D^3 as a 2-MCFL

Orestis Melkonian, Konstantinos Kogkalidis

August 8, 2018

Utrecht University

2-MCFG

Generalization of the CFG over tuples of strings

Introduction/Definition

2-MCFG

Generalization of the CFG over tuples of strings

N-dimensional Dyck Language D^N

Defined over an ordered alphabet of N symbols:
$\{\alpha_1 < \cdots < \alpha_N\}$ s.t. words satisfy two conditions:

1. Equal number of occurrences of all alphabet symbols
2. Any prefix of a word must contain at least as many α_i as $\alpha_{i+1} \quad \forall\, i \ \leq N-1$

D^3 - Some examples

Dyck words

- abc
- aabbcc
- abcabcabacbc

D^3 - Some examples

Dyck words

- abc
- aabbcc
- abcabcabacbc

Non-dyck words

- aabb

D^3 - Some examples

Dyck words

- abc
- aabbcc
- abcabcabacbc

Non-dyck words

- aabb
- aabbbcc

D^3 - Some examples

Dyck words

- abc
- aabbcc
- abcabcabacbc

Non-dyck words

- aabb
- aabbbcc
- abcacb

D^3 - Some examples

Dyck words

- abc
- aabbcc
- abcabcabacbc

Non-dyck words

- aabb
- aabbbcc
- abcacb

ababacbcabcc

First-match policy

D^3 - Some examples

Dyck words

- abc
- aabbcc
- abcabcabacbc

Non-dyck words

- aabb
- aabbbcc
- abcacb

First-match policy

D^3 - Some examples

Dyck words

- abc
- aabbcc
- abcabcabacbc

Non-dyck words

- aabb
- aabbbcc
- abcacb

First-match policy

D^3 - Some examples

Dyck words

- abc
- aabbcc
- abcabcabacbc

Non-dyck words

- aabb
- aabbbcc
- abcacb

First-match policy

D^3 - Some examples

Dyck words

- abc
- aabbcc
- abcabcabacbc

Non-dyck words

- aabb
- aabbbcc
- abcacb

First-match policy

Motivation

Natural Languages

Free word order respecting linear order constraints

Motivation

Natural Languages

Free word order respecting linear order constraints

Programming Languages

Static Analysis of non-standard control flows (e.g. *yield*)

G_0: Grammar of triple insertions

$$\mathsf{S}(xy) \leftarrow \mathsf{W}(x, y). \tag{1}$$

$$\mathsf{W}(\epsilon, xy\mathbf{abc}) \leftarrow \mathsf{W}(x, y). \tag{2}$$

$$\mathsf{W}(\epsilon, x\mathbf{a}y\mathbf{bc}) \leftarrow \mathsf{W}(x, y). \tag{3}$$

$$\ldots$$

$$\mathsf{W}(\mathbf{ab}x\mathbf{c}y, \epsilon) \leftarrow \mathsf{W}(x, y). \tag{60}$$

$$\mathsf{W}(\mathbf{abc}xy, \epsilon) \leftarrow \mathsf{W}(x, y). \tag{61}$$

$$\mathsf{W}(\epsilon, \mathbf{abc}). \tag{62}$$

$$\mathsf{W}(\mathbf{a}, \mathbf{bc}). \tag{63}$$

$$\mathsf{W}(\mathbf{ab}, \mathbf{c}). \tag{64}$$

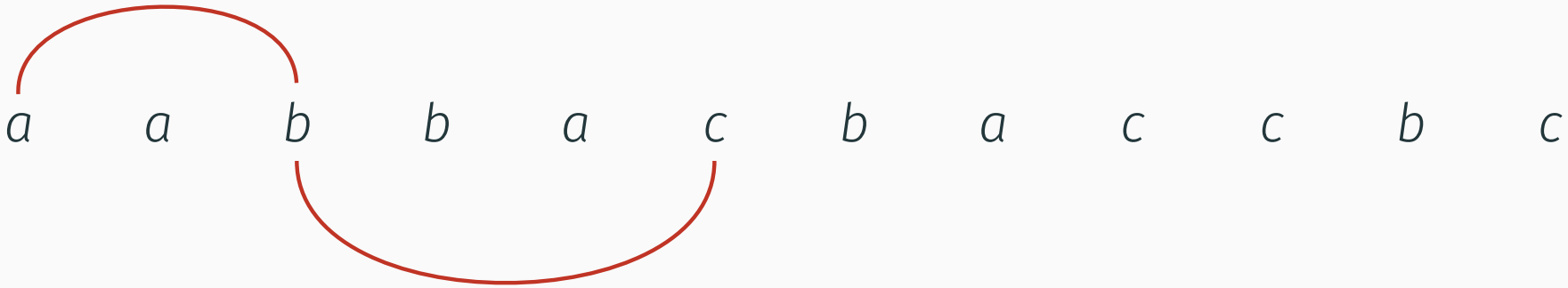
$$\mathsf{W}(\mathbf{abc}, \epsilon). \tag{65}$$

G_0: Grammar of triple insertions

$$a \quad a \quad b \quad b \quad a \quad c \quad b \quad a \quad c \quad c \quad b \quad c$$

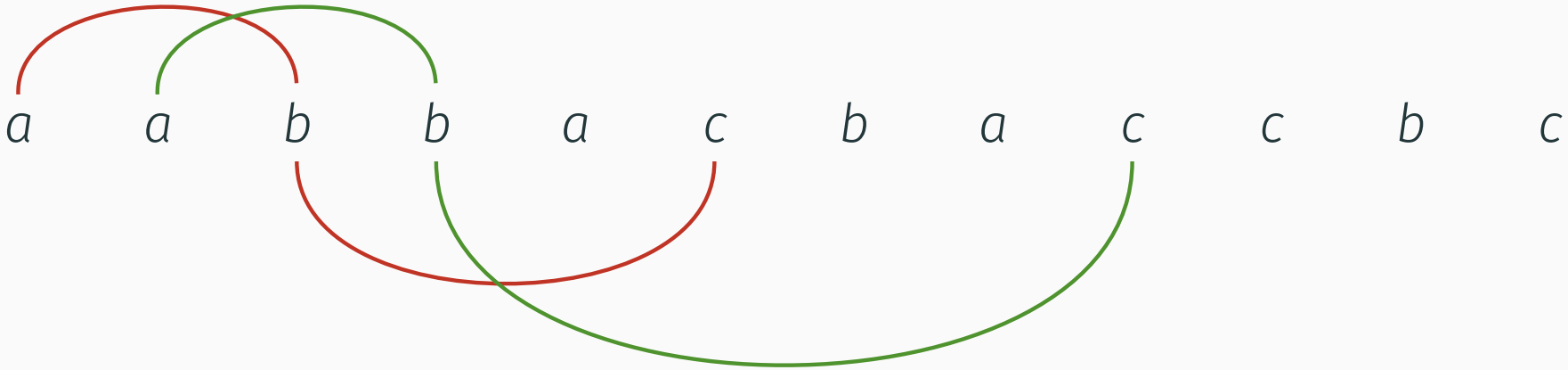
Straddling counter-example

G_0: Grammar of triple insertions



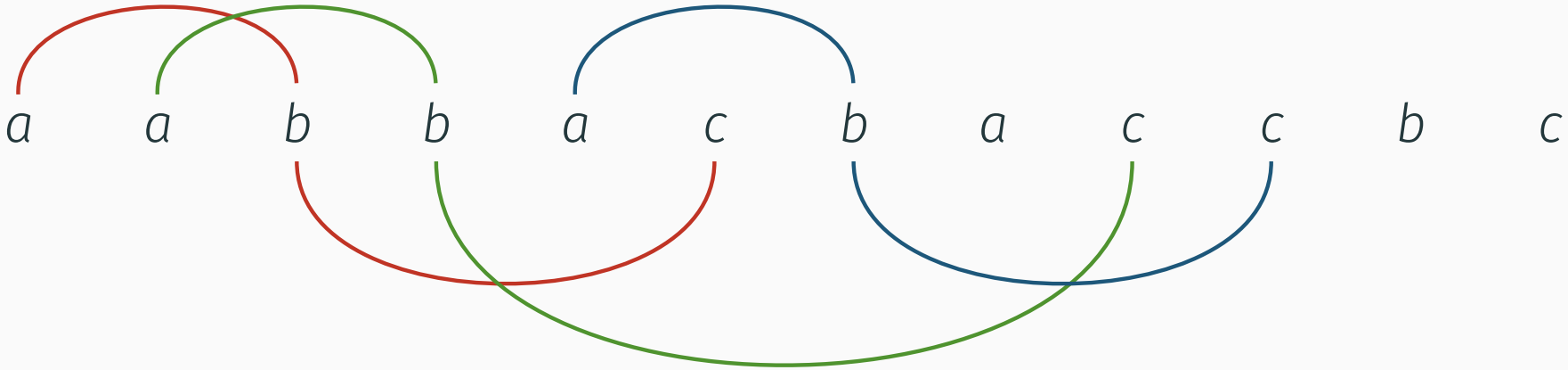
Straddling counter-example

G_0: Grammar of triple insertions



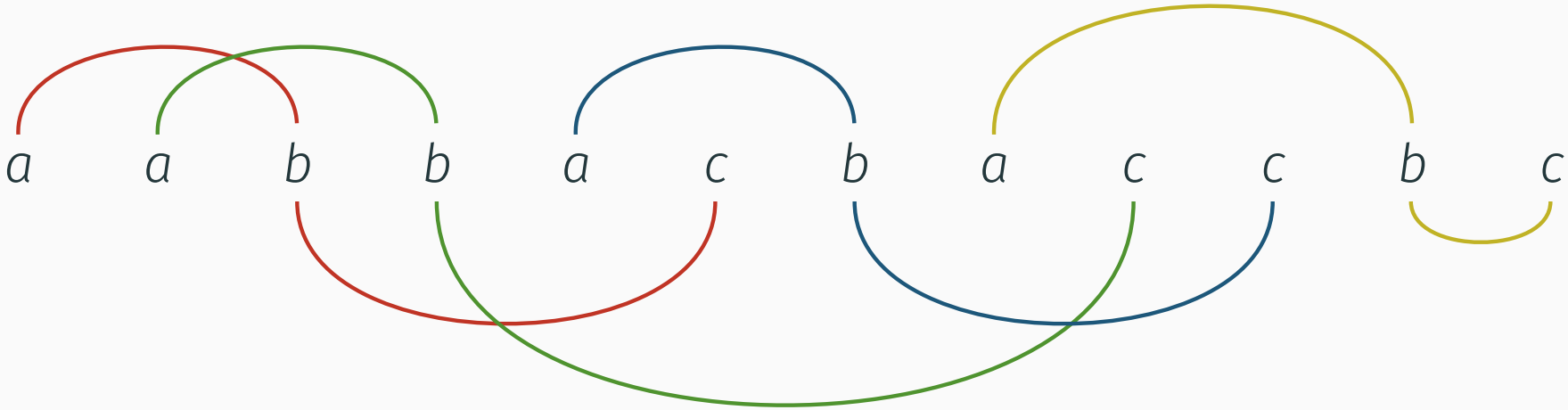
Straddling counter-example

G_0: Grammar of triple insertions



Straddling counter-example

G_0: Grammar of triple insertions



Straddling counter-example

Meta-grammars: Introduction

Notation

$\mathcal{O}_m[\![\text{conclusion} \leftarrow \text{premises} \mid \text{partial orders}]\!]$.

Meta-grammars: Introduction

Notation

$$\mathcal{O}_m[\![\text{conclusion} \leftarrow \text{premises} \mid \text{partial orders}]\!].$$

Meta-grammar G_1

$$\left.\begin{array}{l} \mathcal{O}_2[\![W \leftarrow \epsilon \mid \{a < b < c\}]\!]. \\ \mathcal{O}_2[\![W \leftarrow W_{xy} \mid \{x < y,\ a < b < c\}]\!]. \end{array}\right\} \text{ TRIPLE INSERTION}$$

Meta-grammars: Introduction

Notation

$$\mathcal{O}_m[\![\text{conclusion} \leftarrow \text{premises} \mid \text{partial orders}]\!].$$

Meta-grammar G_1

$$\left.\begin{array}{l} \mathcal{O}_2[\![W \leftarrow \epsilon \mid \{a < b < c\}]\!]. \\ \mathcal{O}_2[\![W \leftarrow W_{xy} \mid \{x < y,\ a < b < c\}]\!]. \end{array}\right\} \text{ triple insertion}$$

$$+$$

$$\left.\mathcal{O}_2[\![W \leftarrow W_{xy}, W_{zw} \mid \{x < y,\ z < w\}]\!].\right\} \text{ interleaving words}$$

G_2: Adding states

$$\left.\begin{array}{l}\mathcal{O}_2[\![A^+ \leftarrow \epsilon \mid \{a\}]\!].\\ \mathcal{O}_2[\![B^+ \leftarrow \epsilon \mid \{b\}]\!].\\ \mathcal{O}_2[\![C^+ \leftarrow \epsilon \mid \{c\}]\!].\end{array}\right\} \text{BASE CASES}$$

$$\left.\begin{array}{l}\mathcal{O}_2[\![C^- \leftarrow A^+, B^+ \mid \{x < y < z < w\}]\!].\\ \mathcal{O}_2[\![B^- \leftarrow A^+, C^+ \mid \{x < y < z < w\}]\!].\\ \mathcal{O}_2[\![A^- \leftarrow B^+, C^+ \mid \{x < y < z < w\}]\!].\\ \mathcal{O}_2[\![A^+ \leftarrow C^-, B^- \mid \{x < y < z < w\}]\!].\\ \mathcal{O}_2[\![B^+ \leftarrow C^-, A^- \mid \{x < y < z < w\}]\!].\\ \mathcal{O}_2[\![C^+ \leftarrow B^-, A^- \mid \{x < y < z < w\}]\!].\\ \forall\, \mathrm{K} \in \mathcal{S} \setminus \mathrm{W} :\ \mathcal{O}_2[\![\mathrm{K} \leftarrow \mathrm{K}_{xy}, \mathrm{W}_{zw} \mid \{x < y,\ z < w\}]\!].\end{array}\right\} \text{COMBINATIONS}$$

$$\left.\begin{array}{l}\mathcal{O}_2[\![W \leftarrow A^+, A^- \mid \{x < y < z < w\}]\!].\\ \mathcal{O}_2[\![W \leftarrow C^-, C^+ \mid \{x < y < z < w\}]\!].\end{array}\right\} \text{CLOSURES}$$

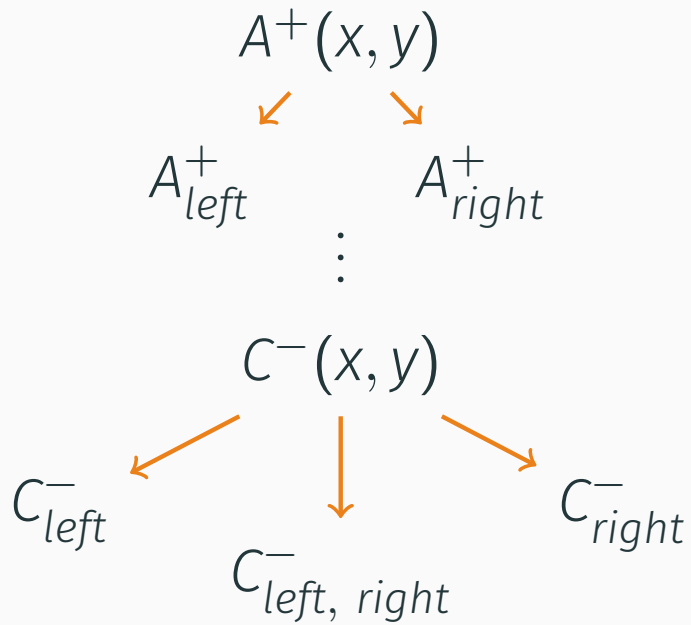
G_3: G_2 + Universal triple insertion

$$G_3 = G_2 + \forall\, K \in \mathcal{S} \setminus W :$$
$$\mathcal{O}_2[\![K \leftarrow K_{xy} \mid \{x < y,\ a < b < c\}]\!].$$

YOU SHALL NOT PARSE!

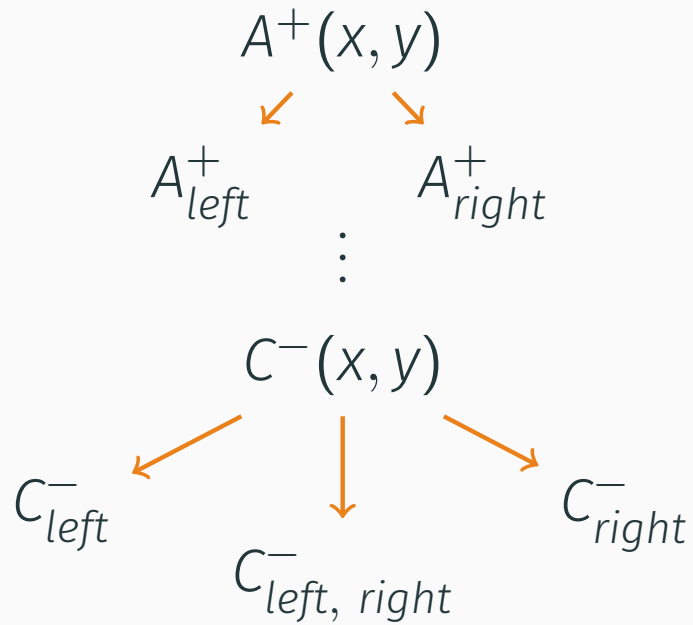
Refining states

Example



Refining states

Example

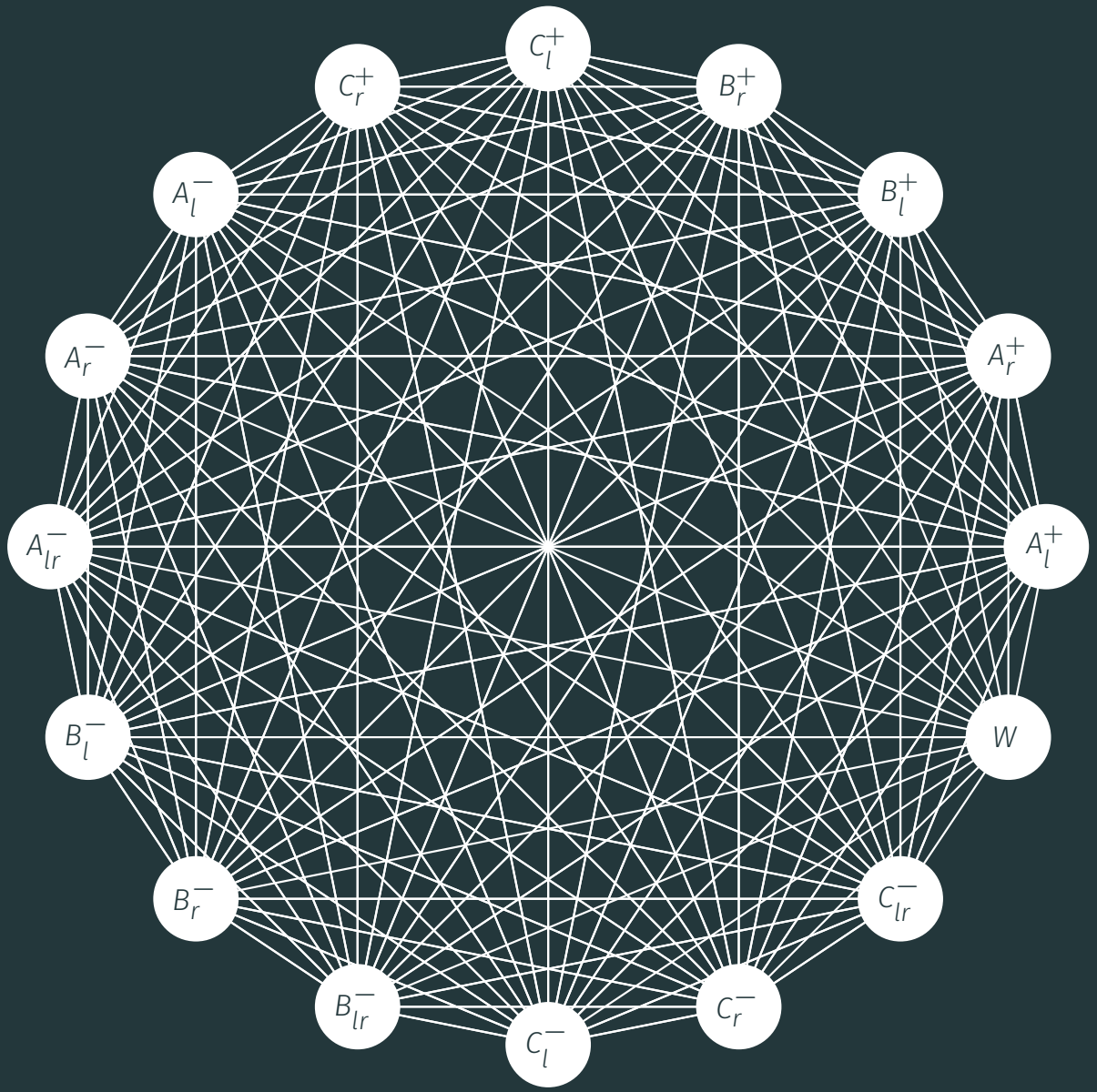


Why?

New orders in interactions

$$C^-(xz,\ wy) \leftarrow A^+_{left}(x,\ y),\ B^+(z,\ w).$$

Refining states: Interactions



G_4: Automatic Rule Inference

State descriptors $\mathcal{D}$

$$
\begin{aligned}
W &\mapsto (\epsilon, \epsilon) \\
A_l^+ &\mapsto (a, \epsilon) \\
A_r^+ &\mapsto (\epsilon, a) \\
B_l^+ &\mapsto (b, \epsilon) \\
B_r^+ &\mapsto (\epsilon, b) \\
C_l^+ &\mapsto (c, \epsilon) \\
C_r^+ &\mapsto (\epsilon, c) \\
A_l^- &\mapsto (bc, \epsilon)
\end{aligned}
\qquad
\begin{aligned}
A_r^- &\mapsto (\epsilon, bc) \\
A_{lr}^- &\mapsto (b, c) \\
B_l^- &\mapsto (ac, \epsilon) \\
B_r^- &\mapsto (\epsilon, ac) \\
B_{lr}^- &\mapsto (a, c) \\
C_l^- &\mapsto (ab, \epsilon) \\
C_r^- &\mapsto (\epsilon, ab) \\
C_{lr}^- &\mapsto (a, b)
\end{aligned}
$$

Automatic Rule Inference: Example

$$\textsc{Case of } \mathrm{A}^{-}_{lr}(x_b,\ y_c) + \mathrm{B}^{-}_{lr}(z_a,\ w_c)$$

PERMUTATION

$$\vdots$$

$$(zxw,\ y)$$

$$\vdots$$

$$(xzw,\ y)$$

$$\vdots$$

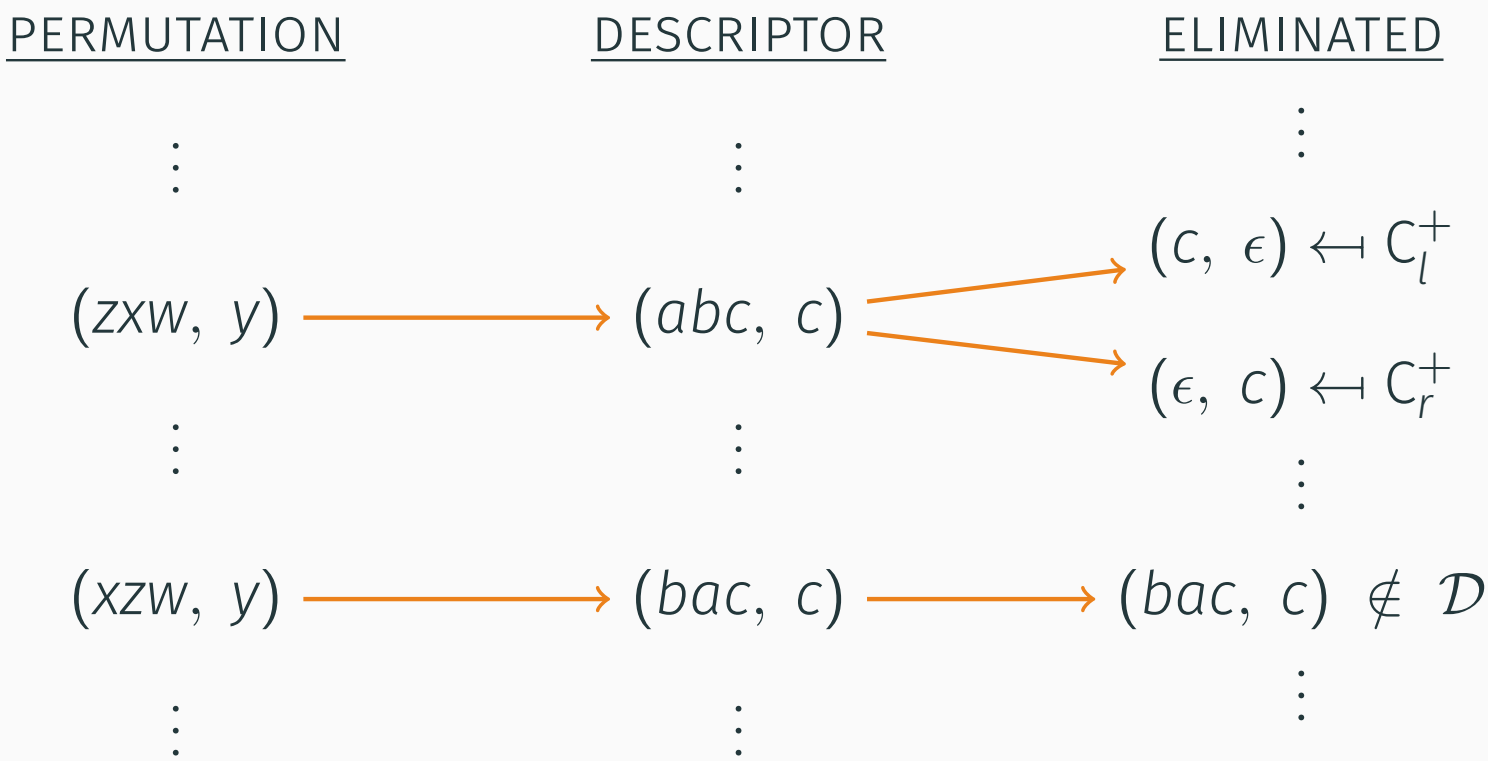
Automatic Rule Inference: Example

Case of $\mathrm{A}^{-}_{lr}(x_b,\ y_c) + \mathrm{B}^{-}_{lr}(z_a,\ w_c)$

PERMUTATION	DESCRIPTOR
$\vdots$	$\vdots$
$(zxw,\ y)$	$(abc,\ c)$
$\vdots$	$\vdots$
$(xzw,\ y)$	$(bac,\ c)$
$\vdots$	$\vdots$

Automatic Rule Inference: Example

Case of $\mathrm{A}^{-}_{lr}(x_b,\ y_c) + \mathrm{B}^{-}_{lr}(z_a,\ w_c)$

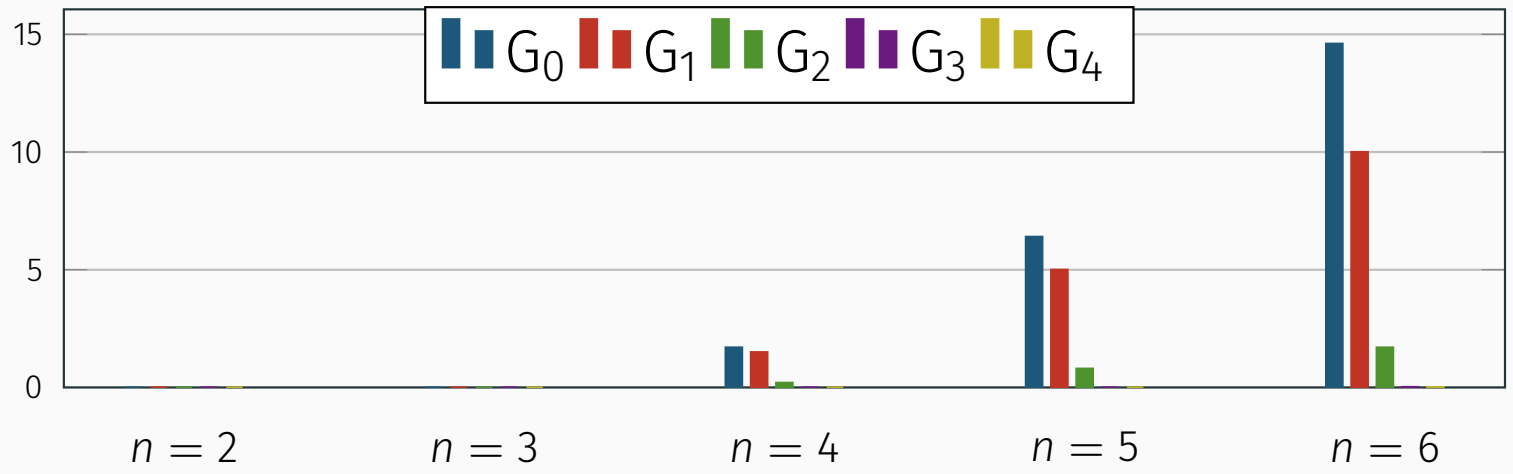


DSL in Python ($ pip install dyck)

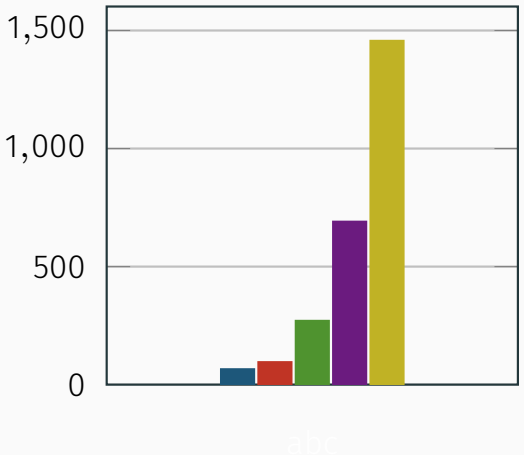
```
from dyck import *
G3 = Grammar(initial='W',
  # Base Cases
  O('W', {(a, b, c)}),
  O('A-', {(b, c)}), O('B-', {(a, c)}), O('C-', {(a, b)}),
  O('A+', {(a,)}),   O('B+', {(b,)}),   O('C+', {(c,)}),
  # Combinations
  O('C- <- A+, B+', {(x, y, z, w)}),
  O('B- <- A+, C+', {(x, y, z, w)}),
  O('A- <- B+, C+', {(x, y, z, w)}),
  O('C+ <- B-, A-', {(x, y, z, w)}),
  O('B+ <- C-, A-', {(x, y, z, w)}),
  O('A+ <- C-, B-', {(x, y, z, w)}),
  forall(all_states, lambda K: O('K <- K, W', {(x, y), (z, w)})),
  # Closures
  O('W <- A+, A-', {(x, y, z, w)}),
  O('W <- C-, C+', {(x, y, z, w)}),
  # Universal Triple Insertion
  forall(all_states, lambda K: O('K <- K', {(x, y), (a, b, c)})))
```

Results

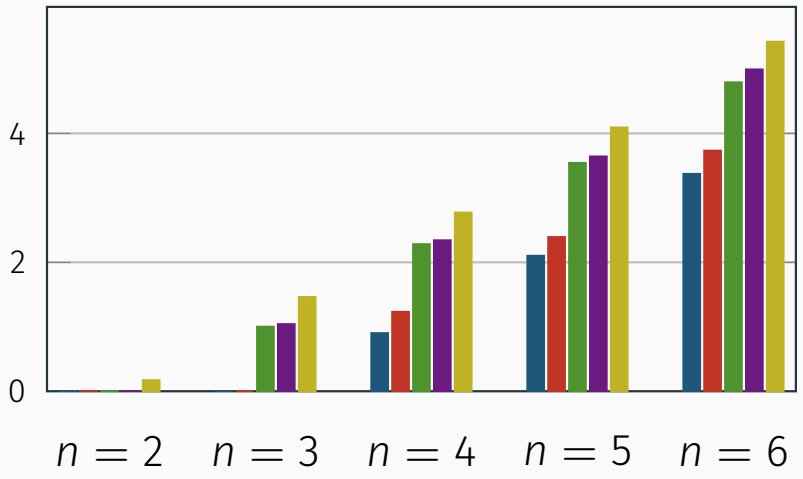
Counter-examples (%)



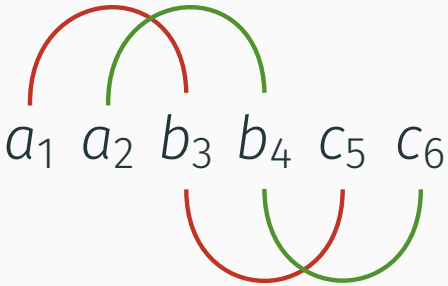
Rule size



Computation Time (in log(sec))

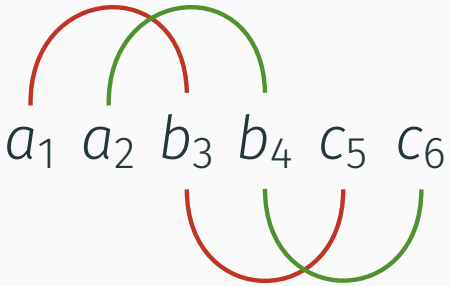


Correspondences: Promotion on Young Tableaux



1	2
3	4
5	6

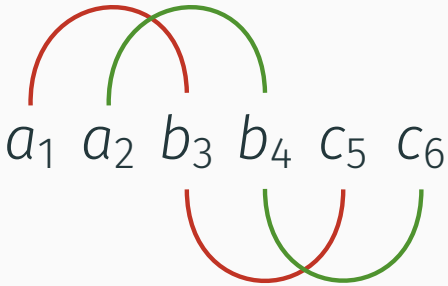
Correspondences: Promotion on Young Tableaux



1	2
3	4
5	6

•	1
2	3
4	5

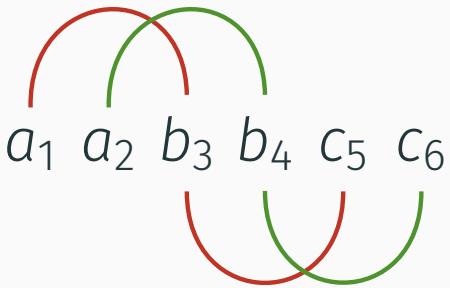
Correspondences: Promotion on Young Tableaux



1	2
3	4
5	6

1	•
2	3
4	5

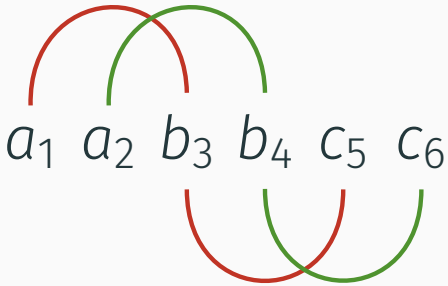
Correspondences: Promotion on Young Tableaux



1	2
3	4
5	6

1	3
2	▪
4	5

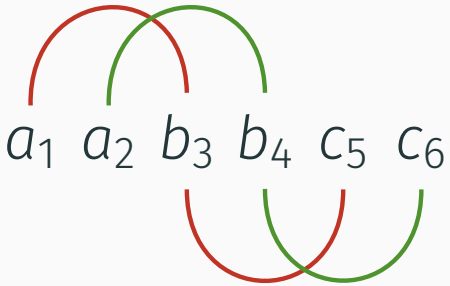
Correspondences: Promotion on Young Tableaux



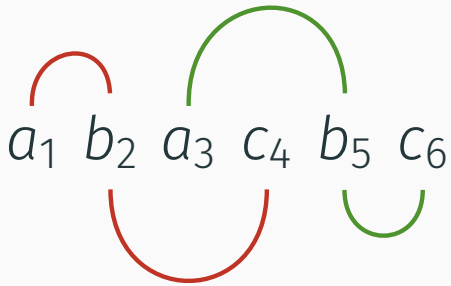
1	2
3	4
5	6

1	3
2	5
4	.

Correspondences: Promotion on Young Tableaux



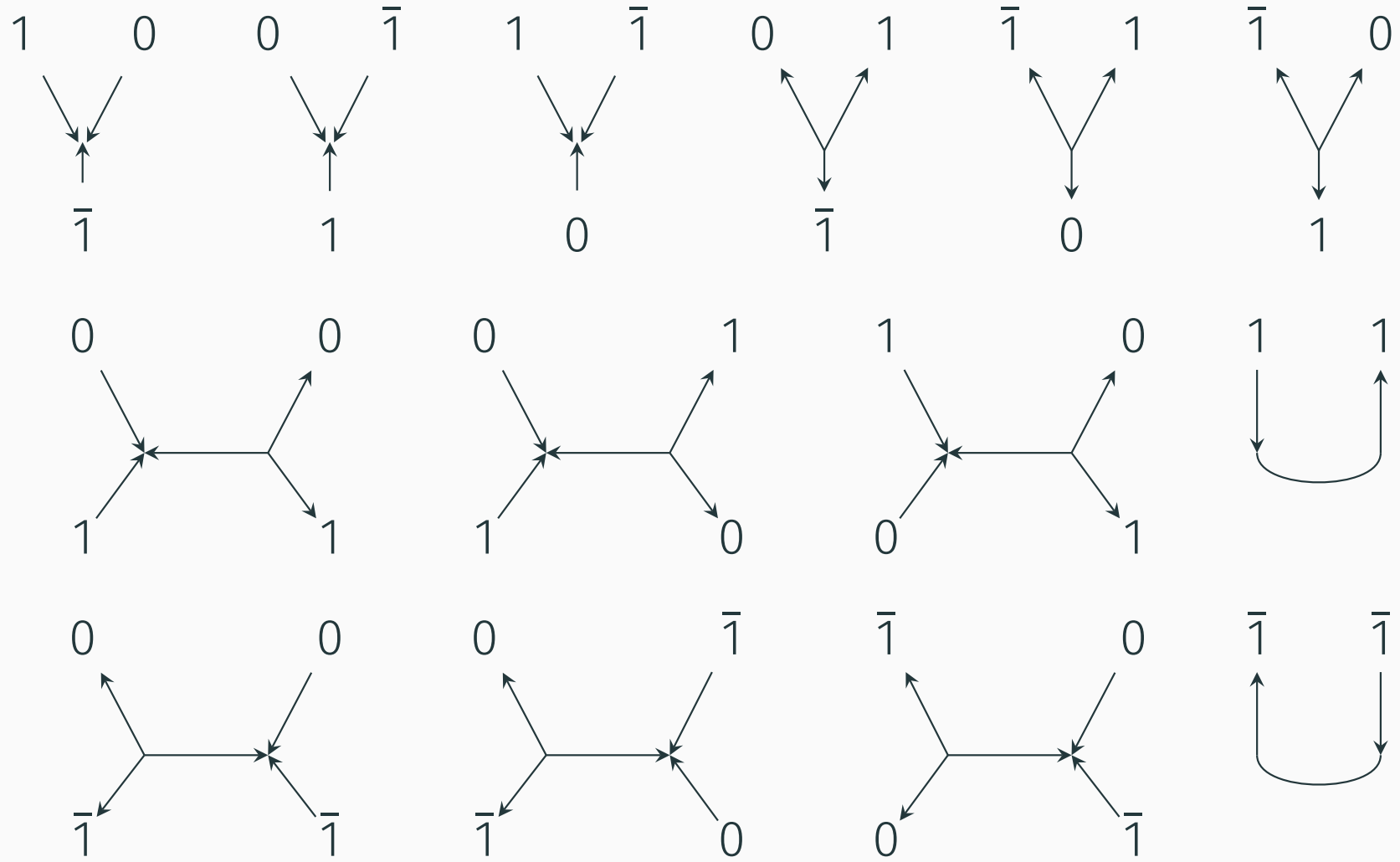
1	2
3	4
5	6



1	3
2	5
4	6

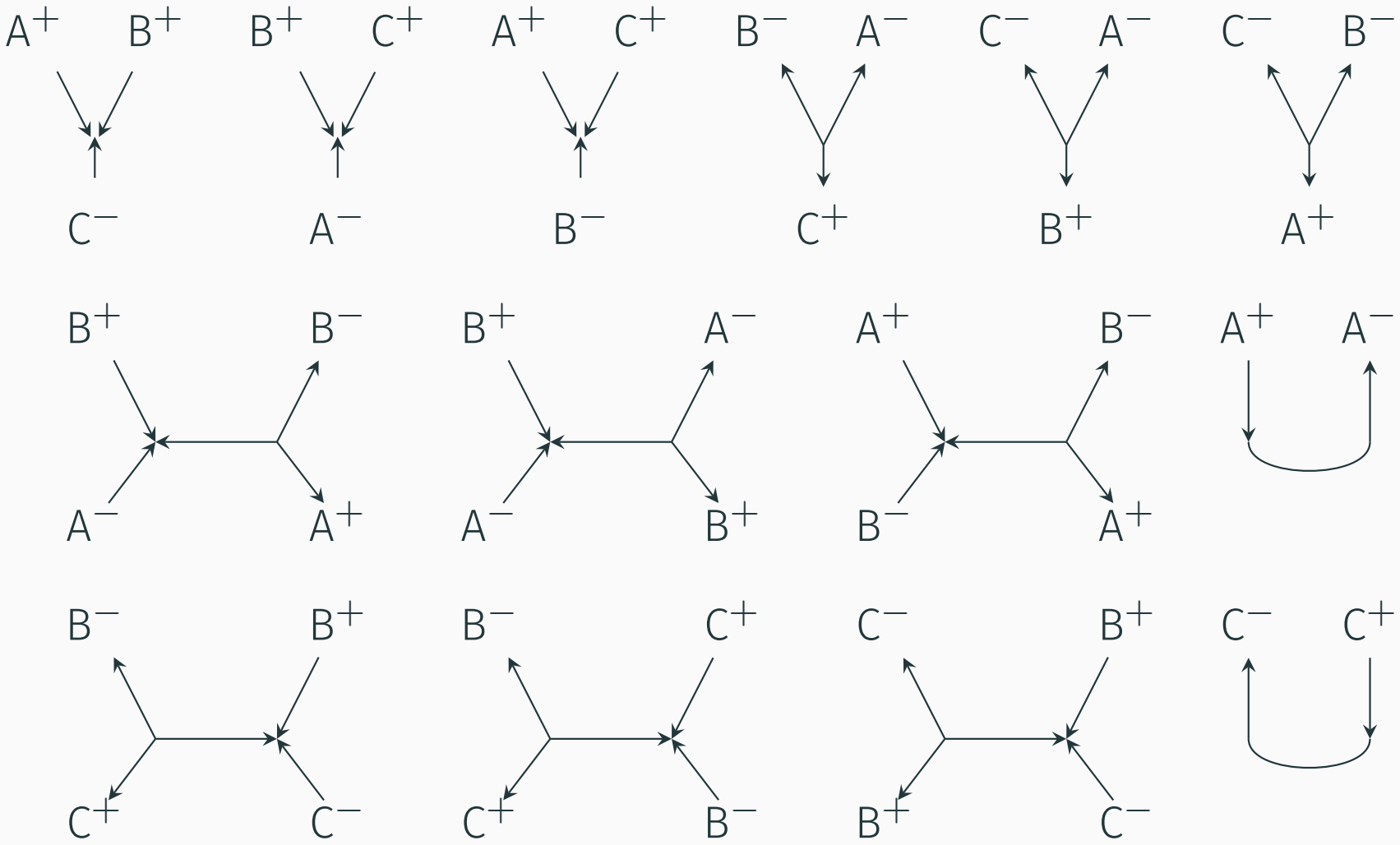
Correspondences: Spider Webs

Growth rules



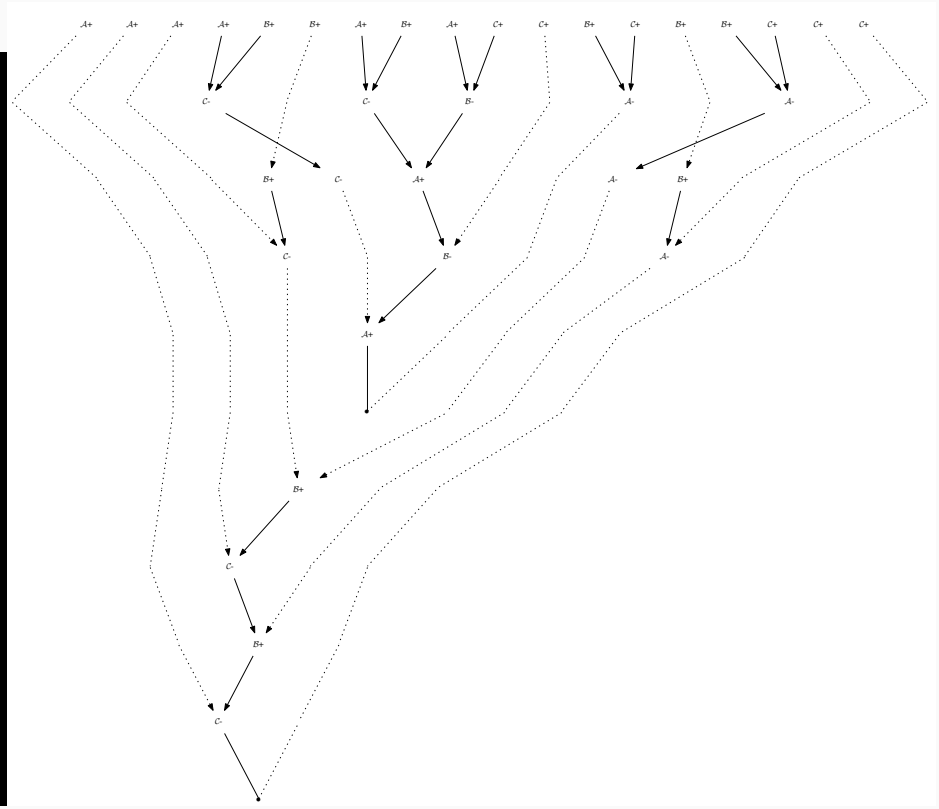
Correspondences: Spider Webs

Growth rules



Python Visualization Package ($ pip install dyckviz)

```
λ dyckviz --word 'aaaabbabaccbcbbccc'
1.      aabbaccbacbabaccbc
2.      aabbabaccbcaacbbcc
3.      abacbacaabbaccbbcc
4.      abaabbaccbaccabbcc
5.      aaabbabaccbcbaccbc
6.      abaacaabbaccbbccbc
7.      aabaabbaccbbccacbc
8.      abaaacbabaccbcbbcc
9.      abaabaccbcaabbbccc
10.     aabacbbcaabbaccbc
11.     aabbacaabbaccbbccc
12.     aaabbaccbacbabbccc
13.     abaacbbacbabaccbcc
14.     aabbaacbabaccbccbc
15.     aaabaabbaccbbccbcc
16.     aaabbbacbabaccbccc
17.     aaaabbabaccbcbbccc
18.     aaabacbbcaabbbcccc
```



Conclusion

- Conjecture still open :(
- Lots of fun along the way
- We are confident we have a complete 3-MCFG, though
  - Currently mechanizing the proof using Coq

Questions?