Representing Multidimensional Trees

David Brown, Colin Kern,
Alex Lemann, Greg Sandstrom
Earlham College

Introduction

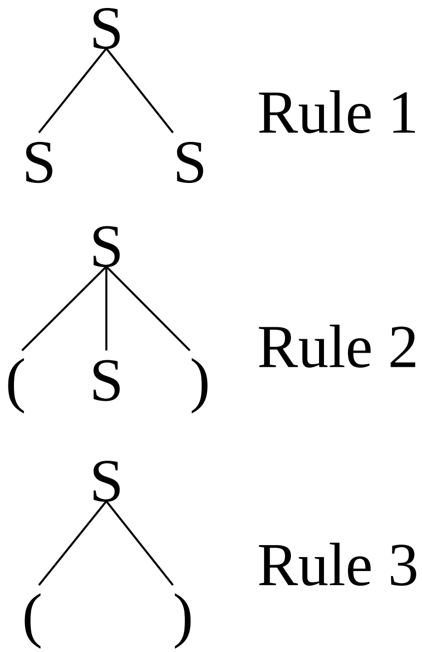
- Presentation Layout
  - Dave: An Introduction to Multidimensional Trees and Grammars
  - Alex: Representing Multidimensional Trees
  - Colin: A Chomsky Normal Form (CNF) Transformation for Multidimensional Grammars.

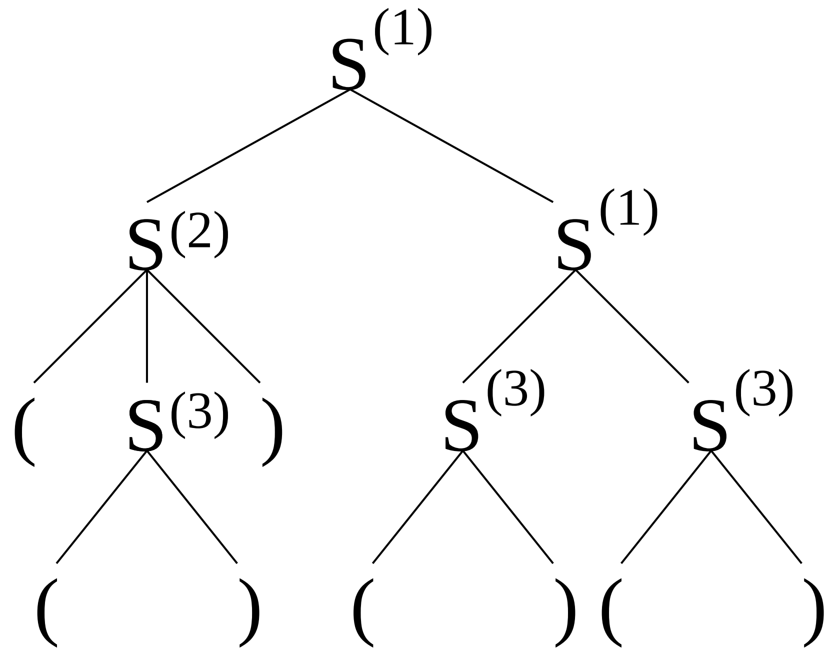
An Example Context-Free Grammar (CFG)

$G:$	$\langle \Sigma, V, S, P \rangle$
$\Sigma:$	$\{(,)\}$
$V:$	$\{S\}$
$S:$	S
$P:$	$\{S \rightarrow SS, S \rightarrow (S), S \rightarrow ()\}$

S	Initial
S S	$S \rightarrow SS$
(S) S	$S \rightarrow (S)$
(S) S S	$S \rightarrow SS$
(( )) S S	$S \rightarrow ()$
(( )) ( ) S	$S \rightarrow ()$
(( )) ( ) ( )	$S \rightarrow ()$

An Example CFG as a Set of Local Trees





An Example Tree-Adjoining Grammar (TAG)

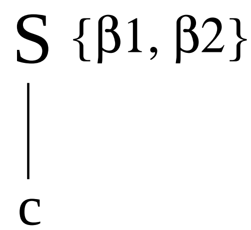
Σ: { a, b, c }

V: { S }

S: S

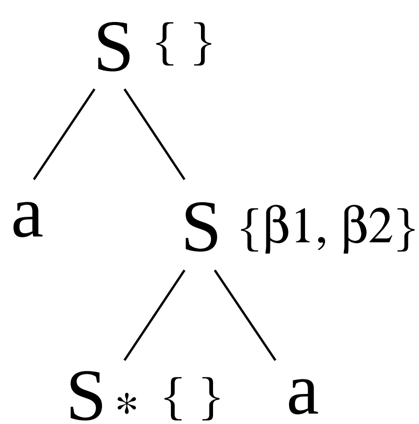
I:

$\alpha 1$:

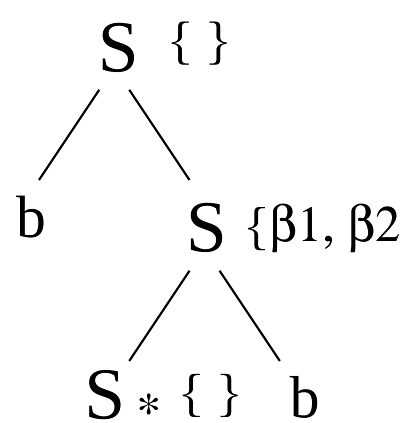


A:

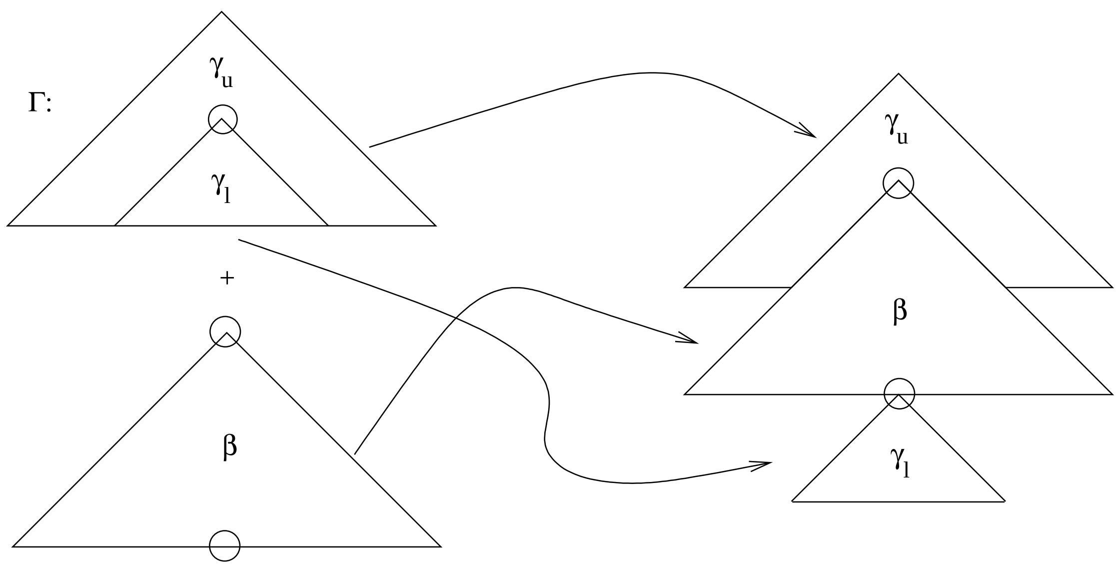
$\beta 1$:

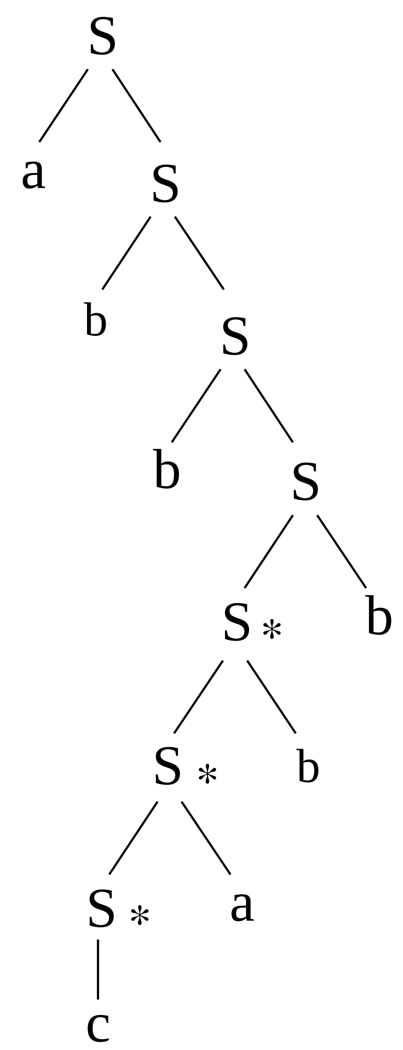


$\beta 2$:



An Example TAG Derivation



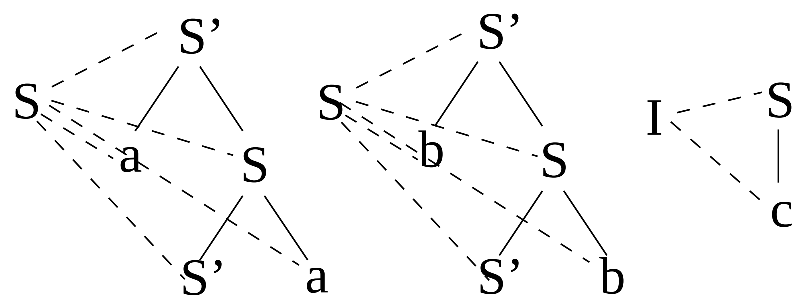


An Example Multidimensional Grammar

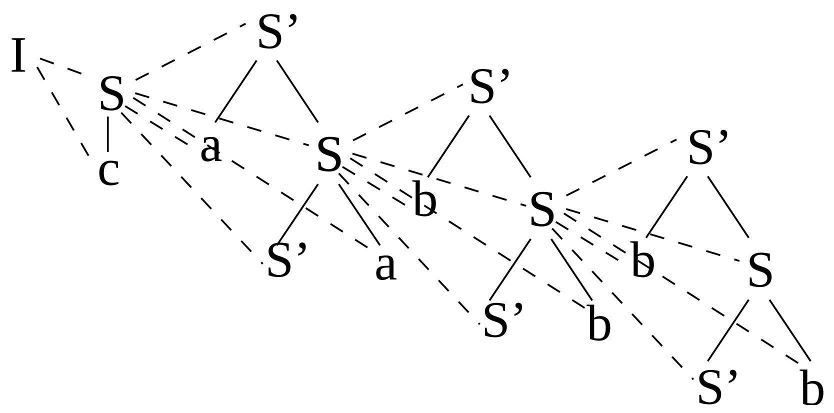
Σ: { a, b, c }

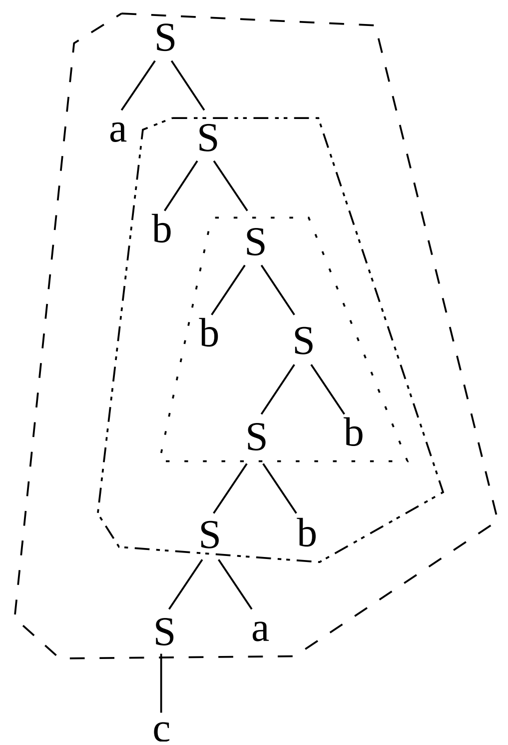
V: { I, S, S' }

S: I

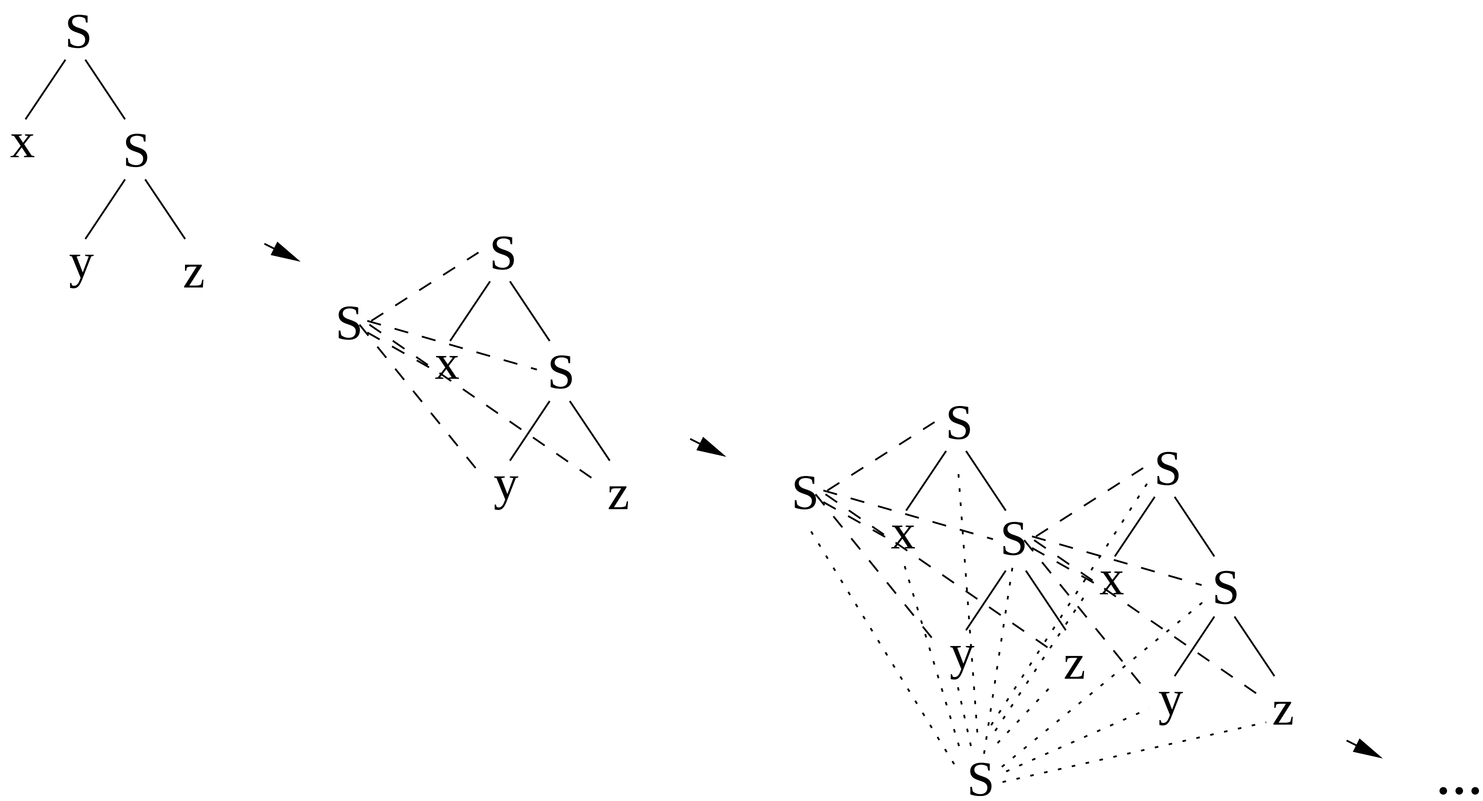
P: 

A Multidimensional Derivation and 2D Yield





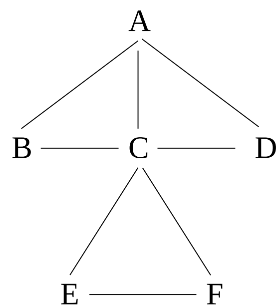
An Infinite Hierarchy Equivalent to Weir's Control Language Hierarchy

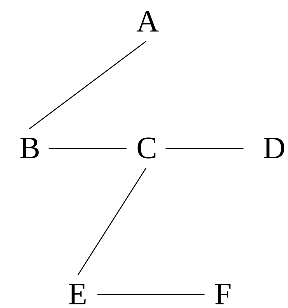


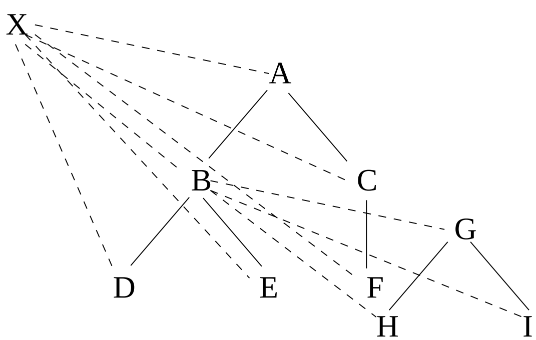
Representing Multidimensional Trees

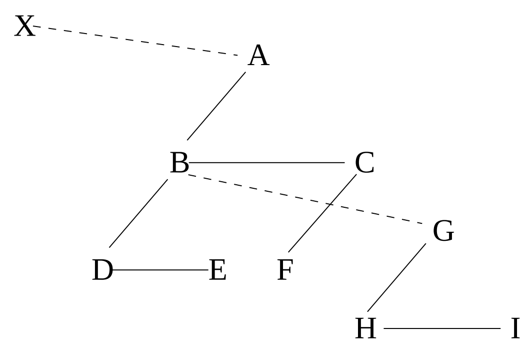
David Brown, Colin Kern,
Alex Lemann, and Greg Sandstrom
Earlham College

Left-child Right-sibling Form



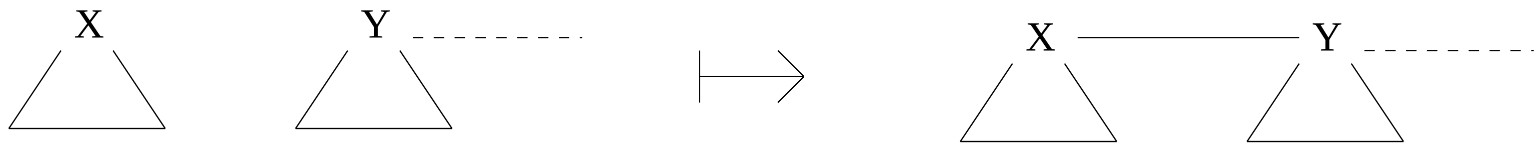




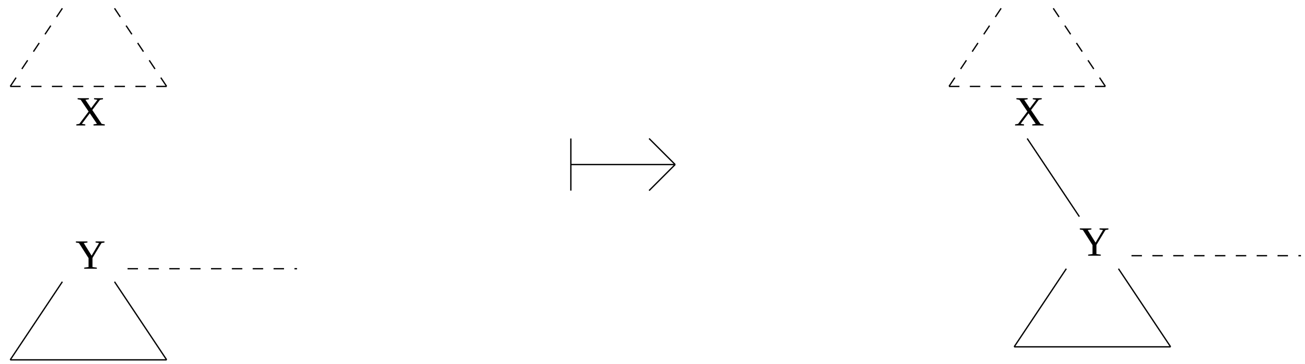


ExLeft and ExUp Definitions

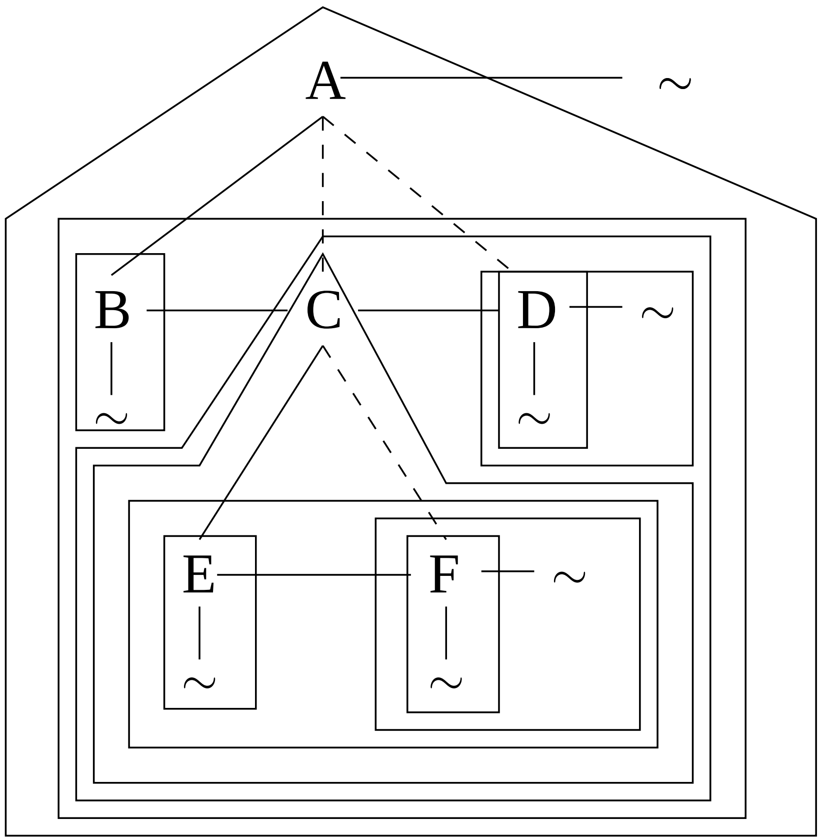
ExLeft



ExUp



Building a 2d tree using ExUp and ExLeft



$\text{ExUp}(F, \sim) = t1$

$\text{ExLeft}(t1, \sim) = f1$

$\text{ExUp}(E, \sim) = t2$

$\text{ExLeft}(t1, f1) = f2$

$\text{ExUp}(D, \sim) = t3$

$\text{ExLeft}(t3, \sim) = f3$

$\text{ExUp}(C, f2) = t4$

$\text{ExLeft}(t4, f3) = f4$

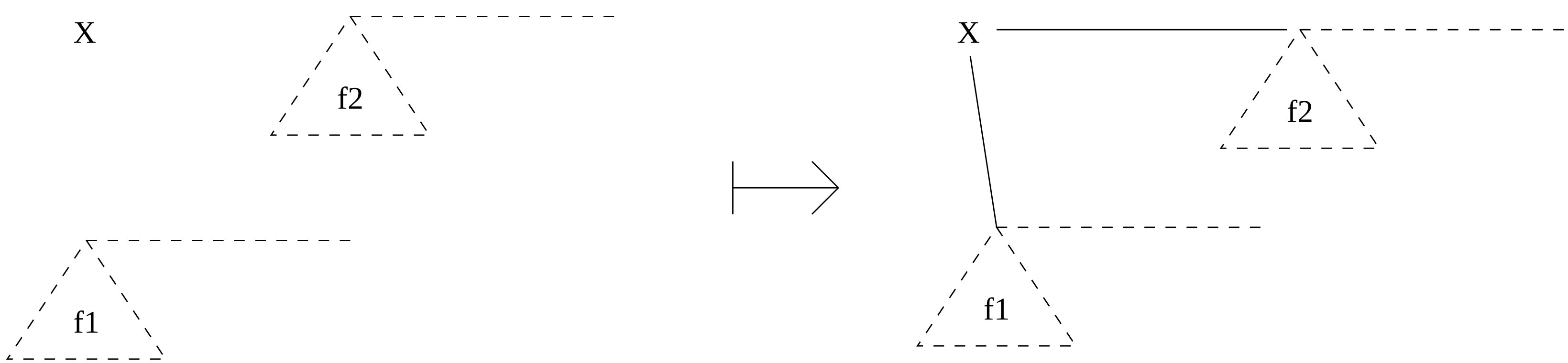
$\text{ExUp}(B, \sim) = t5$

$\text{ExLeft}(t5, t4) = f5$

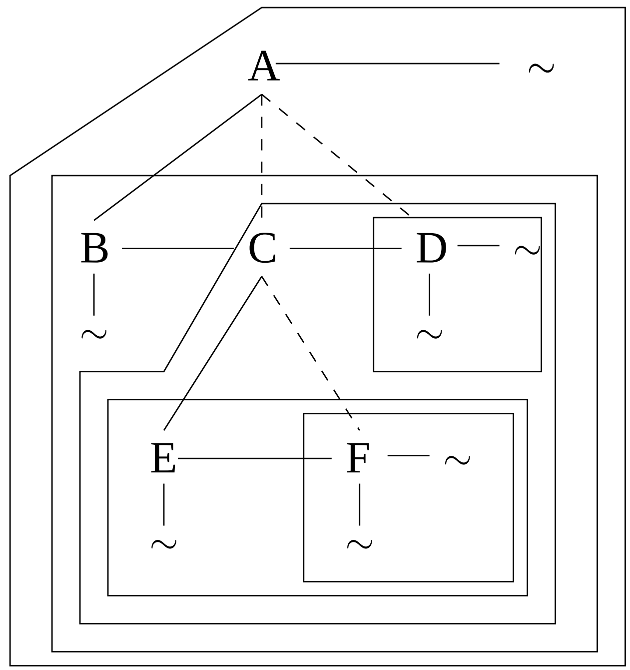
$\text{ExUp}(A, f5) = t6$

$\text{ExLeft}(t6, \sim) = \text{final tree}$

Unified Constructor for Tree-ordered Forest Definition

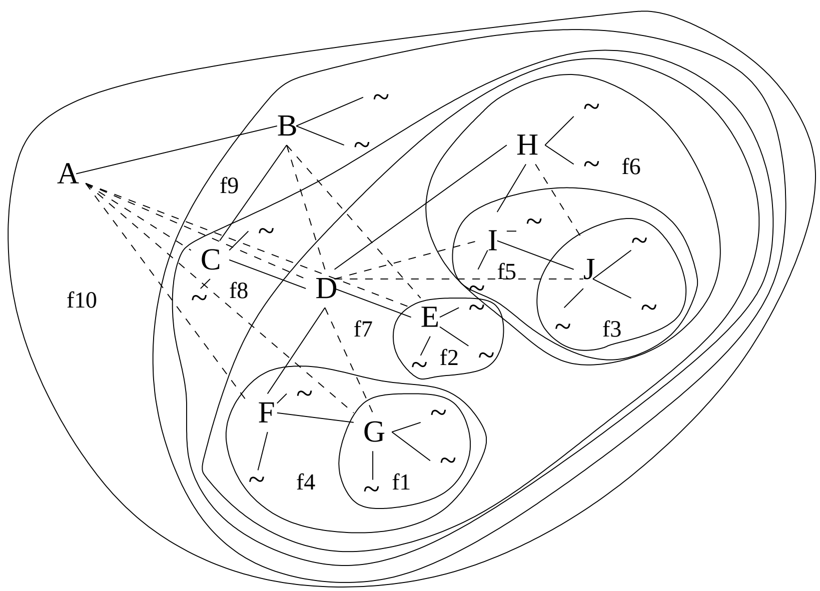


Building a 2d tree using the unified constructor



$$T(F, \sim, \sim) = f1$$
$$T(E, f1, \sim) = f2$$
$$T(D, \sim, \sim) = f3$$
$$T(C, f3, f2) = f4$$
$$T(B, f4, \sim) = f5$$
$$T(A, \sim, f5)$$

Building a 3d tree using the unified constructor



$T(J, \sim, \sim, \sim) = f3$

$T(I, f3, \sim, \sim) = f5$

$T(H, \sim, f5, \sim) = f6$

Then we create $f4$:

$T(G, \sim, \sim, \sim) = f1$

$T(F, f1, \sim, \sim) = f4$

And the last child of D is $f2$:

$T(E, \sim, \sim, \sim) = f2$

Next we combine them in D:

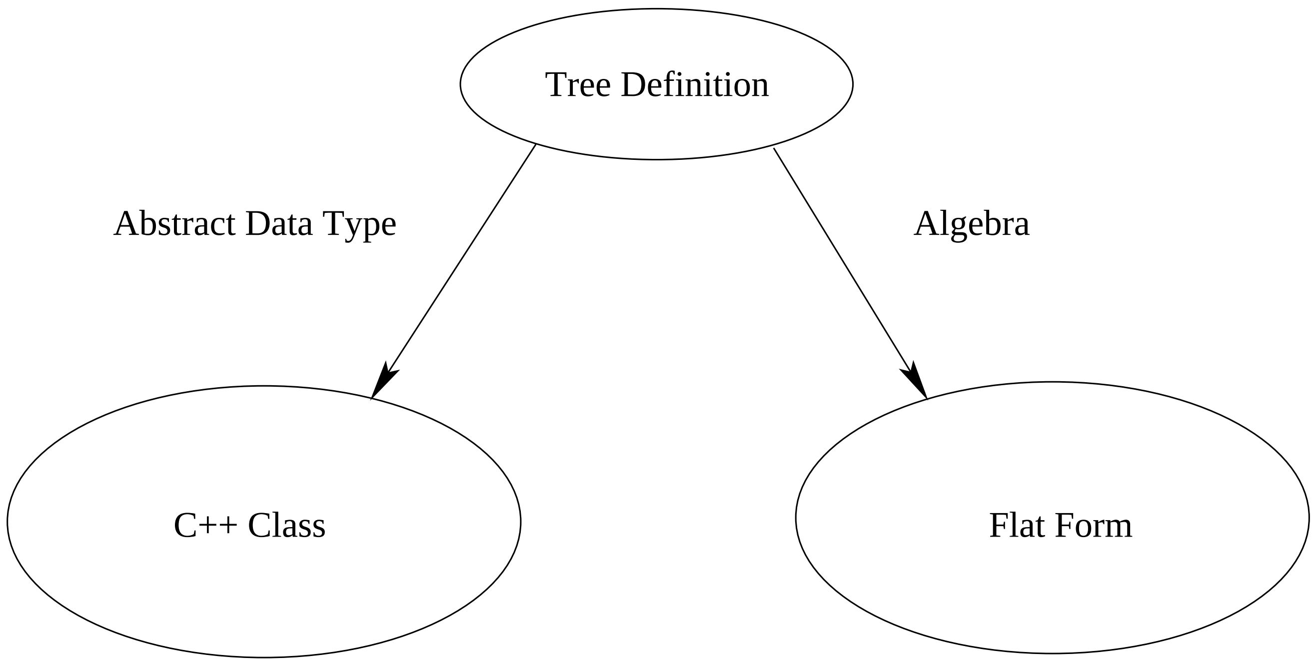
$T(D, f2, f4, f6) = f7$

And then we can continue to build the rest of the tree:

$T(C, f7, \sim, \sim) = f8$

$T(B, \sim, f8, \sim) = f9$

$T(A, \sim, \sim, f9) = f10$


Tree Definition
Abstract Data Type
Algebra
C++ Class
Flat Form

Abstract data type realized as a C++ class

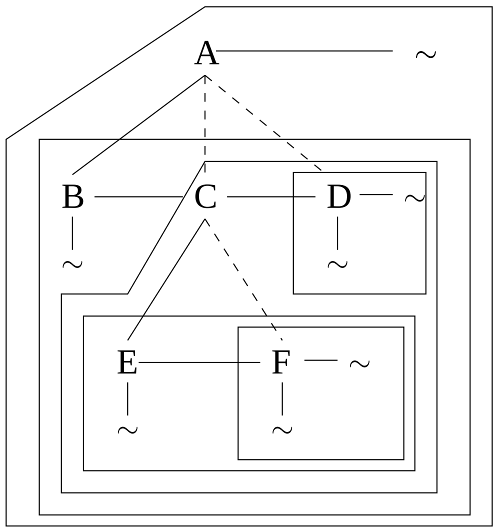
```
template<class label_type>
class forest
{
 public:
  forest(label_type label, vector<forest*> links)

  void set_label(label_type new_label);
  void set_link(std::size_t link_number, forest* new_link);

  label_type get_label( ) const;
  tree* get_successor(std::size_t link_number) const;

 private:
  label_type label;
  vector<forest*> link;
}
```

Flat form representation



$$T(F, \sim, \sim) = f1 \qquad F(\sim, \sim)$$

$$T(E, f1, \sim) = f2 \qquad E(F(\sim, \sim), \sim)$$

$$T(D, \sim, \sim) = f3 \qquad D(\sim, \sim)$$

$$T(C, f3, f2) = f4 \qquad C(D(\sim, \sim), E(F(\sim, \sim), \sim))$$

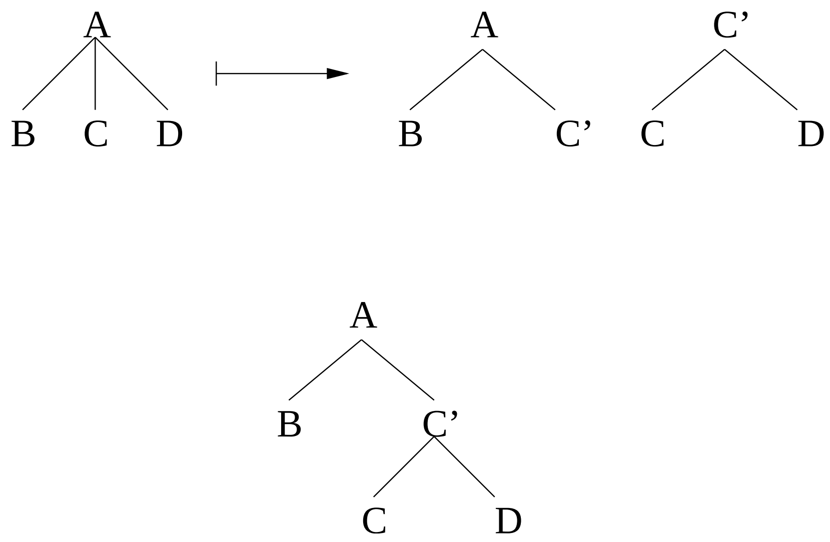
$$T(B, f4, \sim) = f5 \qquad B(C(D(\sim, \sim), E(F(\sim, \sim), \sim)), \sim)$$

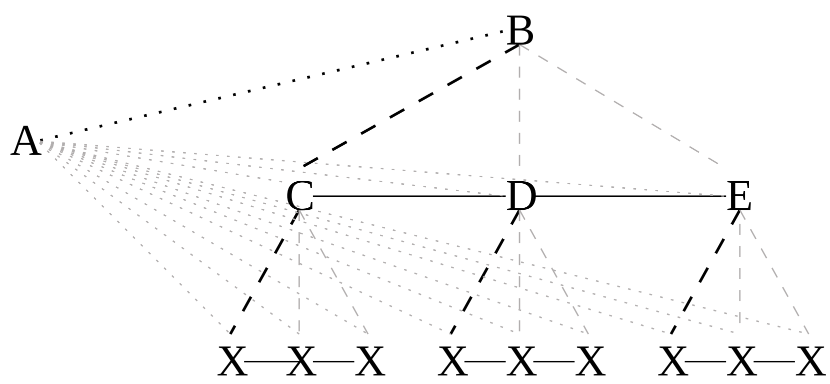
$$T(A, \sim, f5) \qquad A(\sim, B(C(D(\sim, \sim), E(F(\sim, \sim), \sim)), \sim))$$

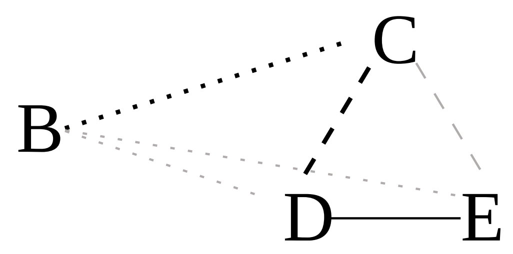
A CNF Transformation for Multidimensional Grammars

David Brown, Colin Kern,
Alex Lemann, Greg Sandstrom
Earlham College

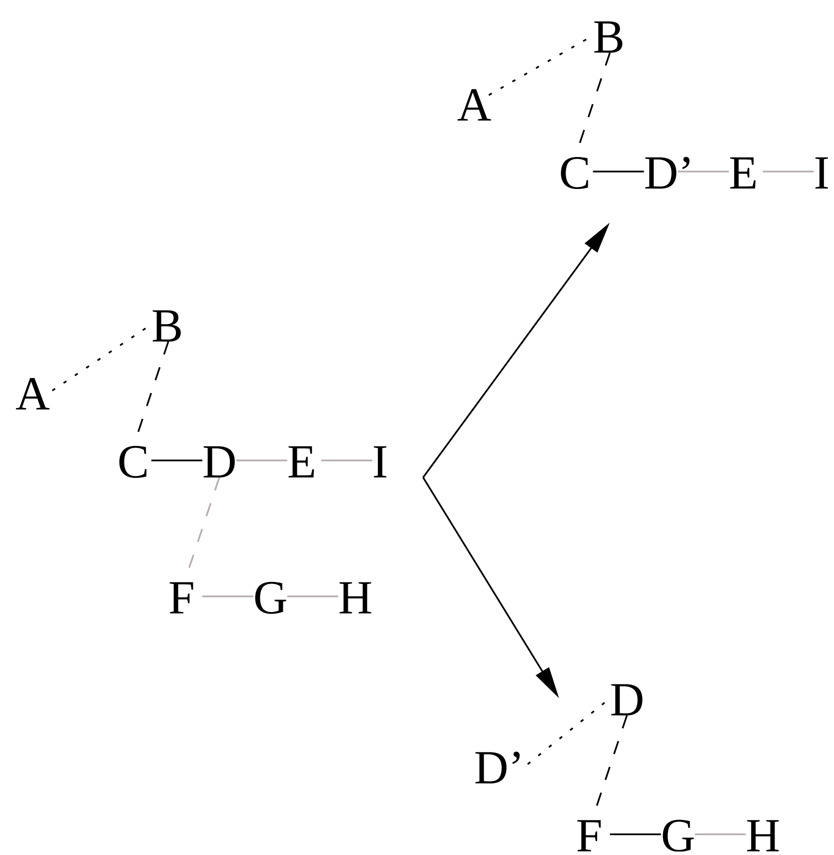
Chomsky Normal Form



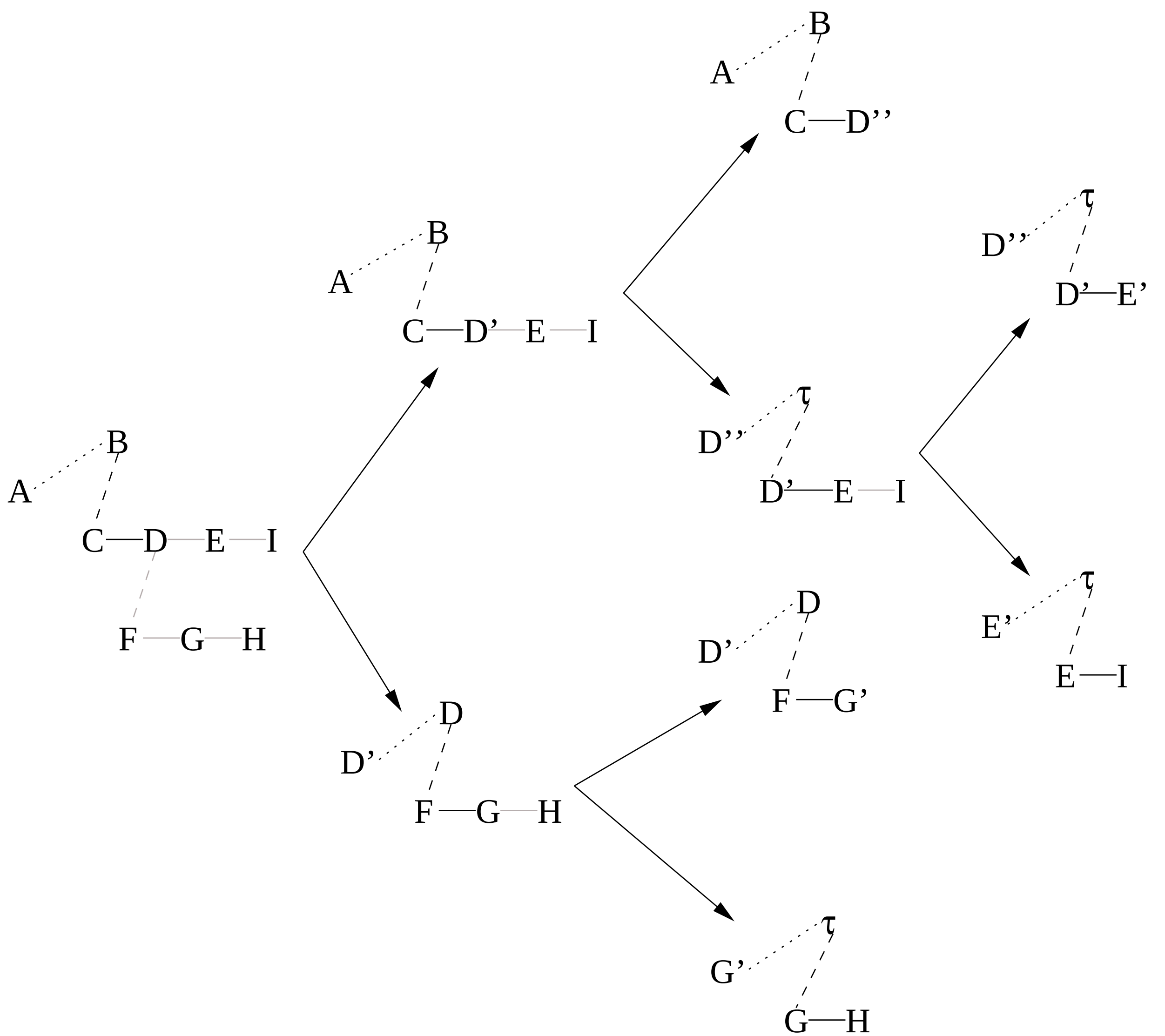

B
A
C
D
E
X X X X X X X X X


C
B
D
E

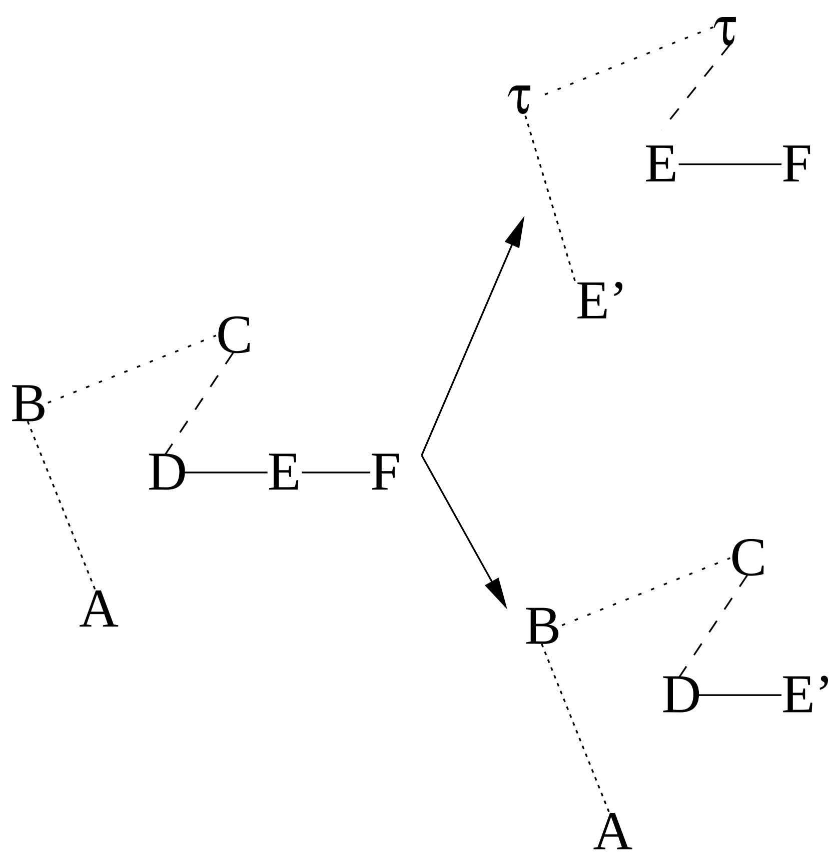
An example factoring of a 3d tree



The entire factoring of the tree

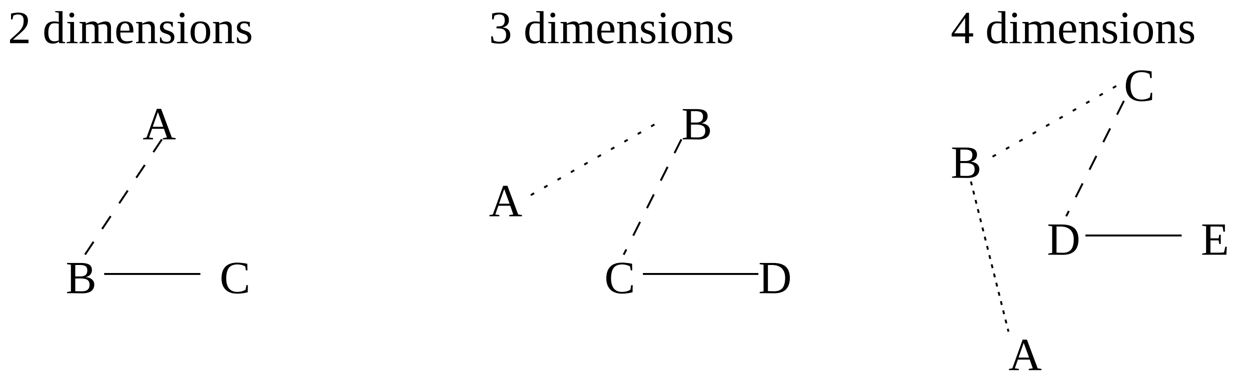


A 4-dimensional local tree of a grammar being factored into 2-branching local trees

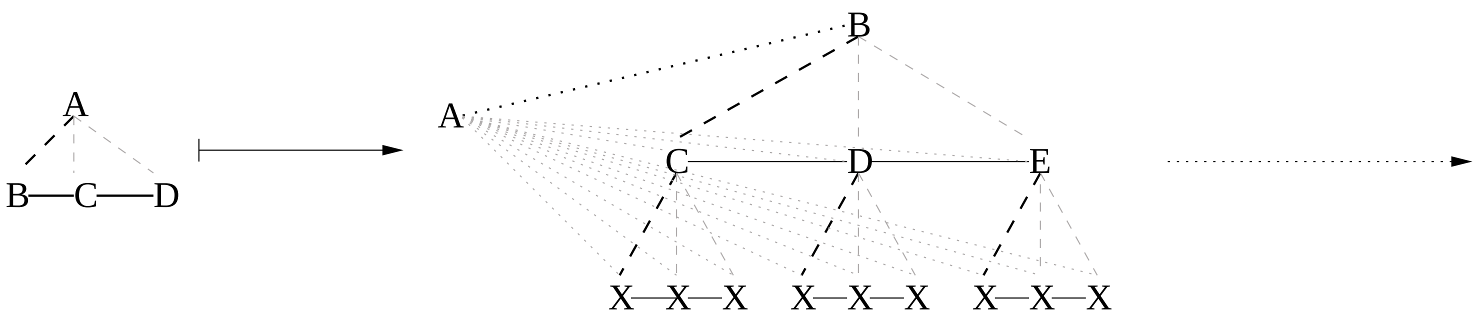


The transformation algorithm:

- Traverse the tree in a depth-first method.
- For every node, check each link for a successor that breaks the 2-branching definition.
- For every such successor, split the tree into two trees:
  - A tree with the subtree rooted at the successor removed, with the current node renamed to a unique label.
  - The subtree rooted at the successor with the current node at the root.
- Use τ nodes to fill out the nodes of a new tree if the dimension is less than the dimension of the original tree.
- Repeat on each factored tree until no more factors are created.



# of nodes = # of dimensions + 1


A
B
C
D
A
B
C
D
E
X
X
X
X
X
X
X
X
X

$$
\begin{aligned}
N_3(0) &= 1 \\
N_3(d) &= N_3(d-1)^2 - N_3(d-1) + 2
\end{aligned}
$$

We can solve this recursion:

$$N_3(d) = \Omega(k^{(2^{(d-1)})})$$

The growth is hyper-exponential in the dimension

The 2-branching trees show linear growth in the dimension.
The 3-branching trees show hyper-exponential growth in the dimension.

Theorem 1 *For a full d-dimensional, n-branching local tree, the number of local trees in the factored form required is equal to the total number of 1-dimensional links.*

While the number of local trees in the grammar grows by a factor that is hyper-exponential in the dimension, the growth is optimal in the sense that it differs only by a constant factor from the growth in the number of nodes in arbitrary branching local trees as a function of the dimension.

Definition 1 *Tree-ordered Forests*

- $\sim$ *is an (empty)* (i,d)*-forest for all* $0 \leq i \leq d$

- *If* t_1, t_2, $\ldots$, t_d *are, respectively,* $(0,d)$-, $(1,d)$-, $\ldots$, $(d-1,d)$*-forests and* $X \in \Sigma$ *then* $T(X, t_1, t_2, \ldots, t_d)$ *is a* (j,d)*-forest for all* $0 \leq j \leq i$, *where* i *is the smallest dimension such that* t_i *is not empty, or d if all* t_k *are empty. Here each* t_k *is the successor of the new node labeled* X *in the* k*th dimension.*

- *Nothing else is a tree-ordered forest.*