

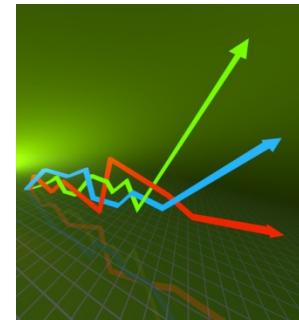
Continuous Queries over Text Streams

Vassilis Christophides

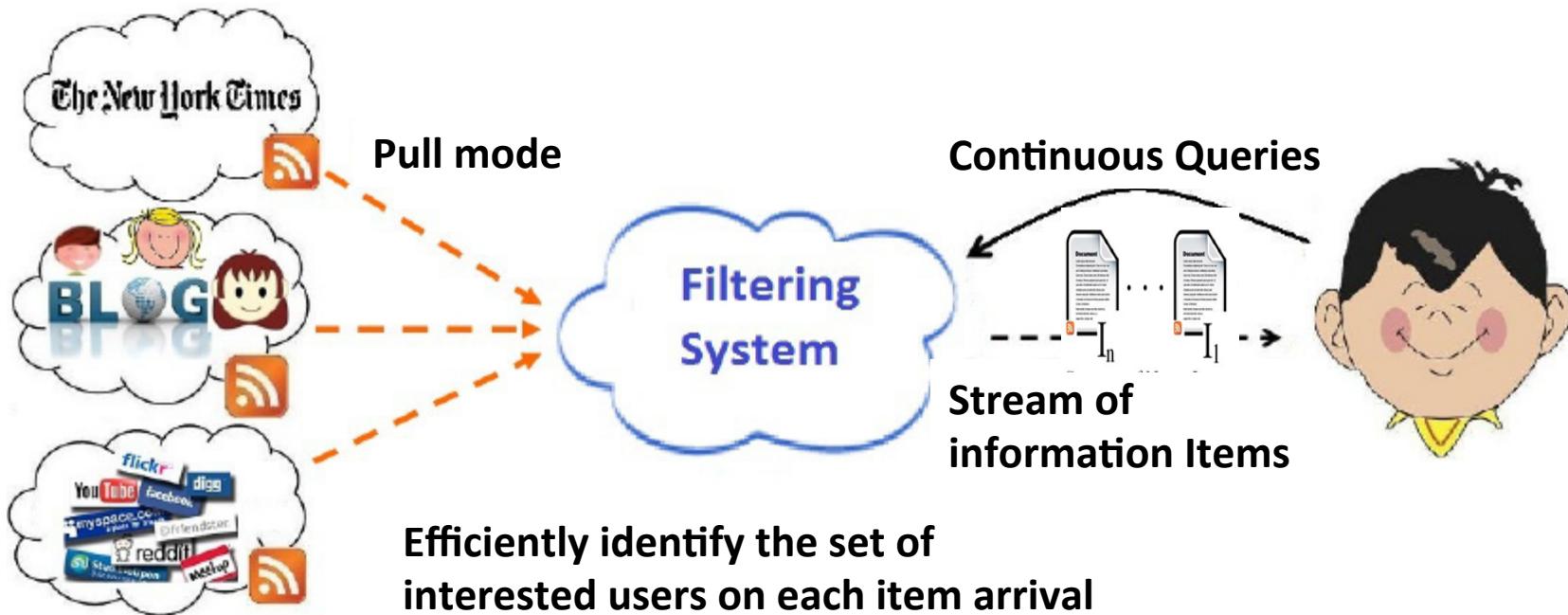
Joint work with Michel Scholl, Bernd Amann,
Cédric du Mouza, Nicolas Travers, Dan Vodislav

Text Streams in Web 2.0

- Thousands of professional news sources:
 - ~5000 (Google or Yahoo! News search)
- Millions of citizen news sources:
 - ~133,000,000 blogs (Technorati)
- A dozen of social media sources:
 - Twitter(~400M tweets/day), Facebook, LinkedIn(Alexa)
- Very large volumes of varying quality information published at different rates
 - Near Real Time Information Awareness



How Users can Find Information of Interest?



- Keyword-based continuous queries
 - Subscriptions in a content-based Pub/Sub system
 - Top-k queries using scoring functions
 - Algebraic plans in a web syndication system

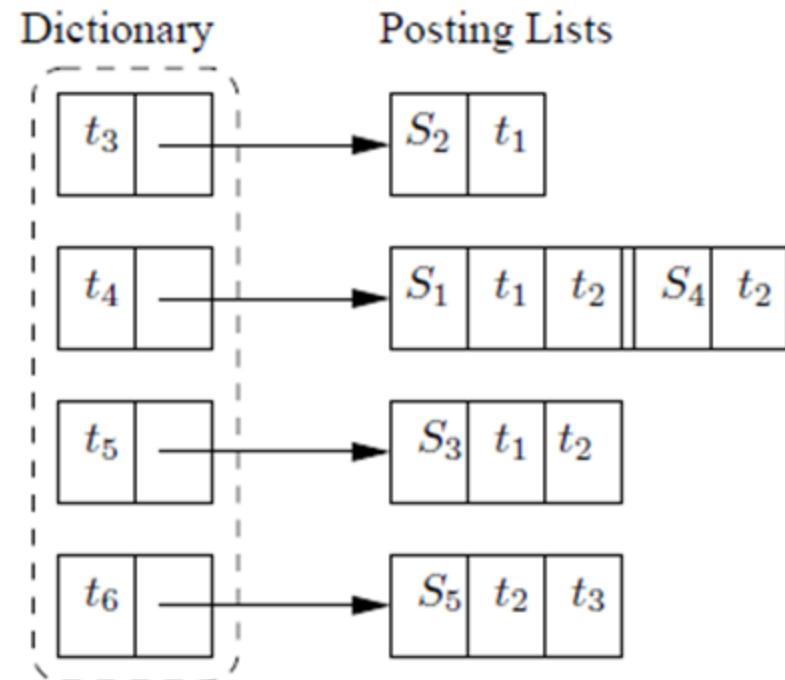
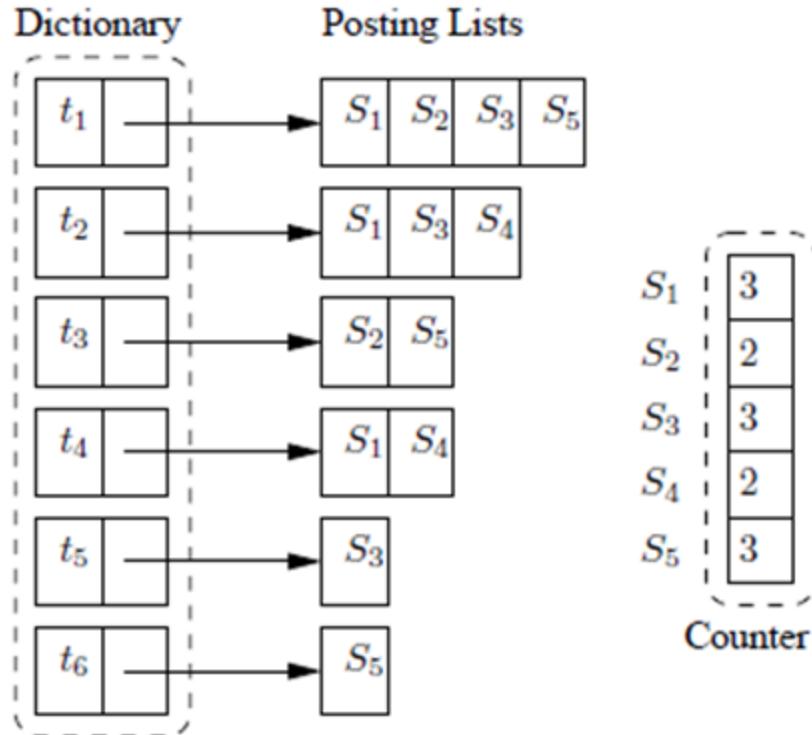
Subscriptions in a Pub/Sub system

- **Broad matching:** $S_i \subseteq I_k$ (requires counting)
 - the terms of a subscription S_i must be a subset of the terms appearing in an information item I_k
- **Challenge:** scalable index structures and efficient matching algorithms for ([HVTCMS11])
 - Millions of short size (2-3 terms) subscriptions submitted
 - High publication rates of medium-sized (52) items
 - Large vocabulary sizes of 1.5M terms

Index Structures for Conjunctive Textual Subscriptions [HKCMS12]

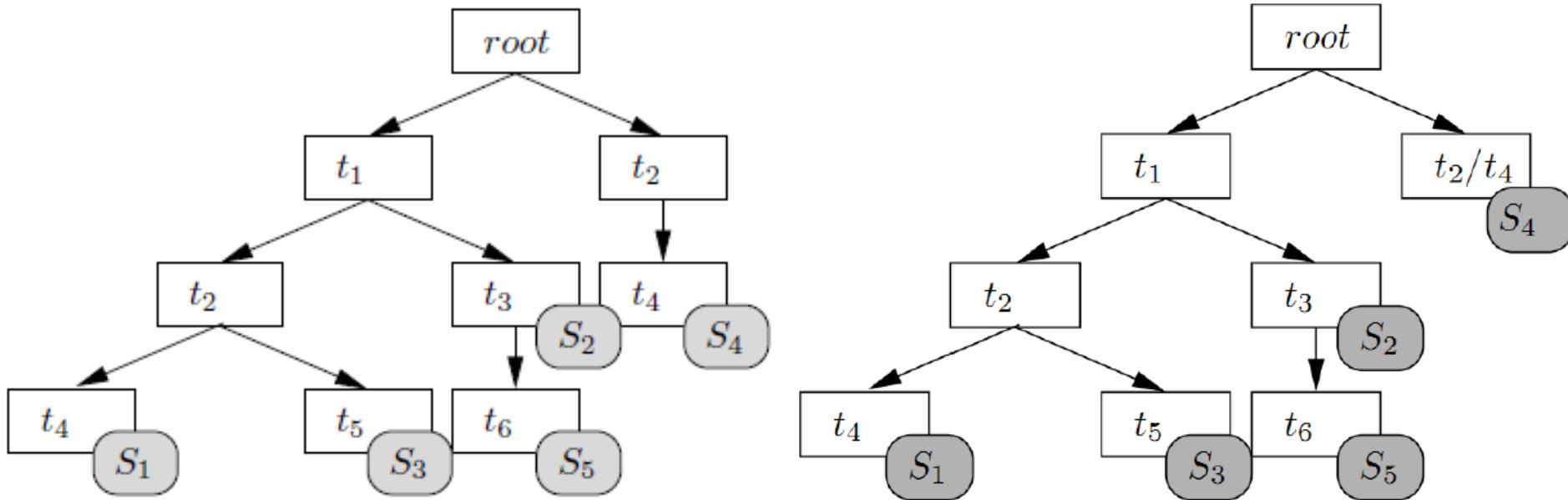
- Three indexes implementing different *counting techniques*
 - Count-based Inverted Lists: flat search space with explicit counting
 - Ranked-key Inverted Lists: nested search space with shorter posting lists for frequent terms
 - Patricia Ordered Trie: hierarchical search space with fast pruning of irrelevant subscriptions and implicit counting

Inverted Lists with or without Counting



$$\begin{aligned}
 S_1 &= t_1 \wedge t_2 \wedge t_4, S_2 = t_1 \wedge t_3, S_3 = t_1 \wedge t_2 \\
 &\wedge t_5, S_4 = t_2 \wedge t_4, S_5 = t_1 \wedge t_3 \wedge t_6
 \end{aligned}$$

Regular and Patricia Order-based Tries

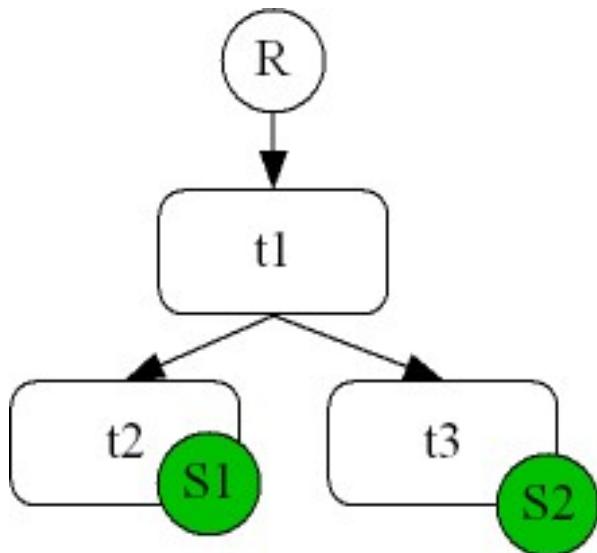


$$S_1 = t_1 \wedge t_2 \wedge t_4, S_2 = t_1 \wedge t_3, S_3 = t_1 \wedge t_2 \\ \wedge t_5, S_4 = t_2 \wedge t_4, S_5 = t_1 \wedge t_3 \wedge t_6$$

- We consider a **total ordering** amongst the vocabulary terms: $\forall t_k, t_l \in V_S, \text{ORDER}(t_l) < \text{ORDER}(t_k)$

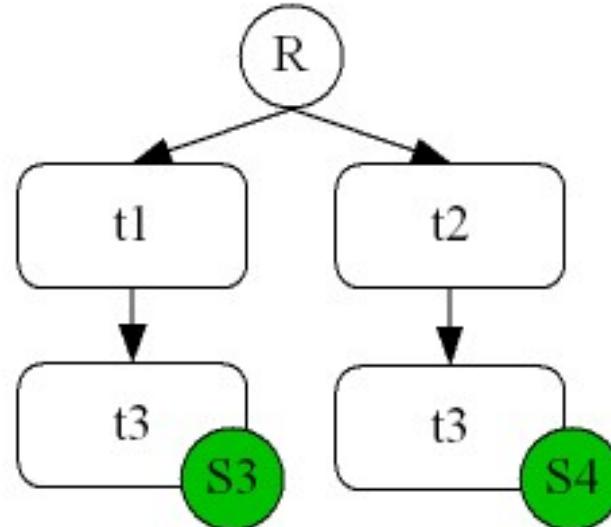
Common Prefix Factorization

- Any two subscriptions S_i and S_j share a prefix if and only if $\forall t_k \in (S_i \cup S_j) \setminus (S_i \cap S_j)$ and $\forall t_l \in (S_i \cap S_j) \Rightarrow \text{ORDER}(t_l) < \text{ORDER}(t_k)$



$$S1 = \{t2, t1\}$$

$$S2 = \{t1, t3\}$$



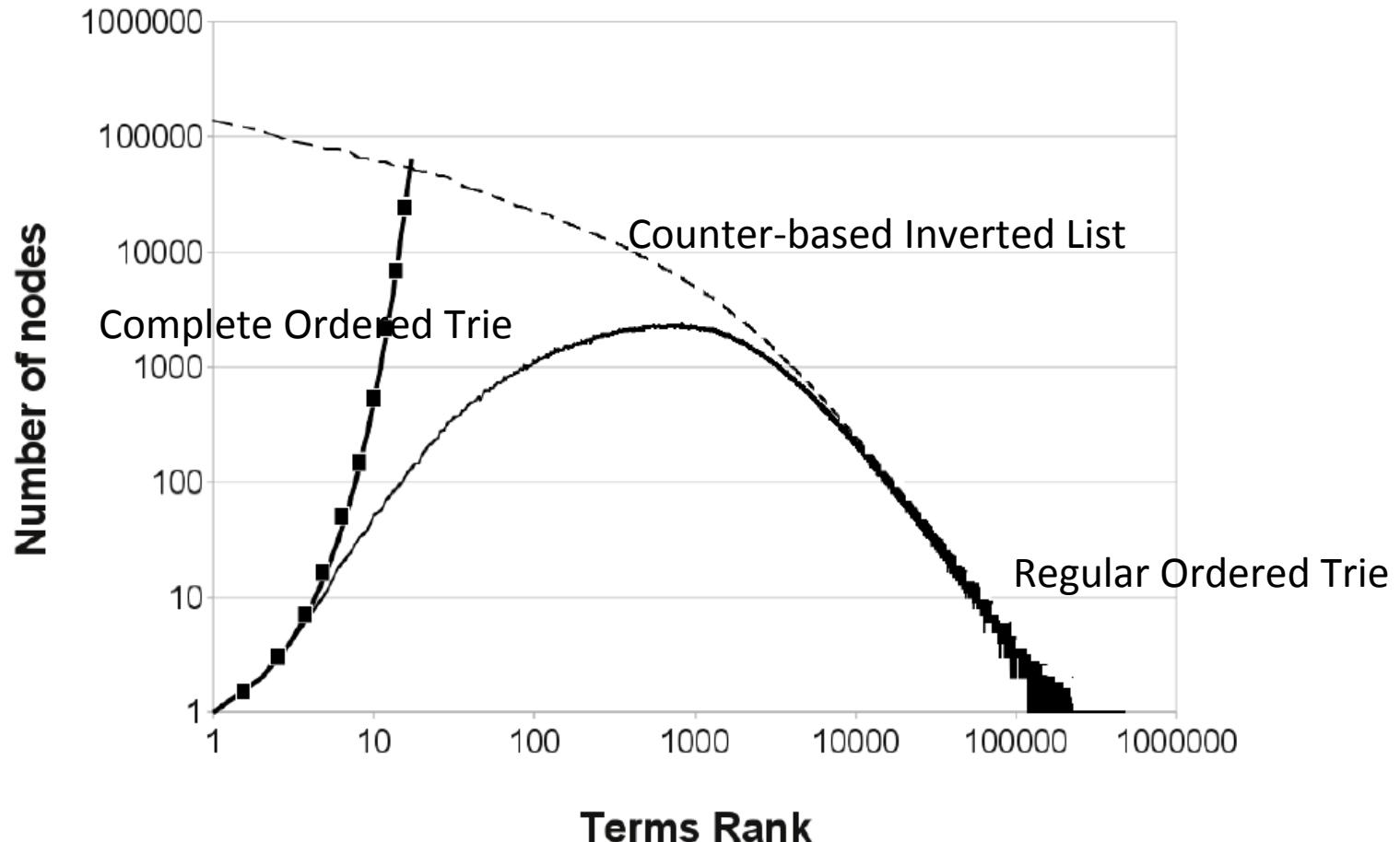
$$S3 = \{t3, t1\}$$

$$S4 = \{t3, t2\}$$

Index Evaluation

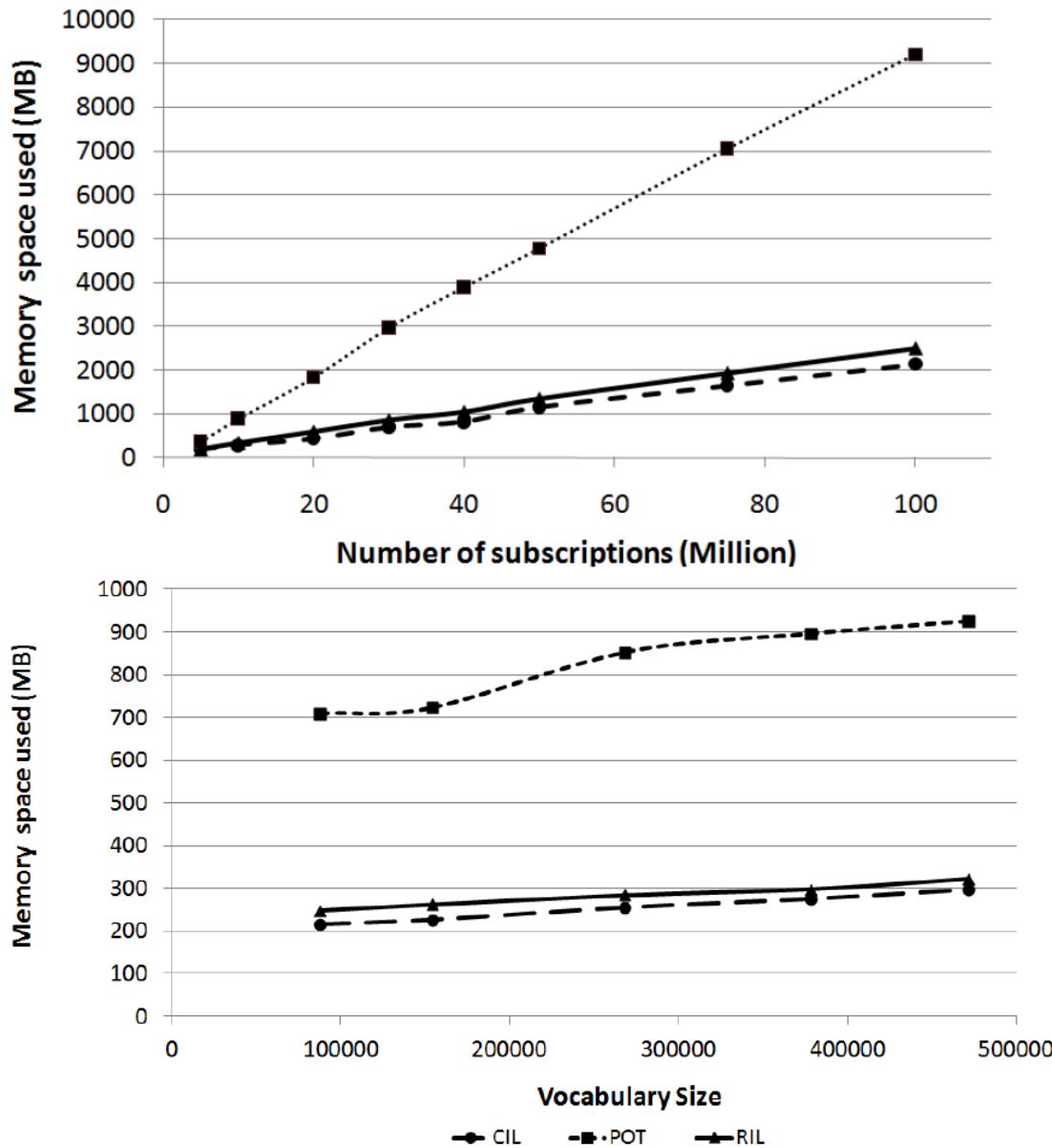
- Determine how the **morphology** of the indexes is affected by:
 - Subscription size
 - Terms' order
 - Terms' occurrences distribution
- Study the **scalability** and the **performance** of the indexes:
 - Number of subscriptions $|S|$
 - Vocabulary size $|V_S|$

Nodes vs Term's Rank



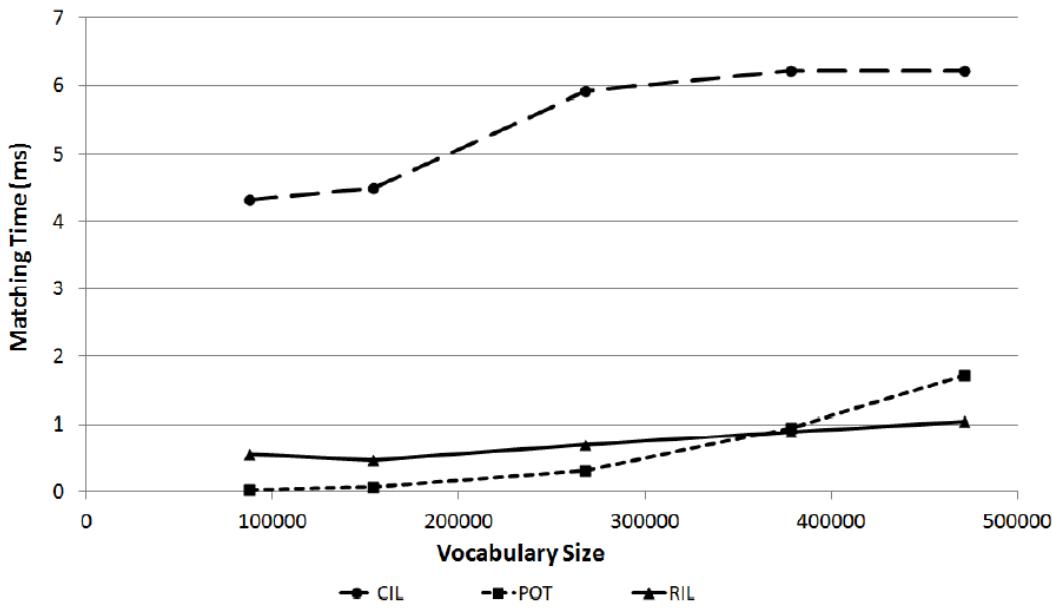
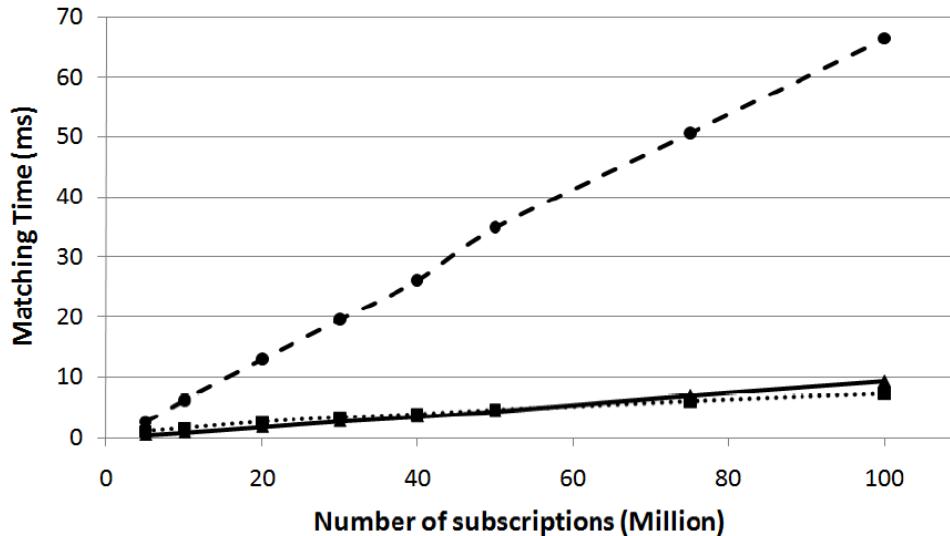
- Factorization gains depend on the subscription length
- Actual order of terms has limited impact on index size

Memory Footprint when Scaling $|S|$ & $|V_S|$



- The size of all indexes **scales linearly** with the # of subs
- Inverted lists are **less sensitive** to the size of vocabularies but strongly impacted by the number of subscriptions

Matching Time when Scaling $|S|$ & $|V_S|$



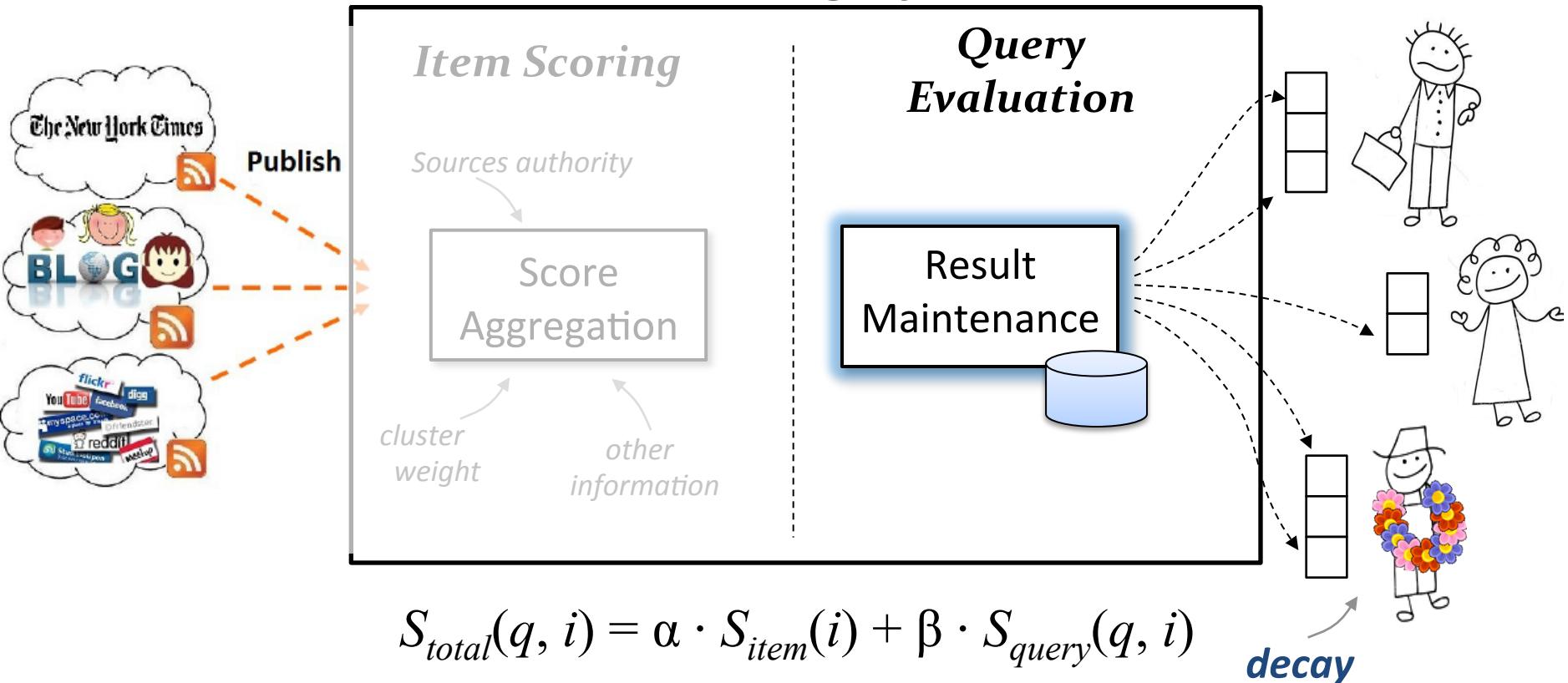
- Matching time in all indexes **scales linearly** with the # subs
 - POT outperforms RIL for large $|S|$
- POT outperforms RIL with one magnitude order for small vocabularies, RIL provides better performances for large ones

Putting All Together

- Trade-off between the memory footprint and matching performance of CIL vs. ROT indexes
 - RIL appears as a good compromise with memory footprint close to that of CIL and matching time close to ROT (and can even outperform ROT for some settings)
- Short subscriptions factorize more POT
- When terms' frequency distribution in subscriptions follows terms' distribution in items, the search space for frequent terms in CIL is extremely large w.r.t. ROT
 - When they follow the inverse distribution the number of visited nodes drastically drops for both indexes

Top-k Queries using Scoring Functions

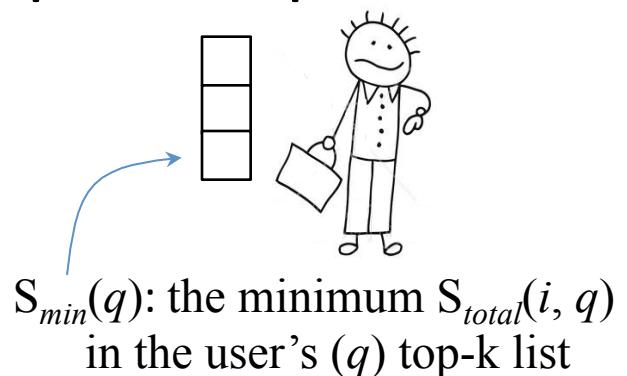
Real Time Filtering System



- **Challenge:** Top-k query result maintenance when queries feature non-homogeneous scoring functions

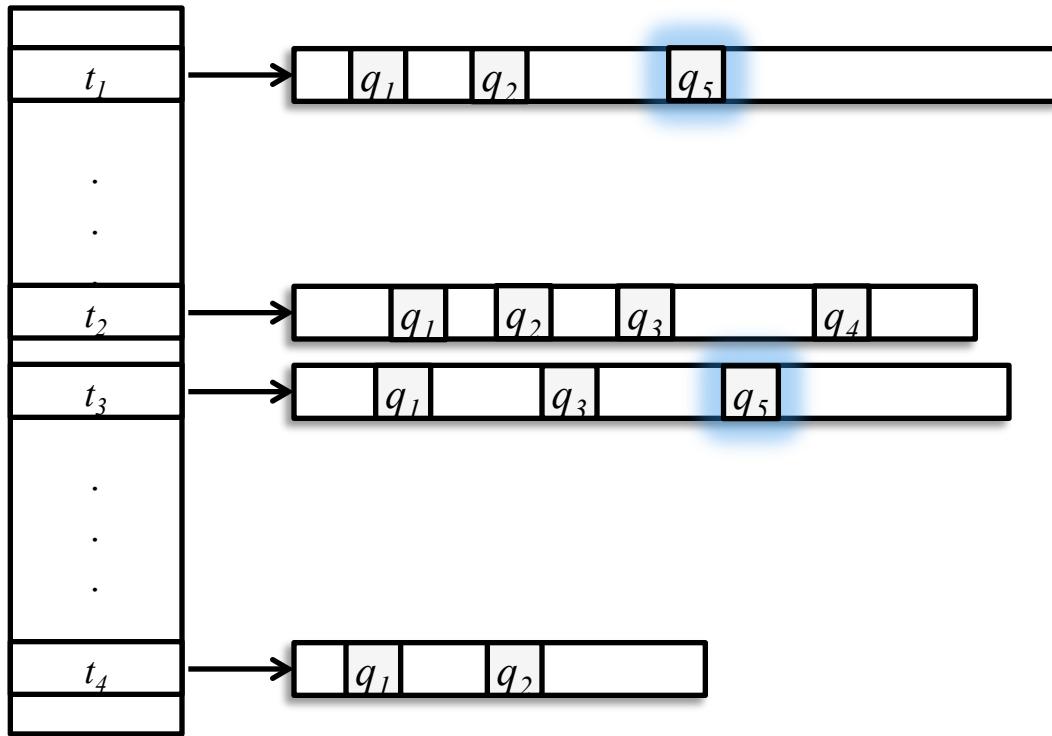
Continuous Top-k Textual Query Evaluation [VAC12]

- Given a set of queries Q , a decayed ranking function S_{decay} and an item stream $I(\tau)$, maintain for each query $q \in Q$ its top-k result $R(q, \tau, k)$ at any time instant τ
- We denote by $S_{min}(q, \tau)$ the score of the last item i in $R(q, \tau, k)$ at time instant τ :
 - For a set Q and an item i arriving at some timestamp τ_i , update the top-k result for all queries $q \in Q$ where $S_{total}(q, i) > S_{min}(q, \tau_i)$



One-dimensional Query Representation

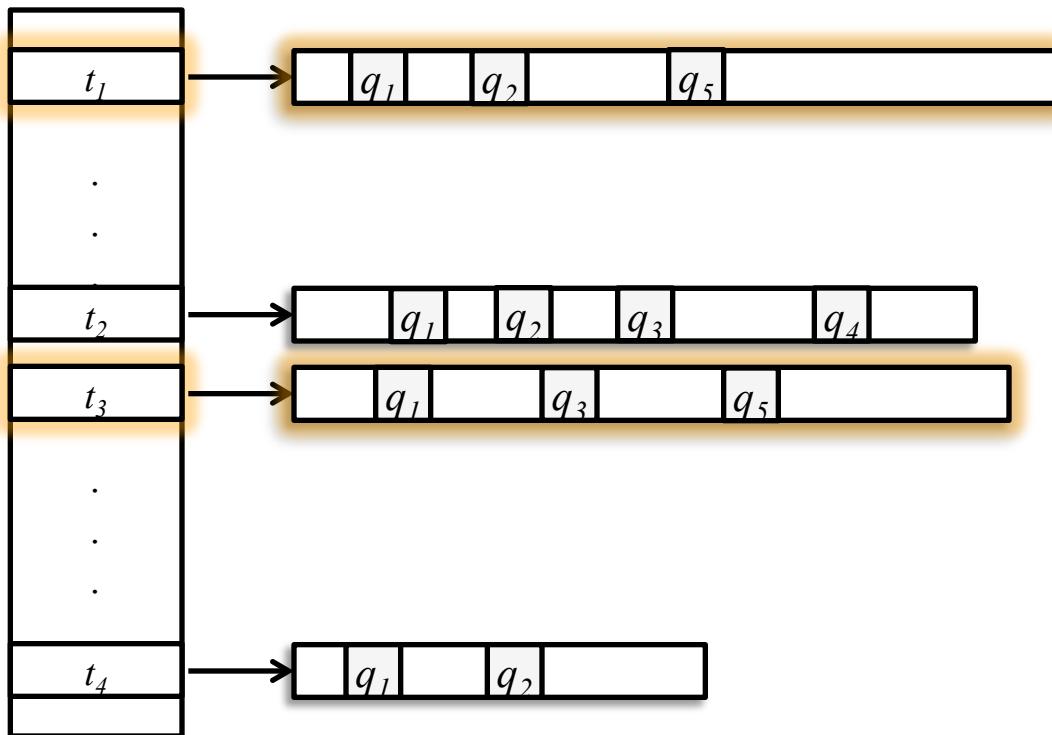
- A dictionary of terms to *queries*



$q_1 = \{t_1, t_2, t_3, t_4\}$
 $q_2 = \{t_1, t_2, t_4\}$
 $q_3 = \{t_2, t_3\}$
 $q_4 = \{t_2\}$
 $q_5 = \{t_1, t_3\}$

One-dimensional Query Representation

- A dictionary of terms to *queries*
 - On item arrival: $i = \{t_1, t_3, t_9, \dots\}$



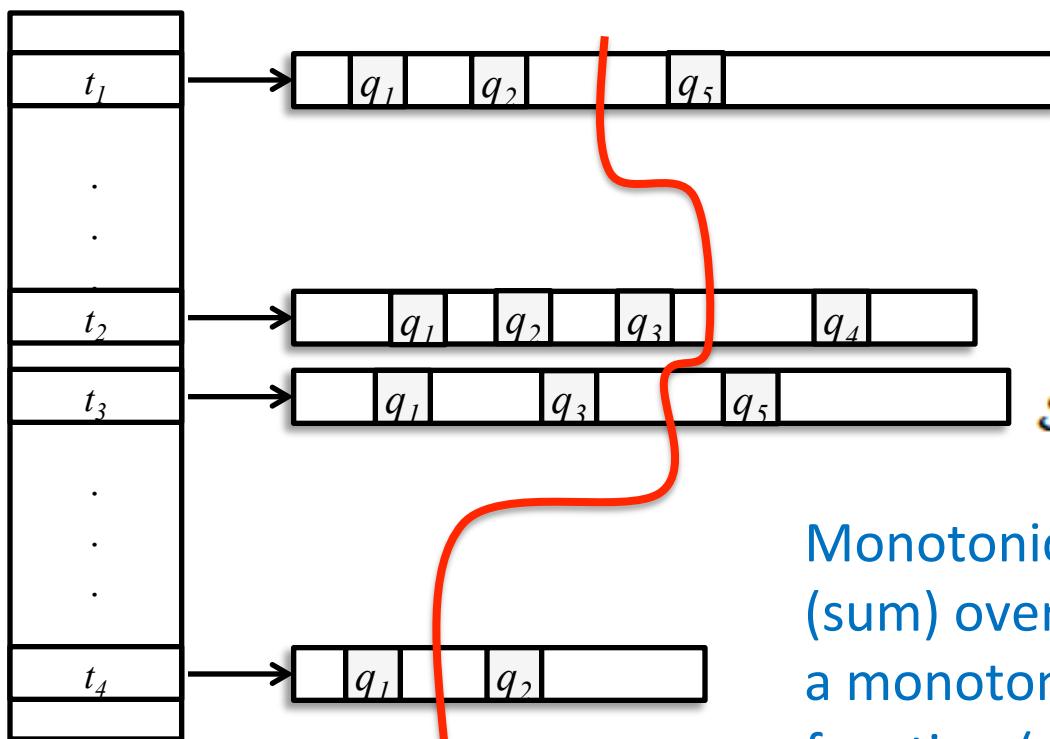
$$\begin{aligned}q_1 &= \{t_1, t_2, t_3, t_4\} \\q_2 &= \{t_1, t_2, t_4\} \\q_3 &= \{t_2, t_3\} \\q_4 &= \{t_2\} \\q_5 &= \{t_1, t_3\}\end{aligned}$$

$$S_{total}(q, i) > S_{min}(q)$$

One-dimensional Query Representation

- Baseline COL-Filter [Haghani10]:

- TA-like algorithm [Fagin01]
- Sorts queries by $w_{q,t}/S_{min}(q)$



$$\begin{aligned} q_1 &= \{t_1, t_2, t_3, t_4\} \\ q_2 &= \{t_1, t_2, t_4\} \\ q_3 &= \{t_2, t_3\} \\ q_4 &= \{t_2\} \\ \mathbf{q}_5 &= \{\mathbf{t}_1, \mathbf{t}_3\} \end{aligned}$$

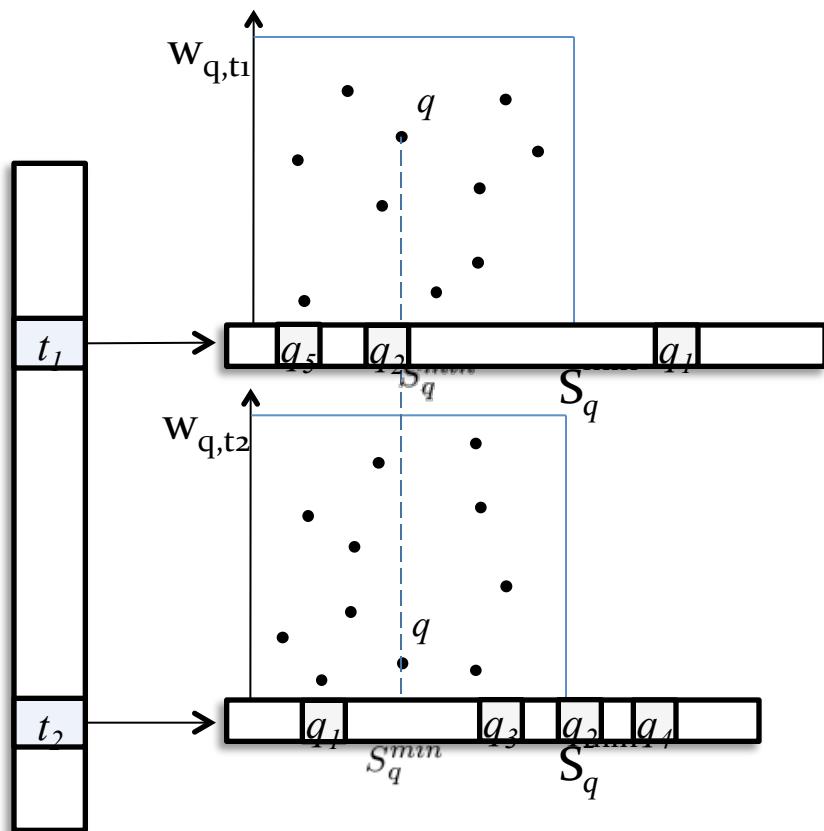
considers only query-dependent scores : cosine similarity

$$S_{query}(q, i) = \sum_{t' \in V} \omega_{q,t'} \cdot \omega_{i,t'}$$

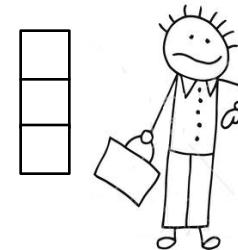
Monotonic aggregation function (sum) over a set of scores obtained by a monotonic weight combination function (multiplication)

Two-dimentional Query Representation

$$S_{total}(q, i) = \alpha \cdot S_{item}(i) + \beta \cdot S_{query}(q, i)$$



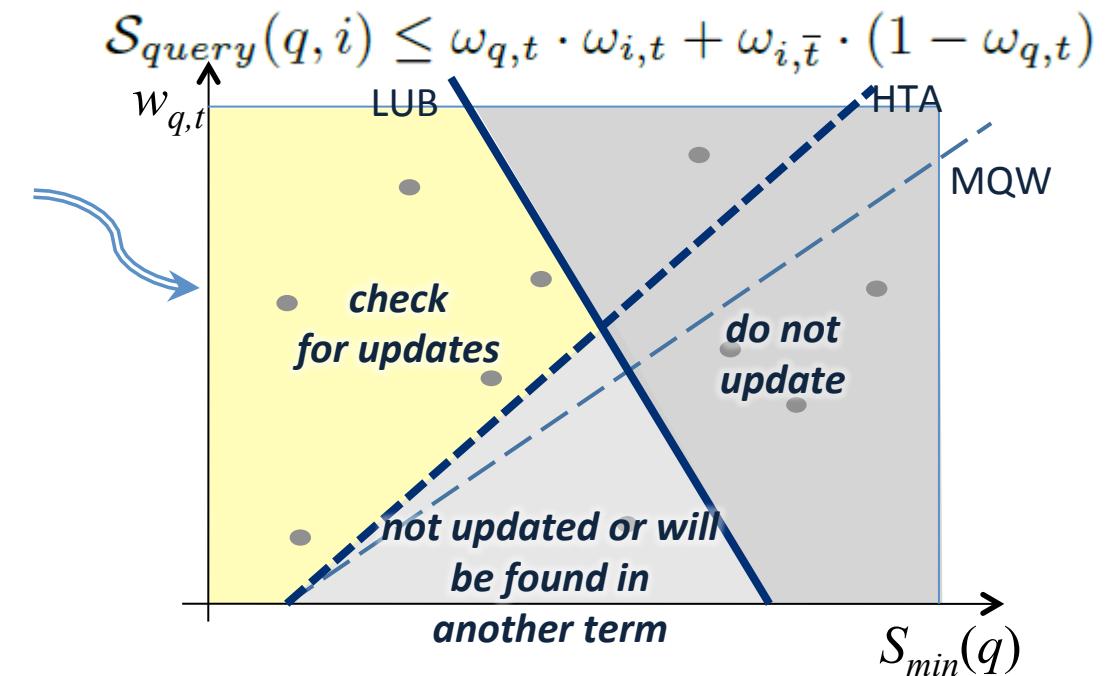
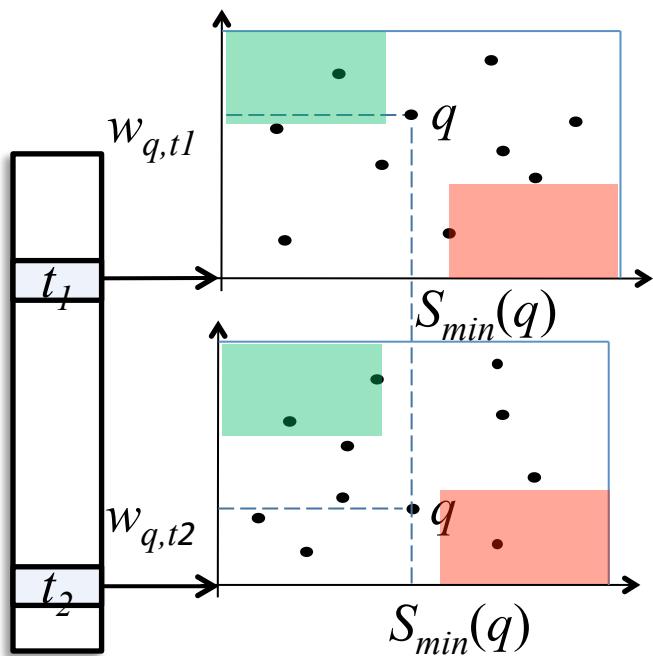
$$q = \{t_1, t_2\}$$



$S_{min}(q)$: the minimum $S_{total}(i, q)$ in the user's (q) top-k list

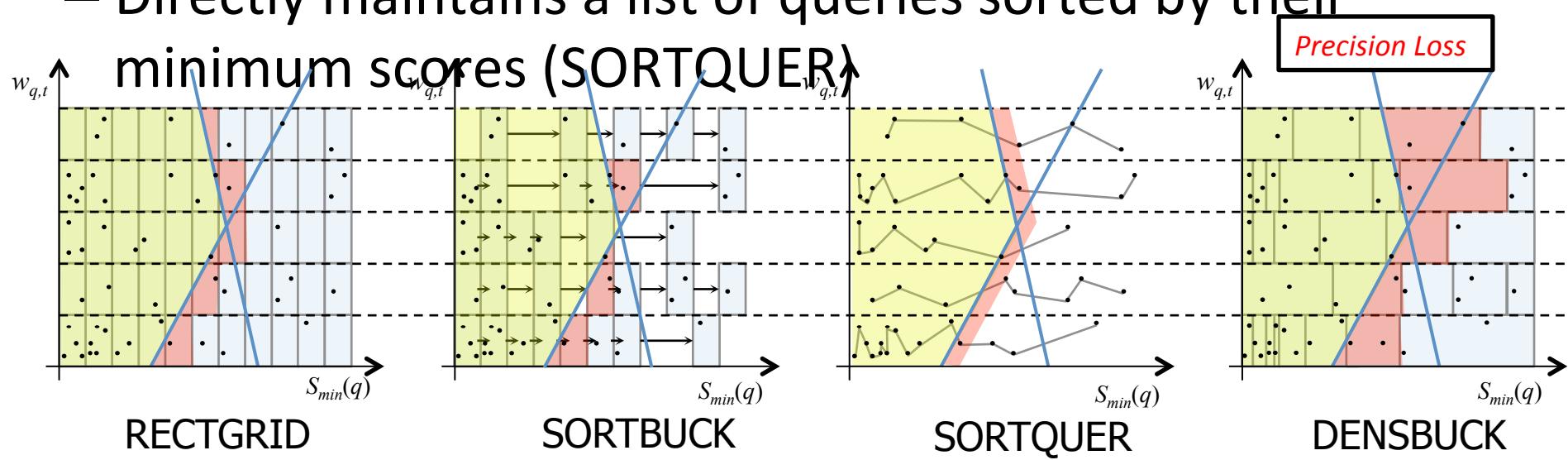
A Constraint-based Model for Non-Homogeneous Scoring Functions

- Suppose an item arrives: $i = \{t_1, t_2, \dots\}$
 - How can we bound the number of queries whose result list need to checked for possible updates?
 - We have proven a number of constraints depending on the term and the item we consider

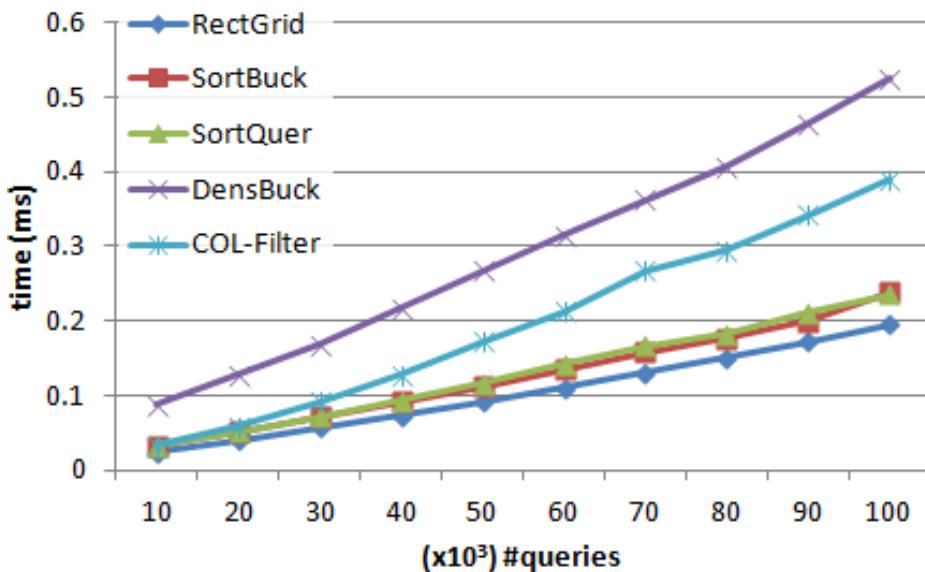


Spatial Indexes for Maintaining Continuous Query Results

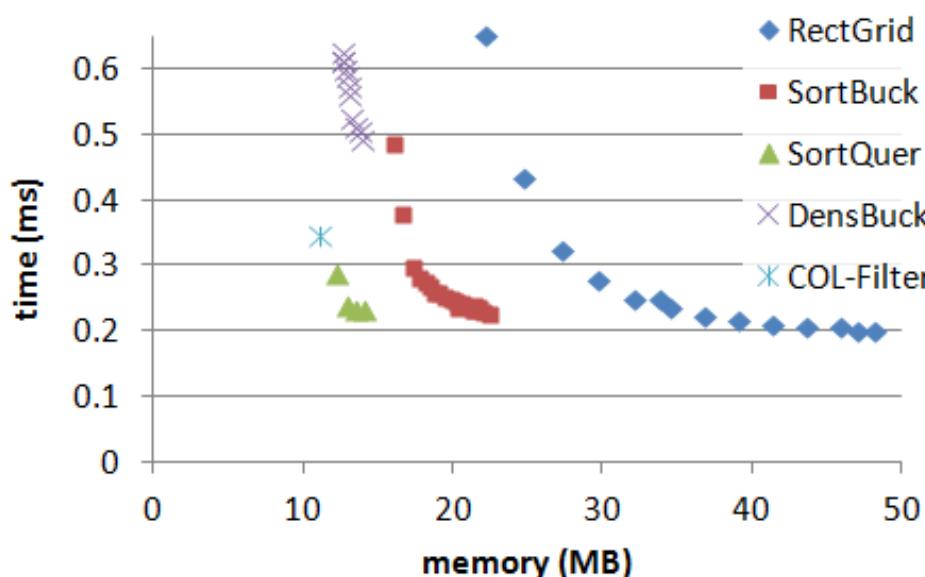
- We consider variations of spatial structures with different space, filtering and update time costs
 - Grids: two-dimensional space is partitioned into **fixed size buckets** of queries (**static** RECTGRID, **dynamic** SORTBUCK)
 - R-Trees: **adjusts bucket widths** to maintain a min/max constant number of queries per bucket (DENSBUCK)
 - Directly maintains a list of queries sorted by their **minimum scores** (SORTQUER)



Index Evaluation



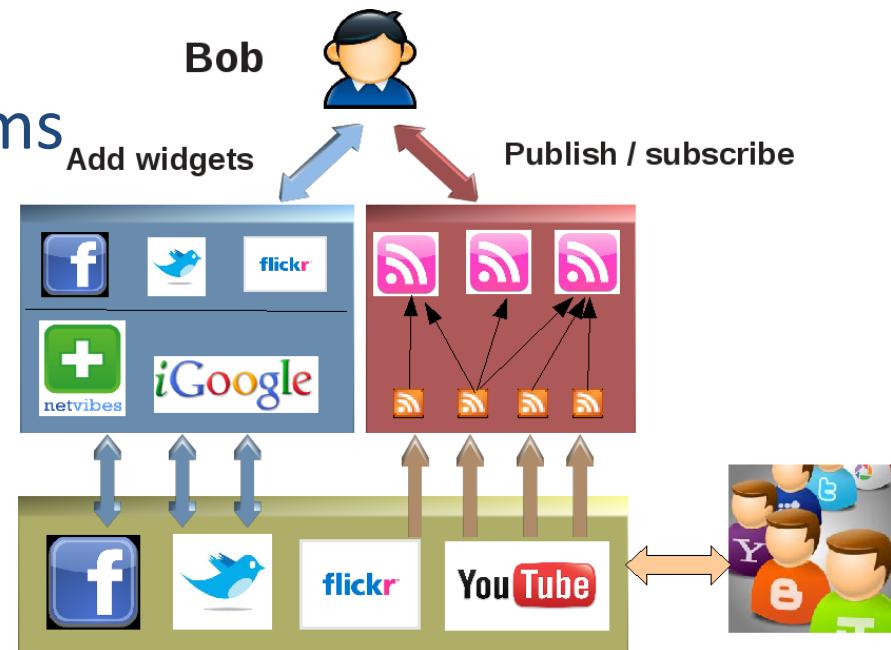
- In all indexes the average time per item to retrieve updated queries scales linearly to the # of queries
- Trade-off between index memory and time (filter + update)



Using 20% more memory in SortQuer, we achieve 55% better performance than state-of-art COL-Filter

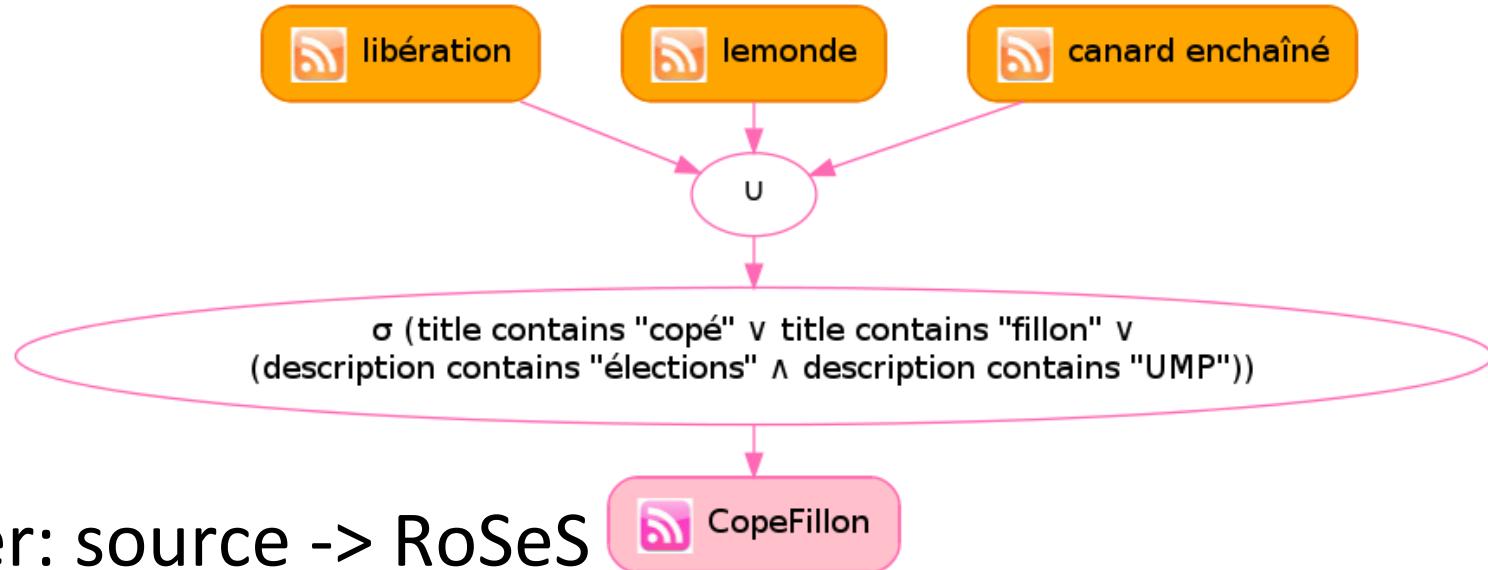
Data-Centric Aggregation of RSS Feeds

- Facilitate creation, aggregation and personalization of RSS feeds using declarative languages
- The RoSeS Data Model:
 - Semi-structured data streams
 - Algebraic operators for querying and merging RSS feeds
 - DB-inspired multi-query optimization techniques for a large-number of active subscriptions



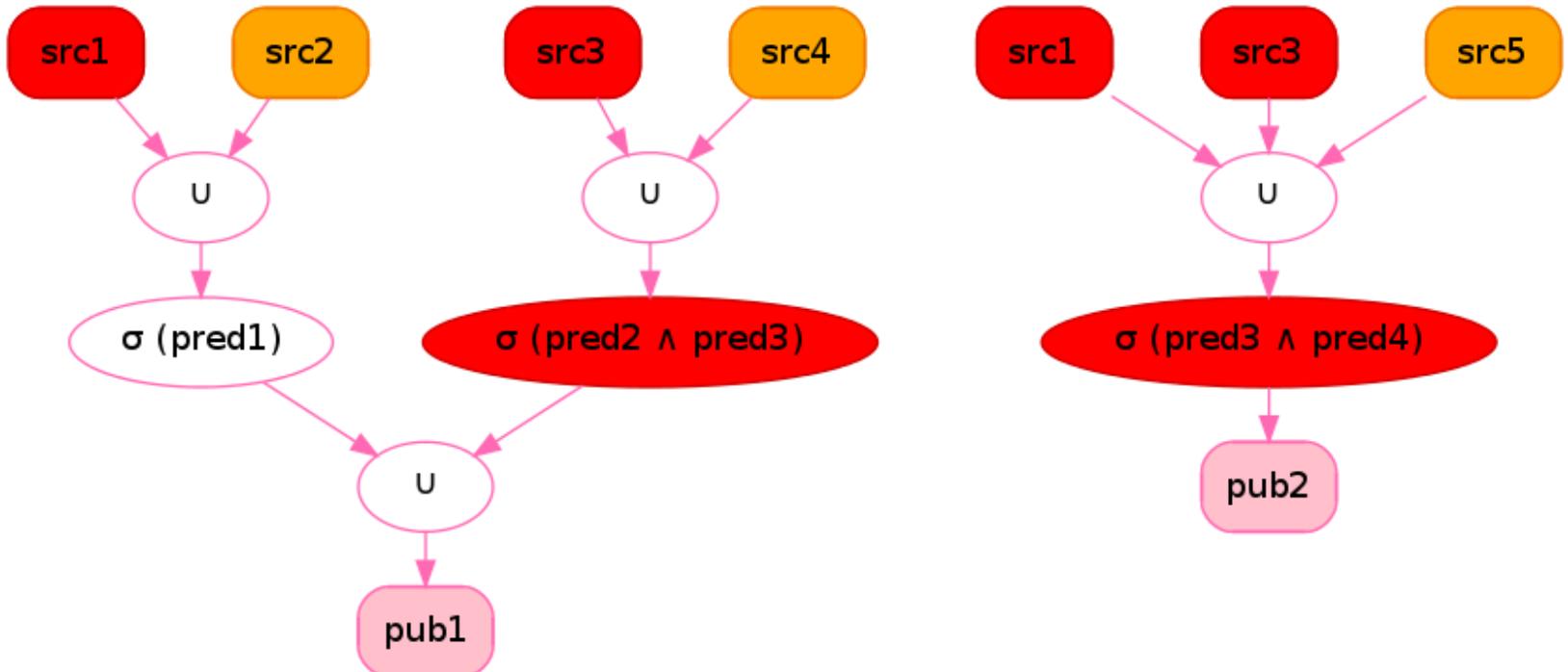
Web Syndication Language in RoSSES

```
create feed CopeFillon  
from liberation | lemonde | canardEnchaine  
where title contains "copé" or title contains "fillon" or  
(description contains "élections" and description contains "UMP");
```



- register: source -> RoSeS
- create: RoSeS -> RoSeS
- subscribe: RoSeS -> target

Optimizing Multiple ROSES Queries



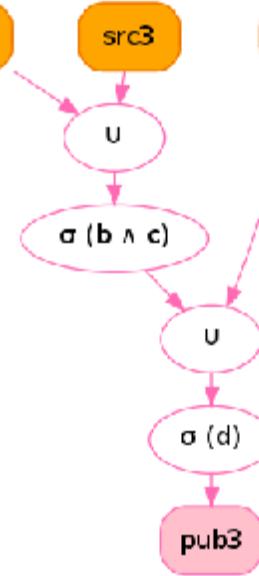
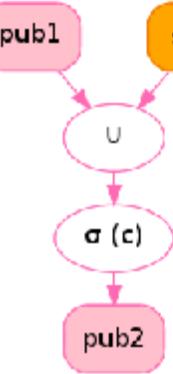
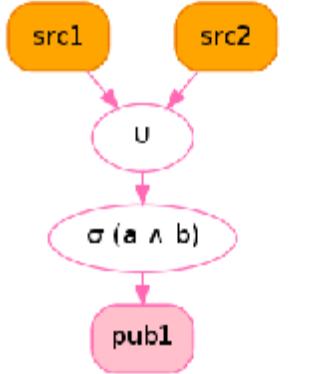
- Long-running global query execution plan using algebraic operators connected by inter-operator queues
- Challenge: Cost-based continuous query optimization
 - Static query rewriting and dynamic plan adaptation

$$cost^{Total}(G) = \sum_{v \in V(G)} (cost^{Read}(v) + cost^{Eval}(v) + cost^{Write}(v))$$

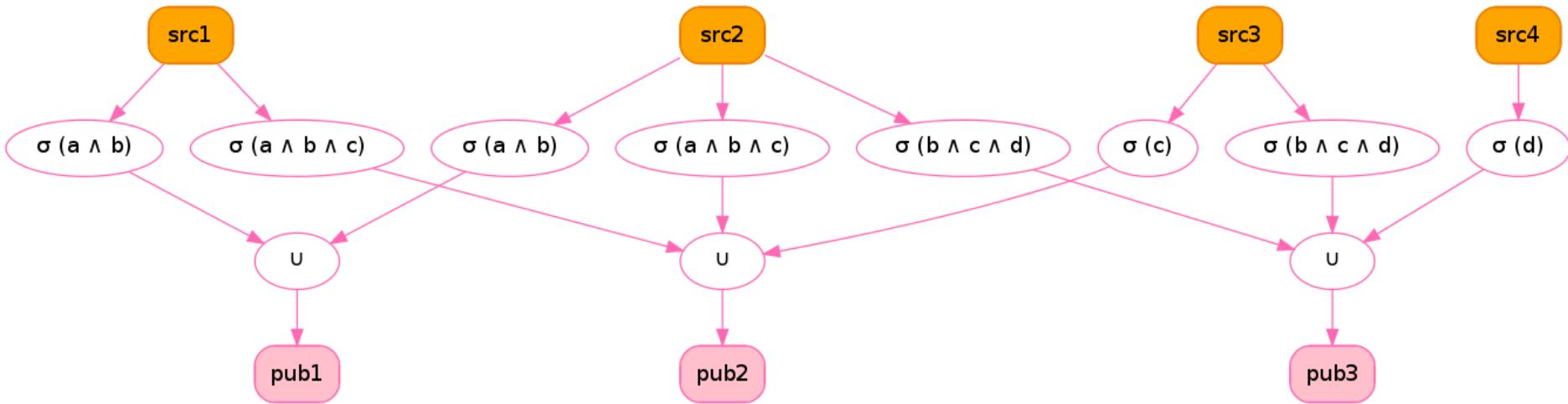
ROSES Cost Model

Vertex v	Read cost $cost^{Read}(v)$	Evaluation cost $cost^{Eval}(v)$	Write cost $cost^{Write}(v)$
s	0	0	0
$\sigma_{pred}(v')$	$rate(v')$	$cost^{Eval}(\text{pred}) \cdot rate(v')$	$selectivity(\text{pred}) \cdot rate(v')$
$v_1 \cup \dots \cup v_n$	$\sum_{i=1}^n rate(v_i)$	0	$\sum_{i=1}^n rate(v_i)$
$\omega_w^{time}(v')$	$rate(v')$	$rate(v')$	$rate(v')$
$\omega_N^{count}(v')$	$rate(v')$	0	$rate(v')$
$v_1 \bowtie_{pred^J} \omega_w^{time}(v_2)$	$rate(v_1) \cdot rate(v_2) \cdot w$	$cost^{Eval}(\text{pred}^J) \cdot cost^{Read}(v)$	$selectivity(\text{pred}^J) \cdot cost^{Read}(v)$
$v_1 \bowtie_{pred^J} \omega_N^{count}(v_2)$	$rate(v_1) \cdot N$	$cost^{Eval}(\text{pred}^J) \cdot cost^{Read}(v)$	$selectivity(\text{pred}^J) \cdot cost^{Read}(v)$
$p(v')$	$rate(v')$	0	0

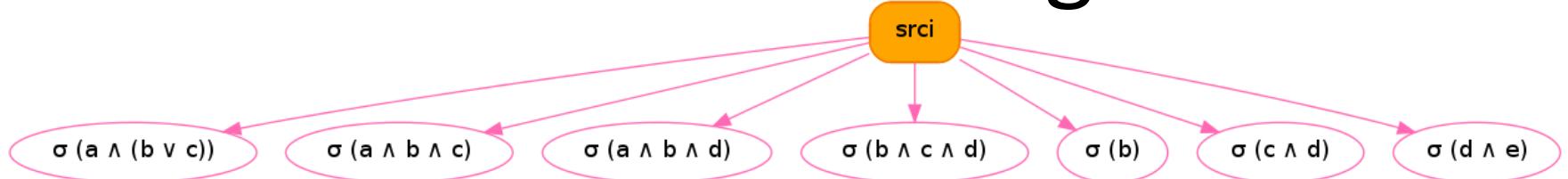
Query Rewriting: Normalization



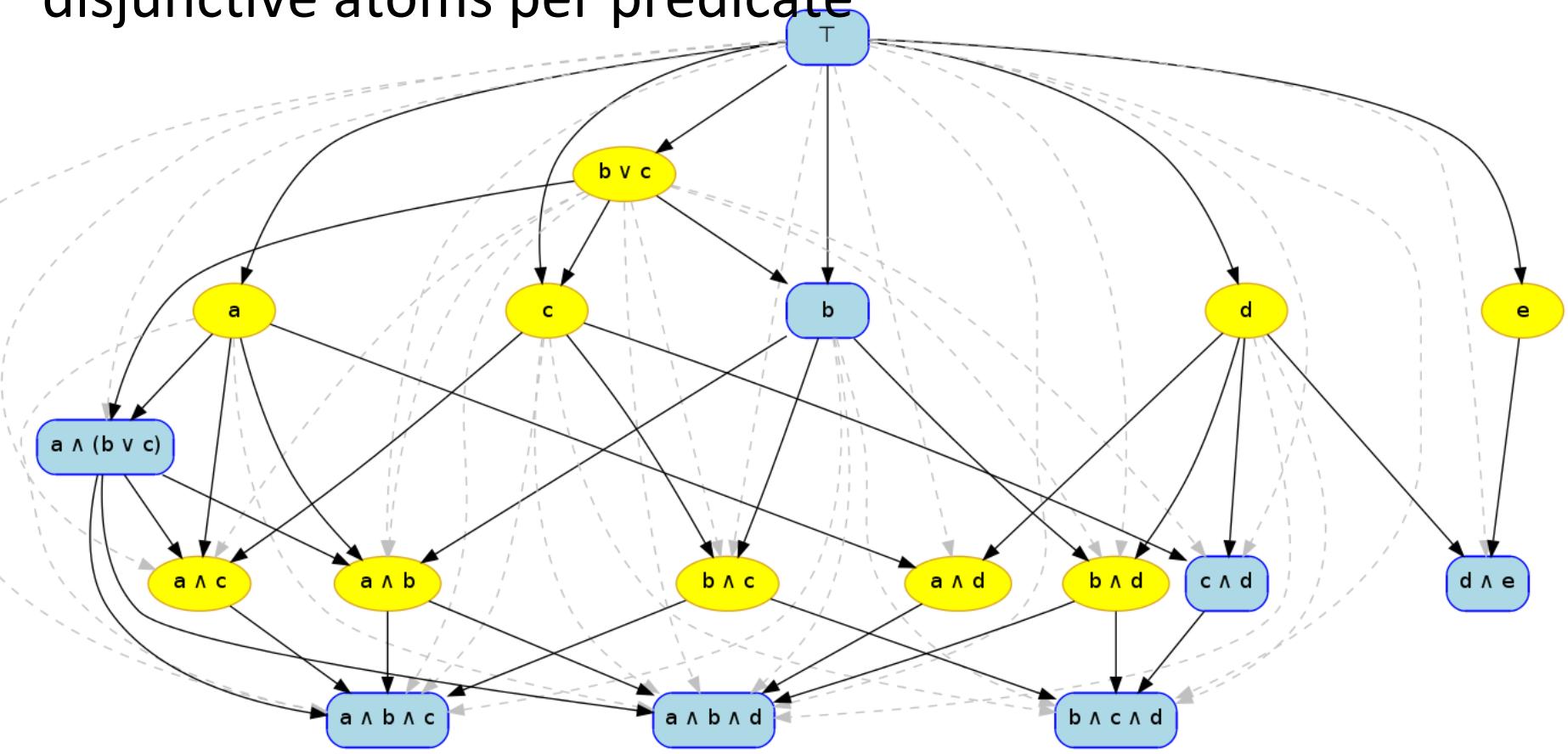
- Push selections to sources
- Unfold published feed definitions
- Group cascading selections
- CNF



Factorization of Filtering Predicates



- $O(m \cdot 2^n)$ where m : # filtering predicates and n : # disjunctive atoms per predicate

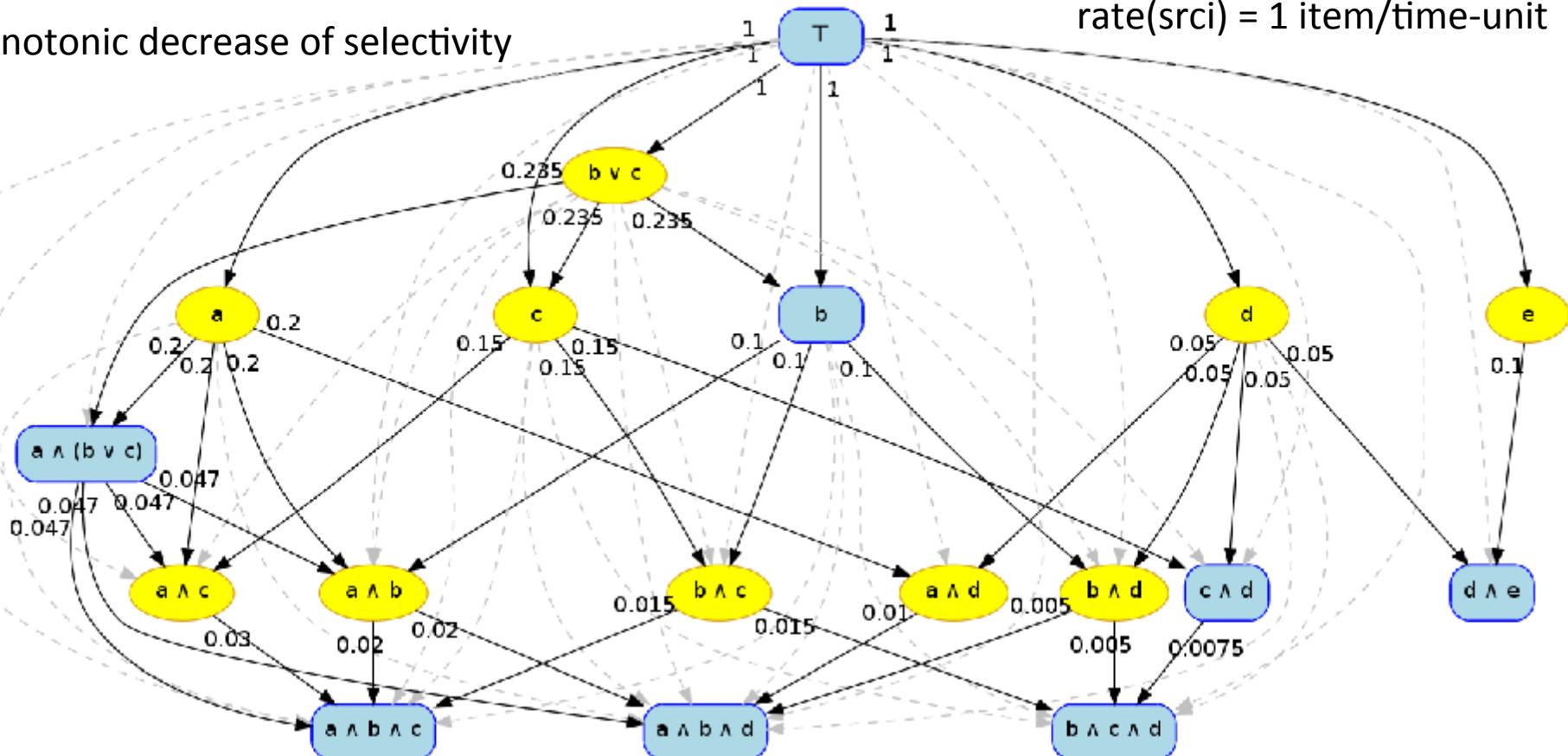


Selectivity of Subsumed Predicates

- $\text{cost}(x \rightarrow y) = \text{sel}(x, \text{srci}) \cdot \text{rate}(\text{srci})$
 - $\text{sel}(x, \text{srci})$: term occurrence frequency in source srci
 - $\text{sel}(x \wedge y) = \text{sel}(x) * \text{sel}(y)$
 - $\text{sel}(x \vee y) = \text{sel}(x) + \text{sel}(y) - \text{sel}(x) * \text{sel}(y)$

Monotonic decrease of selectivity

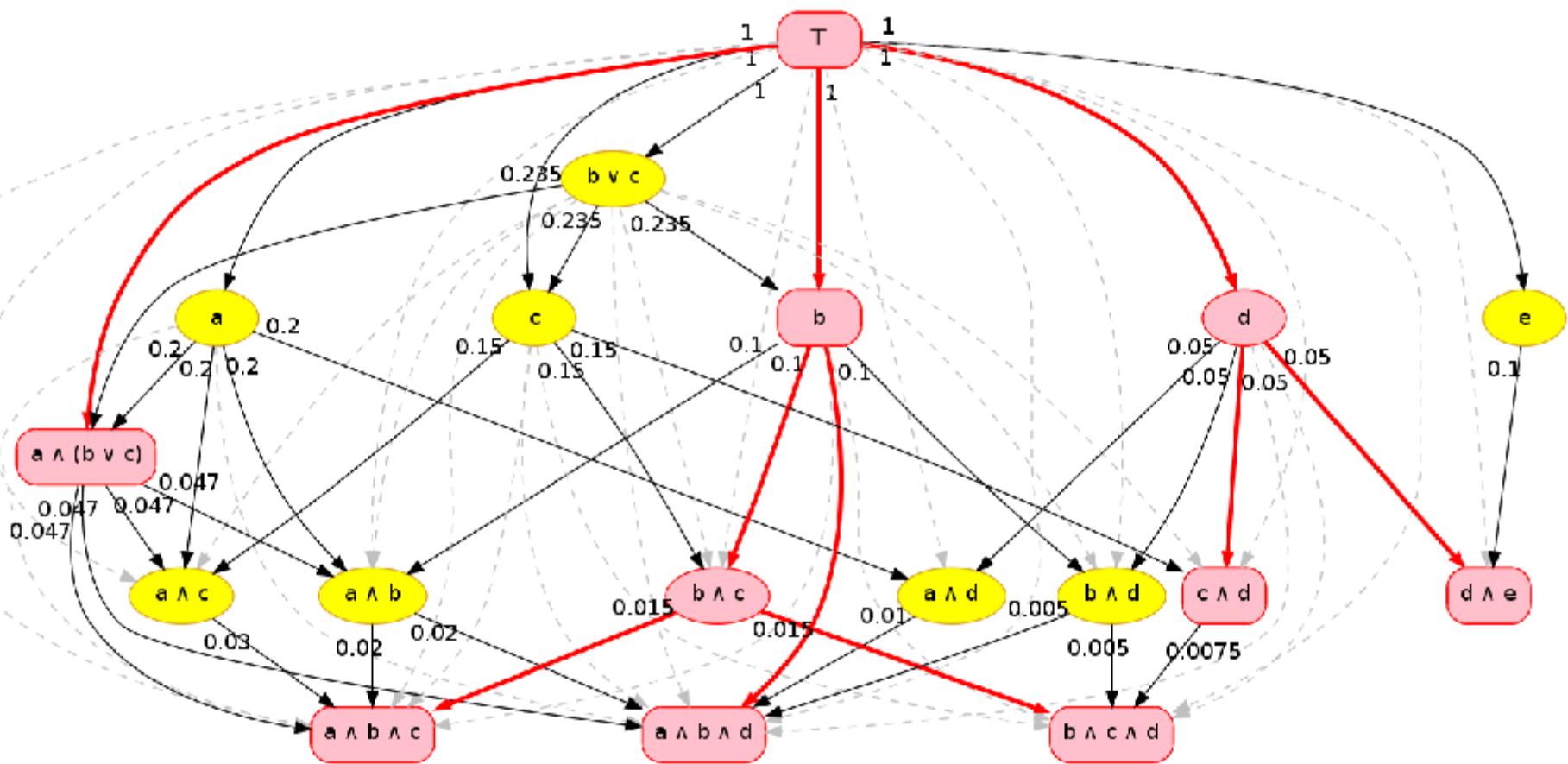
$\text{rate}(\text{srci}) = 1 \text{ item/time-unit}$



Steiner Minimum-cost tree (SMT)

- Given a directed weighed graph $G(V,E,c)$, where
 - V is the set of vertices capturing filtering predicates,
 - E is the set of directed edges capturing subsumption relationships among them with a cost $c: E \rightarrow [0..1]$, and
 - a subset of vertices $T \subseteq V$ called terminals (i.e. the root feed r and its filtering predicates)
- RoSeS multi-selection optimization is reduced to the problem of finding a **Steiner Minimum-cost tree (SMT)** rooted at r which spans all vertices T

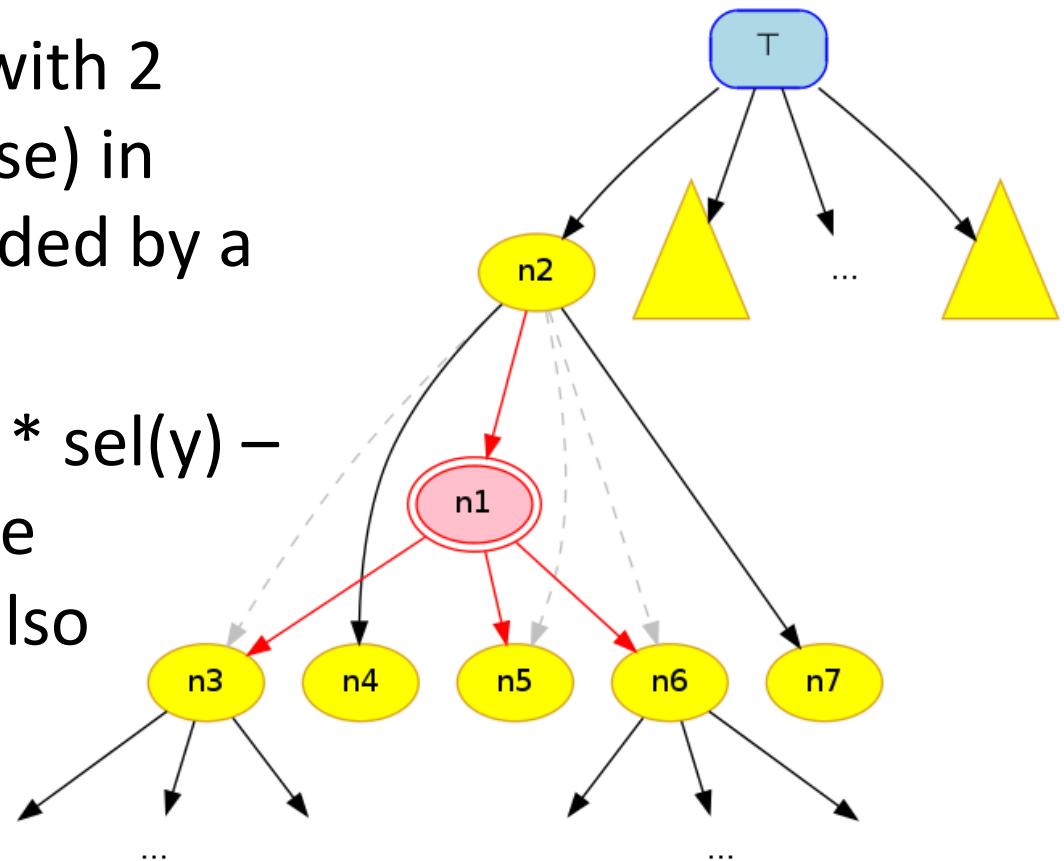
SMT Example



- NP-complete problem (approximation algorithms)

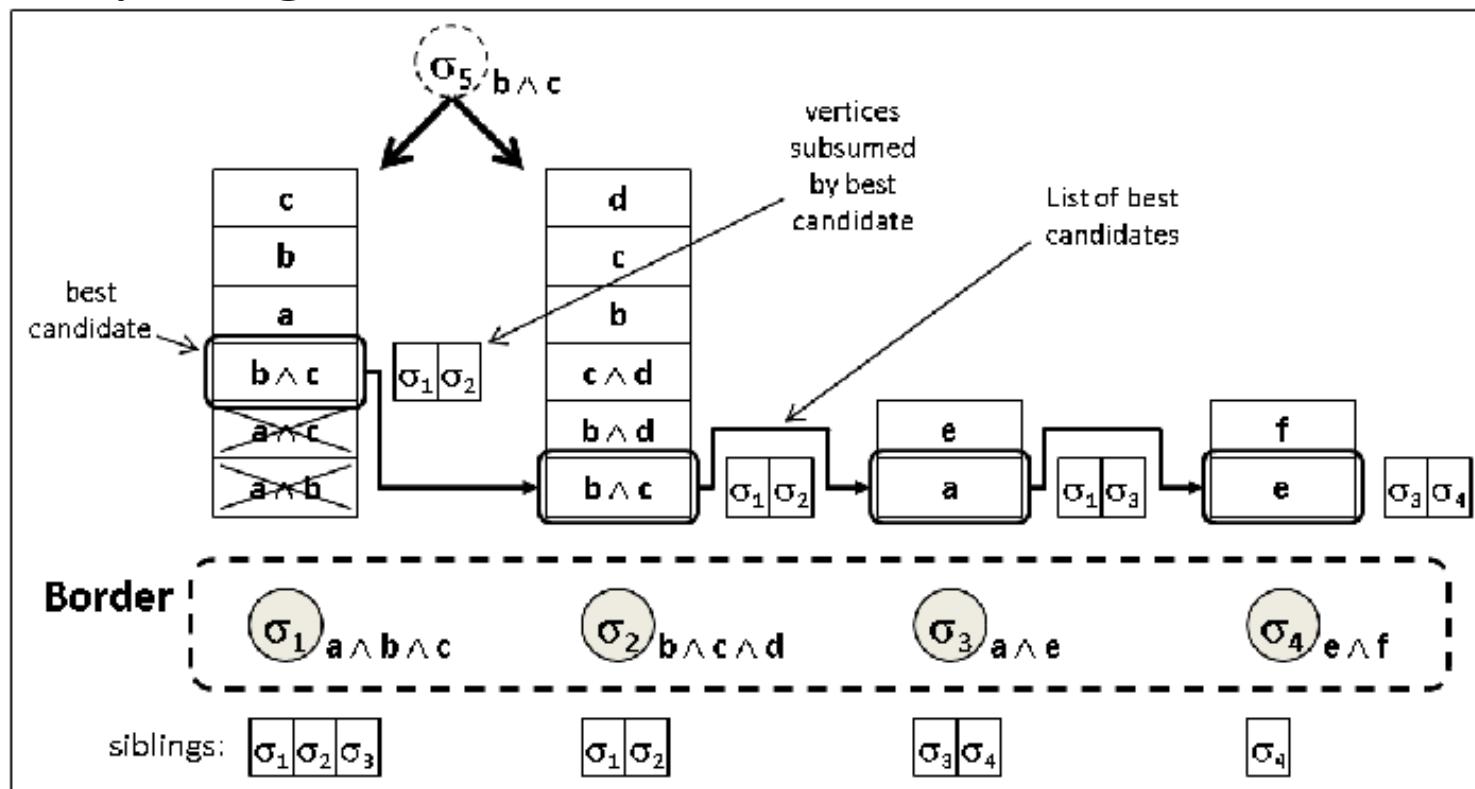
SMT Approximation for Optimal Predicate Factorization [CACTV11]

- **VCA (Very Clever Algorithm)**
 - Bottom-up algorithm with 2 phases (expand-collapse) in each iteration step guided by a cost benefit heuristic
 - $\text{benefice}(x, y) = (n - 1) * \text{sel}(y) - n * \text{sel}(x)$ where n is the number of y children also subsumed by x



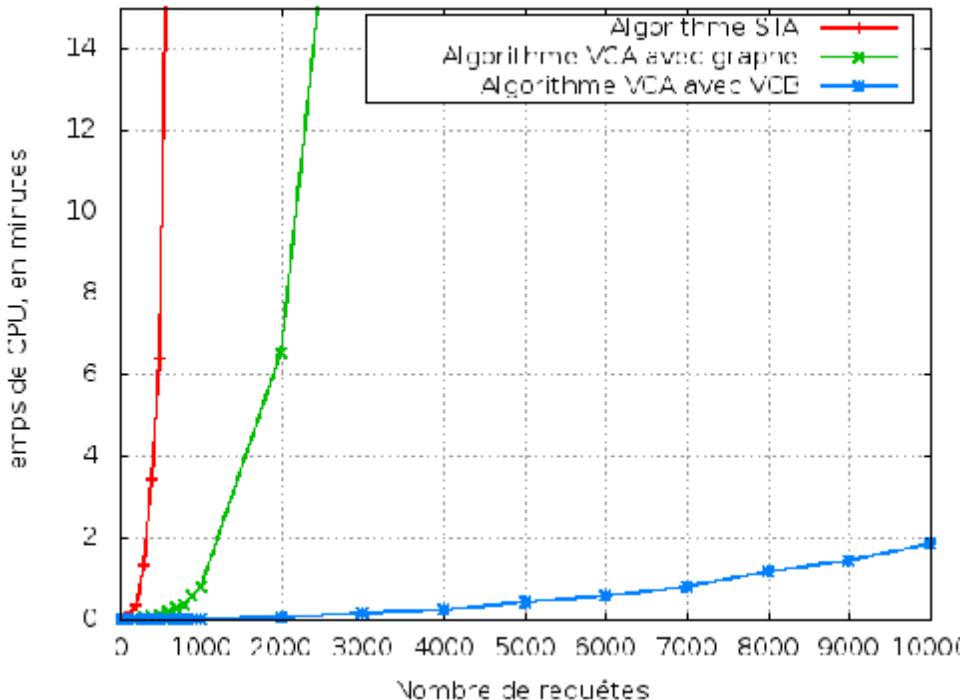
Very Clever Border

- Generate in advance for each border node the list of its subsuming candidate predicates
 - Order the list by the selectivity of the corresponding candidates and continuously update it without re-computing the set of candidates at each iteration step

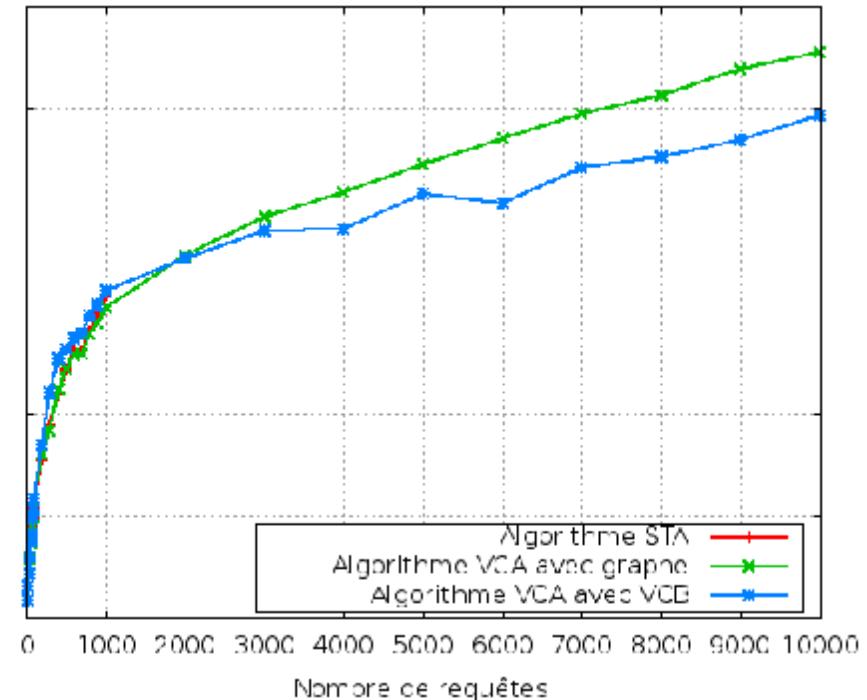


Experimental Evaluation

- Workload of 10000 conjunctive queries
 - over one source with up to 3 atomic conjuncts



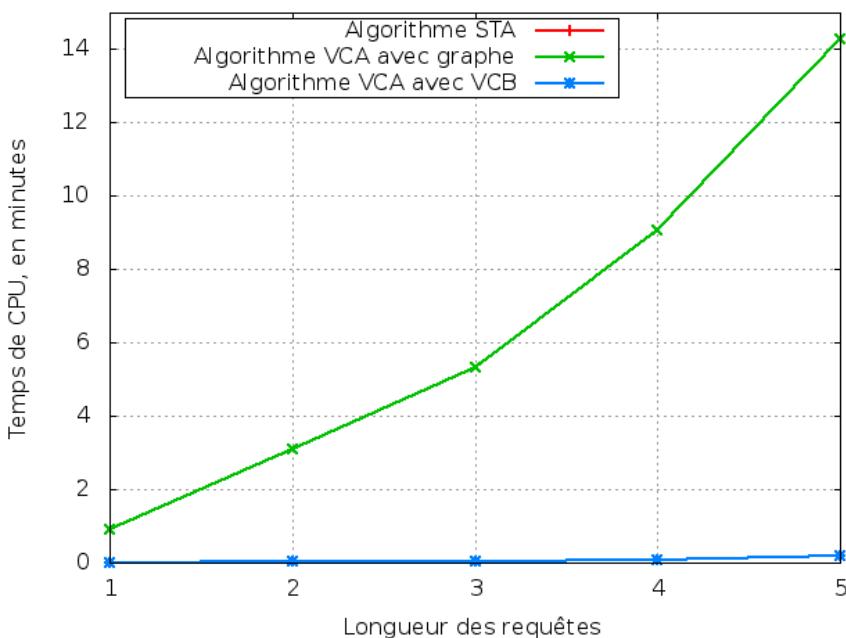
VCA algorithm + VCB scales well and is able to optimize 10000 queries in two minutes



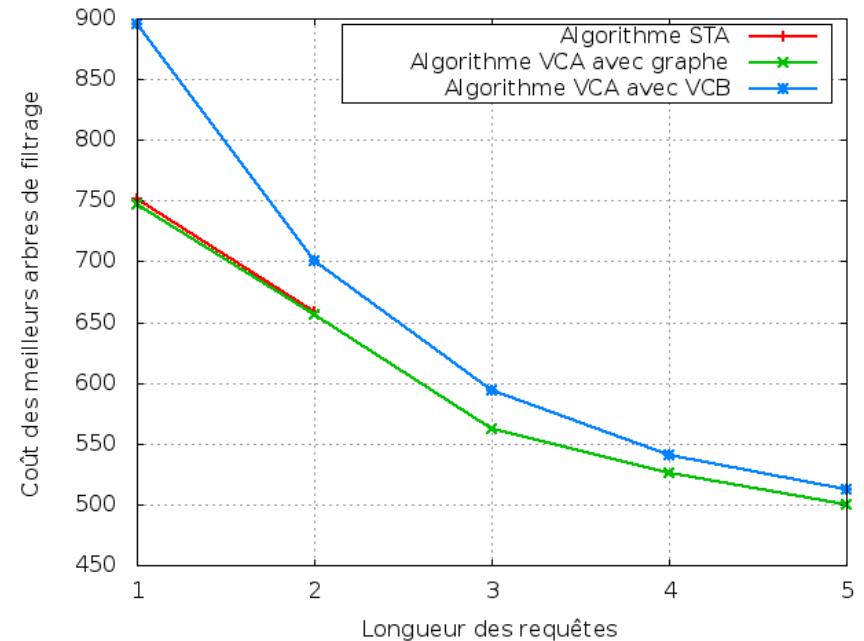
filter plans generated by all three algorithms have a similar evaluation cost

Experimental Evaluation

- Workload of 1000 disjunctive queries of fixed length
 - five sets of queries with an increasing number of atomic disjuncts from 1 to 5



VCA algorithm + VCB scales well with the length of queries



Increasing number of disjuncts favour factorization and reduces the obtained tree *evaluation cost*

Social Text Stream Analytics

- Emerging Trends Monitoring:
 - look for words or phrases of high co-occurrence
- Real World Events Detection:
 - Look for set of tweets exhibiting statistically important similarity in content, metadata or the involved user-community
- Usually formulated as a clustering problem:
 - Several machine learning (ML) algorithms have been proposed e.g. k-means, TStream

Temporal Aspects of Social Streams

- Understanding how the **number** and the **entropy** of the detected clusters is affected by changes in **core properties** such as
 - **Centroid** (represent the geometric center of clusters summarizing their *content* and *vocabulary*)
 - **Shape** (captures a notion of *points distance from the cluster centroid* and highlights *its structure*)
 - **Density** (reveals the *granularity* and *tensions* of user conversations in tweets)
- Compared to the #tags under which tweets are initially posted the detected clusters are expected to capture **fine-grained subtopics** of the conversation conducted in a social stream

Acknowledgements

- Nelly Vouzoukidou (Ms UoC, Ph.D LIP6)
- Zeinab Hmedeh (Ph.D CNAM)
- Harris Kourdounakis (Ms UoC)
- Sofia Kleisarxaki (Ms UoC, Ph.D UoC)
- Jordi Creus Tomas (Ph.D LIP6)

References

- [VAC12] N. Vouzoukidou, B. Amann, V. Christophides. Processing Continuous Text Queries Featuring Non-Homogeneous Scoring Functions. In Proc. of the ACM Conference on Information and Knowledge Management (CIKM), Maui Hawaii 2012.
- [HKCMS12] Z. Hmedeh, H. Kourdounakis, V. Christophides, C. du Mouza, M. Scholl, N. Travers. "Subscription Indexes for Web Syndication Systems". In Proc. of the 15th International Conference on Extending Database Technology (EDBT), March 26-30, 2012 Berlin, Germany.
- [HVTcms11] Z. Hmedeh, N. Vouzoukidou, N. Travers, V. Christophides, C. du Mouza, Michel Scholl, "Characterizing Web Syndication Behavior and Content". In Proc. of the 12th Inter. Conf. on Web Information System Engineering (WISE), October 13 – 14, 2011, Sydney, Australia.
- [CACTV11] J. Creus, B. Amann, V. Christophides, N. Travers, and Dan Vodislav, Optimizing large collections of continuous content-based RSS aggregation queries. 27èmes Journées Bases de Données Avancées (BDA), 24-27 Octobre 2011, Rabat, Maroc.

In memory of Michel Scholl



<http://cedric.cnam.fr/~scholl/hommage>