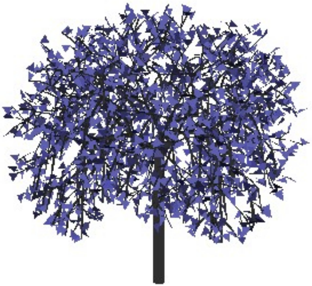
# The Christmas Tree Dilemma

Sarah Berkemer

Bioinformatics Leipzig

Winterseminar 2016, Bled

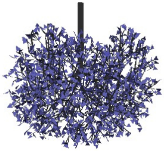
# Introduction



**This is a Tree..**

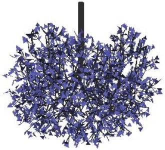
# Definitions

**This is a Tree..**

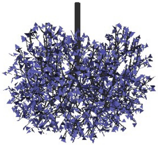


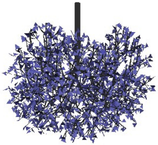
# Definitions

**This is a Tree..**



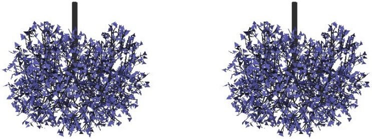
**..and this is a Forest.**





# Definitions

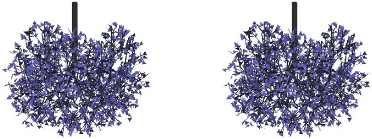
**This is a Forest..**



**..and this is a Forest, too.**

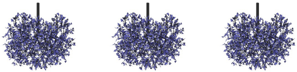
# Definitions

**This is a Forest..**

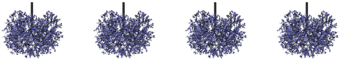


**..and this is a Forest, too.**

**..and this..**



**..and this, too.**



# Introduction II

**This are christmas trees...**

# Definitions

**This is a Christmas tree..**

# Definitions

**This is a Christmas tree..**

**..and this, too.**

# Definitions

**This is a Christmas tree..**

**..and this, too.**

**This is a Christmas forest..**

# Definitions

**This is a Christmas tree..**

**..and this, too.**

**This is a Christmas forest..**

**..and this, too.**

# Introduction III

**These are Christmas ornaments..**

# Introduction III

**These are Christmas ornaments..**

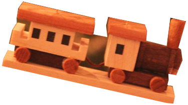
**..and this, too.**

# Introduction III

**These are Christmas ornaments..**


**..and this, too.**


**And this..**

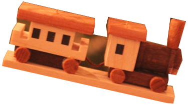



# Introduction III

**These are Christmas ornaments..**


**..and this, too.**


**And this..**



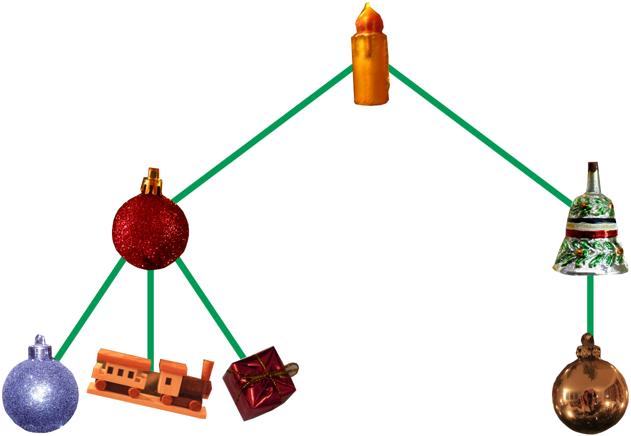

**..and this, too.**

# Basics

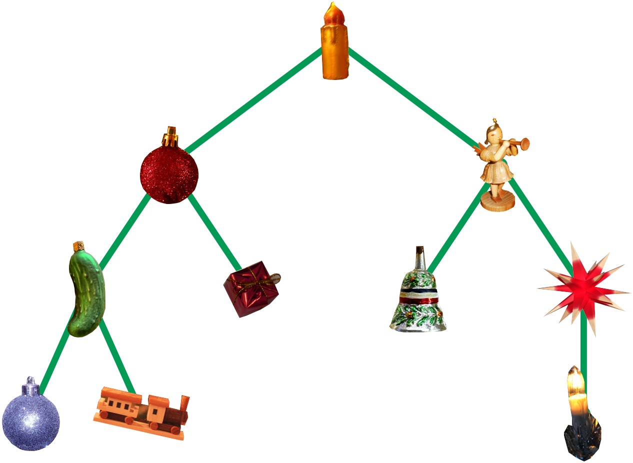
## Assume that

- christmas tree = tree
- christmas forest = forest
- christmas ornament = label

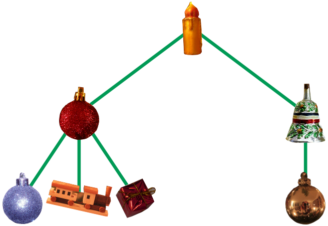
# Our Christmas Tree



# The Neighbours' Tree



# Higher, better, nicer, stronger..



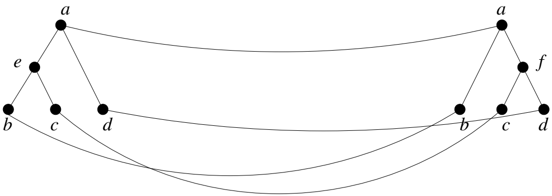
?
$\Rightarrow$

# Tree Editing

## Definition

A mapping between two ordered trees (forests) $T_1$ and $T_2$ is a binary relation on pairs of vertices $(x, y)$ and $(x', y')$ with $x, x' \in V(T_1), y, y' \in V(T_2)$ such that the following conditions hold:

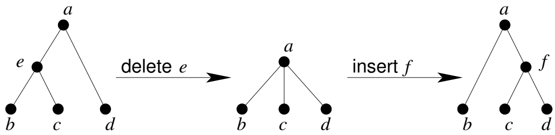
- one-to-one condition: $x = x' \Leftrightarrow y = y'$
- ancestor condition: $x$ ancestor of $x' \Leftrightarrow y$ ancestor of $y'$
- sibling condition: $x$ left sibling of $x' \Leftrightarrow y$ left sibling of $y'$



# Tree Editing

## Operations

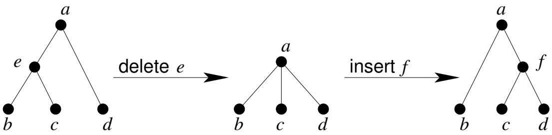
- relabeling
- deletion
- insertion



# Tree Editing

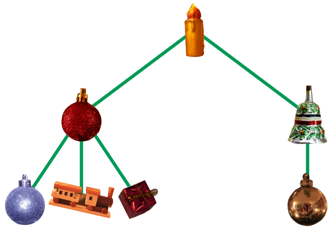
## Operations

- relabeling
- deletion
- insertion




Aim: Minimize number of operations!
$\Rightarrow$ can be done with a DP algorithm.

# Tree Editing



$\Rightarrow$

Insert

Insert

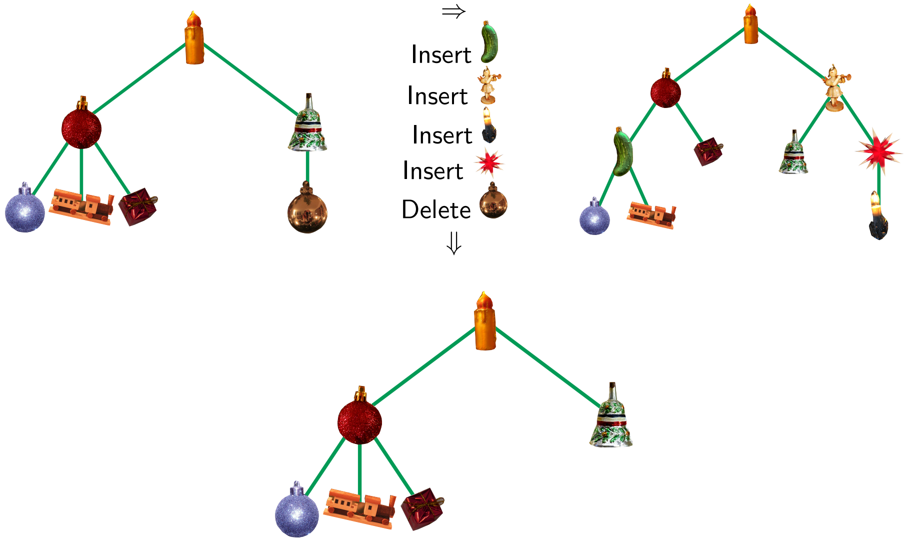
Insert

Insert

Delete

$\Downarrow$

# Tree Editing



# Tree Alignment

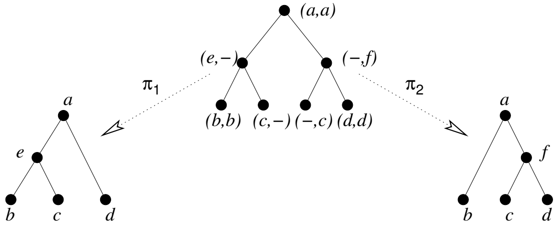
## Definition

A mapping of labels on forests $F_1$ and $F_2$ based on $(\mathcal{A} \cup \{-\}) \times (\mathcal{A} \cup \{-\})$ and restrictions $\pi_1(G)$ and $\pi_2(G)$ by considering either the first or the second coordinate, with $G$ being the resulting alignment forest.

# Tree Alignment

## Definition

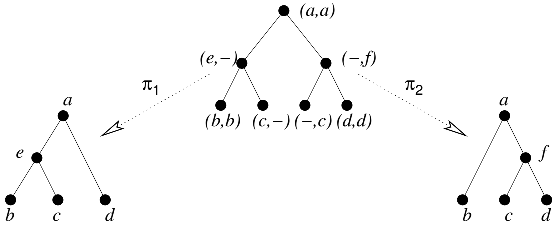
A mapping of labels on forests $F_1$ and $F_2$ based on $(\mathcal{A} \cup \{-\}) \times (\mathcal{A} \cup \{-\})$ and restrictions $\pi_1(G)$ and $\pi_2(G)$ by considering either the first or the second coordinate, with $G$ being the resulting alignment forest.



# Tree Alignment

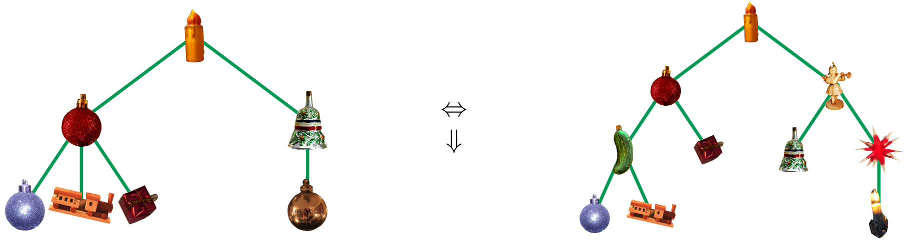
## Definition

A mapping of labels on forests $F_1$ and $F_2$ based on $(\mathcal{A} \cup \{-\}) \times (\mathcal{A} \cup \{-\})$ and restrictions $\pi_1(G)$ and $\pi_2(G)$ by considering either the first or the second coordinate, with $G$ being the resulting alignment forest.

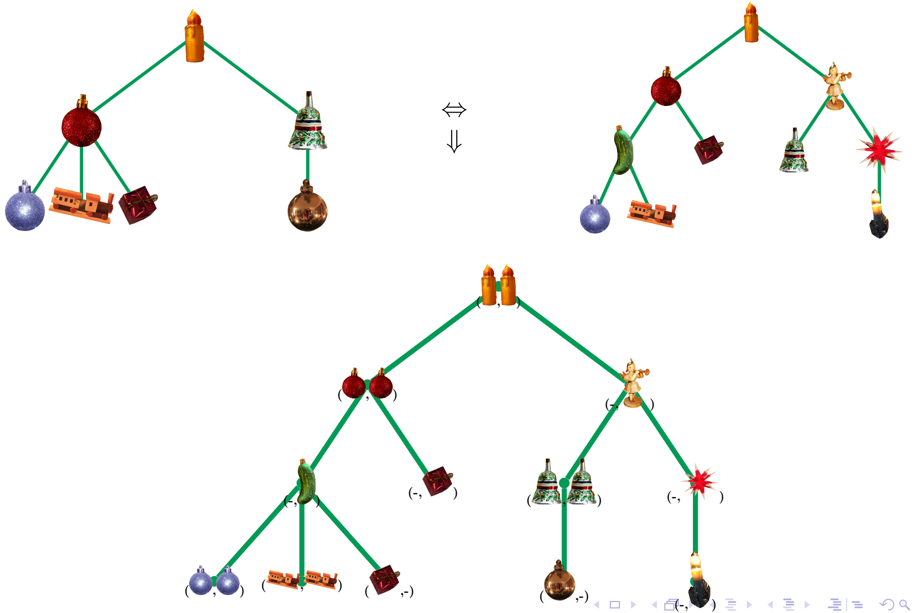



The cost of the alignment is the sum of the cost of label pairs.
$\Rightarrow$ can be done with a DP algorithm.

# Tree Alignment



# Tree Alignment



# Grammars

## Definition

Context-free grammars with production rules of the form $V \to \alpha$, where $V$ is a non-terminal and $\alpha$ is a string of terminals and/or non-terminals.

# Grammars

## Definition

Context-free grammars with production rules of the form $V \rightarrow \alpha$, where $V$ is a non-terminal and $\alpha$ is a string of terminals and/or non-terminals.

## The Formula

grammar $+$ scoring algebra $+$ index structure $=$ DP over arbitrary data structures

# Grammars

## Definition

Context-free grammars with production rules of the form $V \rightarrow \alpha$, where $V$ is a non-terminal and $\alpha$ is a string of terminals and/or non-terminals.

## The Formula

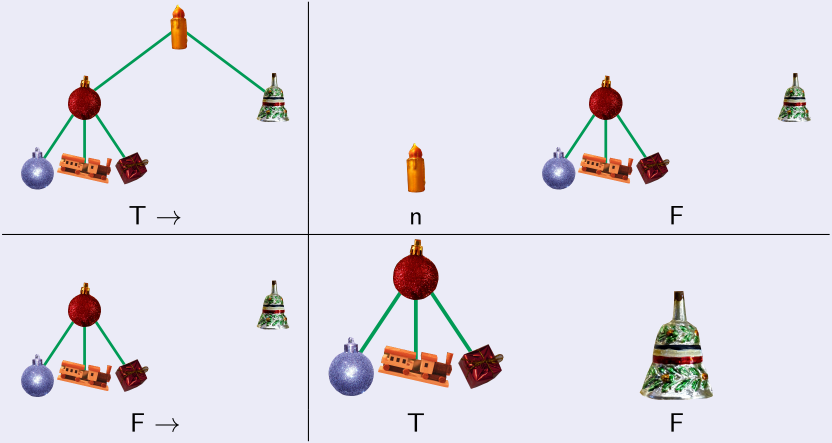
grammar + scoring algebra + index structure = DP over arbitrary data structures

## The Task

1. find a grammar describing your problem
2. find a scoring algebra
3. find an index structure and iteration rules for your data structure (if not done yet)

# Grammars

## Traversing Forests

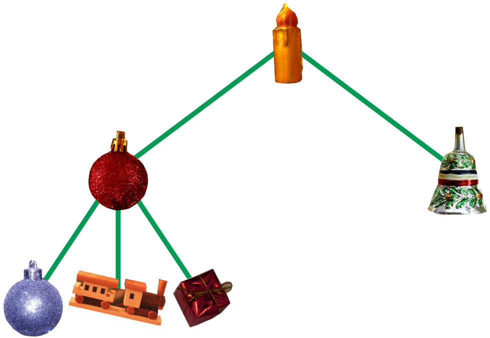


# Grammars

## Tree Editing

$$\begin{aligned}
\binom{S}{S} &\rightarrow \binom{F}{F} && [start]\\
\binom{F}{F} &\rightarrow \binom{\epsilon}{\epsilon} && [end]\\
\binom{F}{F} &\rightarrow \binom{F}{F} \circ \binom{T}{T} && [iter]\\
\binom{T}{T} &\rightarrow \binom{F}{F}\binom{x}{x} && [align]\\
\binom{T}{T} &\rightarrow \binom{F}{F}\binom{-}{x} && [indel]\\
\binom{T}{T} &\rightarrow \binom{F}{F}\binom{x}{-} && [delin]
\end{aligned} \tag{1}$$

The index structure is based on **postorder** of trees.

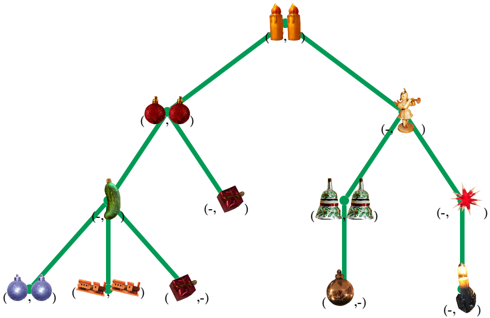


# Grammars

## Tree Alignment

$$
\begin{aligned}
\binom{S}{S} &\rightarrow \binom{F}{F} && [start] \\
\binom{F}{F} &\rightarrow \binom{\epsilon}{\epsilon} && [end] \\
\binom{F}{F} &\rightarrow \binom{T}{T} \circ \binom{F}{F} && [iter] \\
\binom{T}{T} &\rightarrow \binom{n}{n}\binom{F}{F} && [align] \\
\binom{T}{T} &\rightarrow \binom{-}{n}\binom{F}{F} && [indel] \\
\binom{T}{T} &\rightarrow \binom{n}{-}\binom{F}{F} && [delin]
\end{aligned}
\tag{1}
$$

The index structure is based on **preorder** of trees.



# Why?

- DP algorithms on trees:
  - small parsimony problem
  - phylogenetic targeting
  - tree editing
  - tree alignment (with affine gap costs)
- automatized DP on various data structures (in future)
- inside/outside: DP on probabilities

# Challenge

Convert this title:

*Why tree alignment doesn't have to suck*

# Challenge

Convert this title:

*Why tree alignment doesn't have to suck*

into something:

- referring to trees or forests
- but no other plants
- funny
- but not too funny ;)

The best submission will receive a *christmas cucumber*!

# Acknowledgements

## Thanks to...

- Peter Stadler
- Christian Höner zu Siederdissen

# Acknowledgements

## Thanks to...

- Peter Stadler
- Christian Höner zu Siederdissen

## ..and..

- Sophia for pictures,
- Rojin for planting trees,
- Marc for the story of 'Weihnachtsgurke'

# Acknowledgements

**Thanks to...**

- Peter Stadler
- Christian Höner zu Siederdissen

**..and..**

- Sophia for pictures,
- Rojin for planting trees,
- Marc for the story of 'Weihnachtsgurke'

**Thank you for your attention!**

# Proof for christmas cucumber!