Abductive Plan Recognition By Extending Bayesian Logic Programs

Sindhu V. Raghavan & Raymond J. Mooney
The University of Texas at Austin

Plan Recognition

- Predict an agent's top-level plans based on the observed actions
- *Abductive reasoning* involving inference of cause from effect
- Applications
  - Story Understanding
  - Strategic Planning
  - Intelligent User Interfaces

Plan Recognition in Intelligent User Interfaces

$ cd test-dir

$ cp test1.txt my-dir

$ rm test1.txt

What task is the user performing?

move-file

Which files and directories are involved?

test1.txt and test-dir

Data is relational in nature - several files and directories and several relations between them

Related Work

- First-order logic based approaches *[Kautz and Allen, 1986; Ng and Mooney, 1992]*
  - Knowledge base of plans and actions
  - Default reasoning or logical abduction to predict the best plan based on the observed actions
  - Unable to handle uncertainty in data or estimate likelihood of alternative plans
- Probabilistic graphical models *[Charniak and Goldman, 1989; Huber et al., 1994; Pynadath and Wellman, 2000; Bui, 2003; Blaylock and Allen, 2005]*
  - Encode the domain knowledge using Bayesian networks, abstract hidden Markov models, or statistical n-gram models
  - Unable to handle relational/structured data
- Statistical Relational Learning based approaches
  - Markov Logic Networks for plan recognition *[Kate and Mooney, 2009; Singla and Mooney, 2011]*

Our Approach

- Extend *Bayesian Logic Programs* (BLPs) ***[Kersting and De Raedt, 2001]*** for plan recognition
- BLPs integrate *first-order logic* and *Bayesian networks*

- Why BLPs?
  - ***Efficient grounding mechanism*** that includes only those variables that are relevant to the query
  - Easy to extend by incorporating any type of ***logical inference*** to construct networks
  - Well suited for capturing ***causal relations*** in data

Outline

- ✓ Motivation
- ❑ Background
  - ✧ Logical Abduction
  - ✧ Bayesian Logic Programs (BLPs)
- ❑ Extending BLPs for Plan Recognition
- ❑ Experiments
- ❑ Conclusions

Logical Abduction

❑ Abduction

- Process of finding the best explanation for a set of observations

❑ Given

- Background knowledge, *B*, in the form of a set of (Horn) clauses in first-order logic
- Observations, *O*, in the form of atomic facts in first-order logic

❑ Find

- A hypothesis, *H*, a set of assumptions (atomic facts) that logically entail the observations given the theory:

$$B \cup H \models O$$

- Best explanation is the one with the fewest assumptions

Bayesian Logic Programs (BLPs)

[Kersting and De Raedt, 2001]

- Set of Bayesian clauses $a|a_1,a_2,....,a_n$
  - Definite clauses that are universally quantified
  - Range-restricted, i.e variables{head} $\subseteq$ variables{body}
  - Associated conditional probability table (CPT)
    - **P(head|body)**
- Bayesian predicates a, a_1, a_2, …, a_n have finite domains
  - Combining rule like noisy-or for mapping multiple CPTs into a single CPT.

Inference in BLPs

[Kersting and De Raedt, 2001]

- Logical inference
  - Given a BLP and a query, *SLD resolution* is used to construct proofs for the query
- Bayesian network construction
  - Each ground atom is a random variable
  - Edges are added from ground atoms in the body to the ground atom in head
  - CPTs specified by the conditional probability distribution for the corresponding clause
  - $P(X) = \prod_i P(X_i \mid Pa(X_i))$
- Probabilistic inference
  - *Marginal probability* given evidence
  - *Most Probable Explanation (MPE)* given evidence

BLPs for Plan Recognition

- SLD resolution is *deductive inference*, used for predicting observations from top-level plans

- Plan recognition is *abductive* in nature and involves predicting the top-level plan from observations

BLPs cannot be used as is for plan recognition

Extending BLPs for Plan Recognition

BLPs + Logical Abduction

=

BALPs

BALPs – Bayesian Abductive Logic Programs

Logical Abduction in BALPs

- Given
  - ✧ A set of observation literals $O = \{O_1, O_2, \ldots . O_n\}$ and a knowledge base KB
- Compute a set abductive proofs of O using *Stickel's abduction algorithm* ***[Stickel 1988]***
  - ✧ Backchain on each O_i until it is proved or assumed
  - ✧ A literal is said to be *proved* if it unifies with a fact or the head of some rule in KB, otherwise it is said to be *assumed*
- Construct a Bayesian network using the resulting set of proofs as in BLPs.

Example – Intelligent User Interfaces

- Top-level plan predicates
  - copy-file, move-file, remove-file
- Action predicates
  - cp, rm
- Knowledge Base (KB)
  - cp(Filename,Destdir) | copy-file(Filename,Destdir)
  - cp(Filename,Destdir) | move-file(Filename,Destdir)
  - rm(Filename) | move-file(Filename,Destdir)
  - rm(Filename) | remove-file(Filename)
- Observed actions
  - cp(test1.txt, mydir)
  - rm(test1.txt)

Abductive Inference

Assumed literal

copy-file(test1.txt,mydir)

cp(test1.txt,mydir)

cp(Filename,Destdir) | copy-file(Filename,Destdir)

Abductive Inference

Assumed literal

copy-file(test1.txt,mydir) move-file(test1.txt,mydir)

cp(test1.txt,mydir)

cp(Filename,Destdir) | move-file(Filename,Destdir)

Abductive Inference

Match existing assumption

copy-file(test1.txt,mydir) **move-file(test1.txt,mydir)**

cp(test1.txt,mydir) **rm(test1.txt)**

rm(Filename) | move-file(Filename,Destdir)

Abductive Inference

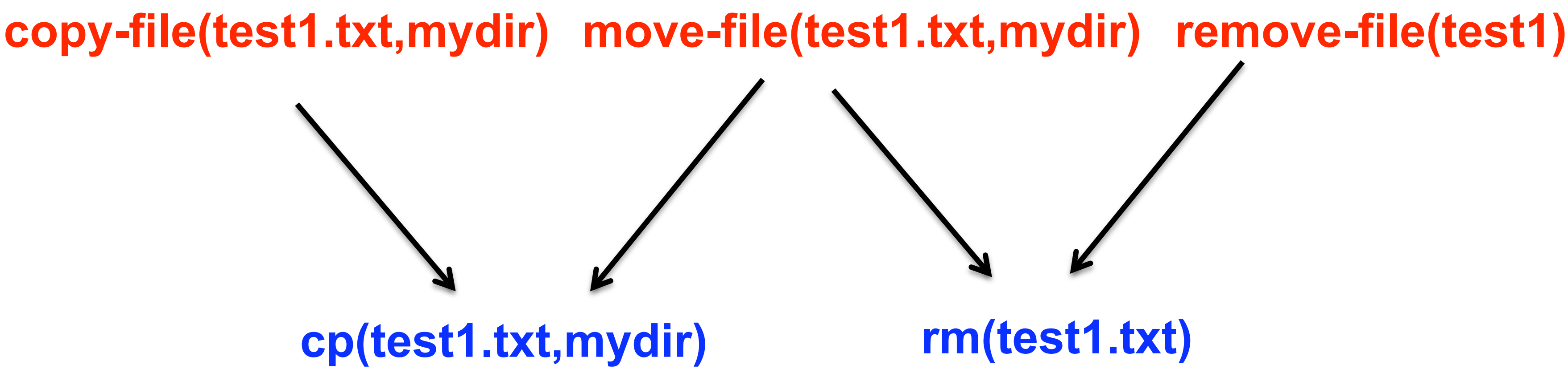
Assumed literal

copy-file(test1.txt,mydir) **move-file(test1.txt,mydir)** **remove-file(test1)**

cp(test1.txt,mydir) **rm(test1.txt)**

rm(Filename) | remove-file(Filename)

Structure of Bayesian network



Probabilistic Inference

- Specifying probabilistic parameters
  - Noisy-and
    - **Specify the CPT for combining the evidence from conjuncts in the body of the clause**
  - Noisy-or
    - **Specify the CPT for combining the evidence from disjunctive contributions from different ground clauses with the same head**
    - **Models *"explaining away"***
  - Noisy-and and noisy-or models reduce the number of parameters learned from data

Probabilistic Inference

copy-file(test1.txt,mydir)   move-file(test1.txt,mydir)   remove-file(test1)

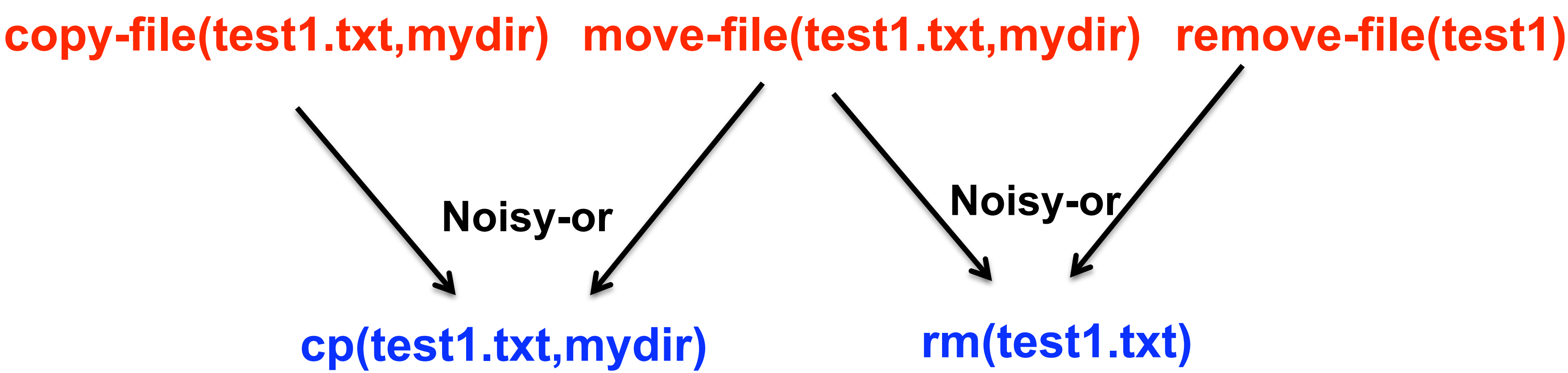
Noisy-or   Noisy-or

cp(test1.txt,mydir)   rm(test1.txt)

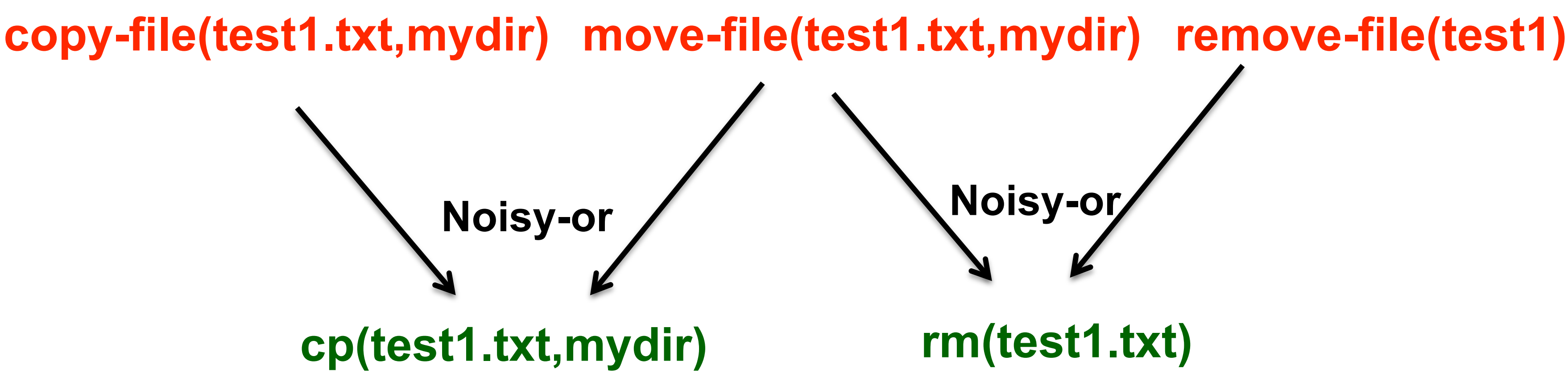
Probabilistic Inference

- **Most Probable Explanation (MPE)**
  - For **multiple plans**, compute MPE, the most likely combination of truth values to all unknown literals given this evidence
- **Marginal Probability**
  - For **single top level plan** prediction, compute marginal probability for all instances of plan predicate and pick the instance with maximum probability
  - When exact inference is intractable, *SampleSearch* ***[Gogate and Dechter, 2007]***, an approximate inference algorithm for graphical models with deterministic constraints is used

Probabilistic Inference



Probabilistic Inference



Evidence

Probabilistic Inference

Query variables

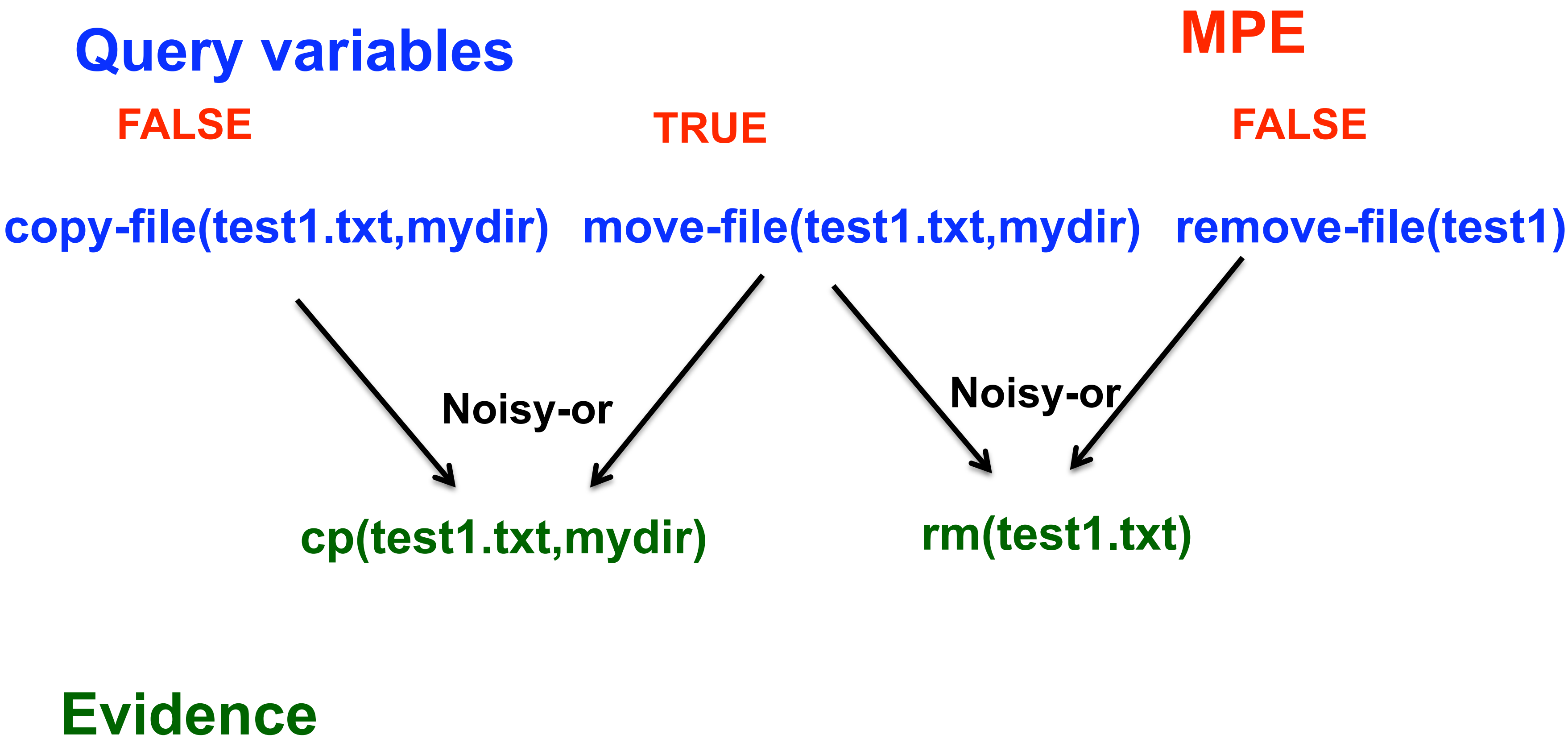
copy-file(test1.txt,mydir) **move-file(test1.txt,mydir)** **remove-file(test1)**

Noisy-or **Noisy-or**

cp(test1.txt,mydir) **rm(test1.txt)**

Evidence

Probabilistic Inference



Probabilistic Inference

Query variables

MPE

FALSE

TRUE

FALSE

copy-file(test1.txt,mydir)

move-file(test1.txt,mydir)

remove-file(test1)

Noisy-or

Noisy-or

cp(test1.txt,mydir)

rm(test1.txt)

Evidence

Parameter Learning

- Learn noisy-or/noisy-and parameters using the EM algorithm adapted for BLPs ***[Kersting and De Raedt, 2008]***
- Partial observability
  - In plan recognition domain, data is partially observable
  - Evidence is present only for *observed actions* and *top-level plans*; sub-goals, noisy-or, and noisy-and nodes are not observed
- Simplify learning problem
  - Learn noisy-or parameters ***only***
  - Used logical-and instead of noisy-and to combine evidence from conjuncts in the body of a clause

Experimental Evaluation

- Monroe *(Strategic planning)*
- Linux *(Intelligent user interfaces)*
- Story Understanding *(Story understanding)*

Monroe and Linux

[Blaylock and Allen, 2005]

❑ Task

- ✧ **Monroe** involves recognizing top level plans in an *emergency response domain* (artificially generated using HTN planner)
- ✧ **Linux** involves recognizing top-level plans based on *linux commands*
- ✧ *Single* correct plan in each example

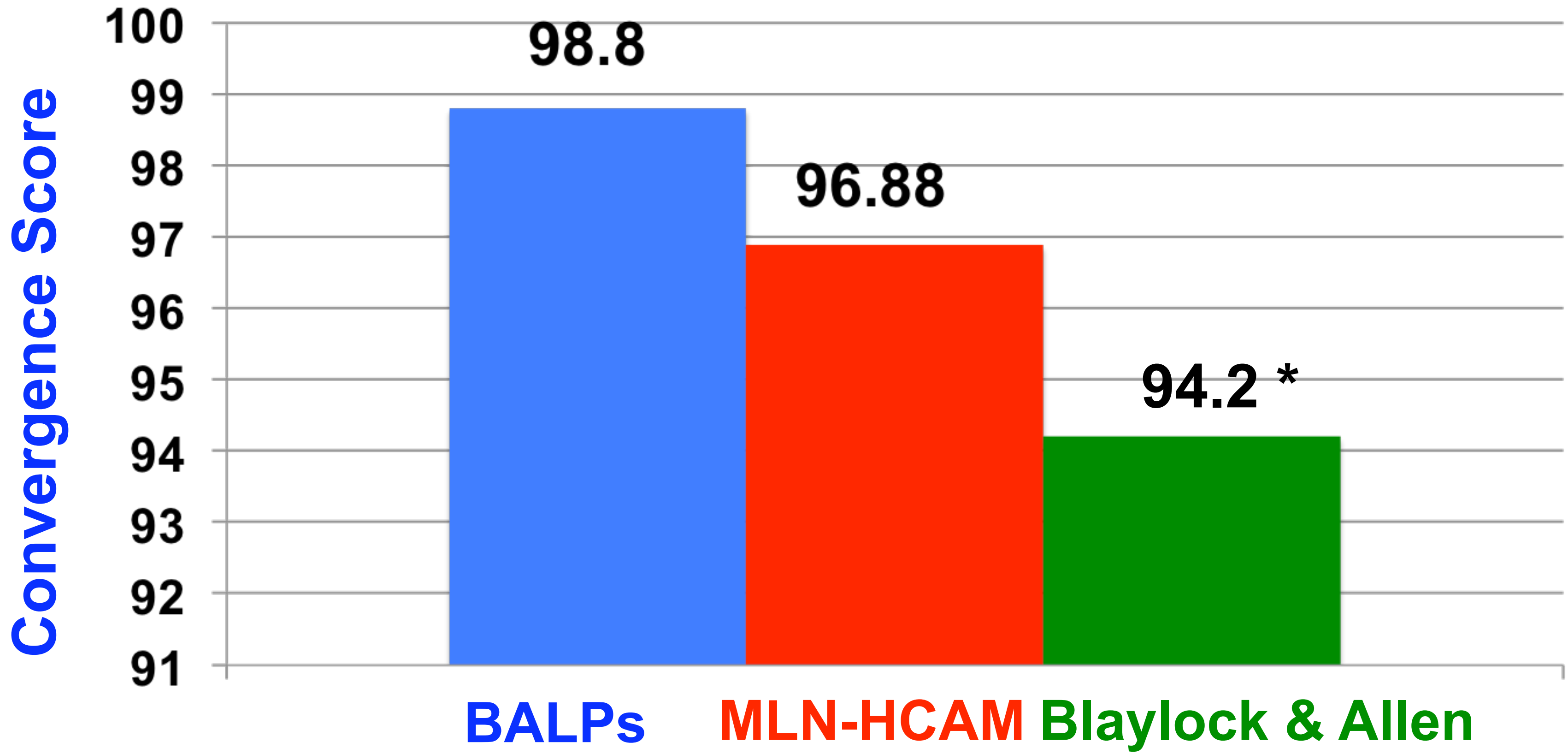
❑ Data

	No. examples	Avg. observations / example	Total top-level plan predicates	Total observed action predicates
Monroe	**1000**	**10.19**	**10**	**30**
Linux	**457**	**6.1**	**19**	**43**

Monroe and Linux

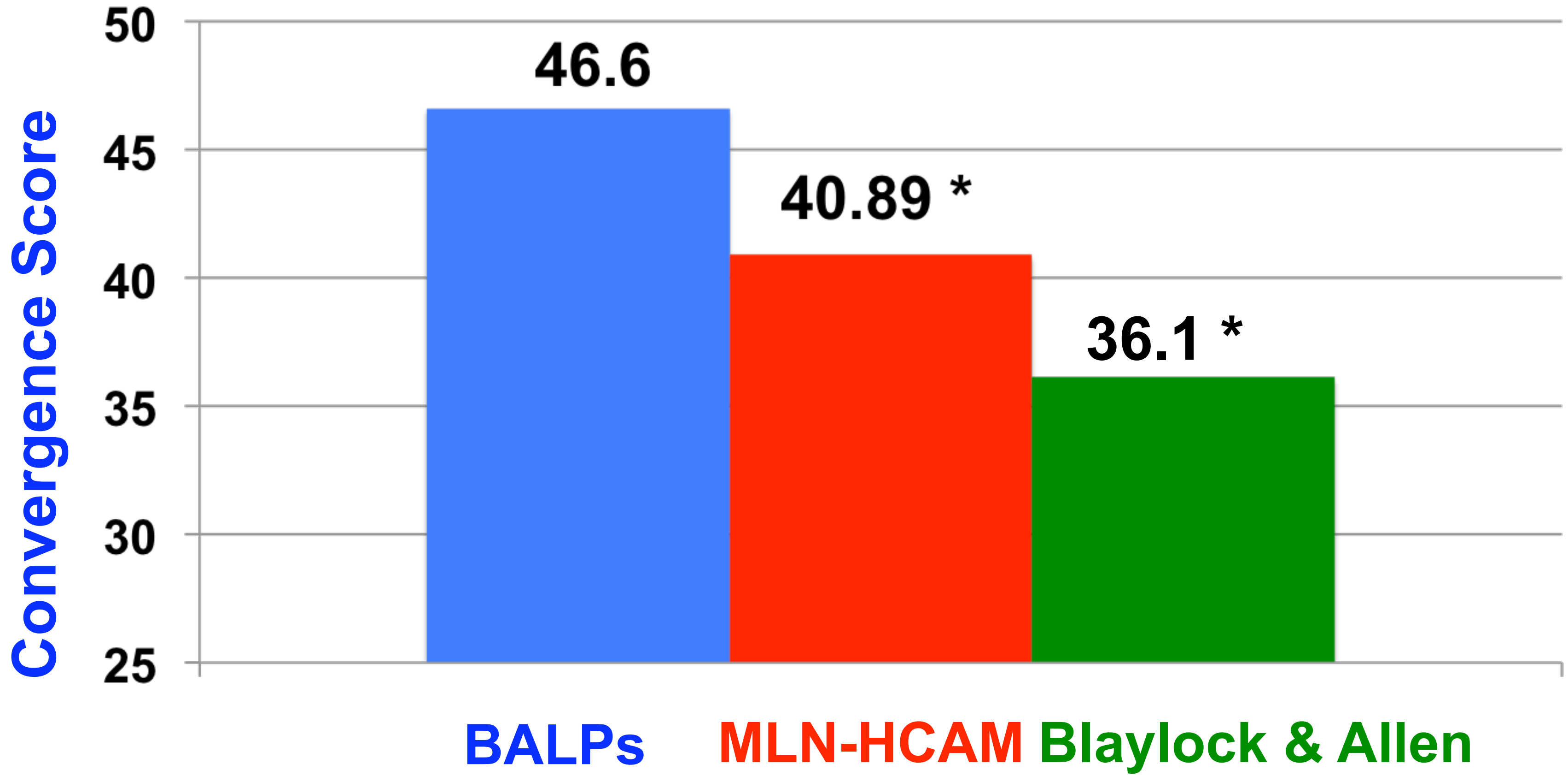
- Methodology
  - Manually encoded the knowledge base
  - Learned noisy-or parameters using EM
  - Computed **marginal probability** for plan instances
- Systems compared
  - BALPs
  - MLN-HCAM *[Singla and Mooney, 2011]*
    - **MLN-PC and MLN-HC do not run on Monroe and Linux due to scaling issues**
  - Blaylock and Allen's system *[Blaylock and Allen, 2005]*
- Performance metric
  - ***Convergence score -*** measures the fraction of examples for which the plan predicate was predicted correctly

Results on Monroe



** - Differences are statistically significant wrt BALPs*

Results on Linux

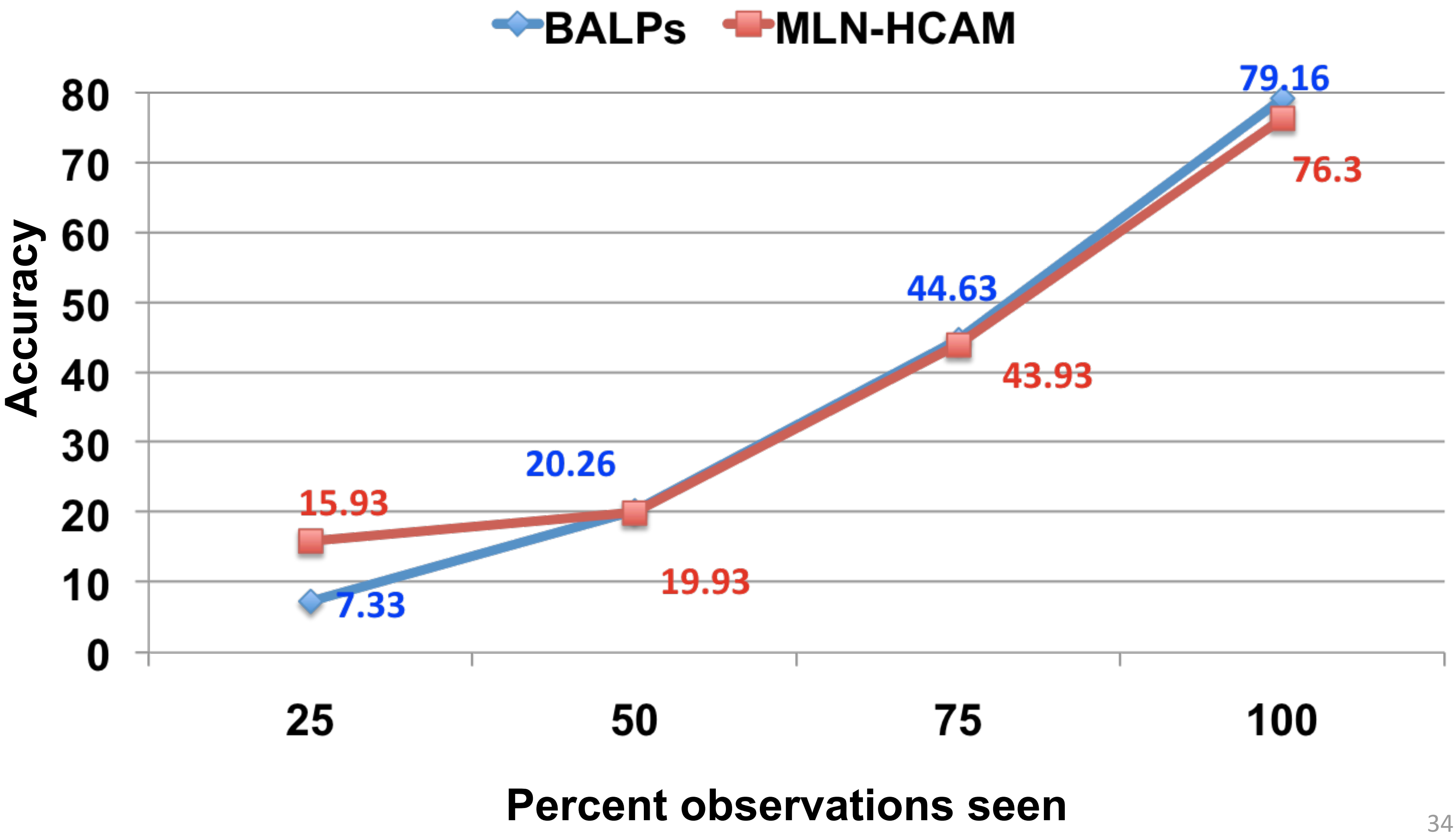


** - Differences are statistically significant wrt BALPs*

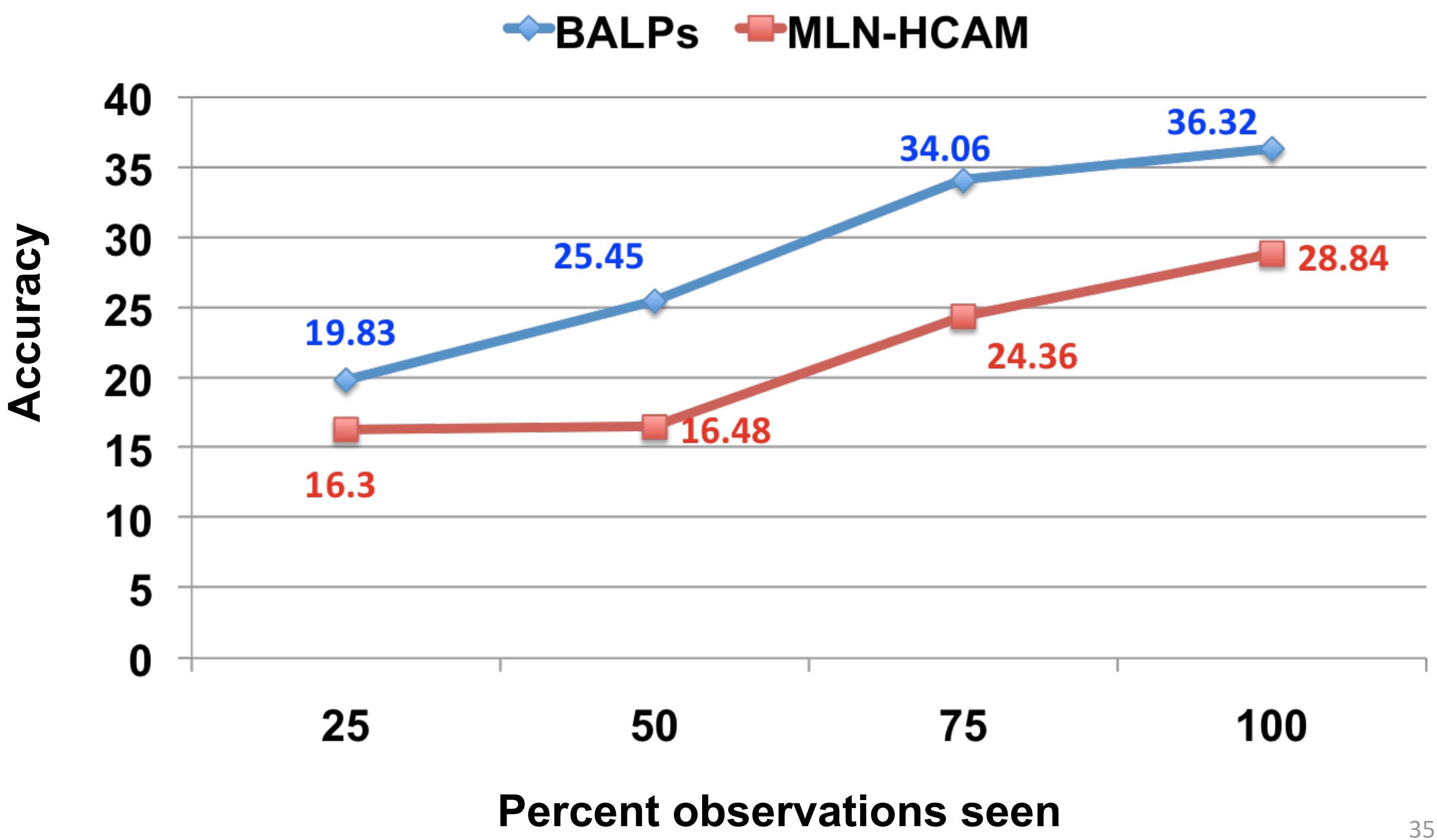
Experiments with partial observability

- Limitations of convergence score
  - Does not account for predicting the plan arguments correctly
  - Requires all the observations to be seen before plans can be predicted
- Early plan recognition with partial set of observations
  - Perform plan recognition after observing the ***first*** **25%, 50%, 75%,** and **100%** of the observations
  - Accuracy – Assign partial credit for the predicting plan predicate and a subset of the arguments correctly
- Systems compared
  - BALPs
  - MLN-HCAM ***[Singla and Mooney, 2011]***

Results on Monroe



Results on Linux



Story Understanding

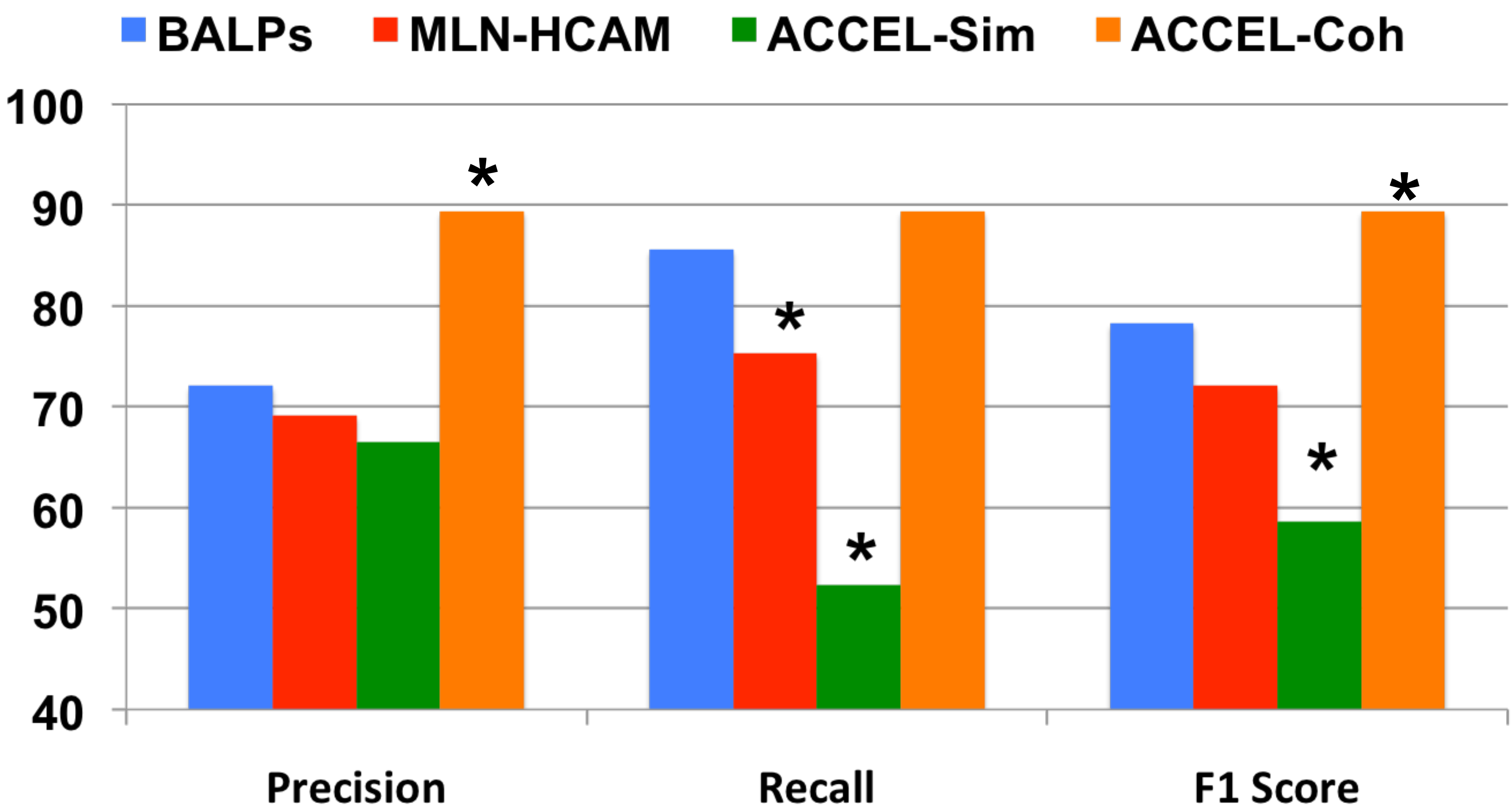
[Charniak and Goldman, 1991; Ng and Mooney, 1992]

- Task
  - Recognize character’s top level plans based on actions described in narrative text
  - *Multiple* top-level plans in each example
- Data
  - 25 examples in development set and 25 examples in test set
  - 12.6 observations per example
  - 8 top-level plan predicates

Story Understanding

- **Methodology**
  - Knowledge base was created for ACCEL ***[Ng and Mooney, 1992]***
  - Parameters set manually
    - **Insufficient number of examples in the development set to learn parameters**
  - Computed ***MPE*** to get the best set of plans
- **Systems compared**
  - BALPs
  - MLN-HCAM ***[Singla and Mooney, 2011]***
    - **Best performing MLN model**
  - ACCEL-Simplicity ***[Ng and Mooney, 1992]***
  - ACCEL-Coherence ***[Ng and Mooney, 1992]***
    - **Specific for Story Understanding**

Results on Story Understanding



** - Differences are statistically significant wrt BALPs*

Conclusion

- BALPS – Extension of BLPs for plan recognition by employing logical abduction to construct Bayesian networks
- Automatic learning of model parameters using EM
- Empirical results on all benchmark datasets demonstrate advantages over existing methods

Future Work

- Learn abductive knowledge base automatically from data

- Compare BALPs with other probabilistic logics like ProbLog ***[De Raedt et. al, 2007]***, PRISM ***[Sato, 1995]*** and Poole's Horn Abduction ***[Poole, 1993]*** on plan recognition

Questions