

# Enumeration on Trees under Relabelings

Antoine Amarilli (LTCI, Télécom ParisTech, Université Paris-Saclay), Pierre Bourhis (CRISTAL, CNRS UMR 9189 & Inria Lille), Stefan Mengel (CNRS, CRIL UMR 8188)

## Problem Description

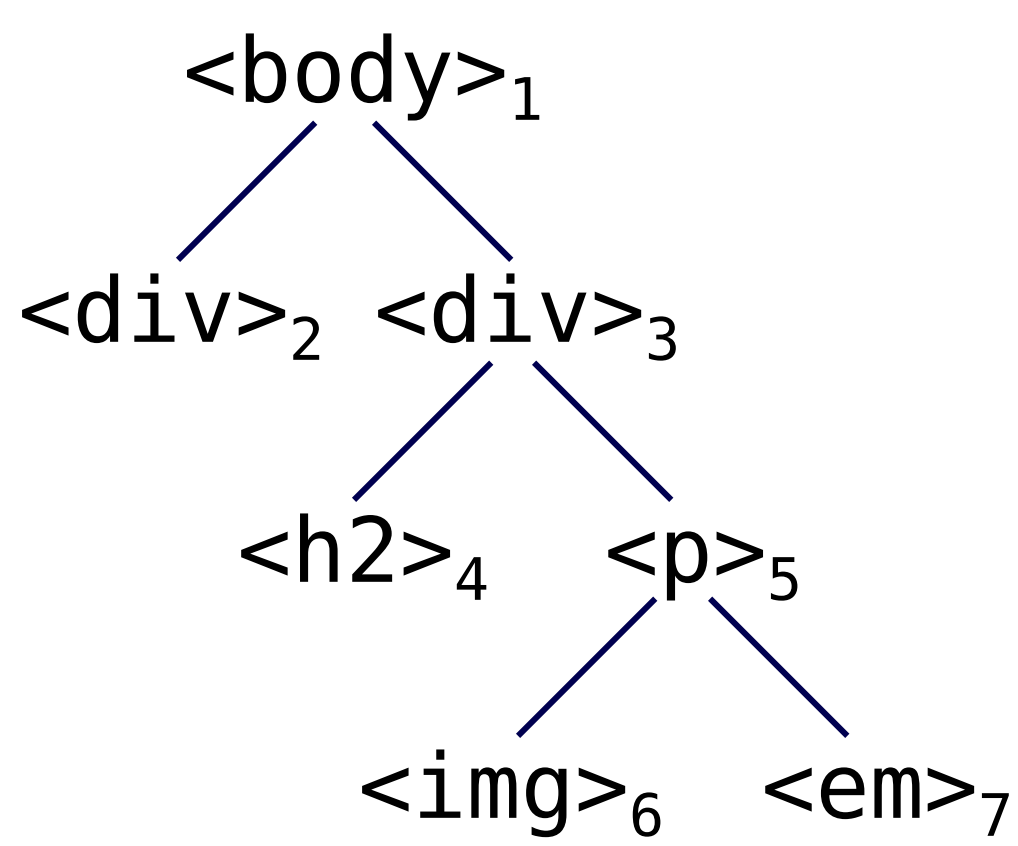
Compute all answers of an **MSO query** on an input **tree** and maintain them efficiently under **relabeling updates**

- **MSO = first-order logic + quantification over sets**

**Example:** find all pairs of a **h2 section title** and of an **image** located in the same section

- **Tree** with nodes labeled in a **fixed finite alphabet**
- **Updates** are **relabelings**: change the **label** of a node

**Example tree:** **Example answer set:**  $\{\langle 4, 6 \rangle\}$



**Example updates:**

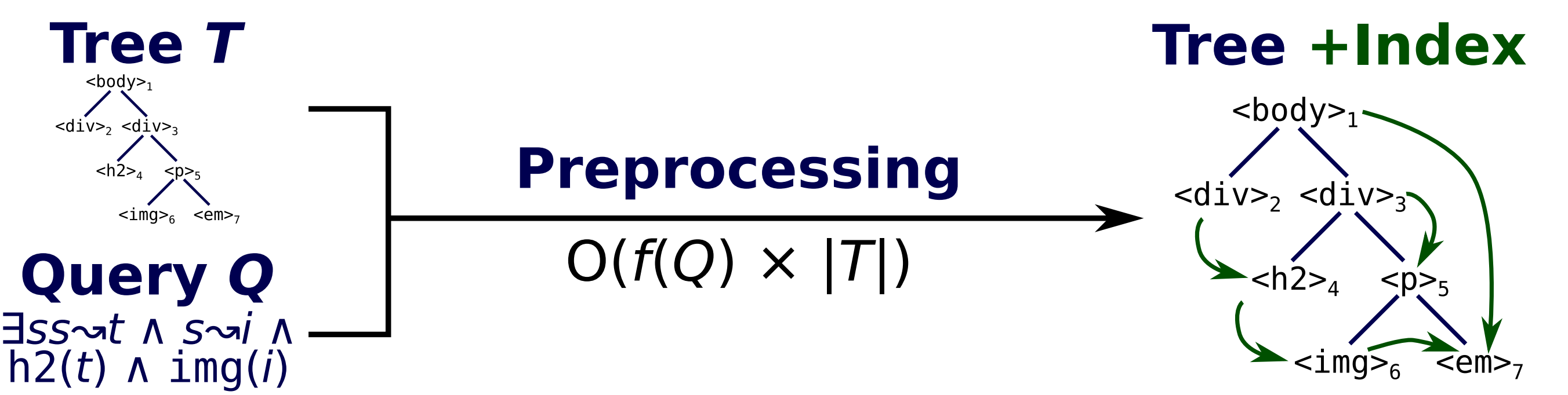
- **Relabel** node 7 to **<img>**  
**New answer set:**  $\{\langle 4, 6 \rangle, \langle 4, 7 \rangle\}$

- **Relabel** node 4 to **<h3>**  
**New answer set:**  $\emptyset$

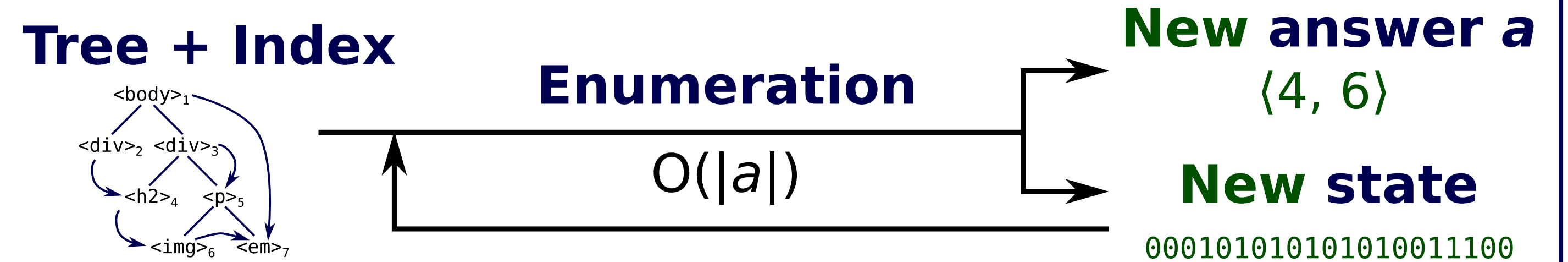
- **Complexity** is in the input tree, i.e., **data complexity**

## Enumeration Algorithm with Updates

- **Phase 1: Index** the input tree  $T$  for the query  $Q$



- **Phase 2: Enumerate** all query answers (no duplicates)



- **Phase 3: Update** the index when a node is relabeled



## Main Result and Existing Work Comparison

**Theorem** (Bagan in 2006; Kazana and Segoufin in 2013):

Enumerate all **answers** to an MSO query on a tree with linear preprocessing and delay **linear** in each answer (so **constant delay** if the free variables are first-order)

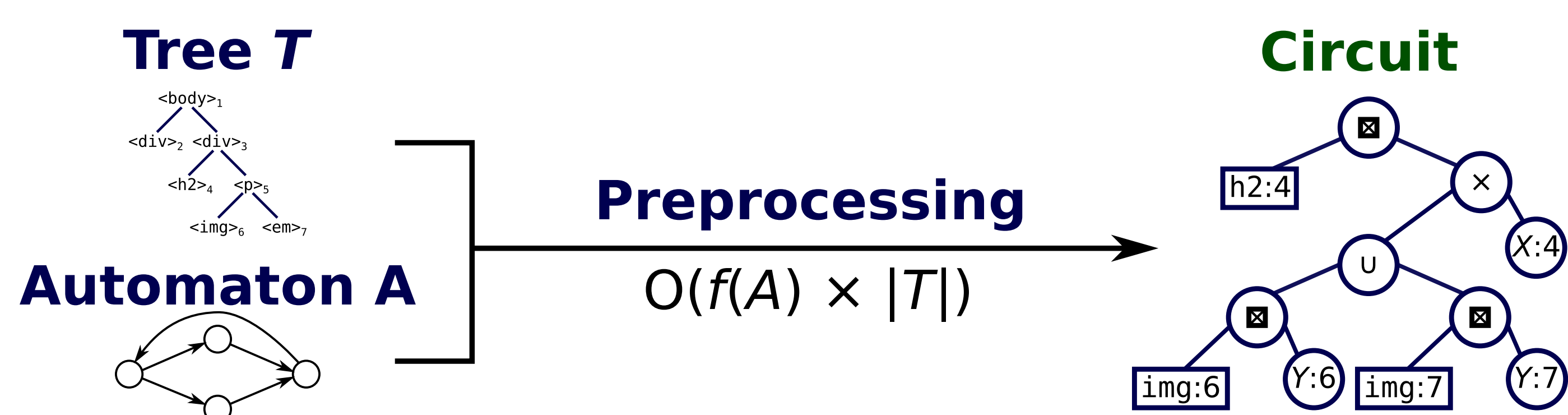
**Our main result:**

→ We can also handle **relabeling updates** in log time

Work	Delay	Updates
Bagan'06 and Kazana&Segoufin'13	$O(1)$	$O( T )$ : re-index the tree
Losemann&Martens'14	$O(\log^2  T )$	$O(\log^2  T )$
Niewerth&Segoufin'18	$O(1)$	$O(\log  T )$ , only on strings
<b>Our work</b>	$O(1)$	$O(\log  T )$ , only relabelings

## Proof Approach: Knowledge Compilation

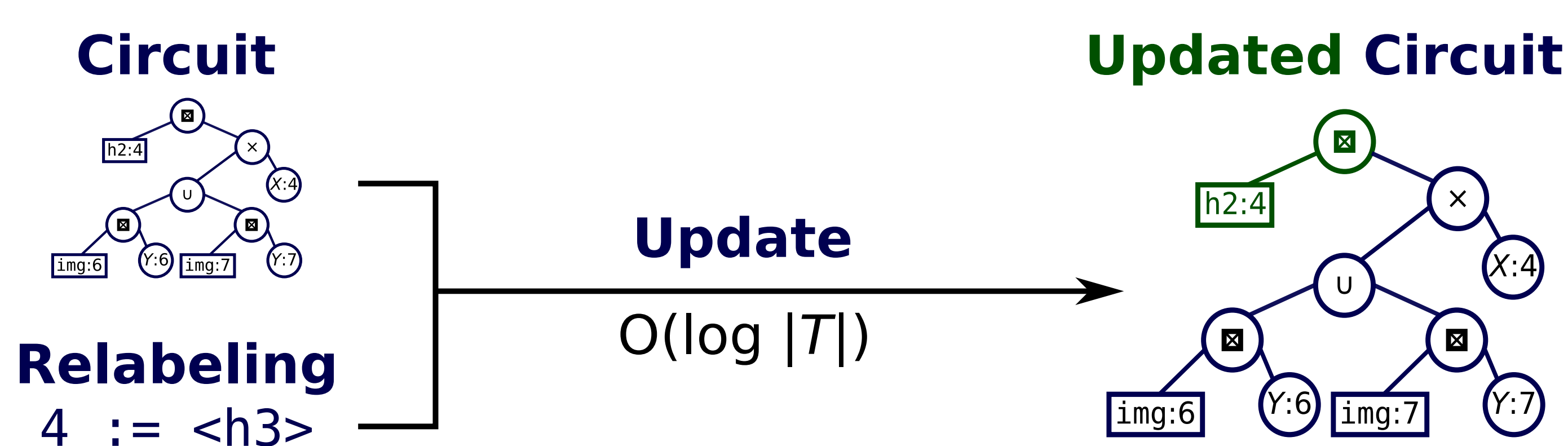
- **Phase 1: Preprocessing:** represent all query answers as a **circuit** that satisfies some structural **conditions**



- **Phase 2: Enumeration:** general enumeration method on any circuit that satisfies the conditions



- **Phase 3: Updates:** change the value of some variables and re-evaluate parts of the circuit



## Consequences for Application-Oriented Queries

- **Group-by aggregation:** For each **group** (*page section*):  
→ **how many** images are in the section? (*counting*)  
→ what is the **total size** of these images? (*sum*)

→ Enumerate **group-value pairs** with **delay**  $O(\log |T|)$

- **Parameters:** find all **images** in a **user-chosen section**

→ Enumerate answers with **constant delay** and **change the parameters** (=relabel) in  $O(\log |T|)$

## Proof Tool: Hybrid Circuits

- **Boolean gates:** Capture a Boolean value, depend on the **labeling**
- **Set-valued gates:** Capture a set of **answers** for each Boolean **valuation**

**Variable NOT AND OR**  $\boxed{img:6}$   $\neg$   $\wedge$   $\vee$  **Variable Union Product Test**  $\odot_{X:4}$   $\cup$   $\otimes$   $\boxtimes$

- **Hybrid circuit example**

**Valuation** of the **Boolean gates:**

$h2:4 := 1$   $img:6 := 1$   $img:7 := 0$

**Captured set:**  $\{\langle X:4, Y:6 \rangle\}$

