

**CCGtown: Alat Anotasi Combinatory  
Categorial Grammar (CCG) Semi-otomatis  
Berbasis Web**

**Proposal Tugas Akhir**

**Kelas TA NLP**

**Wisnu Adi Nurcahyo  
NIM: 1301160479**



**Program Studi Sarjana Informatika  
Fakultas Informatika  
Universitas Telkom  
Bandung  
2020**

## Lembar Persetujuan

**CCGtown: Alat Anotasi Combinatory Categorical Grammar  
(CCG) Semi-otomatis Berbasis Web**

***CCGtown: A Web-based Semi-automatic Combinatory  
Categorical Grammar (CCG) Annotation Tool***

**Wisnu Adi Nurcahyo  
NIM: 1301160479**

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

Bandung, 13 November 2020  
Menyetujui

Calon Pembimbing 1

Calon Pembimbing 2

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

Said Al Faraby, S.T., M.Sc.  
NIP: 15890019

# Abstrak

TBA.

**Kata Kunci:** natural language processing, combinatory categorial grammar, annotation tool

# 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 . . . . .	2
1.3 Tujuan . . . . .	2
1.4 Rencana Kegiatan . . . . .	2
<b>II Kajian Pustaka</b>	<b>3</b>
2.1 Categorical Grammar . . . . .	3
2.2 Combinatory Categorical Grammar . . . . .	4
2.3 Lambda Calculus . . . . .	5
2.4 CCGweb . . . . .	6
<b>III Perancangan Sistem</b>	<b>7</b>
<b>Daftar Pustaka</b>	<b>8</b>
<b>Lampiran</b>	<b>9</b>

# Bab I

## Pendahuluan

### 1.1 Latar Belakang

Riset pemrosesan bahasa alami untuk bahasa Indonesia saat ini masih terbilang sedikit. Bahkan, masih banyak area riset yang belum tersentuh seperti contohnya *combinatory categorial grammar* (CCG). Sementara itu, riset mengenai CCG untuk bahasa Inggris sudah cukup matang. Adapun untuk bahasa lainnya (seperti bahasa Vietnam) sudah mulai menggunakan CCG di dalam penelitiannya [4]. Agar dapat menerapkan CCG di dalam aplikasi yang dibangun, *tools* seperti CCG *parser* dan CCG *supertagger* harus tersedia terlebih dahulu. Masing-masing dari *tools* tersebut memerlukan *dataset* agar dapat memberikan hasil yang akurat.

Umumnya terdapat dua cara yang paling sering digunakan untuk mengembangkan CCG *supertagger* maupun CCG *parser* bahasa lokal yaitu (1) membangun *dataset* CCG *supertag* secara manual maupun semi-otomatis atau (2) melakukan transfer *dataset* dari CCGbank (atau dari sumber lainnya) ke dalam bahasa lokal dengan cara melakukan alih bahasa dan bila perlu melakukan penyesuaian untuk *supertag*-nya [2]. Proses pembangunan *dataset* umumnya menggunakan bantuan *annotation tool* agar proses anotasinya menjadi lebih mudah. Salah satu *annotation tool* yang dapat digunakan adalah CCGweb [1].

Tugas akhir dengan judul “CCGtown: Alat Anotasi Combinatory Categorical Grammar (CCG) Semi-otomatis Berbasis Web” berusaha untuk membangun alat anotasi CCG baru dengan UI/UX yang lebih baik dari CCGweb. Selain itu, dengan bantuan NLTK alat anotasi ini dapat melakukan *generate* untuk CCG *derivation*-nya kemudian pengguna dapat mengubah *derivation*-nya apabila diperlukan. Tujuan dari dibangunnya alat anotasi CCG ini adalah untuk mempermudah proses anotasi yang repetitif. Selanjutnya, *dataset* CCG pertama untuk bahasa Indonesia diharapkan dapat dipublikasikan.

## 1.2 Perumusan Masalah

Rumusan masalah yang akan diangkat yaitu:

1. Bagaimana proses pembangunan alat anotasi untuk CCG?
2. Apakah CCGtown dapat digunakan dengan mudah?

## 1.3 Tujuan

Tujuan yang diharapkan dapat tercapai oleh tugas akhir ini yaitu:

1. Membangun dan merilis alat anotasi CCG semi-otomatis.
2. Dapat digunakan untuk membangun *dataset* CCG.

## 1.4 Rencana Kegiatan

Rencana kegiatan yang akan dilakukan adalah sebagai berikut:

- Studi literatur
- Studi *tools* yang tersedia
- Studi bahasa pemrograman yang akan digunakan
- Perancangan sistem *annotation tool* CCGtown
- Membangun *annotation tool* CCGtown
- Menguji kemudahan dalam menggunakan CCGtown

## 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 [6]. Karakteristik yang paling terlihat dari CG adalah bentuk ekstrim 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*). Dalam hal ini, *category* dapat diartikan sebagai *syntactic type* dari suatu kata.

Secara formal, *syntactic type* didefinisikan sebagai himpunan bagian dari suatu *semigroup*  $M$  yang tunduk pada tiga operasi yaitu 2.1, 2.2, dan 2.3 dimana  $A$ ,  $B$ , dan  $C$  merupakan himpunan bagian dari  $M$  [3]. Adapun  $A \cdot B$  dibaca  $A$  times  $B$ ,  $C/B$  dibaca  $C$  over  $B$ , dan  $A \setminus C$  dibaca  $A$  under  $C$ . Selanjutnya, dapat dilihat bahwasannya untuk semua  $A, B, C \subseteq M$  sehingga kita dapatkan 2.4 dan 2.5. Terakhir, persamaan 2.6 dapat diabaikan apabila dihadapkan dengan *multiplicative system* yang tidak asosiatif. Sementara itu, apabila *semigroup*-nya merupakan sebuah *monoid* dengan identitas 1 maka kita dapatkan 2.7 dimana  $I = \{1\}$ .

$$A \cdot B = \{x \cdot y \in M \mid x \in A \wedge y \in B\} \quad (2.1)$$

$$C/B = \{x \in M \mid \forall_{y \in B} x \cdot y \in C\} \quad (2.2)$$

$$A \setminus C = \{y \in M \mid \forall_{x \in A} x \cdot y \in C\} \quad (2.3)$$

$$A \cdot B \subseteq C \quad \text{jika dan hanya jika} \quad A \subseteq C/B \quad (2.4)$$

$$A \cdot B \subseteq C \quad \text{jika dan hanya jika} \quad B \subseteq A \setminus C \quad (2.5)$$

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\NP)/NP :  $\lambda x.\lambda y. suka(y, x)$   
 rendang  $\vdash$  NP : *rendang'*

Gambar 2.1: Kamus yang memetakan token kata ke bentuk CCG *lexicon*-nya.

$$(A \cdot B) \cdot C = A \cdot (B \cdot C) \quad (2.6)$$

$$I \cdot A = A = A \cdot I \quad (2.7)$$

Ada beberapa notasi berbeda untuk *category* dalam merepresentasikan *directional*-nya. Notasi yang paling umum digunakan adalah “*slash notation*” yang dipelopori oleh Bar-Hillel, Lambek, dan kemudian dimodifikasi dalam kelompok teori yang dibedakan sebagai tata bahasa “*combinatory*” *categorial grammar* (CCG). Sebagai contoh, *category* (S\NP)/NP merupakan suatu *functor* yang memiliki dua buah notasi *slash* yaitu \ 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*, \, mengindikasikan bahwa argumen dari suatu *functor* X\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 Categorial Grammar

Combinatory Categorial Grammar (CCG) merupakan salah satu formalisme tata bahasa yang gaya aturannya diturunkan dari *categorial grammar* dengan beberapa penambahan aturan dan istilah baru [7]. Di CCG, *category* dapat dipasangkan dengan *semantic representation*. Dalam hal ini, *semantic representation* yang dimaksud adalah abstraksi fungsi lambda (dalam *lambda calculus*, *lambda function*). Sebagai contoh, *category* (S\NP)/NP dapat dipasangkan dengan fungsi lambda  $\lambda x.f x$  sehingga dapat ditulis menjadi (S\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 seperti pada Gambar 2.1. Apabila kita memiliki kalimat “Pamungkas dan Setyo menyukai rendang”, maka kita dapatkan:



Pamungkas	dan	Setyo	menyukai	rendang
NP	CONJ	NP	(S\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 *type rising*. Untuk mendapatkan hasil yang diinginkan, kita lakukan *type rising* sebelum *forward application* di akhir. Sehingga, kita dapatkan:

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

Berdasarkan hasil evaluasi tersebut, kita dapatkan *query* 2.8 yang diperoleh dari kalimat “Pamungkas dan Setyo menyukai rendang”. Demikian itu, komputer dapat melakukan komputasi berdasarkan *query* yang telah diperoleh. 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.

$$suka(pamungkas', rendang') \wedge suka(setyo', rendang') \quad (2.8)$$

## 2.3 Lambda Calculus

*Lambda calculus* ( $\lambda$ -*calculus*) merupakan sebuah formalisme yang dikembangkan oleh Alonzo Church sebagai alat yang digunakan untuk memahami konsep komputasi yang efektif [5]. 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.

## 2.4 CCGweb

TBA.

## **Bab III**

# **Perancangan Sistem**

TBA.

## Daftar Pustaka

- [1] Kilian Evang, Lasha Abzianidze, and Johan Bos. CCGweb: a new annotation tool and a first quadrilingual CCG treebank. In *Proceedings of the 13th Linguistic Annotation Workshop*, pages 37–42, Florence, Italy, August 2019. Association for Computational Linguistics.
- [2] Julia Hockenmaier and Mark Steedman. CCGbank: A corpus of CCG derivations and dependency structures extracted from the Penn treebank. *Computational Linguistics*, 33(3):355–396, 2007.
- [3] J. Lambek. *Categorial and Categorical Grammars*, pages 297–317. Springer Netherlands, Dordrecht, 1988.
- [4] Kiet Van Nguyen and Ngan Luu-Thuy Nguyen. Vietnamese transition-based dependency parsing with supertag features, 2019.
- [5] Raul Rojas. A tutorial introduction to the lambda calculus. *CoRR*, abs/1503.09060, 2015.
- [6] Mark Steedman. Categorial grammar. Technical report, 1992.
- [7] Mark Steedman. A very short introduction to ccg. Technical report, 1996.

## Lampiran