Entity Resolution:
Past, present and yet-to-come
From structured to heterogeneous, to crowd-sourced, to deep learned

EDBT 2020, Copenhagen
30th March - 2nd April

George Papadakis
University of Athens
gpapadis@di.uoa.gr

Ekaterini Ioannou
University of Tilburg
Ekaterini.Ioannou@uvt.nl

Themis Palpanas
University of Paris
French University Institute
themis@mi.parisdescartes.fr

https://research.tilburguniversity.edu/en/projects/4ger

Structure Outline

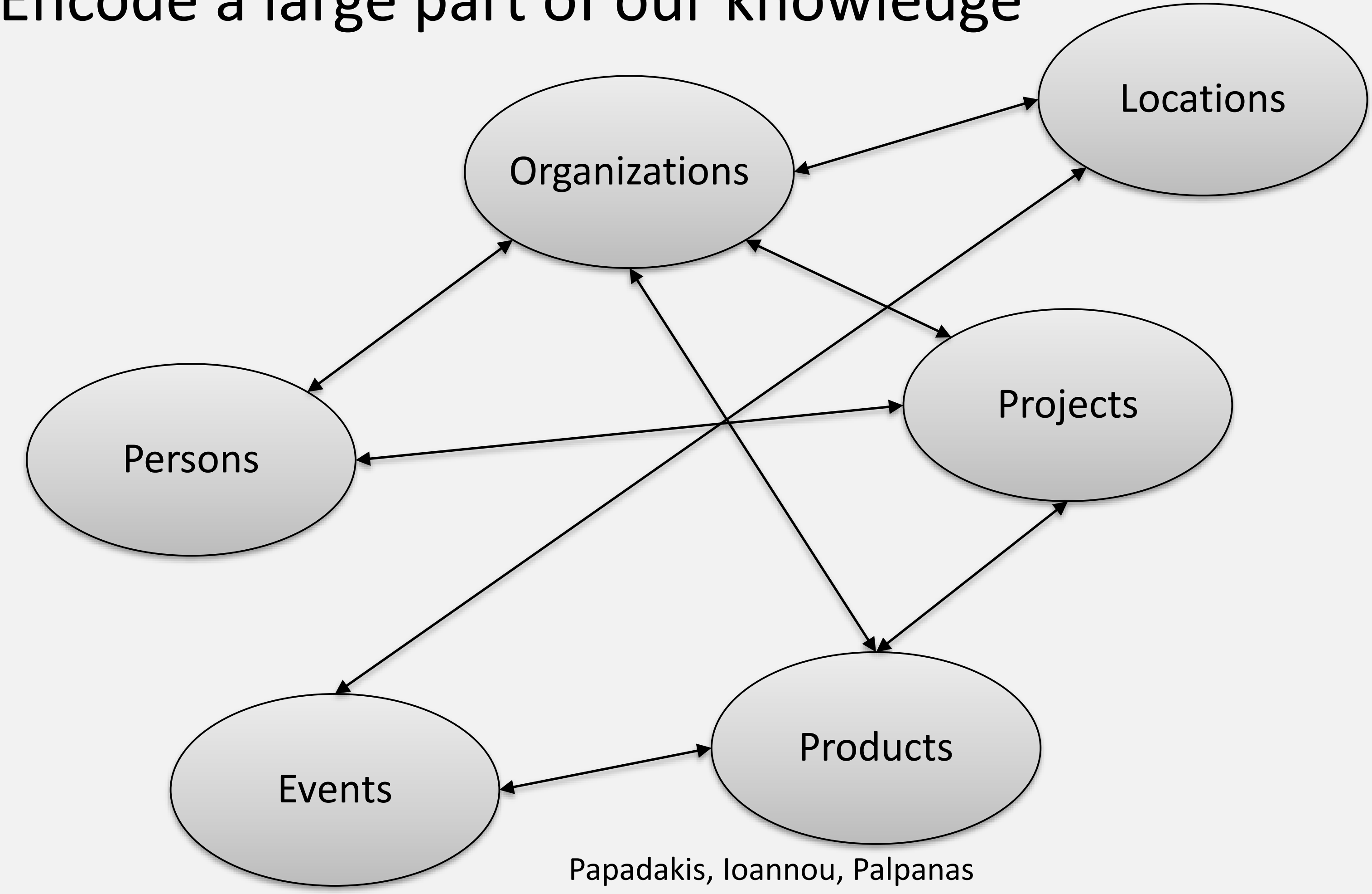
- Introduction
- Four Generations
- Entity Resolution Revisited: Leveraging External Knowledge
- Challenges and Final Remarks

Part A – Introduction

  - Motivation
  - Preliminaries
- Four Generations
- Entity Resolution Revisited: Leveraging External Knowledge
- Challenges and Final Remarks

Motivation

- Entities → invaluable asset for numerous current applications and systems
- Encode a large part of our knowledge



Matching, Linkage, Reconciliation, etc.

- Many names, descriptions, or IDs (URIs) are used for the same real-world "entity"
- Example:

Matching, Linkage, Reconciliation, etc.

- Many names, descriptions, or IDs (URIs) are used for the same real-world “entity”
- Example:

London 런던 ලන්ඩන් लंडन लंदन લંડન ለንደን ロンドン লন্ডন ลอนดอน இலண்டன் ლონდონი Llundain Londain Londe Londen Londen Londen Londinium London Londona Londonas Londoni Londono Londra Londres Londrez Londyn Lontoo Loundres Luân Đôn Lunden Lundúnir Lunnainn Lunnon لندن لندن لندن لوندون לאנדאן לונדון Λονδίνο Лёндан Лондан Лондон Лондон Лондон Լոնդոն 伦敦 …

Matching, Linkage, Reconciliation, etc.

- Many names, descriptions, or IDs (URIs) are used for the same real-world "entity"
- Example:

London 런던 ලන්ඩන් लंडन लंदन લંડન ለንደን ロンドン লন্ডন ลอนดอน இலண்டன் ლონდონი Llundain Londain Londe Londen Londen Londen Londinium London Londona Londonas Londoni Londono Londra Londres Londrez Londyn Lontoo Loundres Luân Đôn Lunden Lundúnir Lunnainn Lunnon لندن لندن لندن لوندون לאנדאן לונדון Λονδίνο Лёндан Лондан Лондон Лондон Лондон Լոնդոն 伦敦 …

capital of UK, host city of the IV Olympic Games, host city of the XIV Olympic Games, future host of the XXX Olympic Games, city of the Westminster Abbey, city of the London Eye, the city described by Charles Dickens in his novels, …

Matching, Linkage, Reconciliation, etc.

- Many names, descriptions, or IDs (URIs) are used for the same real-world “entity”
- Example:

London 런던 ލަންޑަން लंडन लंदन લંડન ለንደን ロンドン লন্ডন ลอนดอน இலண்டன் ლონდონი Llundain Londain Londe Londen Londen Londen Londinium London Londona Londonas Londoni Londono Londra Londres Londrez Londyn Lontoo Loundres Luân Đôn Lunden Lundúnir Lunnainn Lunnon لندن لندن لندن لوندون לאנדאן לונדון Λονδίνο Лёндан Лондан Лондон Лондон Лондон Լոնդոն 伦敦 …

capital of UK, host city of the IV Olympic Games, host city of the XIV Olympic Games, future host of the XXX Olympic Games, city of the Westminster Abbey, city of the London Eye, the city described by Charles Dickens in his novels, …

http://sws.geonames.org/2643743/
http://en.wikipedia.org/wiki/London
http://dbpedia.org/resource/Category:London
…

Disambiguation, Deduplication, etc.

- Plethora of different “entities” have the same name
- Example:
    - London, KY
    - London, Laurel, KY
    - London, OH
    - London, Madison, OH
    - London, AR
    - London, Pope, AR
    - London, TX
    - London, Kimble, TX
    - London, MO
    - London, London, MI
    - London, London, Monroe, MI
    - London, Uninc Conecuh County, AL
    - London, Uninc Conecuh County, Conecuh, AL
    - London, Uninc Shelby County, IN
    - London, Uninc Shelby County, Shelby, IN
    - London, Deerfield, WI
    - London, Deerfield, Dane, WI
    - London, Uninc Freeborn County, MN
    - ...

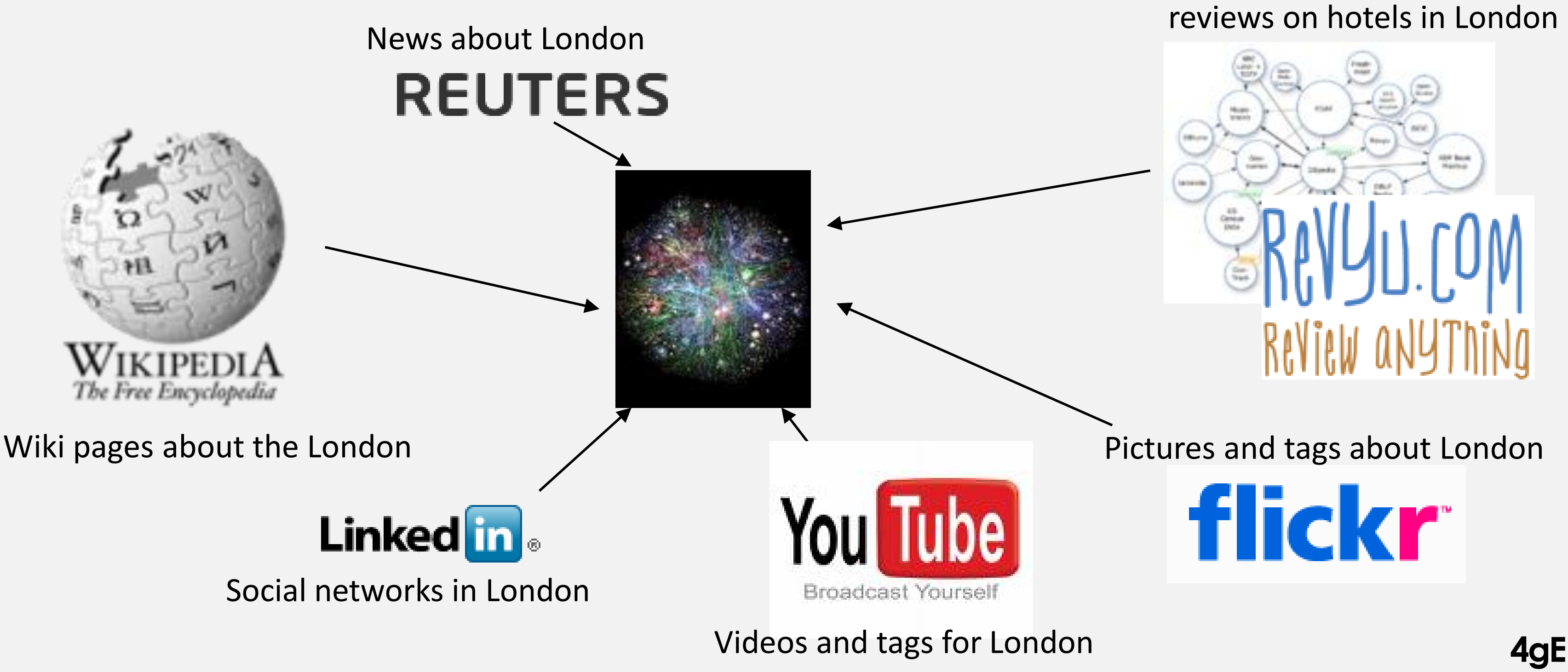
Disambiguation, Deduplication, etc.

- Plethora of different “entities” have the same name
- Example:
    - London, KY
    - London, Laurel, KY
    - London, OH
    - London, Madison, OH
    - London, AR
    - London, Pope, AR
    - London, TX
    - London, Kimble, TX
    - London, MO
    - London, London, MI
    - London, London, Mo
    - London, Uninc Conec
    - London, Uninc Conec
    - London, Uninc Shelby
    - London, Uninc Shelby
    - London, Deerfield, W
    - London, Deerfield, Da
    - London, Uninc Freebo
    - …

- London, Jack
  2612 Almes Dr
  Montgomery, AL
  (334) 272-7005
- London, Jack R
  2511 Winchester Rd
  Montgomery, AL 36106-33
  (334) 272-7005
- London, Jack
  1222 Whitetail Trl
  Van Buren, AR 72956-7368
  (479) 474-4136
- London, Jack
  7400 Vista Del Mar Ave
  La Jolla, CA 92037-4954
  (858) 456-1850
- …

Entities in today’s settings

- Content providers provide valuable information describing (part of) real-world “entities”
- ER are required for data integration, link discovery, query answering, Web / object-oriented searching, etc.



News about London

reviews on hotels in London

Wiki pages about the London

Pictures and tags about London

Social networks in London

Videos and tags for London

Entity Resolution

- Identifies and aggregates the different entity profiles that describe the same objects [1,2,3,4]
- Primary usefulness:
  - Improves data quality and integrity
  - Fosters re-use of existing data sources
- Example application domains:
  - Linked Data
  - Building Knowledge Graphs
  - Census data
  - Price comparison portals

Types of Entity Resolution

- The given entity collections can be of two types:

  clean + dirty [3,5]

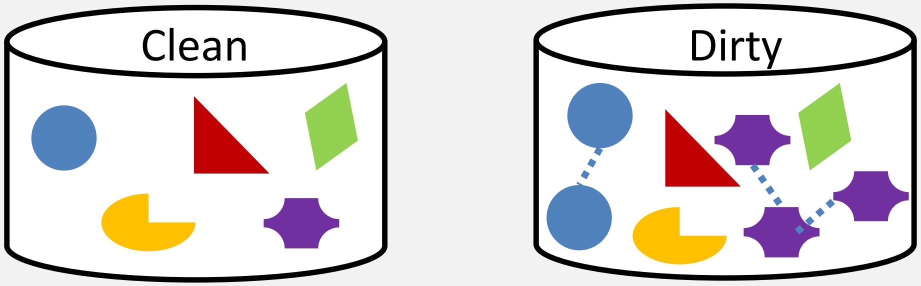
- Clean:
  - Duplicate-free data
  - E.g., DBLP, ACM Digital Library, Wikipedia, Freebase
- Dirty:
  - Contain duplicate entity profiles
  - E.g., Google Scholar, CiteseerX

Types of Entity Resolution

- The given entity collections can be of two types:

  clean + dirty [3,5]

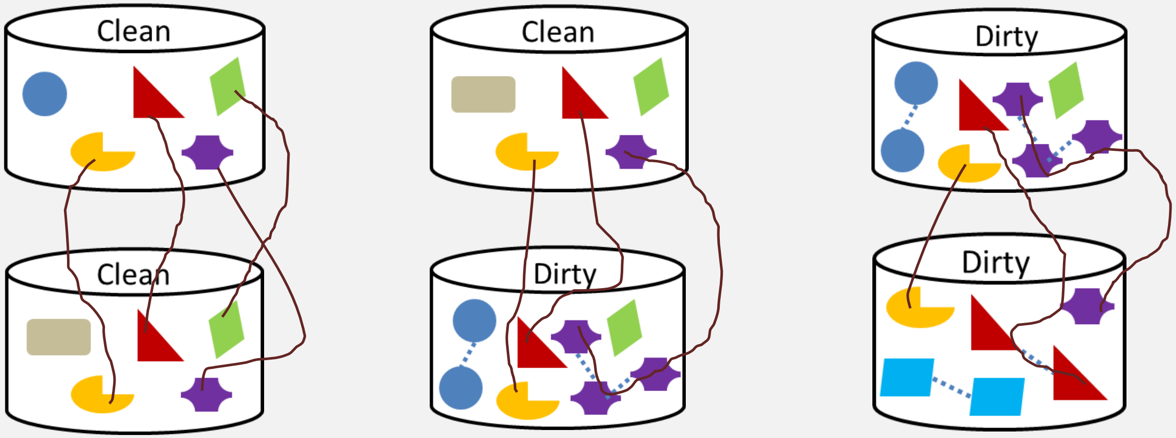
- Clean:
  - Duplicate-free data
  - E.g., DBLP, ACM Digital Library, Wikipedia, Freebase
- Dirty:
  - Contain duplicate entity profiles
  - E.g., Google Scholar, CiteseerX



Types of Entity Resolution

- Based on the quality of input, we distinguish entity resolution into 3 sub-tasks:
  1. Clean-Clean ER (a.k.a. ***Record Linkage*** in databases)
  2. Dirty-Clean ER
  3. Dirty-Dirty ER

  } Equivalent to **Dirty ER** (a.k.a. ***Deduplication*** in databases)



References

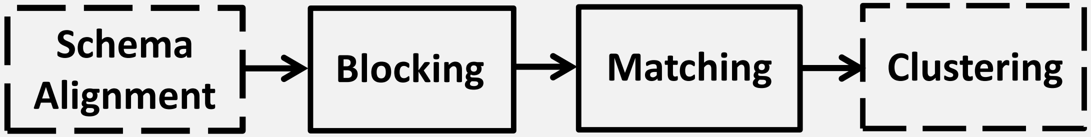
1. X. L. Dong, D. Srivastava. Big Data Integration. Synthesis Lectures on Data Management, Morgan & Claypool Publishers 2015, pp. 1-198.
2. A. K. Elmagarmid, P. G. Ipeirotis, V. S. Verykios. Duplicate Record Detection: A Survey. IEEE Trans. Knowl. Data Eng. 19(1): 1-16 (2007).
3. V. Christophides, V. Efthymiou, K. Stefanidis. Entity Resolution in the Web of Data. Synthesis Lectures on the Semantic Web: Theory and Technology, Morgan & Claypool Publishers 2015.
4. P. Christen. Data Matching - Concepts and Techniques for Record Linkage, Entity Resolution, and Duplicate Detection. Data-Centric Systems and Applications, Springer 2012, ISBN 978-3-642-31163-5, pp. I-XIX, 1-270.
5. P. Christen. A Survey of Indexing Techniques for Scalable Record Linkage and Deduplication. IEEE Trans. Knowl. Data Eng. 24(9): 1537-1555 (2012).

- Introduction

Part B – Four Generations

  - Generation 1: tackling Veracity
  - Generation 2: tackling Volume and Veracity
  - Generation 3: tackling Variety, Volume and Veracity
  - Generation 4: tackling Velocity, Variety, Volume and Veracity
- Entity Resolution Revisited: Leveraging External Knowledge
- Challenges and Final Remarks

Generation 1: Tackling Veracity



- Earliest approach
- Scope:
  - Structured data
- Goal:
  - Achieve high accuracy despite inconsistencies, noise, or errors in entity profiles
- Assumptions:
  - Known schema → custom, schema-based solutions

Step 1: Schema Alignment / Matching

- Scope:
  - Record Linkage
- Goal:
  - Create mappings between equivalent attributes of the two schemata, e.g., *profession ≡ job*
- Types of Solutions:
  - Structure-based
  - Instance-based
  - Usage-based
  - Hybrid

Step 1: Schema Alignment / Matching

- Taxonomy of Main Schema Matching Methods (in chronological order)

Method	Category	Type of Evidence
Cupid [1]	Structure-based	Name similarity, Constraints, Contextual similarity
Similarity Flooding [2]	Structure-based	Name similarity, Contextual similarity
COMA [3]	Hybrid	Name similarity, Constraints, Contextual similarity
Distribution-based [4]	Instance-based	Value distribution

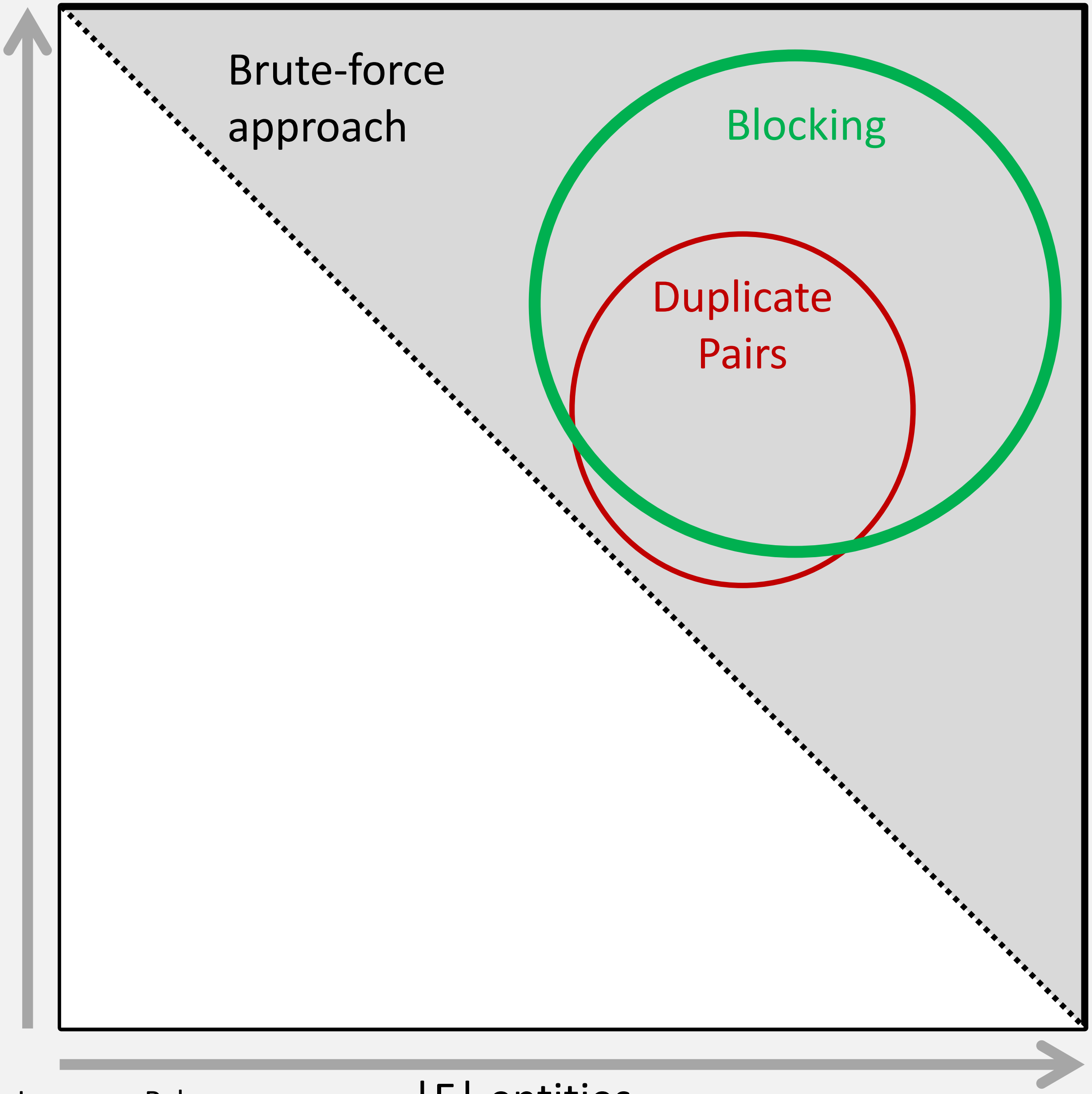
Step 2: Blocking

- Scope:
  - Both Deduplication and Record Linkage
- Goal:
  - ER is an inherently quadratic problem, $O(n^2)$: every entity has to be compared with all others
  - Blocking groups similar entities into blocks
    - Comparisons are executed only inside each block
    - Complexity is now quadratic to the size of the block (much smaller than dataset size!)

Computational cost

Input:
Entity Collection E

E.g.: For a dataset with 100,000 entities:
~10^{10} comparisons,
If 0.05 msec each →
>100 hours in total



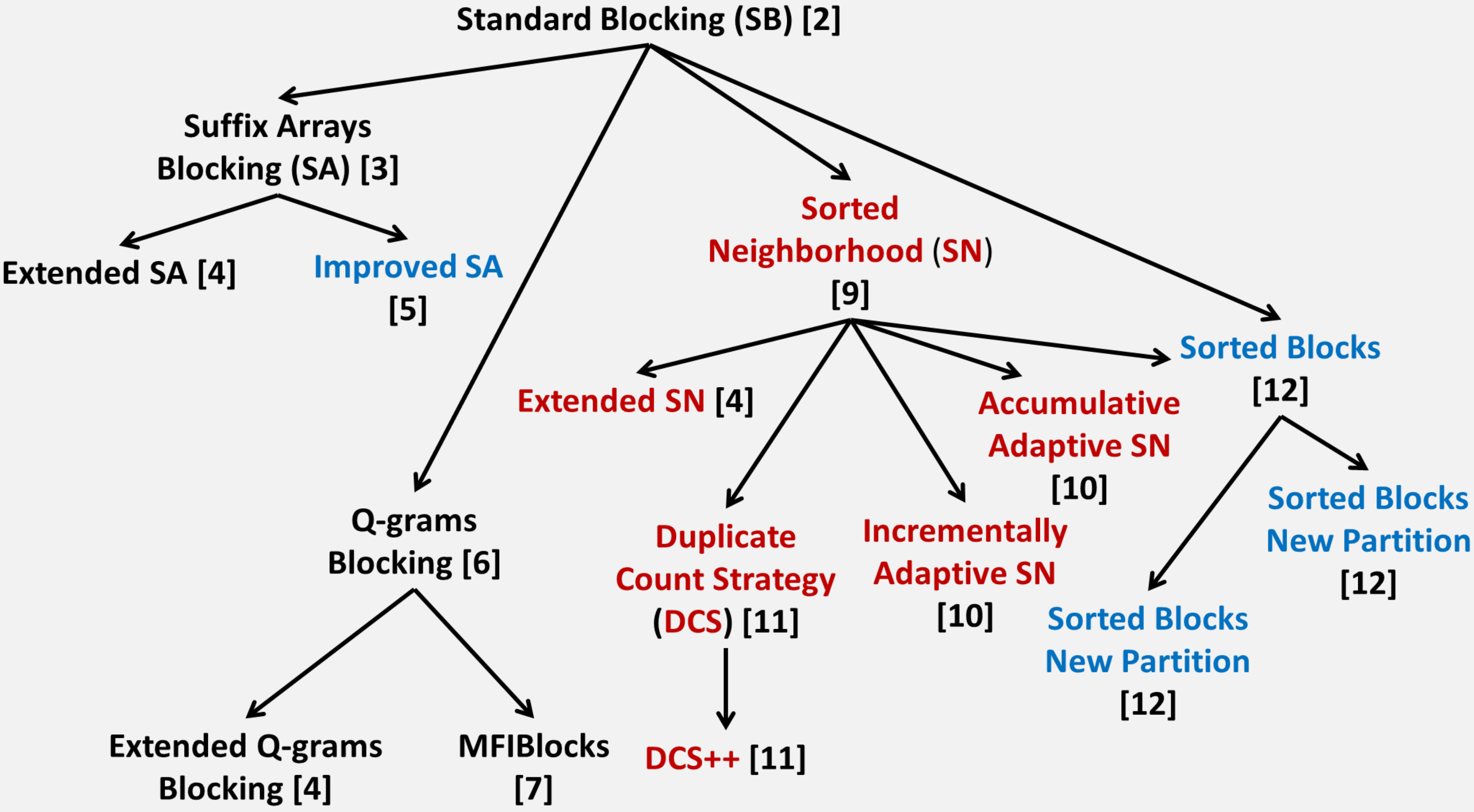
General Principles of Blocking

1. Represent each entity by *one or more* signatures called blocking keys
   - Focus on string values
2. Place into blocks all entities having the *same or similar* blocking key
3. Two matching profiles can be detected as long as they co-occur in at least one block
   - Trade-off between recall and precision!

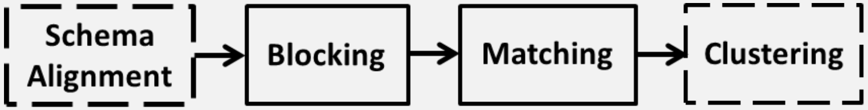
Taxonomy of Blocking Methods [1]

Method	Key Type	Redundancy awareness	Matching awareness	Key selection
Standard Blocking [2]	Hash-based	Red.-free	Static	Non-learning
Suffix Arrays [3] + [4,5]	Hash-based	Red.-positive	Static	Non-learning
Q-grams Blocking [6] + [4]	Hash-based	Red.-positive	Static	Non-learning
MFIBlocks [7]	Hash-based	Red.-positive	Static	Non-learning
Sorted Neighborhood [9] + [4,10]	Sort-based	Red.-neutral	Static	Non-learning
Duplicate Count Strategy [11]	Sort-based	Red.-neutral	Dynamic	Non-learning
Sorted Blocks [12]	Hybrid	Red.-neutral	Static	Non-learning
ApproxDNF [13]	Hash-based	Red.-positive	Static	Learning-based
Blocking Scheme Learner [14]	Hash-based	Red.-positive	Static	Learning-based
CBlock [15]	Hash-based	Red.-positive	Static	Learning-based
FisherDisjunctive [16]	Hash-based	Red.-positive	Static	Learning-based

Genealogy Tree of Non-learning Blocking Methods [1]

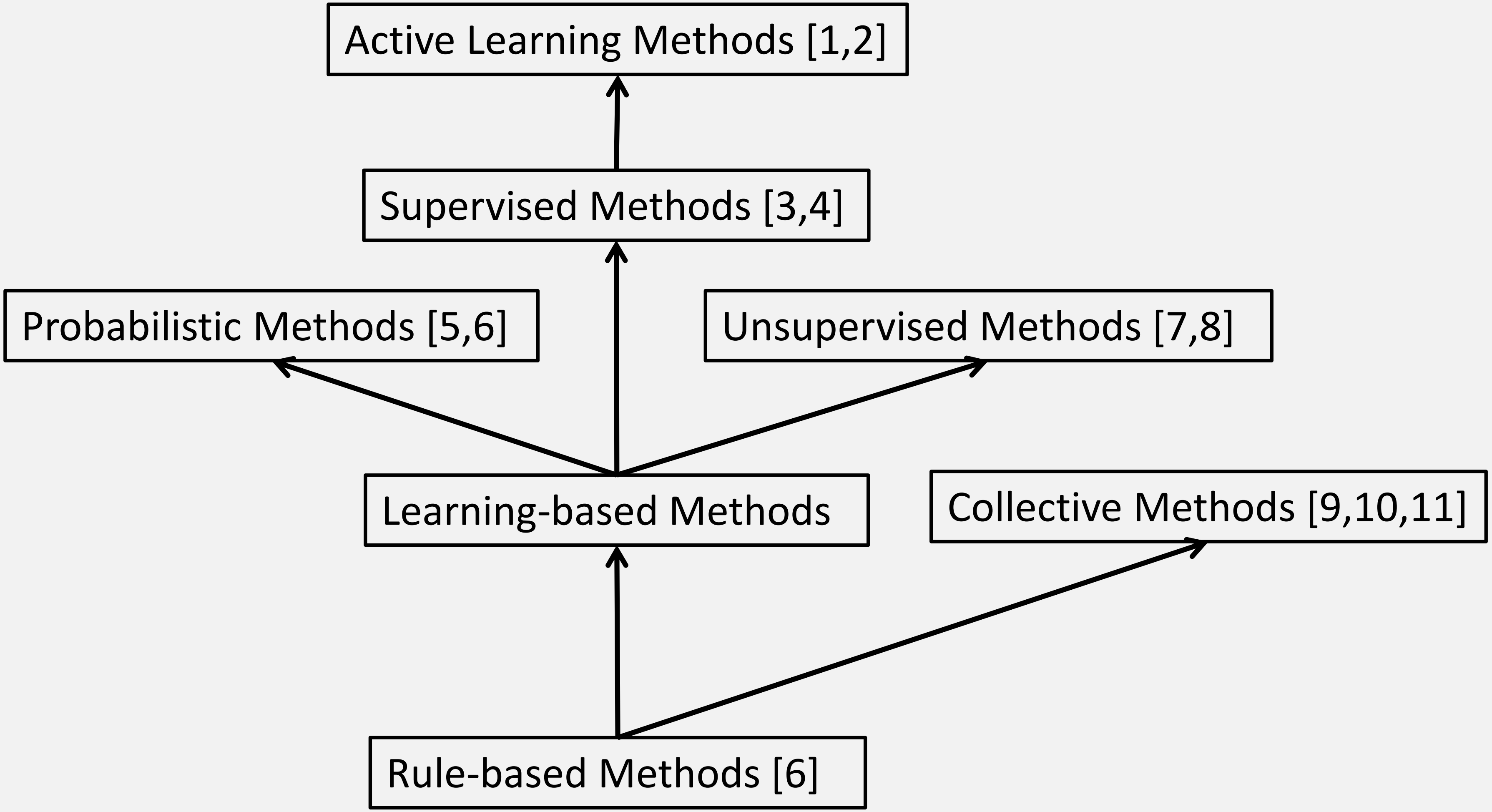


Step 3: Matching



- Estimates the similarity of candidate matches.
- Input
  - A set of blocks
    - Every distinct comparison in any block is a candidate match
- Output
  - Similarity Graph
    - Nodes → entities
    - Edges → candidate matches
    - Edge weights → matching likelihood (based on similarity score)

Evolution of Matching



Active Learning Methods [1,2]

Supervised Methods [3,4]

Probabilistic Methods [5,6]

Unsupervised Methods [7,8]

Learning-based Methods

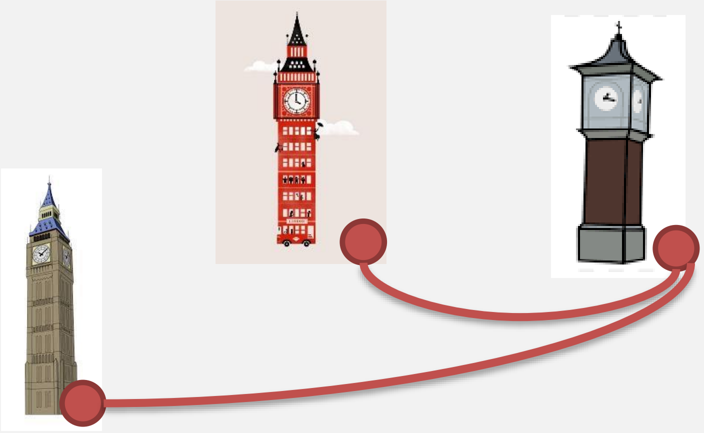
Collective Methods [9,10,11]

Rule-based Methods [6]

All are heavily based on string similarity measures [6].

Step 4: Clustering

- Partitions the matched pairs into equivalence clusters i.e., groups of entity profiles describing the same real-world object



- Input
  - Similarity Graph:
    - Nodes → entities
    - Edges → candidate matches
    - Edge weights → matching likelihood (based on similarity score)
- Output
  - Equivalence Clusters

Clustering Algorithms for Record Linkage

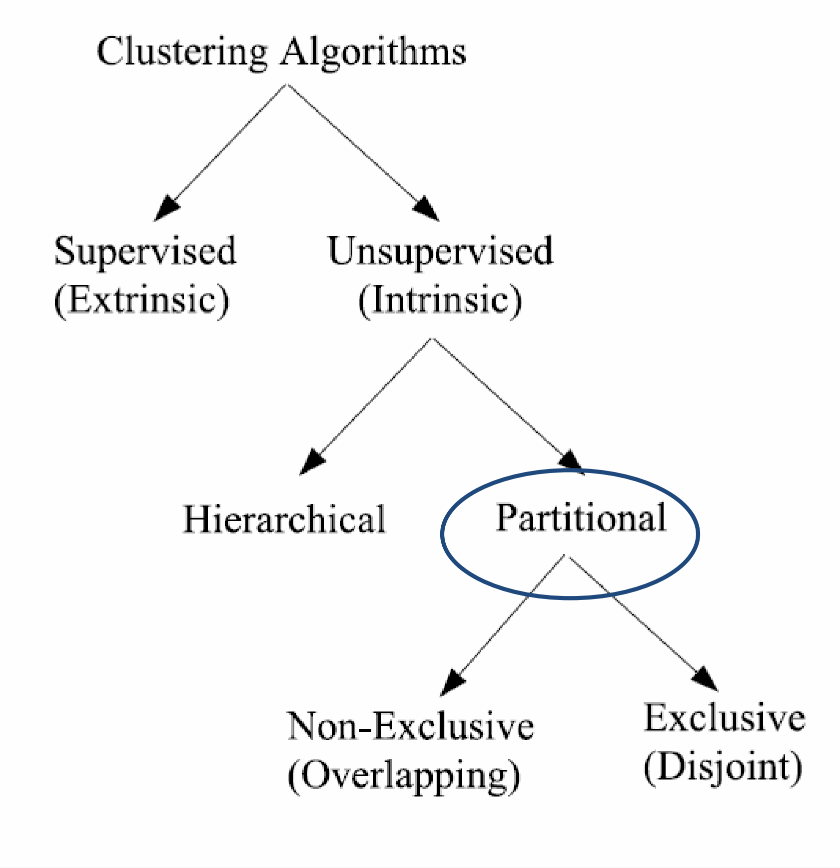
Relies on 1-1 constraint

- 1 entity from source dataset matches to 1 entity from target dataset

1. Unique Mapping Clustering [1][2]
   - Sorts all edges in decreasing weight
   - Starting from the top, each edge corresponds to a pair of duplicates **if**:
     - None of the adjacent entities has already been matched
     - predefined threshold < edge weight
2. Row-Column Clustering [3]
   - efficient approximation of the Hungarian Algorithm
3. Best Assignment Clustering [4]
   - efficient, heuristic solution to the assignment problem in unbalanced bipartite graphs

Clustering Algorithms for Deduplication

- A wealth of literature on clustering algorithms
- Requirements:
  - Partitional and disjoint Algorithms
    - Sometimes overlapping may be desirable
  - Goal: Sets of clusters that
    - maximize the intra-cluster weights
    - minimize the inter-cluster edge weights



Classification of clustering algorithms [6]

4gER (courtesy of Oktie Hassanzadeh)

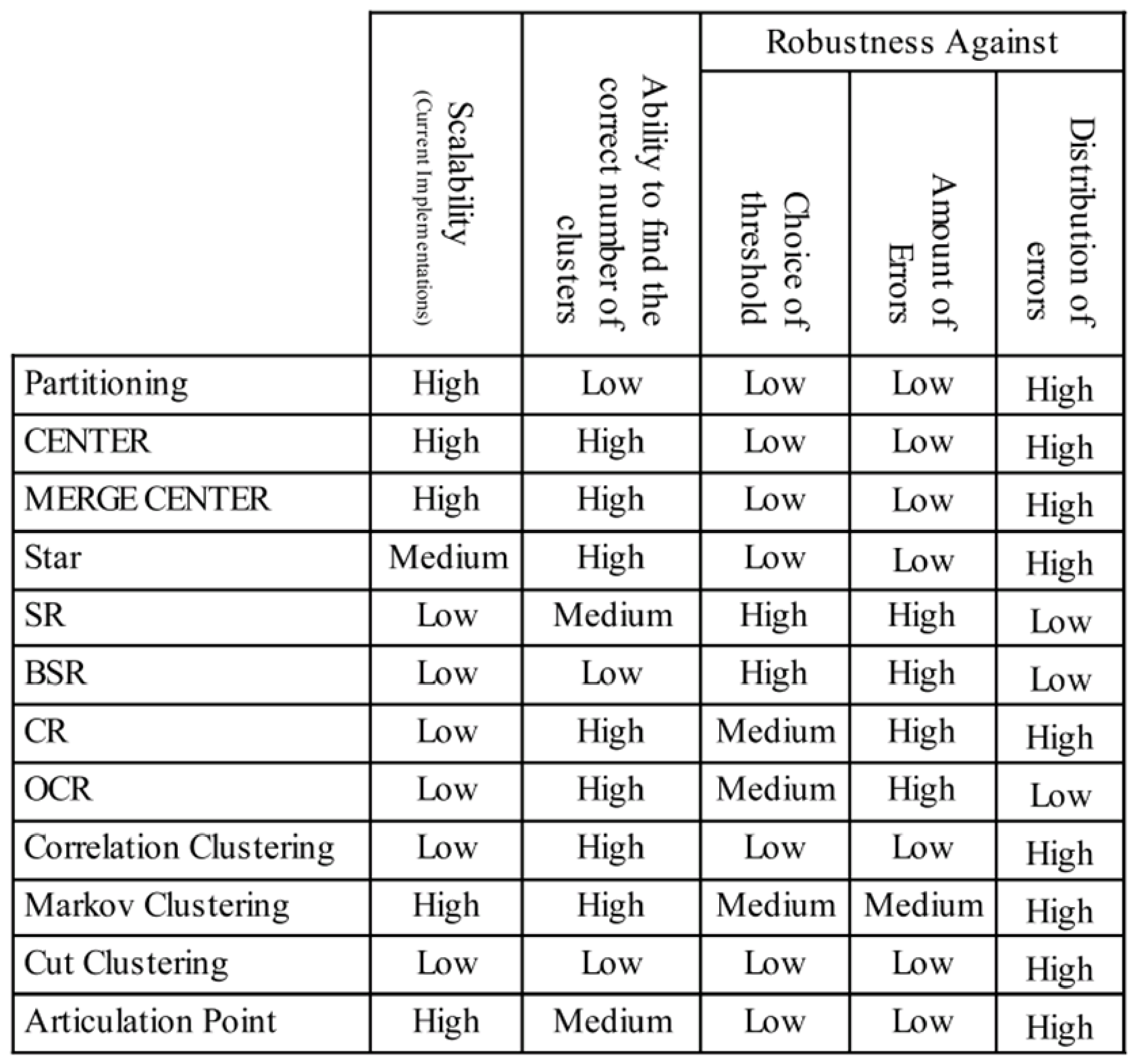
Dirty ER Clustering Algorithms Characteristics [3]

- Most important feature *"**Unconstrained algorithms**"*
  - Algorithms need to be able to *predict* the correct number of clusters
- Need to scale well
  - Time complexity < O(n^2)
- Need to be robust with respect to characteristics of the data
  - E.g., distribution of the duplicates
- Need to be capable of finding 'singleton' clusters
  - Different from many clustering algorithms
    - E.g., algorithms proposed for image segmentation

4gER

(courtesy of Oktie Hassanzadeh)

Summary of Experimental Results [3]



	Scalability (Current Implementations)	Ability to find the correct number of clusters	Robustness Against		
			Choice of threshold	Amount of Errors	Distribution of errors
Partitioning	High	Low	Low	Low	High
CENTER	High	High	Low	Low	High
MERGE CENTER	High	High	Low	Low	High
Star	Medium	High	Low	Low	High
SR	Low	Medium	High	High	Low
BSR	Low	Low	High	High	Low
CR	Low	High	Medium	High	High
OCR	Low	High	Medium	High	Low
Correlation Clustering	Low	High	Low	Low	High
Markov Clustering	High	High	Medium	Medium	High
Cut Clustering	Low	Low	Low	Low	High
Articulation Point	High	Medium	Low	Low	High

Schema Matching References

1. J. Madhavan, P. A. Bernstein, and E. Rahm. Generic schema matching with cupid. In VLDB, pages 49–58, 2001.
2. S. Melnik, H. Garcia-Molina, and E. Rahm. Similarity flooding: A versatile graph matching algorithm and its application to schema matching. In ICDE, pages 117–128, 2002.
3. H.-H. Do and E. Rahm. COMA: a system for flexible combination of schema matching approaches. In VLDB, pages 610–621, 2002.
4. M. Zhang, M. Hadjieleftheriou, B. C. Ooi, C. M. Procopiuc, D. Srivastava. Automatic discovery of attributes in relational databases. In SIGMOD, pages 109–120, 2011.
5. H. W. Kuhn. The hungarian method for the assignment problem. Naval research logistics quarterly, 2(1-2):83–97, 1955.
6. L. Ramshaw and R. E. Tarjan. On minimum-cost assignments in unbalanced bipartite graphs. HP Labs, Palo Alto, CA, USA, Tech. Rep. HPL-2012-40R1, 2012.

Blocking References – Part I

1. George Papadakis, Dimitrios Skoutas, Emmanouil Thanos, Themis Palpanas: A Survey of Blocking and Filtering Techniques for Entity Resolution. CoRR abs/1905.06167 (2019)
2. I. P. Fellegi and A. B. Sunter. A theory for record linkage. Journal of the American Statistical Association, 64(328):1183–1210, 1969.
3. A. N. Aizawa and K. Oyama. A fast linkage detection scheme for multi-source information integration. In WIRI, pages 30–39, 2005.
4. P. Christen. A survey of indexing techniques for scalable record linkage and deduplication. IEEE TKDE, 24(9):1537–1555, 2012.
5. T. de Vries, H. Ke, S. Chawla, and P. Christen. Robust record linkage blocking using suffix arrays. In CIKM, pages 305–314, 2009
6. R. Baxter, P. Christen, and T. Churches. A comparison of fast blocking methods for record linkage. In KDD Workshops, 2003.
7. B. Kenig and A. Gal. Mfiblocks: An effective blocking algorithm for entity resolution. Inf. Syst., 38(6):908–926, 2013.
8. M. A. Hernández and S. J. Stolfo. The merge/purge problem for large databases. In SIGMOD, pages 127–138, 1995.
9. S. Yan, D. Lee, M. Kan, and C. L. Giles. Adaptive sorted neighborhood methods for efficient record linkage. In JCDL, pages 185–194, 2007.

Blocking References – Part II

11. U. Draisbach, F. Naumann, S. Szott, and O. Wonneberg. Adaptive windows for duplicate detection. In ICDE, pages 1073–1083, 2012.
12. U. Draisbach and F. Naumann. A generalization of blocking and windowing algorithms for duplicate detection. In ICDKE, pages 18–24, 2011
13. M. Bilenko, B. Kamath, and R. J. Mooney. Adaptive blocking: Learning to scale up record linkage. In ICDM, pages 87–96, 2006
14. M. Michelson and C. A. Knoblock. Learning blocking schemes for record linkage. In AAAI, pages 440–445, 2006
15. A. D. Sarma, A. Jain, A. Machanavajjhala, and P. Bohannon. An automatic blocking mechanism for large-scale de-duplication tasks. In CIKM, pages 1055–1064, 2012.
16. M. Kejriwal and D. P. Miranker. An unsupervised algorithm for learning blocking schemes. In ICDM, pages 340–349, 2013.

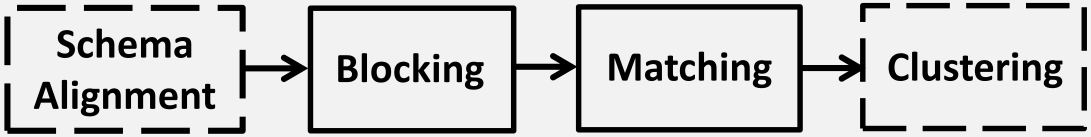
Matching References

1. K. Qian, L. Popa, P. Sen. Active Learning for Large-Scale Entity Resolution. CIKM 2017: 1379-1388
2. .J. Fisher, P. Christen, Q. Wangl. Active Learning Based Entity Resolution Using Markov Logic. PAKDD (2) 2016: 338-349
3. Reyes-Galaviz, O.F., Pedrycz, W., He, Z., Pizzi, N.J. A supervised gradient-based learning algorithm for optimized entity resolution. Data Knowl. Eng. 112, 106–129 (2017)
4. P. Christen. Automatic record linkage using seeded nearest neighbour and support vector machine classification. KDD 2008: 151-159.
5. A. Rasch, R. Schulze, W. Gorus, J. Hiller, S. Bartholomäus, S. Gorlatch. High-performance probabilistic record linkage via multi-dimensional homomorphisms. SAC 2019: 526-533.
6. A. K. Elmagarmid, P. G. Ipeirotis, V. S. Verykios. Duplicate Record Detection: A Survey. IEEE Trans. Knowl. Data Eng. 19(1): 1-16 (2007)
7. A. Jurek, J. Hong, Y. Chi, W. Liu. A novel ensemble learning approach to unsupervised record linkage. Inf. Syst. 71: 40-54 (2017)
8. A. Jurek, Deepak P. It Pays to Be Certain: Unsupervised Record Linkage via Ambiguity Minimization. PAKDD (3) 2018: 177-190.X
9. . Dong, A. Y. Halevy, J. Madhavan. Reference Reconciliation in Complex Information Spaces. SIGMOD Conference 2005: 85-96.O
10. . Benjelloun, H. Garcia-Molina, D. Menestrina, Q. Su, S. E. Whang, J. Widom. Swoosh: a generic approach to entity resolution. VLDB J. 18(1): 255-276 (2009).
11. I. Bhattacharya, L. Getoor. Collective entity resolution in relational data. TKDD 1(1): 5 (2007).

Clustering References

1. S. Lacoste-Julien, K. Palla, A. Davies, G. Kasneci, T. Graepel, Z. Ghahramani. SIGMa: simple greedy matching for aligning large knowledge bases. KDD 2013: 572-580
2. F. M. Suchanek, S. Abiteboul, P. Senellart. PARIS: Probabilistic Alignment of Relations, Instances, and Schema. PVLDB 5(3): 157-168 (2011)
3. O. Hassanzadeh, F. Chiang, R. J. Miller, H. C. Lee. Framework for Evaluating Clustering Algorithms in Duplicate Detection. PVLDB 2(1): 1282-1293 (2009)
4. H. W. Kuhn. The hungarian method for the assignment problem. Naval research logistics quarterly, 2(1-2):83–97, 1955.
5. L. Ramshaw and R. E. Tarjan. On minimum-cost assignments in unbalanced bipartite graphs. HP Labs, Palo Alto, CA, USA, Tech. Rep. HPL-2012-40R1, 2012.
6. A. Jain, R. Dubes, “Algorithms for Clustering Data”, Prentice Hall, 1988.

Generation 2: Tackling **Volume** and Veracity



- Same workflow as Generation 1
- Scope:
  - (tens of) millions of structured entity profiles
- Goals:
  - High accuracy despite noise
  - High time efficiency despite the size of data
- Assumptions:
  - Known schema → custom, schema-based solutions

Solution: Parallelization

Two types:

- Multi-core parallelization
  - Single system → shared memory
  - Distribute processing among available CPUs
- Massive parallelization
  - Cluster of independent systems
  - **Map-Reduce** paradigm [1]
    - Data partitioned across the nodes of a cluster
    - Fault-tolerant, optimized execution
    - Map Phase: transforms a data partition into (key, value) pairs
    - Reduce Phase: processes pairs with the same key

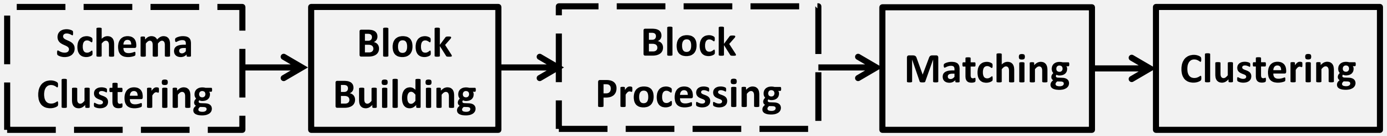
Parallelization Methods per Step

- Blocking
    - Dedoop [2]
    - MapReduce-based Sorted Neighborhood [3]
- Matching
    - Multi-core approaches [7][8]
    - MapReduce-based: Emphasis on **load balancing**
        - BlockSplit & PairRange [4][5]
        - Dis-Dedup [6]
        - Message-passing framework [9]
- Clustering
    - Fast Multi-source Entity Resolution (FAMER) framework [10][11]

Generation 2 References

1. J. Dean and S. Ghemawat. Mapreduce: simplified data processing on large clusters. Commun. ACM, 51(1):107–113, 2008.
2. L. Kolb, A. Thor, and E. Rahm. Dedoop: Efficient deduplication with hadoop. PVLDB, 5(12):1878–1881, 2012.
3. L. Kolb, A. Thor, and E. Rahm. Multi-pass sorted neighborhood blocking with mapreduce. Computer Science - R&D, 27(1):45–63, 2012.
4. L. Kolb, A. Thor, and E. Rahm. Load balancing for mapreduce-based entity resolution. In ICDE, pages 618–629, 2012.
5. W. Yan, Y. Xue, and B. Malin. Scalable load balancing for mapreduce-based record linkage. In IPCCC, pages 1–10, 2013.
6. X. Chu, I. F. Ilyas, and P. Koutris. Distributed data deduplication. PVLDB, 9(11):864–875, 2016.
7. O. Benjelloun, H. Garcia-Molina, H. Gong, H. Kawai, T. E. Larson, D. Menestrina, and S. Thavisomboon. D-swoosh: A family of algorithms for generic, distributed entity resolution. In ICDCS, page 37, 2007.
8. Hung-sik Kim and Dongwon Lee. Parallel linkage. In CIKM, pages 283–292, 2007.
9. V. Rastogi, N. N. Dalvi, and M. N. Garofalakis. Large-scale collective entity matching. PVLDB, 4(4):208–218, 2011.
10. A. Saeedi, E. Peukert, and E. Rahm. Comparative evaluation of distributed clustering schemes for multi-source entity resolution. In ADBIS, pages 278–293, 2017.
11. A. Saeedi, M. Nentwig, E. Peukert, and E. Rahm. Scalable matching and clustering of entities with FAMER. CSIMQ, 16:61–83, 2018.

G3: Tackling **Variety**, Volume and Veracity



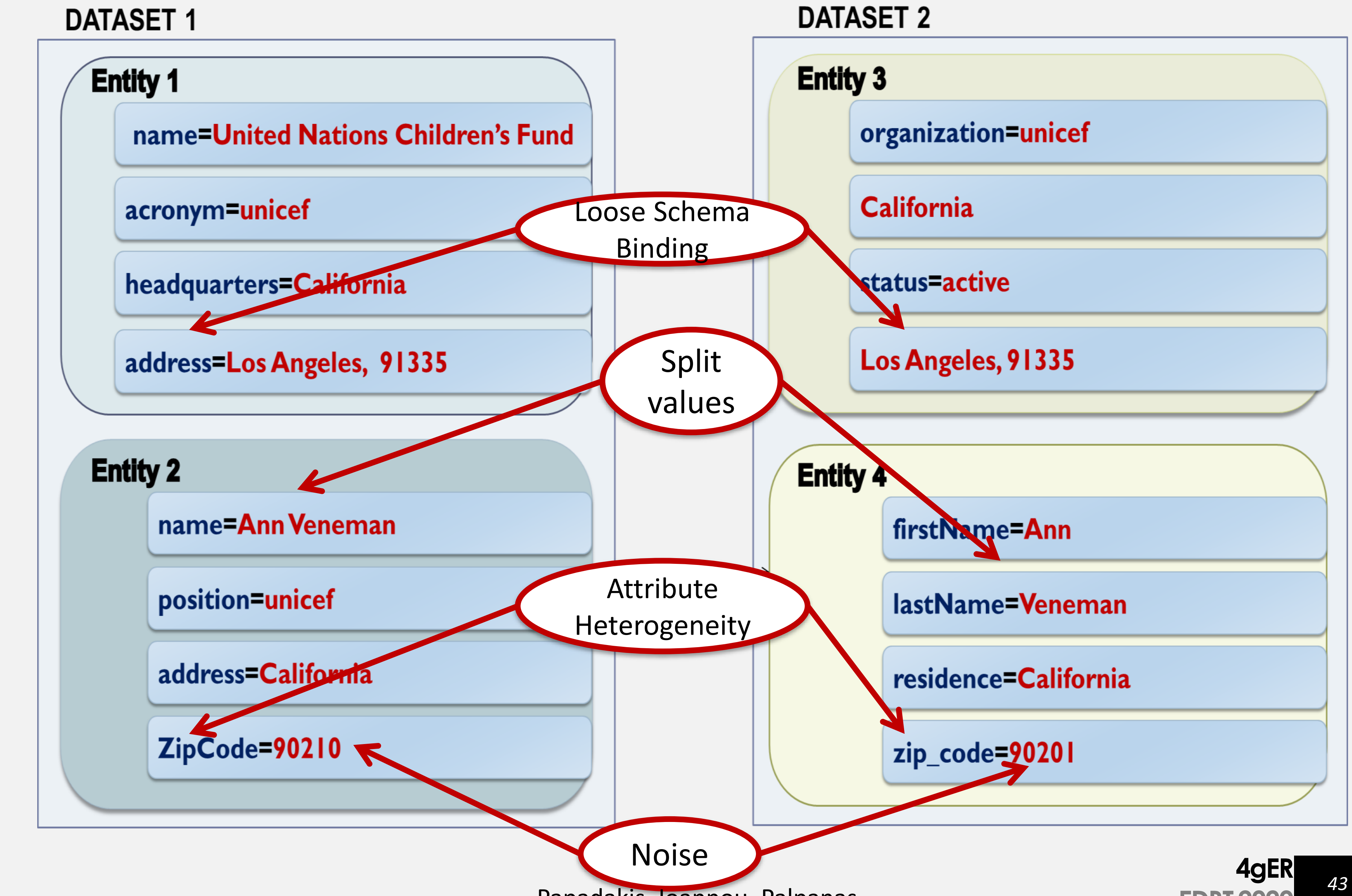
- Scope:
  - User-generated Web Data

    Voluminous, (semi-)structured datasets.
    - BTC09: 1.15 billion triples, 182 million entities.

    Users are free to add attribute values and/or attribute names → unprecedented levels of schema heterogeneity.
    - Google Base: 100,000 schemata for 10,000 entity types
    - BTC09: 136,000 attribute names

    Several datasets produced by automatic information extraction techniques → noise, tag-style values.

Example of Web Data



DATASET 1

Entity 1

name=United Nations Children's Fund

acronym=unicef

headquarters=California

address=Los Angeles, 91335

Entity 2

name=Ann Veneman

position=unicef

address=California

ZipCode=90210

DATASET 2

Entity 3

organization=unicef

California

status=active

Los Angeles, 91335

Entity 4

firstName=Ann

lastName=Veneman

residence=California

zip_code=90201

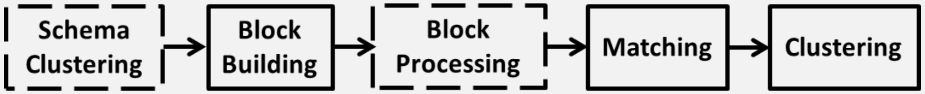
Loose Schema Binding

Split values

Attribute Heterogeneity

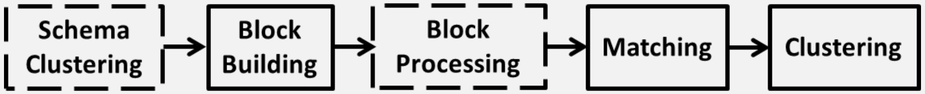
Noise

Schema Clustering



- Schema Matching → not applicable
- Instead, partition attributes according to their syntactic similarity, regardless of their semantic relation
- Goal:
  - Facilitate next steps
- Scope:
  - Both Clean-Clean and Dirty ER
- Attribute Clustering [1][2][3]
  - Create a graph, where every node represents an attribute
  - For each attribute name/node n_i
    - Find the most similar node n_j
    - If **$sim(n_i,n_j) > 0$**, add an edge $<n_i,n_j>$
  - Extract connected components
  - Put all singleton nodes in a **"glue" cluster**

Block Building



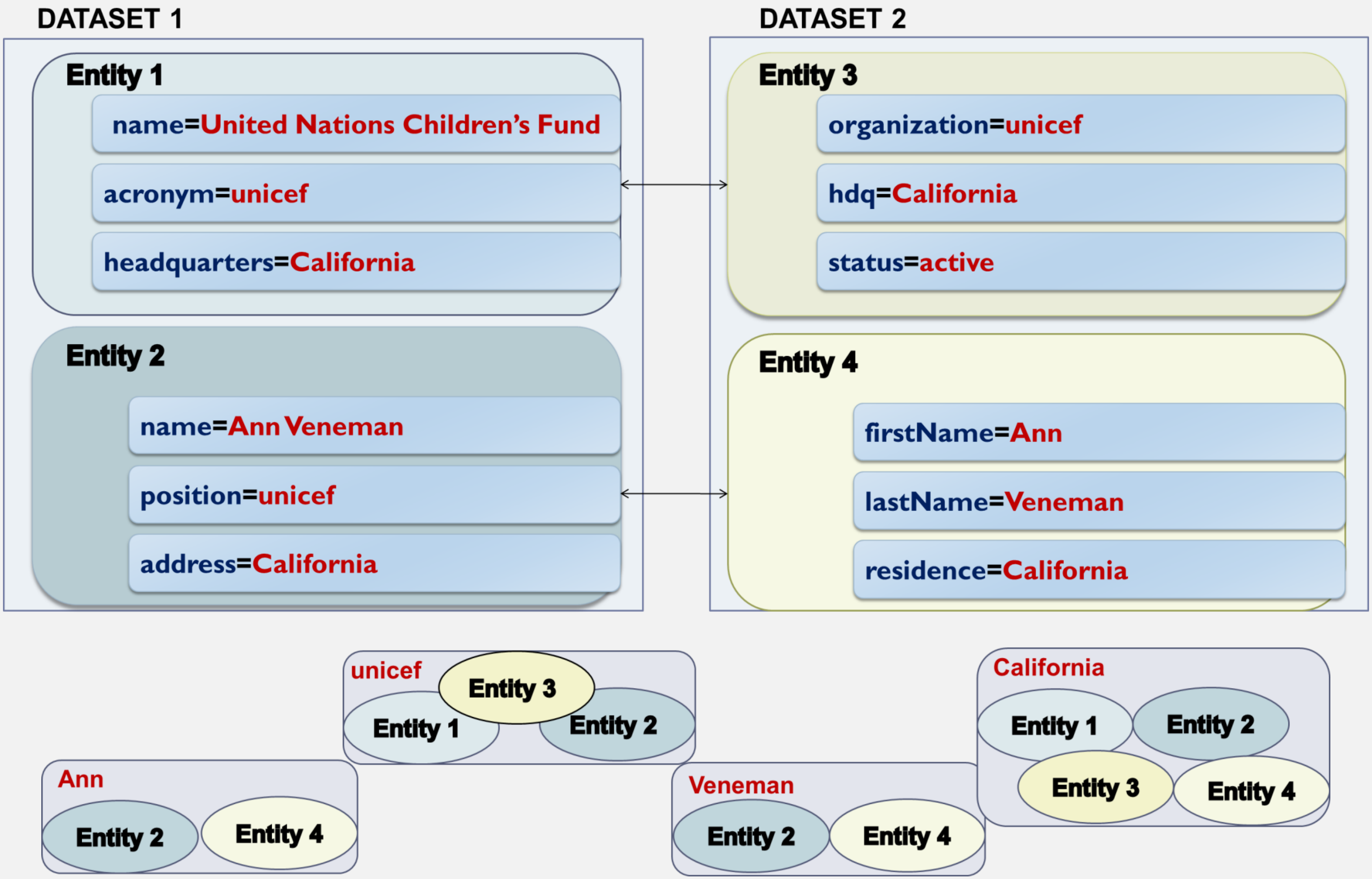
- Unlike Blocking in G1/G2, it considers all attribute values and completely ignores all attribute names

  → **schema-agnostic functionality**

- Core approach: Token Blocking [1]
  1. Given an entity profile, extract all tokens that are contained in its attribute values.
  2. Create one block for every distinct token with frequency > 2 → each block contains all entities with the corresponding token.

     Pros:
     - Parameter-free
     - Efficient
     - Unsupervised

Example of Token Blocking



DATASET 1

Entity 1
- name=United Nations Children's Fund
- acronym=unicef
- headquarters=California

Entity 2
- name=Ann Veneman
- position=unicef
- address=California

DATASET 2

Entity 3
- organization=unicef
- hdq=California
- status=active

Entity 4
- firstName=Ann
- lastName=Veneman
- residence=California

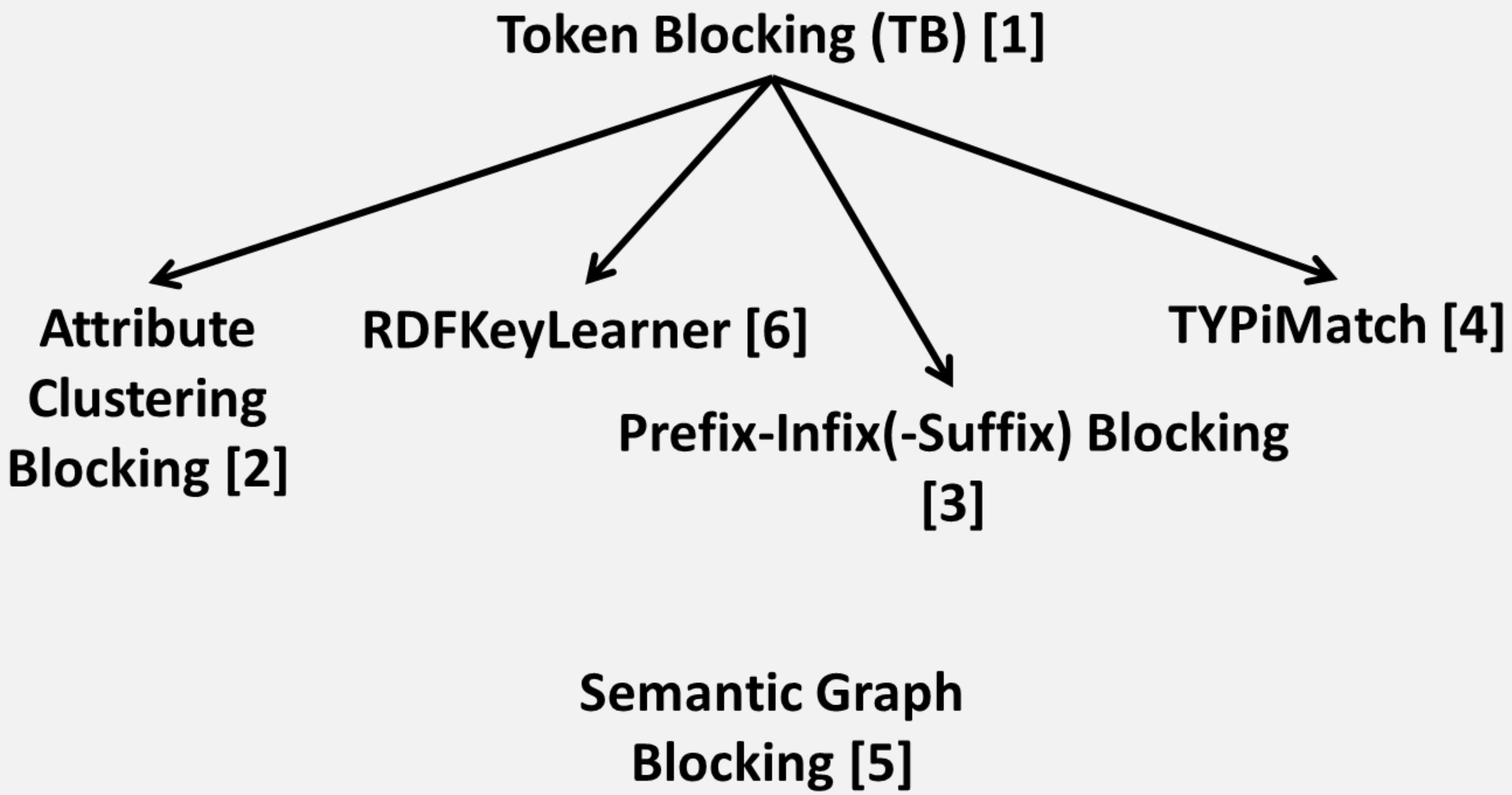
unicef: Entity 3, Entity 1, Entity 2

California: Entity 1, Entity 2, Entity 3, Entity 4

Ann: Entity 2, Entity 4

Veneman: Entity 2, Entity 4

Genealogy of Block Building Techniques [8]



Token Blocking (TB) [1]

Attribute Clustering Blocking [2]

RDFKeyLearner [6]

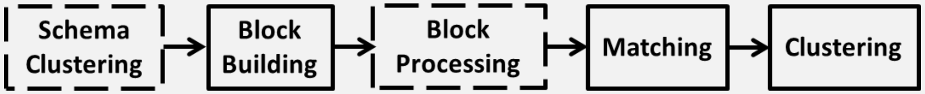
Prefix-Infix(-Suffix) Blocking [3]

TYPiMatch [4]

Semantic Graph Blocking [5]

MapReduce-based parallelizations in [7]

Block Processing



- High Recall due to redundancy
- Low Precision due to:
  1. the blocks are overlapping → **redundant comparisons**
  2. high number of comparisons between irrelevant entities → **superfluous comparisons**

Solution:

restructure the original blocks so as to increase precision at no significant cost in recall

Block Processing Techniques

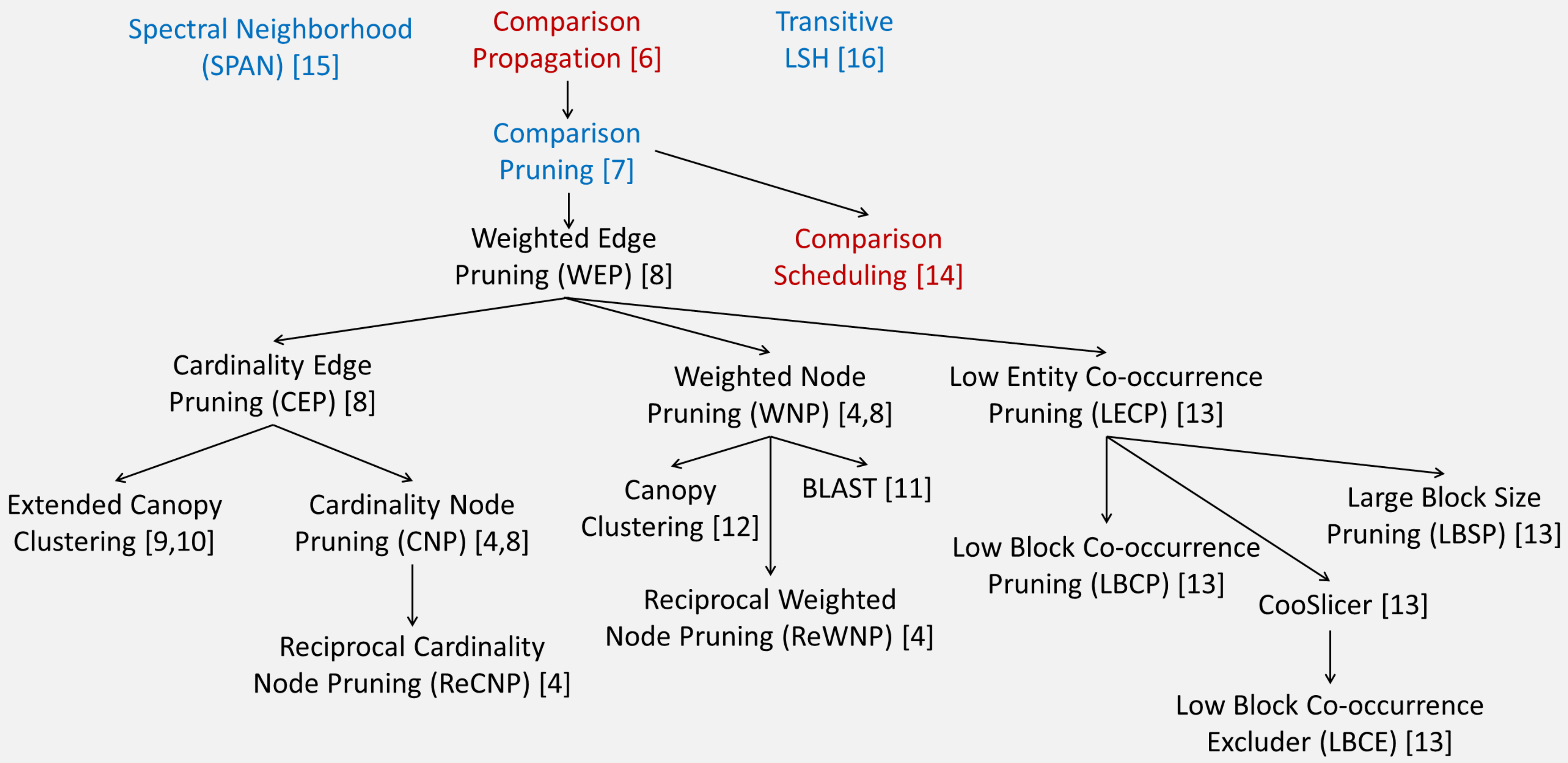
Generic approach

- Assign a matching likelihood score to each item
- Discard items with low costs

Block-centric methods

- Block Purging [1,2,3]
- Block Filtering [4]
- Block Clustering [5]

Comparison Cleaning Methods [17]



Spectral Neighborhood (SPAN) [15]

Comparison Propagation [6]

Transitive LSH [16]

Comparison Pruning [7]

Weighted Edge Pruning (WEP) [8]

Comparison Scheduling [14]

Cardinality Edge Pruning (CEP) [8]

Weighted Node Pruning (WNP) [4,8]

Low Entity Co-occurrence Pruning (LECP) [13]

Extended Canopy Clustering [9,10]

Cardinality Node Pruning (CNP) [4,8]

Canopy Clustering [12]

BLAST [11]

Large Block Size Pruning (LBSP) [13]

Low Block Co-occurrence Pruning (LBCP) [13]

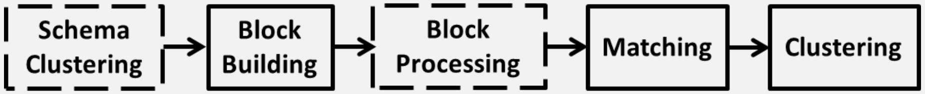
CooSlicer [13]

Reciprocal Weighted Node Pruning (ReWNP) [4]

Reciprocal Cardinality Node Pruning (ReCNP) [4]

Low Block Co-occurrence Excluder (LBCE) [13]

Entity Matching



- Collective approaches to tackle Variety
- Most methods crafted for Clean-Clean ER
- General outline of SiGMa [1], PARIS [2], LINDA [3], RiMOM-IM [4,5]
  - Bootstrap with a few reliable seed matches.
  - Using value and neighbor similarity, propagate initial matches to neighbors.
  - Order candidate matches in descending overall similarity
  - Iteratively mark the top pair as a match if it satisfies a constraint
  - Recompute the similarity of the neighbors
  - Update candidate matches order
- MinoanER [6] performs a specific number of steps, rather than iterating until convergence

Entity Clustering

- Methods of G1 & G2 are still applicable
  - Only difference: similarity scores extracted in a schema-agnostic fashion, not from specific attributes

- SplitMerge [1]
  - inherently capable of handling heterogeneous semantic types

[1] M. Nentwig, A. Groß, and E. Rahm. Holistic entity clustering for linked data. In ICDM Workshops, pages 194–201, 2016.

Schema Clustering References

1. G. Papadakis, E. Ioannou, T. Palpanas, C. Niederée, W. Nejdl. A Blocking Framework for Entity Resolution in Highly Heterogeneous Information Spaces. IEEE Trans. Knowl. Data Eng. 25(12): 2665-2682 (2013)
2. G. Simonini, S. Bergamaschi, H. V. Jagadish. BLAST: a Loosely Schema-aware Meta-blocking Approach for Entity Resolution. PVLDB 9(12): 1173-1184 (2016)
3. G. Papadakis, L. Tsekouras, E. Thanos, G. Giannakopoulos, T. Palpanas, M. Koubarakis. The return of JedAI: End-to-End Entity Resolution for Structured and Semi-Structured Data. PVLDB 11(12): 1950-1953 (2018)

Block Building References

1. G. Papadakis, E. Ioannou, C. Niederée, P. Fankhauser. Efficient entity resolution for large heterogeneous information spaces. WSDM 2011: 535-544
2. G. Papadakis, E. Ioannou, T. Palpanas, C. Niederée, W. Nejdl. A Blocking Framework for Entity Resolution in Highly Heterogeneous Information Spaces. IEEE Trans. Knowl. Data Eng. 25(12): 2665-2682 (2013)
3. G. Papadakis, E. Ioannou, C. Niederée, T. Palpanas, W. Nejdl. Beyond 100 million entities: large-scale blocking-based resolution for heterogeneous data. WSDM 2012: 53-62
4. Y. Ma, T. Tran. TYPiMatch: type-specific unsupervised learning of keys and key values for heterogeneous web data integration. WSDM 2013: 325-334
5. J. Nin, V. Muntés-Mulero, N. Martínez-Bazan, and J. Larriba-Pey. On the use of semantic blocking techniques for data cleansing and integration. In IDEAS, pages 190–198, 2007.
6. D. Song and J. Heflin. Automatically generating data linkages using a domain-independent candidate selection approach. In ISWC, pages 649–664, 2011.
7. V. Christophides, V. Efthymiou, K. Stefanidis. Entity Resolution in the Web of Data. Synthesis Lectures on the Semantic Web: Theory and Technology, Morgan & Claypool Publishers 2015.
8. George Papadakis, Dimitrios Skoutas, Emmanouil Thanos, Themis Palpanas: A Survey of Blocking and Filtering Techniques for Entity Resolution. CoRR abs/1905.06167 (2019)

Block Processing References – Part I

1. G. Papadakis, E. Ioannou, C. Niederée, P. Fankhauser. Efficient entity resolution for large heterogeneous information spaces. WSDM 2011: 535-544
2. G. Papadakis, E. Ioannou, T. Palpanas, C. Niederée, W. Nejdl. A Blocking Framework for Entity Resolution in Highly Heterogeneous Information Spaces. IEEE Trans. Knowl. Data Eng. 25(12): 2665-2682 (2013)
3. G. Papadakis, E. Ioannou, C. Niederée, T. Palpanas, W. Nejdl. Beyond 100 million entities: large-scale blocking-based resolution for heterogeneous data. WSDM 2012: 53-62
4. G. Papadakis, G. Papastefanatos, T. Palpanas, M. Koubarakis. Scaling Entity Resolution to Large, Heterogeneous Data with Enhanced Meta-blocking. EDBT 2016: 221-232
5. J. Fisher, P. Christen, Q. Wang, E. Rahm. A Clustering-Based Framework to Control Block Sizes for Entity Resolution. KDD 2015: 279-288
6. G. Papadakis, E. Ioannou, C. Niederée, T. Palpanas, W. Nejdl. Eliminating the redundancy in blocking-based entity resolution methods. JCDL 2011: 85-94.
7. G. Papadakis, E. Ioannou, C. Niederée, T. Palpanas, W. Nejdl. To compare or not to compare: making entity resolution more efficient. SWIM 2011: 3.
8. G. Papadakis, G. Koutrika, T. Palpanas, W. Nejdl. Meta-Blocking: Taking Entity Resolutionto the Next Level. IEEE Trans. Knowl. Data Eng. 26(8): 1946-1960 (2014).
9. P. Christen. A Survey of Indexing Techniques for Scalable Record Linkage and Deduplication. IEEE Trans. Knowl. Data Eng. 24(9): 1537-1555 (2012).
10. G. Papadakis, G. Alexiou, G. Papastefanatos, G. Koutrika. Schema-agnostic vs Schema-based Configurations for Blocking Methods on Homogeneous Data. PVLDB 9(4): 312-323 (2015).

Block Processing References – Part II

11. G. Simonini, S. Bergamaschi, H. V. Jagadish. BLAST: a Loosely Schema-aware Meta-blocking Approach for Entity Resolution. PVLDB 9(12): 1173-1184 (2016)
12. A. McCallum, K. Nigam, L. H. Ungar. Efficient clustering of high-dimensional data sets with application to reference matching. KDD 2000: 169-178.
13. D. C. Nascimento, C. E. S. Pires, and D. G. Mestre. Exploiting block co-occurrence to control block sizes for entity resolution. Knowledge and Information Systems, pages 1–42, 2019.
14. G. Papadakis, E. Ioannou, T. Palpanas, C. Niederée, and W. Nejdl. A blocking framework for entity resolution in highly heterogeneous information spaces. IEEE TKDE, 25(12):2665–2682, 2013.
15. L. Shu, A. Chen, M. Xiong, and W. Meng. Efficient spectral neighborhood blocking for entity resolution. In ICDE, pages 1067–1078, 2011.
16. R. C. Steorts, S. L. Ventura, M. Sadinle, and S. E. Fienberg. A comparison of blocking methods for record linkage. In Privacy in Statistical Databases, pages 253–268, 2014.
17. George Papadakis, Dimitrios Skoutas, Emmanouil Thanos, Themis Palpanas: A Survey of Blocking and Filtering Techniques for Entity Resolution. CoRR abs/1905.06167 (2019)

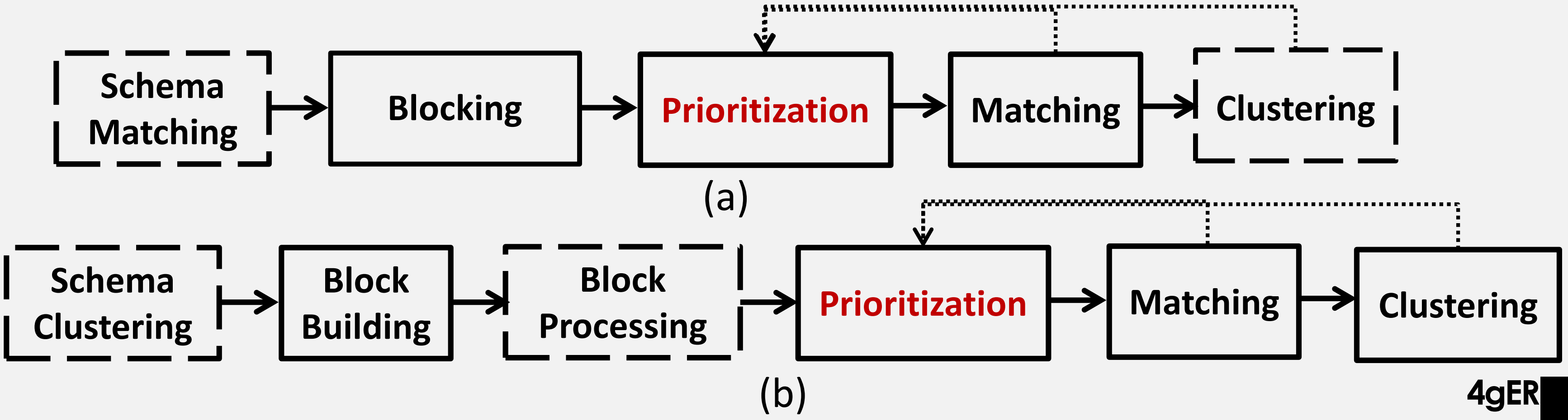
Entity Matching References

1. S. Lacoste-Julien, K. Palla, A. Davies, G. Kasneci, T. Graepel, Z. Ghahramani. SIGMa: simple greedy matching for aligning large knowledge bases. KDD 2013: 572-580
2. F. M. Suchanek, S. Abiteboul, P. Senellart. PARIS: Probabilistic Alignment of Relations, Instances, and Schema. PVLDB 5(3): 157-168 (2011)
3. C. Böhm, G. de Melo, F. Naumann, and G. Weikum. LINDA: distributed web-of-data-scale entity matching. In CIKM, pages 2104–2108, 2012.
4. J. Li, J. Tang, Y. Li, and Q. Luo. Rimom: A dynamic multistrategy ontology alignment framework. TKDE, 21(8):1218–1232, 2009.
5. C. Shao, L. Hu, J. Li, Z. Wang, T. L. Chung, and J.-B. Xia. Rimom-im: A novel iterative framework for instance matching. J. Comput. Sci. Technol., 31(1):185–197, 2016.
6. V. Efthymiou, G. Papadakis, K. Stefanidis, and V. Christophides. MinoanER: Schema-agnostic, non-iterative, massively parallel resolution of web entities. In EDBT, pages 373–384, 2019.

G4: Tackling **Velocity**, Variety, Volume and Veracity

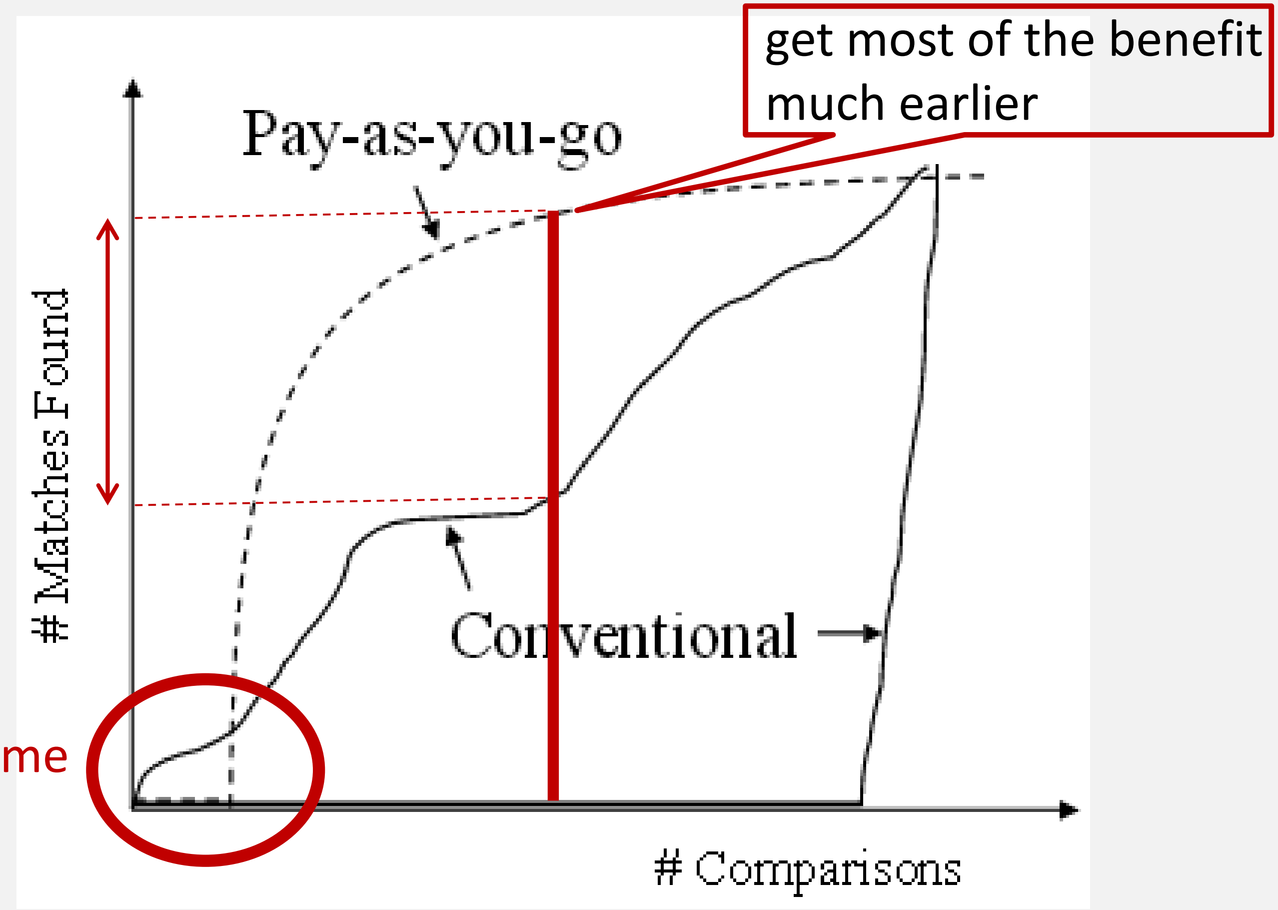
- Scope:
  - Applications with increasing data volume and time constraints
    - Loose ones (e.g., minutes, hours) → Progressive ER
    - Strict ones (i.e., seconds) → Real-time (On-line) ER

- End-to-end workflows for Progressive ER



Progressive Entity Resolution

Unprecedented, increasing volume of data → applications requiring partial solutions to produce useful results

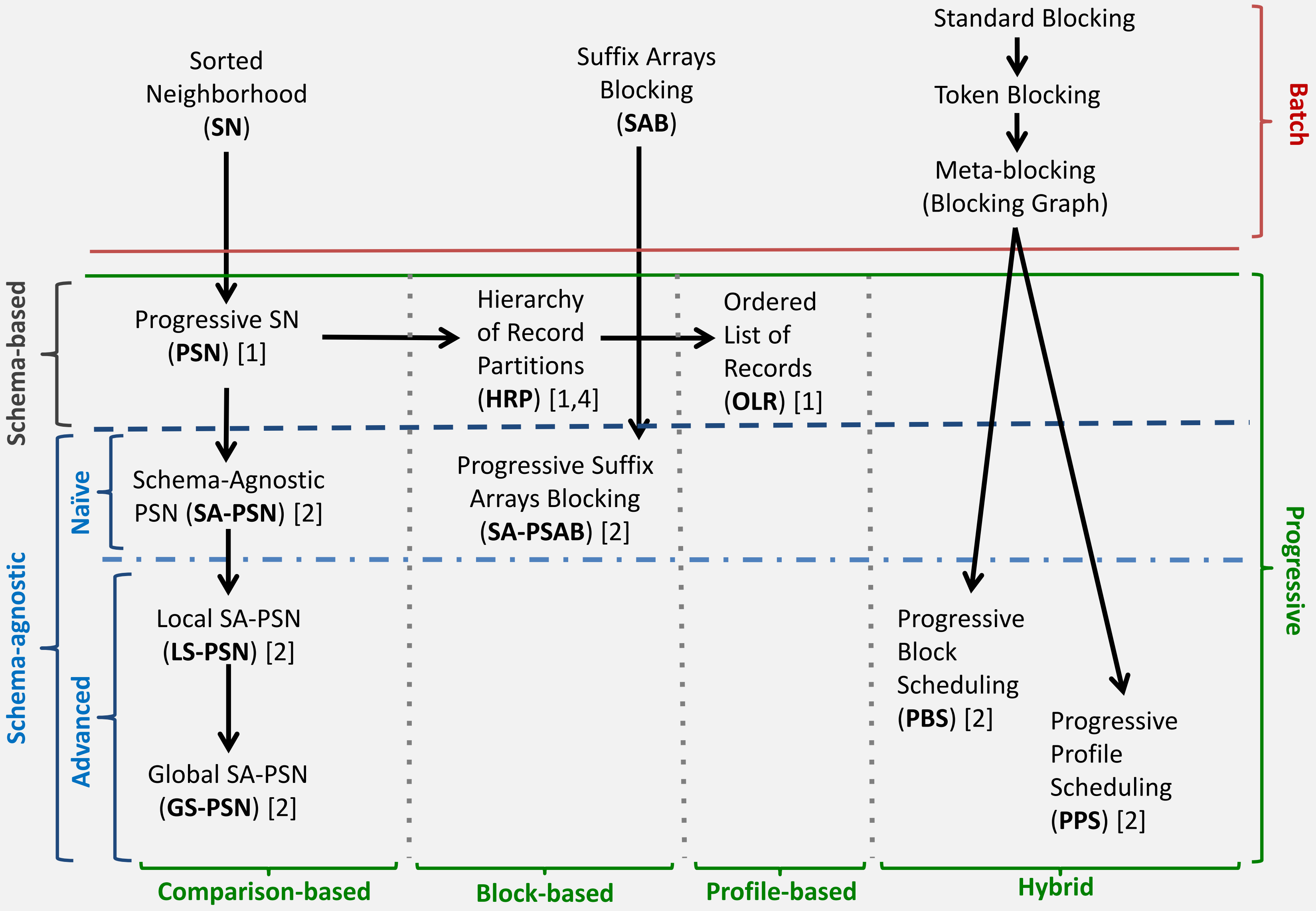


Outline Progressive ER

- Requires:
  - Improved Early Quality
  - Same Eventual Quality

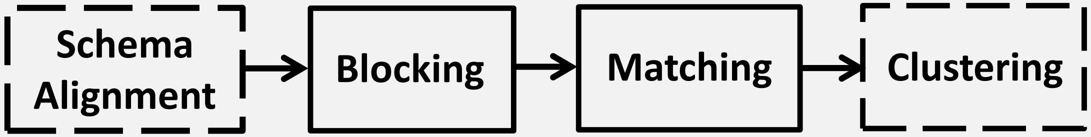
- Prioritization
  - Defines **optimal processing order** for a set of entities
  - Static Methods [1,2]:
    - Guide which records to compare first, **independently** of Entity Matching results
  - Dynamic Methods [3]:
    - If $c_{i,j}$ is a duplicate, then check $c_{i+1,j}$ and $c_{i,j+1}$ as well.
    - Assumption:
      - Oracle for Entity Matching

Taxonomy of Static Prioritization Methods



Real-time Entity Resolution

Same workflow as Generations 1 and 2:


Schema Alignment
Blocking
Matching
Clustering

Same scope (so far):

- Structured data

Different input:

- stream of query entity profiles

Different goal:

- resolve each query over a large dataset in the shorted possible time (& with the minimum memory footprint)

Techniques per workflow step

Incremental Blocking

- **DySimII** [1] - extends Standard Blocking
- **F-DySNI** [2,3] - extends Sorted Neighborhood
- **(S)BlockSketch** [4] - bounded matching time, constant memory footprint

Incremental Matching

- **QDA** [5] - SQL-like selection queries over a single dataset
- **QuERy** [6] - complex join queries over multiple, overlapping, dirty DSs
- **EAQP** [7] - queries under data
- Evolving matching rules [8]

Incremental Clustering

- Incremental Correlation Clustering [9]

Progressive ER References

1. S. E. Whang, D. Marmaros, and H. Garcia-Molina. Pay-as-you-go entity resolution. TKDE, 25(5):1111–1124, 2013.
2. T. Papenbrock, A. Heise, and F. Naumann. Progressive duplicate detection. TKDE, 27(5):1316–1329, 2015.
3. G. Simonini, G. Papadakis, T. Palpanas, S. Bergamaschi. Schema-Agnostic Progressive Entity Resolution. IEEE Trans. Knowl. Data Eng. 31(6): 1208-1221 (2019)
4. Y. Altowim and S. Mehrotra. Parallel progressive approach to entity resolution using mapreduce. In ICDE, pages 909–920, 2017.

Incremental ER References

1. B. Ramadan and P. Christen, H. Liang, and R. W. Gayler, and D. Hawking. Dynamic similarity-aware inverted indexing for real-time entity resolution. In PAKDD Workshops, pages 47–58, 2013.
2. B. Ramadan and P. Christen. Forest-based dynamic sorted neighborhood indexing for real-time entity resolution. In CIKM, pages 1787–1790, 2014.
3. B. Ramadan and P. Christen, H. Liang, and R. W. Gayler. Dynamic sorted neighborhood indexing for real-time entity resolution. J. Data and Information Quality, 6(4):15:1–15:29, 2015.
4. D. Karapiperis, A. Gkoulalas-Divanis, V. S. Verykios. Summarization Algorithms for Record Linkage. EDBT 2018: 73-84.
5. H. Altwaijry, D. V. Kalashnikov, and S. Mehrotra. QDA: A query-driven approach to entity resolution. TKDE, 29(2):402–417, 2017.
6. H. Altwaijry, S. Mehrotra, and D. V. Kalashnikov. Query: A framework for integrating entity resolution with query processing. PVLDB, 9(3):120–131, 2015.
7. E. Ioannou, W. Nejdl, C. Niederée, and Y. Velegrakis. On-the-fly entity-aware query processing in the presence of linkage. PVLDB, 3(1): 429–438, 2010.
8. S. E. Whang and H. Garcia-Molina. Entity resolution with evolving rules. PVLDB, 3(1):1326–1337, 2010.
9. A. Gruenheid, X. L. Dong, and D. Srivastava. Incremental record linkage. Proc. VLDB Endow., 7(9):697–708, May 2014. ISSN 2150-8097.

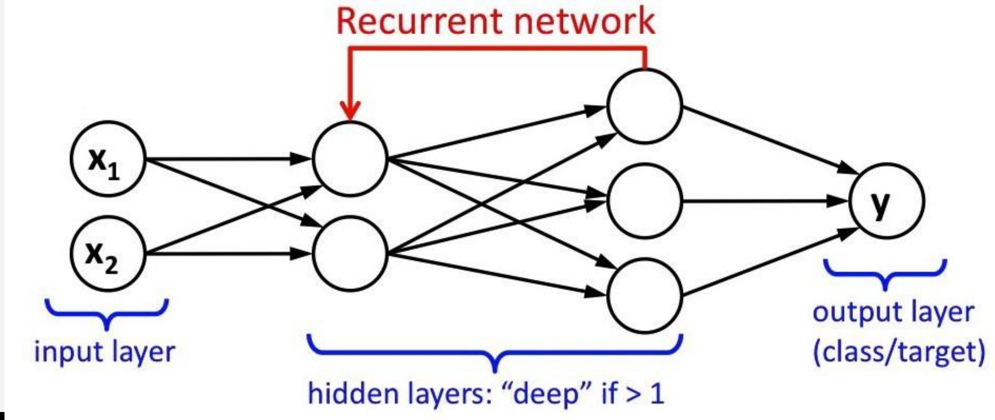
- Introduction
- Four Generations

Part C – Entity Resolution Revisited: Leveraging External Knowledge

  - **Deep Learning for Entity Resolution**
  - Crowd-sourced Entity Resolution
- Challenges and Final Remarks

Deep Learning

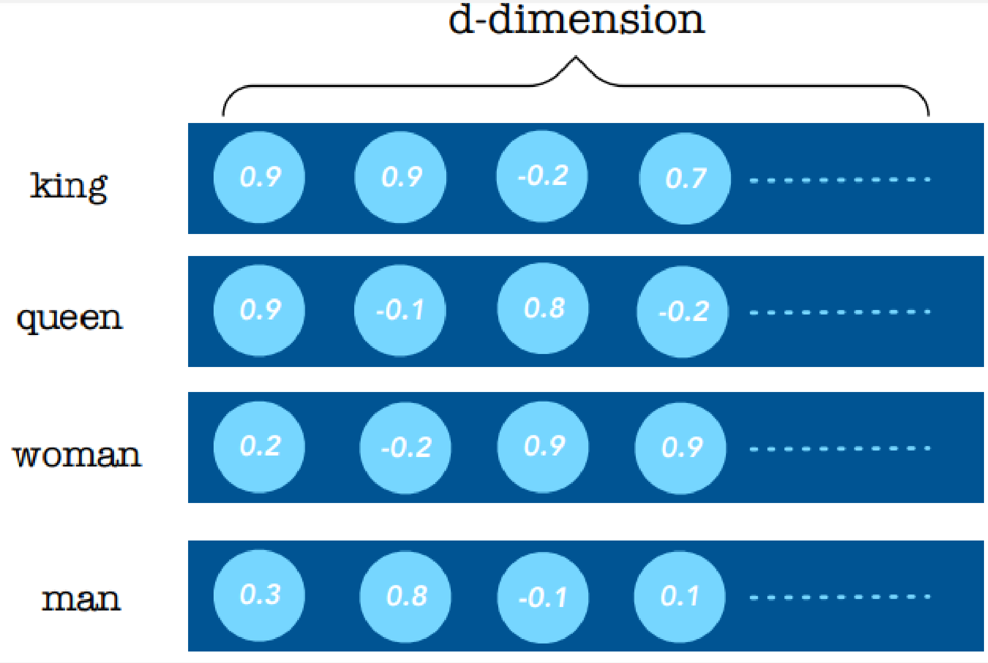
- Specific class of Machine Learning / Data Mining
- Teaches computers to do what comes naturally to humans: learn by example
- Goal: learn a complicated function from the data
- Ideal for complex tasks involving multi-dimensional data
- Has transformed many fields, e.g., computer vision, speech recognition, natural language processing, etc.
  - Similar performance, or even better, to human expert performance
- Details in [1]



Embeddings

- Based on the distributional hypothesis

  i.e., words appearing in the same context share meaning
- Each word is represented as a distribution of weights (positive or negative) across specific dimensions
- Goal: capture semantic string similarities
- Popular embeddings pre-trained over huge corpora:
  - Word2Vec [5]
  - Glove [6]
  - fastText [7]



Deep Learning for Schema Matching

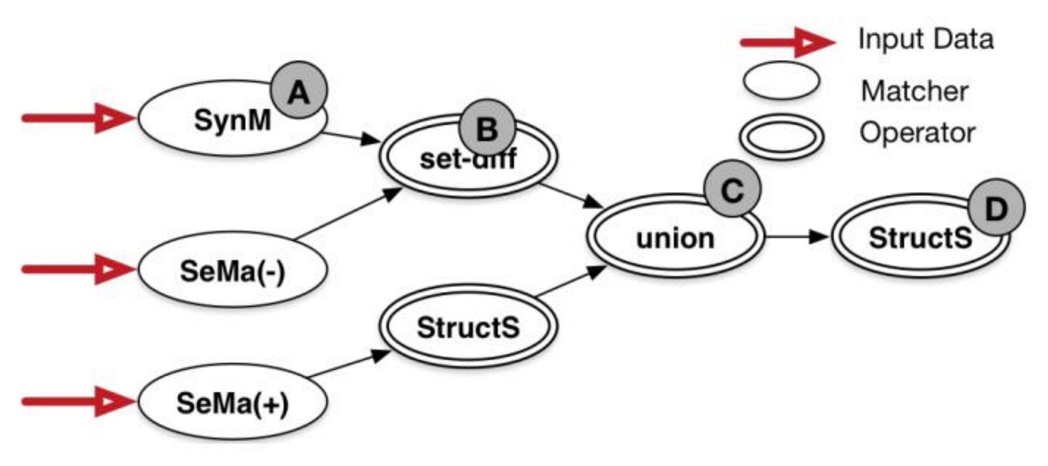
SEMPROP [2]

Two types of matchers:

1. Semantic Matcher (SeMa) based on **Coherent Groups**

if the average cosine similarities between all vectors in X > δ → SeMa(+), otherwise SeMa(-)

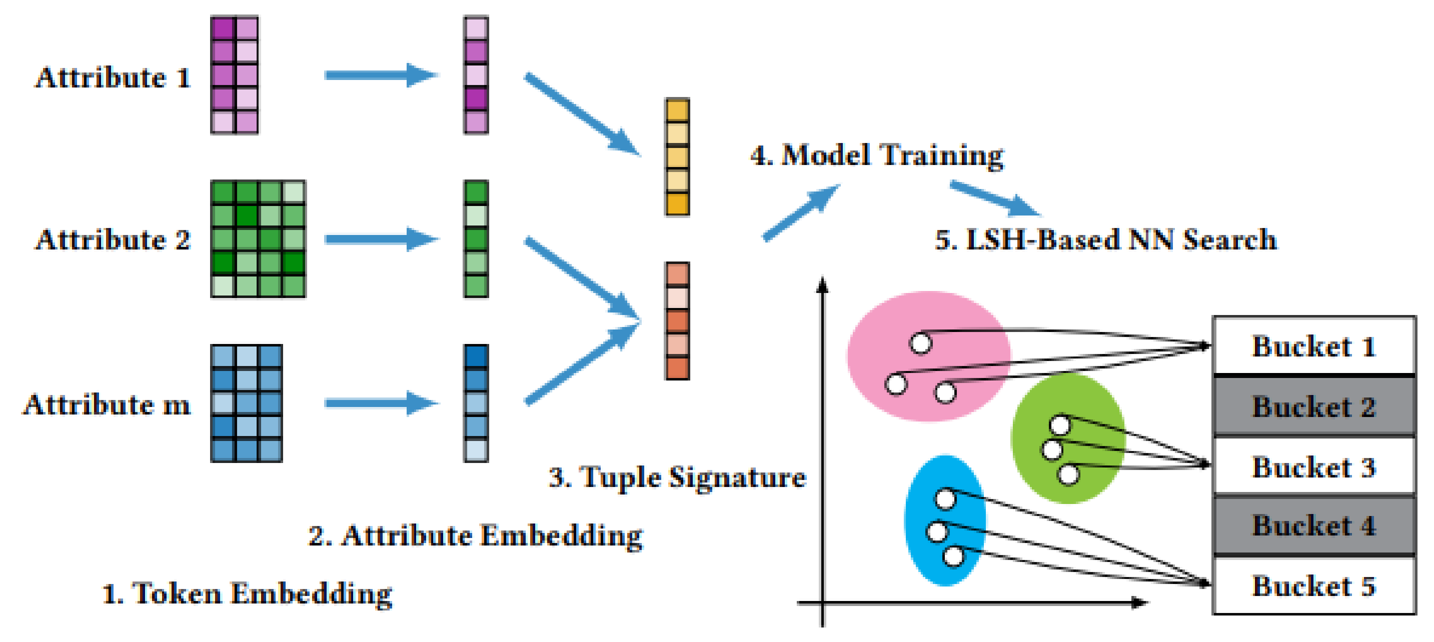
2. Syntactic Matcher (SYNM)
   1. Instance matcher (Jaccard similarity between two sets of values)
   2. Name matcher



Deep Learning for Blocking

AutoBlock [3]

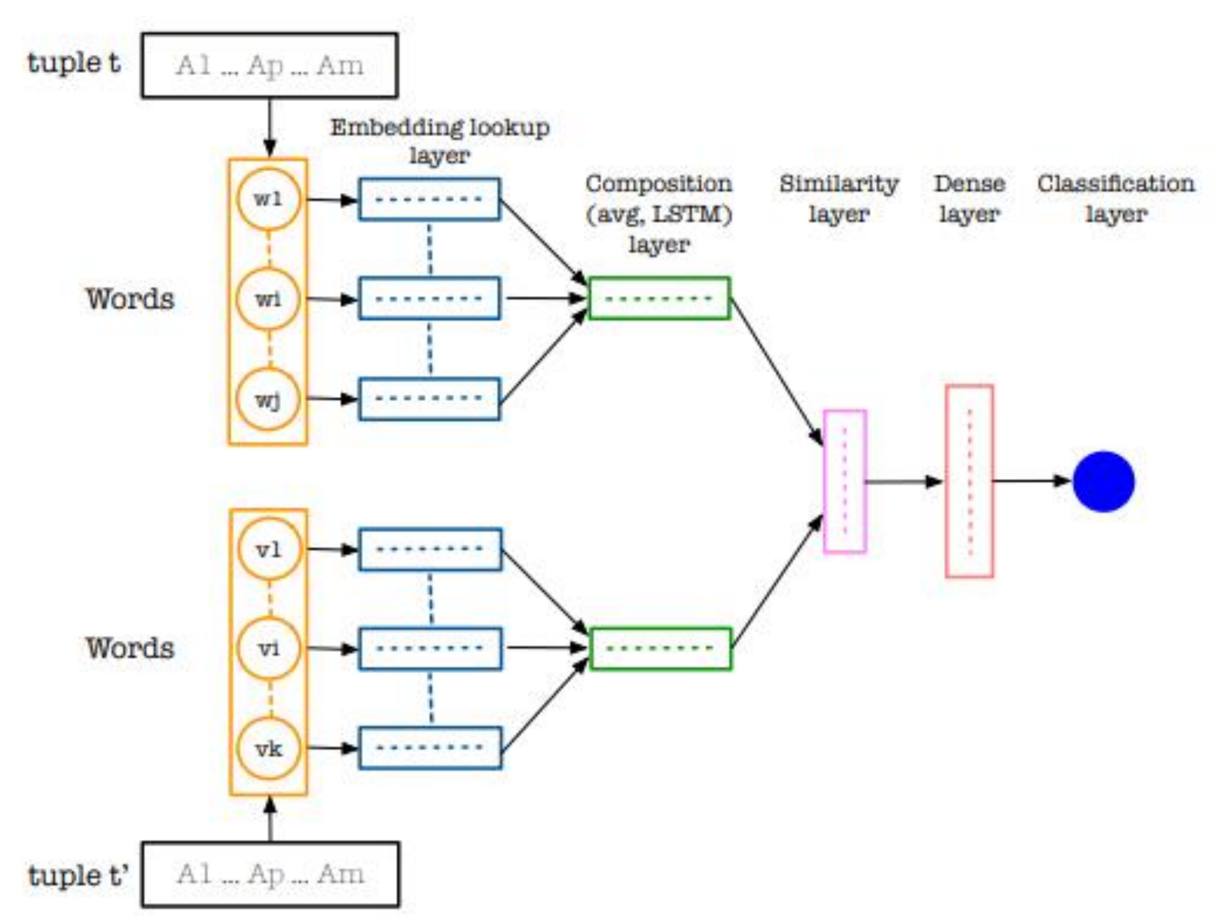
- Hands-off approach
- Combines similarity-preserving representation learning with nearest neighbor search



Deep Learning for Matching – Part I

DeepER [4]

- Extracts tuple embeddings from word embeddings



DeepER [4]

1. Straightforward approach:
    - Average word embeddings in each attribute
    - Concatenate attribute embeddings
    - **Entity Similarity**: k-dimensional cosine similarity (k=#attributes)
    - **Pros:** Simple & efficient
    - **Cons:** Ignores word order
2. Compositional Approach – RNN with LSTM
    - Encode a sequence of words from all attribute values into a x-dimensional vector
    - Bidirectional RNNs capture dependencies from both directions
    - Semantically related attributes should have the same order
    - **Entity Similarity**: x-dimensional vector from vector difference or hadamard product

- Considers out-of-vocabulary cases, e.g., Vocabulary Retrofitting
- Blocking: Multi-Probe LSH based on embeddings

Deep Learning for Matching – Part II

DeepMatcher [8]

Assumes that Schema Matching and Blocking are already in place

Three step approach:

1. Attribute embedding

   each attribute value is tokenized and converted into a sequence of embedding vectors

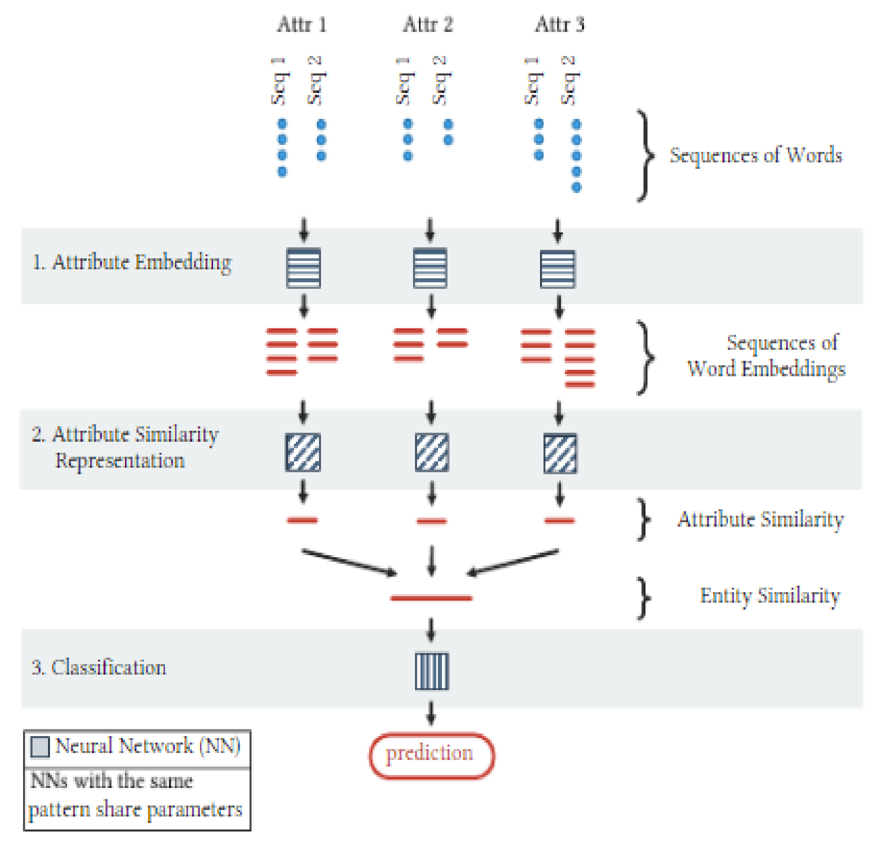
2. Attribute similarity representation

   one vector with the similarity per attribute

   Entity Similarity: concatenate attribute similarities

3. Binary classification

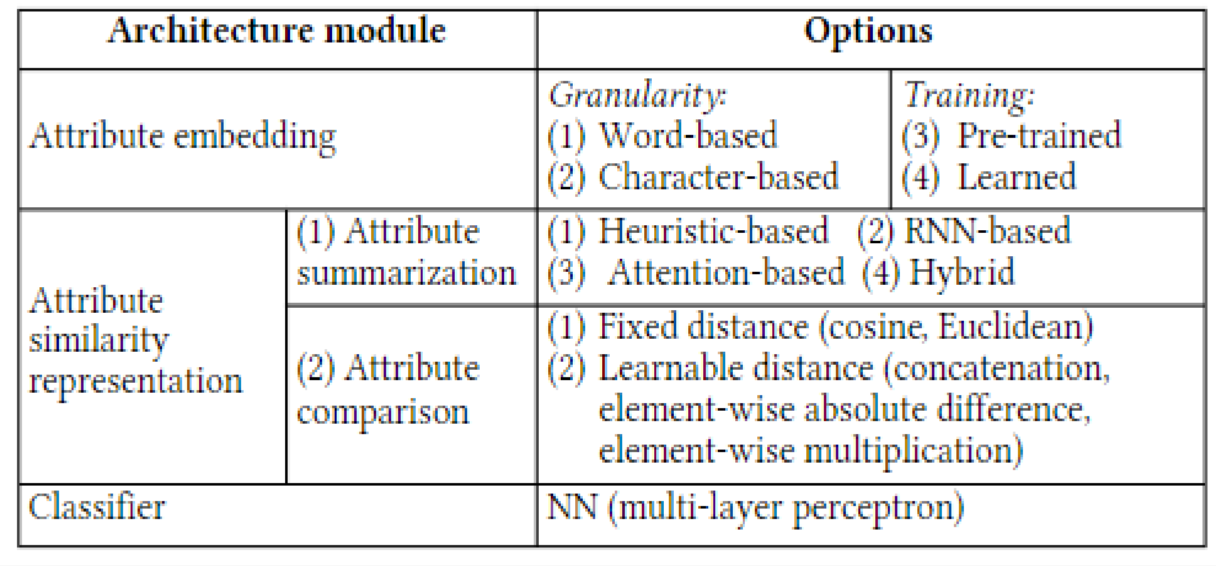
The choices for these steps frame the **solution space** for Deep Learning-based ER



Papadakis, Ioannou, Palpanas

DeepMatcher

Experimental Analysis including part of the possible solutions over real-world datasets



Architecture module		Options	
Attribute embedding		*Granularity:* (1) Word-based (2) Character-based	*Training:* (3) Pre-trained (4) Learned
Attribute similarity representation	(1) Attribute summarization	(1) Heuristic-based (2) RNN-based (3) Attention-based (4) Hybrid	
	(2) Attribute comparison	(1) Fixed distance (cosine, Euclidean) (2) Learnable distance (concatenation, element-wise absolute difference, element-wise multiplication)	
Classifier		NN (multi-layer perceptron)	

Main conclusions:

For Generations 1 and 2

Deep Learning does not outperform existing state-of-the-art solutions,

- significantly lower time efficiency (very high training time)
- requires too many labelled instances
- similar effectiveness,

unless the attribute values involve very high levels of noise

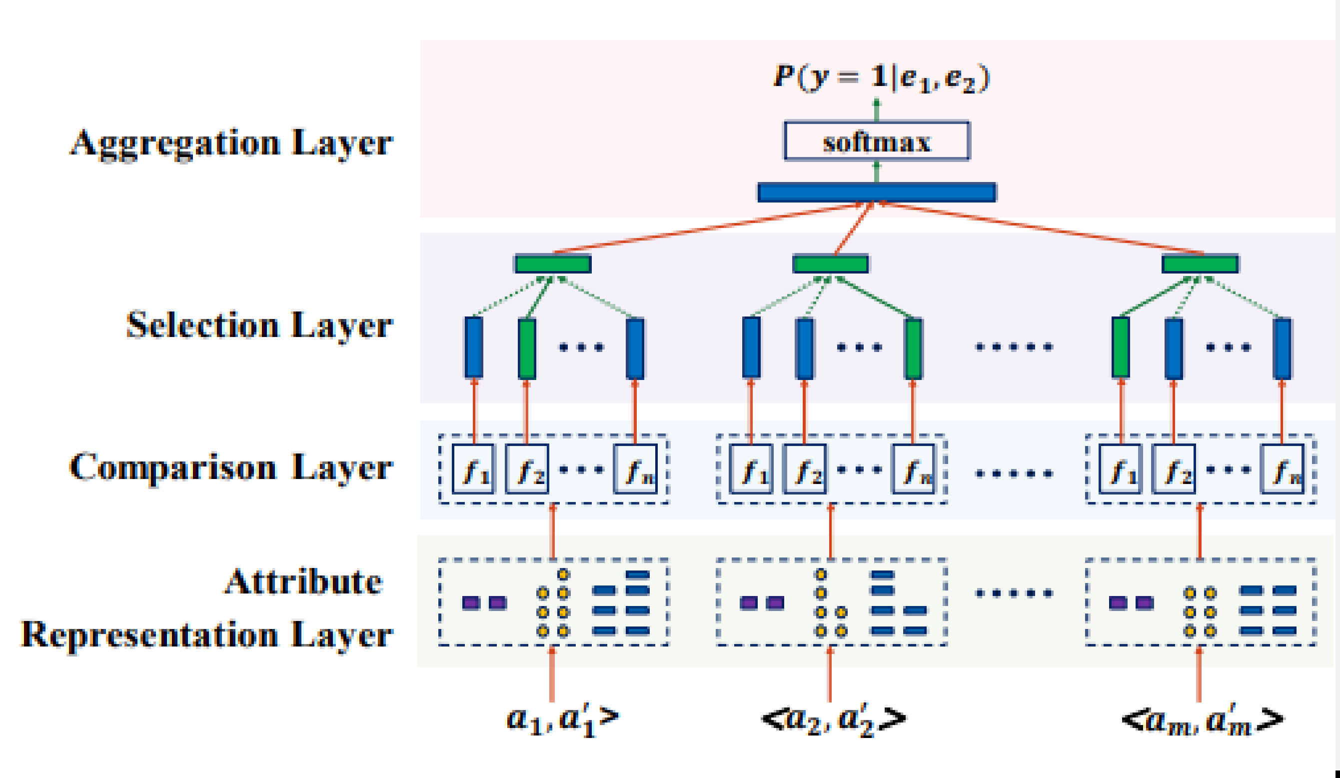
For Generation 3

Deep Learning yields a schema-agnostic approach that achieves the highest F-Measure

Deep Learning for Matching – Part III

Multi-Perspective Matching [9]

- Adaptively selects the optimal similarity measures for heterogenous attributes
- Considers 8 similarity measures:
  - Numeric attributes: relative difference, absolute difference
  - String attributes: exact similarity, edit distance, Jaro similarity, Smith and Waterman sim.
  - Textual attributes: RNN similarity [8], Hybrid similarity [8]



Deep Learning References

1. Ian J. Goodfellow, Yoshua Bengio, Aaron C. Courville. Deep Learning. Adaptive computation and machine learning, MIT Press 2016, ISBN 978-0-262-03561-3, pp. 1-775
2. R. C. Fernandez, E. Mansour, A. A. Qahtan, A. K. Elmagarmid, I. F. Ilyas, S. Madden, M. Ouzzani, M. Stonebraker, N. Tang. Seeping Semantics: Linking Datasets Using Word Embeddings for Data Discovery. ICDE 2018: 989-1000
3. W. Zhang, H. Wei, B. Sisman, X. L. Dong, C. Faloutsos, D. Page. AutoBlock: A Hands-off Blocking Framework for Entity Matching. WSDM 2020: 744-752
4. M. Ebraheem, S. Thirumuruganathan, S. R. Joty, M. Ouzzani, N. Tang. Distributed Representations of Tuples for Entity Resolution. PVLDB 11(11): 1454-1467 (2018)
5. J. Pennington, R. Socher, and C. D. Manning. Glove: Global vectors for word representation. In EMNLP, pages 1532–1543
6. T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In NIPS, pages 3111–3119, 2013
7. P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov. Enriching word vectors with subword information. TACL, 5:135–146, 2017
8. S. Mudgal, H. Li, T. Rekatsinas, A.H. Doan, Y. Park, G. Krishnan, R. Deep, E. Arcaute, V. Raghavendra. Deep Learning for Entity Matching: A Design Space Exploration. SIGMOD Conference 2018: 19-34
9. C. Fu, X. Han, L. Sun, B. Chen, W. Zhang, S. Wu, H. Kong. End-to-End Multi-Perspective Matching for Entity Resolution. IJCAI 2019: 4961-4967

- Introduction
- Four Generations

Part C – Entity Resolution Revisited: Leveraging External Knowledge

  - Deep Learning for Entity Resolution
  - **Crowd-sourced Entity Resolution**
- Challenges and Final Remarks

Crowd-sourcing

- Process/work divided among a large number of people, either paid or unpaid
- Idea: tasks are simple for human intelligence, but complex for computers
- Approach:
  - Break a problem into microtasks, called Human Intelligence Tasks (**HITS**)
  - Choose an online community
    - Amazon Mechanical Turk
    - Figure Eight (former CrowdFlower)
  - Assign to every individual, called **worker**, a series of HITs
  - Each worker is paid per executed HIT → **monetary cost**
  - Popular for solving many tasks, e.g., CrowdDB

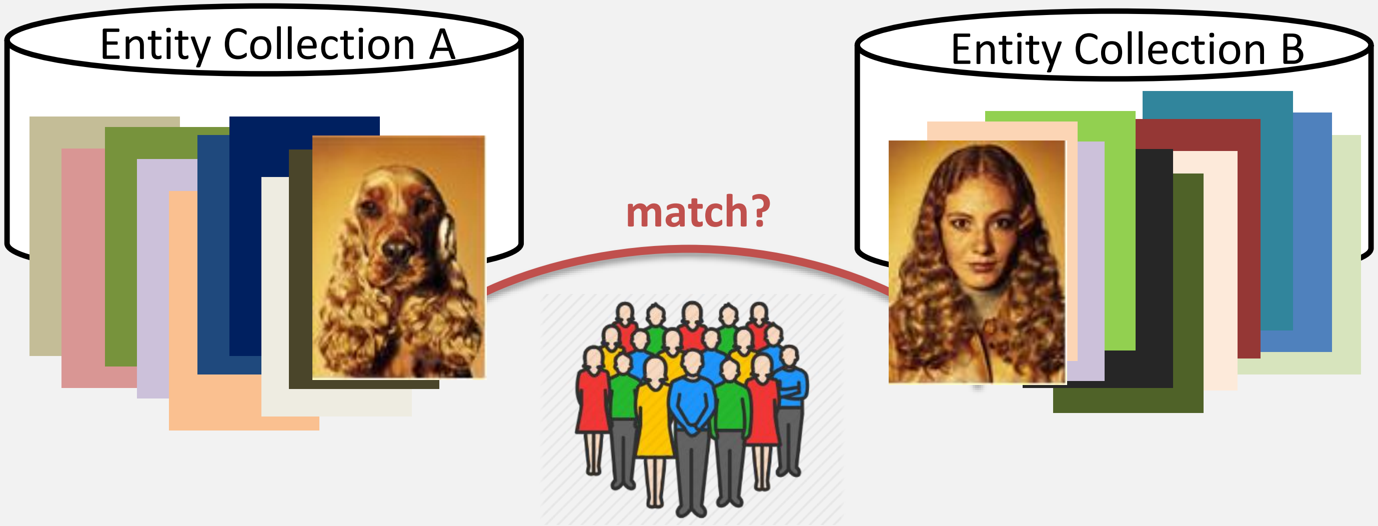
Crowd-sourcing for Entity Resolution

- Delegate the **entity matching decisions** to the workers

  i.e., transform pairwise comparisons into HITs
- Challenges:
  1. Generating HITs
  2. Formulating HITs
  3. Balancing accuracy and monetary cost
  4. Restricting the labor cost

Challenge 1: Generating HITs

- Fully Crowd-based Approach:
  - Create one HIT per pair of candidate match
- Pros:
  - Straightforward implementation
- Cons:
  - Quadratic complexity → huge monetary cost



Hybrid HITs Generation

- Two-step Approach:
  1. Produce candidate matches using G1 workflow (i.e., Blocking → Matching)
  2. Generate HITs for part of the candidate matches

- Two approaches:
  - CrowdER [8]
  - ZenCrowd [9]

CrowdER [8]

1. It automatically discards highly dissimilar pairs of entities
2. Ambiguous pairs (where similarity ≥ threshold) are formulated as cluster-based HITS (see below)

Pros:

- Most pairs are quickly filtered out
- Significantly lower number of HITs

Cons:

- Significant crowd-sourcing overhead

ZenCrowd [9]

- Automatic step:
  - for every entity, rank the most similar ones using TF-IDF
  - threshold on the ranking function **o**r on the number of retrieved documents
- Crowdsourcing step:
  - Dynamically assesses the quality of worker decisions using a probabilistic model:
    - Each worker is assigned a prior probability based on the training set
    - As new decisions are made, the unreliable workers are ignored → threshold on probability estimates for accepting a pair of entities as a match

Challenge 2: Formulating HITs

- Goal: find the best UI for presenting HITs to workers
- Naive Approach: one HIT per pair of candidate matches

  I.e., “*is* p_i *matching with* p_j” ?

Pros:

- Simple implementation
- Easy and comprehensible task

Cons:

- Quadratic complexity wrt to time and monetary cost
- Not scalable

Pair-based HITs [8]

- A single HIT contains ***k*** questions of the form

  “is p_i matching with p_j” ?
- Workers should check each question individually

Pros:

- Complexity is reduced to O(n^2/k)
- Lower time and monetary costs than naïve approach

Cons:

- Still, very high complexity

Cluster-based HITs [8]

- A single HIT contains ***k*** entities
- Each worker should mark all matches between all possible pairs

Pros:

- Complexity is further lowered to $O(n^2/k^2)$
- A HIT that contains many matches requires fewer comparisons than a pair-based HIT

Cons:

- Still, very high complexity
- Slightly lower accuracy
- Two duplicates are matched only if they co-occur in a HIT

Pair-based vs. cluster-based HITs

Trade-off between accuracy and cost [8]:

- The pair-based HITs are simpler, allowing workers to provide more accurate responses.
- The cluster-based HITs enable humans to mark many pairs of records with a few clicks.
- Generating Cluster-based HITs is an NP-hard problem,
  - CrowdER solves it in a greedy way

Hybrid HITs [10]

- Main idea: the error rate of workers is different for different profile pairs
  - The most “difficult” profile pairs (i.e., the high error-rate pairs) should form pair-based HITs
  - All other profile pairs (i.e., the low error-rate pairs) should form cluster-based HITs
- In practice, generating the best hybrid hits within the given budget is an optimization task
  - Waldo [10] proposes algorithms with probabilistic guarantees for solving it

Attribute-based HITs

- An entity may contain complex structures and attributes → overwhelming information for a worker
- Solution: **Crowdlink** [14]
  - Each pair of entities is decomposed into attribute-level HITs
  - A probabilistic framework selects the ***k*** best attributes that satisfy the user requirements

Challenge 3: Balancing accuracy and monetary cost

Goal: minimize the monetary cost, while maximizing accuracy

Generic approach [1]:

1. Exploit the transitive relations between detected duplicates
   - **positive transitivity**: if $e_i \equiv e_j$ and $e_j \equiv e_k$, then $e_i \equiv e_k$
   - **negative transitivity**:(a.k.a anti-transitivity): if $e_i \equiv e_j$, but $e_j \neq e_k$, then $e_i \neq e_k$

2. Optimize the order of HITs

   Find matches before non-matches to make the most of transitivity.
   NP-hard problem → approximately solved with heuristics

Random Ordering [3]

```
Initialize a similarity graph G=(V,E),
while ( E ≠ {} )
        pick a pair of entities $e_i$-$e_j$
        crowd-source $e_i$-$e_j$
        if ( $e_i \equiv e_j$ )
                contract edge <$e_i$, $e_j$>
```

Pros:

- It performs relatively well both in theory and practice

Cons:

- It ignores the edge weights

Edge-centric ordering [1]

```
Initialize a cluster graph G=(V,0)
$E_{sort}$ = candidate matches sorted in decreasing likelihood
while ( $E_{sort} \neq \{\}$ )
        get the next pair of candidate matches $e_i$-$e_j$
        if ( $cluster_i = cluster_j$ )
                deduced match
        else
                if an edge between $cluster_i$ =& $cluster_j$
                        deduced non-match
                else
                        crowd-source $e_i$-$e_j$
                        update cluster graph
```

Node-centric ordering [3]

```
$V_{sort}$ = sort entities in decreasing overall likelihood
for each $v_i$ in $V_{sort}$
    $V^i_{sort}$ = candidate matches in decreasing likelihood
    for each $v_j$ in $V^i_{sort}$
        crowd-source $e_i$-$e_j$
        if ($e_i \equiv e_j$ )
            break;
```

Maximizing Progressive Recall [4]

Iteratively crowd-source the pair that maximizes the *expected* marginal gain in recall

Core notions:

- Edge benefit:

  Expected #matches detected by crowdsourcing a pair

- Node benefit:

  Expected #matches that could be positively inferred if any of the incident edges is a match

Extending Ordering Algorithms [4]

- Extended Edge-centric Ordering
  - in every iteration, the **top-w** weighted edges are selected
  - the one with maximal edge benefit is crowd-sourced

- Extended Node-centric Ordering
  - in every iteration, the **top-w** weighed nodes are selected
  - the one with maximal node benefit is processed for crowd-sourcing

For **w=1**, we get the original algorithms

Probabilistic framework for Question Selection [2]

Core idea:

- Transform the output of a good similarity function into a probability function
- Estimate the expected accuracy by asking a particular question (in combination with transitive closure)
- Iteratively crowd-source the pair with the highest expected accuracy

Implementation:

- Analytically computing the optimal order is #P-hard
- Approximate solution based on heuristics
  e.g., discard pairs with very high or low probabilities.
- Alternate solution: iteratively crowd-source the pair closer to 0.5

Perfect vs. Noisy Workers

Problem:

- Previous works assume that workers are infallible
- Unrealistic assumption:
  - High accuracy workers have an error rate up to 25% [11,12], due to
    - lack of domain expertise,
    - individual biases,
    - task complexity and ambiguity
    - tiredness
    - malicious behaviors
- These works amplify worker errors, compromising overall ER accuracy

Solution:

- Generic approach to tackling noisy workers:
  - Assign the same HIT to multiple workers
  - Reconcile their responses through **majority voting**
  - Still, errors are possible [11,12]
- Need for specialized approaches that **inherently** tackle noisy workers

Adaptive Crowd-based Deduplication [12]

Three phases:

1. Pruning
   - Automatically eliminates record pairs with low similarities
2. Cluster generation
   - Applies correlation clustering on the results of initial crowd-sourced pairs
3. Cluster refinement
   - More new HITs to adjust the original disjoint clusters using **split** and **merge** operations

Pros:

- Higher accuracy. Reconciles inconsistent crowd results, instead of computing their transitive closure.

Cons:

- Higher monetary cost

Attribute Labeling and Clustering (ALC) [15]

- Crowdsource several attribute labels per entity
  - E.g., label the attribute “Type of celebrity” with “Actor/Actress”, “Singer” or “Athlete”
- Use attribute labels as blocking
  - Only pairs with common labels are crowdsourced
- Strategies for error mitigation
  - Majority voting
  - Approximate matching
- Probabilistic model optimizes the labelling process for a given recall

Partial-order based Framework [17]

- Crowdsourced ER is modelled as a DAG based on a partial-order of comparisons:
  - c_{ij} **dominates** c_{kl} if it has no smaller similarities than on every attribute
  - c_{ij} **strongly dominates** c_{kl} if it has larger similarity on at least one attribute
- For each crowdsourced comparison with sufficient confidence:
  - If it is labelled as "`match`" → all comparisons that dominate it are also labelled as "`match`"
  - If it is labelled as "`non-match`" → all comparisons that it dominates are also labelled as "`non-match`"
- Intelligent question selection:
  - serially (one-by-one) or in parallel

bDENSE [18]

- Crowdsourced ER is modeled as the Maximum Likelihood Clustering (MLC) of the similarity graph
  - NP-hard task
  - Spectral-Connected-Components for approximation
    - merges two clusters only when the overall evidence indicates that it is likely that their entities are matching
- Question selection:
  - In each iteration, crowdsource the comparison that maximizes the accuracy of MLC
    - Based an ρ-ratio, which considers the strength of positive and negative links between two disjoint sets of entities

Probabilistic ER With Crowd Errors [11, 16]

- Crowdsourced ER is modelled as clustering problem over an uncertain similarity graph:
  - Edge weights: (matching probability) = the ratio of workers who voted “`match`”

- Goal:
find the maximum likelihood (and transitively-closed) clustering

- Solution:
in each iteration, crowdsource the pair that maximizes the **reliability** of a clustering
  - Considers global information unlike [12,18]

Pair-wise Error Correction Layer [13]

- Flexible approach that can be combined with any method assuming infallible workers in three ways:
  1. Lazy
  2. Eager
  3. Adaptive
- Goal:

  maximize **Progressive F-measure**
- Outline:
  - Asks random queries before adding a node to an equivalence cluster
  - Adds the node to the cluster only if the crowd gives #log|C| positives answers
  - Merge phase to boost recall
  - Split phase to boost precision

Challenge 4: Restricting the labor cost

Limitations of most approaches:

- They crowd-source part of the end-to-end ER workflow
- They involve a high developer cost
- Task-specific implementations
- Restricted to ER problems with large budgets

Corleone [5]

- Combines crowdsourcing with active learning to offer:
  - an end-to-end solution
  - generic enough to support any application
  - involves no implementation cost
  - suitable for lay users

- Input comprises:
  - the data to be resolved
  - short HITs description for workers
  - few labeled pairs

Falcon [6] & CloudMatcher [7]

Corleone limitations:

- not scalable to large datasets
- runs in-memory on a single machine

Solutions:

- Falcon
  - scales to 1-2.5M entities for only ~$60 in 2-14 hours
  - runs Corleone on a cluster using MapReduce
  - exploits crowd-time to run machine tasks
- CloudMatcher
  - implements Falcon as a cloud service

References – Part I

1. J. Wang, G. Li, T. Kraska, M. J. Franklin, and J. Feng. Leveraging transitive relations for crowdsourced joins. In SIGMOD, pages 229–240, 2013.
2. S. E. Whang, P. Lofgren, and H. Garcia-Molina. Question selection for crowd entity resolution. PVLDB, 6(6):349–360, 2013.
3. N. Vesdapunt, K. Bellare, and N. N. Dalvi. Crowdsourcing algorithms for entity resolution. PVLDB, 7(12):1071–1082, 2014.
4. D. Firmani, B. Saha, and D. Srivastava. Online entity resolution using an oracle. PVLDB, 9(5):384–395, 2016.
5. C. Gokhale, S. Das, AnHai Doan, J. F. Naughton, Narasimhan Rampalli, Jude W. Shavlik, Xiaojin Zhu. Corleone: hands-off crowdsourcing for entity matching. SIGMOD Conference 2014: 601-6122.
6. S. Das, P. Suganthan G. C., AnHai Doan, J. F. Naughton, G. Krishnan, R. Deep, E. Arcaute, V. Raghavendra, Y.Park. Falcon: Scaling Up Hands-Off Crowdsourced Entity Matching to Build Cloud Services. SIGMOD Conference 2017: 1431-14463.
7. Y. Govind, E. Paulson, P. Nagarajan, P. Suganthan G. C., AnHai Doan, Y. Park, G. Fung, D. Conathan, M. Carter, M. Sun. CloudMatcher: A Hands-Off Cloud/Crowd Service for Entity Matching. PVLDB 11(12): 2042-2045 (2018).
8. Jiannan Wang, Tim Kraska, Michael J. Franklin, Jianhua Feng. CrowdER: Crowdsourcing Entity Resolution. PVLDB 5(11): 1483-1494 (2012).

References – Part II

9. Gianluca Demartini, Djellel Eddine Difallah, Philippe Cudré-Mauroux.ZenCrowd: leveraging probabilistic reasoning and crowdsourcing techniques for large-scale entity linking. WWW 2012: 469-478.
10. Vasilis Verroios, Hector Garcia-Molina, and Yannis Papakonstantinou. Waldo: An adaptive human interface for crowd entity resolution. In SIGMOD, pages 1133–1148, 2017.
11. V. K. Yalavarthi, X. Ke, and A. Khan. Select Your Questions Wisely: For Entity Resolution with Crowd Errors. In CIKM, pages 317-326, 2017.
12. S. Wang, X. Xiao, and C.-H. Lee. Crowd-Based Deduplication: An Adaptive Approach. In SIGMOD, pages 1263-1277, 2015.
13. S. Galhotra, D. Firmani, B. Saha, and D. Srivastava. Robust entity resolution using random graphs. In SIGMOD, pages 3–18, 2018.
14. C. J. Zhang, R. Meng, L. Chen, and F. Zhu. Crowdlink: An error-tolerantmodel for linking complex records. In ExploreDB, pages 15–20, 2015.
15. A. R. Khan and H. Garcia-Molina. Attribute-based crowd entity resolution. In CIKM, pages 549–558, 2016.
16. X. Ke, M. Teo, A. Khan, V. K. Yalavarthi. A Demonstration of PERC: Probabilistic Entity Resolution With Crowd Errors. PVLDB 11(12): 1922-1925 (2018)

References – Part III

17. C. Chai, G. Li, J. Li, D. Deng, and J. Feng. Cost-effective crowdsourced entity resolution: A partial-order approach. In SIGMOD, pages 969–984, 2016.
18. V. Verroios and H. Garcia-Molina. Entity resolution with crowd errors. In ICDE, pages 219–230, 2015.

- Introduction
- Four Generations
- Entity Resolution Revisited:
- Leveraging External Knowledge

Part D – Challenges and Final Remarks

Conclusions

Most promising works focus on:

1. Deep Learning

- Pros:
  - High accuracy
- Cons:
  - High training time
  - too many training instances

2. Crowd-sourcing.

- Pros:
  - High accuracy
- Cons:
  - High monetary cost
  - Not scalable to very large datasets

Challenges

Many challenges ahead

- Address shortcomings of Deep Learning
  - e.g., transfer learning for reducing labelling cost
- Cover gaps
  - e.g., incremental ER for semi-structured data
- New domains
  - e.g., adapt aforementioned techniques to privacy-preserving Entity Resolution

ER Systems

- Literature focuses on stand-alone methods
- More emphasis on end-to-end systems
  - Examples: Magellan [1], JedAI [2]
  - Partially cover the 4 generations
  - More efforts meeting the following requirements [1,3]:
    - open-source, extensible systems
    - process data of any structuredness
    - no coding for users
    - guidelines for creating effective solutions
    - covers the entire end-to-end pipeline exploit
    - a wide range of techniques

Automatic Configuration

Facts:

- Several parameters in every method
  - Applies to all generations and workflow steps
- Performance sensitive to internal configuration
- Manual fine-tuning required

Open Research Directions:

- Plug-and-play methods
- Data-driven configuration

Thank You!

References

1. P. Konda, S. Das, P. S. G. C., A. Doan, A. Ardalan, J. R. Ballard, H. Li, F. Panahi, H. Zhang, J. F. Naughton, S. Prasad, G. Krishnan, R. Deep, and V. Raghavendra. Magellan: Toward building entity matching management systems. PVLDB, 9(12):1197–1208, 2016.
2. G. Papadakis, L. Tsekouras, E. Thanos, G. Giannakopoulos, T. Palpanas, and M. Koubarakis. Domain- and structureagnostic end-to-end entity resolution with jedai. SIGMOD Record, 48(4):31, 2019.
3. B. Golshan, A. Y. Halevy, G. A. Mihaila, and W. Tan. Data integration: After the teenage years. In PODS, pages 101–106, 2017.