General CD Grammar Systems and Their Simplification

Radim Kocman Zbyněk Křivka Alexander Meduna

Centre of Excellence IT4Innovations
Faculty of Information Technology
Brno University of Technology
Božetěchova 2, Brno 612 66, Czech Republic
{ikocman,krivka,meduna}@fit.vutbr.cz

LTA 2017

Table of Contents

CD Grammar System

Definition – CD Grammar System

$$\Gamma = (N, T, P_1, P_2, \ldots, P_n, S), n \geq 1$$

N is the alphabet of nonterminals

T is the alphabet of terminals, $N \cap T = \emptyset$

S is the start symbol, $S \in N$

P_i (component) is a finite set of context-free rules, $1 \leq i \leq n$

CD Grammar System

Definition – CD Grammar System

$$\Gamma = (N, T, P_1, P_2, \ldots, P_n, S), n \geq 1$$

N is the alphabet of nonterminals

T is the alphabet of terminals, $N \cap T = \emptyset$

S is the start symbol, $S \in N$

P_i (component) is a finite set of context-free rules, $1 \leq i \leq n$

Our Setting

$n = 2$ and we use the $*$ and t modes

CD Grammar System

Definition – CD Grammar System

$$\Gamma = (N, T, P_1, P_2, \ldots, P_n, S), n \geq 1$$

N is the alphabet of nonterminals

T is the alphabet of terminals, $N \cap T = \emptyset$

S is the start symbol, $S \in N$

P_i (component) is a finite set of context-free rules, $1 \leq i \leq n$

Our Setting

$n = 2$ and we use the $*$ and t modes

Known Results

(1) $CD^{\varepsilon}_{\infty}(*) = \mathbf{CF}$ and (2) $CD^{\varepsilon}_{2}(t) = \mathbf{CF}$

General CD Grammar System

General CD Grammar System

- components can contain general (or phrase-structure) rules

General CD Grammar System

General CD Grammar System

- components can contain general (or phrase-structure) rules

Generative Power

The generative power does not change with the model.
It is still **RE**—the same as with a single component.

General CD Grammar System

General CD Grammar System

- components can contain general (or phrase-structure) rules

Generative Power

The generative power does not change with the model.
It is still **RE**—the same as with a single component.

Our Approach

- we further restrict each component separately
- the generative power should remain unchanged

Restricted Components

Restricted Components

Context-Free Component

It contains only context-free rules.

Restricted Components

Context-Free Component

It contains only context-free rules.

Homogeneous Component

Let $G = (N, T, P, S)$ be a grammar. If $x \rightarrow y \in P$ and $x \in \{A\}^+$ for some $A \in N$, then $x \rightarrow y$ is a *homogeneous rule*.
A homogeneous component has all its rules homogeneous.
It can still define **RE** by itself.

Restricted Components

Context-Free Component

It contains only context-free rules.

Homogeneous Component

Let $G = (N, T, P, S)$ be a grammar. If $x \rightarrow y \in P$ and $x \in \{A\}^+$ for some $A \in N$, then $x \rightarrow y$ is a *homogeneous rule*.
A homogeneous component has all its rules homogeneous.
It can still define **RE** by itself.

Evenly Homogeneous Component

If also $y \in \{B\}^+$ for some $B \in (N \cup T)$ and $|x| = |y|$, then $x \rightarrow y$ is an *evenly homogeneous rule*.
An evenly homogeneous component has all its rules evenly homogeneous.
It can generate only single symbol results on its own.

Kuroda Normal Form

Definition

Let $G = (N, T, P, S)$ be a grammar. G is in Kuroda normal form if every rule $p \in P$ has one of these three forms:

- $AB \rightarrow CD$,
- $A \rightarrow BC$,
- $A \rightarrow a$,

where $A, B, C, D \in N$ and $a \in (T \cup \{\varepsilon\})$.

Table of Contents

Reduced Forms

Idea—Transformation

- from any general grammar
- only two restricted components
- small number of non-context-free rules
- working in the $*$ and t modes

Reduced Forms

Idea—Transformation

- from any general grammar
- only two restricted components
- small number of non-context-free rules
- working in the $*$ and t modes

Goal

For a general grammar, $G = (N, T, P, S)$, construct a two-component general CD grammar system, $\Gamma = (N', T, H, I, S)$, such that
H is purely context-free, I contains only two rules,
$L_*(\Gamma) = L(G)$, and $L_t(\Gamma) = L(G)$.

Transformations

Goal

For a general grammar, $G = (N, T, P, S)$, construct a two-component general CD grammar system, $\Gamma = (N', T, H, I, S)$, such that H is purely context-free, I contains only two rules, $L_*(\Gamma) = L(G)$, and $L_t(\Gamma) = L(G)$.

Transformation 1

- I is homogeneous, $N' = N \cup \{0, 1\}$
- $I = \{11 \rightarrow 00,\ 0000 \rightarrow \varepsilon\}$

Transformation 2

- I is evenly homogeneous, $N' = N \cup \{0, 1, 2\}$
- $I = \{11 \rightarrow 00,\ 0000 \rightarrow 2222\}$

Construction Procedure

- let $G = (N, T, P, S)$ be a grammar
- G satisfies Kuroda normal form

Injection g for $m \geq 3$

from NonContextFree(P) to $(\{01\}^{+}\{00\}\{01\}^{+} \cap \{01, 00\}^{m})$

Construction Procedure

- let $G = (N, T, P, S)$ be a grammar
- G satisfies Kuroda normal form

Injection g for $m \geq 3$

from NonContextFree(P) to $(\{01\}^+\{00\}\{01\}^+ \cap \{01, 00\}^m)$

Example

$$m = 5: \quad 0100010101$$
$$0101000101$$
$$0101010001$$

Construction Procedure

- let $G = (N, T, P, S)$ be a grammar
- G satisfies Kuroda normal form

Injection g for $m \geq 3$

from NonContextFree(P) to $(\{01\}^+\{00\}\{01\}^+ \cap \{01, 00\}^m)$

Transformation 1

- For every $AB \rightarrow CD \in P$ where $A, B, C, D \in N$, add $A \rightarrow Cg(AB \rightarrow CD)$ and $B \rightarrow \mathrm{rev}(g(AB \rightarrow CD))D$ to H.
- For every $A \rightarrow x \in P$ where $A \in N$ and $x \in (\{\varepsilon\} \cup T \cup N^2)$, add $A \rightarrow x$ to H.

Example Transformation 1

Example

$P = \{\dots,\ A \to x,\ AB \to CD,\ EF \to GH\}$

Consider $m = 4$.

- $A \to x$:
  $A \to x$
- $AB \to CD$:
  $A \to C01010001$ and $B \to 10001010D$
- $EF \to GH$:
  $E \to G01000101$ and $F \to 10100010H$

Construction Procedure

- let $G = (N, T, P, S)$ be a grammar
- G satisfies Kuroda normal form

Injection g for $m \geq 3$

from NonContextFree(P) to $(\{01\}^+\{00\}\{01\}^+ \cap \{01, 00\}^m)$

Transformation 2

- For every $AB \to CD \in P$ where $A, B, C, D \in N$, add $A \to Cg(AB \to CD)$ and $B \to \text{rev}(g(AB \to CD))D$ to H.
- For every $A \to x \in P$ where $A \in N$ and $x \in (\{\varepsilon\} \cup T \cup N^2)$, add $A \to x$ to H.
- Add $2 \to \varepsilon$ to H.

Basic Ideas (Transformation 1)

Basic idea for the $*$ mode

(a) Modified rules and component I simulate the derivation steps made by non-context-free rules in G. That is, $xABy \Rightarrow xCDy$ according to $AB \rightarrow CD \in P$, where $x, y \in (N \cup T)^*$, in G is simulated in Γ

$$\begin{aligned} xABy &\Rightarrow_H xCg(AB \rightarrow CD)By \\ &\Rightarrow_H xCg(AB \rightarrow CD)\,\mathrm{rev}(g(AB \rightarrow CD))Dy \\ &\Rightarrow_I^{2m-1} xCDy. \end{aligned}$$

Component I actually verifies that the simulation of $xABy \Rightarrow xCDy$ is made properly.

(b) Remaining rules simulate the use of context-free rules in G.

Verification Process

Example

Original rule: $AB \rightarrow CD$

Original derivation: $\ldots AB \ldots \Rightarrow \ldots CD \ldots$

Transformed rules: $A \rightarrow C01010001$, $B \rightarrow 10001010D$

Verification process:

$$\ldots AB \ldots$$
$$\ldots C01010001B \ldots$$
$$\ldots C0101000110001010D \ldots$$
$$\ldots C0101000000001010D \ldots$$
$$\ldots C010100001010D \ldots$$
$$\ldots C01011010D \ldots$$
$$\ldots C01000010D \ldots$$
$$\ldots C0110D \ldots$$
$$\ldots C0000D \ldots$$
$$\ldots CD \ldots$$

Verification Code Properties

Case 1—Only one part

…01010001…

Case 2—Wrong order

…10001010 01010001…

Case 3—Partially processed

…01000010…

Case 4—Wrong parts

…010001001010…

Basic Ideas (Transformation 1)

Basic idea for the t mode

Recall that, during the generation of a sentence, a CD grammar system working in the t mode switches its components only if the process is not finished and there are no possible derivations with the previous component.

The first derivation in the t mode has to simulate all rules in G without completing the verification process for non-context-free rules.

Nonetheless, we prove that the verification process can be done successfully afterwards for all simulated rules at once.

Table of Contents

Resulting Properties

Properties of Resulting Systems

- computationally complete
- very reduced number of non-context-free rules
  - these rules are used only for the verification process
  - stored in the separate component
  - the rules are either homogeneous or evenly homogeneous
- *the structure is close to the original grammar*
- *suitable for parallelization*

Resulting Properties

Properties of Resulting Systems

- computationally complete
- very reduced number of non-context-free rules
  - these rules are used only for the verification process
  - stored in the separate component
  - the rules are either homogeneous or evenly homogeneous
- *the structure is close to the original grammar*
- *suitable for parallelization*

Other forms with partially similar properties

- Kuroda/Penttonen Normal Form
- Geffert Normal Forms
- Homogenous Grammars with a Reduced Number of Non-Context-Free Productions (A. Meduna, D. Kolář, 2002)

Resulting Properties

Close Derivation Simulation (the $*$ mode)

Consider grammatical models X and Y. If there is a constant k such that for every derivation of the form

$$x_0 \Rightarrow x_1 \Rightarrow \ldots \Rightarrow x_n$$

in X, where x_0 is its start symbol, there is a derivation of the form

$$x_0 \Rightarrow^{k_1} x_1 \Rightarrow^{k_2} \ldots \Rightarrow^{k_n} x_n$$

in Y, where $k_i \leq k$ for each $1 \leq i \leq n$, we say that Y closely simulates X.

Resulting Properties

Close Derivation Simulation (the $*$ mode)

Consider grammatical models X and Y. If there is a constant k such that for every derivation of the form

$$x_0 \Rightarrow x_1 \Rightarrow \ldots \Rightarrow x_n$$

in X, where x_0 is its start symbol, there is a derivation of the form

$$x_0 \Rightarrow^{k_1} x_1 \Rightarrow^{k_2} \ldots \Rightarrow^{k_n} x_n$$

in Y, where $k_i \leq k$ for each $1 \leq i \leq n$, we say that Y closely simulates X.

Possible Advantages

- we can utilize actions that were coupled with the original rules
- we can check the correctness of the simulation in any stage

Multi-derivation

Informal Definitions

- Multi-derivations are performed so that during a derivation step, the current sentential form may be rewritten at several positions, not just at a single position.
- Uniform derivations always rewrite at all possible positions at once.

Multi-derivation

Informal Definitions

- Multi-derivations are performed so that during a derivation step, the current sentential form may be rewritten at several positions, not just at a single position.
- Uniform derivations always rewrite at all possible positions at once.

Definition

Let Γ be a general CD grammar system, n be a positive integer, and $u_i \Rightarrow_{P_k} v_i$, $1 \leq i \leq n$. Then, Γ makes a direct multi-derivation step from $u_1 u_2 \dots u_n$ to $v_1 v_2 \dots v_n$, symbolically written as
$u_1 u_2 \dots u_n \ {}_{multi}{\Rightarrow}_{P_k} v_1 v_2 \dots v_n$.

Multi-derivation

Informal Definitions

- Multi-derivations are performed so that during a derivation step, the current sentential form may be rewritten at several positions, not just at a single position.
- Uniform derivations always rewrite at all possible positions at once.

Definition

Let Γ be a general CD grammar system, n be a positive integer, and $u_i \Rightarrow_{P_k} v_i$, $1 \leq i \leq n$. Then, Γ makes a direct multi-derivation step from $u_1 u_2 \ldots u_n$ to $v_1 v_2 \ldots v_n$, symbolically written as
$u_1 u_2 \ldots u_n \;{}_{multi}{\Rightarrow}_{P_k} v_1 v_2 \ldots v_n$.

- Both components H and I allow the free use of multi-derivations.
- Multi-derivations cannot disturb the generation process in any way.

Parallelization Problem

Problem

Can we meaningfully parallelize the sentence generation process?

We have

- a very demanding task
- several available processors that we can use to solve the task

We want

- speed up the task
- maximize the use of all available processors
    - the task should be distributed equally across the processors
    - we should keep the synchronization between processors to a minimum
    - each processor should preferably do only simple operations

Parallelization Problem

Case 1

Context-Free Grammars

Solution

1. start with one processor
2. split the task if the sentential form has several nonterminals
3. (re-balance the load)
4. connect the final parts of the sentence

Parallelization Problem

Case 2

General Grammars

Problems

- there is almost no restriction how the left side of the rule can look like
- if we split the sentential form, we need to synchronize the edges

Normal Forms?

- Geffert Normal Forms—cannot be parallelized
- Kuroda Normal Form—more restricted left sides
  - still requires synchronization on the edges
  - number of non-context-free rules is not restricted

Parallelization Problem

Case 3

Transformation 1 with the t mode

Solution

- the task is split into two phases
- in the first phase, H works as a context-free grammar
- in the second phase:
  - $I = \{11 \to 00,\ 0000 \to \varepsilon\}$
  - the synchronization is not needed—we only validate the result
  - we gradually connect partially validated parts

Bibliography

Radim Kocman, Zbyněk Křivka, and Alexander Meduna.
Rule-homogeneous cd grammar systems.
In *AFL 2017 (abstract)*, 2017.

Radim Kocman, Zbyněk Křivka, and Alexander Meduna.
General cd grammar systems and their simplification.
Journal of Automata, Languages and Combinatorics (submitted), 2018?

Thank you!
Any questions?