Jurnal Sistem Informasi, Teknik Komputer dan Teknologi Pendidikan (JUSTIKPEN)
Volume 5, Nomor 2 Edisi Oktober- Maret 2026, Page 217-224
ISSN 2828-7924 (media online) | DOI: https://doi.org/10.55338/justikpen.v5i2.505

Implementasi Bentuk Normal *Chomsky* dalam Penyederhanaan Tata Bahasa Bebas Konteks Berbasis Pemrograman Website

Celline Aloyshima Haris[1], I Wayan Sugianta Nirawana[2], Nouval Aulia Rahman[3], Fawwaz Muabil Athalla[4] Muhammad Arief Akbar[5], Muhammad Khaidir Akbar[6], Akmal Disar[7], Dinda Adhani[8], Muhammad Ardi Winata[9], Egi Octavian[10], Nandio Oktaviansyah Ramadhan[11]

[1,2,3,4,5,6,7,8,9,10,11]Pendidikan Komputer, Universitas Mulawarman, Samarinda, Indonesia
Email: [1]ariefakbar13032002@gmail.com, [2]fawwaznoris@gmail.com [3]nouvalacount@gmail.com, [4]dindadhn124@gmail.com, [5]mkhaidirakbar14@gmail.com, [6]akmaldiaar@gmail.com, [7]ardiwinata469@gmail.com, [8]meldaadawiyah73@gmail.com, [9]octavianegi@gmail.com, [10]nandioramadhan@gmail.com
Email Penulis Korespondensi: [1]ariefakbar13032002@gmail.com

Abstrak - Teori Bahasa dan Otomata merupakan salah satu pilar utama dalam ilmu komputer yang mempelajari bagaimana bahasa formal direpresentasikan, dibentuk, dan diproses oleh mesin komputasi. Di dalam kajian ini, *Context-Free Grammar* (CFG) menjadi aspek fundamental karena digunakan untuk mendefinisikan struktur sintaks bahasa pemrograman, melakukan analisis sintaks, serta menjadi dasar bagi berbagai algoritma kompilasi dan pemrosesan bahasa alami. Namun, bentuk CFG yang tidak terstruktur secara konsisten dapat menimbulkan kompleksitas pada proses parsing dan analisis komputasional. Untuk mengatasi hal tersebut, diperlukan bentuk normal yang lebih sistematis, yaitu *Chomsky Normal Form* (CNF). CNF menyederhanakan aturan produksi menjadi pola baku A → BC atau A → a, sehingga grammar menjadi lebih terstandarisasi dan mendukung penggunaan algoritma parsing seperti Cocke–Younger–Kasami (CYK). Hasil kajian menunjukkan bahwa transformasi CFG ke CNF tidak hanya memberikan bentuk grammar yang lebih sederhana, tetapi juga meningkatkan efisiensi dan akurasi proses parsing. Implementasi program berbasis CNF juga memberikan gambaran praktis mengenai bagaimana teori automata dapat diterapkan dalam sistem komputasi modern. Oleh karena itu, pemahaman mengenai CNF memiliki nilai strategis dalam pengembangan compiler, interpreter, perangkat lunak analisis sintaks, serta teknologi pemrosesan bahasa alami. Kajian ini diharapkan dapat memperkuat kompetensi mahasiswa dan peneliti dalam memahami serta mengimplementasikan teori bahasa formal dalam konteks komputasi.

Kata kunci: *Chomsky Normal Form, Context-Free Grammar,* Teori Bahasa dan Otomata*,* Transformasi *Grammar, Parsing CYK,* Implementasi *Python.*

Abstract - The Theory of Formal Languages and Automata is a fundamental area of computer science that investigates how formal languages are defined, structured, and processed by computational systems. Within this domain, the Context-Free Grammar (CFG) serves as a foundational model for describing the syntactic structure of programming languages, conducting syntactic analysis, and supporting various compilation and natural language processing techniques. However, the flexible and often irregular structure of CFGs can introduce significant complexity during parsing and computational analysis. To address this issue, a more standardized representation is required, namely the Chomsky Normal Form (CNF). CNF simplifies production rules into two structural patterns—A → BC and A → a—allowing grammars to be processed more systematically and enabling the effective use of parsing algorithms such as the Cocke–Younger–Kasami (CYK) algorithm. The results indicate that transforming CFG into CNF not only produces a more structured and standardized grammar but also enhances the efficiency and reliability of parsing processes. Furthermore, the CNF-based Python implementation demonstrates how theoretical concepts in automata and language theory can be applied to real-world computational tasks.

Keywords: Chomsky Normal Form, Context-Free Grammar, Automata Theory, Grammar Transformation, CYK Parsing, Python Implementation.

1. PENDAHULUAN

Teori bahasa automata merupakan salah satu teori komputasi. Teori bahasa automata dapat dijadikan suatu gagasan mendasar dalam komputasi yang menjadi *tools* untuk mengenali suatu persoalan atau masalah karena dapat memberikan konsep dan prinsip untuk suaru permasalahan yang berkolerasi dengan bidang Ilmu Komputer[1]. Meskipun menjadi landasan penting, banyak mahasiswa mengalami kesulitan dalam memahami materi Teori Bahasa dan Otomata karena sifatnya yang abstrak, sehingga berdampak pada hasil belajar mereka[7].Dalam kajian ini, bahasa formal digunakan untuk merepresentasikan struktur sintaksis secara matematis, sedangkan automata berperan sebagai model komputasi yang dapat mengenali dan memproses bahasa yang didefinisikan oleh grammar tertentu.

Submitted: 1**2/12/2025**; Accepted: **04/01/2026**; Published: **15/01/2026**

Volume 5, Nomor 2 Edisi Oktober- Maret 2026, Page 217-224
ISSN 2828-7924 (media online) | DOI: https://doi.org/10.55338/justikpen.v5i2.505

Salah satu komponen utama dalam teori bahasa formal adalah *Context-Free Grammar* (CFG). CFG memiliki peran penting dalam perancangan bahasa pemrograman, pengembangan compiler, analisis sintaks, hingga pengolahan bahasa alami (*Natural Language Processing*). Sebagai contoh implementasi nyata, algoritma CYK yang berbasis pada struktur tata bahasa ini telah diterapkan untuk memisahkan elemen pembentuk kalimat dalam Bahasa Indonesia[9].Namun, meskipun fleksibel, bentuk aturan produksi pada CFG sering kali kompleks dan tidak terstruktur secara konsisten, sehingga menyulitkan proses analisis dan komputasi algoritmik.

Untuk mempermudah proses parsing dan menganalisis struktur bahasa, diperlukan bentuk standar dari CFG yang lebih teratur dan efisien. Salah satu bentuk normal yang paling banyak digunakan adalah *Chomsky Normal Form* (CNF). Bentuk normal Chomsky merupakan bentuk normal yang akan berguna untuk tata bahasa bebas konteks (*Context Free Grammar*). Bentuk normal Chomsky dapat dibentuk dari tata bahasa bebas konteks yang telah mengalami penghilangan useless, unit dan ε. Ciri khas bentuk normal Chomsky dapat dilihat dengan adanya terminal atau dua variabel pada aturan produksi ruas kanan[2]. Dalam CNF, aturan produksi dibatasi pada dua pola, yaitu $A \rightarrow BC$ dan $A \rightarrow a$. Pembatasan ini bertujuan untuk menyederhanakan struktur grammar sehingga lebih mudah diproses oleh algoritma parsing, khususnya algoritma Cocke–Younger–Kasami (CYK), yang mengharuskan grammar berada dalam bentuk CNF agar dapat dijalankan. Algoritma CYK ini sangat efektif dalam mendeteksi struktur kalimat dan memberikan rekomendasi perbaikan, namun mutlak memerlukan format CNF agar dapat bekerja secara efisien [10].

Permasalahan yang sering terjadi ketika melakukan penyederhanaan tata bahasa adalah tidak adanya hubungan interaktif antara media yang digunakan dengan pengguna. Media yang bersifat statis membuat pengguna hanya menerima informasi secara pasif tanpa kesempatan untuk berlatih memasukkan grammar, menguji validitas aturan, maupun melihat proses transformasi CFG ke CNF secara langsung. Kondisi ini menyebabkan pemahaman konsep menjadi kurang mendalam dan tidak berlangsung secara konstruktif. Untuk mengatasi kendala tersebut, pendekatan pembelajaran berbasis teknologi interaktif atau visualisasi (seperti *Virtual Reality* atau simulasi web) terbukti dapat membantu mahasiswa memahami konsep automata dengan lebih baik [8].Proses penyederhanaan tata bahasa juga dapat menimbulkan kerumitan jika tidak menggunakan media visual, yaitu ketika melakukan proses penyederhanaan terdapat pohon penurunan yang ambigu yakni pohon penurunan yang berbeda yang menghasilkan aturan produksi yang tidak berarti dan tidak efisien[3].

Transformasi CFG menjadi CNF tidak hanya berfungsi sebagai penyederhanaan struktur grammar, tetapi juga menjadi langkah penting untuk meningkatkan efisiensi komputasi dalam berbagai aplikasi. Implementasi CNF digunakan pada perangkat lunak seperti compiler, interpreter, sistem deteksi kesalahan sintaks, hingga berbagai metode pemrosesan bahasa alami.Selain dalam pengembangan bahasa, konsep automata juga memiliki penerapan luas dalam simulasi perangkat logika sehari-hari, seperti pada perancangan sistem *vending machine* jamu tradisional [11].Oleh karena itu, pemahaman mengenai prosedur transformasi CFG ke CNF memiliki nilai strategis dalam pendidikan maupun penelitian informatika.

Berdasarkan pentingnya peran CNF dalam teori dan praktik komputasi modern, penelitian atau kajian ini disusun untuk memberikan pemahaman yang lebih mendalam mengenai konsep dasar bahasa formal, karakteristik Chomsky Normal Form, prosedur transformasi CFG menjadi CNF, serta contoh implementasinya dalam pemrograman. Dengan demikian, tulisan ini diharapkan mampu memberikan kontribusi terhadap pengembangan kompetensi analisis sintaks dan pemrograman pada ranah ilmu komputer.

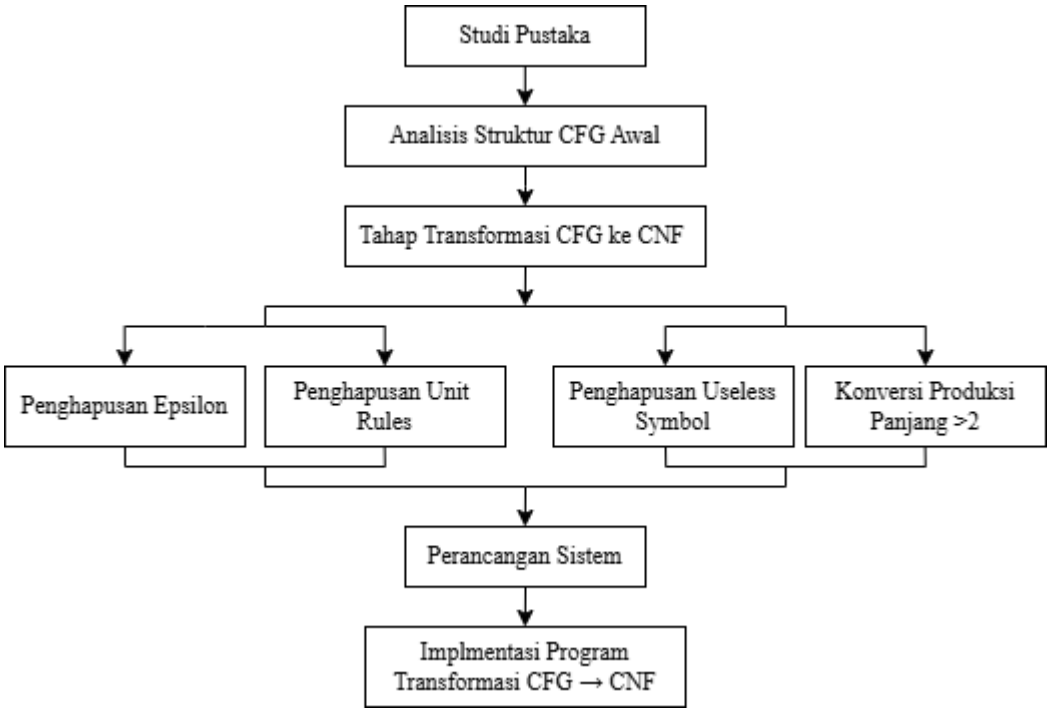
2. METODOLOGI PENELITIAN

Metode penelitian mengenai implementasi Bentuk Normal Chomsky (CNF) dalam penyederhanaan tata bahasa bebas konteks (CFG) untuk pemrograman disusun sebagai kerangka kerja yang digunakan untuk menyelesaikan permasalahan utama penelitian, yaitu bagaimana melakukan transformasi CFG menjadi bentuk CNF secara sistematis dan bagaimana hasil transformasi tersebut dapat diterapkan dalam proses pembelajaran pemrograman. Sebelum memasuki uraian teknis mengenai implementasi CFG untuk bentuk CNF dalam implementasi pemrograman, penelitian ini perlu menjelaskan secara sistematis langkah-langkah yang digunakan dalam proses perancangan,

Submitted: 1**2/12/2025**; Accepted: **04/01/2026**; Published: **15/01/2026**

Jurnal Sistem Informasi, Teknik Komputer dan Teknologi Pendidikan (JUSTIKPEN)
Volume 5, Nomor 2 Edisi Oktober- Maret 2026, Page 217-224
ISSN 2828-7924 (media online) | DOI: https://doi.org/10.55338/justikpen.v5i2.505

pembangunan, dan pengujian sistem. Setiap tahapan disusun secara terstruktur agar metode yang digunakan adalah metode penelitian deskriptif analitis dengan pendekatan rekayasa perangkat lunak *(software engineering approach)* serta didukung oleh eksperimen komputasional. Kerangka penelitian ini berfungsi sebagai pendekatan terstruktur untuk menjelaskan seluruh tahapan yang dilakukan, adapun kerangka kerja penelitian yang digunakan dalam studi ini dapat digambarkan pada Gambar berikut:



Gambar 1. Kerangka Penelitian

2.1 Context-Free Grammar (CFG)

Context-Free Grammar (CFG) merupakan salah satu model formal yang digunakan dalam teori bahasa dan otomata untuk mendefinisikan struktur sintaks sebuah bahasa. CFG terdiri dari empat komponen, yaitu:
V : himpunan variabel (non-terminal)
Σ : himpunan terminal
R : himpunan aturan produksi
S : simbol awal
Aturan produksi pada CFG dapat dituliskan dalam bentuk:

$$A \rightarrow \alpha \quad (1)$$

dengan **A** adalah variabel dan **α** adalah rangkaian terminal dan/atau variabel. CFG banyak digunakan dalam perancangan *compiler*, *interpreter*, dan analisis sintaks bahasa pemrograman karena struktur produksinya dapat menggambarkan hierarki aturan sintaks secara jelas. Namun, bentuk produksi dalam CFG sering tidak teratur. Beberapa aturan bisa sangat panjang, mengandung ε-production, unit production, atau simbol-simbol yang tidak relevan. Hal ini menyebabkan proses parsing menjadi lebih kompleks dan kurang efisien, terutama ketika menggunakan algoritma berbasis matriks seperti Cocke–Younger–Kasami (CYK).
Untuk mengatasi hal tersebut, CFG perlu dinormalisasi ke dalam bentuk yang lebih terstruktur, yaitu *Chomsky Normal Form* (CNF).

Submitted: 1**2/12/2025**; Accepted: **04/01/2026**; Published: **15/01/2026**

Jurnal Sistem Informasi, Teknik Komputer dan Teknologi Pendidikan (JUSTIKPEN)
Volume 5, Nomor 2 Edisi Oktober- Maret 2026, Page 217-224
ISSN 2828-7924 (media online) | DOI: https://doi.org/10.55338/justikpen.v5i2.505

2.2 Chomsky Normal Form (CNF)

Chomsky Normal Form (CNF) adalah bentuk standar dari tata bahasa bebas konteks yang membatasi setiap aturan produksi ke dalam dua bentuk saja, yaitu:

$$A \rightarrow BC \quad (2)$$

atau

$$A \rightarrow a \quad (3)$$

dengan: **A, B, C** adalah variabel (non-terminal), dan **a** adalah terminal.

Aturan ini membuat grammar lebih terstruktur, lebih mudah diproses secara komputasional, dan kompatibel dengan algoritma parsing CYK. Untuk mengubah grammar ke CNF, sejumlah proses harus dilakukan, antara lain:

1. Eliminasi epsilon production ($A \rightarrow \varepsilon$)
2. Eliminasi unit production ($A \rightarrow B$)
3. Penghapusan variabel tidak berguna
4. Konversi produksi panjang menjadi bentuk biner
5. Normalisasi terminal agar sesuai aturan CNF

Transformasi CFG ke CNF tidak mengubah bahasa yang dihasilkan, melainkan hanya mengubah bentuk produksinya. Oleh karena itu, CNF menjadi standar penting dalam pembelajaran komputasi formal maupun implementasi algoritma parsing modern.

2.3 Analisis Struktur CFG Awal

Analisis struktur CFG awal dilakukan untuk memahami karakteristik tata bahasa yang akan dikonversi ke Chomsky Normal Form (CNF). Tahap ini penting karena kualitas hasil transformasi CNF sangat dipengaruhi oleh bentuk awal grammar yang digunakan. Pada penelitian ini, CFG yang dianalisis merupakan grammar dasar yang digunakan untuk merepresentasikan ekspresi sederhana yang umum dijumpai dalam pembelajaran pemrograman, seperti operasi aritmetika, simbol terminal sederhana, maupun struktur produksi lainnya. Analisis dimulai dengan mengidentifikasi empat komponen utama grammar, yaitu himpunan variabel (V), himpunan terminal (Σ), himpunan aturan produksi (R), dan simbol awal (S). Setiap aturan produksi dicek apakah mengandung bentuk-bentuk yang tidak sesuai dengan standar CNF, seperti:

1. Epsilon production ($A \rightarrow \varepsilon$)
   Aturan ini harus dicatat sejak awal karena akan dihilangkan pada tahap transformasi.
2. Unit production ($A \rightarrow B$)
   Produksi satu variabel menuju variabel lain akan menambah kompleksitas parsing sehingga perlu dihilangkan.
3. Produksi panjang ($A \rightarrow B\ C\ D\ \ldots$)
   Aturan seperti ini harus dipecah menjadi produksi biner ketika dikonversi ke CNF.
4. Produksi campuran ($A \rightarrow aB$ atau $A \rightarrow Ba$)
   Aturan ini tidak sesuai CNF karena terminal tidak boleh berada berdampingan dengan non-terminal.
5. Variabel atau simbol tidak produktif dan tidak dapat dijangkau
   Simbol yang tidak menghasilkan terminal atau tidak dapat dicapai dari simbol awal akan dihapus untuk menyederhanakan grammar.

Hasil analisis struktur CFG awal digunakan sebagai dasar untuk menentukan langkah transformasi yang diperlukan di tahap berikutnya. Dengan memahami letak aturan yang harus dihilangkan, diperbaiki, atau dipisahkan, proses transformasi dapat dilakukan secara lebih terarah dan sesuai dengan flowchart yang menggambarkan setiap prosedur secara sistematis.

Submitted: 1**2/12/2025**; Accepted: **04/01/2026**; Published: **15/01/2026**

Jurnal Sistem Informasi, Teknik Komputer dan Teknologi Pendidikan (JUSTIKPEN)
Volume 5, Nomor 2 Edisi Oktober- Maret 2026, Page 217-224
ISSN 2828-7924 (media online) | DOI: https://doi.org/10.55338/justikpen.v5i2.505

2.4 Tahap Transformasi CFG ke CNF

Transformasi Context-Free Grammar (CFG) menjadi Chomsky Normal Form (CNF) merupakan proses sistematis yang bertujuan mengubah seluruh aturan produksi grammar menjadi bentuk standar tanpa mengubah bahasa yang dihasilkannya. Proses transformasi ini penting karena CNF menyediakan struktur produksi yang lebih terprediksi dan lebih mudah diolah secara komputasional, terutama untuk algoritma parsing berbasis matriks seperti Cocke–Younger–Kasami (CYK). Tahapan transformasi dilakukan secara berurutan agar grammar akhir memenuhi aturan CNF yang hanya memperbolehkan dua bentuk produksi, yaitu $A \rightarrow BC$ dan $A \rightarrow a$.

Tahap pertama adalah eliminasi *epsilon production* ($A \rightarrow \varepsilon$). Aturan produksi ε harus dihilangkan karena dapat menciptakan ambiguitas dalam proses parsing. Proses ini meliputi identifikasi variabel *nullable*, penghapusan aturan ε, serta penambahan aturan baru untuk mempertahankan bahasa dengan cara menghilangkan variabel nullable pada produksi terkait. Tahap kedua adalah eliminasi *unit production* ($A \rightarrow B$), yaitu aturan yang memetakan satu variabel ke variabel lainnya secara langsung. Aturan ini diganti dengan seluruh produksi milik variabel yang dituju sehingga grammar menjadi lebih ringkas dan bebas dari chain-rule yang tidak diperlukan.

Tahap berikutnya adalah eliminasi simbol tak berguna. Sebuah simbol dikatakan tidak berguna apabila tidak dapat menghasilkan string terminal atau tidak dapat dicapai dari simbol awal. Proses ini dilakukan dengan mengidentifikasi simbol produktif, mengecek ketercapaian setiap simbol dari S, dan menghapus produksi yang mengandung simbol tidak produktif maupun tidak reachable. Tahap selanjutnya adalah konversi produksi panjang ke bentuk biner. Produksi yang memiliki lebih dari dua variabel di sisi kanan harus dipecah menggunakan variabel baru, sehingga seluruh produksi menjadi bentuk biner, misalnya $A \rightarrow B\ X_1$ diikuti $X_1 \rightarrow C\ X_2$ dan seterusnya. Tahap terakhir adalah normalisasi terminal dalam produksi campuran. Apabila sebuah terminal muncul berdampingan dengan variabel, terminal tersebut harus diganti dengan variabel baru yang merepresentasikan terminal tunggal, sehingga seluruh aturan mengikuti pola CNF secara konsisten.

Hasil dari seluruh rangkaian proses tersebut menghasilkan grammar yang sepenuhnya berada pada bentuk Chomsky Normal Form. Transformasi ini tidak mengubah bahasa yang dihasilkan, tetapi menyederhanakan struktur grammar sehingga lebih mudah diproses oleh algoritma parsing dan mendukung implementasi pada perangkat lunak seperti compiler, interpreter, dan sistem pemrosesan bahasa alami.

2.5 Perancangan Sistem

Perancangan sistem pada penelitian ini dilakukan untuk mengimplementasikan prosedur transformasi CFG ke CNF menggunakan bahasa HTML, CSS dan Javascript. Perancangan sistem ini mengikuti tahapan-tahapan, di mana setiap proses transformasi dibuat modular agar mudah diuji, dimodifikasi, dan digunakan kembali. Perancangan dilakukan menggunakan pendekatan fungsi terstruktur, sehingga setiap tahapan transformasi memiliki fungsi khusus yang menangani tugas tertentu.

2.6 Implementasi Algoritma Verifikasi String Berbasis Chomsky Normal Form

Implementasi pemrograman dilakukan berbasis website dalam penelitian ini difokuskan pada pengembangan *parser* sederhana yang bertujuan untuk memvalidasi penerimaan string oleh tata bahasa bebas konteks yang telah disederhanakan ke dalam Bentuk Normal Chomsky (CNF). Algoritma yang dibangun menggunakan bahasa pemrograman Python ini mengadopsi pendekatan *bottom-up parsing*, di mana struktur CNF dinilai sangat kompatibel karena setiap langkah reduksi dapat direpresentasikan sebagai penggabungan pasangan token atau variabel secara terstruktur hingga mencapai simbol awal. Dalam implementasinya, aturan produksi dikelola menggunakan struktur data *dictionary* yang secara ketat mematuhi batasan format CNF, yakni setiap aturan harus berbentuk satu variabel menghasilkan tepat dua variabel non-terminal ($A \rightarrow BC$) atau satu variabel menghasilkan satu terminal ($A \rightarrow a$).

Submitted: 1**2/12/2025**; Accepted: **04/01/2026**; Published: **15/01/2026**

Jurnal Sistem Informasi, Teknik Komputer dan Teknologi Pendidikan (JUSTIKPEN)
Volume 5, Nomor 2 Edisi Oktober- Maret 2026, Page 217-224
ISSN 2828-7924 (media online) | DOI: https://doi.org/10.55338/justikpen.v5i2.505

3. HASIL DAN PEMBAHASAN

Dalam hal ini menguraikan hasil analisis mendalam dan implementasi teknis mengenai transformasi *Context-Free Grammar* (CFG) menjadi *Chomsky Normal Form* (CNF) serta pengujian validitas string menggunakan parser yang telah dikembangkan. Penelitian ini berfokus pada penyelesaian masalah kompleksitas struktur CFG yang sering kali tidak konsisten, yang dapat menghambat efisiensi proses parsing dalam sistem komputasi. Hasil pembahasan mencakup simulasi transformasi manual pada tata bahasa sampel, implementasi logika algoritma dalam lingkungan pemrograman Python, dan evaluasi kinerja parser dalam memvalidasi string input. Analisis ini bertujuan untuk membuktikan bahwa bentuk normal yang dihasilkan mampu meningkatkan efisiensi komputasi tanpa mengubah bahasa yang didefinisikan oleh tata bahasa awal.

3.1 Analisis Transformasi Tata Bahasa Bebas Konteks

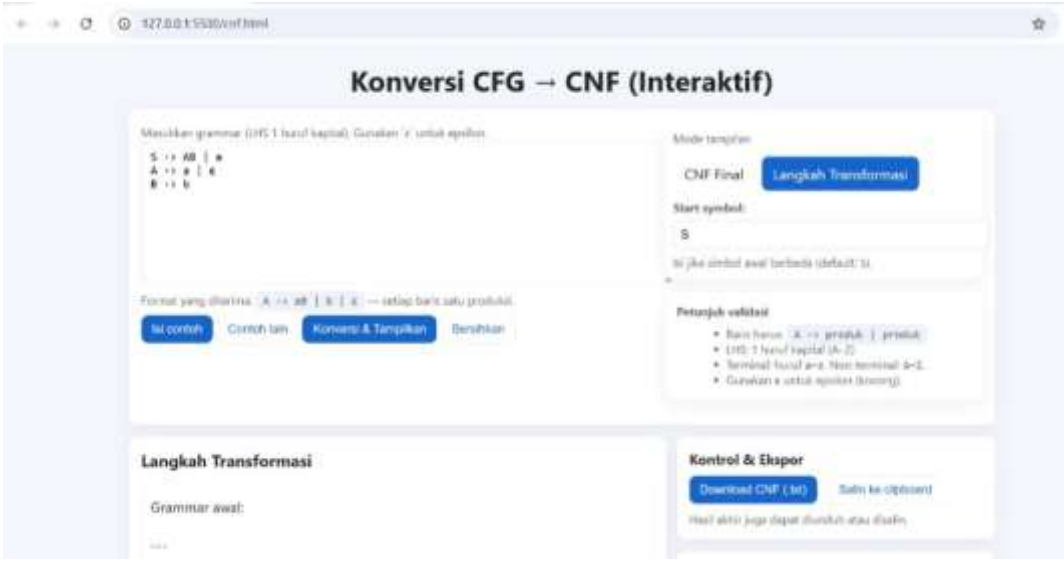
Untuk memvalidasi prosedur transformasi yang telah dirancang, penelitian ini menggunakan studi kasus tata bahasa yang merepresentasikan ekspresi sintaksis sederhana yang umum dalam pemrograman. Proses transformasi dilakukan secara bertahap mengikuti alur metodologi yang telah ditetapkan, dimulai dari analisis komponen dasar hingga normalisasi penuh. Berikut analisis transformasinya:

1. Ditemukan keberadaan *epsilon production* (A→ ε), di mana variabel non-terminal dapat menurunkan string kosong[1]. Hal ini teridentifikasi sebagai sumber ambiguitas utama yang harus dicatat sejak awal karena akan dihilangkan pada tahap transformasi pertama[3].
2. Teridentifikasi adanya *unit production* (A → B), yaitu aturan yang memetakan satu variabel ke variabel lain secara langsung tanpa adanya terminal[3]. Struktur ini dinilai menambah panjang jalur derivasi tanpa menghasilkan simbol terminal, sehingga meningkatkan beban komputasi[3].
3. Terdapat aturan produksi panjang dengan lebih dari dua variabel di sisi kanan (A→ BCD...) serta produksi campuran yang menggabungkan terminal dan variabel dalam satu aturan. Kedua bentuk ini secara eksplisit melanggar batasan CNF yang hanya memperbolehkan pola A → BC atau A→ a[3][5].

3.1.1 Implementasi Perangkat Lunak

Implementasi sistem transformasi dilakukan menggunakan bahasa HTML, CSS dan Javascript. Hal ini memungkinkan setiap tahapan transformasi, mulai dari eliminasi epsilon hingga normalisasi terminal, dijalankan sebagai fungsi terpisah yang terintegrasi.

b. Tampilan awal input grammar yang dikonversi S -> aB, A->a dan B ->b

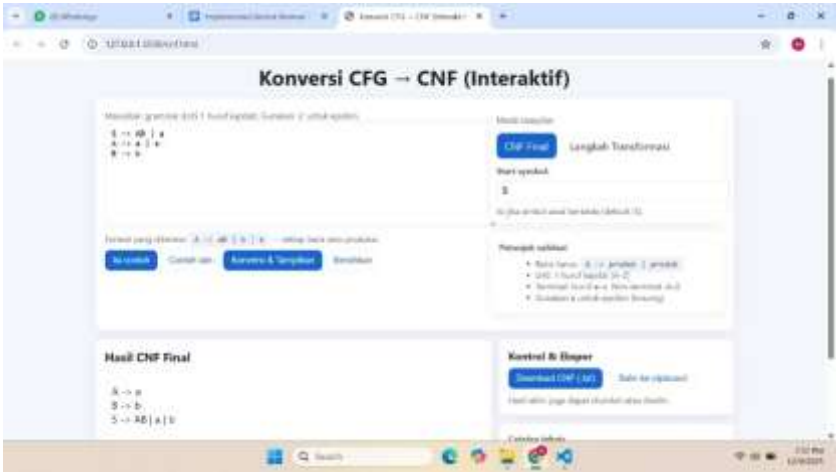


Gambar 2. Tampilan awal Website

Submitted: 1**2/12/2025**; Accepted: **04/01/2026**; Published: **15/01/2026**

Jurnal Sistem Informasi, Teknik Komputer dan Teknologi Pendidikan (JUSTIKPEN)
Volume 5, Nomor 2 Edisi Oktober- Maret 2026, Page 217-224
ISSN 2828-7924 (media online) | DOI: https://doi.org/10.55338/justikpen.v5i2.505

c. Output setelah input grammar yang dikonversi S -> aB, A->a dan B ->b



Gambar 3. Output Website

4. KESIMPULAN

Berdasarkan keseluruhan proses analisis, perancangan, dan pengujian yang telah dilakukan dalam penelitian ini, dapat disimpulkan bahwa implementasi Bentuk Normal Chomsky (CNF) merupakan metode yang sangat efektif untuk menyederhanakan kompleksitas struktur Tata Bahasa Bebas Konteks (CFG). Permasalahan utama yang diidentifikasi pada awal penelitian, yaitu ketidakteraturan aturan produksi CFG yang menghambat efisiensi komputasi, berhasil diatasi melalui serangkaian transformasi sistematis. Dengan mengubah aturan produksi yang mengandung *epsilon production*, *unit production*, serta simbol tak berguna menjadi bentuk tata bahasa menjadi jauh lebih terstruktur. Hasil transformasi ini terbukti tidak hanya merampingkan aturan produksi, tetapi juga menciptakan standarisasi yang memudahkan algoritma *parsing* dalam melakukan penelusuran sintaks tanpa terjebak pada ambiguitas rantai derivasi yang panjang.

UCAPAN TERIMAKASIH

Kami mengucapkan terima kasih kepada rekan-rekan kelompok A TBO yang telah memberikan kerja sama, semangat kebersamaan, serta kontribusi nyata dalam setiap tahap penyusunan, mulai dari pengumpulan data, analisis, hingga penyelesaian. Kehadiran dan bantuan mereka sangat membantu kelancaran proses penelitian ini, tidak lupa juga berterima kasih kepada seluruh pihak yang telah membantu secara langsung maupun tidak langsung, baik dalam bentuk dukungan teknis, motivasi, maupun penyedia informasi yang relavan. Semoga segala bantuan yang diberikan memberikan manfaat yang luas dan menjadi bagian dari upaya bersama untuk meningkatkan kualitas penelitian dibidang teori bahasa formal dan pemrograman.

REFERENCES

[1] Andrian, R., Wamiliana, & Dabukke, A. Y. C. (2015). Aplikasi Pengubah Bentuk Normal Chomsky Menjadi Bentuk Normal Greibach dengan Metode Substitusi. *Jurnal Komputasi*, *3*(2).

[2] N. Khairina *et al.*, *TEORI BAHASA FORMAL DAN AUTOMATA*. [Online]. Available: https://uma.ac.id/

[3] R. Andrian, W. Dan, and I. Indra Pratama, 'Penyederhanaan Tata Bahasa Bebas Konteks dalam Bentuk Normal Chomsky Menggunakan PHP', 2015. [Online]. Available: http://jurnal.fmipa.unila.ac.id/index.php/komputasiHal.31dari168

[4] Naseem, D., et al. "Induction of Chomsky Normal Form in Context-Free Grammar of LL(1) Parser". *Journal of Information & Communication Technology*, Vol. 15, Issue 1.

Submitted: 1**2/12/2025**; Accepted: **04/01/2026**; Published: **15/01/2026**

[5] V. Khodijah, Thoyyibah. T. S. Kom. M. Kom. Agus Heri Yunial. S. Kom. M. Kom,. (2021). BUKU AJAR TEORI BAHASA DAN AUTOMATA

[6] Polubelova, M.I., et al. "Certified Grammar Transformation to Chomsky Normal Form in F*". *Trudy ISP RAN*, 2016.

[7] A. M. Syafar, "Aplikasi Pembelajaran Model SAVAR Mata Kuliah Teori Bahasa Automata Berbasis Virtual Reality," *Jurnal Instek (Informatika Sains dan Teknologi)*, vol. 9, no. 1, hlm. 135-140, 2024.

[8] B. Prabowo, H. C. Rustamadji, dan Y. Fauziah, "Algoritma Cocke Younger Kasami untuk Deteksi Struktur Kalimat dan Merekomendasikannya Menggunakan Algoritma Damerau Levenshtein Distance," *Telematika: Jurnal Telematika dan Teknologi Informasi*, vol. 17, no. 2, hlm. 111–119, 2020.

[9] Erni, F. Titiani, S. A. Putri, dan W. Gata, "Penerapan Konsep Finite State Automata Pada Aplikasi Simulasi Vending Machine Jamu Tradisional," *Jurnal Informatika*, vol. 7, no. 2, hlm. 141-147, 2020.

[10] Jumarniati dan Hardiana, "Analisis Hasil Belajar Mahasiswa pada Mata Kuliah Teori Bahasa dan Automata," *Jurnal Penelitian Matematika dan Pendidikan Matematika*, vol. 6, no. 1, 2024.

[11] Y. Yanto, et al., "The Implementation of CYK Algorithm to Separate Sentence-Forming Elements in Indonesian," *Proxies*, vol. 2, no. 2, hlm. 53, 2019.

Submitted: 1**2/12/2025**; Accepted: **04/01/2026**; Published: **15/01/2026**