

FreqT & Ekeko

Siegfried Nijssen

Simplest Approach

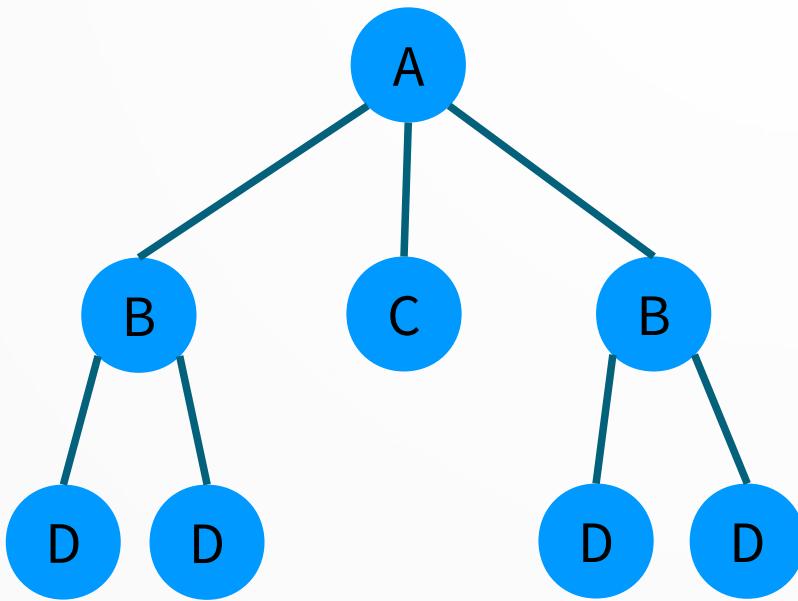
Overview

Use FreqT with tiny modifications to find patterns that can all be expressed as Ekeko patterns:

- 1) Given the original data
- 2) Transform this data
- 3) Run slightly modified FreqT on it
- 4) Turn its output in Ekeko patterns

Original Data: Assumptions

- Input #1: Node-labeled trees, in some order in the input file

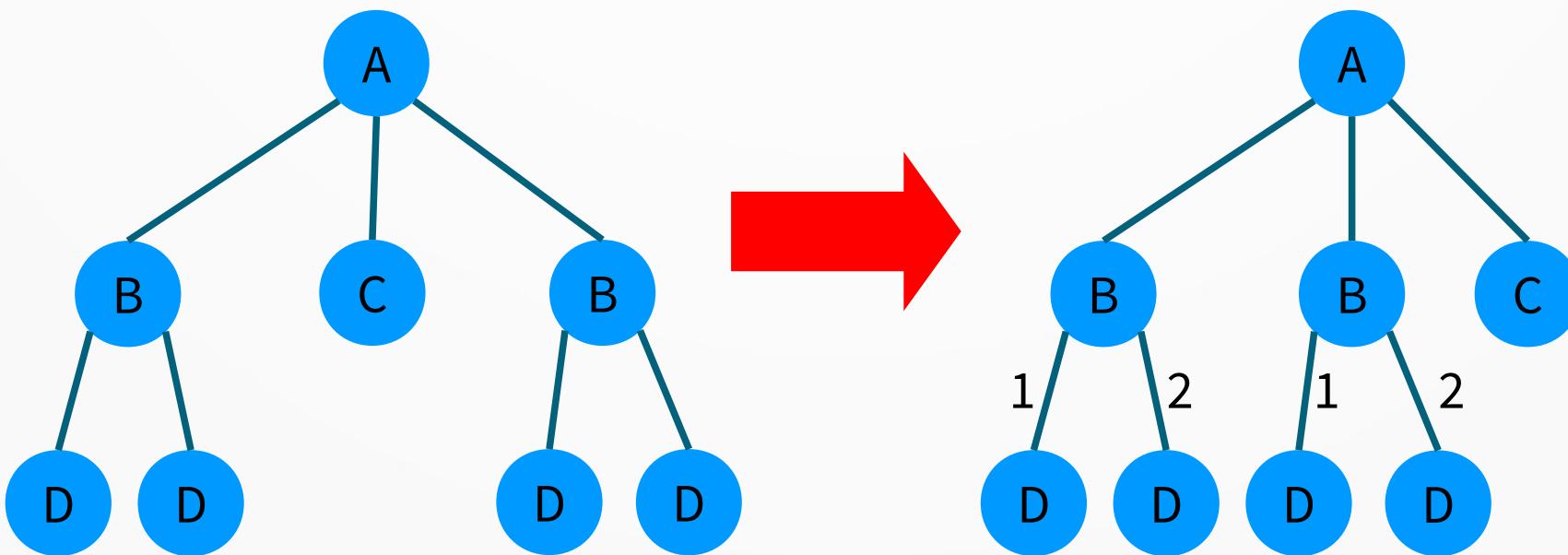


- Input #2: For every label, one of these two annotations:
 - (ordered, degree = n)
 - (unordered, degree ≥ 1)
 - (ordered, degree ≥ 1)

	= (unordered, degree ≥ 1)
	= (ordered, degree = 2)
	= (ordered, degree = 0)

Data Transformation

- For all nodes that are unordered, we will sort the children in increasing order of node labels
 - For nodes with fixed numbers of ordered children, add numbers as edge labels



Pattern & Matching Definition

- Patterns and matching: as in FreqT

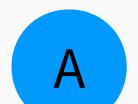
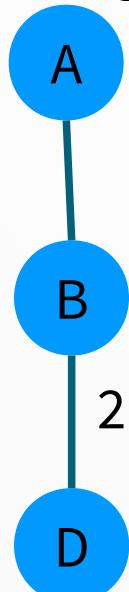


- A = (unordered, degree ≥ 1)
- B = (ordered, degree = 2)
- C D = (ordered, degree = 0)

pattern found using FreqT

FreqT Pattern → Ekeko Pattern

Tree found using induced subtree matching (FreqT)



= (unordered, degree ≥ 1)

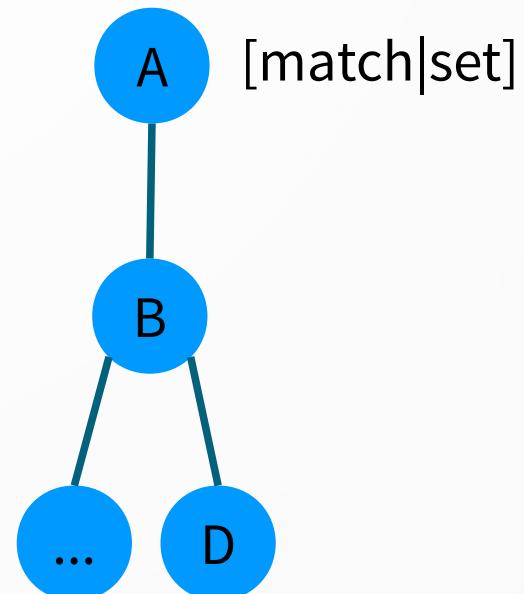


= (ordered, degree = 2)



= (ordered, degree = 0)

Corresponding Tree in
Ekeko

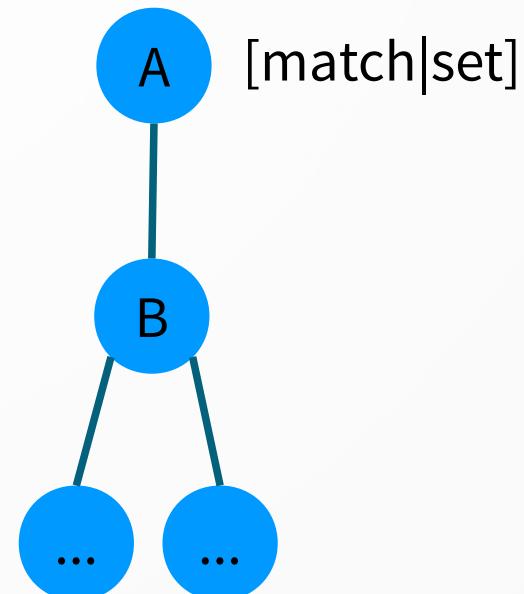


FreqT Pattern → Ekeko Pattern

Tree found using induced subtree matching (FreqT)



Corresponding Tree in
Ekeko



A = (unordered, degree ≥ 1)

B = (ordered, degree = 2)

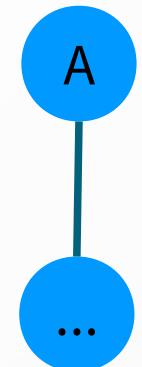
C D = (ordered, degree = 0)

FreqT Pattern → Ekeko Pattern

Tree found using induced subtree matching (FreqT)



Corresponding Tree in
Ekeko



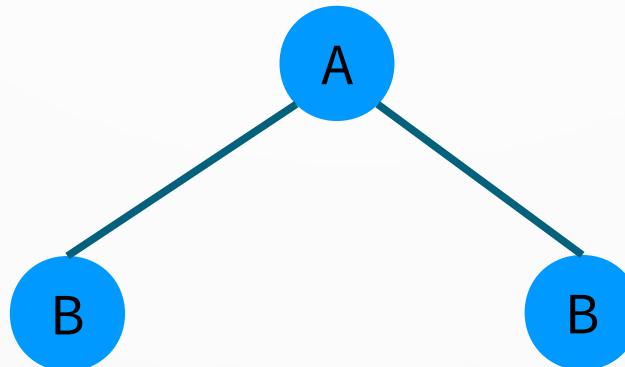
A = (unordered, degree ≥ 1)

B = (ordered, degree = 2)

C D = (ordered, degree = 0)

Constraints

- We need to impose two constraints on FreqT patterns in order to be able to transform them into (unordered) Ekeko patterns
- Constraint 1: don't allow two children of an unordered node to have the same label



As in FreqT patterns are ordered trees: the “sort-the-labels trick” does not work if two labels are identical

Constraints

- We need to impose two constraints on FreqT patterns in order to be able to transform them into (unordered) Ekeko patterns
- Constraint 2: don't allow two children of an (ordered, $\text{degree} \geq 1$) node

Such patterns, which FreqT could easily find, are hard to map into Ekeko patterns, as [match|set] always matches in an unordered fashion

- Note that we can also treat (ordered, $\text{degree} \geq 1$) labels as (unordered, $\text{degree} \geq 1$) labels to match Ekeko