ANALYSIS ABOUT LEGISLATIVE CANDIDATES IN GENERAL ELECTION 2019 BASED ON KOMISI PEMILIHAN UMUM (KPU) DATA

KARIMA MUFIDAH
NRP 05211640000045

Dosen Pembimbing
Nur Aini Rakhmawati, S.Kom., M.Sc. Eng., Ph.D

DEPARTEMEN SISTEM INFORMASI
Fakultas Teknologi Elektro dan Informatika Cerdas
Institut Teknologi Sepuluh Nopember
Surabaya 2020

TUGAS AKHIR - IS184853

ANALISIS JEJARING SOSIAL TERHADAP CALON ANGGOTA LEGISLATIF DALAM PEMILIHAN UMUM 2019 BERDASARKAN DATA KOMISI PEMILIHAN UMUM (KPU)

KARIMA MUFIDAH
NRP 05211640000045

Dosen Pembimbing
Nur Aini Rakhmawati, S.Kom., M.Sc. Eng., Ph.D

DEPARTEMEN SISTEM INFORMASI
Fakultas Teknologi Elektro dan Informatika Cerdas
Institut Teknologi Sepuluh Nopember
Surabaya 2020

UNDERGRADUATE THESIS - IS184853

SOCIAL NETWORK ANALYSIS ABOUT LEGISLATIVE CANDIDATES IN GENERAL ELECTION 2019 BASED ON KOMISI PEMILIHAN UMUM (KPU) DATA

KARIMA MUFIDAH
NRP 05211640000045

Supervisor
Nur Aini Rakhmawati, S.Kom., M.Sc. Eng., Ph.D

INFORMATION SYSTEM DEPARTMENT
Faculty of Intelligent Electrical and Informatics Technology
Sepuluh Nopember Institute of Technology
Surabaya 2020

LEMBAR PENGESAHAN

Analisis Jejaring Sosial Terhadap Calon Anggota Legislatif Dalam Pemilihan Umum 2019 Berdasarkan Data Komisi Pemilihan Umum (KPU)

TUGAS AKHIR

Disusun Untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer (S.Kom)
pada
Departemen Sistem Informasi
Fakultas Teknologi Elektro dan Informatika Cerdas (ELECTICS)
Institut Teknologi Sepuluh Nopember

Oleh

Karima Mufidah
05211640000045

Surabaya, 15 Agustus 2020

Kepala Departemen Sistem Informasi

Dr. Mudjahidin, ST., MT.
NIP. 197010102003121001

LEMBAR PERSETUJUAN

ANALISIS JEJARING SOSIAL TERHADAP CALON ANGGOTA LEGISLATIF DALAM PEMILIHAN UMUM 2019 BERDASARKAN DATA KOMISI PEMILIHAN UMUM (KPU)

TUGAS AKHIR

Disusun Untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Departemen Sistem Informasi
Fakultas Teknologi Elektro dan Informatika Cerdas
Institut Teknologi Sepuluh Nopember
Oleh :

KARIMA MUFIDAH
NRP. 05211640000045

Disetujui Tim Penguji :Tanggal Ujian : 10 Juli 2020
Periode Wisuda : September 2020

Nur Aini R., S.Kom., M.Sc. Eng., Ph.D **(Pembimbing I)**

Radityo Prasetianto W., S.Kom, M.Kom **(Penguji I)**

Faizal Johan Atletiko, S.Kom., M.T **(Penguji II)**

ANALISIS JEJARING SOSIAL TERHADAP CALON ANGGOTA LEGISLATIF DALAM PEMILIHAN UMUM 2019 BERDASARKAN DATA KOMISI PEMILIHAN UMUM (KPU)

Nama	**:**	**Karima Mufidah**
NRP	**:**	**05211640000045**
Departemen	**:**	**Sistem Informasi FTEIC - ITS**
Pembimbing 1	**:**	**Nur Aini Rakhmawati, S.Kom., M.Sc.Eng., Ph.D.**

ABSTRAK

Pada tahun 2019, Indonesia menggelar pemilu serentak untuk memilih Presiden dan Wakil Presiden, DPR, DPRD Provinsi, DPD, dan DPRD Kabupaten/Kota. Komisi Pemilihan Umum (KPU) telah melakukan seleksi pada calon legislatif (caleg) yang mendaftar pemilu sebelum dilakukan pemungutan suara dimana hasil seleksi tersebut berupa Daftar Calon Tetap (DCT) yang akan dimasukkan ke dalam surat suara. Dari ribuan caleg yang ada dalam DCT, beberapa dari mereka kemungkinan memiliki hubungan, baik itu kerabat, saudara, maupun pasangan. Hal ini dapat diketahui dari data diri caleg pada website KPU yang didapatkan dengan menggunakan web crawler. Untuk mengetahui pola hubungan antar caleg, penelitian ini menggunakan metode Social Network Analysis (SNA), yaitu Teknik untuk memetakan hubungan antar individu pada jejaring sosialnya. Algoritma SNA yang digunakan pada penelitian ini adalah algoritma centrality dan algoritma community detection. Algoritma centrality berfungsi untuk menentukan pengaruh individu terhadap jejaring sosialnya, sedangkan algoritma community detection berfungsi untuk mendeteksi kelompok pada jejaring social berdasarkan perilaku individunya. Sebelum dilakukan proses perhitungan algoritma SNA, dataset harus dilakukan string matching terlebih dahulu untuk mengetahui kesamaan alamat yang

dimiliki oleh masing-masing caleg. Hasil pemetaan hubungan caleg divisualisasikan menggunakan D3js dalam bentuk force directed graph pada sebuah website statis. Berdasarkan perhitungan algoritma centrality, nilai nilai degree centrality dan pagerank tertinggi dimiliki oleh caleg-caleg yang memiliki hubungan kerabat paling banyak, nilai tertinggi pada betweenness centrality dimiliki oleh caleg bernama Ardherisa Marliza, dan untuk nilai tertinggi pada closeness centrality dimiliki oleh caleg bernama Hendarsam Marantoko. Sedangkan untuk hasil algoritma community detection ditemukan bahwa terdapat 60 kelompok yang terdiri dari 2-4 caleg dan /atau anggota yang memiliki hubungan kerabat. Hal ini mengindikasikan bahwa terdapat 60 keluarga yang mengikuti Pemilu 2019, baik dari partai pengusung yang sama maupun tidak. Dari hasil algoritma community detection dapat diketahui bahwa caleg yang memiliki relasi yang banyak hanya memiliki kemungkinan 9,375% untuk menang dalam pemilu.

Kata Kunci : Calon Legislatif, Graph Database, Neo4j, Pemilihan Umum 2019, Social Network Analysis.

SOCIAL NETWORK ANALYSIS ABOUT LEGISLATIVE CANDIDATES IN GENERAL ELECTION 2019 BASED ON KOMISI PEMILIHAN UMUM (KPU) DATA

Nama	**:**	**Karima Mufidah**
NRP	**:**	**05211640000045**
Departemen	**:**	**Sistem Informasi FTEIC - ITS**
Pembimbing 1	**:**	**Nur Aini Rakhmawati, S.Kom., M.Sc.Eng., Ph.D.**

ABSTRACT

In 2019, Indonesia has held simultaneous elections to elect the President and Vice President, the House of Representatives, and the Regional Representative Council. The General Election Commission (KPU) has conducted a selection of legislative candidates who registered for the election before the vote where the results of the selection will be in the form of a Permanent Candidate List (DCT) which will be included in the ballot papers. Through the thousands of candidates in the DCT, some of them are likely to have a relationship, be it a relative or spouse. This can be known from the personal data of candidates on the KPU website obtained by using a web crawler. To find out the pattern of relationships between candidates, this study uses the Social Network Analysis (SNA) method, which is a technique for mapping relationships between individuals on their social networks. SNA algorithm that used in this research is centrality algorithm and community detection algorithm. The centrality algorithm functions to determine the effect of individuals on their social networks, while the community detection algorithm functions to detect groups on social networks based on their individual behavior. Before the SNA algorithm is calculated, the dataset must first be done string matching to find out the similarity of addresses owned by each candidate. The results of mapping the relationship of candidates are visualized using D3js in the form of force

directed graphs on a static website. Based on the calculation of the centrality algorithm, the highest value of degree centrality and pagerank are owned by the candidates who have the most kin relations, the highest value in the betweenness centrality is owned by the candidate named Ardherisa Marliza, and for the highest value in the closeness centrality is owned by the candidate named Hendarsam Marantoko. As for the results of the community detection algorithm, it was found that there were 60 groups consisting of 2-4 candidates and/or members who had relatives. This indicates that there were 60 families participating in the 2019 Election, both from the same supporting party or not. From the results of the community detection algorithm it can be seen that candidates who have a lot of relationships only have a 9.375% chance of winning the election.

Keywords : 2019 General Election, Graph Database, Legislative Candidates, Neo4j, Social Network Analysis

SURAT PERNYATAAN BEBAS PLAGIARISME

Saya yang bertandatangan di bawah ini:

Nama	: Karima Mufidah
NRP	: 05211640000045
Tempat/Tanggal lahir	: Sidoarjo, 01 Mei 1999
Fakultas/Departemen	: FTEIC / Sistem Informasi
Nomor Telp/Hp/email	: karimamufidah@gmail.com

Dengan ini menyatakan dengan sesungguhnya bahwa penelitian/makalah/tugas akhir saya yang berjudul

ANALISIS JEJARING SOSIAL TERHADAP CALON ANGGOTA LEGISLATIF DALAM PEMILIHAN UMUM 2019 BERDASARKAN DATA KOMISI PEMILIHAN UMUM (KPU)

Bebas Dari Plagiarisme Dan Bukan Hasil Karya Orang Lain.

Apabila dikemudian hari ditemukan seluruh atau sebagian penelitian/makalah/tugas akhir tersebut terdapat indikasi plagiarisme, maka saya bersedia menerima sanksi sesuai peraturan dan ketentuan yang berlaku.

Demikian surat pernyataan ini saya buat dengan sesungguhnya dan untuk dipergunakan sebagaimana mestinya.

Surabaya, 14 Agustus 2020

Karima Mufidah
NRP.05211640000045

Halaman ini sengaja dikosongkan

KATA PENGANTAR

Puji syukur kepada Allah SWT yang Maha Pengasih dan Maha Penyayang karena berkat izin dan rahmat-Nya penulis dapat menyelesaikan penelitian pada tugas akhir ini dengan judul "Analisis Jejaring Sosial Terhadap Calon Anggota Legislatif Dalam Pemilihan Umum 2019 Berdasarkan Data Komisi Pemilihan Umum (KPU)" sebagai salah satu syarat kelulusan pada Departemen Sistem Informasi, Fakultas Teknologi Elektro dan Informatika Cerdas, Institut Teknologi Sepuluh Nopember Surabaya. Dalam menyelesaian Tugas Akhir ini, penulis diiringi oleh pihak-pihak yang senantiasa mendukung dan membimbing sehingga penelitian ini berlangsung dengan lancar. Secara khusus penulis mengucapkan terima kasih kepada:

1. Kedua orang tua penulis yaitu Bapak Budi Utomo dan Ibu Sutrami serta kakak penulis yaitu Naufal Hilmi Utomo yang selalu memberikan doa, dukungan dan semangat untuk menyelesaikan studi S1 dengan sebaik-baiknya.
2. Bapak Dr. Mudjahidin, S.T., M.T. selaku Ketua Departemen Sistem Informasi ITS Surabaya serta seluruh dosen pengajar beserta staf dan karyawan di Departemen Sistem Informasi, FTEIC ITS Surabaya selama penulis menjalani perkuliahan.
3. Ibu Nur Aini Rakhmawati, S.Kom., M.Sc., Eng. Ph. D selaku dosen pembimbing selama pengerjaan tugas akhir yang senantiasa meluangkan waktu, tenaga dan ilmunya serta selalu memotivasi untuk menyelesaikan tugas akhir ini.
4. Teman-teman SCI 2014 yang menemani penulis dari awal SMA hingga saat ini dan senantiasa tiada henti-hentinya selalu membahas tentang tugas akhir sehingga memicu penulis untuk segera menyelesaikannya.

5. Teman dekat saya yaitu Dinda, Gusti, Refing, dan Gushan yang selalu menemani dan mendukung saya selama ini.
6. Teman-teman Artemis yang telah menemani penulis sejak masa maba hingga sekarang.
7. Kepada seluruh teman-teman di Lab Akuisisi Data dan Diseminasi Informasi (ADDI) yang telah bersama-sama berjuang mengerjakan tugas akhir, terutama teman-teman seperbimbingan Bu Iin yaitu Wulan dan Rifqy.
8. Berbagai pihak lainnya yang tidak bisa disebutkan satu persatu yang turut menyukseskan dan mendukung penulis menyelesaikan tugas akhir.

Penyusunan laporan tugas akhir ini masih jauh dari kata sempurna, sehingga penulis menerima segala kritik dan saran yang dapat membangun serta perbaikan untuk menjadi lebih baik lagi di masa yang akan datang. Semoga tugas akhir ini dapat bermanfaat bagi pembaca.

Surabaya, 03 Juli 2020

Penulis

DAFTAR ISI

Halaman ini sengaja dikosongkan

DAFTAR GAMBAR

DAFTAR TABEL

Halaman ini sengaja dikosongkan

DAFTAR KODE

BAB I
PENDAHULUAN

Bab pendahuluan menjelaskan tentang identifikasi permasalahan yang dibahas dalam tugas akhir. Identifikasi tersebut meliputi latar belakang masalah, perumusan masalah, batasan masalah, tujuan dan manfaat penelitian, serta relevansi tugas akhir dengan bidang keilmuan system informasi. Berdasarkan uraian pada bab ini, harapannya gambaran umum permasalahan dan solusi masalah pada tugas akhir dapat dipahami.

1.1 Latar Belakang

Tahun 2019 menjadi tahun diselenggarakannya segala jenis Pemilihan Umum (Pemilu) di Indonesia, atau dapat disebut dengan istilah pesta demokrasi. Pemilu 2019 meliputi pemilihan Presiden dan Wakil Presiden, DPR, DPRD Provinsi, DPD, dan DPRD Kabupaten/Kota. Sejak pertengahan tahun 2018, pendaftaran calon legislatif (caleg) telah dibuka. Situs berita Detiknews menyebutkan bahwa antusiasme masyarakat Indonesia sebagai partisipan pemilih dalam pemilu kali ini mencapai 81%, angka tersebut mengalami peningkatan sebesar 10% dibandingkan dengan pemilu 2014. Komisi Pemilihan Umum (KPU) telah menetapkan bahwa daftar calon tetap (DCT) anggota DPR RI berjumlah 7.968 orang untuk merebutkan 575 kursi anggota dewan pada Pemilu 2019 [1]. Jumlah ini meningkat sebanyak 17% dari Pemilu 2014 yang mana pada waktu itu DCT anggota DPR berjumlah 6.607 orang. Dari ribuan orang yang berada dalam DCT, tidak dipungkiri bahwa beberapa dari mereka kemungkinan memiliki kedekatan atau hubungan yang belum tentu diketahui oleh calon pemilih.

Kedekatan hubungan caleg satu dengan caleg yang lain dapat diketahui dengan metode Social Network Analysis (SNA). SNA merupakan proses menggambarkan hubungan interaksi antar individu dalam sebuah lingkup interaksi sosial [2]. Penelitian sebelumnya [3] [4] [5] [6] telah membuktikan bahwa dengan metode SNA dapat diketahui pola hubungan yang dimiliki oleh antar politisi melalui berbagai jejaring sosial, seperti Twitter, blog, situs web, Facebook, dan jejaring sosial lainnya.

Begitu juga dengan penelitian ini, SNA dapat membantu memetakan atau membuat pola hubungan antar individu berdasarkan data Pemilu 2019 pada *website* KPU. Oleh karena itu, metode ini cocok dilakukan untuk menganalisis kedekatan antar caleg pada Pemilu 2019. Penelitian ini dikhususkan untuk menganalisa hubungan sosial yang terbentuk dari anggota DPR pada Pemilu 2019.

Dengan data DCT dan calon terpilih atau pemenang yang ada di *website* KPU dapat apakah ada kaitannya hubungan yang dimiliki caleg terhadap pemenangan dalam pemilu melalui algoritma SNA *centrality* dan *community detection* [7]. Hasil dari algoritma SNA tersebut akan divisualisasikan dalam bentuk graf untuk mempermudah pembaca dalam memahaminya.

Dengan penelitian ini diharapkan dapat memberikan wawasan dan pengetahuan baru kepada para calon pemilih untuk menetapkan pilihan dalam pemilu selanjutnya.

1.2 Rumusan Masalah

Berdasarkan latar belakang masalah di atas, berikut ini adalah beberapa poin perumusan masalah yang akan diangkat pada tugas akhir ini.

1. Bagaimana mengakuisisi data para caleg yang mengikuti Pemilu 2019 yang ada pada *website*?

2. Bagaimana menerapkan beberapa algoritma SNA untuk memetakan pola hubungan antar caleg berdasarkan data pada *website* KPU?
3. Bagaimana cara memvisualisasikan pola hubungan antar caleg berdasarkan hasil perhitungan SNA?

1.3 Batasan Masalah

Berdasarkan rumusan masalah yang telah disebutkan, adapun batasan permasalahan dalam mengerjakan tugas akhir ini adalah sebagai berkut.

1. Studi kasus yang digunakan dalam penelitian ini hanya meliputi caleg DPR yang mengikuti Pemilu 2019.
2. Caleg pada studi kasus ini merupakan caleg yang identitasnya tertera pada DCT yang diumumkan oleh KPU.

1.4 Tujuan

Tujuan dari penelitian ini adalah menganalisis hubungan yang dimiliki oleh caleg yang mengikuti Pemilu 2019 berdasarkan hasil dari perhitungan algoritma SNA dimana hasilnya akan divisualisasikan dalam sebuah aplikasi untuk mempermudah dalam menganalisis hubungan. Dengan aplikasi ini dapat membantu untuk menganalisis dampak kedekatan atau hubungan yang dimiliki antar caleg pada hasil pemungutan suara dalam Pemilu 2019.

1.5 Manfaat

Manfaat yang diperoleh dengan melakukan penelitian ini adalah sebagai berikut.

1.5.1 Bagi Penulis

Penelitian ini memberikan wawasan baru bagi penulis mengenai metode SNA yang digunakan dan hasil yang didapatkan.

1.5.2 Bagi Masyarakat

Penelitian ini memberitahu kedekatan atau hubungan yang dimiliki oleh para caleg yang mengikuti Pemilu 2019.

1.5.3 Bagi Partai Politik

Penelitian ini membantu partai politik untuk menganalisis terkait dampak para caleg yang memiliki kedekatan atau hubungan dalam perolehan suara pada Pemilu 2019. Sehingga diharapkan parpol dapat mempertimbangkan caleg yang akan diusung pada pemilihan periode selanjutnya.

1.6 Relevansi

Tugas akhir ini relevan dengan salah satu bidang minat di Departemen Sistem Informasi yaitu Laboratorium Akuisisi Data dan Diseminasi Informasi (ADDI) dimana pada laboratorium ADDI ini memiliki tiga elemen penting yang menjadi topik penelitian utama, yaitu akuisisi data, *processing paradigm*, dan diseminasi informasi. Dalam tugas akhir yang dikerjakan oleh penulis ini mengambil topik analisis jejaring sosial terhadap data caleg yang terdapat di *website* KPU menggunakan algoritma SNA, yang mana masuk dalam kategori akuisisi data dan

diseminasi informasi pada topik penelitian laboratorium ADDI. Selain itu, tugas akhir yang diusulkan berkaitan dengan beberapa mata kuliah yang ada pada Departemen Sistem Informasi. Yang pertama adalah mata kuliah Visualisasi Informasi. Hal tersebut dikarenakan dalam penelitian ini menggunakan metode D3js untuk memvisualisasikan hasil algoritma SNA, yang mana dipelajari pada mata kuliah Visualisasi Informasi. Yang kedua adalah mata kuliah Teknologi Basis Data. Hal tersebut dikarenakan dalam penelitian ini mengambil data pada situs resmi KPU menggunakan sistem *crawling* dan juga tentang pengelolaan basis data, yang mana dipelajari pada mata kuliah Teknologi Basis Data. Yang terakhir adalah mata kuliah Analisis dan Desain Perangkat Lunak. Hal tersebut dikarenakan dalam penelitian ini visualisasi yang ditampilkan dalam bentuk *webpage*.

Halaman ini sengaja dikosongkan

BAB II
TINJAUAN PUSTAKA

Bab tinjauan pustaka menjelaskan tentang penelitian sebelumnya maupun studi literatur yang berkaitan dengan tugas akhir untuk dijadikan sebagai acuan selama pengerjaan dan landasan teori yang dapat membantu pemahaman selama pengerjaan tugas akhir.

2.1. Penelitian Terkait

Pada subbab ini memaparkan tentang penelitian sebelumnya yang berkaitan dengan tugas akhir untuk digunakan sebagai referensi dalam pengerjaan tugas akhir.

Tabel 2. 1 Studi Literatur

Penelitian 1	
Judul penelitian	**Rancang Bangun Aplikasi Teenstagram untuk *Social Network Analysis* dengan Menggunakan *Sociomatrix* dari Akun Media Sosial Instagram (Studi Kasus: Siswa SMA di Surabaya)** [8]
Penulis	Rheindra Alfarhizi
Ringkasan	Penelitian ini berfokus untuk merancang aplikasi bernama Teenstagram. Tujuannya adalah untuk menciptakan sebuah *platform* yang mampu memvisualisasikan SNA dari akun media sosial Instagram siswa SMA di Surabaya. Dengan adanya *platform* tersebut dapat diketahui kedekatan dari pertemanan yang terbentuk berdasarkan akun media sosial Instagram. Data yang digunakan pada penelitian ini merupakan data *following* dan

	followers di Instagram dalam bentuk JSON yang diperoleh dengan menggunakan *web crawler*. Penelitian ini menggunakan metode *sociomatrix* sebagai hasil represent-tasi data SNA yang divisualisasikan dalam bentuk graf *sociogram*.
Keterkaitan penelitian	Penelitian tersebut memvisualisasikan SNA berdasarkan akun media sosial Instagram dan menggunakan *web crawler* untuk mendapatkan data dalam bentuk JSON, sementara tugas akhir yang diusulkan adalah memvisualisasikan SNA berdasar-kan data calon legislatif 2019 pada situs resmi Komisi Pemilihan Umum (KPU) yang didapatkan dengan menggunakan *web crawler* dalam bentuk JSON.
Penelitian 2	
Judul penelitian	***Strategies Affecting Twitter-based Networking Pattern of South Korean Politicians: Social Network Analysis and Exponential Random Graph Model*** [4]
Penulis	Ho Young Yoon dan Han Woo Park
Ringkasan	Penelitian ini bertujuan untuk mengetahui pola hubungan para politisi Korea Selatan berdasarkan *following* dan *mention* pada media sosial Twitter dengan menggunakan SNA. Data yang digunakan dalam penelitian ini berasal dari informasi akun media sosial Twitter para politisi, seperti *following*, *followers*, *mentions*, jumlah *tweets* dan tanggal pembuatan akun Twitter menggunakan API. Hasil dari penelitian ini divisualisasikan dalam bentuk graf untuk

	menggambarkan pola hubungan antar politisi Korea Selatan. Dengan penelitian ini diharapkan agar dapat memberikan pemahaman yang lebih baik tentang bagaimana para politisi memberikan dukungan politik kepada sesamanya dan bagaimana pesan mereka dapat diterima oleh grup politik lain.
Keterkaitan penelitian	Penelitian tersebut mengimplementasikan SNA untuk mengetahui pola hubungan antar politisi Korea Selatan dengan data yang berasal dari Twitter dan hasilnya divisualisasikan dalam bentuk graf, sementara tugas akhir yang diusulkan adalah mengimplementasikan SNA untuk mengetahui pola hubungan antar anggota legislatif dengan data yang berasal dari situs resmi KPU dan divisualisasikan dalam bentuk graf.
Penelitian 3	
Judul penelitian	***The Social Network Analysis of Switzerland Football Team on FIFA World Cup 2014*** [9]
Penulis	Filipe Manuel Clemente, Fernando Manuel Lourenço Martins, Dimitris Kalamaras, Joana Oliveira, Patrícia Oliveira, dan Rui Sousa Mendes
Ringkasan	Penelitian ini bertujuan untuk menganalisis pendekatan antar anggota tim nasional (timnas) sepak bola Switzerland dalam turnamen FIFA World Cup 2014 dengan menggunakan SNA. Data yang digunakan dalam penelitian ini adalah empat per-

	tandingan resmi Switzerland dalam turnamen FIFA World Cup 2014. Data tersebut berupa empat *network graph* yang dihasilkan berdasarkan interaksi rekan satu tim. Pada *network graph* yang dianalisis dalam penelitian ini menunjukkan seorang pemain sebagai indikator sebuah *node* dan *passing* bola antar pemain sebagai indikator *relationship* (koneksi/hubungan). Hasil penelitian ini dapat membantu membuat dan mengembangkan pendekatan baru untuk mempelajari interaksi rekan satu tim.
Keterkaitan penelitian	Penelitian tersebut mengimplementasikan SNA untuk menganalisis pendekatan antar anggota timnas sepak bola Switzerland menggunakan *network graph* berdasarkan pertandingan dalam turnamen FIFA World Cup 2014, sementara tugas akhir yang diusulkan adalah mengimplementasikan SNA untuk menganalisis hubungan antar anggota legislatif menggunakan *graph database* berdasarkan data dari situs resmi KPU.
Penelitian 4	
Judul penelitian	***Mapping Online Social Networks of Korean Politician*** [3]
Penulis	Chien-leng Hsu dan Han Woo Park
Ringkasan	Penelitian ini bertujuan untuk mem-bandingkan jaringan hubungan yang dimiliki anggota Majelis Nasional Korea Selatan pada *platform* Twitter, blog, dan situs resmi Majelis Nasional menggunakan SNA. Anggota Manjelis Nasional dibagi

	menjadi dua bagian, yaitu partai konservatif dan partai liberal. Hasil dari penelitian ini menunjukkan bahwa partai konservatif lebih sering menggunakan situs web dan blog, sedangkan partai liberal lebih sering menggunakan Twitter untuk berkomunikasi dengan politisi lain maupun masyarakat.
Keterkaitan penelitian	Penelitian tersebut mengimplementasikan SNA untuk membandingkan *relationship network* para politisi Korea Selatan berdasarkan data dari *website* Majelis Nasional, blog, dan Twitter dan hasilnya divisualisasikan menggunakan graf, sementara tugas akhir yang diusulkan adalah mengimplementasikan SNA untuk mengetahui hubungan antar anggota legislatif pada Pemilu 2019 berdasarkan data pada *website* KPU dan hasilnya divisualisasikan menggunakan graf.
Penelitian 5	
Judul penelitian	**Analisis Jejaring Sosial Menggunakan *Social Network Analysis* untuk Membantu *Social CRM* bagi UMKM di Cimahi [2]**
Penulis	Asep Id Hadiana dan Wina Witanti
Ringkasan	Penelitian ini bertujuan untuk membantu UMKM di Cimahi me-ngetahui pola dan karakteristik dari para pelanggan melalui media sosial dengan menggunakan SNA. UMKM dapat menjalankan strategi bisnis yang melibatkan pelanggan melalui media sosial atau disebut dengan *Social CRM*. Data yang digunakan dalam penelitian ini

	berasal dari media sosial Twitter berupa *tweet* dengan *sample* kata kunci tertentu yang diambil dan diolah menggunakan *NodeXL Basic*. Pola hubungan interaksi antar aktor dan kata kunci dalam penelitian ini divisu-alisasikan menggunakan graf.
Keterkaitan penelitian	Penelitian tersebut mengimplementasikan SNA untuk mengetahui pola dan karakteristik para pelanggan melalui media sosial Twitter dan divisualisasikan menggunakan graf, sementara tugas akhir yang diusulkan adalah mengimplemen-tasikan SNA untuk mengetahui hubungan yang dimiliki antar anggota legislatif melalui data dari situs resmi KPU dan divisualisasikan menggunakan graf.

Berdasarkan literatur di atas, berikut merupakan tabel keterkaitan penelitian pada literatur dengan tugas akhir yang diusulkan.

Tabel 2. 2 Keterkaitan Penelitian

Literatur	Domain	Metrik	Dataset
Rheindra Alfarhizi [8]	Sosial, pelajar	*Degree centrality*	Data *following* dan *followers* pada akun Instagram.
Ho Young Yoon dan Han Woo Park [4]	Politik	1. *Indegree centrality* 2. *Clustering coefficient*	Data akun Twitter politisi Korea Selatan

		3. *Network density*	(*following, followers, mentions,* dll).
F. M. Clemente, F. M. L. Martins, D. Kalamaras, P. Oliveira dan R. S. Mendes [9]	Olahraga, sepak bola	1. *Total links* 2. *Network density* 3. *Degree centrality* 4. *Degree pestige*	Data interaksi (*passing, serangan*) antar pemain dalam empat pertandingan FIFA World Cup 2014
Chien-leng Hsu dan Han Woo Park [3]	Politik	1. *Degree centrality* 2. *Network density*	Data *official homepage*, akun Twitter, dan blog dari anggota Majelis Nasional
Asep Id Hadiana dan Wina Witanti [2]	Ekonomi, sosial	1. *Degree centrality* 2. *Betweennes s centrality* 3. *Closeness centrality*	Data *tweet* dari Twitter dengan *sample* kata kunci ‘bandrek’ dan ‘bajigur’.

2.2. Dasar Teori

Penelitian kali ini menggunakan beberapa dasar teori sebagai acuan. Subbab ini memaparkan tentang teori-teori yang digunakan selama pengerjaan tugas akhir sehingga lebih jelas dan sistematis.

2.2.1 Komisi Pemilihan Umum (KPU)

Menurut Undang-Undang No. 7 Tahun 2017, Komisi Pemilihan Umum yang selanjutnya disingkat KPU adalah Lembaga Penyelenggara Pemilu yang bersifat nasional, tetap, dan mandiri dalam melaksanakan pemilu [10]. KPU memiliki situs resmi yang berfungsi untuk memberikan informasi kepada masyarakat Indonesia tentang berita terkini dan hal-hal terkait pemilihan umum. Selain itu, situs resmi KPU menyediakan data para calon legislatif (caleg) pada pemilihan umum 2019. Calon pemilih dapat melihat daftar caleg berdasarkan tingkat daerah pilihan, mulai dari DPR, DPRD Provinsi, DPRD Kabupaten/Kota, hingga DPD. Data caleg pada situs resmi KPU menampilkan foto dan detail biodata, termasuk riwayat hidup caleg.

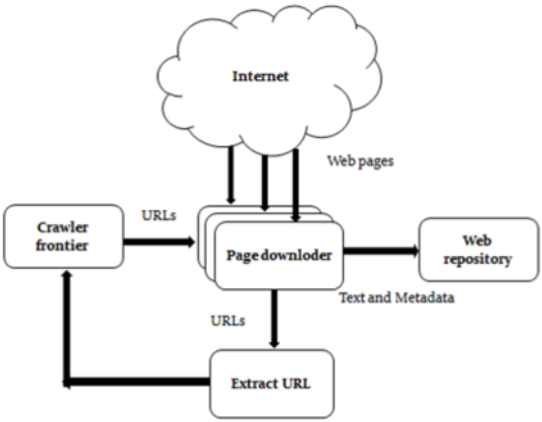
2.2.2 Pemilihan Umum (Pemilu)

Menurut Undang-Undang No.7 Tahun 2017, Pemilihan Umum yang selanjutnya disebut Pemilu adalah sarana kedaulatan rakyat untuk memilih anggota Dewan Perwakilan Rakyat, anggota Dewan Perwakilan Daerah, Presiden dan Wakil Presiden, dan untuk memilih anggota Dewan Perwakilan Rakyat Daerah, yang dilaksanakan secara langsung, umum, bebas, rahasia, jujur, dan adil dalam Negara

Kesatuan Republik Indonesia berdasarkan Pancasila dan Undang-Undang Dasar Negara Republik Indonesia Tahun 1945 [10]. Pada pemilu 2019, terdapat ratusan ribu orang [11] dari seluruh wilayah di Indonesia yang mencalonkan diri menjadi anggota legislatif. Pemilu 2019 dilaksanakan secara serentak di Indonesia pada 17 April 2019 untuk memilih sebanyak 575 anggota DPR RI, 136 anggota DPD RI, 2.207 anggota DPRD Provinsi, dan 17.610 anggota DPRD Kabupaten/Kota [1] untuk periode 2019-2024.

2.2.3 *Web Crawler*

Web crawler, atau dikenal juga dengan *robots*, *spiders*, *worms*, *walkers*, dan *wanderers*, merupakan sebuah perangkat lunak atau *script* yang diprogram untuk menjelajah *World Wide Web* (WWW) secara sistematis dan otomatis. Sistem *web crawling* melakukan pengambilan data dengan cara mengikuti struktur *web* yang saling terhubung sehingga dapat memungkinkan untuk mengambil data dari satu halaman *web* ke halaman *web* yang lain. Penggunaan *web crawler* memiliki tujuan untuk mengumpulkan data yang nantinya akan digunakan dalam pembuatan indeks untuk mempermudah dan mempersingkat proses pencarian.



Gambar 2. 1 Arsitektur *Web Crawler* [12]

Gambar 2.1 menunjukkan arsitektur dari sebuah *web crawler*. Arsitektur tersebut memiliki tiga komponen utama, yaitu *crawler frontier* yang menyimpan daftar URL untuk dikunjungi, kemudian melalui proses *page downloader* untuk mengunduh halaman dari WWW dan yang terakhir diterima oleh *web repository* untuk disimpan ke dalam basis data [12].

2.2.4 *Social Network Analysis* (SNA)

Social Network Analysis (SNA) merupakan proses menggambarkan hubungan interaksi antara aktor dengan aktor lainnya dalam sebuah interaksi sosial [2]. SNA terdiri dari *nodes* (aktor) dan *links* (hubungan). *Nodes* merepresentasikan aktor yang berada dalam sebuah jejaring atau *network*, sedangkan *links* merepresentasikan hubungan yang terjalin antar sesama aktor. SNA memiliki dua level dalam menganalisis sebuah jejaring, yaitu (1)

individual level dan (2) *network level*. SNA pada *individual level* menghasilkan laporan lokasi individu dalam jaringan yang terhubung pada pusat, yang disebut dengan *individual's centrality* [13]. Berikut merupakan algoritma sentralitas pada SNA yang akan digunakan dalam penelitian ini [7].

1. *Degree centrality*, yaitu jumlah koneksi yang dimiliki oleh sebuah *node* baik koneksi yang masuk (*inDegree)* maupun koneksi yang keluar (*outDegree)*.
2. *Closeness centrality*, yaitu jarak rata-rata antara *node* dengan semua *node* yang lain di dalam sebuah jaringan. Ukuran ini menggambarkan kedekatan satu *node* dengan *node* lain. Semakin dekat maka semakin terhubung individu tersebut dengan lainnya. Berikut ini merupakan rumus untuk menghitung *closeness centrality*.

$$C(u) = \frac{n-1}{\sum_{v=1}^{n-1} d(u,v)} \quad (1)$$

   Dimana:
   - *u* adalah sebuah *node.*
   - *n* adalah jumlah *node* dalam komponen yang sama (subgraph atau grup) seperti *u*.
   - *d(u, v)* adalah jarak jalur terpendek antara *node v* dan *u* lainnya.
3. *Betweenness centrality*, yaitu algoritma yang menghitung jalur terpendek antara setiap pasang *node* dalam graf yang terhubung. Setiap *node* menerima nilai berdasarkan jumlah jalur terpendek yang melewati *node*. Semakin pendek jalur yang dilalui sebuah *node* maka semakin

tinggi nilainya. Berikut ini merupakan rumus untuk menghitung *betweeness centrality*.

$$B(u) = \sum_{s \neq u \neq t} \frac{p(u)}{p} \qquad (2)$$

Dimana:

- *u* adalah sebuah *node*.
- *p* adalah jumlah total jalur terpendek antara *node s* dan *t*.

4. *PageRank*, yaitu algoritma yang mengukur jumlah dan kualitas hubungan yang masuk ke sebuah *node* untuk menentukan perkiraan seberapa penting *node* tersebut. *Node* yang lebih mendominasi terhadap jaringan dianggap memiliki hubungan masuk lebih banyak dari *node* berpengaruh lainnya. *PageRank* didefinisikan oleh makalah Google sebagai berikut.

$$PR(u) = (1-d) + d\left(\frac{PR(T1)}{C(T1)} + \ldots + \frac{PR(Tn)}{C(Tn)}\right) \qquad (3)$$

Dimana:

- Asumsikan *page u* memiliki sitasi dari *page T1* sampai *Tn*
- *d* adalah faktor redaman yang nilainya antara 0 dan 1. Biasanya ditentukan 0,85. *d* dapat dianggap sebagai probabilitas pengguna akan terus mengklik.
- *1 – d* adalah probabilitas bahwa sebuah *node* tercapai secara langsung tanpa mengikuti relasi apapun.
- *C(Tn)* didefinisikan sebagai *outDegree* dari *node T*.

Selain algoritma di atas, berikut ini terdapat algoritma *community detection* yang berfungsi untuk mengidentifikasi kelompok berdasarkan perilaku dan preferensi dari aktor (*node*).

1. *Triangle count and clustering coefficient*, yaitu algoritma yang mengukur seberapa banyak *node* yang membentuk segitiga dan sejauh mana *node* cenderung mengelompok bersama. Terdapat dua tipe *clustering coefficient*, yaitu:
   a. *Local clustering coefficient*, merupakan kemungkinan bahwa tetangganya juga terhubung. Perhitungan nilai ini melibatkan penghitungan segitiga.

$$CC(u) = \frac{2R_u}{k_u(k_u - 1)} \tag{4}$$

   Dimana:
   - *u* adalah sebuah *node*.
   - *R(u)* adalah jumlah hubungan melalui tetangga *u* (dapat diperoleh dengan menggunakan jumlah segitiga yang melewati *u*).
   - *k(u)* adalah *degree* dari *u*.

   b. *Global clustering coefficient*, merupakan jumlah yang dinormalisasi dari *local clustering coefficient*.
2. *Strongly connected components*, yaitu algoritma untuk menemukan kelompok dimana setiap *node* dapat dijangkau dari setiap *node* lain dalam kelompok yang sama mengikuti arah hubungan.
3. *Connected components*, yaitu algoritma untuk menemukan kelompok dimana setiap *node* dapat dijangkau dari setiap *node* lain dalam kelompok, terlepas dari arah hubungan.

4. *Label propagation*, yaitu algoritma yang mengumpulkan kluster dengan menyebarkan label berdasarkan mayoritas lingkungan.
5. *Louvain modularity*, yaitu algoritma yang memaksimalkan akurasi pengelompokan yang telah diasumsikan sebelumnya dengan membandingkan bobot dan kepadatan hubungan dengan estimasi atau rata-rata yang ditentukan. *Louvain* mengkualifikasi seberapa baik sebuah *node* ditetapkan pada sebuah kelompok berdasarkan densitas hubungan. Kualifikasi tersebut ditentukan dengan nilai modularitas. Modularitas merupakan ukuran seberapa baik kelompok setelah dipartisi mejadi beberapa kelompok yang lebih kecil.

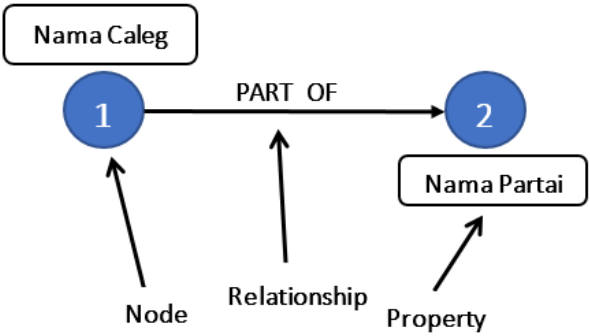
Nilai sentralitas dapat berbeda-beda pada masing-masing individu. Semakin besar nilai sentralitas individu maka semakin terpusat keberadaanya dalam jaringan. Beberapa penelitian telah membuktikan bahwa indeks SNA dapat membantu untuk memposisikan seorang individu dalam suatu jejaring agar mendapatkan suatu pencapaian [13].

SNA dapat digunakan untuk mengambil informasi hubungan interaksi dan letak interaksi antar aktor yang kemudian dapat direpresentasikan dalam bentuk graf. Representasi *social network* dalam bentuk graf dikarenakan graf merupakan tipe representasi yang paling fundamental [2].

2.2.5 *Graph Database*

Graph database merupakan basis data yang struktur datanya menggunakan *nodes* (aktor) dan *edges* (hubungan) untuk menampilkan dan menyimpan

data. Setiap *node* merepresentasikan aktor dan setiap *edge* merepresentasikan koneksi atau hubungan yang terjalin antar sesama aktor.



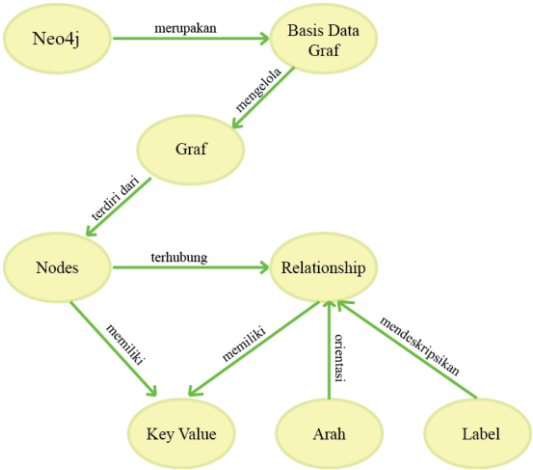
Gambar 2. 2 Komponen Graf

Gambar 2.2 menunjukkan komponen graf dan representasi visual bagaimana sebuah *node* terhubung satu sama lain. *Graph database* lebih ekspresif dan lebih sederhana daripada basis data relasional dan basis data NoSQL lainnya. *Graph database* memiliki spesialisasi dalam merepresentasikan hubungan di antara data yang berskala besar. Selain itu, *graph database* berguna untuk mengelola data yang terhubung (*linked data*) secara mendalam [14].

2.2.6 Neo4j

Neo4j merupakan basis data graf transaksional *open source* yang dirilis pada tahun 2007. Selain dengan menggunakan bahasa pemrograman Java dalam pengoperasiannya, Neo4j juga didukung oleh bahasa pemrograman lain seperti Python untuk membuat graf. Pada Neo4j, *node* (aktor) dan *relationship*

(relasi/hubungan) dapat berisi lebih dari satu properti. Neo4j merupakan basis data yang mengelola graf dan mengoptimalkan struktur graf alih-alih menggunakan tabel. Belakangan ini, Neo4j menjadi basis data graf yang populer.



Gambar 2. 3 Model Graf pada Neo4j

Gambar 2.3 menunjukkan model graf pada Neo4j atau disebut dengan "*property graph model*". Graf properti terdiri dari *nodes* yang dihubungkan dengan *relationship* (hubungan/relasi). Setiap *relationship* terdiri dari dua fitur utama, yaitu nama dan arah. Keduanya memberikan konteks semantik pada *nodes* yang terhubung dengan *relationship* [15].

2.2.7 *Data Driven Document* (D3js)

D3js merupakan *library* JavaScript yang berfungsi untuk memanipulasi dokumen berdasarkan data. D3js merupakan salah satu *tool* untuk menampilkan visualisasi data pada situs web dengan menggunakan HTML, SVG, dan CSS [16]. Pada

situs resmi D3js (https://d3js.org/) terdapat berbagai variasi visualisasi yang dapat diterapkan pada data beserta dengan contohnya.

2.2.8 *Cypher*

Cypher merupakan *graph query language* pada Neo4j yang memungkinkan pengguna untuk menyimpan dan mengambil data dari *graph database*. Sintaks *Cypher* menyediakan cara visual dan logis untuk mencocokkan pola *node* dan hubungan dalam graf. *Cypher* merupakan bahasa deklaratif yang terinspirasi dari SQL. Melalui *Cypher*, pengguna dapat membuat kueri yang ekspresif dan efisien untuk melakukan fungsional *create*, *read*, *update*, dan *delete* [17].

2.2.9 ParseHub

ParseHub merupakan salah satu alat atau media yang digunakan untuk mengumpulkan data dari sebuah *website*. ParseHub memiliki cara kerja yang cukup sederhana dan mudah. Sebelum memulai untuk mengumpulkan data (*scraping*) pada *website* yang diinginkan, pengguna dapat memilih elemen yang diperlukan (tanpa harus mengambil keseluruhan data) untuk diambil dengan cara mengarahkan pointer kemudian klik, kemudian ParseHub akan secara otomatis menerka data lain yang memiliki kesamaan dengan elemen yang dipilih. Cara ini dapat membantu untuk mempercepat pengambilan data karena tidak perlu klik satu per satu pada setiap elemen yang sama. Jika telah menentukan data yang ingin diambil (*extract*), langkah selanjutnya adalah menjalankan program (*run*) yang membutuhkan beberapa saat, tergantung banyaknya data yang

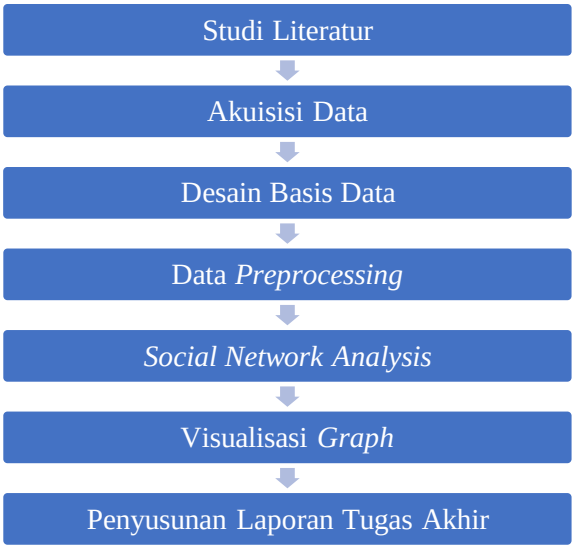
diambil. Setelah proses *running* selesai, data hasil *scraping* dapat diunduh dengan berbagai format, diantaranya adalah CSV/Excel, JSON, dan API [18].

BAB III
METODOLOGI

Bab metodologi menjelaskan tentang langkah-langkah yang dilakukan dalam melaksanakan penelitian, meliputi urutan pelaksanaan proses penelitian, metode yang digunakan dalam setiap proses penelitian, serta data dan peralatan yang digunakan.

3.1 Tahapan Pelaksaan Tugas Akhir

Pada sub bab ini menjelaskan tentang metodologi dalam pengerjaan tugas akhir. Metodologi ini dapat dilihat pada Gambar 3.1.



Gambar 3. 1 Metodologi Tugas Akhir

3.2 Studi Literatur

Proses studi literatur dilakukan untuk mencari materi-materi yang terkait dengan penelitian tugas akhir. Tujuannya adalah untuk dapat memahami konsep, metode, dan teknologi sesuai dengan bahasan sehingga dapat memberikan solusi mengenai permasalahan. Literatur yang digunakan dalam penelitian ini meliputi, pengambilan data menggunakan web crawler, implementasi SNA menggunakan algoritma degree centrality, closeness centrality, dan betweenness centrality dengan Neo4j, serta pembuatan visualisasi graph menggunakan D3js.

Luaran yang dihasilkan dari proses ini adalah mengetahui adanya kesenjangan pengetahuan dan pemahaman yang dibutuhkan berdasarkan literatur yang telah didapat.

3.3 Akuisisi Data

Proses akuisisi data dilakukan untuk mendapatkan informasi yang akan digunakan dalam penelitian ini. Proses akuisisi data dilakukan dengan beberapa tahapan, yaitu:

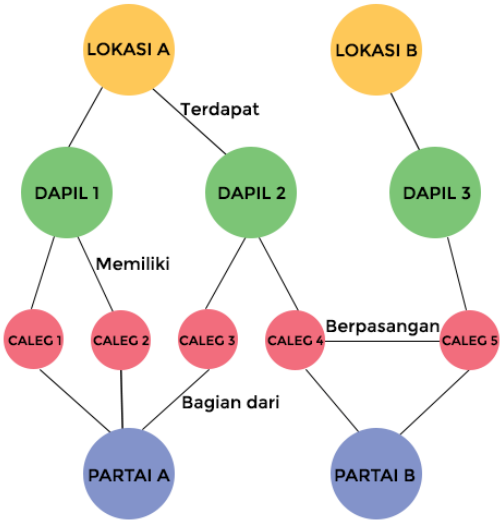
- Mengumpulkan informasi mengenai latar belakang caleg berdasarkan situs resmi KPU. Proses pengumpulan informasi dilakukan menggunakan *web crawler* untuk mengekstrak data dari *file* JSON pada *web* ke *file* CSV. Data yang didapat mencakup daerah pilihan (dapil), partai pengusung, nama, jenis kelamin, tahun lahir, agama, status pernikahan, nama pasangan, pendidikan, pekerjaan, motivasi menjadi anggota legislatif, dan target saat terpilih menjadi anggota legislatif.
- Melakukan validasi data yang telah di dapatkan. Hal ini dilakukan agar hasil penelitian relevan

dengan fakta yang ada. Validasi data dilakukan dengan cara memastikan semua *field* data terisi dan tidak menggunakan data yang terdapat *field* kosong.

Luaran dari proses akuisisi data adalah mendapatkan data yang dibutuhkan untuk melakukan penelitian dalam bentuk *file* JSON.

3.4 Desain Basis Data

Desain basis data merupakan tahapan untuk membuat desain untuk mendapatkan sistem basis data yang diperlukan sesuai dengan tujuan penelitian.



Gambar 3. 2 Desain *Graph Database*

Gambar 3.2 merupakan desain *graph database* yang akan digunakan dalam penelitian ini. Terdapat lokasi, dapil,

partai, dan caleg yang menjadi *node* serta beberapa relasi yang menggabungkan *node-node* tersebut pada *graph*.

3.5 Data *Preprocessing*

Data *preprocessing* merupakan tahap yang dilakukan untuk mengolah data yang telah didapatkan dari tahap akuisisi data. Beberapa tahapan yang dilakukan dalam data *preprocessing* pada penelitian ini antara lain:

1. Mengubah *file* JSON menjadi CSV

   Tujuan dari mengubah format *file* dari JSON ke CSV adalah untuk mempermudah dalam membaca *record* data. Selain itu, format CSV juga lebih mudah digunakan dalam proses *Social Network Analysis* (SNA) pada perangkat lunak Neo4j.

2. Menetapkan nama kolom pada *file* CSV

   Penetapan nama kolom pada *file* CSV sangat penting karena akan digunakan untuk proses SNA. Tujuannya adalah agar lebih mudah dalam memasukkan *query* pada perangkat lunak Neo4j. Nama kolom yang digunakan harus merepresentasikan isi *record* data pada kolom tersebut.

3. Menghapus *record* data (*row*) yang terdapat nilai *null*

   Dalam *file* tersebut terdapat banyak kolom dengan jenis data yang berbeda. Pada penelitian ini tidak menggunakan keseluruhan data yang ada di dalam *file* CSV, melainkan hanya beberapa kolom. Dari kolom-kolom yang terpilih untuk digunakan dalam SNA harus dipastikan nilainya tidak ada yang *null* (kosong) karena hal itu dapat mempengaruhi hasil dari analisis.

4. *String matching*

   String matching merupakan proses untuk mencari *string* yang sekiranya cocok dengan sebuah pola.

Tujuan penggunaan *string matching* pada penelitian ini adalah untuk mengetahui apakah nama yang ada di kolom pasangan caleg juga ada pada kolom nama caleg yang mana berarti pasangan tersebut sama-sama mendaftar caleg pada pemilu ini. Dengan begitu proses pada tahap SNA akan lebih mudah. *String matching* ini dilakukan dengan menggunakan algoritma *Fuzzy string matching* pada Python.

Setelah melakukan tahapan data *preprocessing* di atas maka data siap digunakan untuk analisis *social network*.

3.6 *Social Network Analysis* (SNA)

SNA merupakan proses menganalisis pola hubungan antar caleg berdasarkan data yang telah didapatkan pada proses sebelumnya. SNA terdiri dari *node* (aktor) yang saling terhubung satu sama lain dan memiliki jenis ketergantungan yang beragam, seperti:

- Memiliki dapil yang sama
- Bergabung dengan partai yang sama
- Memiliki lokasi yang sama
- Memiliki agama yang sama, dan lain-lain.

Dalam penelitian ini, caleg bertindak sebagai *node* (aktor) yang akan dianalisis hubungannya dengan caleg lainnya. Untuk menganalisis hubungan yang dimiliki oleh para caleg dapat ditentukan dengan mengetahui nilai sentralitas dan nilai berdasarkan algoritma *community detection*.

Perhitungan algoritma sentralitas pada data caleg meliputi:

1. *Degree centrality*
   Dengan algoritma ini akan diketahui berapa banyak relasi yang dimiliki seorang caleg, baik itu relasi yang masuk atau relasi yang keluar. Selain itu, algoritma ini juga memungkinkan untuk

mengetahui apakah caleg A berhubungan langsung dengan caleg B atau caleg A terhubung dengan caleg B melalui *node* dengan properti yang sama, misalnya sama-sama tergabung dengan partai X.

2. *Closeness centrality*
   Dengan algoritma ini akan diketahui jarak antara satu caleg dengan caleg yang lain berdasarkan jarak rata-rata dari satu caleg ke seluruh caleg yang ada dalam jaringan. Algoritma ini digunakan untuk menemukan caleg mana yang paling cepat mempengaruhi atau menyebarkan informasi ke seluruh *network*.
3. *Betweenness centrality*
   Dengan algoritma ini dapat menunjukkan posisi caleg sebagai perantara dari caleg satu dengan caleg yang lain. Caleg yang memiliki nilai *betweenness centrality* tinggi adalah caleg yang memiliki kemampuan penghubung yang baik antar caleg dalam sebuah jaringan, serta memegang kontrol manipulasi atas informasi.
4. *PageRank*
   Dengan algoritma ini dapat menentukan caleg yang paling populer pada sebuah *network*. Semakin tinggi nilai *PageRank* pada suatu caleg (*node*) maka semakin tinggi pengaruh caleg tersebut terhadap caleg yang lain dalam sebuah jaringan.

Sedangkan untuk perhitungan algoritma *community detection* pada data caleg meliputi:

1. *Triangle count and clustering coefficient*
   Dengan algoritma ini dapat mendeteksi caleg mana yang berada dalam satu komunitas yang sama (misal: partai) dan mengukur kekompakan komunitas tersebut.

2. *Strongly connected components*
   Dengan algoritma ini dapat diketahui siapa saja caleg yang terhubung satu sama lain karena *strongly connected components* hanya ada jika terdapat hubungan antara dua *node* (caleg) dari kedua arah. Dengan begitu, nantinya juga akan diketahui arah hubungan dari sebuah *network*.
3. *Connected components*
   Dengan algortima ini dapat diketahui siapa saja caleg yang terhubung tanpa melihat arah hubungannya. *Connected components* digunakan pada awal analisis untuk memahami struktur graf.
4. *Label propagation*
   Dengan algoritma ini dapat mendeteksi dan menemukan komunitas dengan cara memberi label pada *node* (caleg) sebagai bentuk klasifikasi. Jadi, caleg yang memiliki label yang sama maka dapat diindikasikan bahwa caleg tersebut dalam satu komunitas yang sama juga.
5. *Louvain modularity*
   Dengan algoritma ini dapat mengetahui nilai modularitas pada sebuah komunitas. Nilai modularitas dapat menjadi evaluasi untuk sebuah kelompok terkait dengan hubungan antar *node* yang berada di dalamnya.

Luaran dari proses analisis ini adalah mengetahui pola hubungan antar caleg yang digambarkan oleh *node* berdasarkan nilai sentralitas dan nilai algoritma *community detection* yang dimiliki oleh *node*.

3.7 Dataset Visualisasi

Sebelum membuat visualisasi menggunakan D3js, perlu dilakukan pendefinisian format data yang akan digunakan.

Data yang digunakan untuk keperluan visualisasi dijelaskan pada tabel berikut.

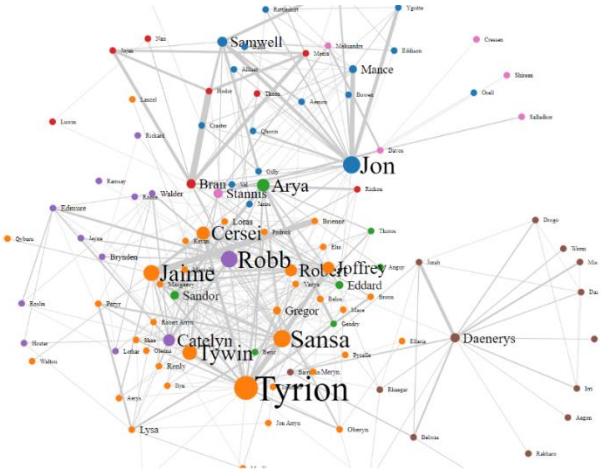
Tabel 3. 1 Dataset Visualisasi

No	Nama Kolom	Tipe Data	Keterangan
1.	Nama	String	Merupakan nama caleg yang mendaftar pada Pemilu 2019
2.	Lokasi	String	Merupakan lokasi tempat caleg tinggal
3.	Partai	String	Merupakan nama partai yang menaungi caleg yang mendaftar Pemilu
4.	Pasangan	String	Merupakan nama pasangan caleg yang statusnya sudah menikah
5.	*Degree Centrality*	Integer	Merupakan hasil perhitungan algoritma *degree centrality* pada SNA
6.	*Closeness Centrality*	Decimal	Merupakan hasil perhitungan algoritma *closeness centrality* pada SNA
7.	*Betweenness Centrality*	Decimal	Merupakan hasil perhitungan algoritma *betweenness centrality* pada SNA
8.	*PageRank*	Decimal	Merupakan hasil perhitungan algoritma *PageRank* pada SNA

9.	*Triangle count and clustering coefficient*	Integer	Merupakan hasil perhitungan algoritma *triangle count and clustering* pada *community detection*
10.	*Strongly connected components*	Integer	Merupakan hasil perhitungan algoritma *strongly connected components* pada *community detection*
11.	*Connected components*	Integer	Merupakan hasil perhitungan algoritma *connected components* pada *community detection*
12.	*Label propagation*	Integer	Merupakan hasil perhitungan algoritma *label propagation* pada *community detection*
13.	*Louvain modularity*	Integer	Merupakan hasil perhitungan algoritma *louvain modularity* pada *community detection*

3.8 Pembuatan Visualisasi Graf

Proses pembuatan visualisasi *graph* bertujuan untuk memberikan gambaran hubungan antar caleg berupa graf berdasarkan hasil dari proses algoritma SNA. Proses ini dilakukan dengan menggunakan D3js. Dari banyaknya variasi visualisasi yang ada pada D3js, pada tugas akhir ini akan menggunakan visualisasi jenis *force directed graph*.



Gambar 3. 3 Contoh Force Directed Graph dengan Studi Kasus Film Seri 'Game of Thones' [19]

Gambar 3.3 menunjukkan contoh visualisasi *force directed graph* yang disertai dengan label nama pada studi kasus hubungan antar karakter film seri 'Game of Thrones'. Visualisasi ini dibuat menggunakan HTML, SVG, dan CSS.

3.9 Penyusunan Laporan Tugas Akhir

Tahapan terakhir dalam metodologi adalah penyusunan laporan tugas akhir. Hal ini dilakukan sebagai bentuk dokumentasi atas terlaksananya tugas akhir. Laporan tugas akhir dibuat sesuai dengan format yang telah ditentukan. Penyusunan laporan tugas akhir dilakukan sejak awal hingga berakhirnya proses pengerjaan tugas akhir.

BAB IV
PERANCANGAN

Bab perancangan menjelaskan tentang hal-hal yang perlu dipersiapkan dalam proses pengerjaan tugas akhir sesuai yang telah diuraikan pada bab sebelumnya. Perancangan ini meliputi akuisisi data, arsitektur sistem, data *preprocessing*, desain basis data, perhitungan algoritma SNA, dan pembuatan visualisasi graf.

4.1. Akuisisi Data

Akuisisi data merupakan proses untuk mendapatkan data caleg dan pemenang dengan cara *crawling website* Pintar Memilih (https://pintarmemilih.id/) dan *website* resmi DPR RI (__). Terdapat dua macam cara dalam melakukan pengambilan data (*crawling*) pada *website* tersebut yaitu dengan *web crawler* Python dan ParseHub. Berikut merupakan penjelasan mengenai akuisisi data terhadap masing-masing *website*.

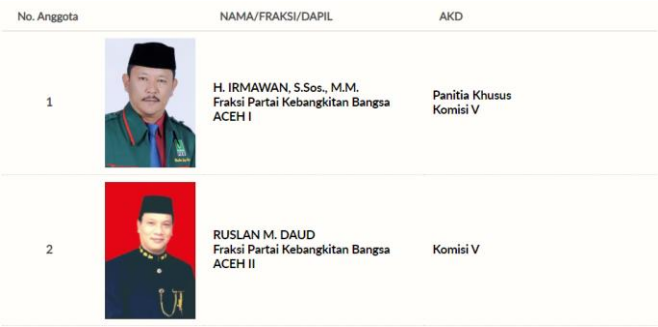
4.1.1 *Crawling* Data Caleg

Crawling data caleg pada *website* Pintar Memilih dilakukan dengan menggunakan aplikasi ParseHub karena selain lebih mudah dan cepat, metode ini dapat menambah pengetahuan baru tentang cara *crawling* pada sebuah *website*. Pada *website* ini, data yang dibutuhkan berupa biodata caleg yang meliputi nama lengkap, alamat, jenis kelamin, partai pengusung, dapil, agama, pendidikan, pekerjaan, dan status pernikahan. Hasil *crawling* menggunakan ParseHub dapat diunduh dan disimpan ke dalam format csv.

4.1.2 *Crawling* Data Pemenang

Crawling data pemenang pada *website* resmi DPR dilakukan dengan menggunakan bahasa pemrograman Python dengan *library* Beautiful Soup yang berfungsi

untuk mengambil data dari HTML. Dengan *library* ini pengambilan data semakin mudah untuk dilakukan karena hanya perlu memanggil elemen yang diinginkan dari HTML.



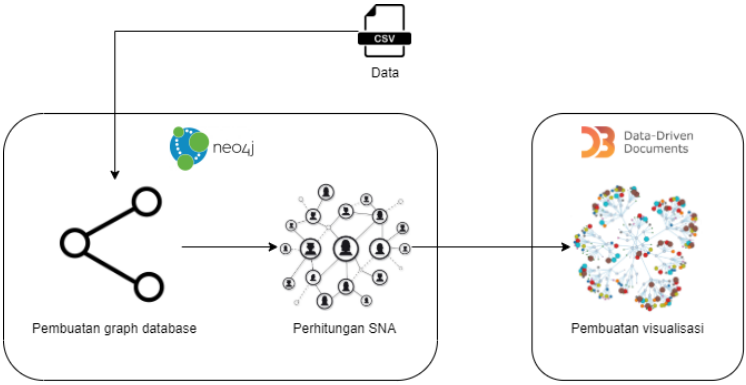
No. Anggota		NAMA/FRAKSI/DAPIL	AKD
1		H. IRMAWAN, S.Sos., M.M. Fraksi Partai Kebangkitan Bangsa ACEH I	Panitia Khusus Komisi V
2		RUSLAN M. DAUD Fraksi Partai Kebangkitan Bangsa ACEH II	Komisi V

Gambar 4. 1 Data Pemenang pada *Website* DPR

Gambar 4.1 menunjukkan tabel anggota (pemenang) pada *website* DPR dimana terdapat empat kolom, yaitu No. Anggota, foto, nama/fraksi/dapil, dan AKD. Namun data yang perlu diambil hanya nama, fraksi (partai), dan dapil anggota yang berada di kolom ketiga pada tabel. Dengan demikian, *library* Beautiful Soup dapat membantu untuk mengambil data pada kolom ketiga saja tanpa harus mengambil keseluruhan data pada tabel. Hal ini tentu saja dapat mempersingkat waktu dalam pengambilan data. Selain *library* Beautiful Soup, terdapat *library requests* yang berfungsi untuk mengirim *request* HTTP dan *library* csv yang berfungsi untuk menyimpan data hasil *crawling* ke dalam format csv.

4.2. Arsitektur Sistem

Pembuatan arsitektur sistem bertujuan untuk memberikan gambaran mengenai sistem yang akan dibuat untuk menyelesaikan permasalahan dalam tugas akhir.



Gambar 4. 2 Arsitektur Sistem

Gambar 4.2 menunjukkan rancangan arsitektur sistem yang mampu untuk mengubah data CSV ke dalam basis data graf, kemudian melakukan tahap perhitungan algoritma SNA pada Neo4j. Setelah mendapat hasil dalam perhitungan SNA kemudian akan dibuat visualisasi yang bersifat statis dengan menggunakan D3js yang akan ditampilkan pada *web*.

4.3. Data *Preprocessing*

Sebelum diolah dalam basis data graf Neo4j, data mentah yang diperoleh dari *crawling website* dilakukan *cleansing* dan *string matching*. *Cleansing* data dilakukan untuk menghapus baris yang memiliki nilai *null* atau kosong. Sedangkan *string matching* dilakukan untuk mengetahui biodata pemenang pada data caleg. Selain itu, *string matching* juga digunakan untuk mengetahui caleg atau pemenang yang memiliki alamat yang sama. Dari kesamaan alamat tersebut dapat mengindikasikan bahwa mereka memiliki hubungan kerabat, baik saudara, pasangan, maupun anak. Proses *cleansing* dan *string matching* dilakukan dengan menggunakan Python.

Setelah melakukan proses *cleansing* dan *string matching*, data selanjutnya akan dipisah menjadi beberapa *file* csv berdasarkan

konten biodata, yaitu nama, jenis kelamin, partai, dapil, agama, pendidikan, pekerjaan, dan status pernikahan. Masing-masing diberi Id agar tidak terjadi redundansi data saat diolah di basis data graf.

4.4. Desain Basis Data

Setelah melalui tahap *preprocessing* selanjutnya adalah mengolah data pada basis data graf Neo4j. Semua *file* csv yang dihasilkan pada tahap data *prepocessing* akan di-*import* ke dalam Neo4j dengan menggunakan bahasa Cypher. Gambar 4.2 menunjukkan representasi data caleg yang telah di-*import* pada basis data graf Neo4j.

Gambar 4. 3 Desain Basis Data Graf

Pada Gambar 4.3 terdapat macam-macam warna yang berbeda untuk merepresentasikan sebuah *node* graf yang ditunjukkan pada Tabel 4.1.

Tabel 4. 1 Keterangan *Node* pada Graf

No	Node	Warna
1	Caleg	Abu-abu
2	Pemenang	Kuning
3	Dapil	Biru tua
4	Pendidikan	Biru muda
5	Partai	Hijau tua
6	Pekerjaan	Hijau muda
7	Status pernikahan	Ungu
8	Agama	Merah muda
9	Jenis kelamin	Merah

Sedangkan untuk keterangan jenis hubungan atau *edge* pada graf ditunjukkan pada Tabel 4.2.

Tabel 4. 2 Keterangan *Edge* pada Graf

No	Edge	Penjelasan
1	LINK	Untuk menghubungkan node caleg dan pemenang ke node atribut biodata seperti agama, dapil, partai, dll
2	KERABAT	Untuk menghubungkan node caleg ke node pemenang atau ke sesamanya yang memiliki alamat yang sama

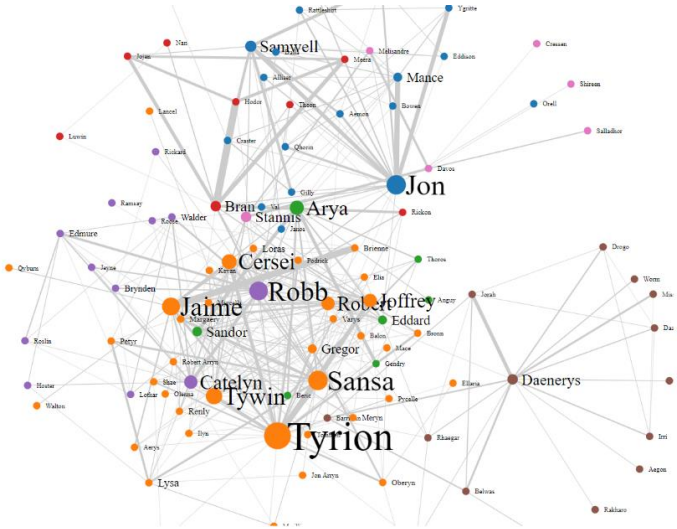
4.5. Perhitungan Algoritma SNA

Perhitungan algoritma SNA dilakukan untuk mengetahui *node* caleg atau pemenang yang berpengaruh pada *network* yang telah dibuat di Neo4j. Algoritma ini meliputi *degree centrality, betweenness centrality, closeness centrality,* dan *page rank.* Selain algoritma tersebut juga digunakan algoritma *community detection* yang terdiri dari *triangle count and clustering coefficient, strongly connected component, connected component, label propagation, Louvain modularity,* dan *validating communities* untuk mengetahui siapa saja yang menjalin hubungan kerabat pada Pemilu 2019. Semua algoritma

tersebut akan dijalankan di Neo4j dengan menggunakan bahasa Cypher. Hasil dari tahapan ini berupa *file* csv dan json yang berisi nilai perhitungan pada masing-masing *node.*

4.6. Pembuatan Visualisasi Graf

Masing-masing nilai perhitungan algoritma SNA akan divisualisasikan ke dalam bentuk *force directed graph* statis menggunakan D3js berbasis *web*. Semakin besar nilai sebuah *node* maka akan semakin berpengaruh *node* tersebut dalam jaringan, begitu juga sebaliknya. Gambar 4.4 menunjukkan gambaran visualisasi *force directed graph* dengan D3js menggunkaan dataset tokoh dalam Game of Thrones.



Gambar 4. 4 *Mock-up Force Directed Graph* dengan Dataset 'Game of Thrones'

BAB V
IMPLEMENTASI

Bab implementasi menjelaskan tentang penerapan metode dalam pengerjaan tugas akhir sesuai dengan perancangan yang dilakukan sebelumnya. Tahapan pada bab ini meliputi lingkungan implementasi, akuisisi data, data *preprocessing*, desain basis data, perhitungan algoritma SNA, dan pembuatan visualisasi graf.

5.1 Lingkungan Implementasi

Dalam pengerjaan tugas akhir ini diperlukan perangkat keras maupun perangkat lunak yang digunakan untuk menunjang proses penelitian. Berikut merupakan Tabel 5.1 dan Tabel 5.2 yang berisi daftar perangkat yang digunakan.

Tabel 5. 1 Spesifikasi Perangkat Keras

No	Hardware	Spesifikasi
1.	Laptop	ASUS ROG GL552VX
2.	Processor	Intel® Core ™ i7-6700HQ
3.	RAM	16 GB
4.	Harddisk	1 TB

Tabel 5. 2 Spesifikasi Perangkat Lunak dan Library

Bahasa Pemrograman	• Python 3.7 • Cypher
Text Editor	• Visual Studio Code • Notepad
Software/Tools	• Microsoft Office 365 • ParseHub • Neo4j • D3js

Library	• Pandas • CSV • Request • Time • Beautiful Soup • Itertools • Fuzzywuzzy

5.2 Akuisisi Data

Tahap awal dalam penelitian ini merupakan pengumpulan data caleg DPR beserta pemenangnya pada Pemilu 2019. Berikut merupakan Tabel 5.3 yang menunjukkan jumlah data yang digunakan dalam penelitian.

Tabel 5. 3 Jumlah Data

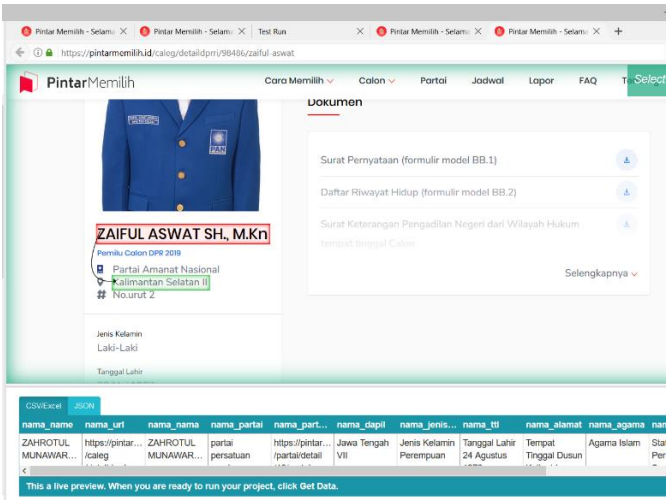
	Jumlah
Caleg	7422
Pemenang	572

Data pada Tabel 5.3 didapatkan dari hasil *crawling* data dari *website* Pintar Memilih dan *website* resmi DPR sebagaimana telah disebutkan pada bab sebelumnya. Proses *crawling* data pada masing-masing *website* akan dijelaskan pada sub bab berikut.

5.2.1 *Crawling* Data Caleg

Sebagaimana telah disebutkan pada bab sebelumnya, *crawling* data caleg dilakukan dengan aplikasi *web crawler* ParseHub. Pada dasarnya aplikasi ini dijalankan tanpa menggunakan kodingan, hanya klik item yang datanya ingin diekstrak dari sebuah *website.* Gambar 5.1 menunjukkan proses *crawling* data caleg menggunakan *web crawler* ParseHub. Pengambilan data ini menggunakan perintah *relative select* yang berfungsi

untuk membuat pola hubungan dari *parent element* (berada di dalam kotak merah) ke *child element* (berada di dalam kotak hijau). Dalam kasus ini, *parent element* adalah nama caleg, sedangkan *child element* adalah partai, dapil, jenis kelamin, alamat, agama, pendidikan, pekerjaan, dan status pernikahan.



Gambar 5. 1 Proses *Crawling* Data Caleg Menggunakan ParseHub

Pada bagian bawah Gambar 5.1 terdapat tabel yang menunjukkan *preview* data yang akan diekstrak. Data tersebut dapat diunduh dalam format csv dan json.

5.2.2 *Crawling* Data Pemenang

Sebagaimana telah disebutkan pada bab sebelumnya, *crawling* data pemenang dilakukan menggunakan Python dengan *library* Beautiful Soup. Kode 5.1 menunjukkan kodingan yang dijalankan untuk mendapatkan data pemenang. Baris ke-17 menunjukkan bahwa data yang akan diambil berada di dalam tabel pada *website* DPR, dan baris ke-18 berfungsi untuk mengambil data pada

kolom ke-3 pada tabel tersebut. Hasil *crawling* selanjutnya disimpan ke dalam *file* dpr.csv dengan kolom 'Nama' seperti yang tertera pada baris ke-6 dan ke-7 pada kodingan.

```
1.  # importing the libraries
2.  from bs4 import BeautifulSoup
3.  import csv
4.  import requests
5.
6.  f = csv.writer(open('dpr.csv', 'w'))
7.  f.writerow(['Nama'])
8.
9.  url="http://www.dpr.go.id/anggota"
10.
11. # Make a GET request to fetch the raw HT
    ML content
12. html_content = requests.get(url).text
13.
14. # Parse the html content
15. soup = BeautifulSoup(html_content, 'html
    .parser')
16.
17. for row in soup.findAll('table')[0].tbod
    y.findAll('tr'):
18.     third_column = row.findAll('td')[2].
    contents
19.     #print (third_column)
20.     f.writerow(third_column)
```

Kode 5. 1 *Crawling* Data Pemenang

5.3 Data *Preprocessing*

Data yang telah dikumpulkan pada proses sebelumnya kemudian dilakukan *preprocessing* sebelum dimasukkan ke dalam basis data graf. Data *preprocessing* yang dilakukan adalah sesusai dengan tahapan berikut.

1. Tahap pertama pada data *preprocessing* adalah *string matching* data pemenang dengan data caleg. Hal ini bertujuan untuk mengetahui biodata pemenang. Proses *string matching* dilakukan menggunakan Python. Kode

5.2 menunjukkan proses *string matching* data pemenang terhadap data caleg. *Library* yang digunakan adalah *itertools*, *fuzzywuzzy*, dan *pandas*. *Library itertools* digunakan untuk melakukan operasi *cartesian product* pada nama pemenang dan nama caleg. *Library*

```
1.  from itertools import product
2.  import itertools
3.  from fuzzywuzzy import fuzz
4.  import pandas as pd
5.
6.  def func(to_find, some_sentences):
7.      list = []
8.      for sub, sentence in product(to_fi
    nd, some_sentences):
9.          ratio = (fuzz.token_sort_ratio
    (sub, sentence))
10.         if ratio > threshold_ratio:
11.             row = (sub, sentence, rati
    o)
12.             list.append(row)
13.                 #print(list)
14.     df = pd.DataFrame(list)
15.     df.columns = ["Pemenang", "Nama",
    "Ratio"]
16.     return df
17.
18. data1 = pd.read_excel('E:\\pemenang.xl
    sx')
19. data2 = pd.read_excel('E:\\biodata_cal
    eg.xlsx', index_col=None)
20. to_find = data1['Nama']
21. some_sentences = data2['Nama']
22. threshold_ratio = 80
23. asd = func(to_find, some_sentences)
24. asd
25. result = pd.merge(asd, data2, on='Nama
    ')
26. result.to_csv('E:/biodata_pemenang.csv
    ', mode='a', index=False, sep=';', hea
    der=False, na rep='Unkown')
```

Kode 5. 2 Proses *String Matching* Data Pemenang Terhadap Data Caleg

fuzzywuzzy digunakan untuk menemukan *string* yang cocok atau memiliki *pattern* yang sama. Sedangkan *library pandas* digunakan untuk membuat *data frame.* Baris ke-6 merupakan pendefinisian fungsi yang menggunakan *library itertools product* dengan parameter to_find dan some_sentences. Parameter to_find merepresentasikan nama pemenang dan parameter some_sentences merepresentasikan nama caleg. Tingkat kecocokan *string* pada data dapat dihitung menggunakan *fuzz* pada *library fuzzywuzzy*. Jika tingkat kecocokan lebih dari rasio yang telah ditentukan (threshold_ratio) seperti yang ditunjukkan pada baris ke-10 berarti kemungkinan besar biodata pemenang ada di dalam data caleg. Hasil dari proses *string matching* data pemenang terhadap data caleg disimpan dalam *file* biodata_pemenang.csv.

2. Tahap kedua adalah *cleansing* data yang bertujuan untuk menghapus baris yang memiliki nilai *null* pada biodata caleg dan pemenang.

```
1. import pandas as pd
2. data = pd.read_excel('E:\\biodata_peme
   nang.xlsx')
3. new_data = data.dropna(axis = 0, how =
   'any')
4. print("Data lama:", len(data), "\nData
    baru:",
5.        len(new_data), "\nJumlah baris
   yang dihapus: ",
6.        (len(data)-len(new_data)))
7. new_data.to_csv('E:/biodata_pemenang_n
   ew.csv', index=False, sep=';', na_rep=
   'Unkown')
```

Kode 5. 3 Proses *Cleansing* Data Pemenang

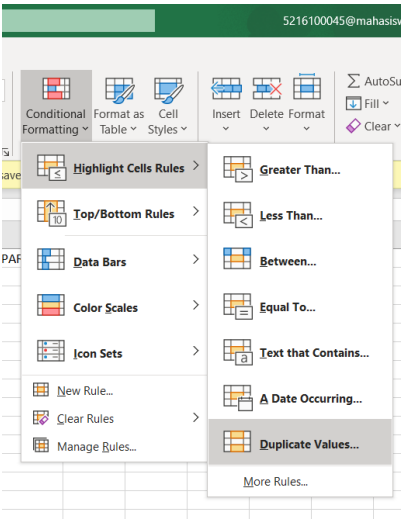
Baris ke-3 pada Kode 5.3 berfungsi untuk menghapus baris yang memiliki nilai *null*. Baris ke-4 digunakan untuk mengetahui jumlah data yang terhapus dan yang tersisa setelah dilakukan proses *cleansing*. Sama halnya

dengan Kode 5.4 yang mana merupakan proses *cleansing* pada data caleg. Hasil *cleansing* pada masing-masing data disimpan pada *file* csv yang baru.

```
1. import pandas as pd
2. data = pd.read_excel('E:\\biodata_caleg.xlsx')
3. new_data = data.dropna(axis = 0, how = 'any')
4. print("Data lama:", len(data), "\nData baru:",
5.        len(new_data), "\nJumlah baris yang dihapus: ",
6.        (len(data)-len(new_data)))
7. new_data.to_csv('E:/biodata_caleg_new.csv', index=False, sep=';', na_rep='Unkown')
```

Kode 5. 4 Proses *Cleansing* Data Caleg

3. Tahap ketiga adalah *string matching* alamat yang dimiliki oleh masing-masing caleg dan pemenang. Tujuannya adalah untuk mencari alamat yang sama. Jika terdapat lebih dari satu caleg atau pemenang yang memiliki alamat yang sama maka kemungkinan besar bahwa mereka memiliki hubungan kerabat karena tinggal di alamat yang sama. Tahap ini dilakukan dengan menggunakan fitur *highlight duplicate value* pada kolom Alamat di Ms. Excel dan menggunakan *library* Python seperti pada proses *string matching* sebelumnya.



Gambar 5. 2 Proses *String Matching* Alamat Menggunakan Ms. Excel

Gambar 5.2 menunjukkan proses *string matching* menggunakan fitur *highlight duplicate value* pada Ms. Excel. Fitur ini hanya dapat digunakan pada satu *file* yang sama. Oleh karena itu, *output* dari cara ini menunjukkan hubungan kerabat sesama pemenang atau sesama caleg saja. Namun ada kemungkinan bahwa terdapat caleg yang memiliki alamat yang sama dengan pemenang, atau sebaliknya. Oleh karena itu dibutuhkan cara *string matching* lain untuk menyocokkan alamat caleg dan alamat pemenang yang berada di dua *file* yang berbeda.

```
1.  from itertools import product
2.  import itertools
3.  from fuzzywuzzy import fuzz
4.  import pandas as pd
5.
6.  def func(to_find, some_sentences):
7.      list = []
8.      for sub, sentence in product(to_fi
    nd, some_sentences):
9.          ratio = (fuzz.token_sort_ratio
    (sub, sentence))
10.         if ratio > threshold_ratio:
11.             row = (sub, sentence, rati
    o)
12.             list.append(row)
13.                 #print(list)
14.     df = pd.DataFrame(list)
15.     df.columns = ["Alamat Pemenang", "
    Alamat Caleg", "Ratio"]
16.     return df
17.
18. data1 = pd.read_excel('E:/dpr_done/yan
    g dipake/biodata_pemenang_new.xlsx')
19. data2 = pd.read_excel
    ('E:/dpr_done/yang dipake/biodata_cale
    g_new.xlsx', index_col=None)
20. to_find = data1['Alamat']
21. some_sentences = data2['Alamat']
22. threshold_ratio = 99
23. asd = func(to_find, some_sentences)
24. asd
25. result = pd.merge(asd, data2, on='Alam
    at')
26. result.to_csv('E:/kerabat_caleg_pemena
    ng.csv', mode='a', index=False, sep=';
    ', na_rep='Unkown')
```

Kode 5. 5 Proses *String Matching* Alamat

Kode 5.5 menunjukkan proses *string matching* alamat menggunakan Python seperti yang digunakan pada proses *string matching* sebelumnya dimana data yang digunakan adalah biodata_pemenang_new.csv dan biodata_caleg_new.csv sebagaimana tertera pada baris

ke-18 dan ke-19. *Threshold ratio* yang digunakan kali ini adalah sebesar 99. Jika nilai *ratio* kemiripan lebih dari 99 maka caleg dan pemenang tersebut tinggal di alamat yang sama. Hasil proses ini disimpan di dalam *file* kerabat_caleg_pemenang.csv.

4. Tahap yang terakhir adalah memisahkan data berdasarkan kolom dengan cara membuat *file* baru. Pada setiap *file* dipastikan untuk menambahkan Id yang nantinya akan digunakan pada saat mengimpor *file* ke basis data graf Neo4j.

	A	B
1	id	agama
2	501	Islam
3	502	Kristen Protestan
4	503	Katolik
5	504	Hindu
6	505	Budha
7	506	Lainnya

Gambar 5. 3 Pemisahan Data Berdasarkan Kolom Agama

Gambar 5.3 menunjukkan hasil proses pemisahan data kolom Agama beserta pemberian Id agar tidak terjadi redundansi data.

Selain pemisahan data, di tahap ini juga membuat *file* baru yang berisi kumpulan Id yang nantinya akan digunakan saat membuat relasi graf.

	A	B	C	D	E	F	G	H
1	idAnggota	idPartai	idDapil	idJk	idAgama	idStatus	idPendidikan	idPekerjaan
2	1	204	357	401	504	601	701	801
3	2	203	344	401	502	601	705	801
4	3	207	365	401	501	601	704	801
5	4	202	323	401	501	601	704	805
6	5	205	369	402	502	602	703	805
7	6	203	357	401	504	601	704	801
8	7	203	357	401	504	601	703	802
9	8	203	323	402	501	601	704	801
10	9	208	312	401	501	601	703	801
11	10	202	301	401	501	601	703	801

Gambar 5. 4 Pembuatan Relasi Graf Berdasarkan Id

Gambar 5.4 menunjukkan bahwa anggota/pemenang yang memiliki id 1 memiliki relasi dengan partai yang memiliki id 204, dapil yang memiliki id 357, dan seterusnya. Jika dilihat kembali Gambar 5.4, 504 merupakan id dari agama Hindu, yang berarti anggota dengan id 1 beragama Hindu. Pembuatan relasi graf berdasarkan id juga dilakukan pada data caleg.

5.4 Desain Basis Data

Setelah melalui tahap *preprocessing*, data siap diimpor ke Neo4j untuk membuat graf. Terdapat beberapa *file* yang diimpor ke Neo4j, antara lain:

1. *File* anggota.csv yang berisi nama anggota/pemenang dan id anggota.

```
1. LOAD CSV WITH HEADERS FROM "file:///a
   nggota.csv" AS csvLine
2. CREATE (anggota:Anggota {id: toIntege
   r(csvLine.id), name: csvLine.anggota}
   )
```

Kode 5. 6 Impor Data Anggota ke Neo4j

Baris ke-1 pada Kode 5.6 digunakan untuk mengimpor data anggota sesuai dengan *path file* dan baris ke-2 berfungsi untuk membuat *node* baru dengan menggunakan nama anggota sebagai *property*-nya.

2. *File* caleg.csv yang berisi nama caleg dan id caleg.

```
1. LOAD CSV WITH HEADERS FROM "file:///c
   aleg.csv" AS csvLine
2. CREATE (caleg:Caleg {id: toInteger(cs
   vLine.id), name: csvLine.caleg})
```

Kode 5. 7 Impor Data Caleg ke Neo4j

Baris ke-2 pada Kode 5.7 berfungsi untuk membuat *node* baru dengan menggunakan nama caleg sebagai *property*-nya.

3. *File* agama.csv yang berisi nama agama dan id agama.

```
1. LOAD CSV WITH HEADERS FROM "file:///ag
   ama.csv" AS csvLine
2. CREATE (agama:Agama {id: toInteger(csv
   Line.id), name: csvLine.agama})
```

Kode 5. 8 Impor Data Agama ke Neo4j

Baris ke-2 pada kode 5.8 berfungsi untuk membuat *node* baru dengan menggunakan nama agama sebagai *property*-nya.

4. *File* dapil.csv yang berisi nama dapil dan id dapil.

```
1. LOAD CSV WITH HEADERS FROM "file:///da
   pil.csv" AS csvLine
2. CREATE (dapil:Dapil {id: toInteger(csv
   Line.id), name: csvLine.dapil})
```

Kode 5. 9 Impor Data Dapil ke Neo4j

Baris ke-2 pada kode 5.9 berfungsi untuk membuat *node* baru dengan menggunakan nama dapil sebagai *property*-nya.

5. *File* jk.csv yang berisi jenis kelamin dan id jenis kelamin.

```
1. LOAD CSV WITH HEADERS FROM "file:///jk
   .csv" AS csvLine
2. CREATE (jk:Jk {id: toInteger(csvLine.i
   d), name: csvLine.jk})
```

Kode 5. 10 Impor Data Jenis Kelamin ke Neo4j

Baris ke-2 pada Kode 5.10 berfungsi untuk membuat *node* baru dengan menggunakan jenis kelamin sebagai *property*-nya.

6. *File* partai.csv yang berisi nama partai dan id partai.

```
1. LOAD CSV WITH HEADERS FROM "file:///p
   artai.csv" AS csvLine
2. CREATE (partai:Partai {id: toInteger(
   csvLine.id), name: csvLine.partai})
```

Kode 5. 11 Impor Data Partai ke Neo4j

Baris ke-2 pada Kode 5.11 berfungsi untuk membuat *node* baru dengan menggunakan nama partai sebagai *property*-nya.

7. *File* perkerjaan.csv yang berisi nama pekerjaan dan id pekerjaan.

```
1. LOAD CSV WITH HEADERS FROM "file:///p
   ekerjaan.csv" AS csvLine
2. CREATE (pekerjaan:Pekerjaan {id: toIn
   teger(csvLine.id), name: csvLine.peke
   rjaan})
```

Kode 5. 12 Impor Data Pekerjaan ke Neo4j

Baris ke-2 pada Kode 5.12 berfungsi untuk membuat *node* baru dengan menggunakan nama pekerjaan sebagai *property*-nya.

8. *File* pendidikan.csv yang berisi jenjang pendidikan dan id pendidikan.

```
1. LOAD CSV WITH HEADERS FROM "file:///p
   endidikan.csv" AS csvLine
2. CREATE (pendidikan:Pendidikan {id: to
   Integer(csvLine.id), name: csvLine.pe
   ndidikan})
```

Kode 5. 13 Impor Data Pendidikan ke Neo4j

Baris ke-2 pada Kode 5.13 berfungsi untuk membuat *node* baru dengan menggunakan jenjang pendidikan sebagai *property*-nya.

9. *File* status.csv yang berisi status pernikahan dan id status.

```
1. LOAD CSV WITH HEADERS FROM "file:///status.csv" AS csvLine
2. CREATE (status:Status {id: toInteger(csvLine.id), name: csvLine.status})
```

Kode 5. 14 Impor Data Status Pernikahan ke Neo4j

Baris ke-2 pada Kode 5.14 berfungsi untuk membuat *node* baru dengan menggunakan status pernikahan sebagai *property*-nya.

10. *File* relasi_anggota.csv dan relasi_caleg.csv yang berisi id untuk membuat relasi pada graf.

```
1. LOAD CSV WITH HEADERS FROM "file:///relasi_anggota.csv" AS csvLine
2. MATCH (anggota:Anggota{id: toInteger(csvLine.idAnggota)}),(partai:Partai{id: toInteger(csvLine.idPartai)})
3. CREATE (anggota)-[:LINK]->(partai)
```

Kode 5. 15 Menghubungkan *Node* Anggota dengan *Node* Partai

Baris ke-2 pada Kode 5.15 berfungsi untuk mencocokkan id Anggota dengan id Partai pada *file* relasi_anggota.csv untuk membuat relasi baru bernama LINK yang ditunjukkan pada baris ke-3.

```
1. LOAD CSV WITH HEADERS FROM "file:///relasi_anggota.csv" AS csvLine
2. MATCH (anggota:Anggota{id: toInteger(csvLine.idAnggota)}),(dapil:Dapil{id: toInteger(csvLine.idDapil)})
3. CREATE (anggota)-[:LINK]->(dapil)
```

Kode 5. 16 Menghubungkan *Node* Anggota dengan *Node* Dapil

Baris ke-2 pada Kode 5.16 berfungsi untuk mencocokkan id Anggota dengan id Dapil pada *file*

relasi_anggota.csv untuk membuat relasi baru bernama LINK yang ditunjukkan pada baris ke-3.

```
1. LOAD CSV WITH HEADERS FROM "file:///relasi_anggota.csv" AS csvLine
2. MATCH (anggota:Anggota{id: toInteger(csvLine.idAnggota)}),(jk:Jk{id: toInteger(csvLine.idJk)})
3. CREATE (anggota)-[:LINK]->(jk)
```

Kode 5. 17 Menghubungkan *Node* Anggota dengan *Node* Jenis Kelamin

Baris ke-2 pada Kode 5.17 berfungsi untuk mencocokkan id Anggota dengan id Jenis Kelamin pada *file* relasi_anggota.csv untuk membuat relasi baru bernama LINK yang ditunjukkan pada baris ke-3.

```
1. LOAD CSV WITH HEADERS FROM "file:///relasi_anggota.csv" AS csvLine
2. MATCH (anggota:Anggota{id: toInteger(csvLine.idAnggota)}),(agama:Agama{id: toInteger(csvLine.idAgama)})
3. CREATE (anggota)-[:LINK]->(agama)
```

Kode 5. 18 Menghubungkan *Node* Anggota dengan *Node* Agama

Baris ke-2 pada Kode 5.18 berfungsi untuk mencocokkan id Anggota dengan id Agama pada *file* relasi_anggota.csv untuk membuat relasi baru bernama LINK yang ditunjukkan pada baris ke-3.

```
1. LOAD CSV WITH HEADERS FROM "file:///re
   lasi_anggota.csv" AS csvLine
2. MATCH (anggota:Anggota{id: toInteger(c
   svLine.idAnggota)}),(status:Status{id:
    toInteger(csvLine.idStatus)})
3. CREATE (anggota)-[:LINK]->(status)
```

Kode 5. 19 Menghubungkan *Node* Anggota dengan *Node* Status Pernikahan

Baris ke-2 pada Kode 5.19 berfungsi untuk mencocokkan id Anggota dengan id Status Pernikahan pada *file* relasi_anggota.csv untuk membuat relasi baru bernama LINK yang ditunjukkan pada baris ke-3.

```
1. LOAD CSV WITH HEADERS FROM "file:///re
   lasi_anggota.csv" AS csvLine
2. MATCH (anggota:Anggota{id: toInteger(c
   svLine.idAnggota)}),(pendidikan:Pendid
   ikan{id: toInteger(csvLine.idPendidika
   n)})
3. CREATE (anggota)-[:LINK]-
   >(pendidikan)
```

Kode 5. 21 Menghubungkan *Node* Anggota dengan *Node* Pendidikan

Baris ke-2 pada Kode 5.20 berfungsi untuk mencocokkan id Anggota dengan id Pendidikan pada *file* relasi_anggota.csv untuk membuat relasi baru bernama LINK yang ditunjukkan pada baris ke-3.

```
1. LOAD CSV WITH HEADERS FROM "file:///re
   lasi_anggota.csv" AS csvLine
2. MATCH (anggota:Anggota{id: toInteger(c
   svLine.idAnggota)}),(pekerjaan:Pekerja
   an{id: toInteger(csvLine.idPekerjaan)}
   )
3. CREATE (anggota)-[:LINK]-
   >(pekerjaan)
```

Kode 5. 20 Menghubungkan *Node* Anggota dengan *Node* Pekerjaan

Baris ke-2 pada Kode 5.20 berfungsi untuk mencocokkan id Anggota dengan id Pekerjaan pada *file* relasi_anggota.csv untuk membuat relasi baru bernama LINK yang ditunjukkan pada baris ke-3.

Tahapan yang dilakukan pada *file* relasi_anggota.csv juga dilakukan pada *file* relasi_caleg.csv untuk membuat relasi caleg pada graf.

Setelah semua *node* telah dibuat, tahap terkahir pada pembuatan desain basis data adalah menghubungkan caleg atau anggota yang memiliki hubungan kerabat berdasarkan hasil *string matching* alamat.

```
1.  MATCH (a:Anggota),(b:Anggota)
2.  WHERE a.id = 329 AND b.id = 330
3.  CREATE k = (a)-[r:KERABAT]->(b)
4.  RETURN k
```

Kode 5. 22 Menghubungan *Node* yang Memiliki Hubungan Kerabat

Kode 5.22 merupakan salah satu contoh kode yang berfungsi untuk mencocokkan anggota dengan id 329 dan 330 yang mana pada hasil *string matching* alamat menunjukkan bahwa keduanya memiliki hubungan kerabat.

Gambar 5. 5 Desain Basis Data Graf

Gambar 5.5 merupakan bentuk basis data graf yang setelah semua *node* dan relasi telah dibuat.

5.5 Perhitungan Algoritma SNA

Tahap ini merupakan tahap yang paling utama dalam penelitian ini. Dengan melakukan perhitungan algoritma SNA pada graf dapat mengetahui pengaruh sebuah *node* dalam graf. Terdapat dua jenis algoritma SNA yang akan dijalankan, yaitu algoritma *centrality* yang berfungsi untuk mendeteksi *node* yang berpengaruh dalam graf dan algoritma *community detection* yang berfungsi untuk mengetahui anggota atau caleg yang menjalin hubungan kerabat.

5.5.1. Algoritma *Centrality*

Terdapat empat algoritma *centrality* yang dilakukan dalam penelitian ini, yaitu:

1. *Degree Centrality*

```
1. CALL algo.degree.stream(null, null, {direction: "both"})
2. YIELD nodeId, score
3. RETURN algo.asNode(nodeId).name AS name, score
4. ORDER BY score DESC
```

Kode 5. 23 Algoritma Degree Centrality

Baris ke-1 pada Kode 5.23 berfungsi untuk menjalankan algoritma *degree centrality*. Parameter pertama menunjukkan nama *node* dan parameter kedua menunjukkan nama relasi. Pada kode tersebut parameter yang digunakan adalah *null* dengan *direction both* karena agar dapat menunjukkan *node* mana yang memiliki relasi masuk dan keluar terbanyak. Hasil yang ditampilkan adalah nama *node* yang diurutkan berdasarkan jumlah relasi terbanyak.

2. *Betweenness Centrality*

```
1. CALL algo.betweenness.stream()
2. YIELD nodeId, centrality
3. RETURN algo.asNode(nodeId).name AS nam
   e, centrality
4. ORDER BY centrality DESC
```

Kode 5. 24 Algoritma *Betweenness Centrality*

Baris ke-1 pada Kode 5.24 berfungsi untuk menjalankan algoritma *betweenness centrality*. Hasil yang ditampilkan adalah nama *node* yang diurutkan berdasarkan nilai *betweenness centrality* yang tertinggi.

3. *Closeness Centrality*

```
1. CALL algo.closeness.stream()
2. YIELD nodeId, centrality
3. RETURN algo.asNode(nodeId).name AS nam
   e, centrality
4. ORDER BY centrality DESC
```

Kode 5. 25 Algoritma *Closeness Centrality*

Baris ke-1 pada Kode 5.25 berfungsi untuk menjalankan algoritma *closeness centrality*. Hasil yang ditampilkan adalah nama *node* yang diurutkan berdasarkan nilai *closeness centrality* yang tertinggi.

4. *PageRank*

```
1. CALL algo.pageRank.stream(null, null,
   {iterations:20, dampingFactor:0.85})
2. YIELD nodeId, score
3. RETURN algo.asNode(nodeId).name AS nam
   e, score
4. ORDER BY score DESC
```

Kode 5. 26 Algoritma *PageRank*

Baris ke-1 pada Kode 5.26 berfungsi untuk menjalankan algoritma *pagerank*. Fungsi *iterations* pada parameter adalah mengatur seberapa banyak iterasi yang digunakan. Pada penelitian ini menggunakan iterasi *default* yaitu 20. Sedangkan

dampingFactor merupakan faktor yang digunakan dalam perhitungan *pagerank* yang dapat diatur antara 0 dan 1 namun pada penelitian ini *dumpingFactor* yang digunakan adalah 0,85 yang mana merupakan nilai *default*.

5.5.2. Algoritma *Community Detection*

Terdapat lima algoritma *community detection* yang digunakan untuk mendeteksi komunitas pada graf. Komunitas yang dimaksud dalam penelitian ini adalah *node* yang memiliki keterikatan hubungan satu sama lain.

1. *Triangle Count and Clustering Coefficient*

```
1. CALL algo.triangle.stream()
2. YIELD nodeA, nodeB, nodeC
3. RETURN algo.getNodeById(nodeA).name AS
   nodeA, algo.getNodeById(nodeB).name AS
   nodeB, algo.getNodeById(nodeC).name AS
   nodeC
```

Kode 5. 27 Algoritma *Triangle Count*

Baris ke-1 pada Kode 5.27 berfungsi untuk menjalankan algoritma *triangle count*. Hasil yang ditampilkan adalah nama *node* yang relasinya membentuk segitiga atau *triangle* sesuai dengan nama algoritmanya.

```
1. CALL algo.triangleCount.stream()
2. YIELD nodeId, triangles, coefficient
3. WHERE coefficient > 0
4. RETURN algo.getNodeById(nodeId).name A
   S name, coefficient
5. ORDER BY coefficient DESC
```

Kode 5. 28 Algoritma *Local Clustering*

Baris ke-1 pada Kode 5.28 berfungsi untuk menjalankan algoritma *local clustering*. *Coefficient* merupakan nilai yang menunjukkan seberapa banyak *neighbors* sebuah *node* saling berhubungan. Hasil yang

ditampilkan adalah nama *node* yang diurutkan berdasarkan nilai *coefficient* yang tertinggi.

2. *Strongly Connected Components*

```
1. CALL algo.scc.stream()
2. YIELD nodeId, partition
3. RETURN partition, collect(algo.getNodeB
   yId(nodeId).name) AS name
4. ORDER BY size(name) DESC
```

Kode 5. 29 Algoritma *Strongly Connected Components*

Baris ke-1 pada Kode 5.29 berfungsi untuk menjalankan algoritma *strongly connected components*. *Partition* merupakan bagian yang terbentuk dari *node* yang memiliki keterkaitan kuat satu sama lain. Hasil yang ditampilkan adalah kode *partition* dan nama-nama *node* yang berada dalam *partition* tersebut.

3. *Connected Components*

```
1. CALL algo.unionFind.stream()
2. YIELD nodeId, setId
3. RETURN setId, collect(algo.getNodeById
   (nodeId).name) AS name
4. ORDER BY size(name) DESC
```

Kode 5. 30 Algoritma *Connected Components*

Baris ke-1 pada Kode 5.30 berfungsi untuk menjalankan algoritma *connected components*. *setId* merupakan *index* id kolom yang dimulai dari 0. Hasil yang ditampilkan adalah setId yang berisi nama-nama *node* yang terhubung melalui sebuah relasi.

4. *Label Propagation*

```
1. CALL algo.labelPropagation.stream("Ang
   gota", null, {iterations:10})
2. YIELD nodeId, label
3. RETURN label, collect(algo.getNodeById
   (nodeId).name) AS name
4. ORDER BY size(name) DESC
```

Kode 5. 31 Algoritma *Label Propagation* dengan Parameter *Node* Anggota

Baris ke-1 pada Kode 5.31 berfungsi untuk menjalankan algoritma *label propagation* pada dengan parameter anggota dan iterasi 10. *Label* merupakan *identifier* yang menyatakan bahwa *node* tersebut berada dalam satu komunitas.

```
1. CALL algo.labelPropagation.stream("Cal
   eg", null, {iterations:10})
2. YIELD nodeId, label
3. RETURN label, collect(algo.getNodeById
   (nodeId).name) AS name
4. ORDER BY size(name) DESC
```

Kode 5. 32 Algoritma *Label Propagation* dengan Parameter *Node* Caleg

Kode 5.32 menunjukkan kode untuk menjalankan algoritma *label propagation* dengan parameter Caleg dan nilai *iterations* yang sama dengan algoritma *label propagation* pada *node* Anggota. Hasil yang ditampilkan adalah *label* yang berisi nama-nama *node* yang tergabung dan menjadi sebuah komunitas sendiri.

5. *Louvain Modularity*

```
1. CALL algo.louvain.stream('Anggota', nu
   ll, {
2.  graph: 'huge',
3.  direction: 'BOTH'
4. }) YIELD nodeId, community, communitie
   s
5. RETURN algo.asNode(nodeId).name as nam
   e, community, communities
6. ORDER BY name ASC
```

Kode 5. 33 Algoritma *Louvain Modularity* dengan Parameter *Node* Anggota

Baris ke-1 pada Kode 5.33 berfungsi untuk menjalankan algoritma *Louvain modularity* dengan parameter *node* Anggota.

```
1. CALL algo.louvain.stream('Caleg', null
   , {
2.  graph: 'huge',
3.  direction: 'BOTH'
4. }) YIELD nodeId, community, communitie
   s
5. RETURN algo.asNode(nodeId).name as nam
   e, community, communities
6. ORDER BY name ASC
```

Kode 5. 34 Algoritma *Louvain Modularity* dengan Parameter *Node* Caleg

Kode 5.34 merupakan kode untuk menjalankan algoritma *Louvain modularity* dengan parameter *node* Caleg. Hasil yang ditampilkan adalah nama *node* yang disertai dengan kode *community*. Jika dua buah *node* atau lebih memiliki kode *community* yang sama maka node tersebut berada dalam satu komunitas.

5.6 Pembuatan Visualisasi Graf

Setelah mendapatkan hasil perhitungan algoritma *centrality* dan *community detection*, langkah selanjutnya adalah membuat visualisasi dari hasil yang didapatkan agar pembaca lebih

mudah dalam membaca dan memahaminya. Visualisasi yang digunakan adalah D3js pada *web browser* yang bersifat statis.

```
1.  <!DOCTYPE html>
2.  <html>
3.      <head>
4.          <script type="text/javascript" src="h
    ttps://d3js.org/d3.v3.js""></script>
5.          <style>
6.              body{ font: Arial 12px; text-
    align: center;}
7.
8.              .link {
9.                stroke: #ccc;
10.             }
11.
12.             .node text {
13.               pointer-events: none;
14.               font: sans-serif;
15.             }
16.         </style>
17.         <link rel="stylesheet" type="text/css
    " href="main.css">
18.     </head>
19.     <body>
20.     <script type="text/javascript">
21.   //script-text
22.     </script>
23.     </body>
24. </html>
```

Kode 5. 35 *html* untuk Visualisasi D3js

Kode 5.35 merupakan kode *html* untuk menampilkan visualisasi pada *web browser* dengan D3js.

```
1.  var margin = {
2.          top: 20,
3.          bottom: 50,
4.          right: 30,
5.          left: 50
6.      };
```

Kode 5. 36 *JavaScript* untuk Mengatur *Margin* pada Visualisasi

Kode 5.36 berfungsi untuk mengatur *margin* pada visualisasi yang akan ditampilkan.

```
1. var svgElement = d3.select("body")
2.                          .append("svg").attr({
   "width": width+margin.left+margin.right, "hei
   ght": height+margin.top+margin.bottom})
3.                          .append("g")
4.                          .attr("transform","tr
   anslate("+margin.left+","+margin.top+")");
```

Kode 5. 37 JavaScript untuk Membuat *svg*

Kode 5.37 berfungsi untuk membuat *svg element* yang merupakan sebuah wadah yang digunakan untuk menggambar visualisasi.

```
1. d3.json("degree.json", function(dataset){
2.         //Extract data from dataset
3.         var nodes = dataset.nodes,
4.             links = dataset.links;
```

Kode 5. 38 *JavaScript* untuk Ekstrak Data

Baris ke-1 pada Kode 5.38 berfungsi untuk memuat data hasil perhitungan algoritma SNA dalam format json. Baris ke-3 dan ke-4 berfungsi untuk mengekstrak *nodes* dan *links* pada dataset.

```
1. var force = d3.layout.force()
2.                          .size([width, height]
   )
3.                          .nodes(nodes)
4.                          .links(links)
5.                          .gravity(0.05)
6.                          .charge(-200)
7.                          .linkDistance(200);
```

Kode 5. 39 *JavaScript* untuk Membuat *Force Layout*

Kode 5.39 berfungsi untuk membuat *force layout* yang bertujuan untuk membuat inisiasi *network graph*.

```
1. var link = svgElement.selectAll(".link")
2.                        .data(links)
3.                        .enter()
4.                        .append("line")
5.                        .attr("stroke-
   width", function(d){ return d.weight/10; })
```

Kode 5. 41 *JavaScript* untuk Membuat *Link* pada Graf

Kode 5.41 berfungsi untuk membuat *link* berdasarkan dataset pada graf yang berupa garis lurus.

```
1. var node = svgElement.selectAll(".node")
2.                        .data(nodes)
3.                        .enter()
4.                        .append("g")
5.                        .attr("class", "node")
6.                        .call(force.drag);
```

Kode 5. 40 *JavaScript* untuk Membuat pada *Node* pada Graf

Kode 5.40 berfungsi untuk membuat *node* berdasarkan dataset pada graf .

```
1. var label = node.append("text")
2.                            .attr("dx", 12)
3.                            .attr("dy", "0.35em")

4.                            .attr("font-
   size", function(d){ return d.influence*1.5>9?
   d.influence*1.5: 9; })
5.                            .text(function(d){ ret
   urn d.character; });
```

Kode 5. 42 *JavaScript* untuk Membuat *Label* pada *Node*

Kode 5.42 berfungsi untuk membuat *label* berupa *text* pada *node*. *Label* ini berdasarkan pada *label* yang sesuai dengan dataset.

```
1.  var circle = node.append("circle")
2.                              .attr("r", function(d){
     return d.influence/2>5 ? d.influence/2 : 5; })

3.                              .attr("fill", function(
    d){ return c10(d.zone*10); });
```

Kode 5. 43 *JavaScript* untuk Membuat *Circle* pada Graf

Kode 5.43 berfungsi untuk membuat lingkaran untuk menggambarkan sebuah *node* pada graf.

```
1.  force.on("tick", function(){
2.                  //Set X and Y of node
3.                  node.attr("r", function(d){ return
    d.influence; })
4.                      .attr("cx", function(d){ return
     d.x; })
5.                      .attr("cy", function(d){ return
     d.y; });
6.                  //Set X, Y of link
7.                  link.attr("x1", function(d){ return
     d.source.x; })
8.                  link.attr("y1", function(d){ return
     d.source.y; })
9.                  link.attr("x2", function(d){ return
     d.target.x; })
10.                 link.attr("y2", function(d){ return
     d.target.y; });
11.                 //Shift node a little
12.                 node.attr("transform", function(d)
    { return "translate(" + d.x + "," + d.y + ")";
    });
13.             });
14.             //Start the force layout calculation
15.             force.start();
```

Kode 5. 44 *JavaScript* untuk Menggambar Graf pada *Layout*

Kode 5.44 berfungsi untuk mengatur *nodes* dan *links* yang digunakan untuk menggambar graf pada *layout*.

Di awal sub bab ini telah disebutkan bahwa data hasil perhitungan algoritma SNA yang digunakan adalah berbentuk json. Kode 5.45 merupakan kode json yang digunakan pada visualisasi D3js.

```
1.  {
2.    "nodes": [
3.    {
4.      "character": "Ahmad Jaya Adiguna",
5.      "id": 0,
6.      "value": 30,
7.      "zone": 1
8.    },
9.  {
10.     "character": "Monalisa S.Si",
11.     "id": 1,
12.     "value": 30,
13.     "zone": 1
14.   }
15. ],
16. "links": [{
17.       "source": 0,
18.       "target": 1,
19.       "weight": 20
20.     }
21. ]
22. }
```

Kode 5. 45 Data *json* untuk Visualisasi

Pada Kode 5.45, *character* merupakan nama anggota atau caleg yang juga menjadi *label* pada *node*, id merupakan angka unik yang digunakan untuk membuat *link*, *value* merupakan nilai atau skor yang dimiliki sebuah *node* berdasarkan hasil perhitungan algoritma SNA, dan *zone* digunakan untuk membuat warna yang membedakan kelompok satu dengan kelompok yang lain. Pada bagian *links*, *source* merupakan *node* asal dan *target* merupakan *node* tujuan pada relasi, sedangkan *weight* digunakan untuk menggambarkan kekuatan relasi.

BAB VI
HASIL DAN PEMBAHASAN

Bab ini menjelaskan tentang hasil dan pembahasan dari implementasi yang telah dilakukan. Pembahasan ini meliputi nilai hasil perhitungan algoritma SNA beserta visualisasinya.

6.1 Algoritma *Centrality*

Pada algoritma *centrality*, *node* yang digunakan dalam perhitungan adalah *node* caleg, anggota, dapil, dan partai karena selain *node* tersebut kurang berpengaruh pada hasil perhitungan. Berikut merupakan analisis hasil pada perhitungan algoritma *centrality* yang meliputi *degree centrality*, *betweenness centrality*, *closeness centrality*, dan *pagerank*.

6.1.1. *Degree Centrality*

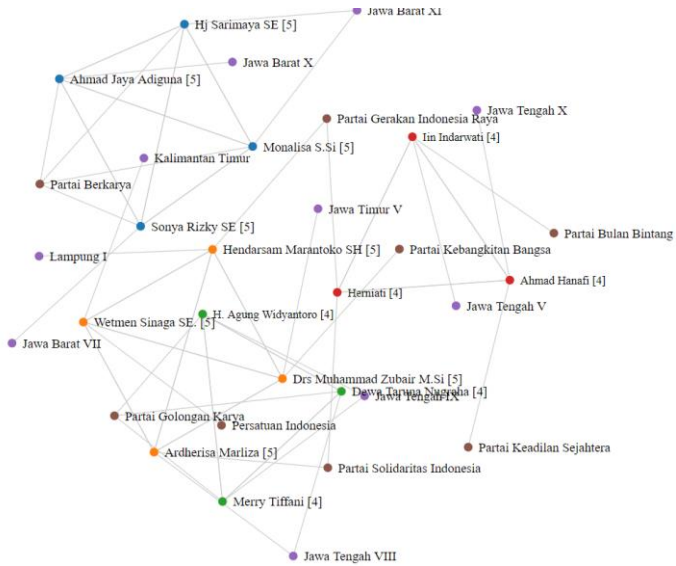
Dengan melakukan perhitungan algoritma *degree centrality* dapat mengetahui *node* mana yang memiliki relasi masuk dan keluar yang paling banyak. Hasil perhitungan algoritma ini tertera pada Tabel 6.1.

Tabel 6. 1 Hasil Perhitungan Algoritma *Degree Centrality*

Nama	Nilai
Ahmad Jaya Adiguna	5
Ardherisa Marliza	5
Hendarsam Marantoko SH	5
Monalisa S.Si	5
Drs Muhammad Zubair M.Si	5
Hj Sarimaya SE	5
Sonya Rizky SE	5
Wetmen Sinaga SE	5
H. Agung Widyantoro	4
Ahmad Hanafi	4
Merry Tiffani	4
Dewa Taruna Nugraha S.Ked	4
Iin Indarwati	4

Herniati	4

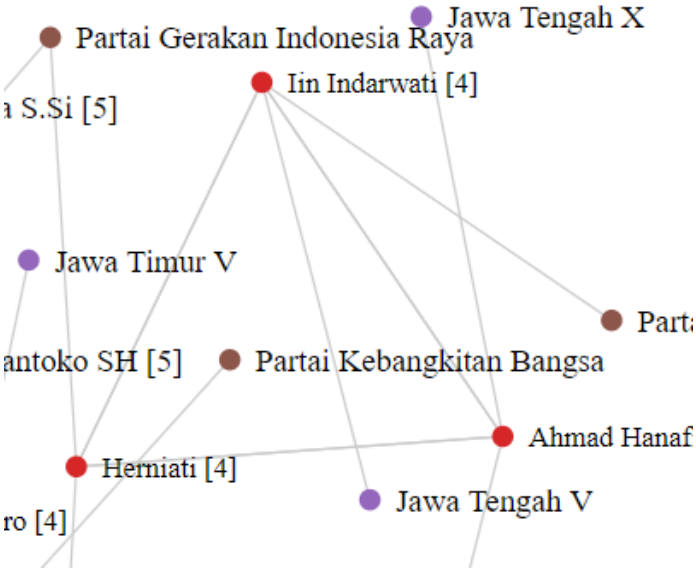
Hasil yang tertera pada Tabel 6.1 diambil berdasarkan 10 nilai tertinggi dalam perhitungan. Dari tabel tersebut menunjukkan bahwa terdapat beberapa caleg yang memiliki nilai *degree centrality* tertinggi. Gambar 6.1 menunjukkan visualisasi graf dari hasil algoritma *degree centrality*.



Gambar 6. 1 Visualisasi *Degree Centrality* [20]

Gambar 6.1 merupakan visualisasi *degree centrality* berdasarkan nama caleg dan anggota yang berada pada Tabel 6.1. Visualisasi ini tidak dilakukan pada semua *node* karena jika dilakukan maka label pada *node* tidak bisa terlihat dengan jelas. Dari visualisasi tersebut dapat diketahui bahwa secara *default node* caleg dan anggota memiliki minimal dua relasi *outgoing* yaitu relasi dengan dapil dan partai. Jika *node* caleg dan anggota memiliki nilai

degree centrality lebih dari dua maka dipastikan bahwa *node* tersebut pasti memiliki relasi kerabat dengan sesama caleg atau anggota.



Gambar 6. 2 Detail Visualisasi *Degree Centrality* [20]

Gambar 6.2 menunjukkan detail visualisasi *degree centrality*. Pada gambar tersebut dapat terlihat bahwa *node* caleg dan/atau anggota memiliki warna yang berbeda. Persamaan warna tersebut menggambarkan bahwa mereka menjalin hubungan kerabat. Berdasarkan gambar tersebut, *node* dengan label Herniati, Ahmad Hanafi, dan Iin Indarwati memiliki hubungan kekerabatan karena warna *node* mereka sama-sama berwarna merah.

6.1.2. *Betweenness Centrality*

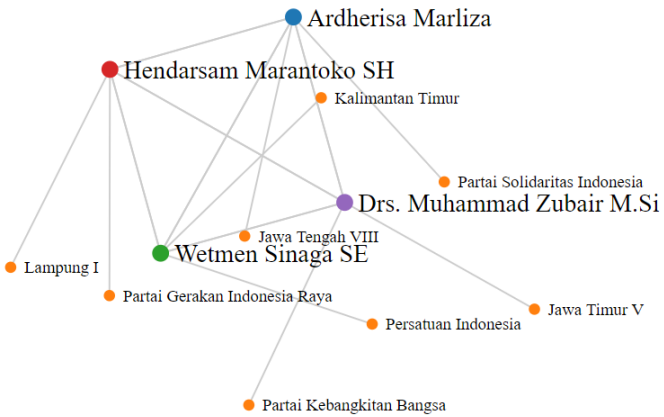
Dengan melakukan perhitungan algoritma *betweenness centrality* dapat mengetahui *node* mana yang paling sering dilewati oleh *node* lainnya atau *node* yang menjadi *shortest*

path untuk menuju ke *node* lain. Hasil perhitungan algoritma ini tertera pada Tabel 6.2.

Tabel 6. 2 Hasil Perhitungan Algoritma *Betweenness Centrality*

Nama	Nilai
Ardherisa Marliza	6.0
Hendarsam Marantoko SH	6.0
Drs Muhammad Zubair M.Si	6.0
Wetmen Sinaga SE	6.0
Ahmad Hanafi MBA	4.0
Dwi Priyo Atmojo	4.0
Herniati	4.0
Iin Indarwati	4.0
Rizki Nurdia Astuti	4.0
Sri Rayani	4.0

Hasil yang tertera pada Tabel 6.2 diambil berdasarkan 10 nilai tertinggi dalam perhitungan. Dari tabel tersebut menunjukkan bahwa caleg bernama Ardherisa Marliza, Hendarsam Marantoko SH, Drs Muhammad Zubair M.Si, dan Wetmen Sinaga memiliki nilai *betweenness centrality* tertinggi yang berarti bahwa caleg ini menjadi penghubung caleg atau anggota lain. Gambar 6.2 menunjukkan visualisasi graf dari hasil algoritma *betweenness centrality*.



Gambar 6. 3 Visualisasi *Betweenness Centrality* [20]

Gambar 6.3 menjelaskan tentang visualisasi *betweenness centrality*. Keempat *node* tersebut memiliki nilai *betweenness centrality* paling tinggi karena setiap *node* dapat menjadi jalur terpendek dari *node* caleg lainnya untuk menuju *node* dapil dan *node* partai.

6.1.3. *Closeness Centrality*

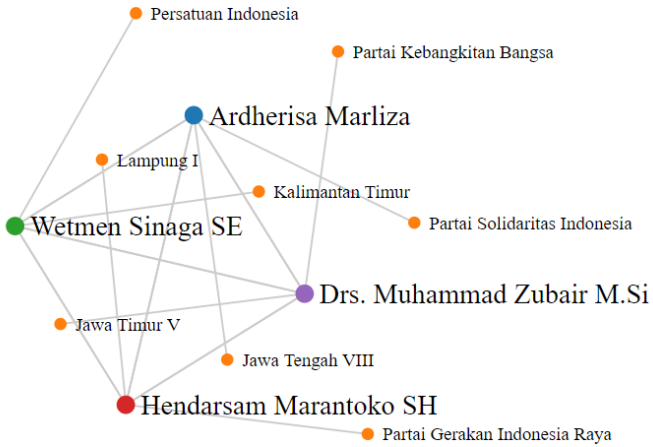
Dengan melakukan perhitungan algoritma *closeness centrality* dapat mengetahui *node* mana yang dapat menyebarkan informasi luas paling cepat kepada satu jaringan graf. Hasil perhitungan algoritma ini tertera pada Tabel 6.3.

Tabel 6. 3 Hasil Perhitungan Algoritma *Closeness Centrality*

Nama	Nilai
Hendarsam Marantoko SH	0.28861980395240344
Wetmen Sinaga SE	0.2885592998270531
Ardherisa Marliza	0.28774496995035276
Drs Muhammad Zubair M.Si	0.28724958263772954
Dwi Priyo Atmojo	0.2805891046221271
Rizki Nurdia Astuti	0.27976220720491846

Sri Rayani	0.2788836549663172
Herniati	0.27686428320008044
Ahmad Hanafi MBA	0.2756583558626214
Dr. Ir. Hetifah Sjaifudian	0.2747367895813582

Hasil yang tertera pada Tabel 6.3 diambil berdasarkan 10 nilai tertinggi dalam perhitungan. Dari tabel tersebut menunjukkan bahwa caleg bernama Hendarsam Marantoko SH memiliki nilai *closeness centrality* tertinggi yang berarti bahwa caleg tersebut dapat menyebarkan informasi dengan cepat kepada caleg lain. Gambar 6.4 menunjukkan visualisasi graf dari hasil algoritma *closness centrality*.



Gambar 6. 4 Visualisasi *Closeness Centrality* [20]

Gambar 6.4 menunjukkan visualisasi *closeness centrality*, yang sebenarnya mirip dengan visualisasi *betweenness centrality*, dimana *node* Hendarsam Marantoko SH memiliki nilai tertinggi diantara *node* caleg dan anggota lainnya. Hal ini dikarenakan *node* Hendarsam Marantoko SH memiliki rata-rata *shortest path* paling paling sedikit, kemudian diikuti oleh *node* caleg yang bernama Wetmen Sinaga SE, Ardherisa Marliza, dan Drs Muhammad Zubair

M.Si. Dengan memiliki nilai *closeness centrality* tertinggi, *node* Hendarsam Marantoko SH dapat menjadi *influencer* terhadap *network* tersebut karena mampu menyebarkan informasi dengan cepat.

6.1.4. *PageRank*

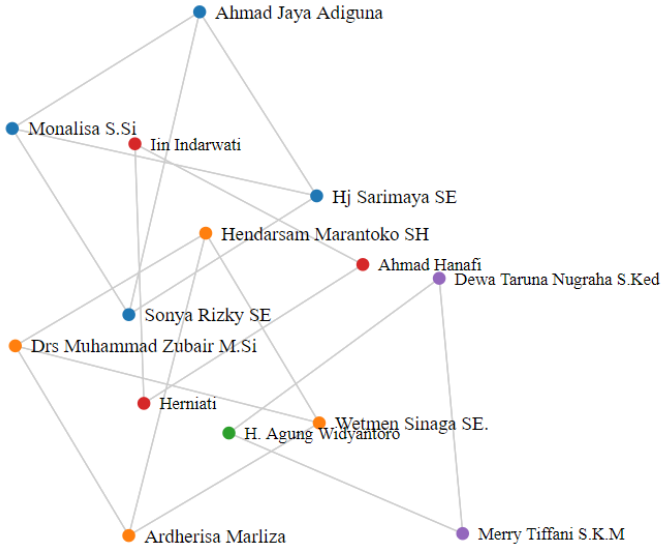
Dengan melakukan perhitungan algoritma *pageRank* dapat mengetahui *node* mana yang paling sering dikunjungi oleh *node* lain. Pada dasarnya, hasil algoritma ini menunjukkan bahwa *node* Islam memiliki nilai *pageRank* tertinggi, namun tujuan pada penelitian ini adalah ingin menunjukkan caleg atau anggota yang berpengaruh dalam Pemilu sehingga hasil yang ditunjukkan berdasarkan perspektif *node* Caleg dan *node* Anggota. Hasil perhitungan algoritma ini tertera pada Tabel 6.4.

Tabel 6. 4 Hasil Perhitungan Algoritma *PageRank*

Nama	Nilai
Ahmad Jaya Adiguna	0.3061222189572619
Ardherisa Marliza	0.3061222189572619
Hendarsam Marantoko SH	0.3061222189572619
Monalisa S.Si	0.3061222189572619
Drs Muhammad Zubair M.Si	0.3061222189572619
Hj Sarimaya SE	0.3061222189572619
Sonya Rizky SE	0.3061222189572619
Wetmen Sinaga SE	0.3061222189572619
H. Agung Widyantoro	0.26086956285193363
Ahmad Hanafi MBA	0.26086956285193363

Hasil yang tertera pada Tabel 6.4 diambil berdasarkan 10 nilai tertinggi dalam perhitungan. Dari tabel tersebut menunjukkan bahwa caleg pada baris 1 sampai 8 memiliki nilai *pagerank* tertinggi. Hal ini menunjukkan bahwa *node* mereka paling banyak dikunjungi atau memiliki relasi yang masuk terbanyak daripada *node* lain. Gambar 6.5

menunjukkan visualisasi graf dari hasil algoritma *pagerank*.



Gambar 6. 5 Visualisasi *PageRank* [20]

Gambar 6.5 menunjukkan visuasliasai *pagerank* berdasarkan *node* yang ada pada Tabel 6.4. Dari visualisasi tersebut dapat diketahui bahwa terdapat hasil algoritma *pagerank* pada kasus ini didasarkan pada anggota atau caleg yang memiliki hubungan kerabat dengan sesamanya. Hal tersebut ditunjukkan dengan adanya beberapa *node* yang membentuk sebuah kelompok. Semakin banyak hubungan kerabat yang dimiliki, maka semakin besar nilai *pagerank* sebuah *node*.

6.2 Algoritma *Community Detection*

Berikut merupakan analisis hasil pada perhitungan algoritma *community detection* yang meliputi *triangle count and clustering coefficient*, *strongly connected components*,

connected components, *label propagation*, dan *Louvain modularity*.

6.2.1. *Triangle Count and Clustering Coefficient*

Perhitungan pada algoritma *triangle count and clustering coefficient* dibagi menjadi dua, yaitu:

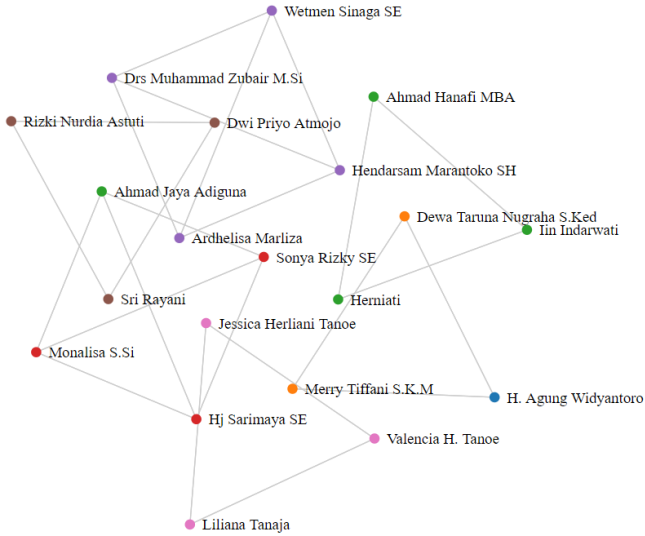
1. *Triangle count*
   Dengan melakukan perhitungan algoritma *triangle count* dapat mengetahui tiga *node* yang saling berhubungan sepeti membentuk segitiga. Hasil perhitungan algoritma ini tertera pada Tabel 6.5.

Tabel 6. 5 Hasil Perhitungan Algoritma *Triangle Count*

Node A	Node B	Node C
H. Agung Widyantoro	Dewa Taruna Nugraha S.Ked	Merry Tiffani S.K.M
Ahmad Hanafi MBA	Herniati	Iin Indarwati
Ahmad Jaya Adiguna	Monalisa S.Si	Hj Sarimaya SE
Ahmad Jaya Adiguna	Monalisa S.Si	Sonya Rizky SE
Ahmad Jaya Adiguna	Hj Sarimaya SE	Sonya Rizky SE
Ardherisa Marliza	Hendarsam Marantoko SH	Drs Muhammad Zubair M.Si
Ardherisa Marliza	Hendarsam Marantoko SH	Wetmen Sinaga SE
Ardherisa Marliza	Drs Muhammad Zubair M.Si	Wetmen Sinaga SE
Dwi Priyo Atmojo	Rizki Nurdia Astuti	Sri Rayani

Hendarsam Marantoko SH	Drs Muhammad Zubair M.Si	Wetmen Sinaga SE
Jessica Herliani Tanoe	Liliana Tanaja	Valencia H. Tanoe
Monalisa S.Si	Hj Sarimaya SE	Sonya Rizky SE

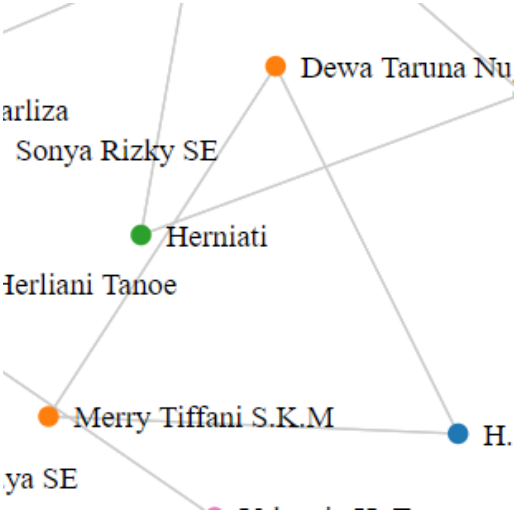
Berdasarkan Tabel 6.5 terdapat 12 pasang *node* caleg yang relasinya berbentuk seperti segitiga. Gambar 6.6 menunjukkan visualisasi graf dari hasil algoritma *triangle count*.



Gambar 6. 6 Visualisasi *Triangle Count* [20]

Berdasarkan visualisasi *triangle count* yang ditunjukkan pada Gambar 6.6 dapat diketahui bahwa terdapat 6 kelompok yang terdeteksi dengan algoritma *triangle count* yang ditunjukkan dengan warna *node* masing-masing. Namun, jika dilihat dengan seksama, *triangle* yang terbentuk dari *node* Dewa Taruna, Merry Tiffani, dan H. Agung Widayantoro sedikit berbeda dari *triangle* lain. Pada *triangle* ini, *node* H. Agung

Widyantoro memiliki warna yang berbeda dengan dua *node* yang lainnya. Hal ini dikarenakan H. Agung Widyantoro merupakan anggota atau caleg terpilih sedangkan Dewa Taruna dan Merry Tiffani merupakan caleg yang kalah dalam pemilu. Hal ini membuktikan bahwa terdapat hubungan kerabat antara anggota dan caleg pada Pemilu 2019. Untuk lebih jelasnya dapat dilihat pada detail visualisasi yang ditunjukkan pada Gambar 6.7.



Gambar 6. 7 Detail Visualisasi *Triangle Count* [20]

2. *Local clustering*

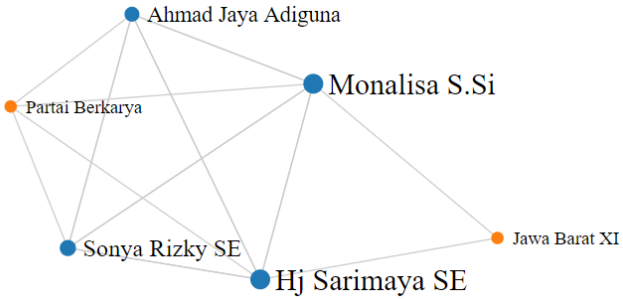
   Dengan melakukan perhitungan algoritma *local clustering* dapat mengetahui *node* pada graf yang cenderung mengelompok bersama. Hasil perhitungan algoritma ini tertera pada Tabel 6.6.

Tabel 6. 6 Hasil Perhitungan Algoritma *Local Clustering*

Nama	Nilai
Monalisa S.Si	0.7
Hj Sarimaya SE	0.7

Abdul Hakim Achmad Aituarauw	0.6666666666666666
Drs. H Abdul Wahab Sinambela Med	0.6666666666666666
Andi Rusnadi	0.6666666666666666
Dewi Oktavianti	0.6666666666666666
Erma Kusumawati	0.6666666666666666
Erna Fonataba	0.6666666666666666
Dr Frida Medina Hayuputri S.Psi	0.6666666666666666
H. Abbas Abubakar SH	0.6666666666666666

Hasil yang tertera pada Tabel 6.6 diambil berdasarkan 10 nilai tertinggi dalam perhitungan. Dari tabel tersebut menunjukkan bahwa caleg bernama Monalisa S.Si dan Hj Sarimaya SE memiliki nilai *coefficient* yang tertinggi. Hal ini berarti *neighbor* mereka juga berhubungan satu sama lain. Gambar 6.8 menunjukkan visualisasi graf dari hasil algoritma *local clustering*.



Gambar 6. 8 Visualisasi *Local Clustering* [20]

Pada Gambar 6.8 terlihat bahwa *node* Monalisa S.Si dan *node* Hj Sarimaya SE memiliki ukuran yang lebih besar dibanding dengan *node* yang lain. Warna biru pada *node* menggambarkan bahwa *node* tersebut merupakan caleg, sedangkan warna kuning pada *node* menggambarkan bahwa *node* tersebut merupakan entitas dapil dan partai. *Node* Monalisa S.Si dan Hj

Sarimaya SE memiliki nilai *local clustering* tertinggi karena *node-node* tersebut memiliki *neighbor* yang saling terhubung. Kedua *node* tersebut memiliki kerabat caleg, dapil, dan partai yang sama.

6.2.2. *Strongly Connected Components*

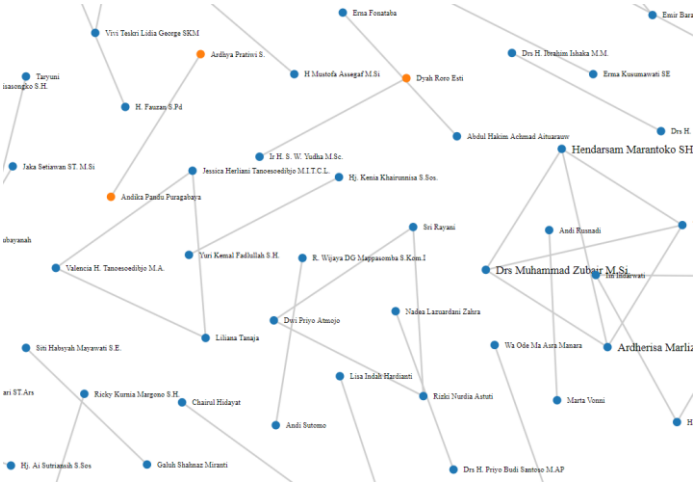
Dengan melakukan perhitungan algoritma *strongly connected components* dapat mengetahui *node* yang memiliki keterkaitan yang kuat satu sama lain. Hasil perhitungan algoritma ini tertera pada Tabel 6.7.

Tabel 6. 7 Hasil Perhitungan Algoritma *Strongly Connected Components*

Partisi	Nama
788	“Ahmad Jaya Adiguna”, “Monalisa S.Si”, “Hj Sarimaya SE”, “Sonya Rizky SE”
1046	“Ardherisa Marliza”, “Hendarsam Marantoko SH”, “Drs Muhammad Zubair M.Si”, “Wetmen Sinaga SE”
91	“H. Agung Widyantoro”, “Dewa Taruna Nugraha S.Ked”, “Merry Tiffani S.K.M”
784	“Ahmad Hanafi MBA”, “Herniati”, “Iin Indarwati”
1615	“Dwi Priyo Atmojo”, “Rizki Nurdia Astuti”, “Sri Rayani”
3242	“Jessica Herliani Tanoe”, “Liliana Tanaja”, “Valencia H. Tanoe”
57	“Dr. H. R. Achmad Dimyati Natakusumah”, “Risya Azzahra Rahimah Natakusumah”
58	“Dyah Roro Esti”, “Ir H. S.W. Yudha M.Sc.”
66	“H. Alex Noerdin”, “Hj. Lury Elza Alex Noerdin”
109	“Hanna Gayatri”, “dr Azzzahrazade MKM”

Hasil yang tertera pada Tabel 6.7 diambil berdasarkan 10 teratas partisi yang diurutkan berdasarkan jumlah *node*

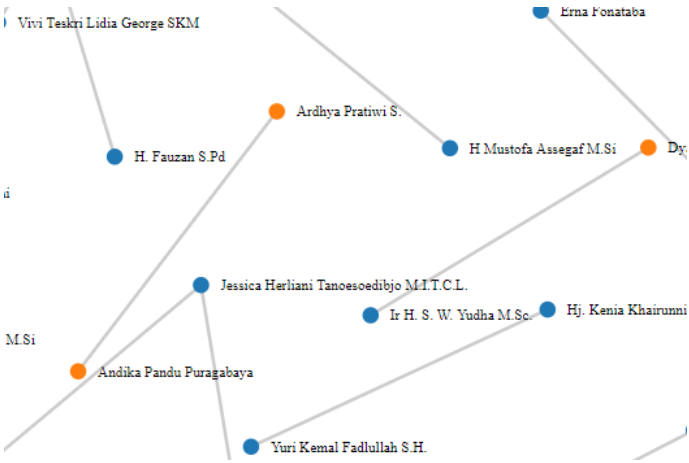
yang berada dalam partisi. Dari tabel tersebut menunjukkan bahwa partisi 788 dan 1046 memiliki *node* terbanyak. Setiap partisi menunjukkan bahwa *node-node* tersebut berada dalam satu kelompok. Gambar 6.8 menunjukkan visualisasi graf dari hasil algoritma *strongly connected components*.



Gambar 6. 9 Visualisasi *Strongly Connected Components* [20]

Pada Gambar 6.9 menunjukkan banyak partisi yang terdiri dari 2-4 *node*. Partisi tersebut menggambarkan hubungan kerabat antar *node*. Total jumlah partisi yang terbentuk berdasarkan algoritma ini adalah sebanyak 60 partisi. Hal ini mengindikasikan bahwa terdapat 60 keluarga yang mengikuti pemilihan caleg pada tahun 2019 baik dengan partai pengusung yang sama maupun tidak.

Pada visualisasi tersebut, terdapat dua jenis warna untuk membedakan *node*. Warna biru menunjukkan bahwa *node* tersebut merupakan caleg yang kalah dalam pemilihan, sedangkan warna oranye merupakan caleg yang menang dalam pemilihan atau yang telah menjadi anggota DPR.



Gambar 6. 10 Detail Visualisasi *Strongly Connected Components* [20]

Pada Gambar 6.10 menunjukkan bahwa partisi yang terbentuk dapat berupa *node* sesama caleg, *node* sesama anggota, atau bahkan *node* antar caleg dan anggota. Namun hanya terdapat satu partisi yang semua *node* nya terdiri dari anggota, yaitu *node* Andika Pandu Puragabaya dan Ardhya Pratiwi.

6.2.3. *Connected Components*

Dengan melakukan perhitungan algoritma *connected components* dapat mengetahui *node* yang terhubung baik secara langsung maupun tidak langsung dalam graf. Algoritma disebut juga dengan *weakly connected components* yang mana merupakan kebalikan dari algoritma *strongly connected components*. Hasil perhitungan algoritma ini tertera pada Tabel 6.8.

Tabel 6. 8 Hasil Perhitungan Algoritma *Connected Components*

SetId	Nama
0	"A. A. Bagus Adhi Mahendra Putra", "Prof. Dr. Hendrawan Supratikno", "H. Syaifullah Tamliha", "Ir. H. Ahmad Riza Patria", "Hillary

	Brigitta Lasut", "Drs. I Made Urip", "I Ketut Kariyasa Adnyana", "Diah Pitaloka", "Ir. H. Achmad Hafisz Tohir", "Fadhlullah", dst.

Tabel 6.8 menunjukkan bahwa hanya terdapat satu hasil yang mana berisi semua *node* dalam jaringan. Hal ini dikarenakan semua *node* dapat saling menjangkau melalui perantara *node* lain. SetId bernilai 0 karena *index* kolom dimulai dari 0. Hasil algoritma ini tidak perlu divisualisasikan karena bentuknya sama dengan *network* yang sebenarnya.

6.2.4. *Label Propagation*

Dengan melakukan perhitungan algoritma *label propagration* dapat mengetahui *node* yang membentuk suatu kelompok atau komunitas tersendiri dalam graf. Pada perhitungan algoritma kali ini menggunakan dua parameter yang berbeda, yaitu Anggota dan Caleg. Hasil perhitungan untuk parameter Anggota tertera pada Tabel 6.9.

Tabel 6. 9 Hasil Perhitungan Algoritma *Label Propagation* dengan Parameter Anggota

Label	Nama
329	"Andika Pandu Puragabaya", "Ardhya Pratiwi S."

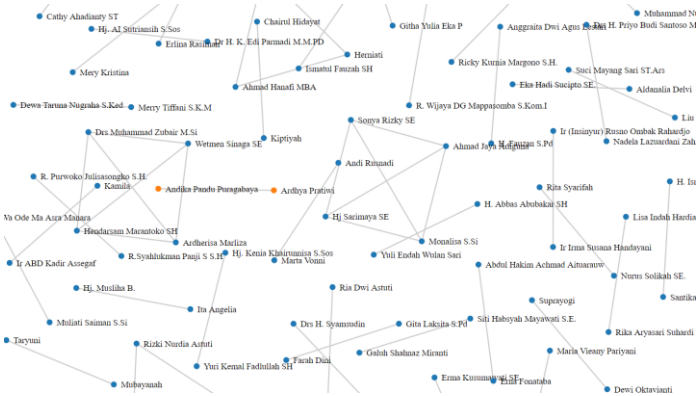
Pada Tabel 6.9 menunjukkan bahwa terdapat dua anggota yang memiliki label yang sama sehingga mereka dapat dikatakan satu kelompok. Sedangkan untuk hasil perhitungan algoritma *label propagation* dengan parameter Caleg tertera pada Tabel 6.10.

Tabel 6. 10 Hasil Perhitungan Algoritma *Label Propagation* dengan Parameter Caleg

Label	Nama

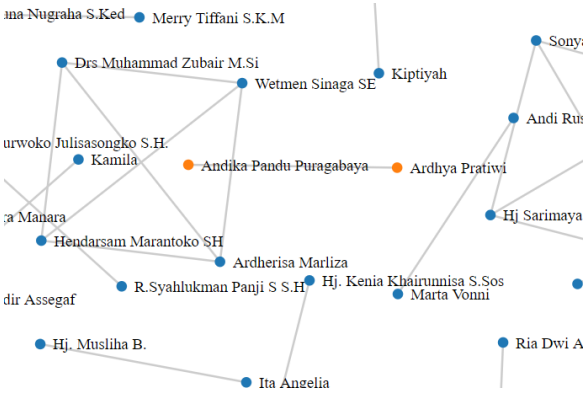
3810	"Ahmad Jaya Adiguna", "Monalisa S.Si", "Hj Sarimaya Se", "Sonya Rizky Se"
2723	"Ardherisa Marliza", "Hendarsam Marantoko Sh", "Drs Muhammad Zubair M.Si.", "Wetmen Sinaga Se."
2779	"Ahmad Hanafi Mba", "Herniati", "Iin Indarwati"
4576	"Dwi Priyo Atmojo", "Rizki Nurdia Astuti", "Sri Rayani"
3440	"Jessica Herliani Tanoesoedibjo M.I.T.C.L.", "Liliana Tanaja", "Valencia H. Tanoesoedibjo M.A."
3314	"Ir Abd Kadir Assegaf", "Kamila"
1798	"Abdul Hakim Achmad Aituarauw", "Erna Fonataba"
1871	"Abdul Manap", "Faisol"
1793	"Drs. H Abdul Wahab Sinambela Med", "Erma Kusmawati Se"
1682	"Aldanalia Delvi", "Eka Hadi Sucipto Se."

Hasil yang tertera pada Tabel 6.10 diambil berdasarkan 10 teratas yang diurutkan berdasarkan jumlah *node* yang memiliki label yang sama. Gambar 6.11 menunjukkan visualisasi graf dari hasil algoritma *label propagation*.



Gambar 6. 11 Visualisasi *Label Propagation* [20]

Gambar 6.11 menunjukkan visualisasi *label propagation* yang mana sekilas terlihat seperti visualisasi *strongly connected components*. Perbedaan dari visualisasi *label propagation* dan *strongly connected components* adalah jika pada *strongly connected components* terdapat partisi yang terdiri dari *node* caleg dan anggota sedangkan pada *label propagation* tidak ada. Fungsi warna pada visualisasi ini sam dengan visualisasi sebelumnya, yaitu warna biru menggambarkan *node* caleg dan warna oranye menggambarkan *node* anggota.



Gambar 6. 12 Detail Visualisasi *Label Propagation* [20]

Gambar 6.12 merupakan detail viusalisasi yang menunjukkan bahwa meskipun mayoritas *node* merupakan caleg, masih terdapat dua *node* anggota yang memiliki label yang sama.

6.2.5. *Louvain Modularity*

Dengan melakukan perhitungan algoritma *louvain modularity* dapat mengetahui *community node* pada graf. Jika nilai *community*-nya sama, maka *node* tersebut berada dalam satu komunitas yang sama. Pada perhitungan algoritma ini juga menggunakan dua parameter yang berbeda seperti yang dilakukan sebelumnya, yaitu parameter Anggota dan Caleg. Hasil perhitungan untuk parameter Anggota tertera pada Tabel 6.11.

Tabel 6. 11 Hasil Perhitungan Algoritma *Louvain Modularity* dengan Parameter Anggota

Nama	Community	Communities
Andika Pandu Puragabaya	329	Null
Ardhya Pratiwi S.	329	Null

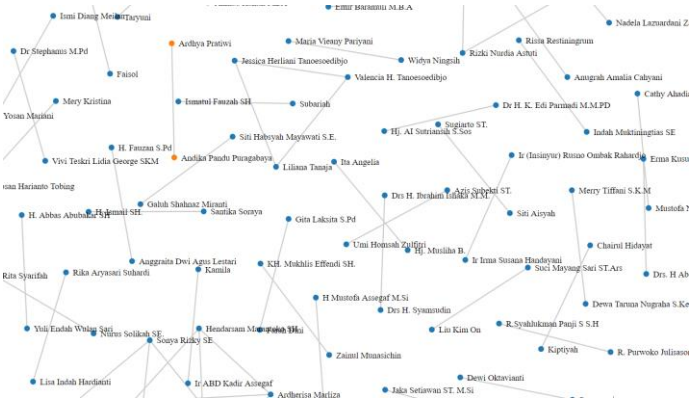
Pada Tabel 6.11 menunjukkan bahwa terdapat dua anggota yang memiliki *community* yang sama dan nilai *null* pada kolom *communities* merupakan nilai *default*. Sedangkan hasil perhitungan untuk parameter Caleg tertera pada Tabel 6.12.

Tabel 6. 12 Hasil Perhitungan Algoritma *Louvain Modularity* dengan Parameter Caleg

Nama	Community	Communities
Aldanalia Delvi	1089	Null
Eka Hadi Sucipto SE	1089	Null
Dr drg Armelia Sari M.Kes	1142	Null

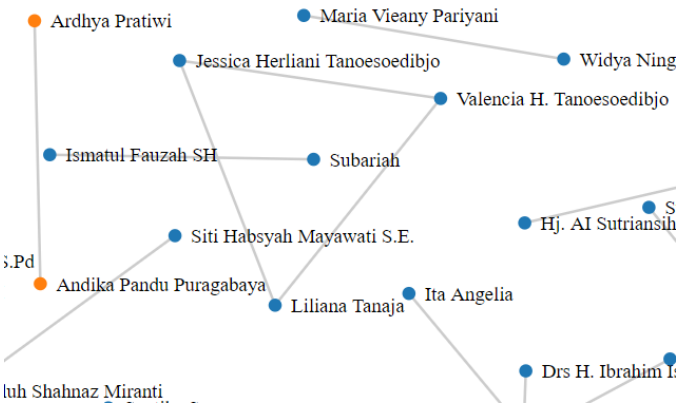
Emir Baramuli M.B.A.	1142	Null
Drs. H Abdul Wahab Sinambela Med	1200	Null
Erma Kusmawati SE	1200	Null
Abdul Hakim Achmad Aituarauw	1205	Null
Erna Fonataba	1205	Null
Abdul Maap	1278	Null
Faisol	1278	Null

Hasil yang tertera pada Tabel 6.12 diambil berdarkan 10 id *community* yang diurutkan dari id terkecil. Terdapat 50 *community* berbeda yang setiap *community*-nya terdiri dari 2-4 caleg. Gambar 6.13 menunjukkan visualisasi graf dari hasil algoritma *louvain modularity*.



Gambar 6. 13 Visualisasi *Louvain Modularity* [20]

Visualisasi *louvain modularity* yang ditunjukkan pada Gambar 6.13 sama dengan visualisasi *label propagation* karena tujuan dari kedua algoritma tersebut mirip. Id *community* yang dihasilkan pada algoritma ini sama dengan id label yang dihasilkan pada algoritma *label propagation* sehingga hasil visualisasinya sama.



Gambar 6. 14 Detail Visualisasi *Louvain Modularity* [20]

Detail visualisasi *louvain modularity* yang ditunjukkan pada Gambar 6.14 juga sama dengan detail visualisasi *label propagation*, yaitu hanya terdapat satu komunitas pada anggota DPR yang terdiri dari Andika Pandu Puragabaya dan Ardhya Pratiwi.

Halaman ini sengaja dikosongkan

BAB VII
KESIMPULAN DAN SARAN

Bab kesimpulan dan saran membahas tentang kesimpulan dari penelitian SNA terhadap Pemilu 2019 yang telah dilakukan dan saran yang diusulkan untuk penelitian selanjutnya.

7.1. Kesimpulan

Kesimpulan yang telah didapatkan berdasarkan hasil analisis terhadap caleg dan anggota DPR pada Pemilu 2019 adalah:

1. Caleg yang bernama Ahmad Jaya Adiguna, Ardherisa Marliza, Hendarsam Marantoko SH, Monalisa S.Si, Drs Muhammad Zubair M.Si, Hj Sarimaya SE, Sonya Rizky SE, dan Wetmen Sinaga SE memiliki nilai *degree centrality* 10 dimana merupakan nilai tertinggi. Hal tersebut menunjukkan bahwa mereka memiliki relasi yang lebih banyak daripada caleg dan anggota yang lain.
2. Caleg yang bernama Ardherisa Marliza, Hendarsam Marantoko SH, Wetmen Sinaga SE, dan Muhammad Zubair memiliki nilai *betweenness centrality* 6,0 yang merupakan nilai tertinggi. Hal ini menunjukkan bahwa caleg tersebut dapat menjadi alternatif yang efektif bagi caleg disekitarnya untuk menyebarkan informasi.
3. Caleg yang bernama Hendarsam Marantoko SH memiliki nilai *closeness centrality* 0,2886 yang merupakan nilai tertinggi di antara caleg dan anggota lainnya. Hal ini menunjukkan bahwa caleg tersebut mudah dan cepat dalam menyebarkan informasi pada jejaring sosialnya.
4. Caleg yang bernama Ahmad Jaya Adiguna, Ardherisa Marliza, Hendarsam Marantoko SH, Monalisa S.Si, Drs Muhammad Zubair M.Si, Hj Sarimaya SE, Sonya Rizky SE, dan Wetmen Sinaga SE memiliki nilai *pagerank* 0,3061 dimana merupakan nilai tertinggi. Hal

tersebut dapat diartikan bahwa mereka merupakan caleg yang paling populer.

5. Pada algoritma *community detection* berhasil mengidentifikasi 60 kelompok yang terdiri dari 1 kelompok yang semua anggotanya merupakan anggota DPR, 10 kelompok yang anggotanya terdiri dari caleg dan anggota DPR, dan 49 kelompok yang anggotanya terdiri dari caleg yang tidak menang dalam pemilihan.
6. Berdasarkan hasil algoritma *community detection*, caleg yang memiliki hubungan kerabat dengan caleg yang lain memiliki peluang sebanyak 9,375% untuk menang atau dengan kata lain cenderung tidak terpilih atau kalah dalam pemilihan.
7. Pada kasus ini, semua algoritma *community detection* cocok untuk diimplementasikan kecuali *connected components* karena algoritma tersebut tidak membantu untuk mendeteksi sebuah komunitas.

7.2. Saran

Beberapa saran yang dapat diberikan untuk penelitian selanjutnya antara lain:

1. Perlu dilakukan pembersihan data untuk menghapus *missing value* pada baris biodata caleg dan anggota.
2. Perlu dilakukan *string matching* untuk mengetahui caleg atau anggota yang memiliki alamat yang sama untuk menentukan jenis relasinya.
3. Pada penelitian selanjutnya dapat menggunakan *Graph Data Science* (GDS) pada Neo4j untuk pengolahan data yang kompleks dengan hasil yang lebih akurat.
4. Visualisasi hasil algoritma SNA dapat menggunakan Neo4j Bloom yang merupakan salah satu fitur Neo4j agar tidak berpindah-pidah platform dan lebih cepat.

DAFTAR PUSTAKA

[1] A. R. Prasetia, “PENGARUH POLITIK IDENTITAS MELALUI MEDIA SOSIAL TERHADAP GENERASI MILENIAL DAN PELAKSANAAN PEMILU,” in *Conference On Communication and News Media Studies*, 2019, vol. 1, p. 21.

[2] A. I. Hadiana and W. Witanti, “Analisis Jejaring Sosial Menggunakan Social Network Analysis untuk Membantu Social CRM bagi UMKM di Cimahi,” 2017.

[3] C. Hsu and H. W. Park, “Mapping online social networks of Korean politicians,” *Gov. Inf. Q.*, vol. 29, no. 2, pp. 169–181, 2012.

[4] H. Y. Yoon and H. W. Park, “Strategies affecting Twitter-based networking pattern of South Korean politicians: Social network analysis and exponential random graph model,” *Qual. Quant.*, vol. 48, no. 1, pp. 409–423, 2014.

[5] N. Dokoohaki, F. Zikou, D. Gillblad, and M. Matskin, “Predicting swedish elections with twitter: A case for stochastic link structure analysis,” in *Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2015*, 2015, pp. 1269–1276.

[6] Y. Nam, Y.-O. Lee, and H. W. Park, “Measuring web ecology by Facebook, Twitter, blogs and online news: 2012 general election in South Korea,” *Qual. Quant.*, vol. 49, no. 2, pp. 675–689, 2015.

[7] M. Needham and A. E. Hodler, *Graph Algorithms: Practical Examples in Apache Spark and Neo4j*. O’Reilly Media, 2019.

[8] N. A. Rakhmawati, R. Alfarhizi, and I. Hafidz, “Penerapan Social Network Analysis dengan Menggunakan Metode Sociomatrix pada Akun

Instagram Siswa SMA di Surabaya," *SISTEMASI*, 2020, doi: 10.32520/stmsi.v9i2.807.

[9] F. M. Clemente, F. M. L. Martins, D. Kalamaras, J. Oliveira, P. Oliveira, and R. S. Mendes, "The social network analysis of Switzerland football team on FIFA World Cup 2014," *J. Phys. Educ. Sport*, vol. 15, no. 1, p. 136, 2015.

[10] "Undang-Undang No. 7 Tahun 2017 Tentang Pemilu," 2017.

[11] F. Rahman, I. F. Ulfah, and O. R. Danar, "Pemilu 2014: Politisi Pencari Kursi di Senayan," *J. Transform.*, vol. 4, no. 2, pp. 1–13, 2018.

[12] T. V Udapure, R. D. Kale, and R. C. Dharmik, "Study of web crawler and its different types," *IOSR J. Comput. Eng.*, vol. 16, no. 1, pp. 1–5, 2014.

[13] J. Lee and C. J. Bonk, "Social network analysis of peer relationships and online interactions in a blended class using blogs," *Internet High. Educ.*, vol. 28, pp. 35–44, 2016.

[14] B.-H. Yoon, S.-K. Kim, and S.-Y. Kim, "Use of graph database for the integration of heterogeneous biological data," *Genomics Inform.*, vol. 15, no. 1, p. 19, 2017.

[15] R. kumar Kaliyar, "Graph databases: A survey," in *International Conference on Computing, Communication & Automation*, 2015, pp. 785–790.

[16] F. Bao and J. Chen, "Visual framework for big data in d3. js," in *2014 Ieee Workshop on Electronics, Computer and Applications*, 2014, pp. 47–50.

[17] F. Holzschuher and R. Peinl, "Performance of graph query languages: comparison of cypher, gremlin and native access in Neo4j," in *Proceedings of the Joint EDBT/ICDT 2013 Workshops*, 2013, pp. 195–204.

[18] L. Ciechanowski, D. Jemielniak, and P. A. Gloor,

"TUTORIAL: AI research without coding: The art of fighting without fighting: Data science for qualitative researchers," *J. Bus. Res.*, vol. 117, pp. 322–330, 2020.

[19] M. S. Z. Rivzi, "Game of Thrones Force Directed Graph," 2019. https://bl.ocks.org/mohdsanadzakirizvi/6fc325042ce110e1afc1a7124d087130 (accessed Nov. 15, 2019).

[20] K. Mufidah, "Visualisasi Social Network Analysis," 2020. https://zenodo.org/record/3928860 (accessed Jul. 03, 2020).

Halaman ini sengaja dikosongkan

LAMPIRAN

Link source code: https://zenodo.org/record/3928860

Halaman ini sengaja dikosongkan

BIODATA PENULIS

Penulis lahir di Sidoarjo pada tanggal 1 Mei 1999. Penulis merupakan anak kedua dari 2 bersaudara. Penulis telah menempuh pendidikan formal, yaitu: SD Negeri Krian 4, SMP Negeri 1 Krian, dan SMA Negeri 1 Krian.

Pada tahun 2016, penulis melanjutkan pendidikan ke jenjang S1 melalui jalur SNMPTN di Departemen Sistem Informasi, Fakultas Teknologi Elektro dan Informatika Cerdas, Institut Teknologi Sepuluh Nopember (ITS) Surabaya. Selama menjadi mahasiswa, penulis aktif mengikuti beberapa kegiatan kemahasiswaan, antara lain sebagai pengurus HMS pada periode 2017/2018 dan 2018/2019 di Departemen Human Resource Development (HRD) serta menjadi Koordinator Sponsorship Information Systems Expo tahun 2018. Selain itu, dalam hal akademik penulis juga menjadi asisten dosen pada mata kuliah Evaluasi dan Audit Teknologi Informasi.

Pada tahun ke-4 perkuliahan, penulis memiliki ketertarikan dalam mempelajari pengolahan data dan mengambil bidang minat Akuisisi Data dan Diseminasi Informasi (ADDI). Penulis dapat dihubungi melalui email: karimamufidah@gmail.com.

Halaman ini sengaja dikosongkan