Decision problems for Clark-congruential languages

Makoto Kanazawa[1] *Tobias Kappé*[2]

[1]Hosei University, Tokyo

[2]University College London

Work performed at the National Institute of Informatics, Tokyo.

ICGI, September 5, 2018

Introduction

Suppose you know the following Japanese phrase:

猫は椅子で眠る　The cat sleeps in the chair.

Introduction

Suppose you know the following Japanese phrase:

猫は椅子で眠る The cat sleeps in the chair.

You also know that *dog* is 犬. Now, you can form:

犬は椅子で眠る The dog sleeps in the chair.

Introduction

This works because 猫 and 犬 are nouns.

Introduction

This works because 猫 and 犬 are nouns.

Replacing nouns (probably) preserves grammatical correctness.

Introduction

This works because 猫 and 犬 are nouns.

Replacing nouns (probably) preserves grammatical correctness.

猫 and 犬 are (almost) *syntactically congruent*:

$$u\text{猫}v \in \text{Japanese} \quad \text{“} \iff \text{”} \quad u\text{犬}v \in \text{Japanese}$$

Introduction

Idea: use syntactic congruence to drive learning.[1]

[1] Clark 2010.

Introduction

Idea: use syntactic congruence to drive learning.[1]

When (for all we know) $uwv \in L \iff uxv \in L$, presume $w \equiv_L x$.

[1] Clark 2010.

Introduction

Idea: use syntactic congruence to drive learning.[1]

When (for all we know) $uwv \in L \iff uxv \in L$, presume $w \equiv_L x$.

...but how to represent the language?

[1] Clark 2010.

Introduction

Definition (Informal)

A grammar is *Clark-congruential* (*CC*) if words derived from the same symbol are syntactically congruent for its language.

A *language* is CC when there exists a CC grammar that describes it.

Introduction

Definition (Informal)

A grammar is *Clark-congruential* (CC) if words derived from the same symbol are syntactically congruent for its language.

A *language* is CC when there exists a CC grammar that describes it.

Example

Consider these grammars for $L = \{a, b\}^+$:

$$\begin{aligned} G_1 : &\quad S \rightarrow SS + a + b \\ G_2 : &\quad S \rightarrow TS + a + b, \quad T \rightarrow a + b + \epsilon \end{aligned}$$

Introduction

Definition (Informal)

A grammar is *Clark-congruential* (*CC*) if words derived from the same symbol are syntactically congruent for its language.

A *language* is CC when there exists a CC grammar that describes it.

Example

Consider these grammars for $L = \{a, b\}^+$:

$$\begin{aligned} G_1 &: \quad S \to SS + a + b \\ G_2 &: \quad S \to TS + a + b, \quad T \to a + b + \epsilon \end{aligned}$$

If S derives w and x in G_1, then $uwv \in L$ implies $uxv \in L$ — G_1 is CC.

Definition (Informal)

A grammar is *Clark-congruential* (*CC*) if words derived from the same symbol are syntactically congruent for its language.

A *language* is CC when there exists a CC grammar that describes it.

Example

Consider these grammars for $L = \{a, b\}^+$:

$$\begin{aligned} G_1 :& \quad S \to SS + a + b \\ G_2 :& \quad S \to TS + a + b, \quad T \to a + b + \epsilon \end{aligned}$$

If S derives w and x in G_1, then $uwv \in L$ implies $uxv \in L$ — G_1 is CC.

However: T derives a and ϵ in G_2. Now, $a \in L$ but $\epsilon \notin L$ — G_2 is not CC.

Introduction

Let G be a CC grammar describing L.

Introduction

Let G be a CC grammar describing L.

In the *minimally adequate teacher* (*MAT*) model, the learner can query:

- Given $w \in \Sigma^*$, does $w \in L(G)$ hold?
- Given a grammar H, does $L(G) = L(H)$ hold? If not, give a counterexample.

Introduction

Let G be a CC grammar describing L.

In the *minimally adequate teacher* (*MAT*) model, the learner can query:

- Given $w \in \Sigma^*$, does $w \in L(G)$ hold?
- Given a grammar H, does $L(G) = L(H)$ hold? If not, give a counterexample.

Theorem (Clark 2010)

Let L be a CC language; L is "MAT-learnable".
That is, given a MAT for L, we can construct a CC grammar for L.

Introduction

Let G be a CC grammar describing L.

In the *minimally adequate teacher* (MAT) model, the learner can query:

- Given $w \in \Sigma^*$, does $w \in L(G)$ hold?
- Given a grammar H, does $L(G) = L(H)$ hold? If not, give a counterexample.

Theorem (Clark 2010)

Let L be a CC language; L is "MAT-learnable".
That is, given a MAT for L, we can construct a CC grammar for L.

Question

Let L be a CC language; is L "MAT-teachable"?
That is, given a CC grammar for L, can we construct a MAT for L?

Introduction

Let G be a CC grammar describing L.

In the *minimally adequate teacher* (MAT) model, the learner can query:

- Given $w \in \Sigma^*$, does $w \in L(G)$ hold?
- Given a grammar H, does $L(G) = L(H)$ hold? If not, give a counterexample.

Is this decidable?

Theorem (Clark 2010)

Let L be a CC language; L is "MAT-learnable".
That is, given a MAT for L, we can construct a CC grammar for L.

Question

Let L be a CC language; is L "MAT-teachable"?
That is, given a CC grammar for L, can we construct a MAT for L?

Context

Equivalence problem

Given grammars G_1 and G_2, does $L(G_1) = L(G_2)$ hold?

[2] Bar-Hillel, Perles, and Shamir 1961.

Context

Equivalence problem

Given grammars G_1 and G_2, does $L(G_1) = L(G_2)$ hold?

Congruence problem

Given a grammar G, and $w, x \in \Sigma^*$, are w and x syntactically congruent for $L(G)$?

[2] Bar-Hillel, Perles, and Shamir 1961.

Context

Equivalence problem

Given grammars G_1 and G_2, does $L(G_1) = L(G_2)$ hold?

Congruence problem

Given a grammar G, and $w, x \in \Sigma^*$, are w and x syntactically congruent for $L(G)$?

Equivalence and congruence are undecidable for grammars in general.[2]

[2] Bar-Hillel, Perles, and Shamir 1961.

Context

Equivalence problem

Given grammars G_1 and G_2, does $L(G_1) = L(G_2)$ hold?

Congruence problem

Given a grammar G, and $w, x \in \Sigma^*$, are w and x syntactically congruent for $L(G)$?

Equivalence and congruence are undecidable for grammars in general.[2]

Recognition problem

Given a class of grammars $\mathcal{C}$ and a grammar G, does G belong to $\mathcal{C}$?

[2] Bar-Hillel, Perles, and Shamir 1961.

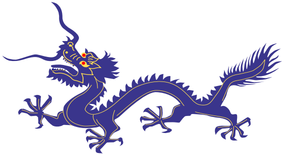
Context

CC languages

Context

Context-free languages

CC languages

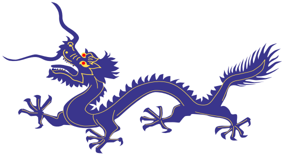


Context

Context-free languages

CC languages

Pre-NTS languages



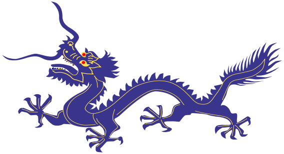
Context

Context-free languages

CC languages

Pre-NTS languages

NTS languages



Context

	Congruence	Equivalence	Recognition
NTS	✓[3]	✓[3]	✓[3,4]
Pre-NTS	✓[5]	✓[5]	✗[6]

[3] Sénizergues 1985.
[4] Engelfriet 1994.
[5] Autebert and Boasson 1992.
[6] Zhang 1992.

Context

	Congruence	Equivalence	Recognition
NTS	✓[3]	✓[3]	✓[3,4]
Pre-NTS	✓[5]	✓[5]	✗[6]
Clark-congruential	✓	✓	†

[3] Sénizergues 1985.
[4] Engelfriet 1994.
[5] Autebert and Boasson 1992.
[6] Zhang 1992.

Preliminaries

A *congruence* on Σ^* is an equivalence $\equiv$ on Σ^* such that

$$\frac{w \equiv w' \qquad x \equiv x'}{wx \equiv w'x'}$$

Preliminaries

A *congruence* on Σ^* is an equivalence $\equiv$ on Σ^* such that

$$\frac{w \equiv w' \qquad x \equiv x'}{wx \equiv w'x'}$$

Every language L induces a *syntactic congruence* $\equiv_L$:

$$\frac{\forall u, v \in \Sigma^*.\ uwv \in L \iff uxv \in L}{w \equiv_L x}$$

Preliminaries

A *Context-Free Grammar* (*CFG*) is a tuple $G = \langle V, \rightarrow, I \rangle$.

$$\frac{\alpha B \gamma \in (\Sigma \cup V)^* \qquad B \rightarrow \beta}{\alpha B \gamma \Rightarrow_G \alpha \beta \gamma}$$

Preliminaries

A *Context-Free Grammar* (*CFG*) is a tuple $G = \langle V, \rightarrow, I \rangle$.

$$\frac{\alpha B \gamma \in (\Sigma \cup V)^* \qquad B \rightarrow \beta}{\alpha B \gamma \Rightarrow_G \alpha \beta \gamma}$$

$$L(G, \alpha) = \{w \in \Sigma^* : \alpha \Rightarrow_G^* w\}$$

Preliminaries

A *Context-Free Grammar* (*CFG*) is a tuple $G = \langle V, \to, I \rangle$.

$$\frac{\alpha B \gamma \in (\Sigma \cup V)^* \qquad B \to \beta}{\alpha B \gamma \Rightarrow_G \alpha \beta \gamma}$$

$$L(G, \alpha) = \{w \in \Sigma^* : \alpha \Rightarrow_G^* w\} \qquad\qquad L(G) = \bigcup_{A \in I} L(G, A)$$

Preliminaries

A *Context-Free Grammar* (*CFG*) is a tuple $G = \langle V, \rightarrow, I \rangle$.

$$\frac{\alpha B \gamma \in (\Sigma \cup V)^* \qquad B \rightarrow \beta}{\alpha B \gamma \Rightarrow_G \alpha \beta \gamma}$$

$$L(G, \alpha) = \{w \in \Sigma^* : \alpha \Rightarrow_G^* w\} \qquad\qquad L(G) = \bigcup_{A \in I} L(G, A)$$

Definition (More formal)

We say G is CC when for $A \in V$ and $w, x \in L(G, A)$, we have $w \equiv_{L(G)} x$.

Preliminaries

We assume a total order $\preceq$ on Σ.

Preliminaries

We assume a total order $\preceq$ on Σ.

This order extends to a total order on Σ^*:

- If w is shorter than x, then $w \preceq x$.
- If w and x are of equal length, compare lexicographically.

Preliminaries

We assume a total order $\preceq$ on Σ.

This order extends to a total order on Σ^*:

- If w is shorter than x, then $w \preceq x$.
- If w and x are of equal length, compare lexicographically.

For $\alpha \in (\Sigma \cup V)^*$ with $L(G, \alpha) \neq \emptyset$, write $\vartheta_G(\alpha)$ for the $\preceq$-minimum of $L(G, \alpha)$.

Deciding congruence

Let G be CC.

We mimic an earlier method to decide congruence.[7]

[7] Autebert and Boasson 1992.

Deciding congruence

Let G be CC.

We mimic an earlier method to decide congruence.[7]

Let $\rightsquigarrow_G$ be the smallest rewriting relation such that

$$\frac{A \to \alpha \qquad L(G, \alpha) \neq \emptyset}{\vartheta_G(\alpha) \rightsquigarrow_G \vartheta_G(A)}$$

[7] Autebert and Boasson 1992.

Deciding congruence

Let G be CC.

We mimic an earlier method to decide congruence.[7]

Let $\rightsquigarrow_G$ be the smallest rewriting relation such that

$$\frac{A \to \alpha \qquad L(G, \alpha) \neq \emptyset}{\vartheta_G(\alpha) \rightsquigarrow_G \vartheta_G(A)}$$

Lemma

If $w \rightsquigarrow_G x$, then $w \equiv_{L(G)} x$.

[7] Autebert and Boasson 1992.

Deciding congruence

Lemma

$w \in L(G)$ if and only if $w \rightsquigarrow_G \vartheta_G(A)$ for some $A \in I$.

Deciding congruence

Lemma

$w \in L(G)$ if and only if $w \rightsquigarrow_G \vartheta_G(A)$ for some $A \in I$.

Example

Let $G = \langle \{S\}, \{S \to SS + (S) + \epsilon\}, \{S\} \rangle$; this grammar is CC.

Deciding congruence

Lemma

$w \in L(G)$ if and only if $w \rightsquigarrow_G \vartheta_G(A)$ for some $A \in I$.

Example

Let $G = \langle \{S\}, \{S \to SS + (S) + \epsilon\}, \{S\} \rangle$; this grammar is CC.

$\rightsquigarrow_G$ is generated by $() \rightsquigarrow_G \epsilon$

$$(()\underline{()})()$$

Deciding congruence

Lemma

$w \in L(G)$ if and only if $w \rightsquigarrow_G \vartheta_G(A)$ for some $A \in I$.

Example

Let $G = \langle \{S\}, \{S \to SS + (S) + \epsilon\}, \{S\} \rangle$; this grammar is CC.

$\rightsquigarrow_G$ is generated by $() \rightsquigarrow_G \epsilon$

$$(()\underline{()})() \rightsquigarrow_G (\underline{()})()$$

Deciding congruence

Lemma

$w \in L(G)$ if and only if $w \rightsquigarrow_G \vartheta_G(A)$ for some $A \in I$.

Example

Let $G = \langle \{S\}, \{S \to SS + (S) + \epsilon\}, \{S\} \rangle$; this grammar is CC.

$\rightsquigarrow_G$ is generated by $() \rightsquigarrow_G \epsilon$

$$(()\underline{()})() \rightsquigarrow_G (\underline{()})() \rightsquigarrow_G ()\underline{()}$$

Deciding congruence

Lemma

$w \in L(G)$ if and only if $w \rightsquigarrow_G \vartheta_G(A)$ for some $A \in I$.

Example

Let $G = \langle \{S\}, \{S \to SS + (S) + \epsilon\}, \{S\} \rangle$; this grammar is CC.

$\rightsquigarrow_G$ is generated by $() \rightsquigarrow_G \epsilon$

$$(()\underline{()})() \rightsquigarrow_G (\underline{()})() \rightsquigarrow_G ()\underline{()} \rightsquigarrow_G \underline{()}$$

Deciding congruence

Lemma

$w \in L(G)$ if and only if $w \rightsquigarrow_G \vartheta_G(A)$ for some $A \in I$.

Example

Let $G = \langle \{S\}, \{S \to SS + (S) + \epsilon\}, \{S\} \rangle$; this grammar is CC.

$\rightsquigarrow_G$ is generated by $() \rightsquigarrow_G \epsilon$

$$(()\underline{()})() \rightsquigarrow_G (\underline{()})() \rightsquigarrow_G ()\underline{()} \rightsquigarrow_G \underline{()} \rightsquigarrow_G \epsilon = \vartheta_G(S)$$

Deciding congruence

Lemma

$w \in L(G)$ if and only if $w \rightsquigarrow_G \vartheta_G(A)$ for some $A \in I$.

Example

Let $G = \langle \{S\}, \{S \to SS + (S) + \epsilon\}, \{S\} \rangle$; this grammar is CC.

$\rightsquigarrow_G$ is generated by $() \rightsquigarrow_G \epsilon$

$$(()\underline{()})() \rightsquigarrow_G (\underline{()})() \rightsquigarrow_G ()\underline{()} \rightsquigarrow_G \underline{()} \rightsquigarrow_G \epsilon = \vartheta_G(S)$$

therefore: $(()())() \in L(G)$.

Deciding congruence

Lemma

$w \in L(G)$ if and only if $w \rightsquigarrow_G \vartheta_G(A)$ for some $A \in I$.

Example

Let $G = \langle \{S\}, \{S \to SS + (S) + \epsilon\}, \{S\} \rangle$; this grammar is CC.

$\rightsquigarrow_G$ is generated by $() \rightsquigarrow_G \epsilon$

$$(()\underline{()})() \rightsquigarrow_G (\underline{()})() \rightsquigarrow_G ()\underline{()} \rightsquigarrow_G \underline{()} \rightsquigarrow_G \epsilon = \vartheta_G(S)$$

therefore: $(()())() \in L(G)$.

From $)()($, we cannot reach ϵ; thus, $)()(\notin L(G)$.

Deciding congruence

Write $\mathcal{I}_G$ for the set of words *irreducible* by $\rightsquigarrow_G$.

Deciding congruence

Write $\mathcal{I}_G$ for the set of words *irreducible* by $\rightsquigarrow_G$.

Lemma

We can create a DPDA M_w *such that* $L(M_w) = \{u\sharp v : uwv \in L(G),\ u, v \in \mathcal{I}_G\}$.

Deciding congruence

Write $\mathcal{I}_G$ for the set of words *irreducible* by $\rightsquigarrow_G$.

Lemma

We can create a DPDA M_w such that $L(M_w) = \{u\sharp v : uwv \in L(G),\ u, v \in \mathcal{I}_G\}$.

Lemma

$L(M_w) = L(M_x)$ if and only if $w \equiv_{L(G)} x$.

Deciding congruence

Write $\mathcal{I}_G$ for the set of words *irreducible* by $\rightsquigarrow_G$.

Lemma

We can create a DPDA M_w *such that* $L(M_w) = \{u\sharp v : uwv \in L(G),\ u, v \in \mathcal{I}_G\}$.

Lemma

$L(M_w) = L(M_x)$ *if and only if* $w \equiv_{L(G)} x$.

Decidable (Sénizergues 1997)

Deciding congruence

Write $\mathcal{I}_G$ for the set of words *irreducible* by $\rightsquigarrow_G$.

Lemma

We can create a DPDA M_w such that $L(M_w) = \{u\sharp v : uwv \in L(G),\ u, v \in \mathcal{I}_G\}$.

Lemma

$L(M_w) = L(M_x)$ if and only if $w \equiv_{L(G)} x$.

Theorem

Let $w, x \in \Sigma^*$. We can decide whether $w \equiv_{L(G)} x$.

Deciding equivalence

Analogous to a result about NTS grammars,[8] we find

Lemma

Let $G_1 = \langle V_1, \to_1, I_1 \rangle$ *and* $G_2 = \langle V_2, \to_2, I_2 \rangle$ *be CC.*

Then $L(G_1) = L(G_2)$ *if and only if*

(i) *for all* $A \in I_1$*, it holds that* $\vartheta_{G_1}(A) \in L(G_2)$ *(and vice versa)*

(ii) *for all pairs* $u \rightsquigarrow_{G_1} v$ *generating* $\rightsquigarrow_{G_1}$*, also* $u \equiv_{L(G_2)} v$ *(and vice versa)*

[8] Sénizergues 1985.

Deciding equivalence

Analogous to a result about NTS grammars,[8] we find

Lemma

Let $G_1 = \langle V_1, \rightarrow_1, I_1 \rangle$ *and* $G_2 = \langle V_2, \rightarrow_2, I_2 \rangle$ *be CC.*

Then $L(G_1) = L(G_2)$ *if and only if*

(i) *for all* $A \in I_1$*, it holds that* $\vartheta_{G_1}(A) \in L(G_2)$ *(and vice versa)*

(ii) *for all pairs* $u \rightsquigarrow_{G_1} v$ *generating* $\rightsquigarrow_{G_1}$*, also* $u \equiv_{L(G_2)} v$ *(and vice versa)*

Finitely many

[8] Sénizergues 1985.

Deciding equivalence

Analogous to a result about NTS grammars,[8] we find

Lemma

Let $G_1 = \langle V_1, \rightarrow_1, I_1 \rangle$ and $G_2 = \langle V_2, \rightarrow_2, I_2 \rangle$ be CC.

Then $L(G_1) = L(G_2)$ if and only if

(i) *for all $A \in I_1$, it holds that $\vartheta_{G_1}(A) \in L(G_2)$* (and vice versa)

(ii) *for all pairs $u \rightsquigarrow_{G_1} v$ generating $\rightsquigarrow_{G_1}$ also $u \equiv_{L(G_2)} v$* (and vice versa)

Decidable

[8] Sénizergues 1985.

Deciding equivalence

Analogous to a result about NTS grammars,[8] we find

Lemma

Let $G_1 = \langle V_1, \rightarrow_1, I_1 \rangle$ *and* $G_2 = \langle V_2, \rightarrow_2, I_2 \rangle$ *be CC.*

Then $L(G_1) = L(G_2)$ *if and only if*

(i) *for all* $A \in I_1$*, it holds that* $\vartheta_{G_1}(A) \in L(G_2)$ (and vice versa)

(ii) *for all pairs* $u \rightsquigarrow_{G_1} v$ *generating* $\rightsquigarrow_{G_1}$*, also* $u \equiv_{L(G_2)} v$ (and vice versa)

Finitely many

[8] Sénizergues 1985.

Deciding equivalence

Analogous to a result about NTS grammars,[8] we find

Lemma

Let $G_1 = \langle V_1, \to_1, I_1 \rangle$ *and* $G_2 = \langle V_2, \to_2, I_2 \rangle$ *be CC.*

Then $L(G_1) = L(G_2)$ *if and only if*

(i) *for all* $A \in I_1$*, it holds that* $\vartheta_{G_1}(A) \in L(G_2)$ *(and vice versa)*

(ii) *for all pairs* $u \rightsquigarrow_{G_1} v$ *generating* $\rightsquigarrow_{G_1}$*, also* $u \equiv_{L(G_2)} v$ *(and vice versa)*

Decidable

[8] Sénizergues 1985.

Deciding equivalence

Analogous to a result about NTS grammars,[8] we find

Lemma

Let $G_1 = \langle V_1, \to_1, I_1 \rangle$ and $G_2 = \langle V_2, \to_2, I_2 \rangle$ be CC.

Then $L(G_1) = L(G_2)$ if and only if

(i) *for all $A \in I_1$, it holds that $\vartheta_{G_1}(A) \in L(G_2)$ (and vice versa)*

(ii) *for all pairs $u \rightsquigarrow_{G_1} v$ generating $\rightsquigarrow_{G_1}$, also $u \equiv_{L(G_2)} v$ (and vice versa)*

Theorem

Let G_1 and G_2 be CC. We can decide whether $L(G_1) = L(G_2)$.

[8] Sénizergues 1985.

Deciding Clark-congruentiality

Given a congruence $\equiv$, we can extend it a congruence $\hat{\equiv}$ on $(\Sigma \cup V)^*$, by stipulating

$$\frac{\vartheta_G(\alpha) \equiv \vartheta_G(\beta)}{\alpha \mathrel{\hat{\equiv}} \beta}$$

Deciding Clark-congruentiality

Given a congruence $\equiv$, we can extend it a congruence $\hat{\equiv}$ on $(\Sigma \cup V)^*$, by stipulating

$$\frac{\vartheta_G(\alpha) \equiv \vartheta_G(\beta)}{\alpha \mathrel{\hat{\equiv}} \beta}$$

Lemma

Let $\equiv$ be a congruence on Σ^*.

The following are equivalent:

(i) *For all $A \in V$ and $w, x \in L(G, A)$, it holds that $w \equiv x$.*

(ii) *For all productions $A \to \alpha$, it holds that $A \mathrel{\hat{\equiv}} \alpha$*

Deciding Clark-congruentiality

Theorem

If $\equiv_{L(G)}$ is decidable, then we can decide whether G is CC.

Proof.

For $A \to \alpha$, check whether $A \mathrel{\hat{\equiv}}_{L(G)} \alpha$, i.e., whether $\vartheta_G(A) \equiv_{L(G)} \vartheta_G(\alpha)$. $\square$

Deciding Clark-congruentiality

Theorem

If $\equiv_{L(G)}$ is decidable, then we can decide whether G is CC.

Proof.

For $A \to \alpha$, check whether $A \mathrel{\hat{\equiv}}_{L(G)} \alpha$, i.e., whether $\vartheta_G(A) \equiv_{L(G)} \vartheta_G(\alpha)$. $\square$

Corollary

If $L(G)$ is a deterministic CFL, then it is decidable whether G is CC.

Conclusion

So, are CC languages “MAT-teachable”?

Conclusion

So, are CC languages "MAT-teachable"?

Yes. . . but there is a slight mismatch:

- (Clark 2010) assumes an *extended* MAT.
- That is, hypothesis grammars may not be CC!

Conclusion

So, are CC languages "MAT-teachable"?

Yes. . . but there is a slight mismatch:

- (Clark 2010) assumes an *extended* MAT.
- That is, hypothesis grammars may not be CC!

Two plausible fixes:

- Adjust learning algorithm to have CC grammars as hypotheses.
- Extend decision procedure, requiring only one grammar to be CC.

Further work

Many open questions:

- Are CC grammars more expressive than pre-NTS grammars?

Further work

Many open questions:

- Are CC grammars more expressive than pre-NTS grammars?
- Is the language of every CC grammar a DCFL?

Further work

Many open questions:

- Are CC grammars more expressive than pre-NTS grammars?
- Is the language of every CC grammar a DCFL?
- Is it decidable whether a given grammar is CC in general?

Bonus: grammar to DPDA

Lemma

Let G be CC, and let R be regular.

We can create a CC grammar G_R such that $L(G_R) = L(G) \cap R$.

Bonus: grammar to DPDA

Lemma

Let G be CC, and let R be regular.

We can create a CC grammar G_R such that $L(G_R) = L(G) \cap R$.

Lemma

Let $h : \Sigma^* \to \Sigma^*$ be a strictly alphabetic morphism, that is, $h(a) \in \Sigma$ for all $a \in \Sigma$.

We can create a CC grammar G^h such that $L(G^h) = h^{-1}(L(G))$.

Bonus: grammar to DPDA

For $a \in \Sigma$, add $\bar{a}$ to Σ.

Let $h : \Sigma \to \Sigma$ be such that $h(a) = h(\bar{a}) = a$.

Create G^h such that $L(G^h) = h^{-1}(L(G))$.

Bonus: grammar to DPDA

For $a \in \Sigma$, add $\bar{a}$ to Σ.

Let $h : \Sigma \to \Sigma$ be such that $h(a) = h(\bar{a}) = a$.

Create G^h such that $L(G^h) = h^{-1}(L(G))$.

Intuition

G^h is the same as G, but positions in every word can be "marked" by $\bar{}$.

Bonus: grammar to DPDA

Note that $\mathcal{I}_G$ is a regular language.

Create G_w such that $L(G_w) = L(G^h) \cap \mathcal{I}_G \bar{w} \mathcal{I}_G$.

Now $G_w = \{u\bar{w}v : uwv \in L(G),\ u, v \in \mathcal{I}_G\}$.

Bonus: grammar to DPDA

Note that $\mathcal{I}_G$ is a regular language.

Create G_w such that $L(G_w) = L(G^h) \cap \mathcal{I}_G \bar{w} \mathcal{I}_G$.

Now $G_w = \{u\bar{w}v : uwv \in L(G),\ u, v \in \mathcal{I}_G\}$.

Intuition

$L(G_w)$ has words in $L(G)$ with w as a marked substring, with context reduced by $\rightsquigarrow_G$.

Bonus: grammar to DPDA

Lemma

Without loss of generality, every rule generating $\rightsquigarrow_{G_w}$ *overlaps and preserves* $\bar{w}$.

Bonus: grammar to DPDA

Lemma

Without loss of generality, every rule generating $\rightsquigarrow_{G_w}$ overlaps and preserves $\bar{w}$.

We can now create a reduction $\rightsquigarrow_{G[w]}$ and a finite set S_w such that

- Every rule generating $\rightsquigarrow_{G[w]}$ contains and preserves $\sharp$.
- $\{x \in \Sigma^* : x \rightsquigarrow_{G[w]} y \in S_w\} = \{u\sharp v : uwv \in L(G),\ u, v \in \mathcal{I}_G\}$

Bonus: grammar to DPDA

Lemma

Without loss of generality, every rule generating $\rightsquigarrow_{G_w}$ overlaps and preserves $\bar{w}$.

We can now create a reduction $\rightsquigarrow_{G[w]}$ and a finite set S_w such that

- Every rule generating $\rightsquigarrow_{G[w]}$ contains and preserves $\sharp$.
- $\{x \in \Sigma^* : x \rightsquigarrow_{G[w]} y \in S_w\} = \{u\sharp v : uwv \in L(G),\ u, v \in \mathcal{I}_G\}$

The DPDA M_w acts by reading $u\sharp v$ up to $\sharp$, putting the input on the stack. Then:

Bonus: grammar to DPDA

Lemma

Without loss of generality, every rule generating $\rightsquigarrow_{G_w}$ overlaps and preserves $\bar{w}$.

We can now create a reduction $\rightsquigarrow_{G[w]}$ and a finite set S_w such that

- Every rule generating $\rightsquigarrow_{G[w]}$ contains and preserves $\sharp$.
- $\{x \in \Sigma^* : x \rightsquigarrow_{G[w]} y \in S_w\} = \{u\sharp v : uwv \in L(G),\ u, v \in \mathcal{I}_G\}$

The DPDA M_w acts by reading $u\sharp v$ up to $\sharp$, putting the input on the stack. Then:

- Pop from the stack or read from input into two buffers (encoded in state).

Bonus: grammar to DPDA

Lemma

Without loss of generality, every rule generating $\rightsquigarrow_{G_w}$ overlaps and preserves $\bar{w}$.

We can now create a reduction $\rightsquigarrow_{G[w]}$ and a finite set S_w such that

- Every rule generating $\rightsquigarrow_{G[w]}$ contains and preserves $\sharp$.
- $\{x \in \Sigma^* : x \rightsquigarrow_{G[w]} y \in S_w\} = \{u\sharp v : uwv \in L(G),\ u, v \in \mathcal{I}_G\}$

The DPDA M_w acts by reading $u\sharp v$ up to $\sharp$, putting the input on the stack. Then:

- Pop from the stack or read from input into two buffers (encoded in state).
- Whenever possible, reduce according to the rules from $\rightsquigarrow_{G[w]}$.

Bonus: grammar to DPDA

Lemma

Without loss of generality, every rule generating $\rightsquigarrow_{G_w}$ overlaps and preserves $\bar{w}$.

We can now create a reduction $\rightsquigarrow_{G[w]}$ and a finite set S_w such that

- Every rule generating $\rightsquigarrow_{G[w]}$ contains and preserves $\sharp$.
- $\{x \in \Sigma^* : x \rightsquigarrow_{G[w]} y \in S_w\} = \{u\sharp v : uwv \in L(G),\ u, v \in \mathcal{I}_G\}$

The DPDA M_w acts by reading $u\sharp v$ up to $\sharp$, putting the input on the stack. Then:

- Pop from the stack or read from input into two buffers (encoded in state).
- Whenever possible, reduce according to the rules from $\rightsquigarrow_{G[w]}$.
- When the buffer resembles S_w and the input and stack are empty, accept.

Bonus: grammar to DPDA

Lemma

Without loss of generality, every rule generating $\rightsquigarrow_{G_w}$ overlaps and preserves $\bar{w}$.

We can now create a reduction $\rightsquigarrow_{G[w]}$ and a finite set S_w such that

- Every rule generating $\rightsquigarrow_{G[w]}$ contains and preserves $\sharp$.
- $\{x \in \Sigma^* : x \rightsquigarrow_{G[w]} y \in S_w\} = \{u\sharp v : uwv \in L(G),\ u, v \in \mathcal{I}_G\}$

The DPDA M_w acts by reading $u\sharp v$ up to $\sharp$, putting the input on the stack. Then:

- Pop from the stack or read from input into two buffers (encoded in state).
- Whenever possible, reduce according to the rules from $\rightsquigarrow_{G[w]}$.
- When the buffer resembles S_w and the input and stack are empty, accept.

With some analysis, we find that $L(M_w) = \{u\sharp v : uwv \in L(G),\ u, v \in \mathcal{I}_G\}$.