Decision problems for Clark-congruential languages

Makoto Kanazawa\textsuperscript{1} \quad Tobias Kappé\textsuperscript{2}

\textsuperscript{1}Hosei University, Tokyo
\textsuperscript{2}University College London

Work performed at the National Institute of Informatics, Tokyo.

ICGI, September 5, 2018
Suppose you know the following Japanese phrase:

猫は椅子で眠る  The cat sleeps in the chair.
Suppose you know the following Japanese phrase:

猫は椅子で眠る \( \text{The cat sleeps in the chair.} \)

You also know that \textit{dog} is 犬. Now, you can form:

犬は椅子で眠る \( \text{The dog sleeps in the chair.} \)
This works because 猫 and 犬 are nouns.
This works because 猫 and 犬 are nouns.

Replacing nouns (probably) preserves grammatical correctness.
This works because 猫 and 犬 are nouns.

Replacing nouns (probably) preserves grammatical correctness.

猫 and 犬 are (almost) syntactically congruent:

\[ u\text{猫}v \in \text{Japanese} \quad “ \iff ” \quad u\text{犬}v \in \text{Japanese} \]
Idea: use syntactic congruence to drive learning.\textsuperscript{1}

\textsuperscript{1}Clark 2010.
Introduction

Idea: use syntactic congruence to drive learning.¹

When (for all we know) \( u w v \in L \iff u x v \in L \), presume \( w \equiv_{L} x \).

¹Clark 2010.
Idea: use syntactic congruence to drive learning.\textsuperscript{1}

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

...but how to represent the language?

\textsuperscript{1}Clark 2010.
Definition (Informal)

A grammar is *Clark-congruent* (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.
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 \( L = \{a, b\}^+ \):

\[
G_1 : \quad S \rightarrow SS + a + b \\
G_2 : \quad S \rightarrow TS + a + b, \quad T \rightarrow a + b + \epsilon
\]
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 $L = \{a, b\}^+$:

$$
G_1 : \quad S \rightarrow SS + a + b \\
G_2 : \quad S \rightarrow TS + a + b, \quad T \rightarrow a + b + \epsilon
$$

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 \( L = \{a, b\}^+ \):

\[
\begin{align*}
G_1 : & \quad S \rightarrow SS + a + b \\
G_2 : & \quad S \rightarrow TS + a + b, \quad T \rightarrow a + b + \epsilon
\end{align*}
\]

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 \( \epsilon \) in \( G_2 \). Now, \( a \in L \) but \( \epsilon \notin L \) — \( G_2 \) is not CC.
Let $G$ be a CC grammar describing $L$. 

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 $L$.

Is this decidable?

*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 $L$?
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 \textit{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.

\textbf{Theorem (Clark 2010)}

\textit{Let $L$ be a CC language; $L$ is \textit{“MAT-learnable”}. That is, given a MAT for $L$, we can construct a CC grammar for $L$.}

\textbf{Question}

Let $L$ be a CC language; is $L$ \textit{“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.

**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?"*
Equivalence problem

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

Warning: Equivalence and congruence are undecidable for grammars in general.

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 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.\(^2\)

---

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 $L(G)$?

Recognition problem
Given a class of grammars $C$ and a grammar $G$, does $G$ belong to $C$?

Equivalence and congruence are undecidable for grammars in general.\(^2\)

CC languages
Context-free languages

CC languages
Context-free languages

CC languages

Pre-NTS languages
Context-free languages

CC languages

Pre-NTS languages

NTS languages
### Context

<table>
<thead>
<tr>
<th></th>
<th>Congruence</th>
<th>Equivalence</th>
<th>Recognition</th>
</tr>
</thead>
<tbody>
<tr>
<td>NTS</td>
<td>✓3</td>
<td>✓3</td>
<td>✓3,4</td>
</tr>
<tr>
<td>Pre-NTS</td>
<td>✓5</td>
<td>✓5</td>
<td>✗6</td>
</tr>
</tbody>
</table>

---

3 Sénizergues 1985.
5 Autebert and Boasson 1992.
<table>
<thead>
<tr>
<th></th>
<th>Congruence</th>
<th>Equivalence</th>
<th>Recognition</th>
</tr>
</thead>
<tbody>
<tr>
<td>NTS</td>
<td>✓3</td>
<td>✓3</td>
<td>✓3,4</td>
</tr>
<tr>
<td>Pre-NTS</td>
<td>✓5</td>
<td>✓5</td>
<td>✗6</td>
</tr>
<tr>
<td>Clark-congruential</td>
<td>✓</td>
<td>✓</td>
<td>✠</td>
</tr>
</tbody>
</table>

---

3 Sénizergues 1985.
5 Autebert and Boasson 1992.
A congruence on $\Sigma^*$ is an equivalence $\equiv$ on $\Sigma^*$ such that

\[
\begin{align*}
    w \equiv w' & \quad x \equiv x' \\
    \hline \\
    wx \equiv w'x' 
\end{align*}
\]
A congruence on $\Sigma^*$ is an equivalence $\equiv$ on $\Sigma^*$ such that

$$w \equiv w' \quad x \equiv x' \quad \Rightarrow \quad wx \equiv w'x'$$

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

$$\forall u, v \in \Sigma^*. \quad uwv \in L \iff uxv \in L\quad \Rightarrow \quad w \equiv_L x$$
A Context-Free Grammar (CFG) is a tuple $G = \langle V, \rightarrow, I \rangle$.

\[
\alpha B \gamma \in (\Sigma \cup V)^* \quad B \rightarrow \beta
\]

\[
\alpha B \gamma \Rightarrow_G \alpha \beta \gamma
\]
A Context-Free Grammar (CFG) is a tuple $G = \langle V, \rightarrow, I \rangle$.

$$\alpha B\gamma \in (\Sigma \cup V)^* \quad B \rightarrow \beta$$

$$\alpha B\gamma \Rightarrow_G \alpha\beta\gamma$$

$L(G, \alpha) = \{w \in \Sigma^* : \alpha \Rightarrow_g^* w\}$
A Context-Free Grammar (CFG) is a tuple $G = \langle V, \rightarrow, I \rangle$. 

\[
\alpha B\gamma \in (\Sigma \cup V)^* \quad B \rightarrow \beta \\
\alpha B\gamma \Rightarrow_G \alpha\beta\gamma 
\]

\[
L(G, \alpha) = \{ w \in \Sigma^* : \alpha \Rightarrow_G^* w \} \\
L(G) = \bigcup_{A \in I} L(G, A)
\]
A Context-Free Grammar (CFG) is a tuple $G = \langle V, \rightarrow, I \rangle$.

$$
\alpha B\gamma \in (\Sigma \cup V)^* \quad B \rightarrow \beta \\
\alpha B\gamma \Rightarrow_G \alpha\beta\gamma
$$

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

Definition (More formal)

We say $G$ is CC when for $A \in V$ and $w, x \in L(G, A)$, we have $w \equiv_{L(G)} x$. 
We assume a total order \( \preceq \) on \( \Sigma \).
We assume a total order $\preceq$ on $\Sigma$. This order extends to a total order on $\Sigma^*$:

- If $w$ is shorter than $x$, then $w \preceq x$.
- If $w$ and $x$ are of equal length, compare lexicographically.
We assume a total order $\preceq$ on $\Sigma$.

This order extends to a total order on $\Sigma^*$:

- 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 $L(G, \alpha)$. 
Let $G$ be CC.

We mimic an earlier method to decide congruence.\(^7\)

\(^7\)Autebert and Boasson 1992.
Deciding congruence

Let $G$ be $\text{CC}$. We mimic an earlier method to decide congruence.\(^7\)

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

\[
A \rightarrow \alpha \quad L(G, \alpha) \neq \emptyset
\]

\[
\vartheta_G(\alpha) \sim_G \vartheta_G(A)
\]

\(^7\)Autebert and Boasson 1992.
Let $G$ be CC.

We mimic an earlier method to decide congruence.$^7$

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

$$A \rightarrow \alpha \quad L(G, \alpha) \neq \emptyset$$

$$\vartheta_G(\alpha) \sim_G \vartheta_G(A)$$

**Lemma**

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

---

Deciding congruence

Lemma

\[ w \in L(G) \text{ if and only if } w \rightsquigarrow_G \emptyset_G(A) \text{ for some } A \in I. \]
Deciding congruence

Lemma

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

Example

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

Lemma

\[ w \in L(G) \text{ if and only if } w \xrightarrow{\sim} G \vartheta_G(A) \text{ for some } A \in I. \]

Example

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

\( \sim \rho_G \) is generated by \( \epsilon \)

\[ \epsilon = (())()() \]
Deciding congruence

Lemma

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

Example

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

\( \sim_G \) is generated by \( (\) \sim_G \epsilon \)

\[ ((\,) (\,) (\,) \sim_G (\,) (\,) (\,) \]
Deciding congruence

Lemma

\[ w \in L(G) \text{ if and only if } w \leadsto_G \varnothing_G(A) \text{ for some } A \in I. \]

Example

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

\( \leadsto_G \) is generated by \( (\leadsto_G \epsilon) \)

\[ \text{(()()()()) } \leadsto_G \text{ (())() } \leadsto_G \text{ ()() } \]
Deciding congruence

**Lemma**

\[ w \in L(G) \text{ if and only if } w \rightsquigarrow_G \varnothing G(A) \text{ for some } A \in I. \]

**Example**

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

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

\[
(()()) () \rightsquigarrow_G (()) () \rightsquigarrow_G () () \rightsquigarrow_G ()
\]
Lemma

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

Example

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

\( \mapsto_G \) is generated by (\) \( \mapsto_G \epsilon \)

\[
( ( ) ) ( ) \mapsto_G ( ) ( ) \mapsto_G ( ) ( ) \mapsto_G ( ) \mapsto_G ( ) \mapsto_G \epsilon = \vartheta_G(S)
\]
Deciding congruence

Lemma

\[ w \in L(G) \text{ if and only if } w \stackrel{\varnothing}{\Rightarrow}_G \varnothing_G(A) \text{ for some } A \in I. \]

Example

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

\( \Rightarrow_G \) is generated by ( ) \( \Rightarrow_G \epsilon \)

\[
\begin{align*}
( ()()() & \Rightarrow_G ()()() \Rightarrow_G ()() \Rightarrow_G () \Rightarrow_G \epsilon = \varnothing_G(S) \\
\end{align*}
\]

Therefore: \( ( ()()() ) \in L(G) \).
Deciding congruence

Lemma

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

Example

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

\( \sim_G \) is generated by \( () \sim_G \epsilon \)

\[
(()()) () \sim_G (()) () \sim_G () () \sim_G () \sim_G \epsilon = \vartheta_G(S)
\]

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

From \( () () () \), we cannot reach \( \epsilon \); thus, \( () () () \not\in L(G) \).
Write $\mathcal{I}_G$ for the set of words \emph{irreducible} by $\sim_G$. 
Write $\mathcal{I}_G$ for the set of words irreducible by $\sim_G$.

**Lemma**

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

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

\textbf{Lemma}

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

\textbf{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 $\sim_G$.

Lemma

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

Lemma

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

Decidable (Sénizergues 1997)
Write $\mathcal{I}_G$ for the set of words irreducible by $\sim_G$.

**Lemma**

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

**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 \(G_1 = \langle V_1, \rightarrow_1, I_1 \rangle\) 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 \(u \leadsto_{G_1} v\) generating \(\leadsto_{G_1}\), 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 \(u \overset{\sim}{\rightarrow}_G v\) generating \(\sim_{G_1}\), also \(u \equiv_{L(G_2)} v\) (and vice versa)

---

\(^8\)Sénizergues 1985.
Deciding equivalence

Analogous to a result about NTS grammars, 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 $u \xrightarrow{G_1} v$ generating $\rightarrow_{G_1}$, also $u \equiv_{L(G_2)} v$ (and vice versa)

---

8 Sénizergues 1985.
Deciding equivalence

Analogous to a result about NTS grammars, 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 \( \varphi_{G_1}(A) \in L(G_2) \) (and vice versa)

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

Finitely many

---

\(^8\)Sénizergues 1985.
Deciding equivalence

Analogous to a result about NTS grammars, 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 $u \Rightarrow_G v$ generating $\Rightarrow_{G_1}$, also $u \equiv_{L(G_2)} v$ (and vice versa)

\[ \text{Decidable} \]

---

---

\[ ^8 \text{Sénizergues 1985.} \]
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 \(u \xrightarrow{G_1} v\) generating \(\xrightarrow{G_1}\), 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 $\widehat{\equiv}$ on $(\Sigma \cup V)^*$, by stipulating

$$\vartheta_G(\alpha) \equiv \vartheta_G(\beta) \quad \frac{\alpha \widehat{\equiv} \beta}{\alpha \widehat{\equiv} \beta}$$
Deciding Clark-congruentiality

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

$$\vartheta_G(\alpha) \equiv \vartheta_G(\beta) \quad \Rightarrow \quad \alpha \hat{\equiv} \beta$$

**Lemma**

Let $\equiv$ be a congruence on $\Sigma^*$.

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 $A \rightarrow \alpha$, it holds that $A \hat{\equiv} \alpha$
Theorem

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

Proof.

For $A \rightarrow \alpha$, check whether $A \overset{\equiv}{L(G)} \alpha$, i.e., whether $\vartheta_G(A) \equiv_{L(G)} \vartheta_G(\alpha)$. \qed
Deciding Clark-congruentiality

**Theorem**

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

**Proof.**

For $A \rightarrow \alpha$, check whether $A \equiv_{L(G)} \alpha$, i.e., whether $\vartheta_G(A) \equiv_{L(G)} \vartheta_G(\alpha)$. □

**Corollary**

If $L(G)$ is a deterministic CFL, then it is decidable whether $G$ is CC.
So, are CC languages “MAT-teachable”?
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!
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.
Many open questions:

- Are CC grammars more expressive than pre-NTS grammars?
Many open questions:

- Are CC grammars more expressive than pre-NTS grammars?
- Is the language of every CC grammar a DCFL?
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?
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 $G$ 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 $h : \Sigma^* \rightarrow \Sigma^*$ 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 $L(G^h) = h^{-1}(L(G))$. 
For $a \in \Sigma$, add $\bar{a}$ to $\Sigma$.

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

Create $G^h$ such that $L(G^h) = h^{-1}(L(G))$. 
For $a \in \Sigma$, add $\bar{a}$ to $\Sigma$.

Let $h : \Sigma \rightarrow \Sigma$ 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 $\bar{\cdot}$.
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 $G_w = \{u\bar{w}v : uwv \in L(G), \ u, v \in \mathcal{I}_G\}$. 
Note that $\mathcal{I}_G$ is a regular language.

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

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

**Intuition**

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

Without loss of generality, every rule generating $\sim_{G_w}$ overlaps and preserves $\tilde{w}$.
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 *\( \# \).*
- \( \{ x \in \Sigma^* : x \rightsquigarrow G[w] \ y \in S_w \} = \{ u\#v : uwv \in L(G), u, v \in \mathcal{I}_G \} \)
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 $\#$.
- $\{x \in \Sigma^* : x \rightsquigarrow_{G[w]} y \in S_w\} = \{u\#v : uwv \in L(G), u, v \in I_G\}$

The DPDA $M_w$ acts by reading $u\#v$ up to $\#$, 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 \( \# \).
- \( \{ x \in \Sigma^* : x \rightsquigarrow_{G[w]} y \in S_w \} = \{ u\#v : uwv \in L(G), u, v \in I_G \} \)

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

- Pop from the stack or read from input into two buffers (encoded in state).
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 $\#$.
- $\{x \in \Sigma^* : x \rightsquigarrow_{G[w]} y \in S_w\} = \{u\#v : uwv \in L(G), u, v \in I_G\}$

The DPDA $M_w$ acts by reading $u\#v$ up to $\#$, 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 $\rightsquigarrow_{G[w]}$. 
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 $\#$.
- $\{x \in \Sigma^* : x \rightsquigarrow_{G[w]} y \in S_w\} = \{u\#v : uwv \in L(G), u, v \in I_G\}$

The DPDA $M_w$ acts by reading $u\#v$ up to $\#$, 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.
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 \( \# \).
- \( \{ x \in \Sigma^* : x \rightsquigarrow_{G[w]} y \in S_w \} = \{ uv : u\#v \in L(G), u, v \in I_G \} \)

The DPDA \( M_w \) acts by reading \( u\#v \) up to \( \# \), 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 \( L(M_w) = \{ u\#v : u\#v \in L(G), u, v \in I_G \} \).