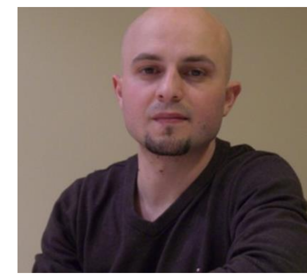
SiGMa: **S**imple **G**reedy **Ma**tching for Aligning Large Knowledge Bases





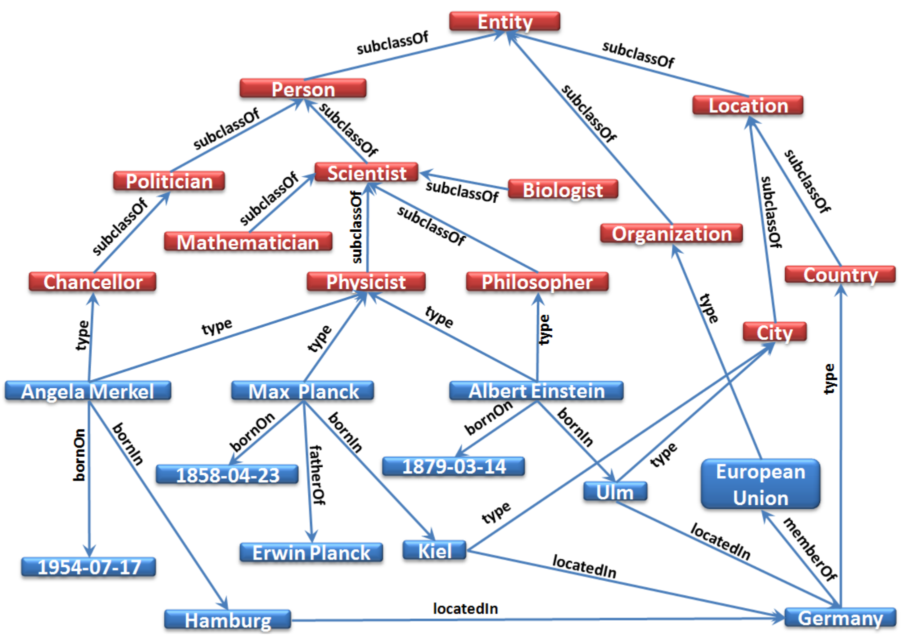


Simon Lacoste-Julien
Konstantina Palla
Zoubin Ghahramani
Thore Graepel
Gjergji Kasneci

KDD 2013 – August 14th 2013

Motivation: merging knowledge bases

YAGO
(Wikipedia based)



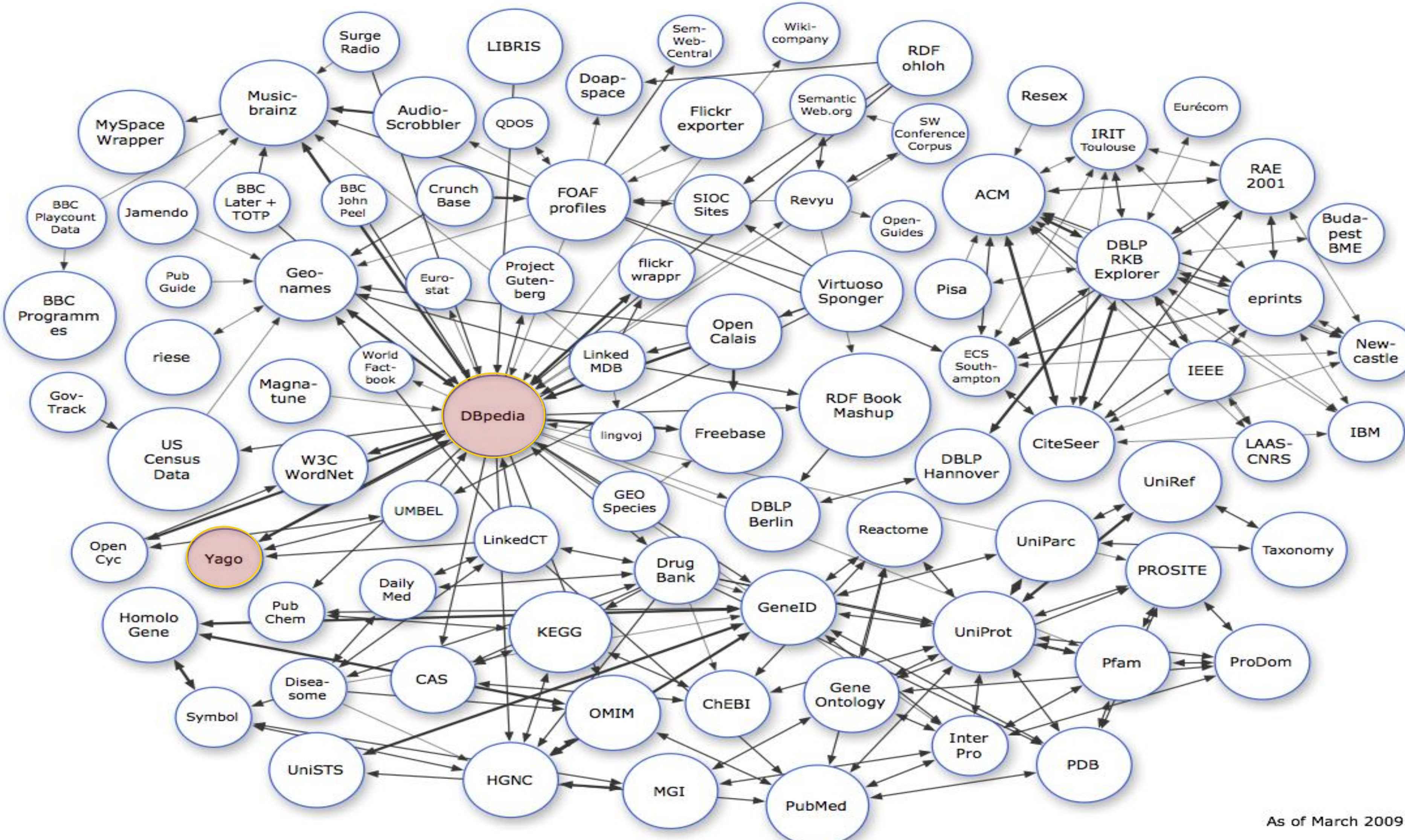
movie database

(John Travolta, ActedIn, Grease)

(Steven Spielberg, Directed, E.T.)

...

Linking open data project

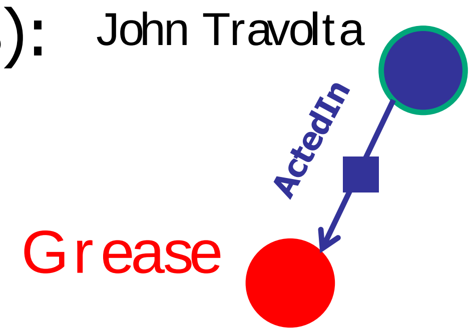


As of March 2009

Outline

- KB alignment formulation
- QAP objective motivation
- SiGMa algorithm
- Experiments

Formalization: knowledge base alignment



- a **knowledge base** is a list of **triples** (**facts**):
  - (entity1, relationship, entity2)
    e.g. (John Travolta, ActedIn, Grease)
- can think as a **graph** on entities
- given a pair of KBs, goal is to find a **1-1 mapping** between their **equivalent entities**
  - we suppose **no duplicate** within each KB
  - we suppose we are given a matching between the relationships
  - the entities have also **attributes** given as triples:
    (entity1, propertyName, value) -> these can be used to construct a **similarity score** between pair of entities
- input: pair of KBs + relationships matching
  output: a ranked list of matched pairs from KB1 & KB2

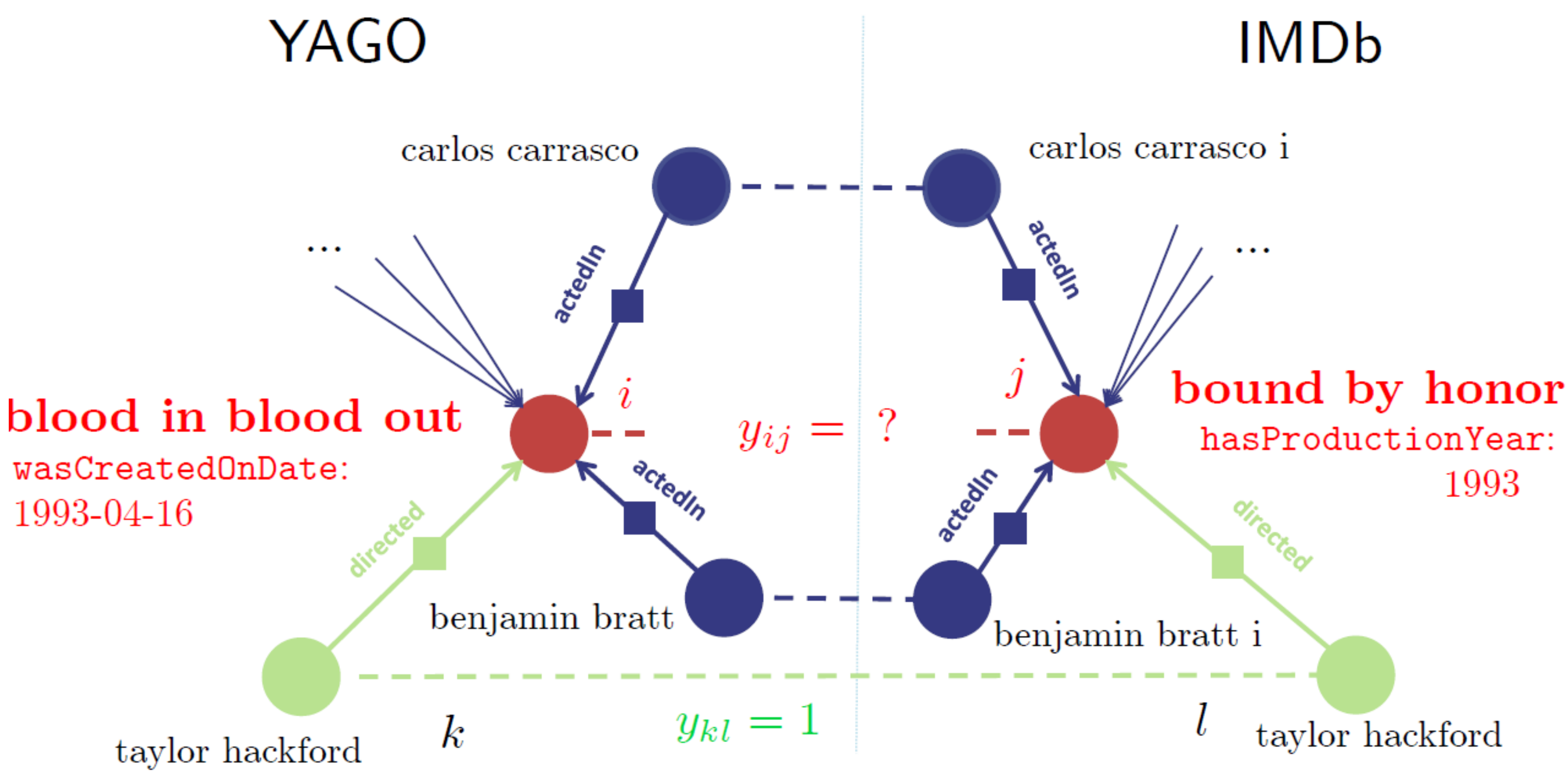
Current approaches

- ontology alignment algorithms (e.g. RiMOM)
  -> do not scale to millions of entities

- record linkage (DB) / entity resolution (NLP)
  - scale using indexing / blocking techniques
  - but typically do not exploit the 1-1 combinatorial structure

- ... SiGMa: scalable greedy algorithm which exploits the 1-1 combinatorial structure

Motivating example & intuition



Use neighbors for: 1) scoring candidates

2) suggest candidates (iterative blocking)

Quadratic Assigment objective

$y_{ij} \in \{0, 1\}$

normalizing weight

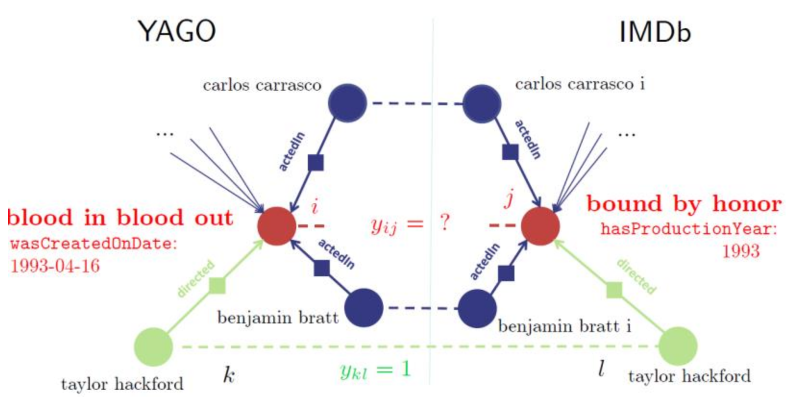
$$\max_{y \in \mathcal{M}} \sum_{(i,j)} y_{ij} \left[s_{ij} + \underbrace{\sum_{(k,l) \in \mathcal{N}_{ij}} y_{kl} w_{ij,kl}}_{} \right]$$

pairwise similarity score

between i and j

graph compatibility score:

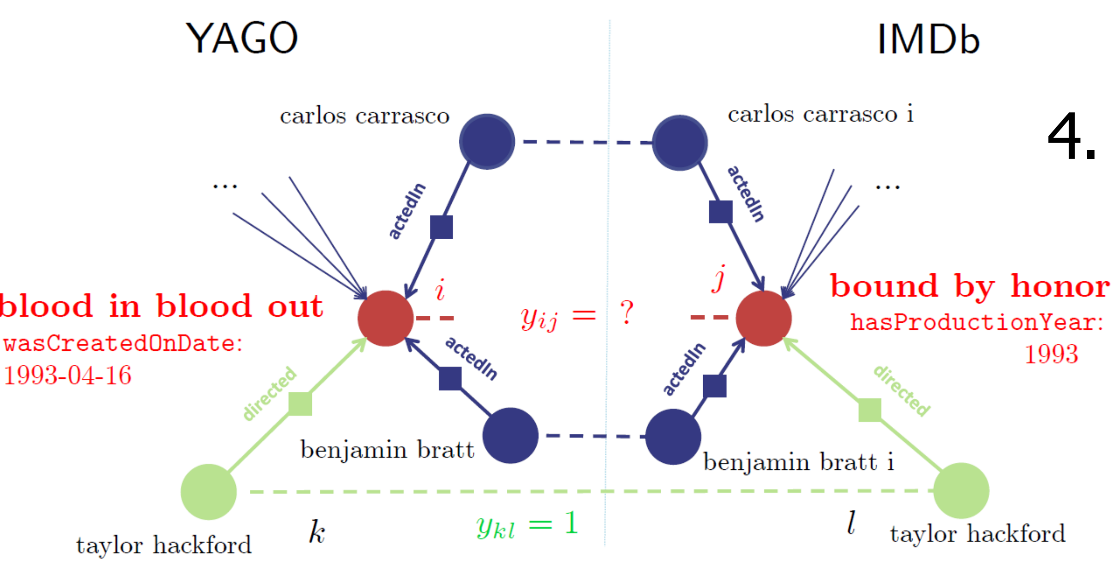
counts the number of valid neighbors which are currently matched (**context**)



Simple Greedy Matching (SiGMa)

$$\sum_{(i,j)} y_{ij} \left[s_{ij} + \sum_{(k,l) \in \mathcal{N}_{ij}} y_{kl}\, w_{ij,kl} \right]$$

1. Start with seed match
2. Put neighbors in S
3. At each iteration:
   a) pick new pair in S which max. increase
   b) add new neighbors in S
4. Stop when variation below threshold



- efficient specialization of agglomerative clustering of [Bhattacharya & Getoor 2007]
- LINDA [Böhm & al. CIKM 12] -> MapReduce on 3B facts!

Experiments: 1) Large-Scale KBs

- Aligning YAGO to IMDb:
  - 4 matched relationships
  - YAGO: 1.5M entities
  - IMDb: 3M entities
  - 50k ground truth pairs (extracted from backlinks)
- Our greedy algorithm SiGMa:
  - run in less than 1 hour (in Python, single threaded!)
    - 50x speedup over PARIS [Suchanek et al. 2011]
  - get 98% precision / 93% recall / 95% F-measure
    - (vs. 57% recall for string matching)
    - sampled precision is above 90%
  - also works without a seed

Experiments: 2) benchmarks

- Also ran on standard Ontology Alignment Evaluation Initiative benchmarks
  - got state-of-the-art results without tweaking parameters

- e.g. Rexa-DBLP OAEI 2009 benchmark:

  Rexa: 13k entities

  DBLP: 1.6M entities
  - SiGMa gets 99% / 94% / 96% in less than 10 minutes
    - vs. 97% / 74% / 84% for best previous result by RiMOM [Li + al. 09] in 36 hours!
    - got 1k new mostly correct matches not in ground truth

When should you use SiGMa?

- When to use SiGMa?
  - 1-1 assumption
    - if not -> use deduplication as pre-processing
    - otherwise, use more general entity resolution algorithms
  - relationships between entities
  - some pair of entities with strong signal
  - large-scale
    - for small scale, use PARIS or standard ontology alignment algorithms

Conclusions & future work

- SiGMa:
  - lightweight iterative greedy algorithm to efficiently align KBs with millions of entities
  - can use tailored similarity measures
  - provides natural tradeoff between precision & recall
  - exploits relationship graph to **score** decisions and to **propose candidates**
  - despite simplicity & greediness, does surprisingly well!
- Future work:
  - find way to revisit decisions efficiently?
  - handle non 1-1 alignments?
  - learn score functions using training data (learning to rank framework)

Thanks for listening!