Prosiding
ANNUAL RESEARCH SEMINAR 2016
6 Desember 2016, Vol 2 No. 1

ISBN : 979-587-626-0 | UNSRI http://ars.ilkom.unsri.ac.id

Semantic Suffix Tree Clustering untuk Peningkatan Hasil Document Retriever pada Sistem Tanya Jawab Bahasa Indonesia

Dininta Isnurthina
Jurusan Teknik Informatika
Fakultas Ilmu Komputer, Universitas Sriwijaya
Paembang, Indonesia
dini.dininta@gmail.com

***Abstract*— Penelitian ini mengusulkan metode *Semantic Suffix Tree Clustering* untuk mengelompokkan dokumen pada komponen *document retriever* sebuah Sistem Tanya Jawab Bahasa Indonesia. Tujuan pengelompokan dokumen pada komponen *document retriever* adalah untuk menghasilkan Sistem Tanya Jawab yang akurat. Sistem memiliki lima komponen, yaitu *preprocessing*, *question analyzer*, *document retriever, document clusterer*, dan *answer finder*.**

Keywords— document clusterer; document retriever; sistem tanya jawab; semantic suffix tree clustering;

I. LATAR BELAKANG

Analisis *cluster* atau *clustering* adalah teknik pengelompokan objek data hanya berbasis pada informasi yang ditemukan dalam data yang mendeskripsikan objek-objek serta relasi diantaranya [5]. Tujuan yang ingin dicapai dari sebuah proses *clustering* adalah objek-objek yang ada dalam satu grup, saling berelasi atau mirip antara satu dan lainnya, tetapi berbeda dengan objek-objek dalam grup lain. Semakin tinggi tingkat kemiripan tiap objek dalam suatu grup dan semakin tinggi tingkat perbedaan tiap grup, maka dapat dinyatakan semakin baik hasil *clustering*-nya. Pada beberapa kasus, *clustering* hanya berperan sebagai titik awal yang dapat dimanfaatkan untuk berbagai tujuan, contohnya peringkasan data. *Clustering* sejak lama telah berperan penting dalam berbagai bidang: psikologi, biologi, statistik, pengenalan pola, *document retrieval*, *machine learning*, dan *data mining*.

Sistem Tanya Jawab adalah sebuah kajian dalam bidang Pemrosesan Bahasa Alami yang secara otomatis memberikan jawaban untuk pertanyaan berbahasa alami [3]. Sistem Tanya Jawab dapat menggunakan basis data atau koleksi dokumen (local atau web) sebagai sumber untuk jawabannya. Sebuah sistem tanya jawab biasanya terdiri dari tiga komponen utama: *question analyzer*, *document retriever*, dan *answer finder* [6]. *Question analyzer* bertujuan untuk mengklasifikasikan pertanyaan sesuai dengan *Expected Answer Type* (EAT) serta mengekstrak kata kunci dari pertanyaan tersebut. *Document retriever* adalah komponen yang memiliki fungsi yang sama dengan mesin pencari, yaitu mengumpulkan informasi yang diinginkan. Komponen *answer finder* mencari jawaban dari kandidat dokumen atau paragraf yang telah ditemukan sebelumnya.

Penelitian mengenai Sistem Tanya Jawab Bahasa Indonesia telah banyak dilakukan. Salah satunya penelitian tentang Sistem Tanya Jawab Bahasa Indonesia untuk pertanyaan *factoid* yang menerapkan algoritma *Support Vector Machine* pada komponen *question analyzer* dan *answer finder*. Pada komponen *passage retriever* menggunakan *inverse document frequency* (idf) [8]. Hasil percobaan pada *passage retriever* menunjukkan bahwa penggunaan idf lebih cocok daripada *term frequency-inverse document frequency* (tf-idf). Pada tahun 2011 dilakukan penelitan tentang Sistem Tanya Jawab untuk pertanyaan *non-factoid* (definisi, alasan, dan metode) Bahasa Indonesia [2]. Pada komponen *question analyzer* analisa pertanyaan didasarkan pada kata tanya yang digunakan. *Document retriever* meggunakan tf-idf dan perhitungan *cosine similarity*. *Answer finder* menggunakan aturan *surface expression*. Dengan menggunakan 90 pertanyaan yang dikumpulkan dari 10 orang Indonesia dan 61 dokumen sumber, diperoleh nilai MRR 0.7689, 0.5925, dan 0.5704 untuk tipe pertanyaan definisi, alasan, dan metode secara berurutan. Penelitian selanjutnya adalah pengembangan Sistem Tanya Jawab Bahasa Indonesia untuk pertanyaan *factoid* dan *non-factoid* dengan menerapkan komponen *clustering* dokumen dalam *document retriever* [4]. Metode yang digunakan adalah *Suffix Tree Clustering* [9]. Penelitian ini menghasilkan tingkat akurasi jawaban yang diberikan sebesar 83%.

Dalam paper ini akan diusulkan metode *Semantic Suffix Tree Clustering* [1] untuk meningkatkan hasil *retrieval* pada komponen *document retriever* dengan penambahan komponen *document clusterer* sehingga menghasilkan Sistem Tanya Jawab yang akurat. Bagian 2 menjelaskan analisis *preprocessing* kalimat. Bagian 3 menjelaskan Sistem Tanya

ISBN : 979-587-626-0 | UNSRI http://ars.ilkom.unsri.ac.id

Jawab. Bagian 4 menjelaskan metode *Semantic Suffix Tree Clustering*. Bagian 5 menjelaskan arsitektur perangkat lunak. Dan bagian 6 berisi kesimpulan dari paper ini.

II. PREPROCESSING KALIMAT

Preprocessing adalah proses awal pengolahan data sebelum proses utama dilakukan [4]. Pada penelitian ini akan dilakukan lima tahap *preprocessing* yaitu *case folding*, *tokenizing*, *stop words removal*, dan *stemming*.

2.1. Case Folding

Case folding adalah tahap penyamarataan karakter pada kalimat pertanyaan masukan dan konten dokumen berupa perubahan huruf besar menjadi huruf kecil dan penghapusan karakter selain huruf. Contoh: "Apakah yang dimaksud dengan sel?" menjadi "apakah yang dimaksud dengan sel".

2.2. Tokenizing

Tokenizing didefinisikan sebagai upaya untuk mengenali batas-batas antara kata-kata. Contoh "Apakah yang dimaksud dengan sel" akan menjadi sekumpulan kata berikut: "apakah", "yang", "dimaksud", "dengan", dan "sel".

2.3. Stop Word Removal

Stop Words adalah kata-kata yang sering muncul dan tidak mengandung makna penting di kalimat masukan, seperti "yang" atau "ke", karenanya dapat dihapus dari masukan untuk meningkatkan kinerja pencarian. Menghapus *stop words* dapat dilakukan dengan menggunakan daftar yang berisi semua *stop words*. Contoh penghapusan *stop words*: "apakah yang dimaksud dengan sel" menjadi "apakah dimaksud dengan sel".

2.4. Stemming

Stemming merupakan suatu proses untuk menemukan kata dasar dari sebuah kata. Dengan menghilangkan semua imbuhan (*affixes*) baik yang terdiri dari awalan (*prefixes*), sisipan (*infixes*), akhiran (*suffixes*) dan *confixes* (kombinasi dari awalan dan akhiran) pada kata turunan. *Stemming* digunakan untuk mengganti bentuk dari suatu kata menjadi kata dasar dari kata tersebut yang sesuai dengan struktur morfologi sebuah bahasa. Contoh: "apakah dimaksud dengan sel" menjadi "apa maksud dengan sel". Terdapat beberapa algoritma yang dikembangkan untuk proses *stemming* bahasa Indonesia, dan algoritma yang diterapkan pada paper ini adalah algoritma Nazief Adriani [7].

2.5. Part-of-Speech Tagging

Part-of-Speech tagging adalah proses penandaan kata pada suatu kalimat sebagai sebuah *part-of-speech*, berdasarkan definisi dan konteks kata tersebut pada kalimat. Contoh: "apakah yang dimaksud dengan sel" ditandai menjadi "apakah/WP yang/SC dimaksud/VBI dengan/IN sel/NN". Hasil *Part-of-Speech tagging* ini akan berperan sebagai penanda semantic untuk diterapkan pada metode *Semantic Suffix Tree Clustering*. Proses *Part-of-Speech tagging* pada penelitian ini dilakukan menggunakan *library* NLP_ITB.

III. SISTEM TANYA JAWAB

Sebuah sistem tanya jawab biasanya terdiri dari tiga komponen utama: *question analyzer*, *document retriever*, dan *answer finder* [6].

A. Analisis proses Question Analyzer

Proses question analyzer berfungsi untuk menganalisis pertanyaan yang diberikan oleh pengguna ke dalam sistem. Tujuannya adalah untuk mengetahui kata tanya yang digunakan di dalam kalimat pertanyaan, kata-kata yang dapat digunakan untuk menggambarkan inti pertanyaan (kata kunci), dan tipe jawaban yang diharapkan.

TABLE I. CONTOH HASIL PROSES *QUESTION ANALYZER* [7]

Pertanyaan: Apakah itu sel? Hasil *Question Analyzer*: – Tipe jawaban yang dimaksud: definisi – Kata kunci: sel
Pertanyaan: Bagaimanakah cara tercepat dalam memenuhi kebutuhan energi untuk kontraksi otot? Hasil *Question Analyzer*: – Tipe jawaban yang dimaksud: metode – Kata kunci: otot, relaksasi, energi,fosfat,asam laktat
Pertanyaan: Mengapa komposisi protoplasma dalam setiap sel makhluk hidup berbeda-beda? Hasil *Question Analyzer*: – Tipe jawaban yang dimaksud: alasan – Kata kunci: protoplasma, sel, zat

B. Analisis proses Document Retriever

Proses *document retriever* bertugas untuk mengumpulkan dokumen-dokumen yang mengandung kata kunci. Pada penelitian ini, pengumpulan dokumen hanya dilakukan dengan mengecek keberadaan kata kunci di dalam dokumen. Apabila dokumen tersebut mengandung kata kunci, maka dokumen tersebut dianggap memiliki jawaban yang dibutuhkan. Setelah mendapat dokumen yang mengandung kata kunci, proses selanjutnya adalah mengelompokkan hasil dokumen menggunakan metode *Semantic Suffix Tree Clustering*.

ISBN : 979-587-626-0 | UNSRI http://ars.ilkom.unsri.ac.id

TABLE I. CONTOH HASIL PROSES *DOCUMENT RETRIEVER*

Pertanyaan: Apakah itu sel?
Hasil *Question Analyzer*:
- Tipe jawaban yang dimaksud: definisi
- Kata kunci: sel

Hasil *Document Retriever*:

1. Judul: Struktur dan Fungsi Sel.
   Isi:
   Pada bab berikut, kita akan mempelajari struktur dan fungsi sel sekaligus komponen kimiawi yang menyusunnya. Selain itu, kita juga akan mengidentifikasi pelbagai organel sel penyusun tubuh tumbuhan dan hewan. Sementara itu, mekanisme transpor yang terjadi pada membran sel juga akan kita pelajari di bagian akhir bab ini. Dengan mempelajari bab ini, kalian diharapkan mampu mengetahui struktur dan fungsi sel, baik sel hewan maupun sel tumbuhan. Sekaligus juga kalian diharapkan mampu membedakan struktur sel hewan dan struktur sel tumbuhan tersebut.
   ….

2. Judul: Organisasi Kehidupan.
   Isi:
   Struktur organisasi kehidupan dimulai dari unit kehidupan kecil yang disebut sel. Sel-sel yang sama struktur dan fungsinya bergabung membentuk jaringan. Beberapa jenis jaringan membentuk struktur yang disebut organ. Beberapa organ yang saling berkaitan membentuk sistem organ, dan selanjutnya beberapa sistem organ itu menyusun terbentuknya organisme.
   ….

3. Judul: Pembelahan Sel.
   Isi:
   Kalian mungkin sering melihat orang yang secara fisik mirip dengan orang lainnya dalam satu keluarga. Coba kalian perhatikan gambar di atas. Adakah persamaan yang kalian temukan antara orang tua dengan anaknya?. Pepatah mengatakan "Buah jatuh tidak jauh dari pohonnya". Pepatah ini dapat diartikan, sifat seorang anak tidak akan jauh dari sifat kedua orang tuanya. Dalam hal ini, kita hanya membicarakan sifat fisik, bukan sifat secara total. Kemiripan antara orang tua dengan anaknya merupakan salah satu bentuk pewarisan sifat. Tahapan awal yang menyebabkan terjadinya pewarisan sifat adalah proses pembelahan sel. Kalian tentunya ingin mengetahui mekanisme pembelahan sel ini bukan?. Nah, untuk itu, kalian perlu mempelajari materi pada bab ini.
   ….

4. Judul: Struktur dan Fungsi Jaringan pada Tumbuhan.
   Isi:
   Budidaya tanaman hias cenderung silih berganti. Mulai kamboja, adenium, hingga anthurium. Tak terkecuali, anggrek pun pernah mengalami masa jayanya. Memang, selain dipajang sebagai hiasan, tanaman ini dapat pula dijadikan komoditas bisnis yang sangat menguntungkan. Terlebih lagi tanaman anggrek dapat tumbuh di beragam tempat, misalnya di tanah, menempel pada tanaman inang, ataupun pada bebatuan. Sehingga sifat uniknya pun dapat bertambah. Budidaya tanaman anggrek tidak begitu sulit. Dapat dilakukan secara konvensional maupun metode kultur jaringan. Metode terakhir banyak dilirik karena cukup efektif. Maka, kalian perlu mengerti, bagaimana kultur jaringan dilakukan?
   ….

5. ….

C. Analisis proses Answer Finder

Proses *answer finder* berfungsi untuk menemukan jawaban atas pertanyaan yang diberikan. Pada paper ini, akan digunakan metode *surface expression* untuk menjawab pertanyaan *factoid* dan *non-factoid* sekaligus. Metode *surface expression* untuk menjawab pertanyaan *non-factoid* telah diteliti sebelumnya [7].

TABLE II. CONTOH HASIL PROSES *ANSWER FINDER* [7]

Pertanyaan: Apakah itu sel? Hasil *Answer Finder*: sel dapat diartikan sebagai unit terkecil penyusun makhluk hidup dan sebagai tempat berlangsungnya aktivitas kehidupan.
Pertanyaan: Bagaimanakah cara tercepat dalam memenuhi kebutuhan energi untuk kontraksi otot?

ISBN : 979-587-626-0 | UNSRI http://ars.ilkom.unsri.ac.id

Hasil *Answer Finder*:
Saat otot berelaksasi/istirahat, kreatin mengikat fosfat sehingga terbentuklah fosfat kreatin yang banyak mengandung energi. Pelepasan fosfat disertai dengan pelepasan energi kerapkali terjadi pada pertengahan pergeseran filamen. Karena itu, proses ini merupakan cara tercepat dalam pemenuhan kebutuhan energi untuk kontraksi otot. Energi yang diperoleh ini tidak bisa digunakan secara langsung, namun harus diubah terlebih dahulu. Untuk proses fermentasi, energi dihasilkan dari glukosa yang diuraikan menjadi asam laktat.

Pertanyaan: Mengapa Komposisi protoplasma dalam setiap sel makhluk hidup berbeda-beda?

Hasil *Answer Finder*:
Komposisi protoplasma dalam setiap sel makhluk hidup berbeda. Sebab, protoplasma tersusun atas berbagai campuran zat. Oleh karena itu, kemungkinan protoplasma yang menyusun sel penyusun organ tubuh tertentu berbeda dengan sel penyusun organ tubuh yang lain.

IV. SEMANTIC SUFFIX TREE CLUSTERING

Janruang dan Guha mengembangkan metode *Semantic Suffix Tree Clustering* (SSTC) yang memanfaatkan hasil kemiripan semantik menggunakan sinonim kata yang diperoleh dari basis data WordNet dan dihitung ukuran kemiripan antar pasangan katanya [1]. Sebuah algoritma *clustering* teks harus memanfaatkan makna dari kata-kata yang ada untuk meningkatkan tingkat akurasi *clustering*-nya. Konsep dasar algoritma *Semantic Suffix Tree Clustering* adalah mengelompokkan dokumen-dokumen yang mirip secara semantik ke dalam *cluster* yang sama.

4.1. Semantic Suffix Tree

Semantic suffix tree adalah struktur data yang merupakan pengembangan dari *suffix tree* [1]. *Semantic suffix tree* pada metode ini menggunakan kesamaan semantik dan kecocokan string sebagai kondisi untuk membentuk sebuah *suffix tree*. Kesamaan semantik adalah pengukuran yang mengambil kumpulan sinonim dari kamus dalam basis data. Persamaan kesamaan semantik ditunjukkan sebagai berikut.

$$SemSim(w_a, w_b) = 1 \text{ if } |synset(w_a) \cap synset(w_b)| \geq 1$$
$$SemSim(w_a, w_b) = 0 \quad otherwise \qquad (1)$$

Persamaan ini menggunakan asosiasi dari kelompok sinonim w_a dan w_b untuk menghitung kesamaannya. Contoh, kelompok sinonim dari kata *eat* = {*feed*, *consume*, *corrode*} dan kelompok sinonim kata *ate* = {*eat*, *feed*, *consume*}. Kedua kelompok sinonim memiliki kata *feed* dan *consume*, yang artinya lebih dari satu kata. Maka, ukuran *SemSim(eat, ate)* sama dengan 1. Gambar 1 menunjukkan algoritma pembentukan *semantic suffix tree*.

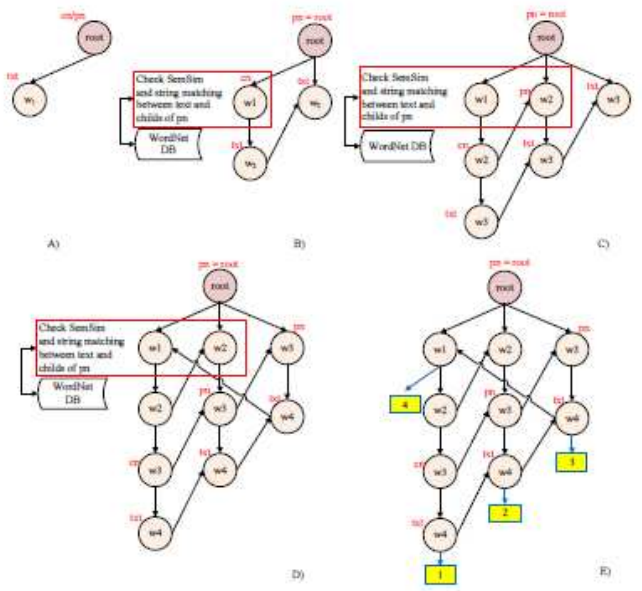
```
Algorithm1:Construct Semantic Suffix Tree
1: input <-- set of String
2: output <-- semantic suffix tree
3: for each word (txt)
4:   if root is empty then{
5:     create a new node
6:     and update position
7:   }else{
8:     add a new node into cn
9:     do until pn = root {
10:      if SemSim = 0 && no match then{
11:        add a new node into pn
12:        and update suffix link
13:        update position
14:      }else{update suffix link
15:            and update position }
16:    }
17:  }
18: }
```

Gambar 1. Algoritma pembentukan *semantic suffix tree*.

Jika diberikan sebuah string dengan urutan 1, 2, 3, dan 4 sebagai $S = \{ w_1, w_2, w_3, w_4 \}$, buat simpul pertama sebagai *cn* (*current node*) sesuai urutan yang diarahkan lewat simpul *root*, seperti ditunjukkan pada Gambar 2-A. Selanjutnya, kata kedua dibentuk dan dihubungkan ke simpul *cn* (penelusuran *on-depth*), serta dijadikan sebagai simpul *cn*. Kemudian terbentuklah *sub-tree* yang baru (penelusuran *on-breadth*) karena kata kedua tidak cocok terhadap semantik maupun anak kata dari simpul *pn* seperti ditunjukkan Gambar 2-B.

Dengan cara yang sama, kata ketiga dan keempat dibangkitkan, membentuk *sub-tree* sehingga simpul *pn* menjadi simpul *root* yang menyerupai tangga, ditunjukkan pada Gambar 2-C dan Gambar 2-D. Akhirnya, posisi semua kata berubah pada simpul daun seperti yang ditunjukkan pada Gambar 1-E. Contoh ini menunjukkan karakteristik *semantic suffix tree* yang bergantung pada urutan kata. Karakteristik ini terbatas pada panjang string.

ISBN : 979-587-626-0 | UNSRI http://ars.ilkom.unsri.ac.id



Gambar 2. Contoh pembentukan *semantic suffix tree* berdasarkan Algoritma 1 dan string w_1, w_2, w_3, *dan* w_4. Gambar 2-A menunjukkan langkah inisialisasi dimana *root* masih kosong. Gambar 2-B dan 2-C menunjukkan pembentukan *semantic suffix tree* bila kesamaan semantiknya sama dengan 0, sehingga tidak ada kecocokan. Gambar 2-D menunjukkan bagaimana merubah penghubung *suffix* jika kesamaan smantik sama dengan 1 sehingga ada kecocokan. Gambar 2-E menunjukkan bagaimana merubah posisi kata bila itu adalah kata yang terakhir.

4.2. Semantic Suffix Tree Clustering

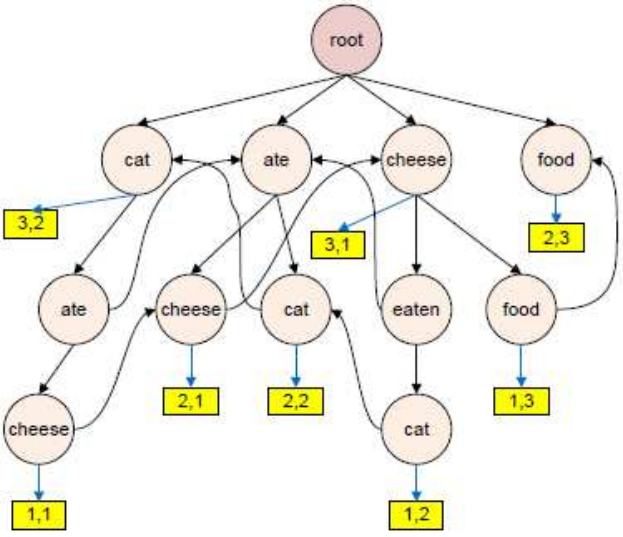
Metode *Semantic Suffix Tree Clustering* mengelompokkan dokumen-dokumen yang sama secara semantik ke dalam *cluster* yang sama. Contoh, jika dokumen *A* berisi *cat ate cheese* dan dokumen B berisi *cheese was eaten by a cat*, maka kedua dokumen A dan B harus dikelompokkan ke dalam *cluster* yang sama karena keduanya memiliki konsep yang sama. Metode SSTC terdiri dari tiga fase sebagai berikut:

```
Algorithm 2: Main algorithm of SSTC
1: input<--set of snippets
2: output<--set of final cluster
3://Phase1:preprocessing & construct tree
4: for each snippets
5:    extract the sentences
6:    for each sentence
7:      detected Subject,Verb, and Object
8:      construction of SSTC(Algorithm1)
9:    }
10: }
11://Phase2:Tree pruning(algorithm3)
12:   2.1 tree pruning step
13:   2.2 compact tree step
14://Phase3:Identify cluster(algorithm4)
15:   3.1 keep base cluster
16:   3.2 finding final cluster
```

Gambar 3. Algoritma *Semantic Suffix Tree Clustering*.

Fase 1: *Preprocessing* dan pembentukan *semantic suffix tree*

Pada tahap *preprocessing* dapat ditentukan panjang *suffix* maksimum untuk *semantic suffix tree*. Metode SSTC hanya menggunakan Subjek, Kata Kerja, dan Objek untuk membentuk *semantic suffix tree*. Sebagai contoh, diberikan tiga buah dokumen D_1 = {*cat, ate, cheese*}, D_2 = {*cheese, was, eaten, by, cat*}, dan D_3 = {*cheese, is, food*}, setelah masuk ke tahap *preprocessing* berubah menjadi D_1 = {*cat, ate, cheese*}, D_2 = {*cheese, eaten, cat*}, dan D_3 = {*cheese, food*}. Tahap ini dapat mengurangi jumlah simpul tanpa kehilangan maknanya. Hasil *semantic suffix tree*-nya dapat dilihat pada Gambar 4.



Gambar 4. Ilustrasi *semantic suffix tree* untuk tiga dokumen D_1 = {*cat, ate, cheese*}, D_2 = {*cheese, was, eaten, by, cat*}, dan D_3 = {*cheese, is, food*}.

Fase 2: Pemangkasan *tree*

Sebuah *semantic suffix tree* mengembalikan *tree* yang besar akibat penelusuran *on-depth* dan *on-breadth*. Hal ini menyebabkan *tree* sulit dieksekusi. Pemangkasan *tree* adalah proses untuk mengurangi jumlah simpul sambil tetap mempertahankan makna tiap simpul dan *sub-tree* itu sendiri.

Fase pemangkasan *tree* melakukan penelusuran dari kiri ke kanan menggunakan metode penelusuran *depth-first*. Salah satu langkahnya adalah dengan memindahkan atau menghapus sebuah cabang (pada baris 3 – 24) lalu merapatkan *tree*-nya (pada baris 27 – 28) ketika langkah pemangkasan *tree* selesai. Algoritma pemangkasan *tree* ditunjukkan pada Gambar 5.

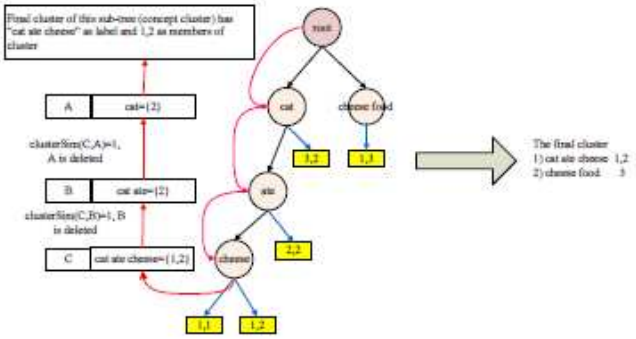
ISBN : 979-587-626-0 | UNSRI http://ars.ilkom.unsri.ac.id

```
Algorithm 3: Tree pruning algorithm
1: assign gn=root
2: for each Ti
3:  if Ti is subset Tj then Ti is deleted
4:  else{
5:     assign pn node
6:     for each pn's child node{
7:       assign cn
8:      for each cn's child node{
9:       if cn has suffix link then {
10:       assign sn
11:       case gn!=sn {
12:        if sn's childs related pn||ccn
13:        then move sn to replace cn
14:        else{move sn's documents to cn
15:             and move csn into cn}
16:        }
17:        case gn==sn {
18:         move cn's documents into pn
19:         and delete cn branch }
20:        }
21:       gn=pn, pn=cn, cn = ccn
22:       }
23:     }
24:  }
25:}//end of tree pruning step
26://compact tree
27: if pn's childs==1&documents==null
28: then pn and cn are combined
```

Gambar 5. Algoritma pemangkasan *tree*.

Fase 3: Identifikasi *cluster*

Gambar 6 menunjukkan hasil pemangkasan *tree* yang kemudian ditelusuri secara *postorder* untuk menghitung *cluster* dan labelnya. Struktur *semantic suffix tree* dibuat simpul per simpul dari kiri ke kanan karena *clustering* akan mengarah ke dalam untuk mempertahankan frasa *prefix* dari sekumpulan dokumen yang ada agar terhindar dari *cluster* yang tidak penting. Hasil labelnya akan lebih mudah dibaca karena label dalam bentuk frasa lebih mudah dibaca dibanding hanya berbentuk satu kata.



Gambar 6. Ilustrasi langkah identifikasi *cluster* yang menggunakan penelusuran *postorder* untuk menghitung kesamaan *cluster*.

```
Algorithm 4: Identify cluster
1: assign gn=root;
2: for each sub-tree{
3:   assign pn node
4:   collect into CanCluster
5:   if pn has child node {
6:     //use stack to keep right node
7:     //and work until stack is empty
8:     for each child of pn node
9:      keep cluster&label to CanCluster
10:      if cn does not child node{
11:           compute ClusSim
12:           filter the useless cluster
13:      }
14:     }
15:   }
16:  transfer CanCluster to final_cluster
17:  and clear CanCluster
18: }
19:return final_cluster
```

Gambar 7. Algoritma identifikasi *cluster*.

Pengukuran kesamaan *cluster* dihitung dengan persamaan ClusSim sebagai berikut.

$$ClusSim(C_a, C_b) = 1 \text{ if } |C_a \cap C_b| = |C_a|, C_a \text{ is deleted}$$
$$\text{or } |C_a \cap C_b| = |C_b|, C_b \text{ is deleted}$$
$$SemSim(C_a, C_b) = 0 \quad \text{otherwise} \qquad (2)$$

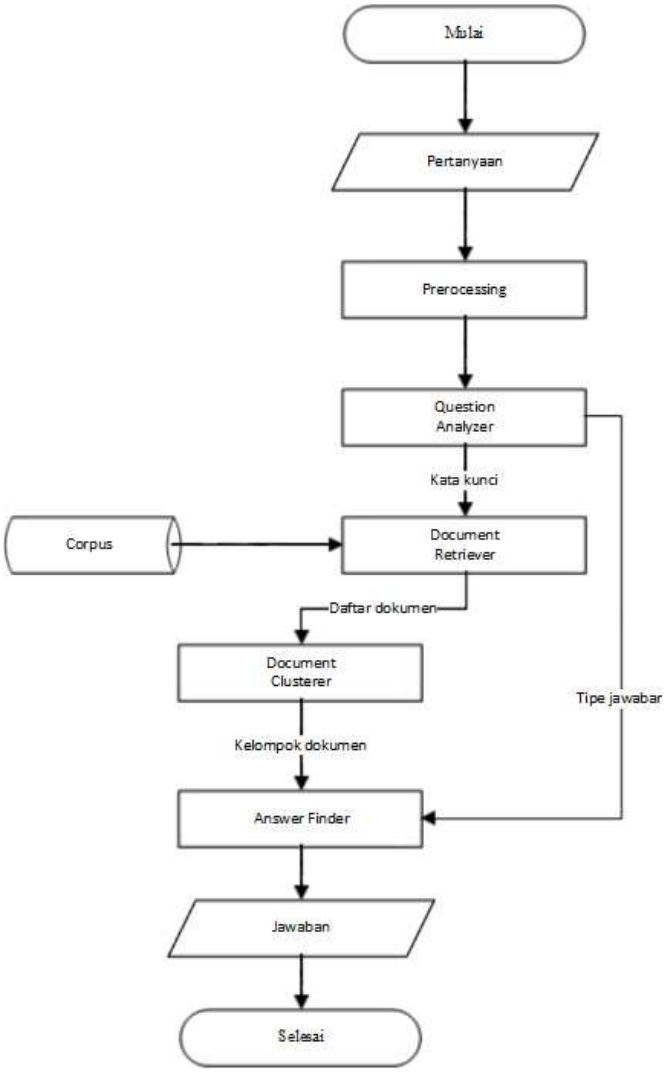
Sebagai contoh, diberikan tiga *cluster cat* = {2}, *cat ate* = {2}, dan *cat ate cheese* = {1,2}. Setelah memeriksa kesamaan *cluster* dan menghapus *cluster* yang tidak berguna, hasil akhir *cluster*-nya adalah *cat ate cheese* = {1,2}. Persamaan ini dapat mengurangi *cluster* yang tidak berguna dan mengembalikan *cluster* yang lebih spesifik. Walaupun dokumen nomor 2 tidak

ISBN : 979-587-626-0 | UNSRI http://ars.ilkom.unsri.ac.id

memiliki frasa *cat ate cheese*, tapi maknanya tetap *cat ate cheese*.

Hasil akhir dari contoh ini, D_1 = {*cat, ate, cheese*}, D_2 = {*cheese, was, eaten, by, cat*}, dan D_3 = {*cheese, is, food*} yang ditunjukkan pada Gambar 6 adalah *cat ate cheese* = {1,2} dan *cheese food* = {3}. *Cluster* yang berlabel adalah *cat ate cheese* adan anggotanya adalah D_1 dan D_2. Dari hasil akhir, metode SSTC dapat mengelompokkan dokumen yang memiliki kesamaan semantic karena dapat mengelompokkan D_1 dan D_2 ke sebuah *cluster* dan memisahkan D_3 ke *cluster* yang lain.

V. ARSITEKTUR PERANGKAT LUNAK



Gambar 8. Alur proses umum perangkat lunak.

Gambar 8 diatas menjelaskan proses umum perangkat lunak yang akan dikembangkan pada penelitian ini. Proses dimulai dari masukan kalimat pertanyaan bahasa Indonesia dengan Ejaan Yang Disempurnakan, selanjutnya masukan diolah pada *preprocessing*. Hasil *preprocessing* dimasukkan ke komponen *Question Analyzer* untuk diambil tipe jawaban dan kata kuncinya. Kata kunci digunakan sebagai queri untuk proses *Document Retriever*. Daftar dokumen yang dihasilkan komponen *Document Retriever* kemudian dikelompokkan berdasarkan kesamaan semantiknya dengan metode *Semantic Suffix Tree Clustering*. Hasil kelompok dokumen dijadikan acuan oleh komponen *Answer Finder* untuk mendapatkan jawaban teratas dibantu dengan tipe jawaban yang telah dihasilkan sebelumnya oleh *Question Analyzer*. Kalimat jawaban pada urutan teratas yang ditemukan oleh *Answer Finder* akan ditampilkan sebagai jawaban yang tepat untuk pertanyaan yang dimasukkan.

VI. KESIMPULAN

Penglompokan dokumen menggunakan metode *Semantic Suffix Tree Clustering* bertujuan untuk meingkatkan hasil *clustering* pada komponen *document retriever* sebuah Sistem Tanya Jawab. Diharapkan metode *Semantic Suffix Tree Clustering* mampu menghasilkan sebuah Sistem Tanya Jawab Bahasa Indonesia yang akurat.

REFERENSI

[1] Janruang, J. & Guha, S. 2011. Semantic Suffix Tree Clustering. First IRAST International Conference on Data Engineering and Internet Technology (DEIT). (*references*)

[2] Yusliani, N. & Purwarianti, A. 2011. Sistem *Question Answering* Bahasa Indonesia Untuk Pertanyaan *Non-Factoid*. Jurnal Ilmu Komputer dan Informasi, Volume 4, Nomor 1.

[3] Zulen, A. A., & Purwarianti, A. 2011. Study and Implementation of Monolingual on Indonesian Question Answering System for Factoid and Non-Factoid Question. In *PACLIC* (pp. 622-631).

[4] Rahmansyah, A. 2015. Implementasi Suffix Tree Clustering pada Sistem Tanya Jawab Bahasa Indonesia untuk Pertanyaan Factoid dan Non-Factoid. Bachelor of Informatics thesis, Informatics, Department of Computer Science, Universitas Sriwijaya, Indralaya, Sumatera Selatan.

[5] Tan, P., Steinbach, M., & Kumar, V. 2005. Introduction to Data Mining. New Jersey: Pearson.

[6] Harabagiu, S. M., Pasca, M. A., & Maiorano, S. J. 2000. Experiments with Open-Domain Textual Question Answering. Department of Computer Science and Engineering, Southern Methodist University, Dallas.

[7] Nazief, B. A., & Adriani, M. 1996. Confix Stripping: Approach to Stemming Algorithm for Bahasa Indonesia. Internal publication, Faculty of Computer Science, University of Indonesia, Depok, Jakarta.

[8] Purwarianti, A., Tsuchiya, M., & Nakagawa, S. 2007. A Machine Learning Approach for Indonesian Question Answering System. In *Artificial Intelligence and Applications* (pp. 573-578).

[9] Zamir, O., & Etzioni, O. 1998. Web Document Clustering: A Feasibility Demonstration. In *Proceedings of the 21st Annual International ACM SIGIR Conference On Research And Development In Information Retrieval* (pp. 45-46). ACM.