**Executive Summary**

- **Machine Reading (MR) requires joint inference**
  - Desirable to have a versatile, easy-to-use language
- **Markov logic network (MLN) is such a language**
  - But current approach to inference is monolithic: one algorithm for entire program
  - Suboptimal scalability and quality
- **Key observation:** MLNs in MR may contain subtasks
  - E.g., NER, coreference, link prediction, etc.
- **Felix hypothesis:** We can get higher scale and quality by exploiting those subtasks
  - Idea: solve subtasks with specialized algorithms
  - Preliminary results show dramatic improvement in both efficiency and quality

**Example Plan in Felix**

- **Task:** Extract person-org affiliations from webpages
  - Subtasks
    - coreference (e.g., pcoref for person coref)
    - link prediction (e.g., affil for affiliations)
- **Predicates**
  - Name, post, role, name2, post2, role2
  - Data manipulation optimized by Felix using a Relational Database Management System (Step 3)
  - Generic MLN inference handled by Tuffy, our prior work of an MLN inference engine [Niu et al. 2011]
  - Joint inference via cycles through operators

**Felix at Work**

- **Felix Approach to MLN Inference**
  1. Recognize specialized subtasks in an MLN program
  2. Compile a logical plan by assigning subtasks to specialized algorithms (aka “statistical operators“)
  3. Optimize data movement between operators
  4. Execute operators according to optimized plan

**Felix Approach to MLN Inference**

![Diagram of Felix Approach to MLN Inference](image)

**Experiments**

- **Validation:** Felix achieves higher efficiency and quality than monolithic
- **Method:** Run Felix, Tuffy, and Alchemy on MLN programs for IE
  - Three tasks: NFL winner-loser, DBLife affiliations, Enron person-phone
  - Baseline extractors: CRF, DBLife Cimple, IBM SystemT
  - Two MLNs each task: PO without coref subtask; P1 with coref
- **Results:**
  - Alchemy crashed on all MLN programs
  - Tuffy crashed on P1 of NFL and DBLife
  - Felix scales and achieves best quality
- **Quality:** P1 > PO > baselines

**Conclusion & Future Work**

- **Conclusion:**
  - Felix achieves higher efficiency and quality by running specialized algorithms for well-studied subtasks buried in MLN programs
- **Future Work:**
  - More specialized subtasks/algorithms, e.g. Cuts
  - More optimization in data movement
  - More systematic recognition/planning
  - Efficient weight learning in Felix (for feedback between operators)

**Specialized Subtasks in Felix**

- **Logistic Regression**
  - $P[\text{tag} = 1] = \frac{1}{1 + \exp(-\sum_w w_i f_i)}$
  - $w_{\text{tag}}$ (pos, fea)
  - $F(\text{pos, fea})$ $\land T(\text{pos, tag})$

- **Conditional Random Field**
  - $P[y = 1] = \frac{1}{Z} \exp \sum_{f_i} f_i w_{\text{tag}, i}$
  - $w_{\text{tag}, i}$ $\land T(\text{pos, tag1})$
  - $\land T(\text{pos, tag2})$

- **Correlation Clustering (Coreference)**
  - $\omega \text{Coref}(x, y)$, $\text{Coref}(y, z) \Rightarrow \text{Coref}(x, z)$
  - $\mathcal{A} \mathcal{D} \mathcal{V} \text{enue}(x, y) \Rightarrow \text{Coref}(x, y)$
  - $\mathcal{S} \mathcal{D} \mathcal{M} \text{I} \text{D}(x, y) \Rightarrow \text{Coref}(x, y)$

- **Input:** Graph $G = (V, w)$ where $w: V \times V \rightarrow \mathbb{R}$
  - **Output:** A partition of $V$
  - **Goal:** Minimize weights of violated edges

- **There are specialized algorithms for them (Step 2)**
  - [Hilbe 2009; Lafferty et al. 2001; Arasu et al. 2009]

**References**

- A. Arasu, C. Re, D. Suciu, Large-scale deduplication with constraints using Dedupalog, ICDE 2009