SkILL – a Stochastic Inductive Logic Learner

Joana Côrte-Real
CRACS & INESC TEC, University of Porto
jcr@dcc.fc.up.pt

SkILL

SkILL is a Probabilistic Inductive Logic Programming tool which can extract non-trivial knowledge from probabilistic data. It takes as input relational data annotated with probabilities and it produces a logical model that can output probabilities as well.

The SkILL system:

- Supports logical constructs, arbitrary variable logical terms and annotated disjunctions
- Can reduce the theory search space using a rank-based strategy, resulting in a polynomiallly bound algorithm
- Takes advantage of the probabilistic nature of the data to perform an efficient search space traversal

Sequential PILP Algorithm

Initially, the algorithm uses the TopLog engine to generate all possible rules (theories of length one). A rule is constructed from literals that are contained in the PBK and are selected by the user. This approach results in rules one that mirror patterns contained in the observations w.r.t. the PBK. The algorithm proceeds by generating theories length greater than one until reaching MaxHypLength.

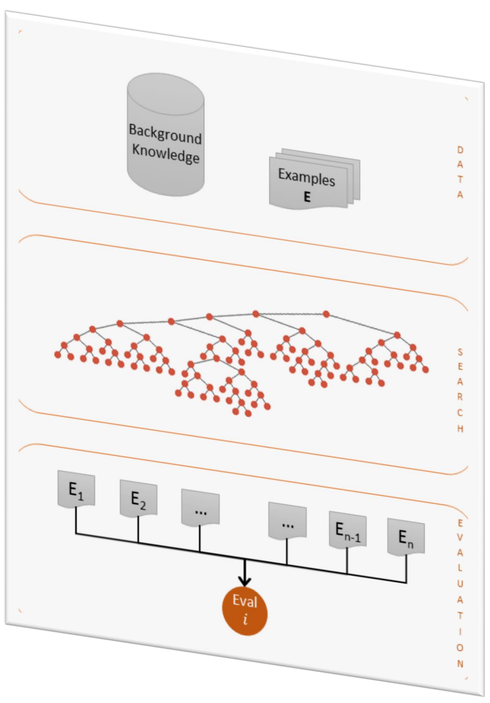
Algorithm 1: SkILL Algorithm

```
1  Input = PBK, PE, MaxHypLength, Psize, Ssize, PRankMetric,
     SRankMetric, EvalMetric
2  Output = Best hypothesis according to EvalMetric
3  Hyps1 = HypsN = AllHyps = generate_hyps_length_one(PBK, PE)
4  for Length = 2; Length ≤ MaxHypLength; Length++ do
5  |   Primary = select_members(HypsN, Psize, PRankMetric)
6  |   Secondary = select_members(Hyps1, Ssize, SRankMetric)
7  |   HypsN = generate_combinations(Primary, Secondary)
8  |   AllHyps = AllHyps ∪ HypsN
9  end
10 return best_hypothesis(AllHyps, EvalMetric)
```

Combining theories to generate new theories with larger length is not a trivial task; possible combinations are N choose K, N being the number of length one theories and K the maximum theory length.

SkILL's search strategy selects candidate theories for two different sets, named Primary and Secondary and new theories are then generated by combining members from each set. In each iteration of the algorithm, the Primary set is filled with the Psize best theories, according to a given ranking metric (argument PRankMetric), from the set of theories generated in the previous iteration.

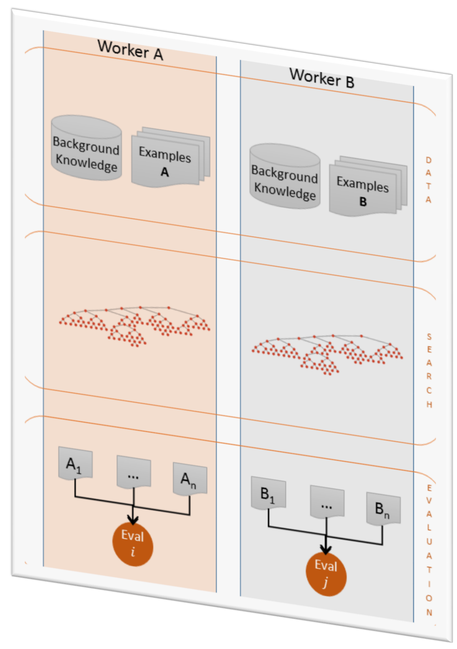
The Secondary set is populated with Ssize theories from the set of theories of length one, according to SRankMetric. Depending on the ranking metrics chosen, the system can generate theories in a fully stochastic way, use best theory or create a heterogeneous mix. The stochastic component of the selection is distinct for each iteration. Finally, all theories are evaluated according to the given evaluation metric (argument EvalMetric), and the best generated theory for all different lengths is returned.



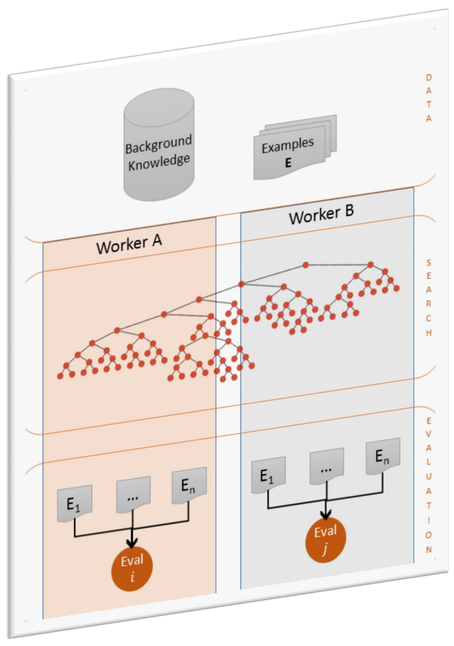
Data Parallelism

Data parallelism in PILP systems can be implemented similarly to ILP. The main idea is to divide the (Probabilistic) Background Knowledge into several chunks and have them execution in different workers.

Depending on the way this division is made, this strategy may cause redundancy, since the same theories may be generated independently in different workers, and therefore evaluated repeatedly.



Search Parallelism



SkILL's algorithm could take advantage of search parallelism both in the rule and in the theory generation steps of the algorithm. In the rule generation step, the algorithm could simply distribute the TopLog rules across several workers.

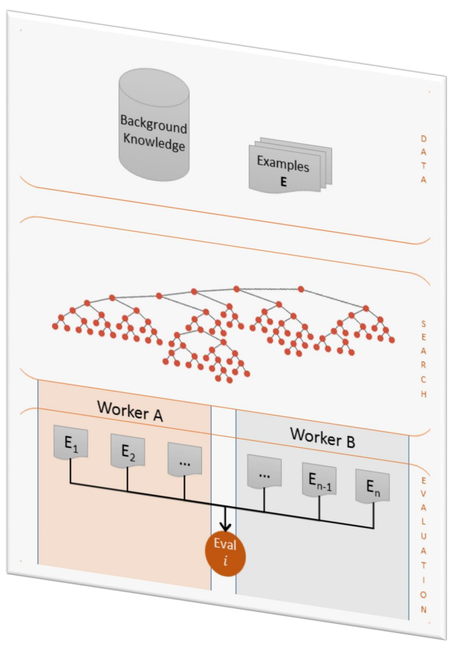
Because of the independent nature of SkILL's ranking population approach, it would be possible to split the theory generation stage between several workers, and unlike in the Data Parallelism approach, there would be no redundancy. At the end of each iteration, the generated theories would be merged

Evaluation Parallelism

The Evaluation Parallelism strategy is a straightforward way to parallelize ILP and PILP.

It consists of distributing the (Probabilistic) Examples across workers and perform the evaluation of each candidate theory in parallel.

This strategy could be particularly relevant in PILP, because the evaluation is probabilistic (not a truth value), and so the complexity of each evaluation is much greater than in the discrete case of ILP



Acknowledgments

Joana Côrte-Real is funded by the FCT grant SFRH/BD/52235/2013.

FCT Fundação para a Ciência e a Tecnologia
MINISTÉRIO DA EDUCAÇÃO E CIÊNCIA
CRACS INESCTEC ASSOCIATE LABORATORY PORTUGAL
UT Austin | Portugal
INTERNATIONAL COLLABORATORY FOR EMERGING TECHNOLOGIES, COLAB