

AUTOMATICALLY GENERATING DATA LINKAGES USING A DOMAIN-INDEPENDENT CANDIDATE SELECTION APPROACH

Dezhao Song and Jeff Heflin

SWAT Lab

Department of Computer Science and Engineering

Lehigh University

Outline

- Introduction
- Related Work
- Algorithms
- Evaluation
- Conclusion and Future work

Introduction



Entity
Coreference

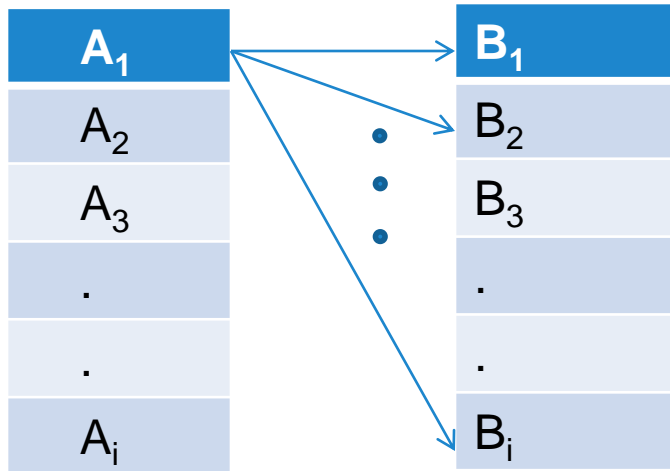
- What is “Data Linkage Generation”
 - Find instances that represent the same real world entity
- What is an instance
 - Person, publication, geographical location name mentions
 - *Dezhao Song* is a Ph.D. student at *Lehigh University*. *He* is in his fourth year at *LU*.
Check *Dezhao*'s website for more details.
 - Database records

Given name	Last name	Address	Zip	Email	Affiliation
Dezhao	Song	Packard Lab	18015	des308	LU
D.	Song	19 Memorial Dr. W.	18015	songdezhao	Lehigh
 - Ontology instances
 - <http://www.informatik.uni-trier.de/~ley/db/indices/a-tree/s/Song:Dezhao.html>
 - <http://data.semanticweb.org/person/dezhao-song/html>



owl:sameAs

The General/Naive Approach



Given name	Last name	Address	Zip	Email	Affiliation
Dezhao	Song	Packard Lab	18015	des308	LU
Dezhao	S.	19 Memorial Dr. W.	18015	songdezhao	Lehigh

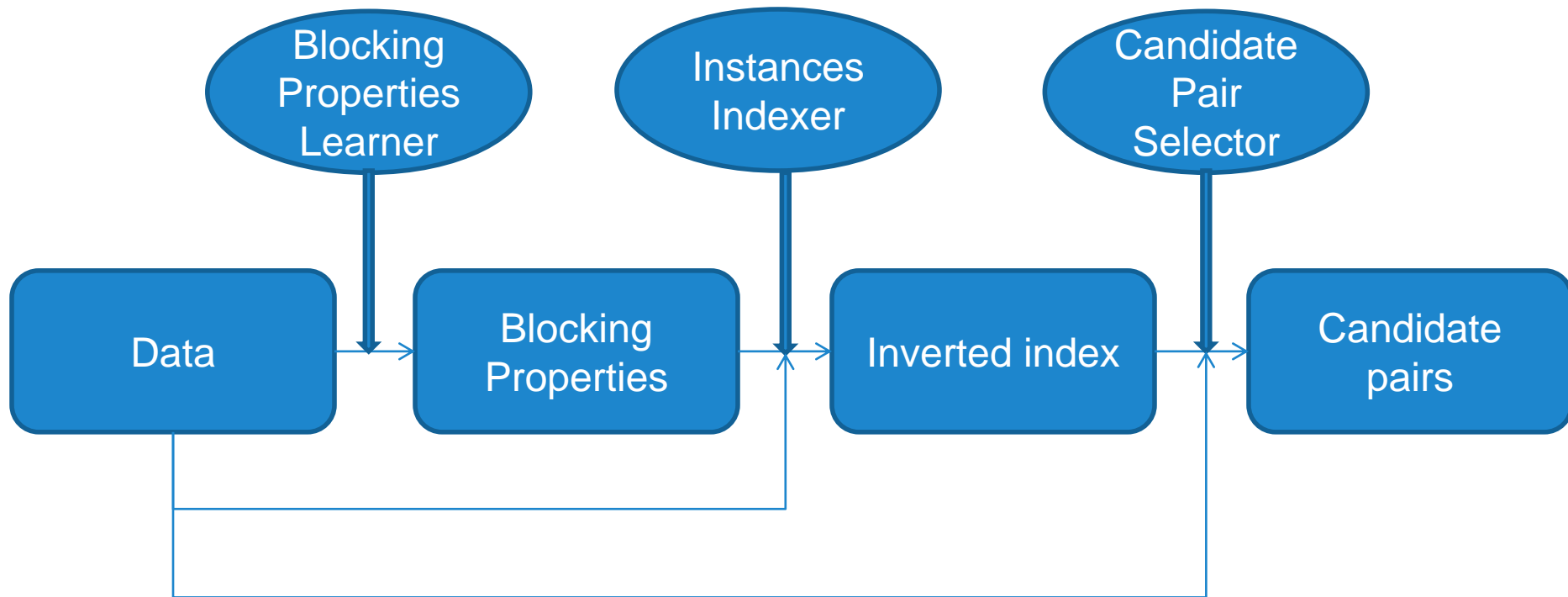
- Comparing all instance pairs between datasets A and B
 - Could be expensive on large datasets
- **Solution:** Reduce the search space by filtering out unlikely coreferent pairs
 - Choose the context used for filtering domain-independently
 - ✓ Maximally reduce pairs that are not coreferent
 - ✓ Retain as many true matches as possible
 - The filtering itself should be efficient
 - The entire entity coreference process should scale well after applying filtering while maintaining good precision and recall

Blocking/
Candidate
Selection

Related Work

- Candidate Selection – selecting a set of instance pairs
 - Ed-Join. Xiao et. al. (VLDB2008.)
 - All-Pairs. Bayardo et. al. (WWW2007.)
 - ASN. Yan et. al. (JCDDL 2007.)
 - BSL. Michelson and Knoblock. (AAAI2006, JAIR2008.)
 - Best five. W. E. Winkler. (Tech Report, U.S. Census Bureau 2005.)
 - Adaptive filtering. Gu and Baxter. (SIAM2004.)
 - Marlin. Bilenko and Mooney. (SIGKDD2003.)

System Framework

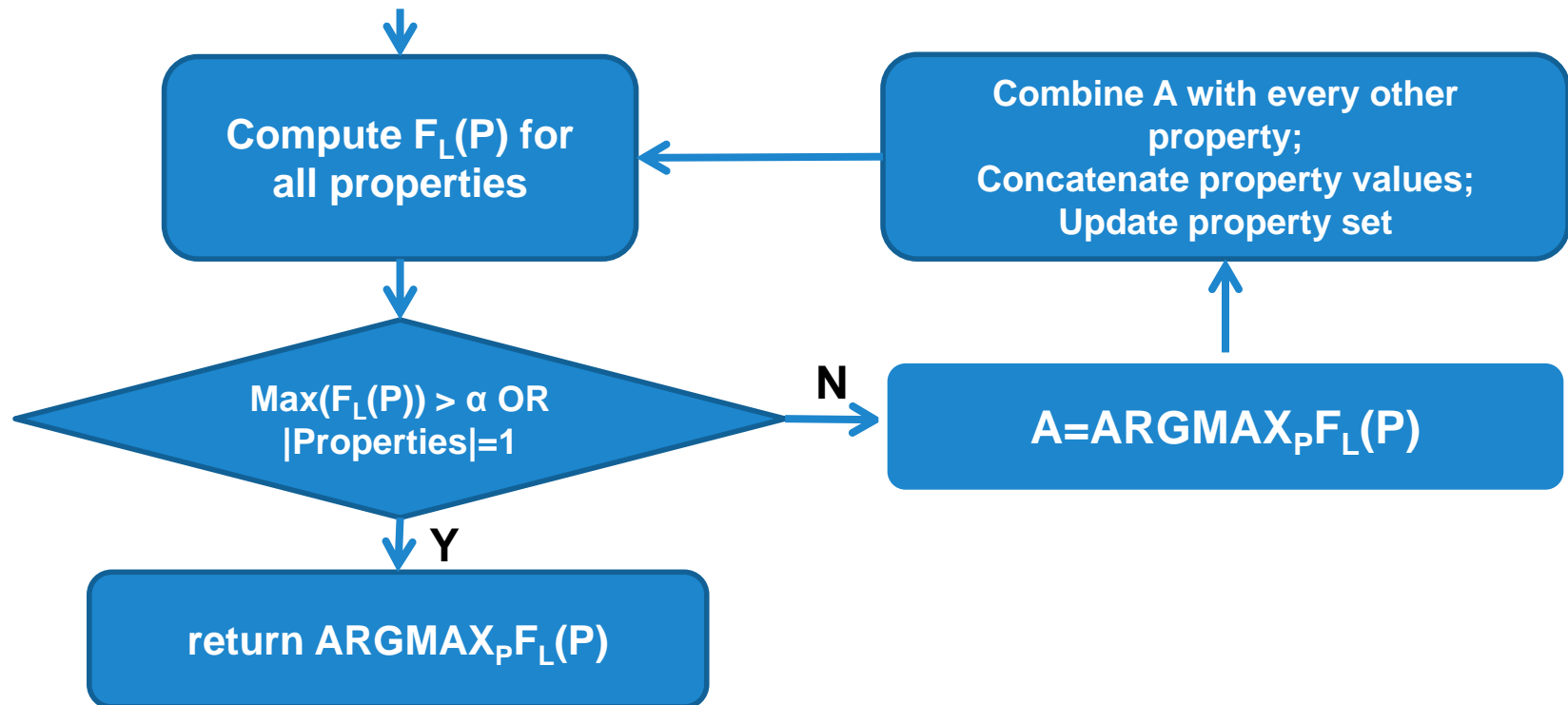


Learning Blocking Properties from RDF Graphs

- Example
 - Given name
 - Last name
- The criterion for choosing candidate selection properties
 - Sufficiently discriminating
 - Discriminability(p) = $\frac{|\text{distinct object values of a property}|}{|\text{triples of a property}|}$
 - Reduce non-coreferent pairs
 - Covering a majority of the instances
 - Coverage(p) = $\frac{|\text{instances that have a value for a property}|}{|\text{instances}|}$
 - Increase coverage on true matches
 - $F_L = 2 * \text{Discriminability} * \text{Coverage} / (\text{Discriminability} + \text{Coverage})$

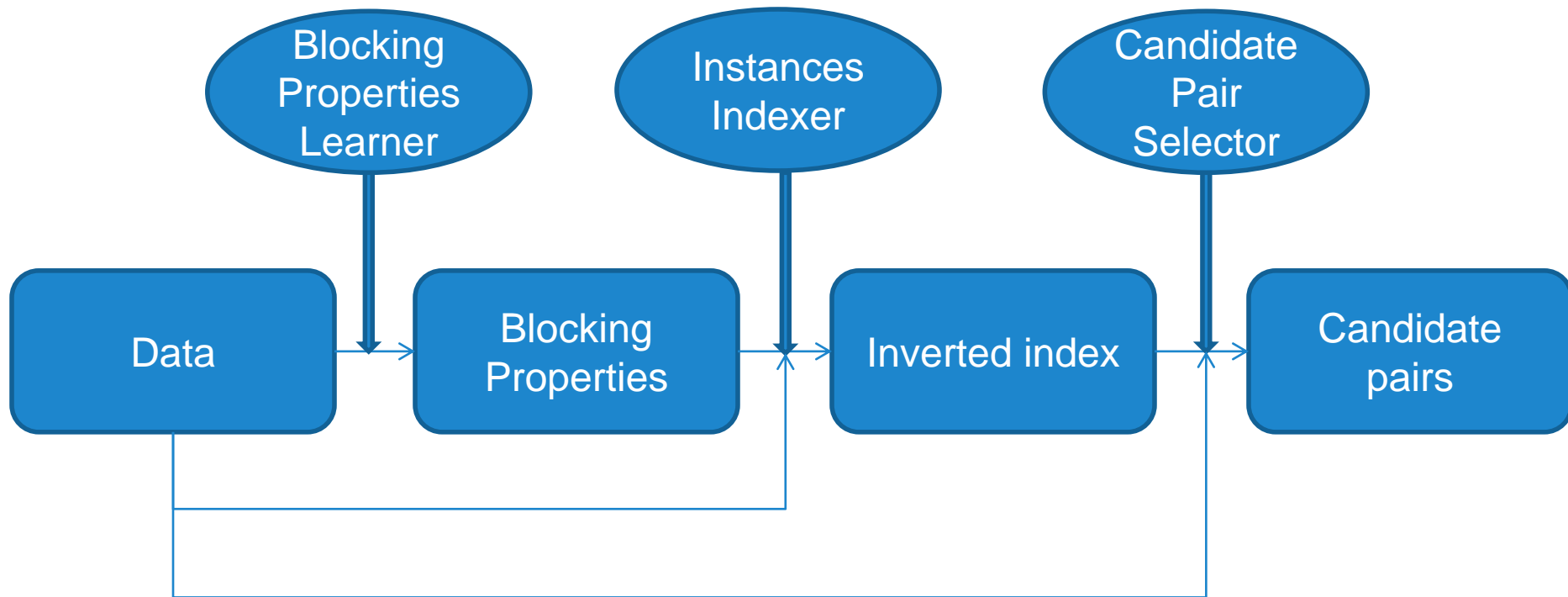
Combining Properties

- A single property may not be sufficient for candidate selection.
- E.g., Last name + given name could do better than last name.



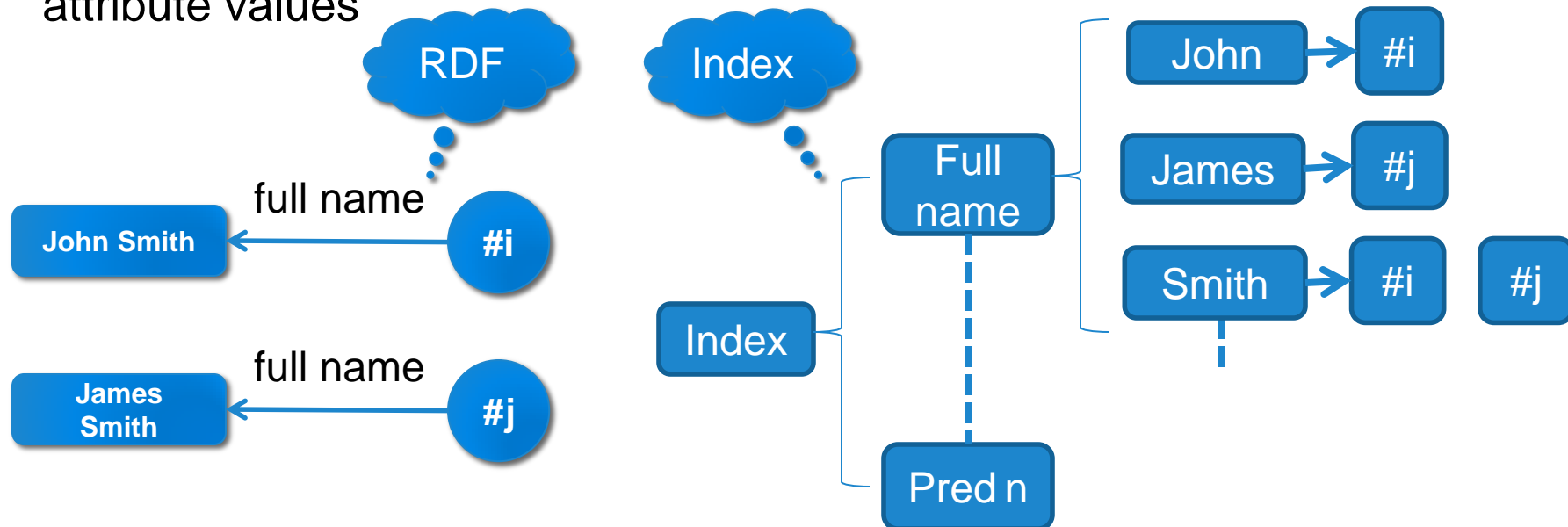
- A set of tuples: (instance, property, value), are formed.

System Framework

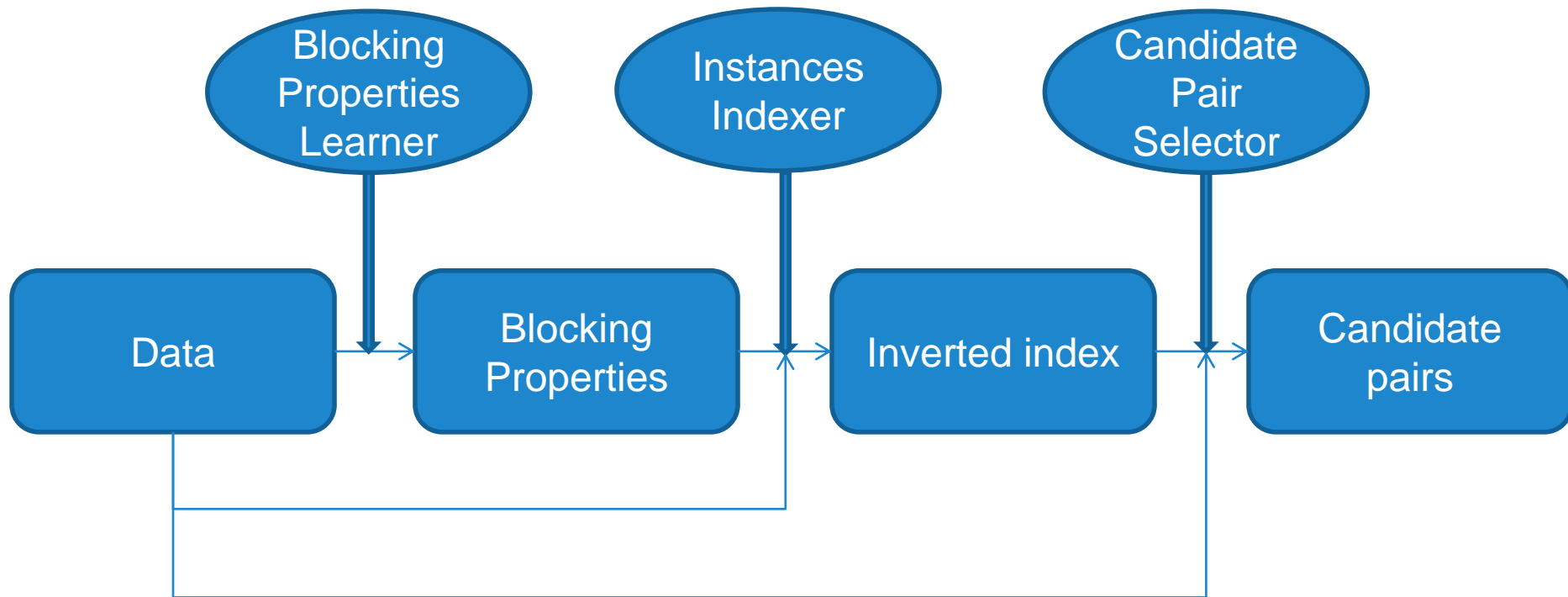


Indexing Instances

- Comparing all pairs of instances on the selected attributes can be expensive on datasets with millions of instances.
- Need a solution to do efficient look-up and only compare instances that at least have one token in common.
- Inverted Index – issue disjunctive queries with tokenized attribute values



System Framework

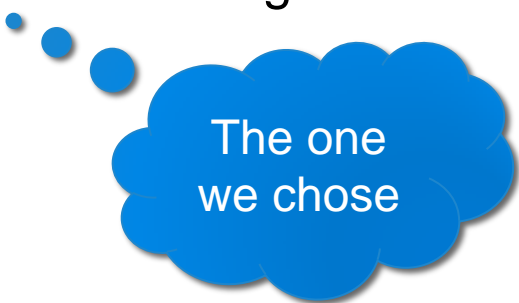


Selecting Candidate Instance Pairs

- For each tuple, the indexer returns a set of tuples whose values share at least one token on one predicate in the learned key.
- Still too many candidate pairs get returned.
- Need a second-level similarity measure to further reduce the size of the candidate set
 - Further reduce non-coreferent pairs
 - Need to be careful to maintain good recall

Alternative Similarity Measures

- Given two tuples (a, prop1, value1) & (b, prop1, value2):
 - Direct_Comp
 - Directly compute the string similarity between their values.
 - Token_Sim
 - Check the percentage of the shared highly similar tokens.
 - Character level bigram
 - Check the percentage of the shared character level bigrams.



The one
we chose

Evaluation

- Evaluation Metrics for candidate selection

- Pairwise Completeness (PC) = $\frac{|\text{True matches in candidate set}|}{|\text{groundtruth coreferent pairs}|}$

- Reduction Ratio (RR) = $\frac{|\text{Candidate Set}|}{|M * N|}$, M and N are the size of two instance sets respectively

- $F_{CS} = 2 * PC * RR / (PC + RR)$

- Runtime of the candidate selection process

- Evaluation Metrics for entity coreference

- Precision = $\frac{|\text{correctly detected pairs}|}{|\text{matches}|}$

- Recall = $\frac{|\text{correctly detected pairs}|}{|\text{all detected pairs}|}$

- F1-score = $2 * \text{Precision} * \text{Recall} / (\text{Precision} + \text{Recall})$

Datasets

Dataset	Size	Coreferent pairs	Properties	Type
RKB Person	100,000	123,719	names, publications, affiliation, etc.	RDF
RKB Publication	100,000	51,697	title, date, proceeding, Journal, etc.	RDF
SWAT Person	100,000	9,482	names, papers, etc.	RDF
Hotel	1125*132	1,028	name, rating, area, price and date.	Segmented posts
Restaurant	331*533	112	name, address, type and city	Segmented posts
Census	10,000	5,000	names, zip, date of birth, phone, age, etc.	CSV

Learned Candidate Selection Key

Dataset	Learned Properties
RKB Person	full-name, job, email, web-addr and phone
RKB Publication	title
SWAT Person	CiteSeer:name and FOAF:name
Hotel	name
Restaurant	name
Census	date-of-birth, surname and address_1

Evaluation on RDF Datasets

Dataset	System	Pairs	RR (%)	PC (%)	F_{cs} (%)	CS Time (s)	Coref F1 (%)	Total Time (s)
RKB Person	bigram	14,024	99.97	99.33	99.65	13.32	94.48	25.45
	All-Pairs Bayardo et. al. WWW2007	680,403	98.64	99.76	99.20	1.34	92.04	195.37
	EdJoin Xiao et. al. VLDB2008	150,074	99.70	99.72	99.71	1.73	92.38	72.79
	Naïve Song and Heflin. CIKM2010	N/A	N/A	N/A	N/A	N/A	91.64	4,766
RKB Publication	bigram	6,831	99.99	99.97	99.98	18.26	99.74	31.73
	All-Pairs	1,527,656	96.64	97.95	97.44	3.93	98.59	877.80
	EdJoin	2,579,333	94.84	98.57	96.67	409.08	99.04	1473.47
	Naive	N/A	N/A	N/A	N/A	N/A	99.55	34,567
SWAT Person	bigram	7,129	99.99	98.72	99.35	13.46	95.07	21.21
	All-Pairs	508,505	98.98	99.91	99.44	1.00	95.06	108.89
	EdJoin	228,830	99.54	99.79	99.66	409.08	95.01	51.66
	Naive	N/A	N/A	N/A	N/A	N/A	95.02	12,140

Dataset	System	Pairs	RR (%)	PC (%)	F _{cs} (%)
Restaurant	bigram	182	99.90	98.21	99.05
	All-Pairs	1,967	98.89	99.11	99.00
	EdJoin	6,715	96.19	96.43	96.31
	BSL	1,306	99.26	98.16	98.71
	ASN	N/A	N/A	<96	<98
	Marlin	78,773	55.35	100.00	71.26
Hotel	bigram	4,142	97.21	94.26	95.71
	All-Pairs	6,953	95.32	95.91	95.62
	EdJoin	17,623	88.13	98.93	93.22
	BSL	27,383	81.56	99.79	89.76
Census	bigram	166,844	99.67	97.76	98.70
	All-Pairs	5,231	99.99	100.00	99.99
	EdJoin	11,010	99.98	99.50	99.74
	BSL	939,906	98.12	99.85	98.98
	Best Five	239,976	99.52	99.16	99.34

Evaluation on Non-RDF Datasets

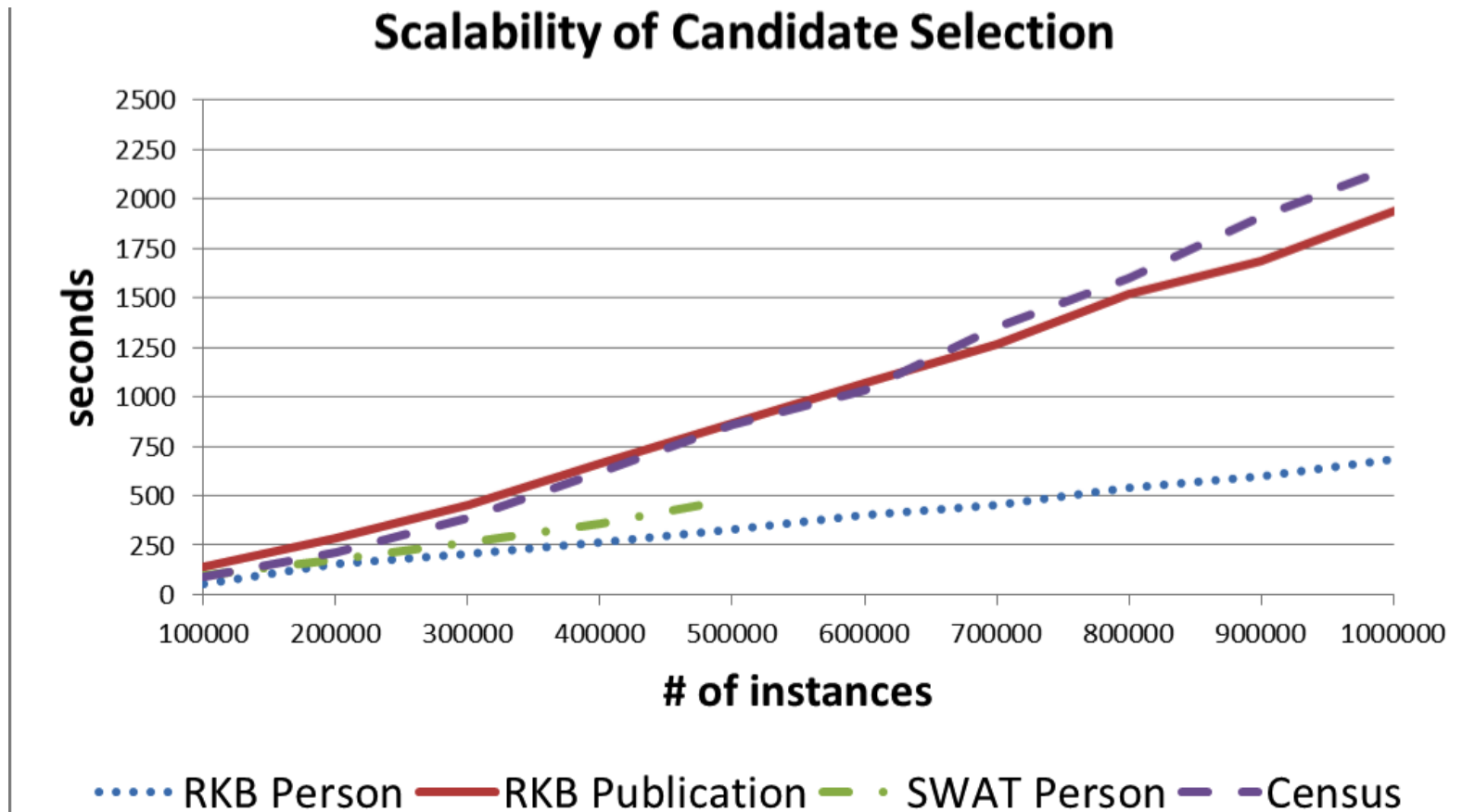
- ◆ Candidate Selection was applied to each of the three full datasets.
 - ◆ We actually ran bigram, All-Pairs, EdJoin to measure their performance
 - ◆ The number of selected pairs for Marlin, BSL and Best Five were estimated based upon their reported RR.
 - ◆ We reference reported results for the rest.

Scalability of Candidate Selection

- The purpose of a blocking algorithm is to help entity coreference algorithms to scale to large datasets.
- A blocking algorithm itself needs to scale well to such datasets.
- A single Sun workstation: 6GB memory and one eight-core Intel Xeon 2.93GHz processor.

Dataset	Size
RKB Person	1 million
RKB Publication	1 million
Census	1 million
SWAT Person	500,000

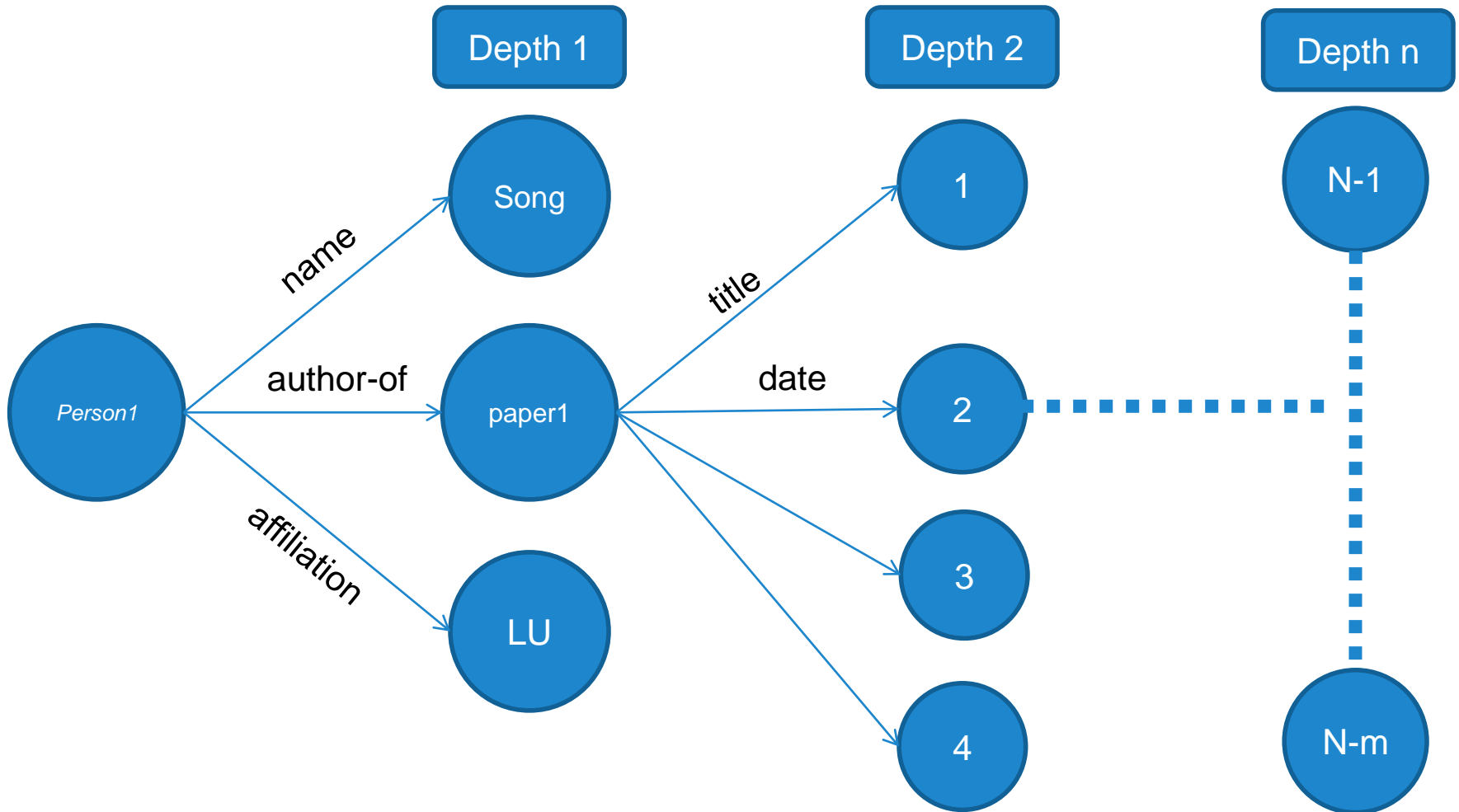
Runtime of Candidate Selection



Impact to the Entire Entity Coreference Process

- Compare to previously developed entity coreference algorithm (Song and Heflin CIKM2010)
 - Runtime Speedup Factor = $\frac{|\text{runtime without candidate selection}|}{|\text{runtime by applying candidate selection}|}$
 - Precision = $\frac{|\text{correctly detected pairs}|}{|\text{all detected pairs}|}$
 - Recall = $\frac{|\text{correctly detected pairs}|}{|\text{matches}|}$
 - F1-score = $2 * \text{Precision} * \text{Recall} / (\text{Precision} + \text{Recall})$

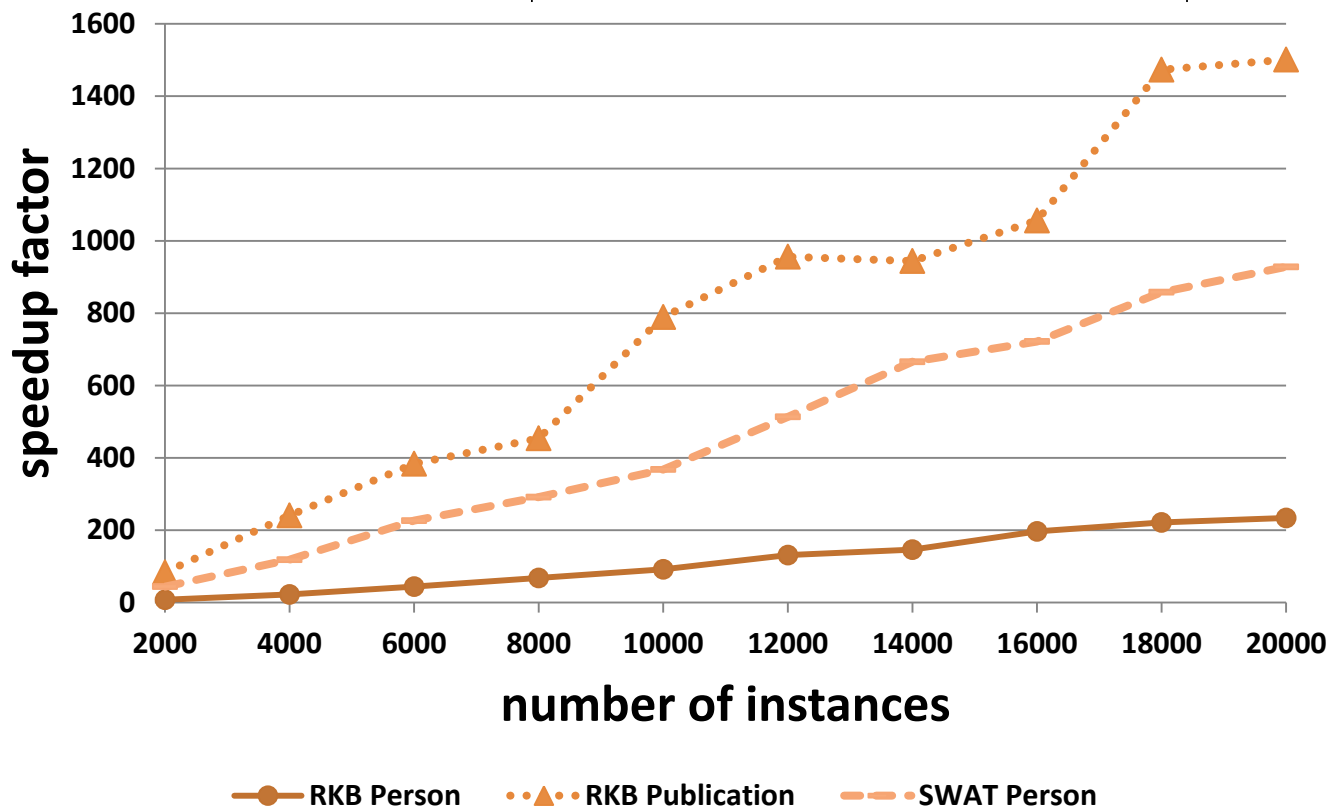
A Sample Neighborhood RDF Graph



Runtime Speedup of the Entire Entity Coreference Process

- Compare to previously developed entity coreference algorithm (Song and Heflin CIKM2010)

- Runtime Speedup Factor =
$$\frac{|\text{runtime without candidate selection}|}{|\text{runtime by applying candidate selection}|}$$



Conclusion and Future Work

- Developed a domain-independent candidate selection scheme for scaling entity coreference processes on structured data.
 - Achieved both high PC ($> 98\%$) and RR ($> 99\%$) on RDF data;
 - The blocking algorithm itself scales to large datasets > 1 million
 - The entire entity coreference process scales well – 10^{2-3}
 - Significant improvements were achieved on the final results of the entire entity coreference process – 1.84% on RKB Person
- Future work
 - Applying to even larger datasets
 - Applying to other entity coreference algorithms
 - Exploring appropriate methods for matching numeric data