Multi-Level Algorithm Selection for ASP

Marco Maratea[1], **Luca Pulina**[2], and Francesco Ricca[3]

[1] DIBRIS, University of Genova
[2] POLCOMING, University of Sassari
[3] DeMaCS, University of Calabria

13th International Conference on Logic Programming and Non-monotonic Reasoning

Lexington, KY, USA September 27-30, 2015

Context

Answer Set Programming (ASP)

- Declarative logic programming paradigm
- Applications in several fields
  - → Artificial Intelligence, Knowledge Representation & Reasoning
  - → Information Integration, Bioinformatics... industrial ones!

Context

Answer Set Programming (ASP)

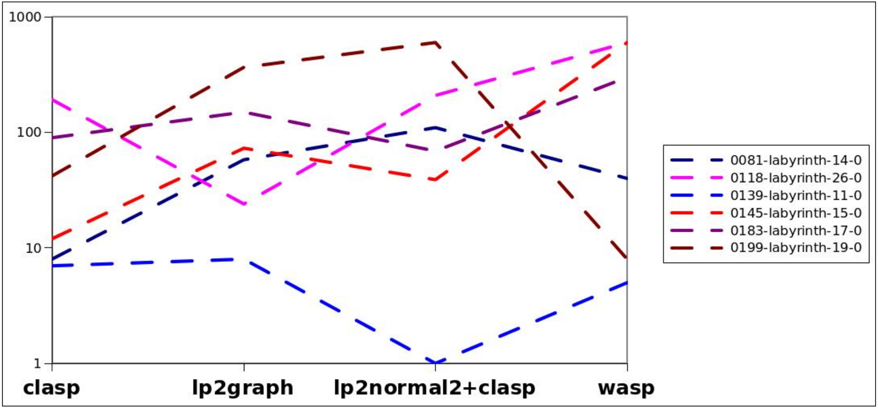
- Declarative logic programming paradigm
- Applications in several fields
  - → Artificial Intelligence, Knowledge Representation & Reasoning
  - → Information Integration, Bioinformatics... industrial ones!

Robust and efficient implementations

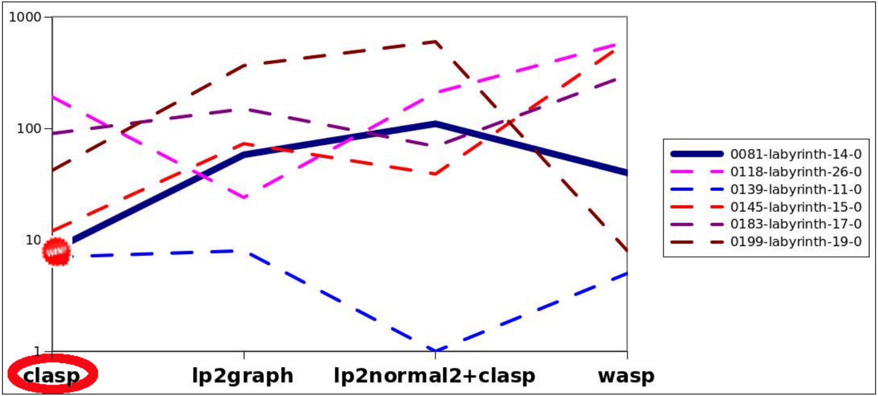
- CLASP, CMODELS, DLV,
- LP2SAT, IDP, WASP, etc.

Continuous improvement in ASP competitions

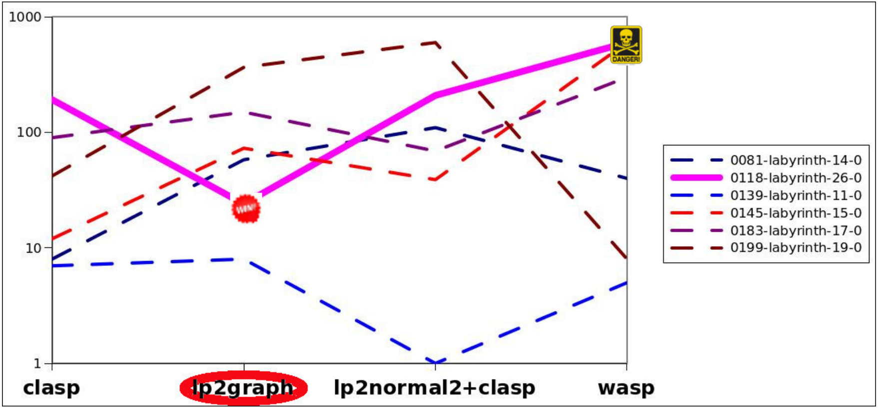
Are state-of-the-art solvers robusts?



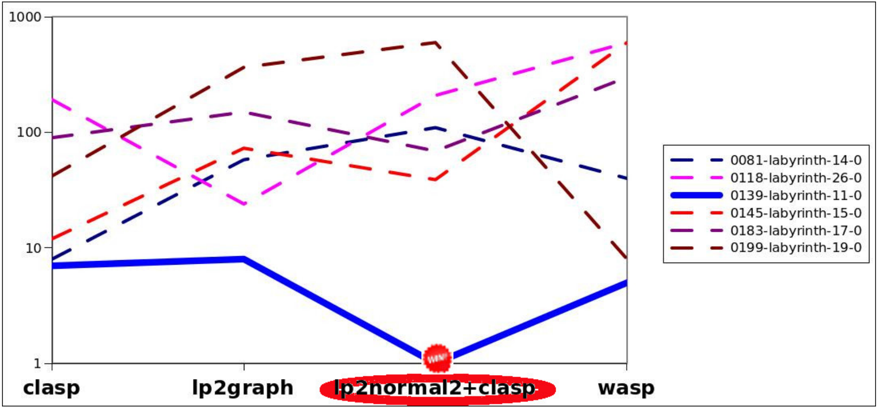
Are state-of-the-art solvers robusts?



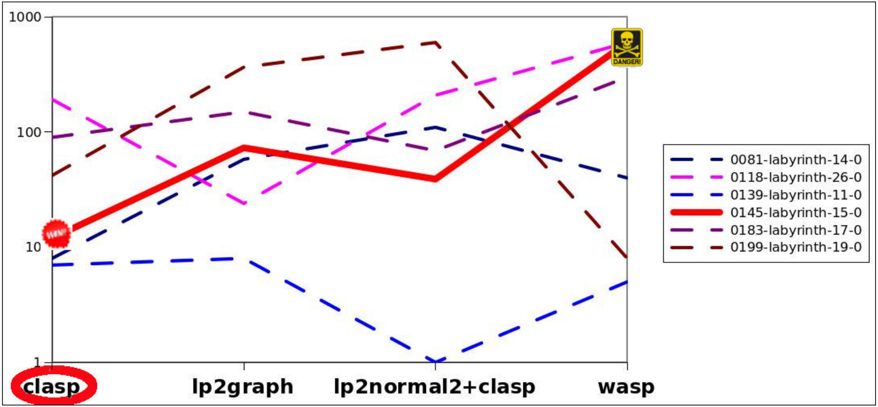
Are state-of-the-art solvers robusts?



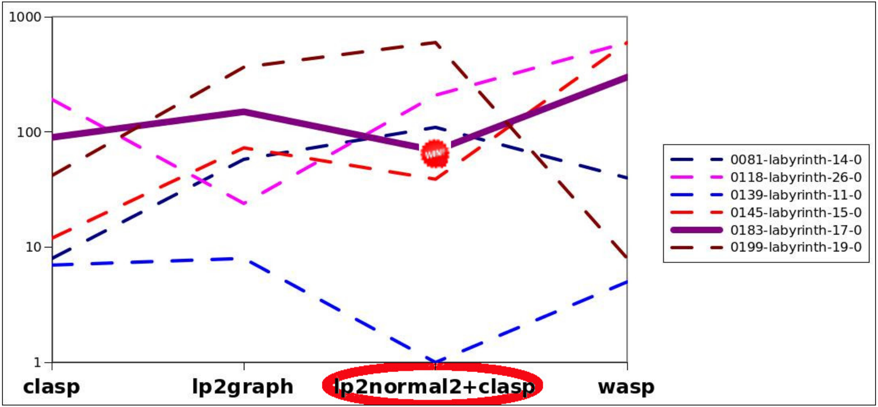
Are state-of-the-art solvers robusts?



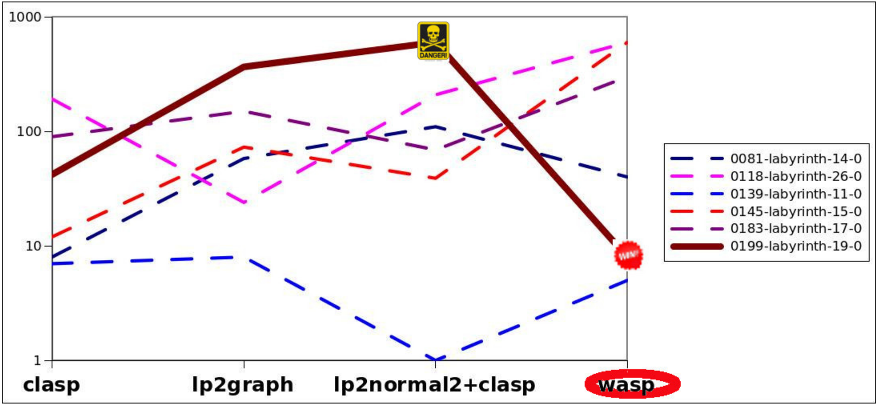
Are state-of-the-art solvers robusts?



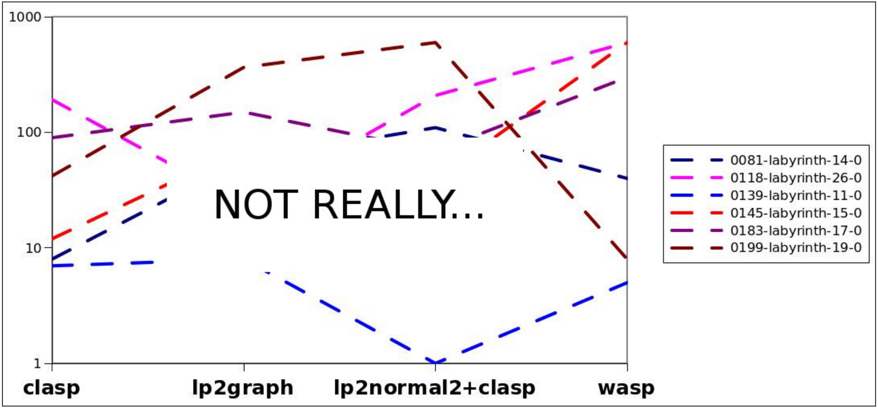
Are state-of-the-art solvers robusts?



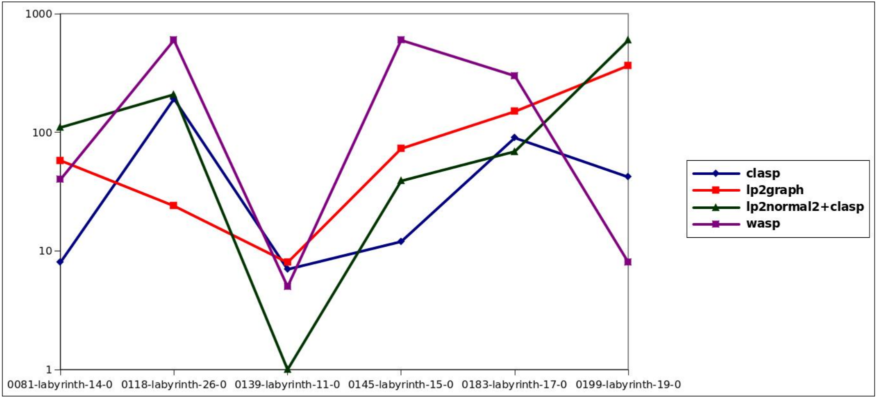
Are state-of-the-art solvers robusts?



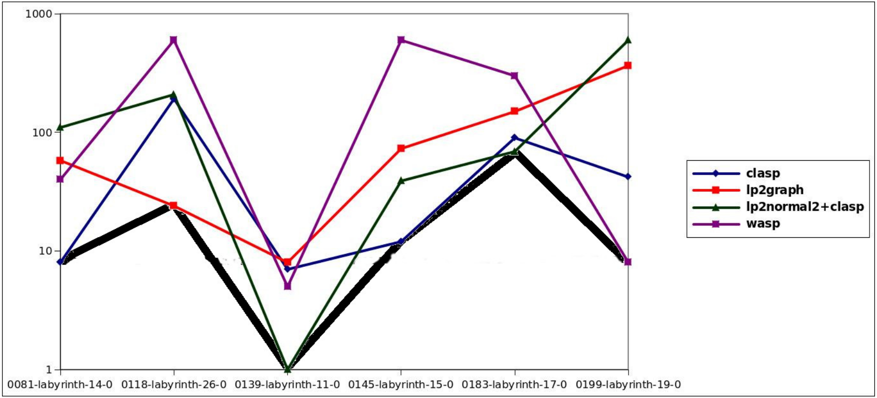
Are state-of-the-art solvers robusts?



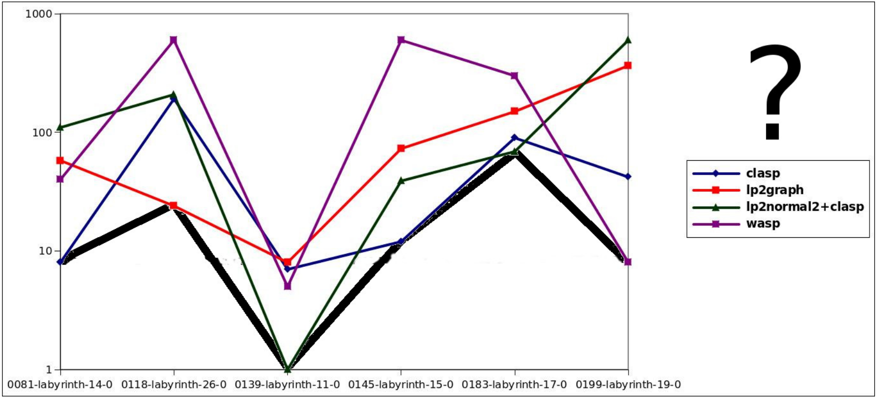
Goal: a robust ASP system



Goal: a robust ASP system



Goal: a robust ASP system



Contribution

Goal

Obtain robust performances from state-of-the-art ASP solvers

Contribution

Goal

Obtain robust performances from state-of-the-art ASP solvers

ME-ASP^{ML}: Extension of the multi-engine ASP system ME-ASP

- Exploits problem information before each level of computation.
  - Leverages features related to both non-ground and ground input.
  - Identifies different classes of non-ground programs.
  - (Possibly) identifies a class of programs where a specific grounder is the most promising.
- Improve solver “prediction” using non-ground features.
- Satisfies robustness requirements
  - It is always more effective than each single engine.
  - Its engine selection policy is always more accurated than a random policy.

Outline

1. Design and Implementation of ME-ASP^{ML}
2. Experimental Analysis
3. Conclusion & Future Work

Two approaches to yield a robust solver

Brute force

Given *m* ASP instances and *n* solvers (engines)

- Run each engine on a separate machine
- Stop all the engines as soon as one solves the instance, or all the engines exhaust resources
- Continue with the next instance (if any)

Two approaches to yield a robust solver

Brute force

Given *m* ASP instances and *n* solvers (engines)

- Run each engine on a separate machine
- Stop all the engines as soon as one solves the instance, or all the engines exhaust resources
- Continue with the next instance (if any)

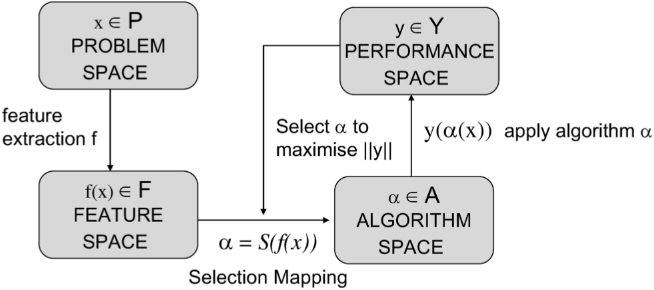
Intelligence

Understand which engine is best for which instance

- Fairly old idea: asset allocation in economics
- "The Algorithm Selection Problem – Abstract Models", Rice 1974
- Algorithm portfolios: SATzilla (SAT), AQME (QBF), Claspfolio (ASP), ...

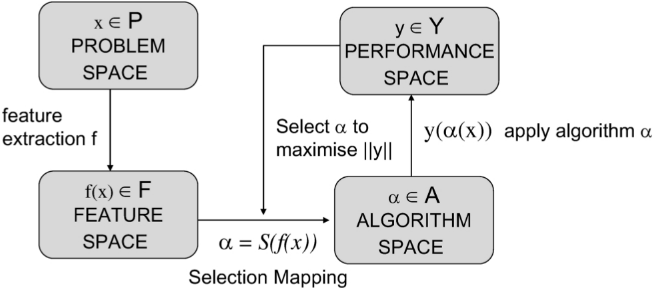
Algorithm Selection Framework at Work

Rice, 1974



Algorithm Selection Framework at Work

Rice, 1974



Key ingredients of the recipe:

- *P*: input programs
- *F*: cheap-to-compute syntactic features
- *A*: pool of solvers
- *Y*: computation of the solution within a CPU time limit
- *S* is determined leveraging machine learning algorithms

Algorithm Selection in ASP

Applied successfully to ASP Solving

- Claspfolio
  - Based on *regression* and *dynamic* features
  - White box: Combines variants of clasp
  - Winner of several ASP competitions
- ME-ASP
  - Based on *classification*
  - *Cheap to compute static features*
  - Black Box: Combines several solvers

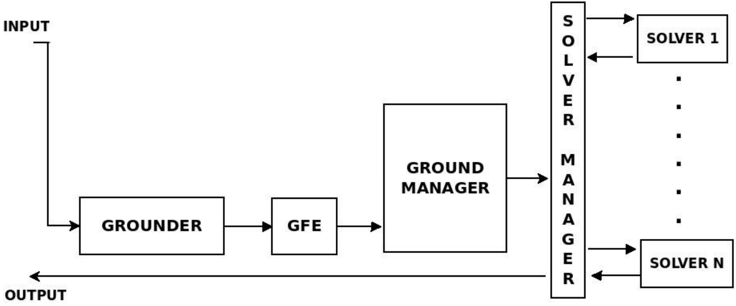
Algorithm Selection in ASP

Applied successfully to ASP Solving

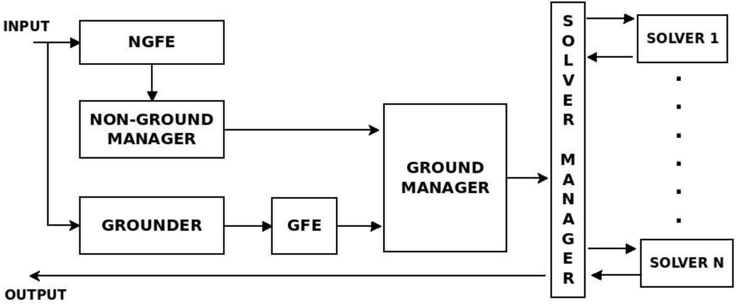
- Claspfolio
  - Based on *regression* and *dynamic* features
  - White box: Combines variants of clasp
  - Winner of several ASP competitions
- ME-ASP
  - Based on *classification*
  - *Cheap to compute static features*
  - Black Box: Combines several solvers

Application limited to Solving Step!

Architecture of ME-ASP



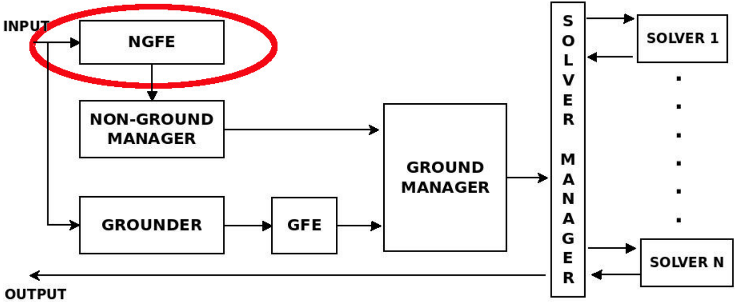
Architecture of ME-ASP^{ML}



Multi-level approach

- Classify programs w.r.t. non-ground features
  - Can be combined with selection of grounder
- Apply solver selection on each sub-class

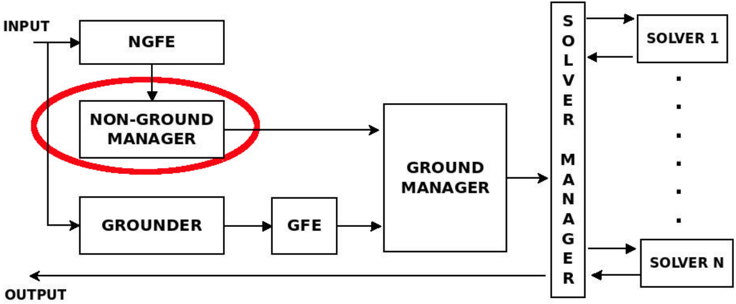
Architecture of ME-ASP^{ML} – Non-Ground Feature Extraction



It computes features from the input (non-ground) program

- e.g., Head-Cycle Free components, presence of queries, stratification property, ...

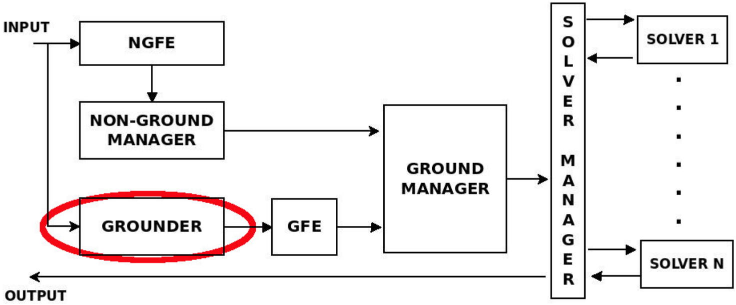
Architecture of ME-ASP^{ML} – Non-Ground Manager



- Identifies the “class” of the input ASP program.
- *Can choose the most promising grounder.*

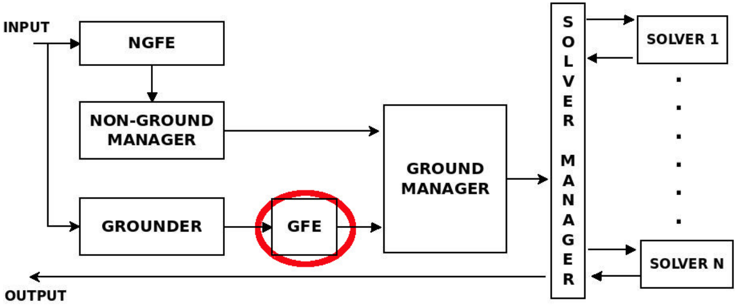
  → see (Maratea, Pulina & Ricca, 2013)

Architecture of ME-ASP^{ML} – Grounder



- Takes as input the non-ground ASP program.
- Returns the related grounded instance.

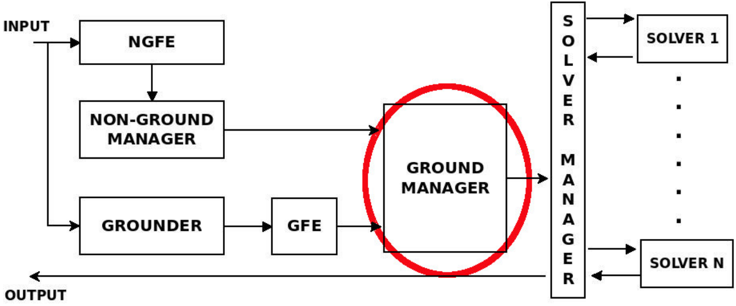
Architecture of ME-ASP^{ML} – Ground Feature Extractor



Aims at computing the syntactic features of the input ground program.

- Features detailed in (Maratea, Pulina & Ricca, 2014).
- ASPCore 2.0 specific features (e.g., number of choice rules, number of aggregates, number of weak constraints, ...).

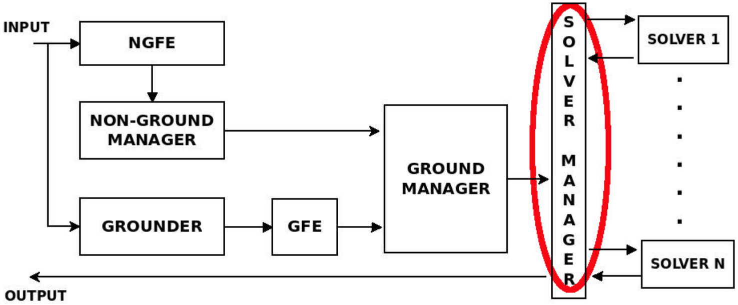
Architecture of ME-ASP^{ML} – Ground Manager



Predicts the solver to run.

- Given the input received by NON−GROUND MANAGER, it selects the proper inductive model (related to the class found in NGM).
- Given the features computed in GFE, it outputs to SOLVER MANAGER the name of the predicted solver.

Architecture of ME-ASP^{ML} – Solver Manager



- It manages the interaction with the solvers.
- It returns the result obtained using the chosen solver.

Current Implementation (1/2)

- Instances and encodings involved in the Fifth ASP Competition
  - **8572 instances** (50% for training purpose, the remaining ones for testing).
    $\rightarrow$ the full list is available at
    `www.mat.unical.it/ricca/downloads/measpmlts.tar.gz`
- NON-GROUND MANAGER
  - Inductive model: list of if-then-else rules obtained running the PART decision list generator on the training instances.
  - Five program classes: C_i, $i \in \{1, \ldots, 4\}$ and Q.
- GROUNDER: GRINGO ver. 4

Current Implementation (2/2)

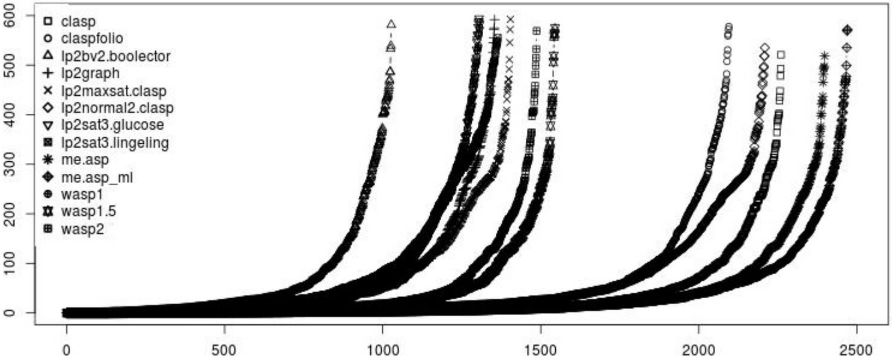
GROUND MANAGER

- Multinomial classification with k-Nearest Neighbors.
- Four different inductive models ($C_1, \ldots, C_4$) related to the program classes + Queries (Q)
  - Training sets composed of the values related to the features computed in GFE
  - Labels correspond to the best solver – in terms of CPU time – on the given instance.

Current Implementation – Solvers involved

Solver	C_1	C_2	C_3	C_4	Q
CLASP (Drescher et al., 2008)	✓	✓	✓	✓	–
LP2BV2+BOOLECTOR (Nguyen et al., 2011)		✓	–	–	–
LP2GRAPH (Gebser et al., 2014)	✓		–	–	–
LP2MAXSAT+CLASP (Bomanson & Janhunen, 2013)		✓		–	–
LP2NORMAL2+CLASP (Bomanson & Janhunen, 2013)	✓	✓	✓	✓	–
WASP1 (Alviano et al., 2013)			✓		
WASP1.5 (Alviano et al., 2013)					✓
WASP2 (Alviano et al., 2013)	✓	✓		–	–

Experiments



- More than 4000 instances
- Time limit: 600 CPU seconds; Memory limit: 2 GB

Conclusion & Future Work

- A multiengine solver is a robust alternative to current state-of-the-art ASP solvers.
- Exploiting a set of inexpensive syntactic features related to non-ground programs can improve the choice of the best engine to run on a given ASP program.

ME-ASP^{ML} is available for download at
www.mat.unical.it/ricca/downloads/measpml.tar.gz

Conclusion & Future Work

- A multiengine solver is a robust alternative to current state-of-the-art ASP solvers.
- Exploiting a set of inexpensive syntactic features related to non-ground programs can improve the choice of the best engine to run on a given ASP program.

ME-ASP^{ML} is available for download at
`www.mat.unical.it/ricca/downloads/measpml.tar.gz`

Future Work

- Deeper study of crucial features
- Dynamic adaptive mechanisms
- Implementation of grounder selection
- Inductive models for the choice of grounder+model generator

Thank you!