INTRODUCTION

Knowledge bases (DBpedia, Freebase, Google Knowledge Graph, NELL, YAGO, etc.) are often incomplete. To infer missing facts from these KBs, we study the problem of mining first-order knowledge in the form of Horn clauses:

\[(w, B \rightarrow H(x, y)).\]

We present the Ontological Pathfinding algorithm and propose a series of parallelization and partitioning techniques to mine 36,625 inference rules from Freebase in 34 hours; no existing approach achieves this scale.

STATE-OF-THE-ART

AMIE: An in-database ILP system, transforming mining algorithms into database queries. Mined 1090 rules from YAGO2 in 3.59 minutes. Not scalable to larger KBs.

Sherlock: An ILP system for web text, mining 30,912 rules from TextRunner (250K facts) in 50 minutes.

Challenge: No existing work scales to the Web, e.g., Freebase (112M entities, 388M facts).

CONTRIBUTIONS

Goal: Mining first-order knowledge from web-scale knowledge bases.

Result: Design the Ontological Pathfinding algorithm to mine 36,625 inference rules from Freebase (112M entities and 388M facts) in 34 hours; publish the first Freebase rule set.

- Partition KB into independent subsets to reduce join sizes. (Improve runtime from 2.55 days to 5.06 hours for a single task.)
- Design a parallel rule mining algorithm for each partition. (Achieve 3-6 times of speedup.)
- Prune inefficient and erroneous candidate rules. (Make joins possible.)

DATASETS

Freebase: Largest public knowledge base with 112M entities and 388M facts, migrating to WikiData.

YAGO2s: Knowledge base extracted from Wikipedia, WordNet, and GeoNames. 2.1M entities, 4.5M facts.


THE PARTITIONING ALGORITHM

- Goal: partition KB into small, independent subsets: \(|T_i| \leq s\) and \(|M_i| \leq m\), and \(\bigcup_i \text{OP}(T_i, M_i)\).
- Greedy assignment:
  \[\text{for all } \ell \text{ if } |T_\ell(M_\ell \cup \{r\})| < |T_\ell(M_\ell \cup \{r\})| \quad M_\ell' \leftarrow M_\ell \cup \{r\}, \quad \text{else } M_\ell' \leftarrow M_\ell \cup \{r\}\]
- Recursive binary partitioning: Until all partitions \(\forall i\) satisfy \(|T_i| \leq s\) and \(|M_i| \leq m\).
- Complexity: \(O((t-1)|S||M|) \rightarrow O(t^{-1}\frac{1}{s}|M|)\).

THE PARTITIONING ALGORITHM

Parallel Rule Mining

- Relational KB model.
- Parallel operations.
- Pruning:
  \[\text{diedIn}(x, z), \text{wasBornIn}(y, z) \rightarrow \text{hasChild}(x, y)\]

EXPERIMENTS

Setup: 64-core machine, 512GB RAM, Spark 1.3.0.

- Overall: Better scale than state-of-the-art–36,625 rules in 33.22 hours.
- Shorter rules are more useful; e.g., trivial extensions and composite rules in length 4 rules:

EFFECT OF PARTITIONING

- Runtime: 2.55 days \(\rightarrow\) 5.06 hours as \(s\): 200M \(\rightarrow\) 5M and \(m\): 10K \(\rightarrow\) 1K.
- Slowest partition: 1.27 days \(\rightarrow\) 38.14 minutes.