LSH Ensemble: Internet-Scale Domain Search

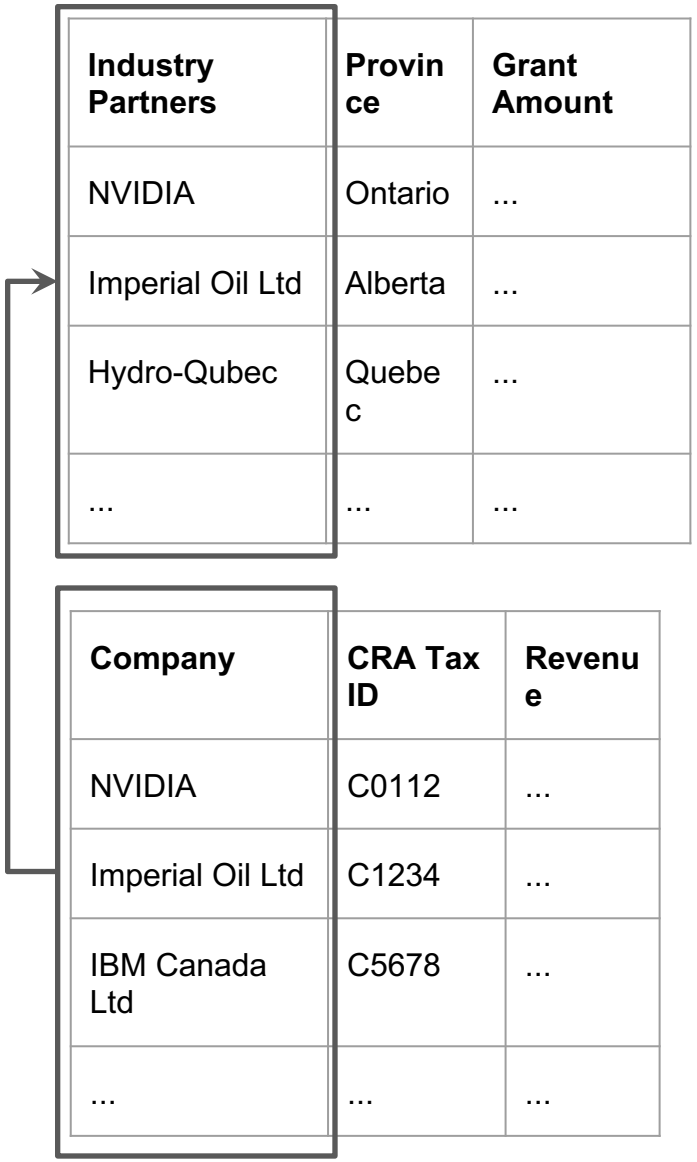
Erkang (Eric) Zhu[+], Fatemah Nargesian[+], Ken Q. Pu[*], Renée J. Miller[+]

+ University of Toronto * University of Ontario Institute of Technology

Alice, a data scientist, wishes to understand the factors affecting companies' funding to the universities.

Scenario 1: she has the two tables at hand.

Solution 1: join the two tables on the Industry Partners and Company columns.



Industry Partners	Province	Grant Amount
NVIDIA	Ontario	...
Imperial Oil Ltd	Alberta	...
Hydro-Qubec	Quebec	...
...	...	...

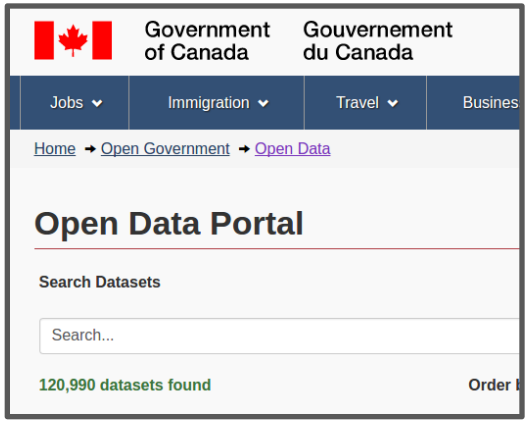
Company	CRA Tax ID	Revenue
NVIDIA	C0112	...
Imperial Oil Ltd	C1234	...
IBM Canada Ltd	C5678	...
...	...	...

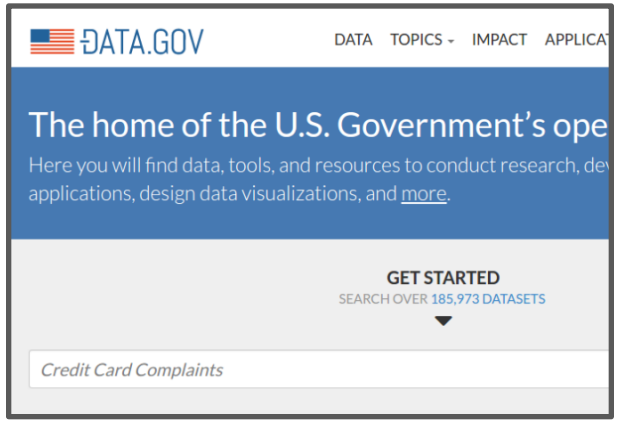
Scenario 2: she has only one table, and a **repository** of MANY tables - manual inspection is not preferred.

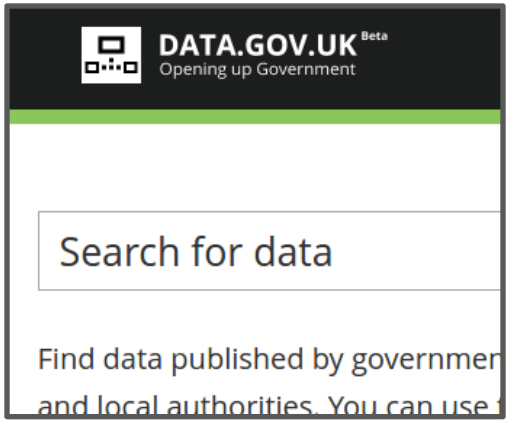
Industry Partners	**Province**	**Grant Amount**
NVIDIA	Ontario	...
Imperial Oil Ltd	Alberta	...
Hydro-Qubec	Quebec	...
...	...	...

Solution 2: requires a search engine for relevant datasets.

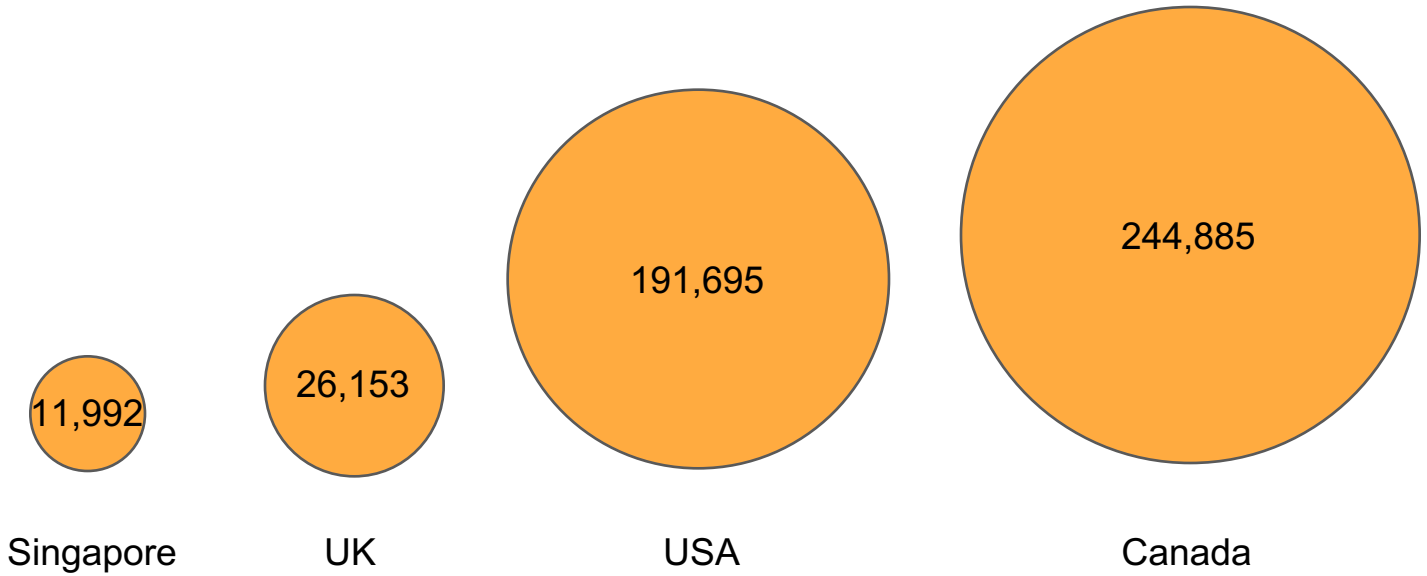
An Internet of Data





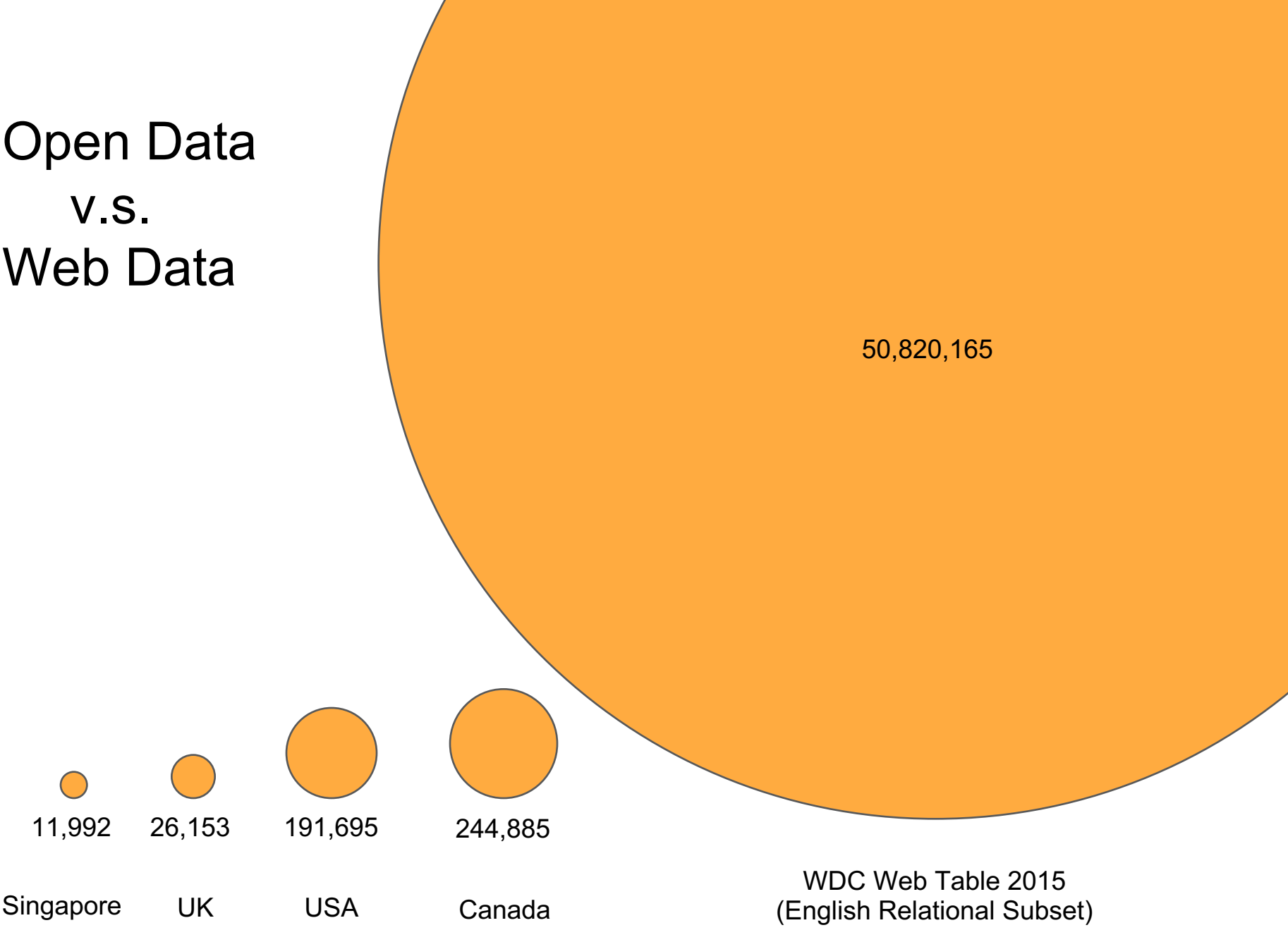


Examples of Open Data (Early 2016)

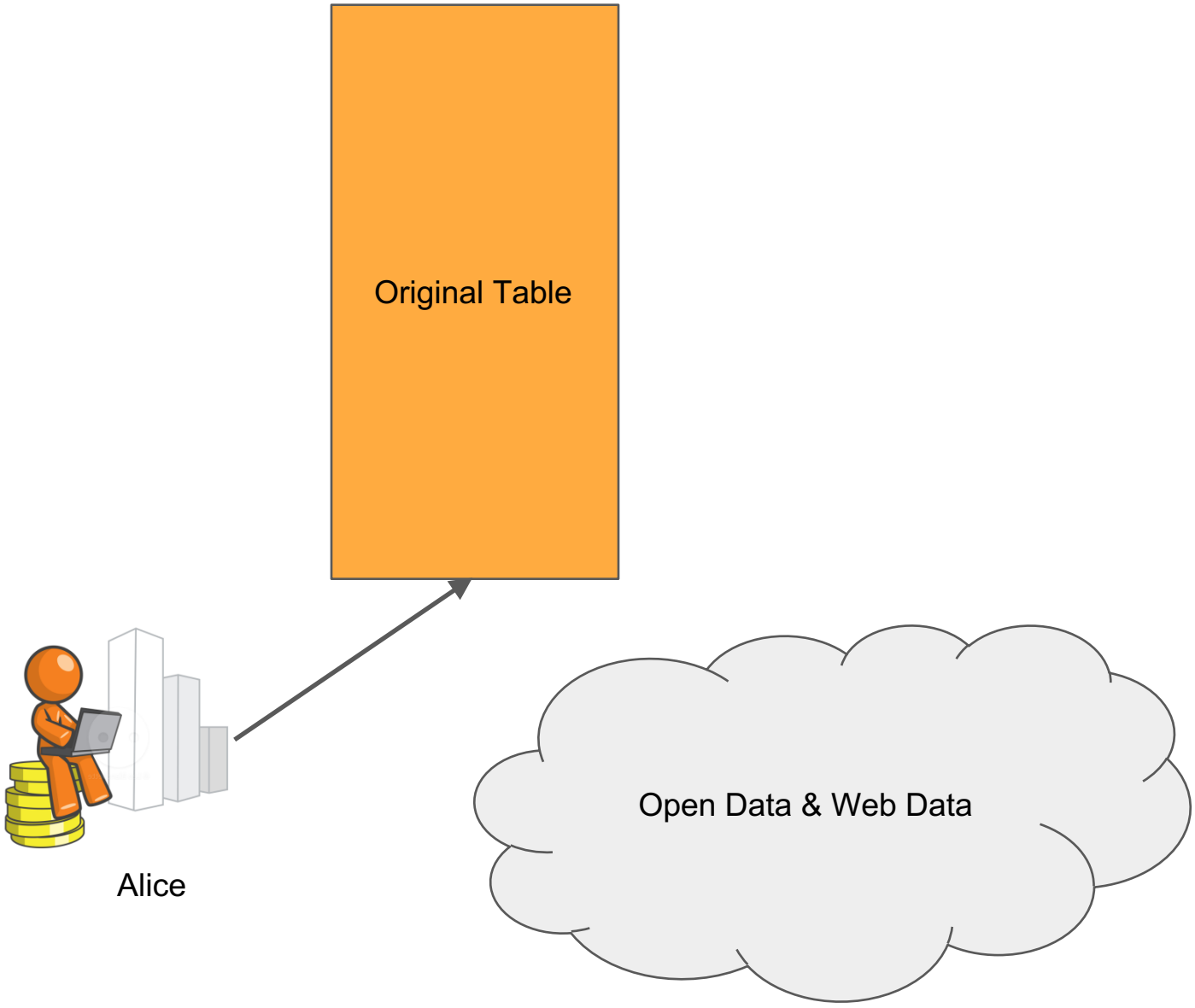


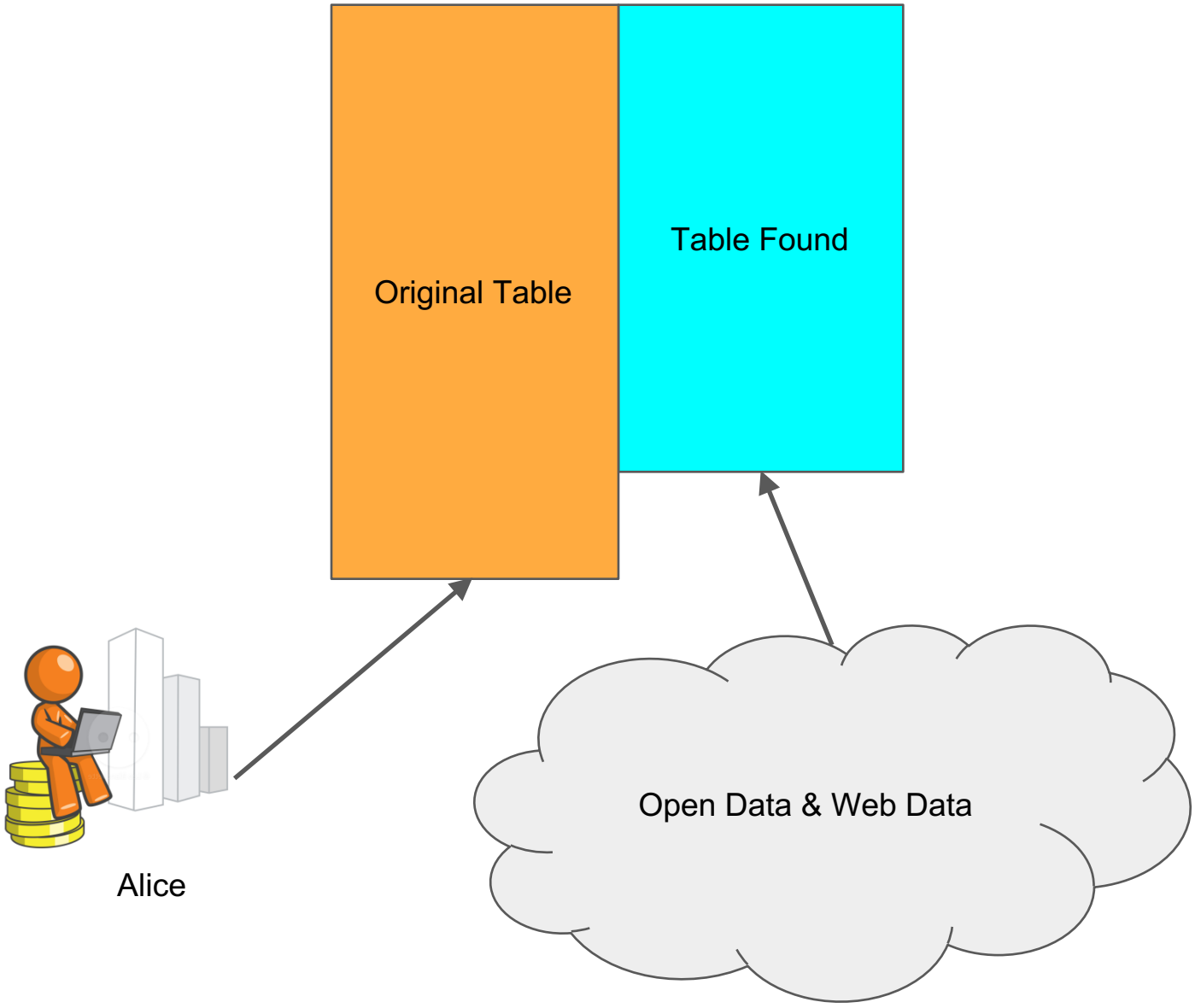
Number of Datasets by Country

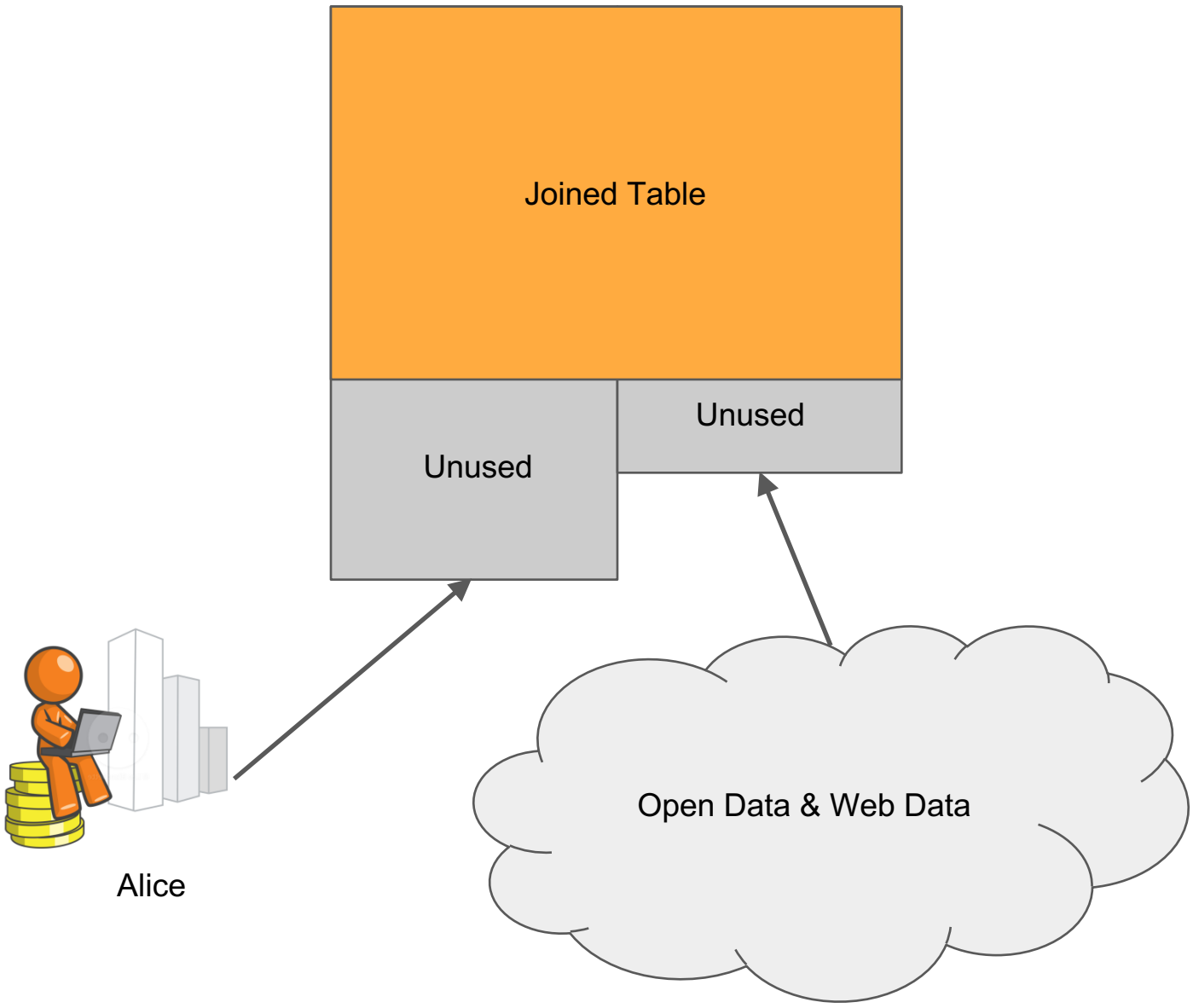
Open Data v.s. Web Data



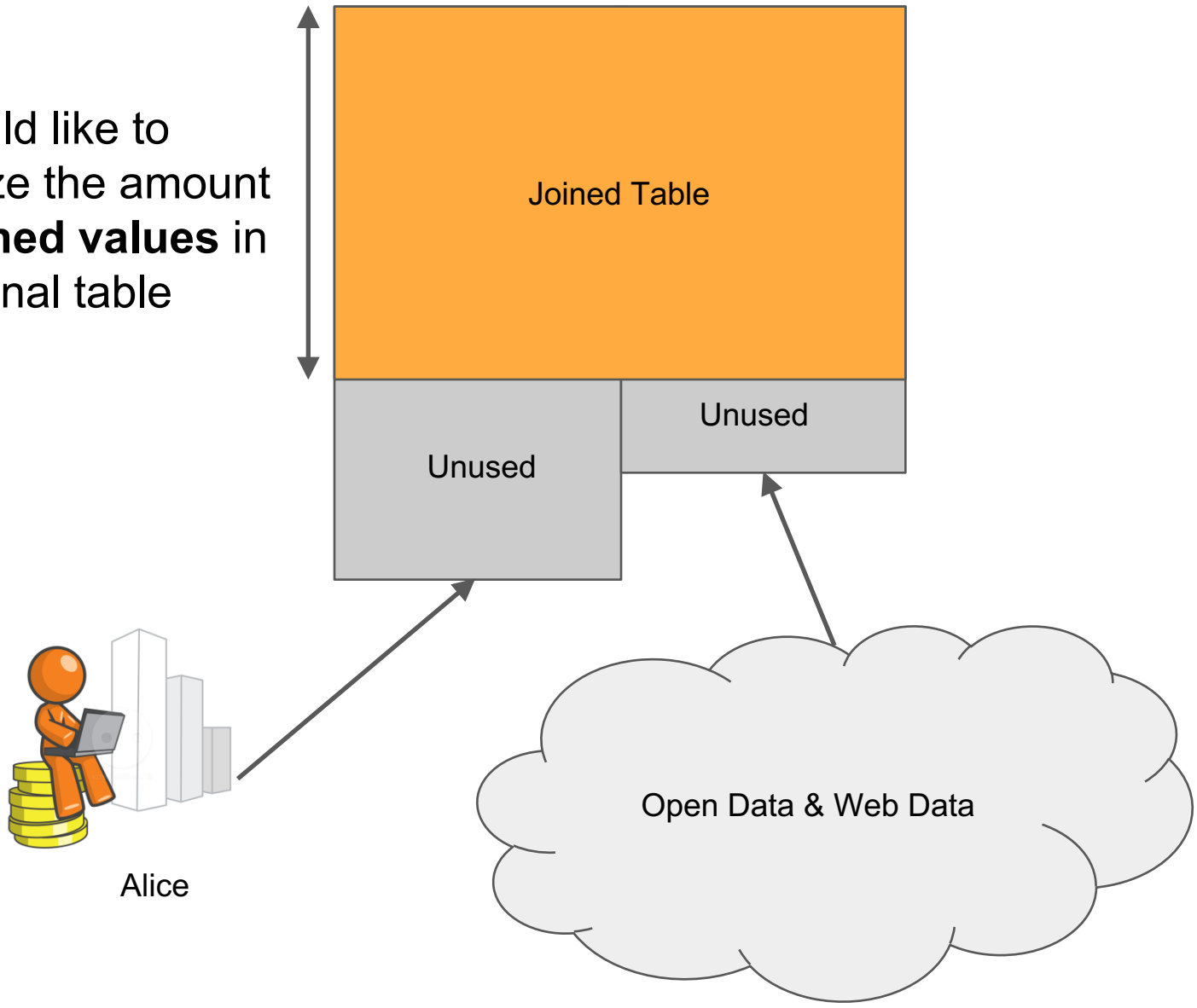
What is a reasonable objective for searching datasets?


Original Table
Alice
Open Data & Web Data

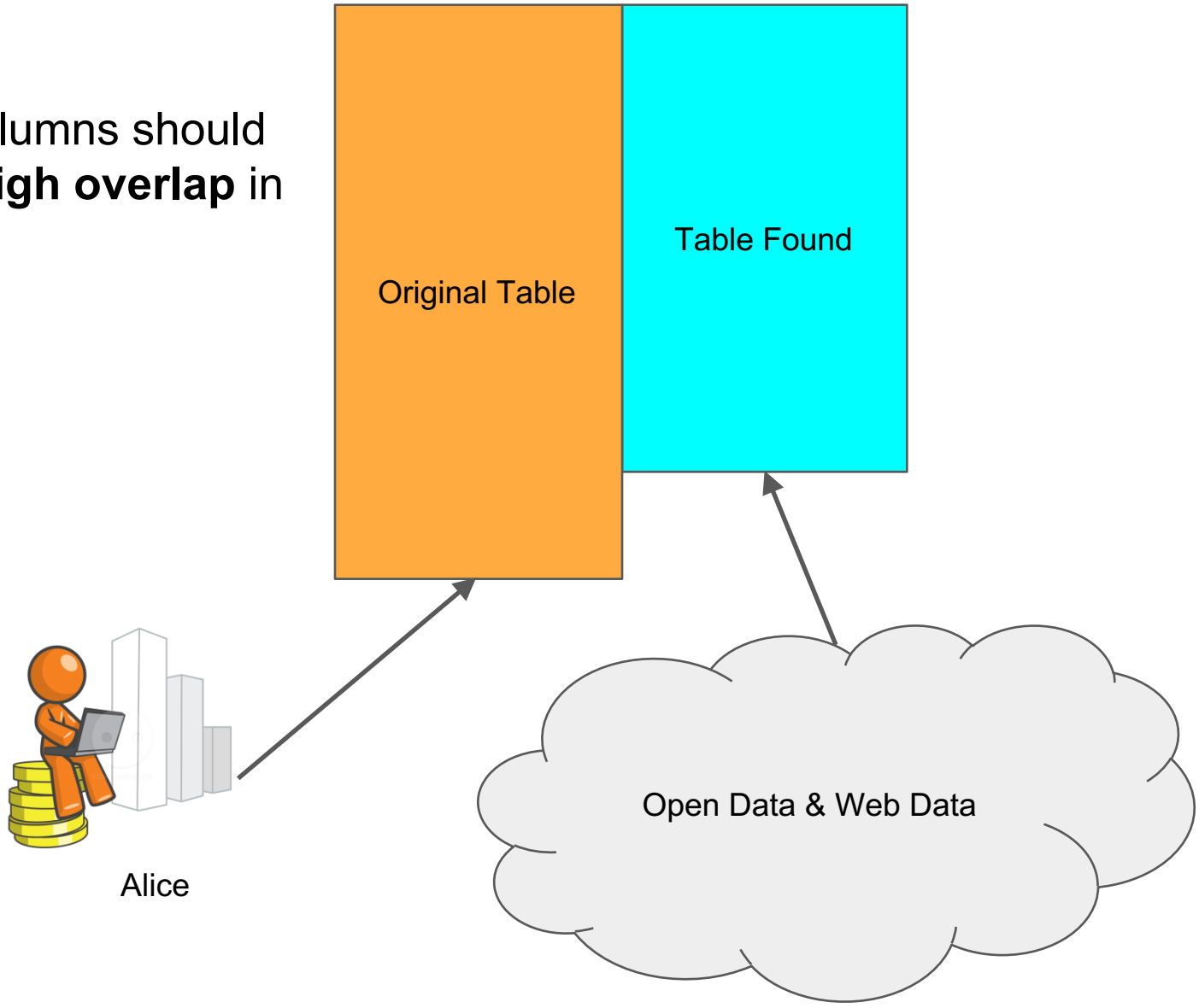

Original Table
Table Found
Open Data & Web Data
Alice


Joined Table
Unused
Unused
Alice
Open Data & Web Data

We would like to maximize the amount of **retained values** in the original table

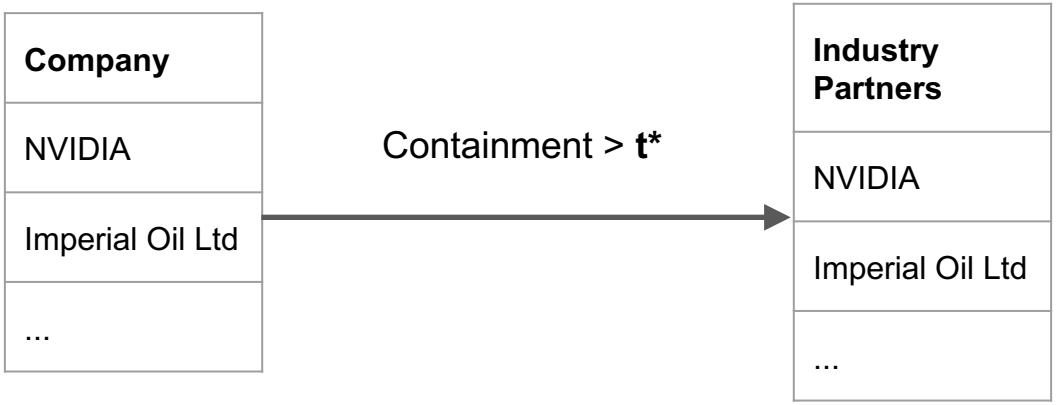


Join columns should have **high overlap** in values



Domain Search

Domain: a set of values in a dataset (e.g. a column in a table)



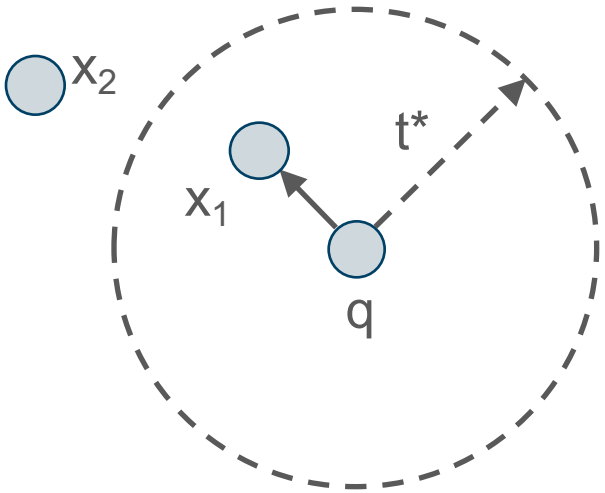
Domain Search: given a query domain Q and **threshold t***, find all domains **X** such that **Containment(Q, X)** is greater than **t***.

$$\text{Containment}(Q, X) = \frac{|Q \cap X|}{|Q|}$$

Approximate Domain Search

We borrow an insight from Approximate Nearest-Neighbour Search.

P(x1) > P(x2)



Existing Solution

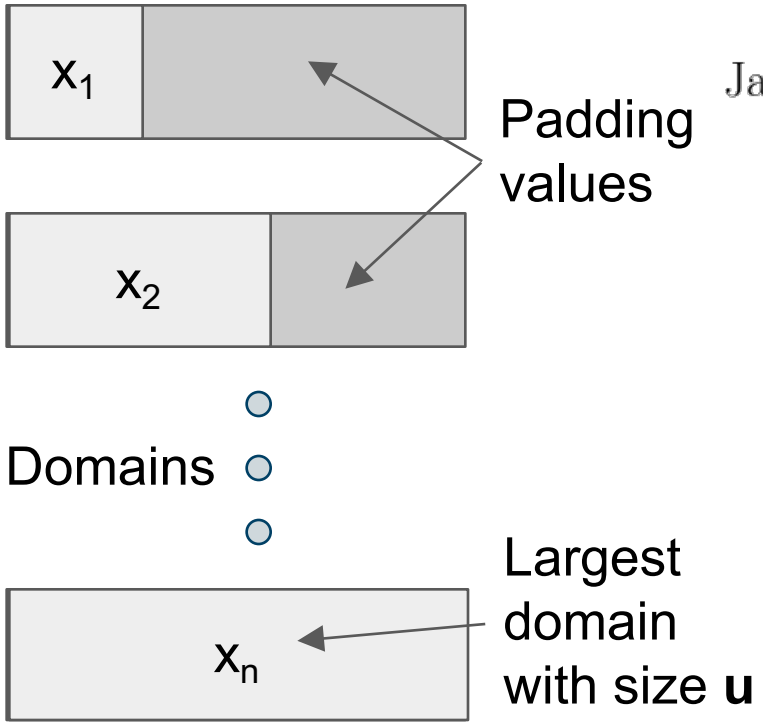
MinHash LSH [Broder 97, Indyk 98] is a search index for Jaccard.

$$\text{Jaccard}(Q, X) = \frac{|Q \cap X|}{|Q \cup X|}$$

- Can be tuned for Jaccard threshold
- Constant space requirement for all domain sizes
- Biased against large domains

Existing Solution

Asymmetric MinHash LSH is a search index for containment [Shrivastava 15].



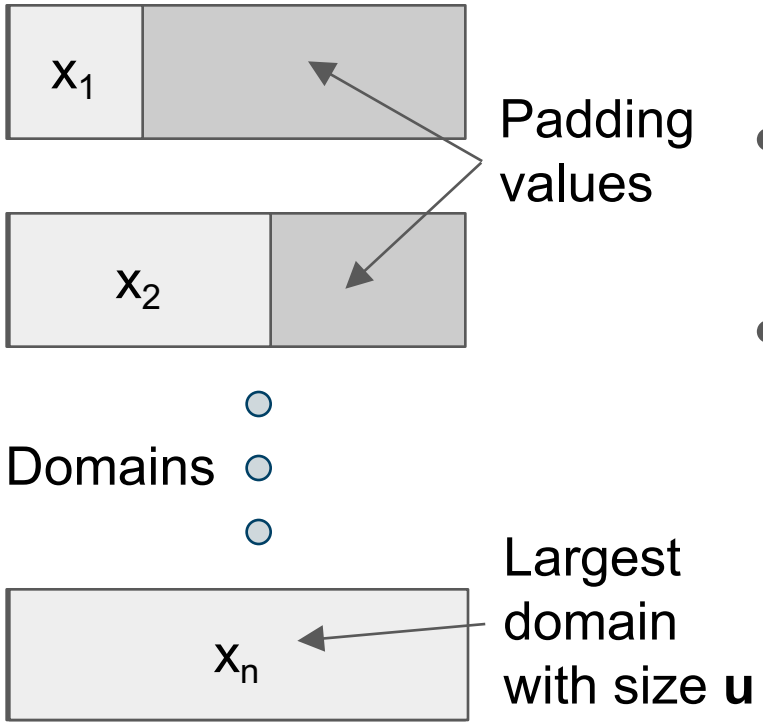
$$\text{Jaccard}(Q, X_{padded}) = \frac{\text{Containment}(Q, X)}{\frac{u}{|Q|} + 1 - \text{Containment}(Q, X)}$$

$$\text{Containment}(Q, X) \propto \text{Jaccard}(Q, X_{padded})$$

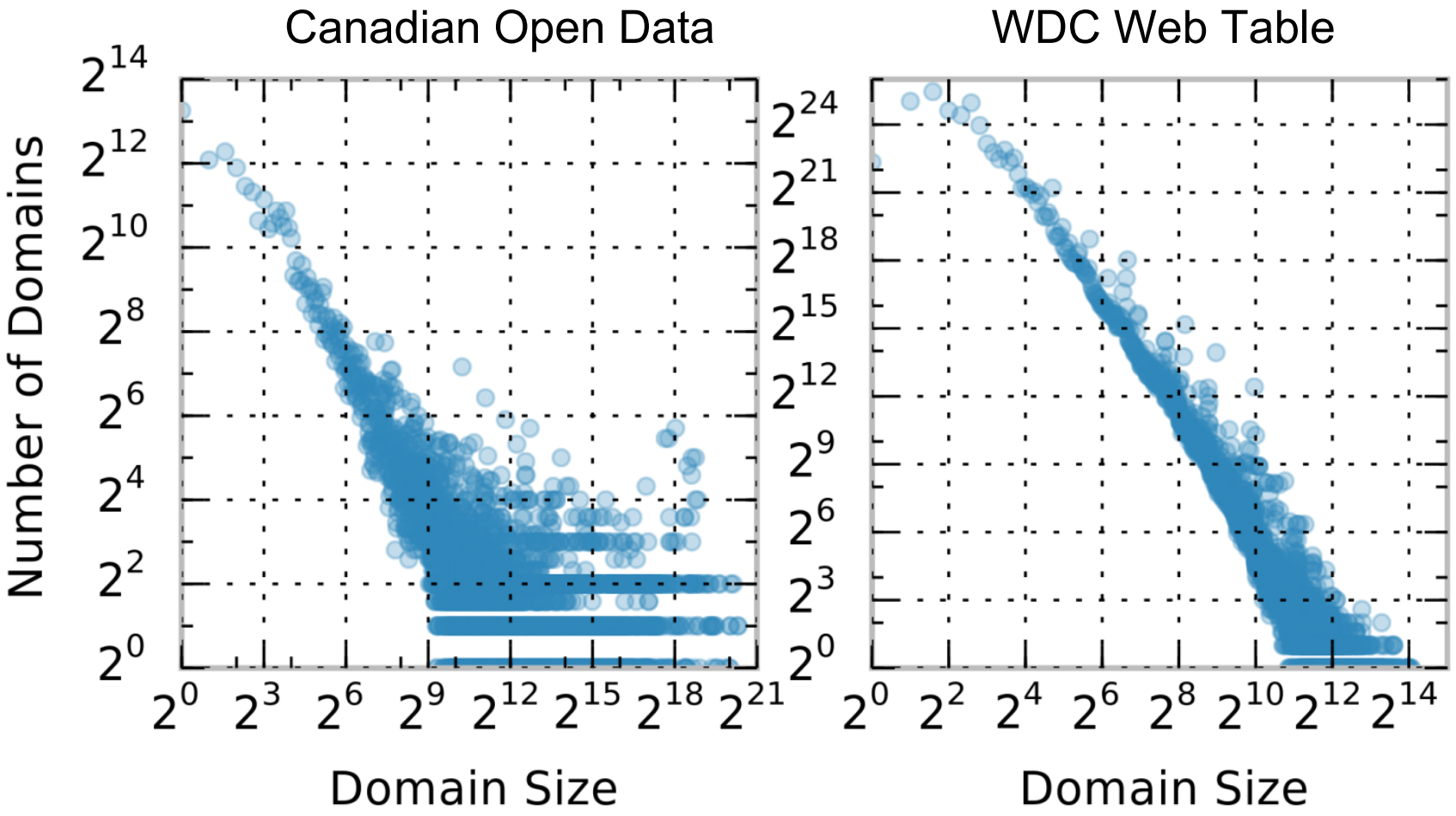
MinHash LSH is used to index the padded domains

Existing Solution

Asymmetric MinHash LSH is a search index for containment [Shrivastava 15].



- Difficult to tune for containment threshold
- Padding reduces MinHash accuracy, especially when domain sizes have a skewed distribution

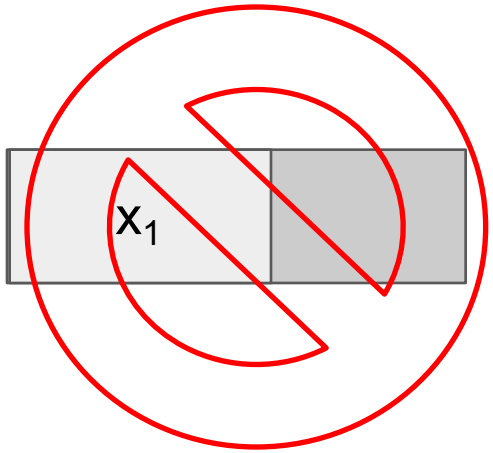

Canadian Open Data
WDC Web Table
Number of Domains
Domain Size
Domain Size
2^{14}
2^{12}
2^{10}
2^{8}
2^{6}
2^{4}
2^{2}
2^{0}
2^{24}
2^{21}
2^{18}
2^{15}
2^{12}
2^{9}
2^{6}
2^{3}
2^{0}
2^{0} 2^{3} 2^{6} 2^{9} 2^{12} 2^{15} 2^{18} 2^{21}
2^{0} 2^{2} 2^{4} 2^{6} 2^{8} 2^{10} 2^{12} 2^{14}

***Research Gap**: we need an index for containment search that maintains high accuracy on **skewed** domain size distribution and is tunable for containment **threshold**.*

LSH Ensemble

(Our Contribution)

Can we use MinHash LSH for containment search without padding?

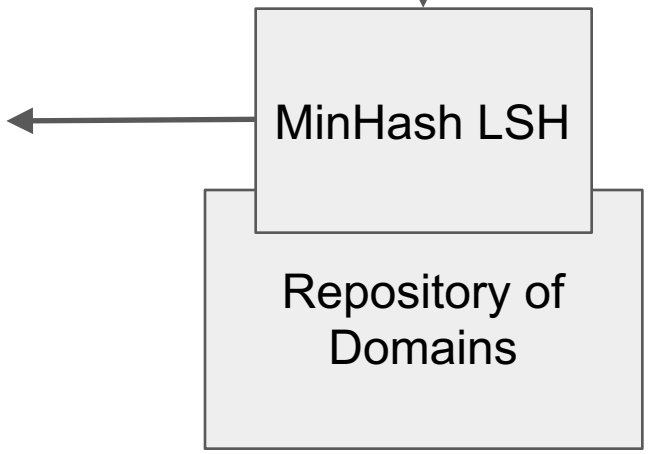


$$\text{Jaccard}(Q, X) = \frac{\text{Containment}(Q, X)}{\frac{|X|}{|Q|} + 1 - \text{Containment}(Q, X)}$$

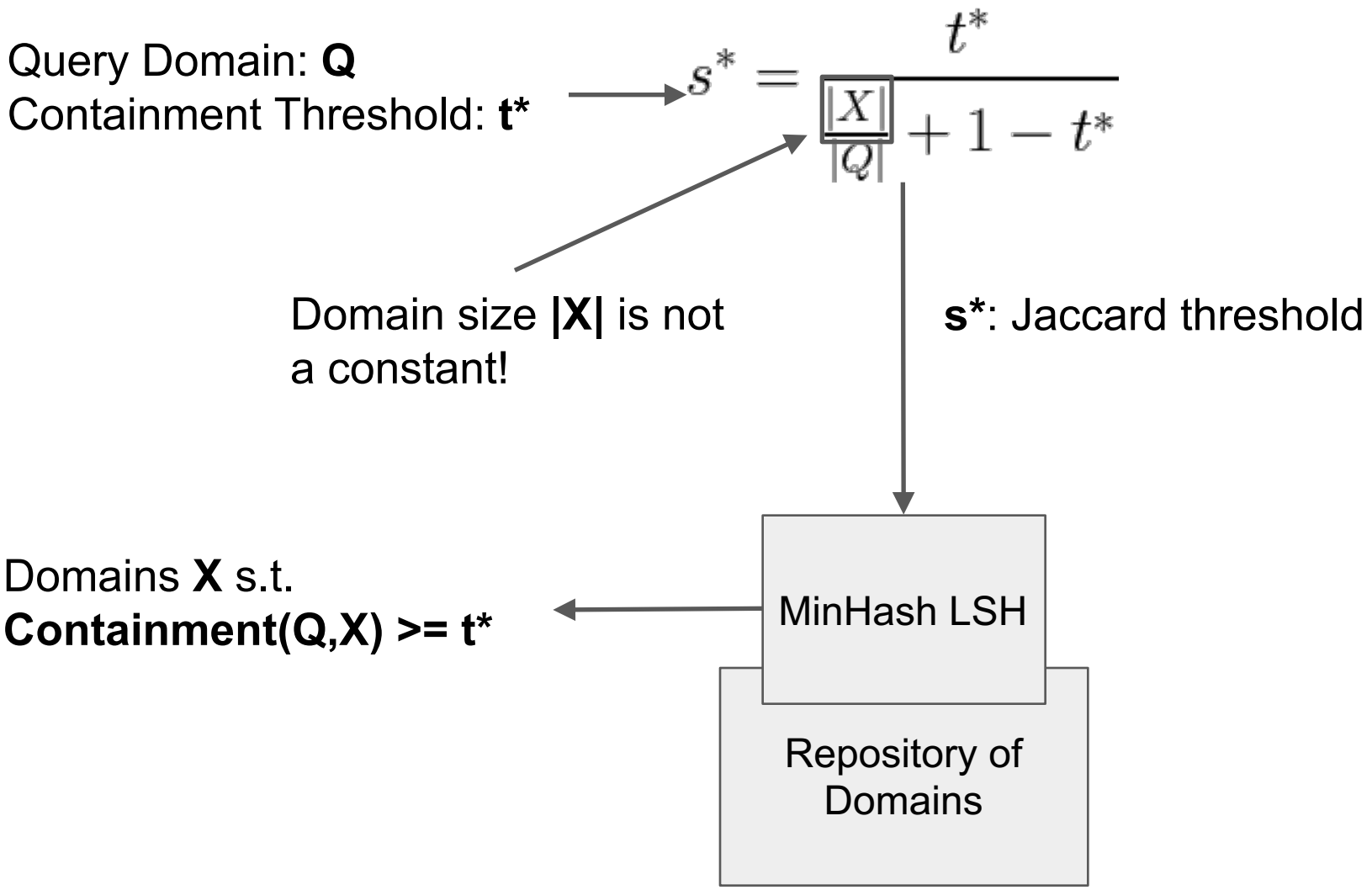
Query Domain: **Q**
Containment Threshold: **t***

$$s^* = \frac{t^*}{\frac{|X|}{|Q|} + 1 - t^*}$$

s*: Jaccard threshold

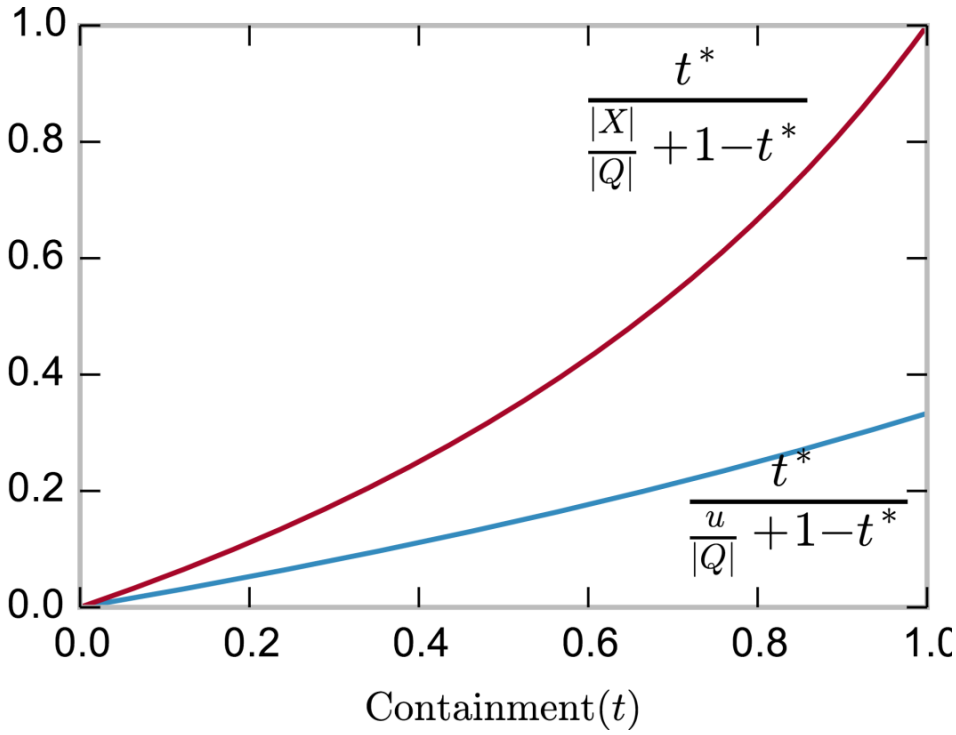


Domains **X** s.t.
Containment(Q,X) >= t*


Query Domain: **Q**
Containment Threshold: **t***
$s^* = \frac{t^*}{\frac{|X|}{|Q|} + 1 - t^*}$
Domain size **|X|** is not a constant!
s*: Jaccard threshold
Domains **X** s.t.
Containment(Q,X) >= t*
MinHash LSH
Repository of Domains

$$\frac{t^*}{\frac{u}{|Q|} + 1 - t^*} \leq \frac{t^*}{\frac{|X|}{|Q|} + 1 - t^*}$$

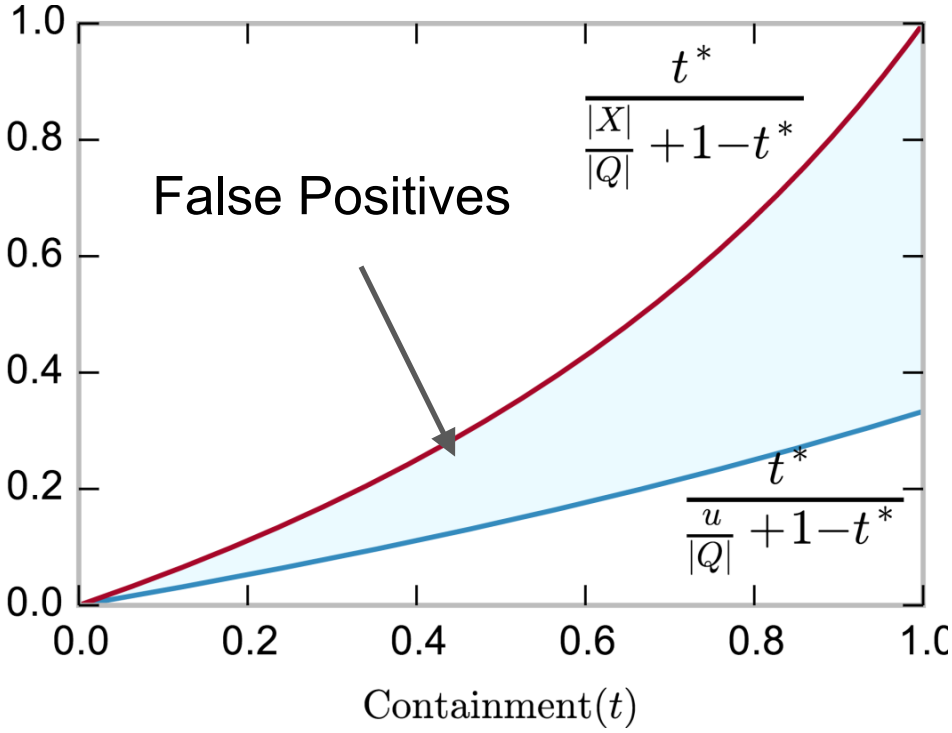
u: the upper bound of domain sizes in range **[l, u]**



$$s_u^* = \frac{t^*}{\frac{u}{|Q|} + 1 - t^*}$$

The new threshold introduces

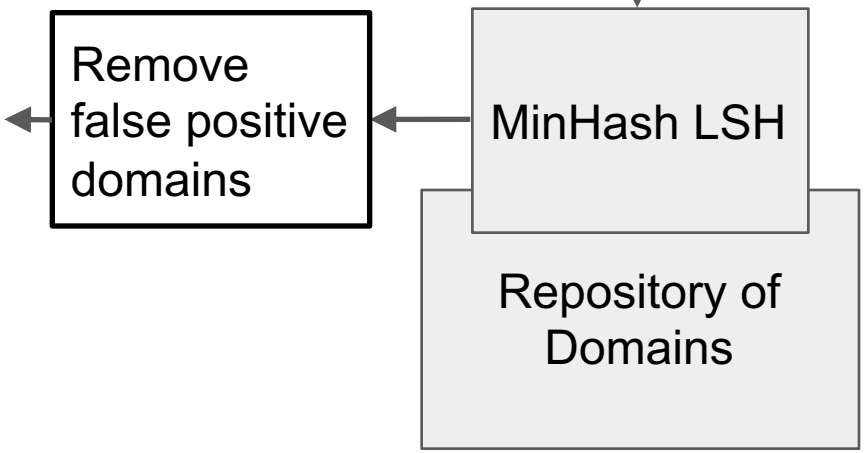
- false positive domains
- no false negative domains



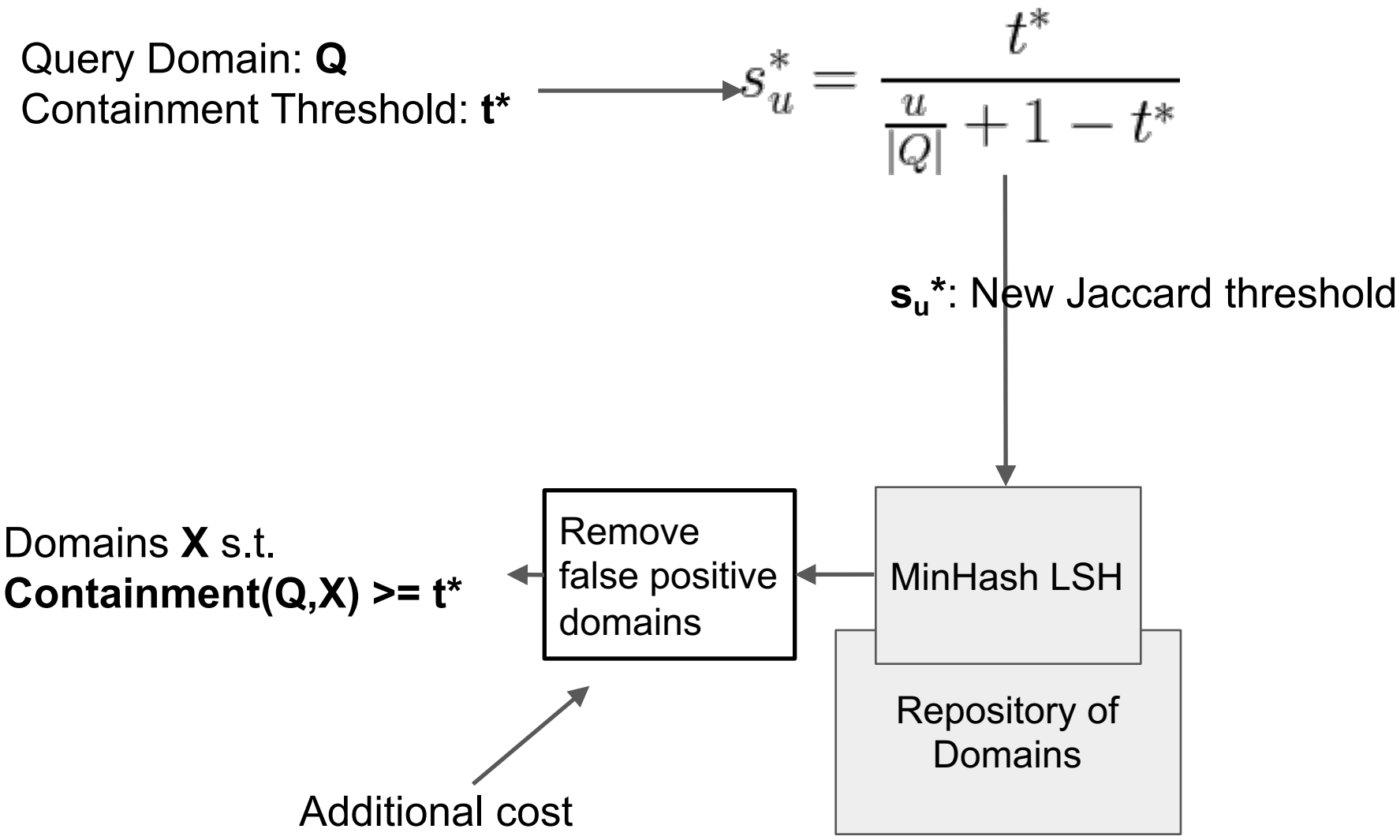
Query Domain: **Q**
Containment Threshold: **t***

$$s_u^* = \frac{t^*}{\frac{u}{|Q|} + 1 - t^*}$$

s_u*: New Jaccard threshold



Domains **X** s.t.
Containment(Q,X) >= t*


Query Domain: **Q**
Containment Threshold: **t***
$s_u^* = \frac{t^*}{\frac{u}{|Q|} + 1 - t^*}$
s_u*: New Jaccard threshold
MinHash LSH
Repository of Domains
Remove false positive domains
Domains **X** s.t.
Containment(Q,X) >= t*
Additional cost

The query cost can reduced if we produce less false positive domains.

$$T_{containment} = T_{Jaccard} + \Theta(correct\ domains) + \Theta(N_{l,u}^{FP})$$

Number of false positives domains in the range **[l, u]**

Number of
domains in [l, u]

$$N_{l,u}^{FP} \leq N_{l,u} \cdot \frac{u - l + 1}{2u}$$

Tight upper bound for
number of false positives
in range **[l, u]**

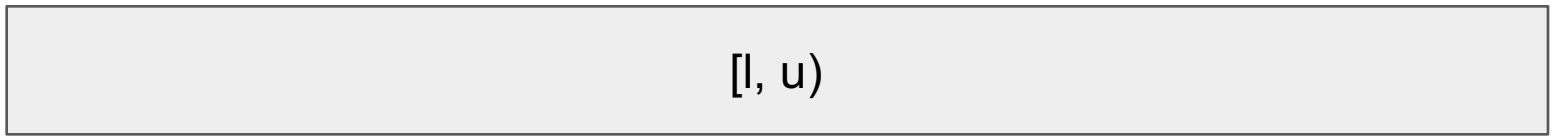
We can reduce the range size to reduce the number of false positives domains.

Number of domains in [l, u]

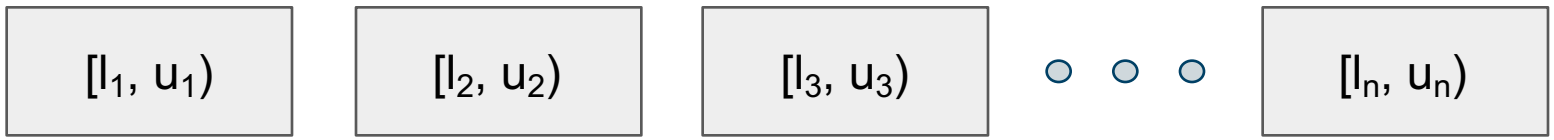
$$N_{l,u}^{FP} \leq N_{l,u} \cdot \frac{u - l + 1}{2u}$$

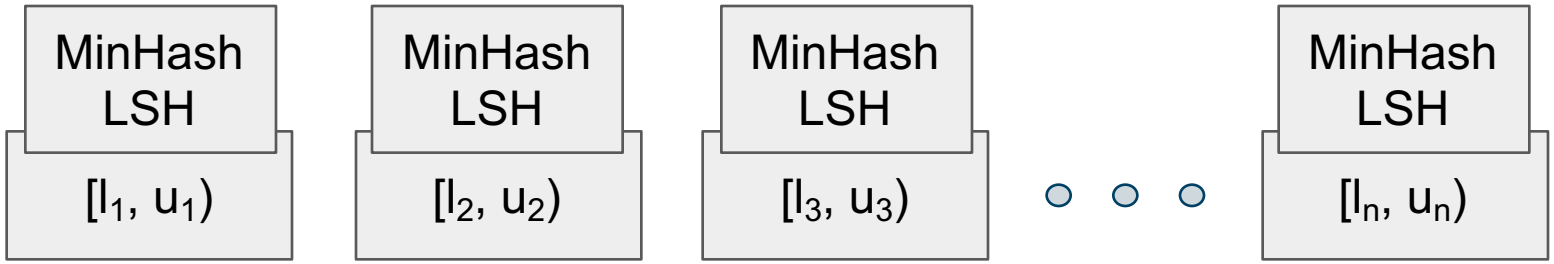
Tight upper bound for number of false positives in range **[l, u]**

Sorted by domain size

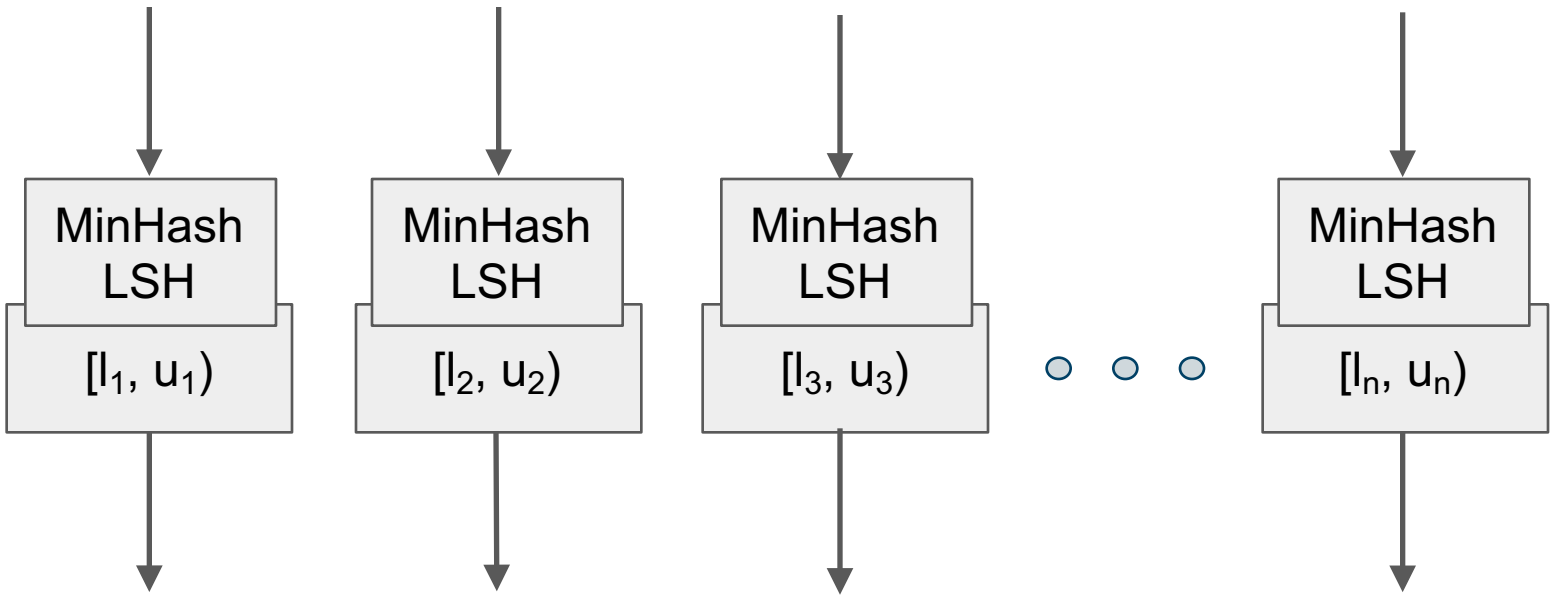


Contiguous domain **partitions**




MinHash LSH
$[l_1, u_1)$
MinHash LSH
$[l_2, u_2)$
MinHash LSH
$[l_3, u_3)$
MinHash LSH
$[l_n, u_n)$

Execute query in parallel



Query cost is determined by the partition with the most false positive domains

*This led us to formulate an **optimization problem** for partitioning using the upper bound of $N_{l,u}^{FP}$ on each partition.*

$$\Pi^* = \arg\min_{\Pi} \max_{1 \leq i \leq n} M_i$$

$$M_i = N_{l_i,u_i} \cdot \frac{u - l + 1}{2u}$$

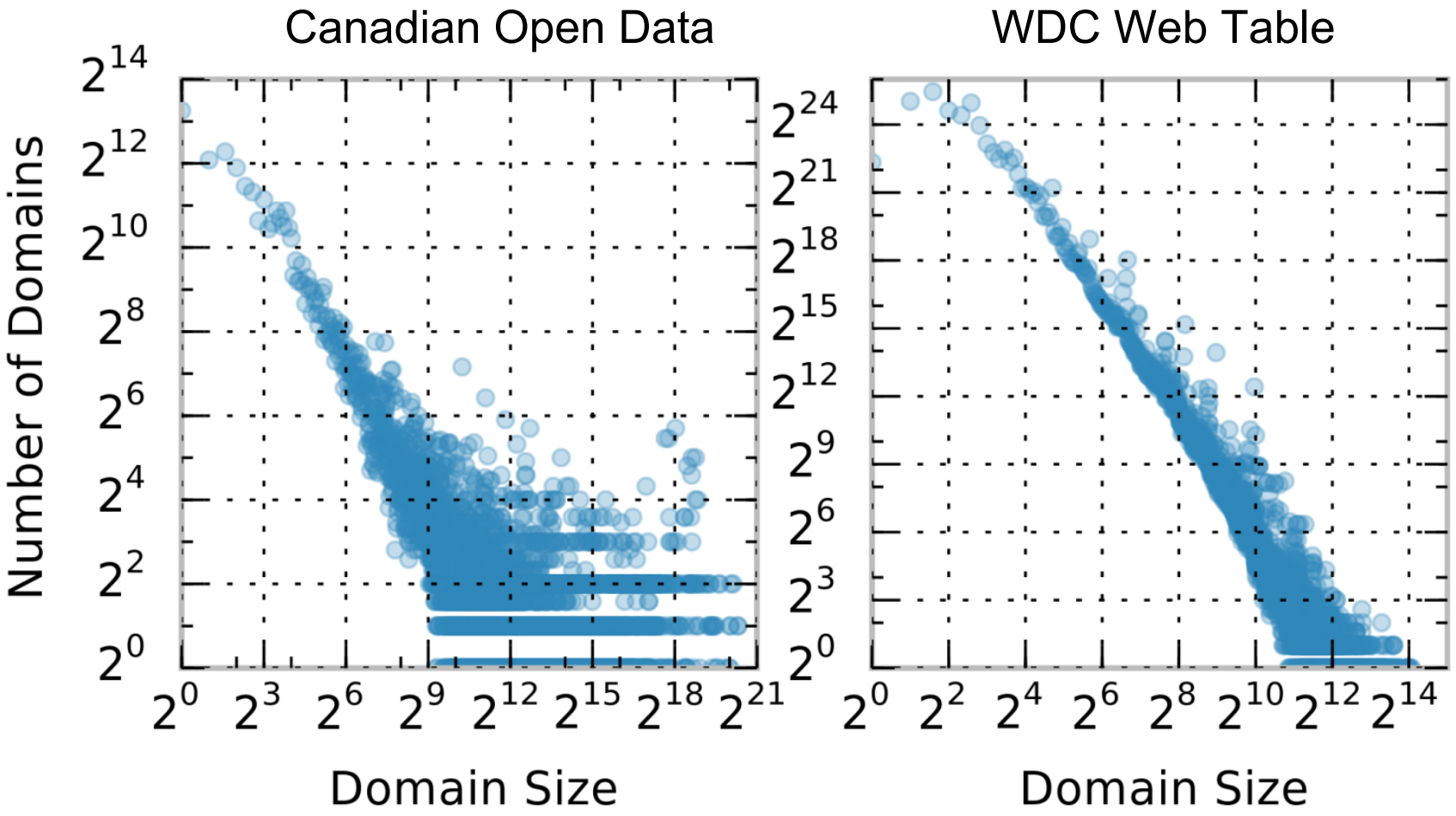
This is equivalent to finding a partitioning such that all partitions have the same M_i.

$$\exists \Pi^* \; s.t. \; M_i = M_j, \; \forall \; i, j$$

So far, we have shown:

1. *Partitioning improves query cost by reducing false positives, while maintaining accuracy*
2. *An optimal partitioning can be verified using a closed form equation*

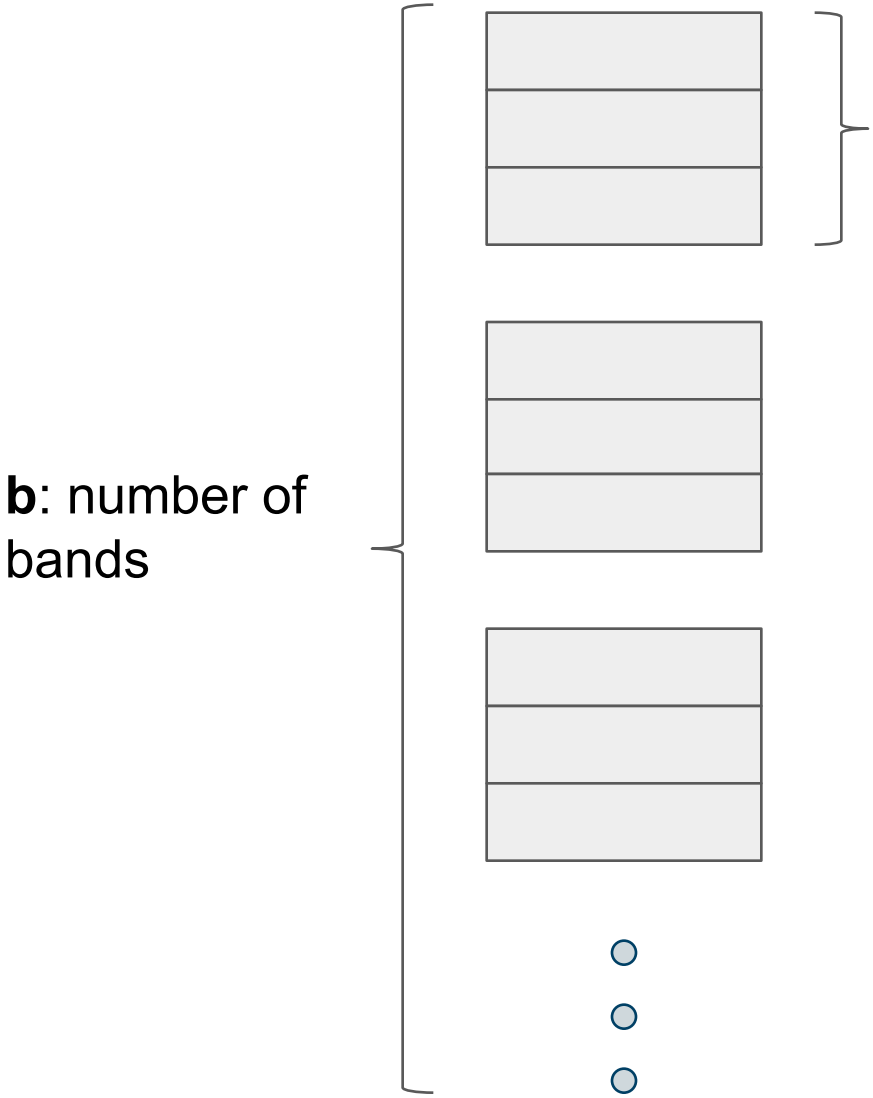
An optimal partitioning for a real-world domain size distribution?


Canadian Open Data
WDC Web Table
Number of Domains
Domain Size
Domain Size
2^{14}
2^{12}
2^{10}
2^{8}
2^{6}
2^{4}
2^{2}
2^{0}
2^{24}
2^{21}
2^{18}
2^{15}
2^{12}
2^{9}
2^{6}
2^{3}
2^{0}
2^{0} 2^{3} 2^{6} 2^{9} 2^{12} 2^{15} 2^{18} 2^{21}
2^{0} 2^{2} 2^{4} 2^{6} 2^{8} 2^{10} 2^{12} 2^{14}

We proved **equi-depth** partitioning is optimal for domains following **power-law** distribution

One last thing: ***tune*** *MinHash LSH for containment threshold*

MinHash LSH [Indyk 98]

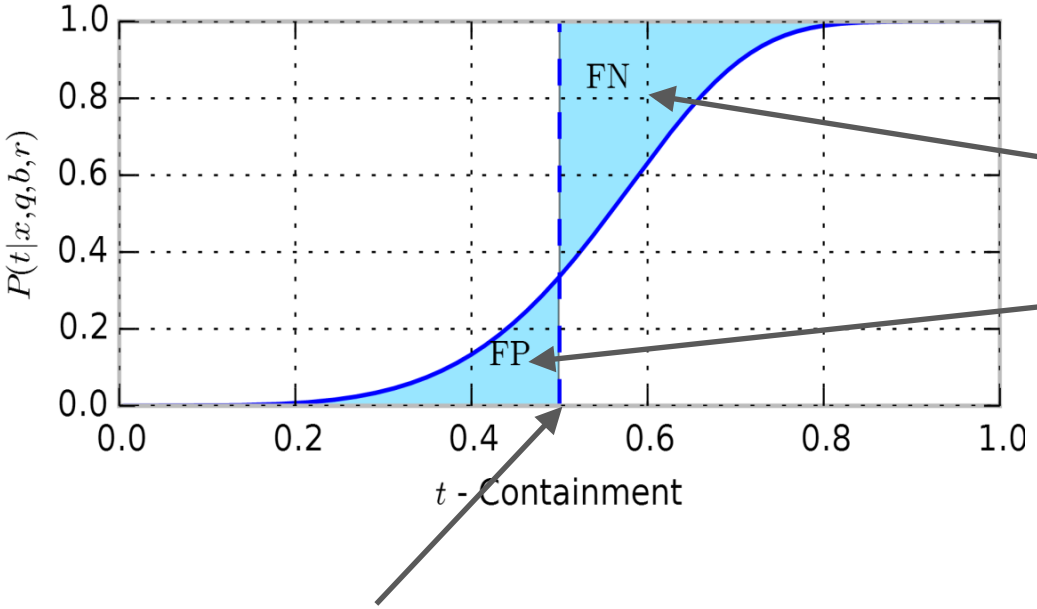


r: number of concatenated hash values in each band

b: number of bands

b and **r** can be self-tuned during query time with structure proposed in LSH Forest [Bawa 05]

We derived the probability of returning with respect to containment given parameters **b** and **r**



Our goal is to minimize the **sum** of false positive and negative probabilities

Containment threshold

Experimental Result

Compared against Asym MinHash [Shrivastava 2015] and MinHash LSH (using our Containment-to-Jaccard conversion) on accuracy, using Canadian Open Data (65,533 domains):

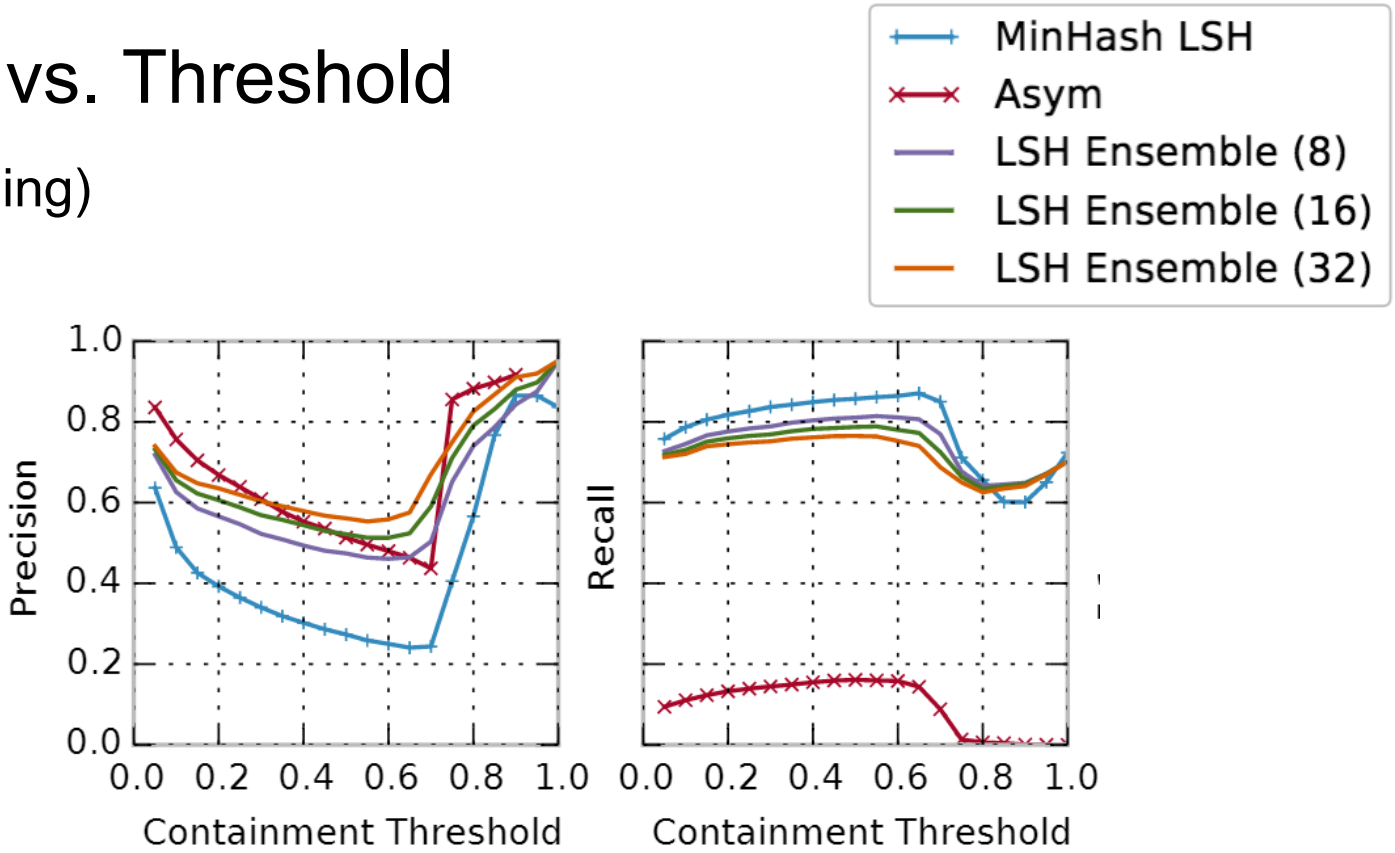
- LSH Ensemble consistently out-performs other techniques
- **More partitions leads to higher accuracy before pruning false positives**

Performance experiment used the complete 2015 WDC English Relational Web Table corpora (263 million domains):

- Mean query time around 3 seconds at 32 partitions
- **More partitions leads to lower query cost**

Accuracy vs. Threshold

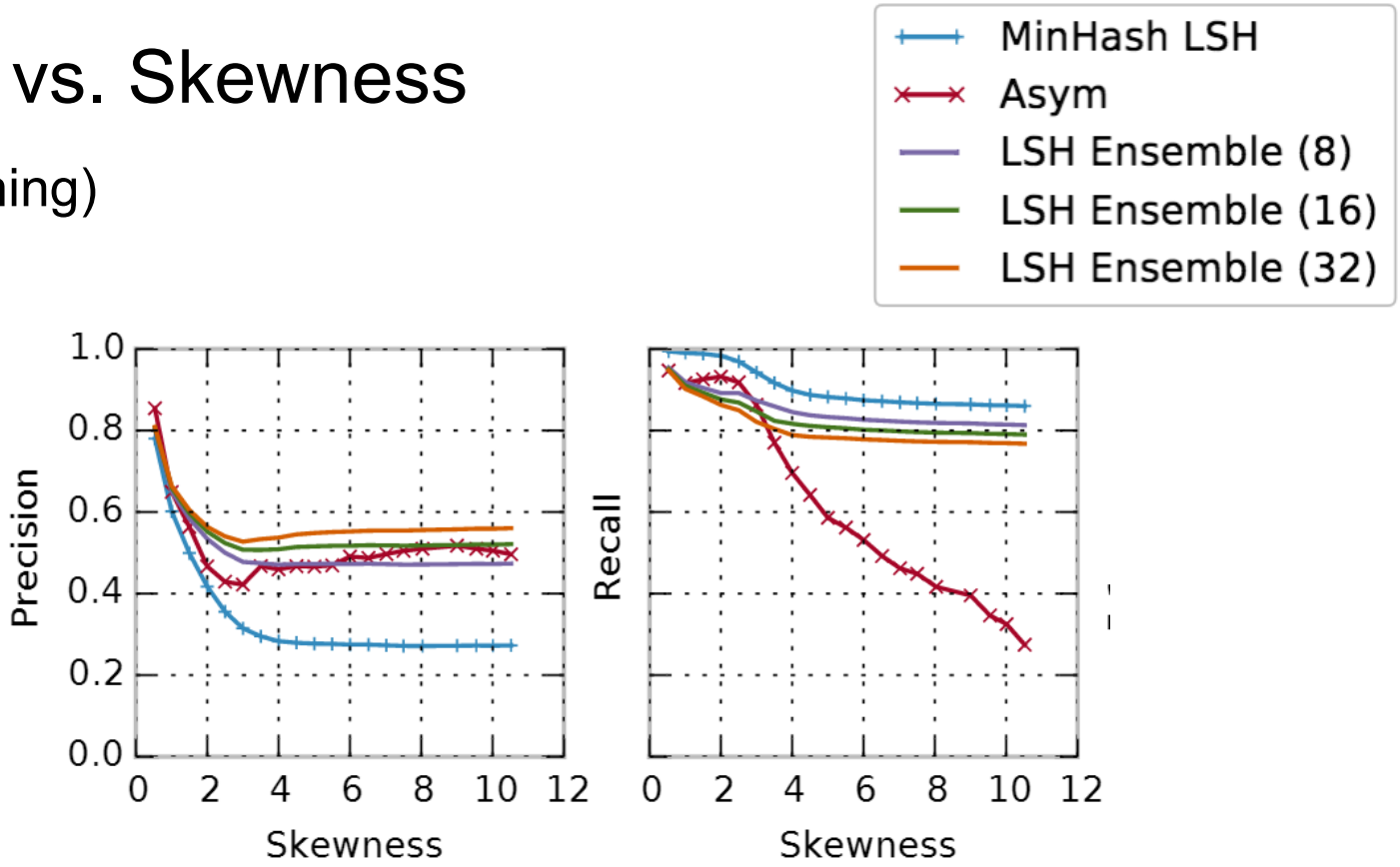
(Before Pruning)



- Creating more partition leads to fewer false positives, while maintaining recall
- Asymmetric MinHash LSH [Shrivastava 15] has high precision, but low recall due to padding

Accuracy vs. Skewness

(Before Pruning)



- Skewness in domain sizes have negative impact on accuracy for all indices
- LSH Ensemble handles skewness better than others

Query Performance

	Mean Query (sec)	Precision Before Pruning (**t*=0.5**)
MinHash LSH	45.13	0.27
LSH Ensemble (8)	7.55	0.48
LSH Ensemble (16)	4.26	0.53
LSH Ensemble (32)	3.12	0.58

On 263 million domains
(WDC Web Table)

Speed up is due to:

- Fewer false positive domains to process (higher precision)
- Parallelization

Recap

LSH Ensemble

- Uses multiple MinHash LSH built on domain size partitions to approximate containment search and maintain accuracy
- Optimal partitioning (equi-depth) for power-law distributions
- Self-tunable at query time given any threshold

Thank you!

Erkang (Eric) Zhu
ekzhu@cs.toronto.edu