

**Membangun sebuah Combinatory Categorical  
Grammar (CCG) Supertagger berbasis  
Maximum Entropy untuk Bahasa Indonesia**

**Proposal Tugas Akhir**

**Kelas TA NLP**

**Wisnu Adi Nurcahyo**

**NIM: 1301160479**



**Program Studi Sarjana Informatika**

**Fakultas Informatika**

**Universitas Telkom**

**Bandung**

**2019**

## **Lembar Persetujuan**

**Membangun sebuah Combinatory Categorical Grammar (CCG)  
Supertagger berbasis Maximum Entropy untuk Bahasa Indonesia**

***Building a Maximum Entropy based Combinatory Categorical  
Grammar (CCG) Supertagger for Bahasa Indonesia***

**Wisnu Adi Nurcahyo  
NIM: 1301160479**

Proposal ini diajukan sebagai usulan pembuatan tugas akhir pada  
Program Studi Sarjana Informatika  
Fakultas Informatika Universitas Telkom

Bandung, 24 Oktober 2019  
Menyetujui

Calon Pembimbing 1

Dr. Ade Romadhony, S.T., M.T.  
NIP: 06840042

## Abstrak

Dalam pemrosesan bahasa alami, combinatory categorial grammar (CCG) merupakan salah satu formalisme tata bahasa yang dapat digunakan untuk membangun sebuah *parser* yang umumnya dikenal sebagai CCG *parser*. CCG *parser* dapat digunakan untuk berbagai macam keperluan dalam pemrosesan bahasa alami. Sebagai contoh, CCG *parser* dapat digunakan untuk memperoleh informasi (*information extraction*) dari suatu kalimat yang kemudian membentuk sebuah *query*. Agar dapat bekerja, CCG *parser* membutuhkan CCG *lexicon*. CCG *lexicon* diperoleh dari proses yang bernama *supertagging*. *Supertagging* adalah proses pelabelan suatu token kata terhadap *supertag*-nya. Perangkat lunak yang melakukan *supertagging* disebut sebagai *supertagger*. Demikian itu, *supertagging* merupakan langkah pertama yang perlu dilakukan sebelum membangun sebuah CCG *parser*. *Supertagger* yang dibangun dalam tugas akhir ini dimaksudkan sebagai produsen CCG *lexicon* bahasa Indonesia untuk riset-riset yang berkenaan dengan CCG di masa yang akan datang.

**Kata Kunci:** natural language processing, combinatory categorial grammar, supertagger, maximum entropy model, bahasa indonesia, haskell

# Daftar Isi

<b>Abstrak</b>	<b>i</b>
<b>Daftar Isi</b>	<b>ii</b>
<b>I Pendahuluan</b>	<b>1</b>
1.1 Latar Belakang . . . . .	1
1.2 Perumusan Masalah . . . . .	1
1.3 Tujuan . . . . .	2
1.4 Batasan Masalah . . . . .	2
1.5 Rencana Kegiatan . . . . .	2
1.6 Jadwal Kegiatan . . . . .	2
<b>II Kajian Pustaka</b>	<b>4</b>
2.1 Categorical Grammar . . . . .	4
2.2 Combinatory Categorical Grammar . . . . .	4
2.3 Category Theory . . . . .	6
2.4 Lambda Calculus . . . . .	6
2.5 Supertagging . . . . .	7
2.6 Maximum Entropy Model . . . . .	7
<b>Daftar Pustaka</b>	<b>8</b>
<b>Lampiran</b>	<b>9</b>

# Bab I

## Pendahuluan

### 1.1 Latar Belakang

Riset pemrosesan bahasa natural untuk bahasa Indonesia saat ini terbilang sedikit. Bahkan, masih banyak area riset yang belum tersentuh seperti contohnya *combinatory categorial grammar* (CCG). CCG merupakan formalisme tata bahasa yang salah satu manfaatnya adalah untuk memperoleh informasi (*information extraction*) dari suatu kalimat. Informasi tersebut diperoleh setelah melakukan *parsing* berdasarkan formalisme CCG dengan menggunakan perangkat lunak bernama CCG *parser*. Untuk dapat melakukan *parsing*, CCG *parser* membutuhkan CCG *lexicon* yang mengandung bentuk formal dari suatu token kata. Bentuk formal yang dimaksud adalah *category* dalam *category theory*. CCG *lexicon* diperoleh dari proses pelabelan suatu token kata terhadap bentuk formalnya yang mana dikenal sebagai *supertagging*. Proses *supertagging* akan menghasilkan *supertag* yang kemudian disebut sebagai CCG *supertag* karena formalisme yang digunakan adalah formalisme CCG. Dalam hal ini, CCG *supertag* adalah CCG *lexicon* itu sendiri.

Tugas akhir dengan judul “Membangun sebuah Combinatory Categorical Grammar (CCG) Supertagger berbasis Maximum Entropy untuk Bahasa Indonesia” berusaha untuk membangun versi awal dari CCG *supertagger* untuk bahasa Indonesia yang mana harapannya dapat menjadi inisiator riset pemrosesan bahasa natural dengan tema CCG sehingga ke depannya akan ada lebih banyak riset mengenai CCG yang tersedia. *Supertagger* yang dimaksud dalam tugas akhir ini akan dibangun dengan menggunakan model Maximum Entropy (MaxEnt) dan implementasinya akan ditulis dalam bahasa pemrograman Haskell. Model MaxEnt digunakan karena keterbatasan *dataset* untuk melakukan *learning*. Adapun bahasa pemrograman Haskell digunakan karena abstraksi bahasanya yang sangat mendekati *category theory* serta kemampuannya yang sangat baik dalam pemrosesan data.

### 1.2 Perumusan Masalah

Rumusan masalah yang akan diangkat yaitu:

1. Mengapa CCG *supertagger* diperlukan?

2. Apa saja yang harus dipersiapkan untuk membangun CCG *supertagger*?
3. Bagaimana proses pembangunan CCG *supertagger*?

### 1.3 Tujuan

Tujuan yang diharapkan dapat tercapai oleh tugas akhir ini yaitu:

1. Mengenalkan alternatif metode yang dapat digunakan dalam pemrosesan bahasa alami untuk bahasa Indonesia.
2. Merilis CCG *supertagger* pertama untuk bahasa Indonesia.
3. Membuka peluang riset untuk CCG *parser* bahasa Indonesia.

### 1.4 Batasan Masalah

Hipotesis dari tugas akhir ini yaitu:

1. Memberikan label CCG untuk proses *learning* merupakan permasalahan utama dari tugas akhir ini.
2. *Supertagger* yang akan dibangun kemungkinan besar memiliki akurasi yang cenderung rendah.
3. CCG *lexicon* sudah dapat digunakan oleh CCG *parser* bahasa Indonesia (apabila ada).

### 1.5 Rencana Kegiatan

Rencana kegiatan yang akan dilakukan adalah sebagai berikut:

- Studi literatur
- Studi *tools* yang tersedia
- Studi bahasa pemrograman yang akan digunakan
- Perancangan sistem *supertagger*
- Membangun *supertagger*
- Memeriksa hasil

### 1.6 Jadwal Kegiatan

Laporan proposal ini akan dijadwalkan sesuai dengan tabel 1.1.

Tabel 1.1: Jadwal kegiatan proposal tugas akhir.

No	Kegiatan	Bulan ke-																							
		1				2				3				4				5				6			
1	Studi Literatur																								
2	Studi <i>Tools</i> yang Tersedia																								
3	Studi Bahasa Pemrograman																								
4	Pengumpulan Data																								
5	Analisis dan Perancangan Sistem																								
6	Implementasi Sistem																								
7	Analisa Hasil Implementasi																								
8	Penulisan Laporan																								

## Bab II

### Kajian Pustaka

#### 2.1 Categorical Grammar

Categorical Grammar (CG) merupakan sebuah istilah yang mencakup beberapa formalisme terkait yang diajukan untuk sintaks dan semantik dari bahasa alami serta untuk bahasa logis dan matematis [2]. Karakteristik yang paling terlihat dari CG adalah bentuk esktrim dari leksikalismenya di mana beban utama (atau bahkan seluruh beban) sintaksisnya ditanggung oleh leksikon. Konstituen tata bahasa dalam *categorical grammar* dan khususnya semua leksikal diasosiasikan dengan suatu *type* atau “*category*” (dalam *category theory*) yang mendefinisikan potensi mereka untuk dikombinasikan dengan konstituen lain untuk menghasilkan konstituen majemuk. *Category* tersebut adalah salah satu dari sejumlah kecil *category* dasar (seperti NP) atau *functor* (dalam *category theory*).

Ada beberapa notasi berbeda untuk *category* dalam merepresentasikan *directional*-nya. Notasi yang paling umum digunakan adalah “*slash notation*” yang dipelopori oleh Bar-Hilel, Lambek, dan kemudian dimodifikasi dalam kelompok teori yang dibedakan sebagai tata bahasa “*combinatory*” *categorical grammar* (CCG). Sebagai contoh, *category*  $(S \backslash NP) / NP$  merupakan suatu *functor* yang memiliki dua buah notasi *slash* yaitu  $\backslash$  dan  $/$ . Masing-masing notasi *slash* tersebut merepresentasikan *directionality* yang berbeda. Notasi *forward slash*,  $/$ , mengindikasikan bahwa argumen dari suatu *functor*  $X/Y$  ada di bagian kanan atau dengan kata lain  $Y$ . Adapun *backward slash*,  $\backslash$ , mengindikasikan bahwa argumen dari suatu *functor*  $X \backslash Y$  ada di bagian kiri atau dengan kata lain  $X$ . Demikian itu, penggunaan notasi *slash* yang tepat sangat penting dikarenakan hal ini dapat mempengaruhi konstituen dari hasil “kombinasi” *category*-nya.

#### 2.2 Combinatory Categorical Grammar

Combinatory Categorical Grammar (CCG) merupakan salah satu formalisme tata bahasa yang gaya aturannya diturunkan dari *categorical grammar* dengan beberapa penambahan aturan dan istilah baru. Di CCG, *category* dapat dipasangkan dengan *combinator*. Dalam hal ini, *combinator* yang dimaksud



adalah abstraksi fungsi lambda (dalam *lambda calculus*). Sebagai contoh, *category*  $(S \setminus NP)/NP$  dapat dipasangkan dengan fungsi lambda  $\lambda x.f x$  sehingga dapat ditulis menjadi  $(S \setminus NP)/NP : \lambda x.f x$ . Adapun pemetaan dari suatu token kata ke *category*-nya menggunakan notasi  $\vdash$ . Sebagai contoh, anggap saja kita memiliki kamus pemetaan sebagai berikut.

$$\begin{aligned}
&Pamungkas \vdash NP : pamungkas' \\
&Setyo \vdash NP : setyo' \\
&dan \vdash CONJ : \lambda x.\lambda y.\lambda f. (f\ x) \wedge (f\ y) \\
&menyukai \vdash (S \setminus NP)/NP : \lambda x.\lambda y. suka(y, x) \\
&rendang \vdash NP : rendang'
\end{aligned}$$

Dengan kamus seperti tersebut, apabila kita memiliki kalimat “Pamungkas dan Setyo menyukai rendang”, maka akan kita dapatkan:

Pamungkas	dan	Setyo	menyukai	rendang
$NP$	$CONJ$	$NP$	$(S \setminus NP)/NP$	$NP$
$: pamungkas'$	$: \lambda x.\lambda y.\lambda f. (f\ x) \wedge (f\ y)$	$: setyo'$	$: \lambda x.\lambda y. suka(y, x)$	$: rendang'$

Ada beberapa operasi yang dapat dilakukan dalam CCG. *Operand* dari operasi yang dimaksud adalah *category*. Berdasarkan contoh di atas, akan ada tiga operasi yang dijalankan yaitu *coordination*, *forward application*, dan *backward application*. Untuk mendapatkan hasil yang diinginkan, kita lakukan *type rising* sebelum *backward application*. Sehingga, kita dapatkan:

Pamungkas	dan	Setyo	menyukai	rendang
$NP$	$CONJ$	$NP$	$(S \setminus NP)/NP$	$NP$
$: pamungkas'$	$: \lambda x.\lambda y.\lambda f. (f\ x) \wedge (f\ y)$	$: setyo'$	$: \lambda x.\lambda y. suka(y, x)$	$: rendang'$
$NP$			$S \setminus NP$	
$: \lambda f. (f\ pamungkas') \wedge (f\ setyo')$			$: \lambda y. suka(y, rendang')$	
$\rightarrow T$				
$S/(S \setminus NP)$				
$: \lambda f. (f\ pamungkas') \wedge (f\ setyo')$				
$\rightarrow$				
$S$				
$: suka(pamungkas', rendang') \wedge suka(setyo', rendang')$				

Berdasarkan hasil evaluasi tersebut, kita dapatkan *query* 2.1 yang diperoleh dari kalimat “Pamungkas dan Setyo menyukai rendang”. Demikian itu,

komputer dapat melakukan komputasi berdasarkan *query* yang telah diperoleh.

$$\text{suka}(\text{pamungkas}', \text{rendang}') \wedge \text{suka}(\text{setyo}', \text{rendang}') \quad (2.1)$$

Kegiatan tersebut merupakan apa yang disebut dengan CCG *parsing*. Untuk dapat melakukan parsing, CCG *lexicon* diperlukan. Untuk mendapatkan CCG *lexicon* kita dapat menggunakan CCG *supertagger* yang akan melakukan pelabelan suatu token kata ke CCG *lexicon* berdasarkan pemetaannya.

## 2.3 Category Theory

*Category Theory* (CT) merupakan formalisme yang dapat digunakan untuk memformalkan struktur matematis. CT mempelajari *category* yang merupakan sebuah representasi dari suatu abstraksi konsep matematis. Suatu *category* memiliki kumpulan *object* dan *morphism*. Untuk mempermudah pemahaman mengenai CT, kita akan gunakan *category of set* (kategori dari himpunan) sebagai contoh. Dalam *category of set*, *object*-nya adalah himpunan dan *morphism*-nya (terkadang disebut dengan *arrow*) adalah fungsi (*function*, sebuah pemetaan). Kemudian, pemetaan dari suatu *category C* ke *category D* yang dipetakan oleh  $F$  ( $F : C \rightarrow D$ ) disebut sebagai *functor*.

## 2.4 Lambda Calculus

*Lambda calculus* ( $\lambda$ -*calculus*) merupakan sebuah formalisme yang dikembangkan oleh Alonzo Church sebagai alat yang digunakan untuk memahami konsep komputasi yang efektif [1]. Formalisme  $\lambda$ -*calculus* cukup populer dan bahkan dijadikan sebagai pondasi teori bagi paradigma pemrograman *functional programming*. Konsep utama dari  $\lambda$ -*calculus* adalah apa yang disebut dengan *expression*. Suatu *expression* dalam  $\lambda$ -*calculus* terdiri dari tiga bagian yaitu *lambda notation* ( $\lambda$ ), *argument* (seperti  $a$ ,  $b$ ,  $c$ ,  $x$ , dan lain-lain), dan *body* yang dipisahkan dengan tanda titik. Sebagai contoh, fungsi lambda  $\lambda x.x$  merupakan sebuah fungsi identitas yang mengambil argumen  $x$  kemudian mengembalikan nilai  $x$  itu sendiri. Dalam hal ini, terlihat bahwa notasi  $\lambda$  merupakan sebuah penanda bagi suatu fungsi lambda. Kemudian, pengubah  $x$  setelah notasi  $\lambda$  merupakan argumen dari fungsi tersebut. Selanjutnya, tanda titik merupakan pemisah antara *head* dan *body* fungsi lambda. Terakhir, setelah tanda titik adalah *body* dari suatu fungsi lambda yang mana berupa *expression*.

Untuk mempermudah pemahaman,  $\lambda$ -*calculus* dapat diperlakukan seperti fungsi tanpa nama. Sebagai contoh, fungsi lambda  $(\lambda x.x + 5)$  apabila diberikan nilai 2 sehingga menjadi  $(\lambda x.x + 5)2$  akan dievaluasi menjadi  $\lambda(2).(2) + 5$ . Demikian itu, nilai yang dikembalikan oleh fungsi tersebut adalah 7. Sama seperti fungsi pada umumnya, konsep ini bernama *substitution* (substitusi). Memahami  $\lambda$ -*calculus* dirasa perlu berhubung dalam tugas akhir ini  $\lambda$ -*calculus* digunakan

sebagai bentuk formal di *category* dalam konteks CCG *lexicon*. Meskipun  $\lambda$ -*calculus* tidak sesederhana yang dijelaskan sebelumnya, setidaknya memahami  $\lambda$ -*calculus* seperti ini sudah cukup untuk dapat membangun *supertagger* yang ada di tugas akhir ini.

## **2.5 Supertagging**

TBA.

## **2.6 Maximum Entropy Model**

TBA.

## Daftar Pustaka

- [1] Raul Rojas. A tutorial introduction to the lambda calculus. *CoRR*, abs/1503.09060, 2015.
- [2] Mark Steedman. Categorical grammar. Technical report, 1992.

## Lampiran