

Counting Relaxed Twig Matches of a Tree



Dongwon Lee
Penn State University

Divesh Srivastava
AT&T Labs - Research

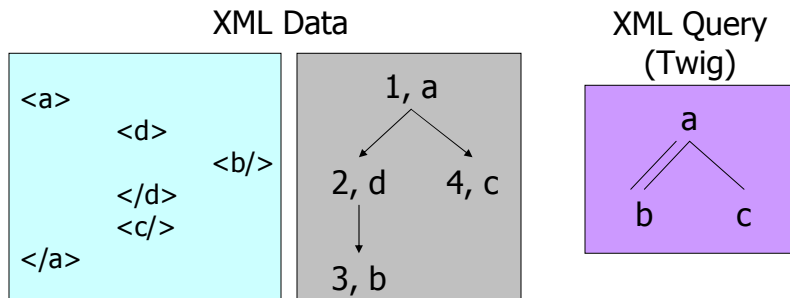


Outline

- [Background & Motivation](#)
- Problem Definition
- Our Approach
- Validation
- Summary

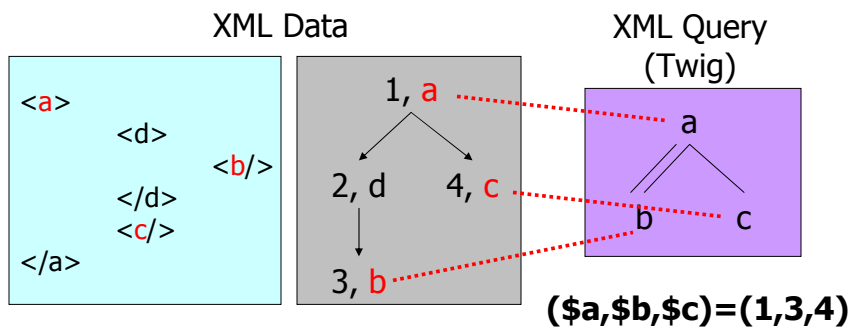
XML as Trees

- XML data/query can be represented as **Trees**
- Treat each element/attribute/string as a node with unique integer id , and value v : (id, v)



XML as Trees

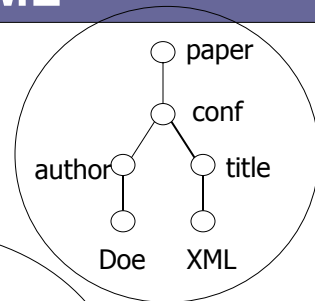
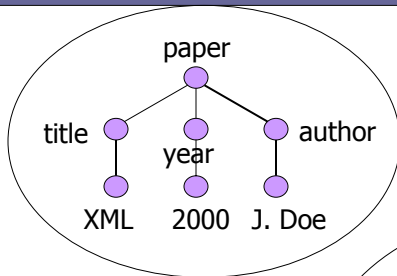
- XML data/query can be represented as **Trees**
- Treat each element/attribute/string as a node with unique integer id , and value v : (id, v)



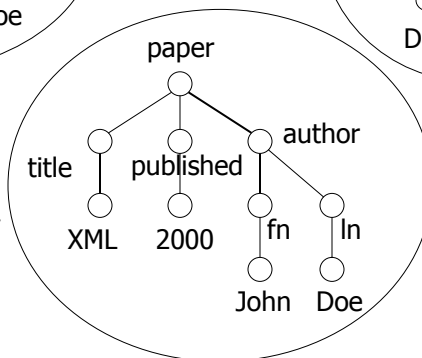
What is Relaxation?

- Looking for a non-stop flight from LA to DC for Saturday 10am
- Travel agent finds instead...
 1. A flight from LA to DC for Saturday **8am**
 2. A flight from LA to **Baltimore** for Saturday 3pm
 3. A flight from LA to DC via **Chicago** for Saturday 3pm
 4. An **Amtrak** train from LA to DC for Saturday 3pm
- Query S is **relaxed** to S' : $\langle S \rangle \subseteq \langle S' \rangle$

Query Relaxation for XML



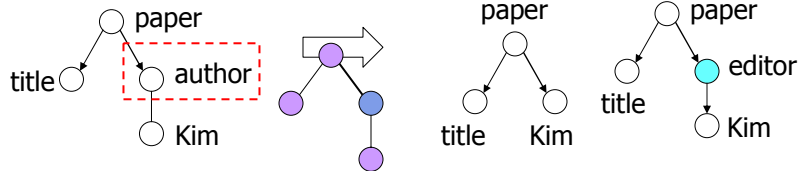
- XML schema is substantially bigger and more complex
- XML allows varied or missing structures and values



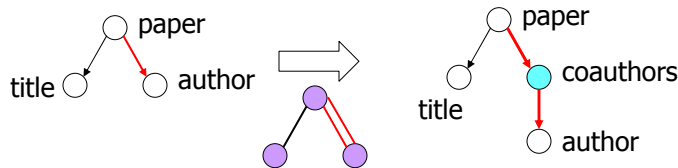
Two Tree Relaxations Types That We Focus



- **Node Relaxation:** allow node v to be **don't-care**



- **Edge Relaxation:** relax **child** to **descendant** relationships



Why need Selectivity Estimation?

- Two important problems in Query Relaxation
 - **Threshold problem:** find all approximate answers with $\text{sim} > \theta$
 - **Top- k problem:** find the best k approximate answers
- More efficient to evaluate Threshold problem than Top- k problem
 - Threshold θ enables the early pruning

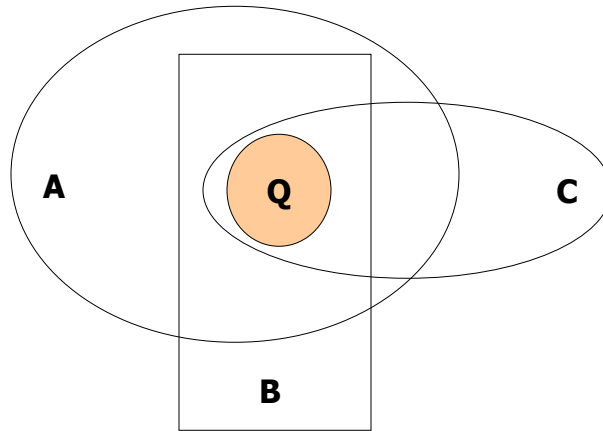
Why need Selectivity Estimation?

- Solving **Threshold problem**:
 - From the original query Q
 - Find all relaxed queries $R_1 \dots R_n$ with $\text{sim} > \theta$
 - $\text{Answer} = \langle R_1 \rangle \cup \dots \cup \langle R_n \rangle$
- Solving **Top-k problem**:
 - For each threshold θ , find all relaxed queries $R_1 \dots R_n$ with $\text{sim} > \theta$
 - Identify the largest threshold θ such that $\text{sel}(R_1 + \dots + R_n) > k$
 - Solve Threshold problem

Outline

- Background & Motivation
- **Problem Definition**
- Our Approach
- Validation
- Summary

Problem Illustration



$$\text{sel}(A+B+C) = \text{sel}(A) + \text{sel}(B) + \text{sel}(C) - \text{sel}(\text{overlap})$$

Relaxed Query Selectivity Prob.


Problem Definition: Given an original query Q and a set of Q 's relaxed queries R_Q , estimate the total number N of the combined matches of R_Q :
$$N = \text{sel}\left(\sum_{r \in R_Q} r\right)$$

Issues

Existing techniques cannot handle this

As little change as possible

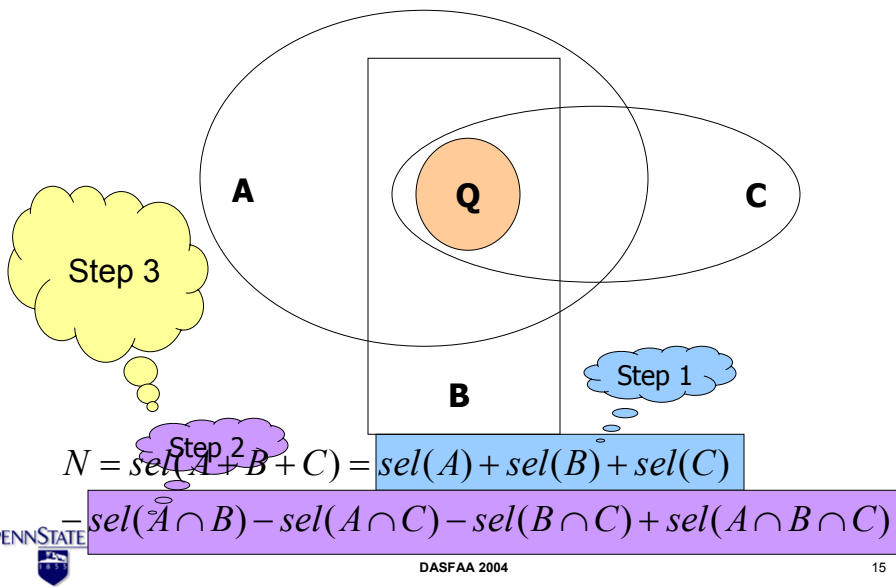
$$N = sel(A + B + C) = sel(A) + sel(B) + sel(C) - sel(A \cap B) - sel(A \cap C) - sel(B \cap C) + sel(A \cap B \cap C)$$

PENNS  DASFAA 2004 13

Outline

- Background & Motivation
- Problem Definition
- **Our Approach**
- Validation
- Summary

Our Approach



Our Approach

- Step 1: Using conventional selectivity estimation techniques, compute $sel(R_i)$ for each relaxed query R_i
 - We use CST method [Chen et al: ICDE 02]
- Step 2: Estimate $sel(\text{overlap})$ by converting the overlap into some formula without intersection operator
- Step 3: Compute $N = sel(R_1) + \dots + sel(R_n) - sel(\text{overlap})$

Step 2: computing sel(overlap)

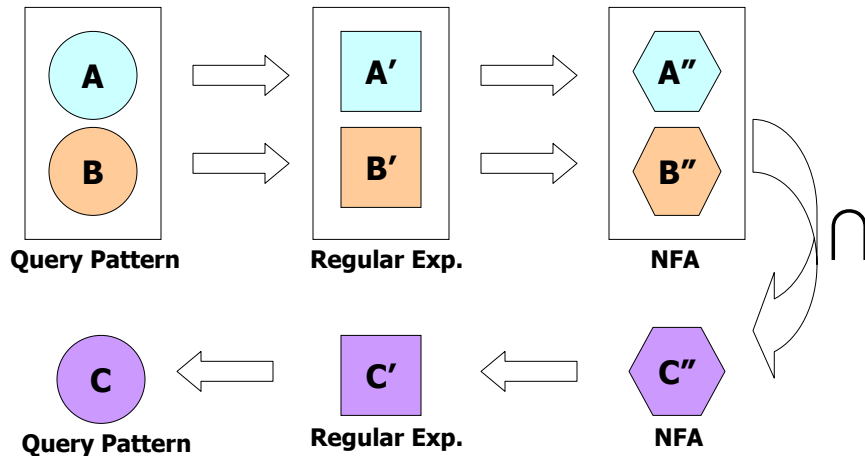
- In Step 1, we computed $\text{sel}(R_i), \dots, \text{sel}(R_j)$
- In Step 2, we need to compute

$$\text{sel}(\text{overlap}) = \text{sel}(\underbrace{R_i \cap \dots \cap R_j}_{\text{overlap}})$$

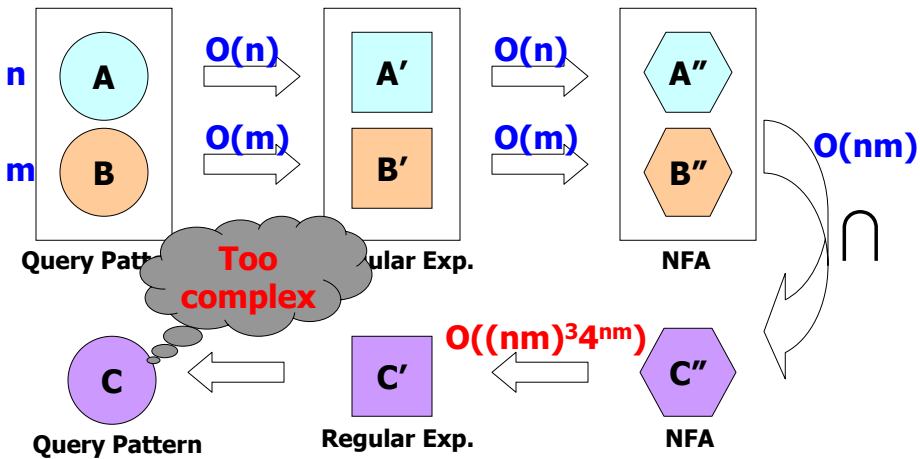
Challenge: How to find selectivity of overlap fast and accurately?

This form cannot be handled by current methods of XML selectivity estimation. Thus, we need to find other alternative overlap formula

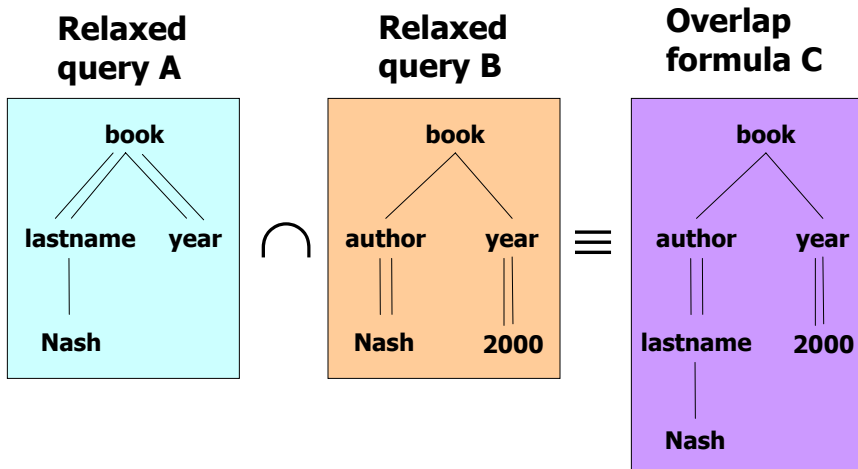
Solution 1: Automata Theory



Solution 1: Automata Theory



Solution 2: Using Query Property



Solution 2: Using Query Property

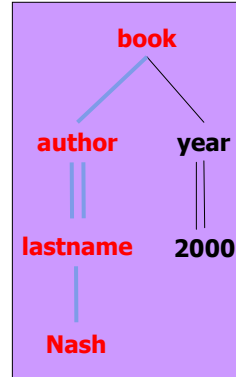
Relaxed query A



Relaxed query B



Overlap formula C

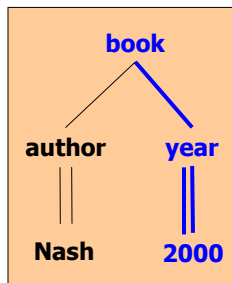


Solution 2: Using Query Property

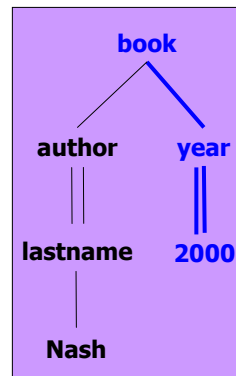
Relaxed query A



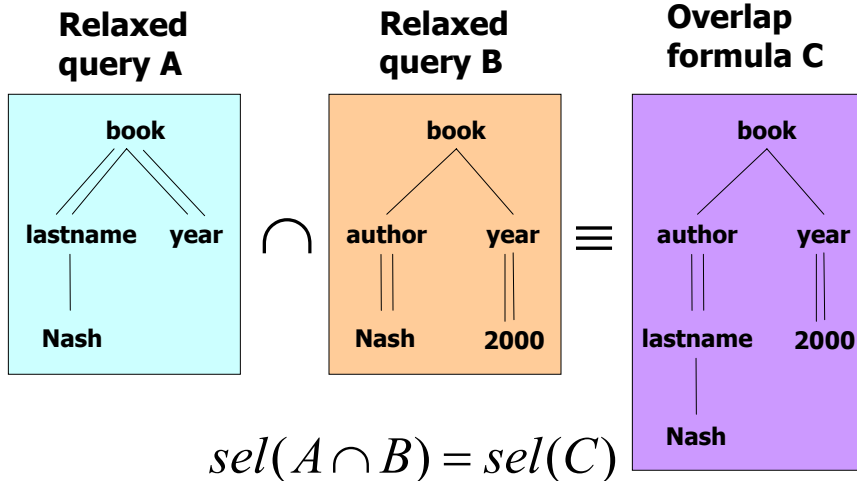
Relaxed query B



Overlap formula C



Solution 2: Using Query Property



Solution 2: Using Query Property

- We noticed that there are some patterns of twigs that lead to overlap formula
- E.g., Pattern 1: $R=A \times B$, $S=A \times B$
 - $R=/u[./b/c]$, $S=/u[./d/v/c] \Leftrightarrow \text{Overlap} = \{\}$
 - $R=/u[./b/c]$, $S=/u[./d/v/c] \Leftrightarrow \text{Overlap} = /u[./b/d/v/c]$
 - $R=/u[./b/c]$, $S=/u[./d/v/c] \Leftrightarrow \text{Overlap} = /u[./b/c/d/v/c]$
 - ...

• We identified 5 such patterns

Solution 2: Using Query Property

$$/ a[// b] \cap / a[// b // d] \equiv / a[// b // d]$$

$$/ a[/ b / c] \cap / a[// b // d] \equiv / a[/ b / c // d]$$

$$/ a[/ c] \cap / a[/ b // c] \equiv \{ \}$$

$$/ a[// b / c] \cap / a[/ d // c] \equiv / a[/ d // b / c]$$

...

Solution 2: Using Query Property

- Algorithm: Divide and Conquer style

- FindOverlapFormula (R_a, R_b)
 - Split inputs R_a, R_b to basic building blocks that match one of the 5 patterns
 - For each matched pattern, correct overlap formula is carefully composed
 - All overlap formulas are weaved together to lead to final overlap formula

Final: computing $sel(R_i + \dots + R_j)$

- In Step 1, we computed $sel(R_i)$, ..., $sel(R_j)$ using CST Tree
- In Step 2, we compute the overlap formula
$$sel(overlap) = sel(R_i \cap \dots \cap R_j) = sel(O)$$
- Then, use the set theory principle to compute the final selectivity estimate

$$\begin{aligned} N = sel(R_i + \dots + R_j) &= sel(R_i) + \dots + sel(R_j) \\ &- sel(R_i \cap R_{i+1}) \dots - sel(R_{j-1} \cap R_j) \dots \\ &\pm sel(R_i \cap \dots \cap R_j) \end{aligned}$$

Outline

- Background & Motivation
- Problem Definition
- Our Approach
- **Validation**
- Summary

Experimental Results

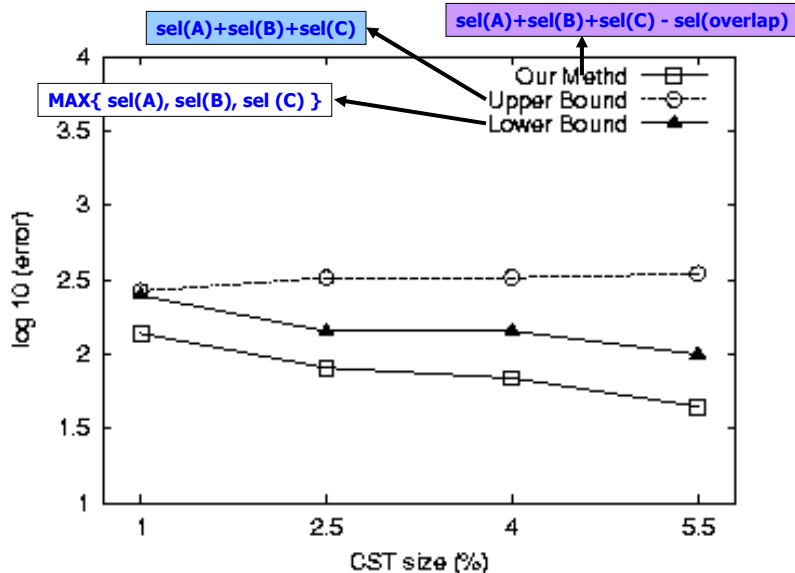
- Tested on two real datasets
 - DBLP (10MB), SPROT (5MB) datasets
- Tested various shapes of queries
 - PATH: path, DS: deep & skinny, BS: bush & shallow, BAL: balanced queries
- Tested different degrees of relaxations
 - A (w. least relaxation occurred), B, C, D (w. most relaxation)
- Used the **average relative squared error**:

$$Error = \frac{1}{|Q|} \sum_{q \in Q} \frac{(sel(q) - sel'(q))^2}{sel'(q)}$$

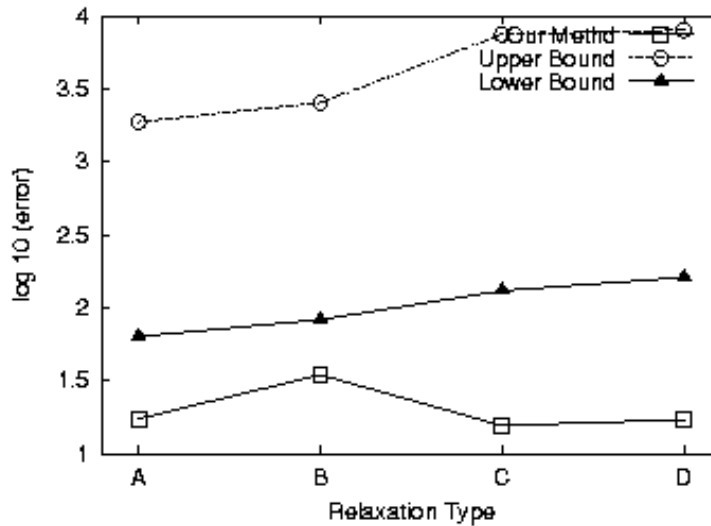
Q: test query set
 sel(q): true selectivity
 sel'(q): our estimate



Data=SPROT, Q=BAL, Type=C



Data=DBLP, Q=PATH



Outline

- Background & Motivation
- Problem Definition
- Our Approach
- Validation
- **Summary**

Summary

- Proposed a solution to **Relaxed Query Selectivity Problem**
 - Minimal change to existing methods
 - High accuracy
- Basis for better evaluation algorithms for
 - Threshold Problem
 - Top- k Problem
- Application
 - Query optimization
 - XML data filtering