Hydra:
Automatically Configuring Algorithms for Portfolio-Based Selection

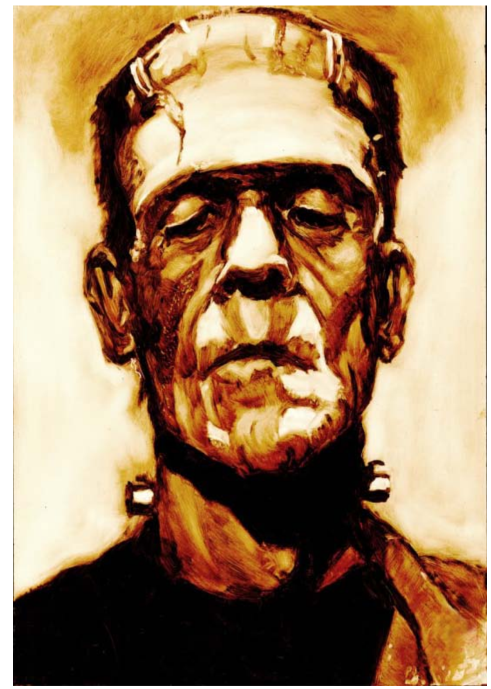
Lin Xu, Holger H. Hoos, Kevin Leyton-Brown

Department of Computer Science
University of British Columbia

Two automated algorithm design ideas

SATzilla
[Xu, Hutter, Hoos, Leyton-Brown, 2007; 2008]
portfolio-based algorithm selection

SATenstein
[KhudaBukhsh, Xu, Hoos, Leyton-Brown, 2009]
algorithm design via automatic configuration



Two automated algorithm design ideas

SATzilla

[Xu, Hutter, Hoos, Leyton-Brown, 2007; 2008]

portfolio-based algorithm selection

Exploit per-instance variation between solvers using learned runtime models

- **practical:** e.g., won 10 medals in 2007, 2009 SAT competitions
- **fully automated:** requires only cluster time rather than human design effort

Key drawback:

- requires a set of **strong, relatively uncorrelated** candidate solvers
- **can't be applied** in domains for which such solvers do not exist

Some particularly related work: [Rice, 1976]; [Leyton-Brown, Nudelman & Shoham, 2003; 2009]; [Guerri & Milano, 2004]; [Nudelman, Leyton-Brown, Shoham & Hoos, 2004]

Two automated algorithm design ideas

- **Instead of manually exploring a design space, build a highly-parameterized algorithm and then configure it automatically**
- Can find **powerful, novel designs**
  - matched or outperformed existing SLS algorithms on six SAT domains
- But: **only produces single algorithms** designed to perform well on the entire training set

SATenstein

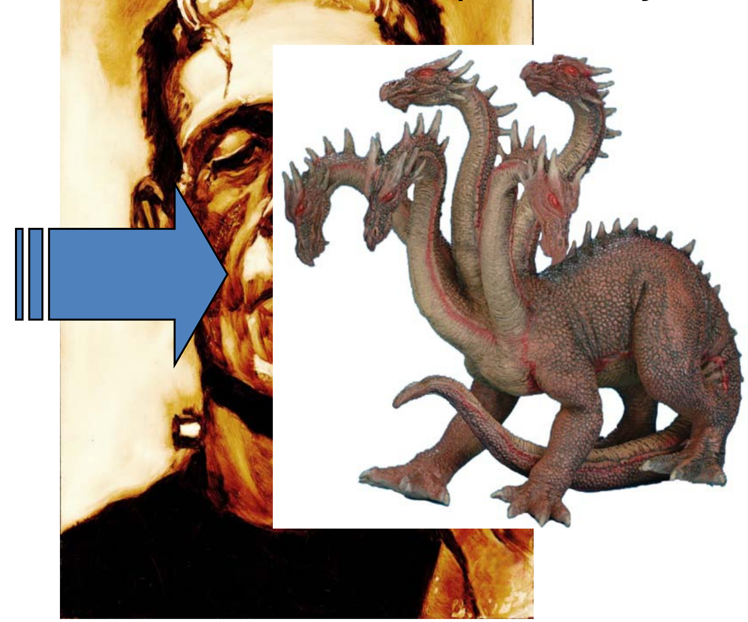
[KhudaBukhsh, Xu, Hoos, Leyton-Brown, 2009]

algorithm design via automatic configuration

Some particularly related work: [Gratch & Dejong, 1992]; [Fukunaga, 2002]; [Balaprakash, Birattari & Stutzle, 2007]; [Hutter, Babic, Hoos & Hu, 2007]; [Ansotegui, Sellmann & Tierney, 2009]; [Hutter, Hoos, Stutzle & Leyton-Brown, 2009]

Two automated algorithm design ideas

Hydra
automatic portfolio synthesis



Starting from a **single parameterized algorithm**, automatically find a set of **uncorrelated configurations** that can be used to build a **strong portfolio.**

Xu, Hoos, Leyton-Brown. Hydra: Automatically Configuring Algorithms for Portfolio-Based Selection

Plan of This Talk

Background

- ***SATzilla: Portfolio-Based Algorithm Selection***
- ***SATenstein: Algorithm Configuration as Design***

Portfolio Synthesis

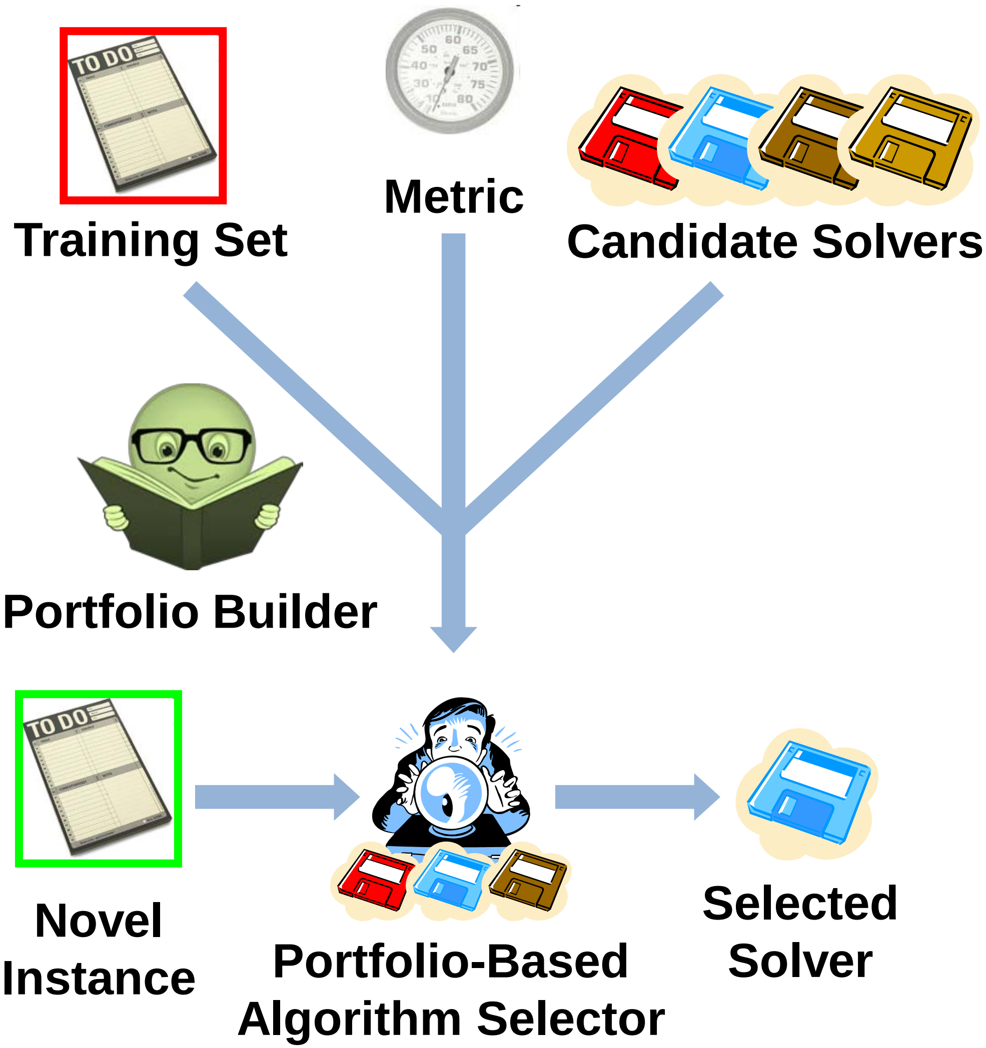
- *Related Work*
- *Hydra*

Experimental Results

Conclusions and Future Work

SATzilla: Portfolio-Based Algorithm Selection

[Xu, Hutter, Hoos, Leyton-Brown, 2007; 2008]



- **Given:**
  - **training set of instances**
  - **performance metric**
  - **candidate solvers**
  - **portfolio builder** ***(incl. instance features)***
- Training:
  - collect performance data
  - portfolio builder learns predictive models
- At Runtime:
  - predict performance
  - select solver

Xu, Hoos, Leyton-Brown. Hydra: Automatically Configuring Algorithms for Portfolio-Based Selection

SATenstein: Automated Algorithm Design

[KhudaBukhsh, Xu, Hoos, Leyton-Brown, 2009]

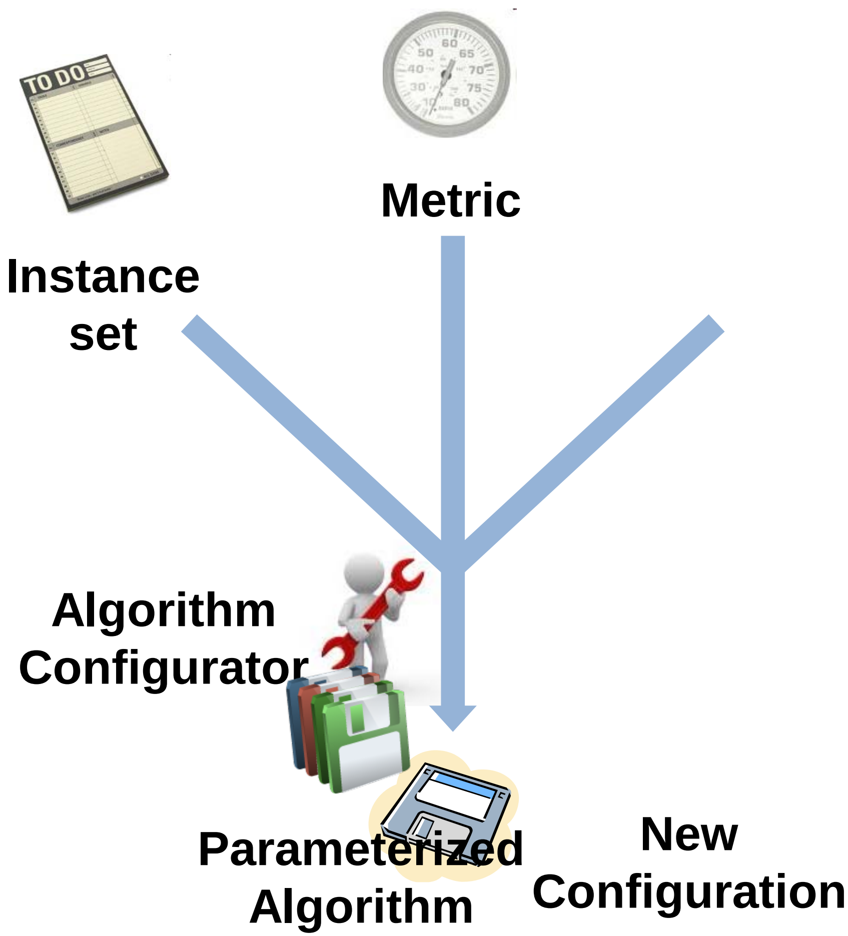
Existing Algorithm Components

Domain Expert

Parameterized Algorithm

- **Designer creates highly-parameterized algorithm from existing components**
- Given:
  - training set of instances
  - performance metric
  - parameterized algorithm
  - algorithm configurator
- Configure algorithm:
  - run configurator on training instances
  - output is a configuration that optimizes metric

SATenstein: Automated Algorithm Design



- Designer creates highly-parameterized algorithm from existing components
- Given:
  - training set of instances
  - performance metric
  - parameterized algorithm
  - algorithm configurator
- Configure algorithm:
  - run configurator on training instances
  - output is a configuration that optimizes metric

Xu, Hoos, Leyton-Brown. Hydra: Automatically Configuring Algorithms for Portfolio-Based Selection

Plan of This Talk

Background
- *SATzilla: Portfolio-Based Algorithm Selection*
- *SATenstein: Algorithm Configuration as Design*

Portfolio Synthesis
- ***Related Work***
- ***Hydra***

Experimental Results

Conclusions and Future Work

Related Work

- Algorithm synthesis; portfolios and online algorithm selection
  [Minton 1993]; [Huberman, Lukose & Hogg 1997]; [Howe et al, 1999]; [Gomes & Selman 2001]; [Carchrae & Beck 2005]; [Gagliolo & Schmidhuber 2006]; [Streeter, Golovin & Smith 2007]; [Roberts & Howe, 2007]; [Gaspero & Schaerf 2007]; [Monette, Deville & van Hentenryck 2009]

- **Two proposals for synthesis of selection-based portfolios:**
  1. **"Boosting as a Metaphor for Algorithm Design"** [L-B et al., 2003; 2009]
  2. **Stochastic Offline Programming** [Malitsky & Sellmann, 2009]
     - partition instances into *k* clusters based on features
     - find best-performing algorithm for each cluster

     ⇒ assumes that all algorithms repeatedly (1) sample from a distribution over heuristics; (2) use the sampled heuristic for one search step

     ⇒ best-performing algorithms identified using a custom optimization method

     ⇒ *our goal is to construct an entirely general method for portfolio synthesis*

- **CP-Hydra** [O'Mahony, Hebrard, Holland, Nugent, & O'Sullivan, 2008]
  – selection-based portfolio for constraint programming

Boosting as a Metaphor for Algorithm Design

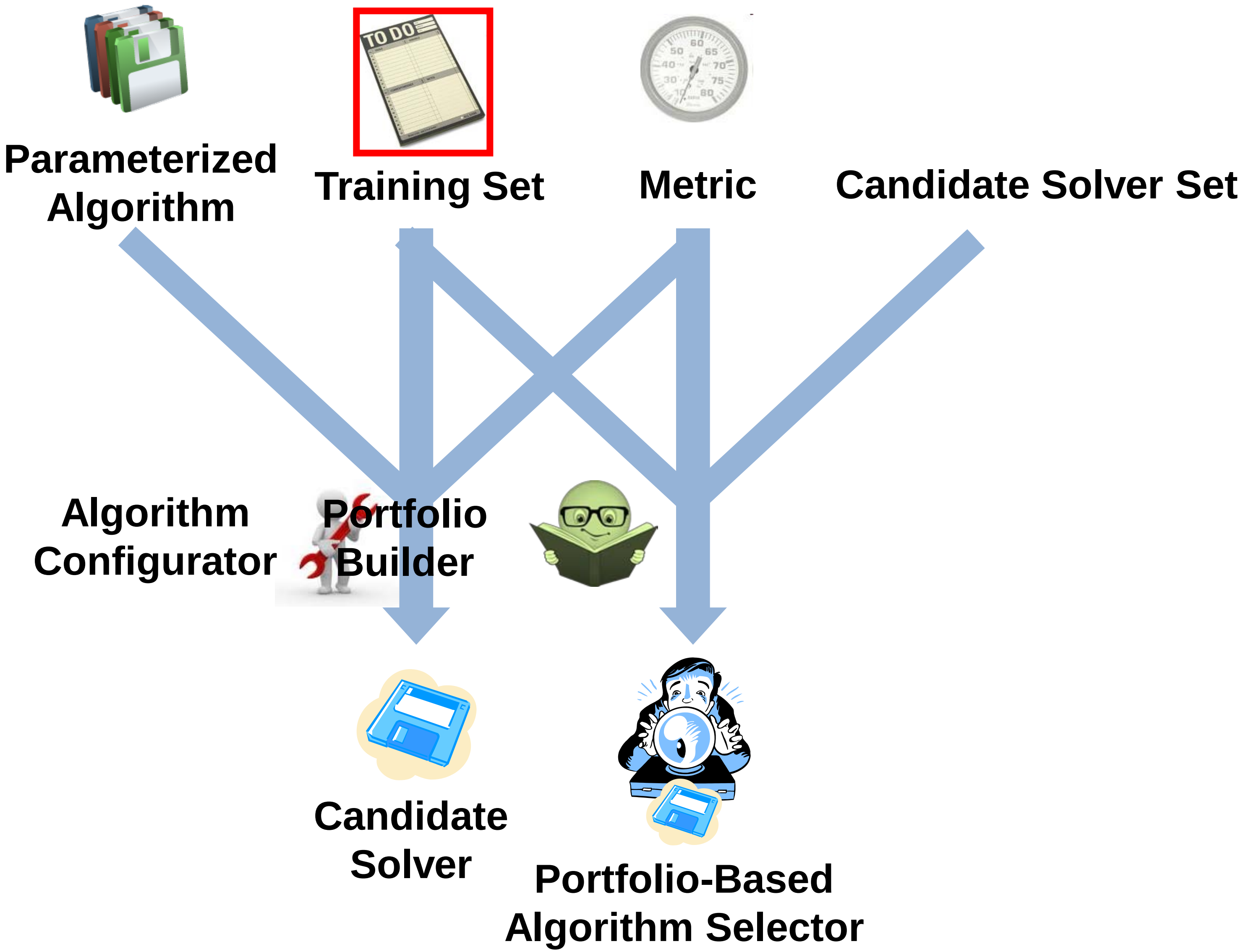
[Leyton-Brown, Nudelman, Andrew, McFadden, Shoham, 2003]; [Leyton-Brown, Nudelman, Shoham, 2009]

- **Core idea**
  - **re-weight instance distribution** to emphasize problems on which an existing portfolio *P* performs poorly
- **Interpretation as an automatic procedure:**
  - generate a new distribution *D* that is hard for *P*
  - find a new solver **maximizing average performance** on *D*
- **We intended to implement this procedure. But:**
  - discovered examples in which the algorithm with best average performance **does not improve the portfolio**
  - thus, the portfolio synthesis procedure can stagnate, even when other, helpful algorithms exist

Hydra: Dynamic Performance Metric

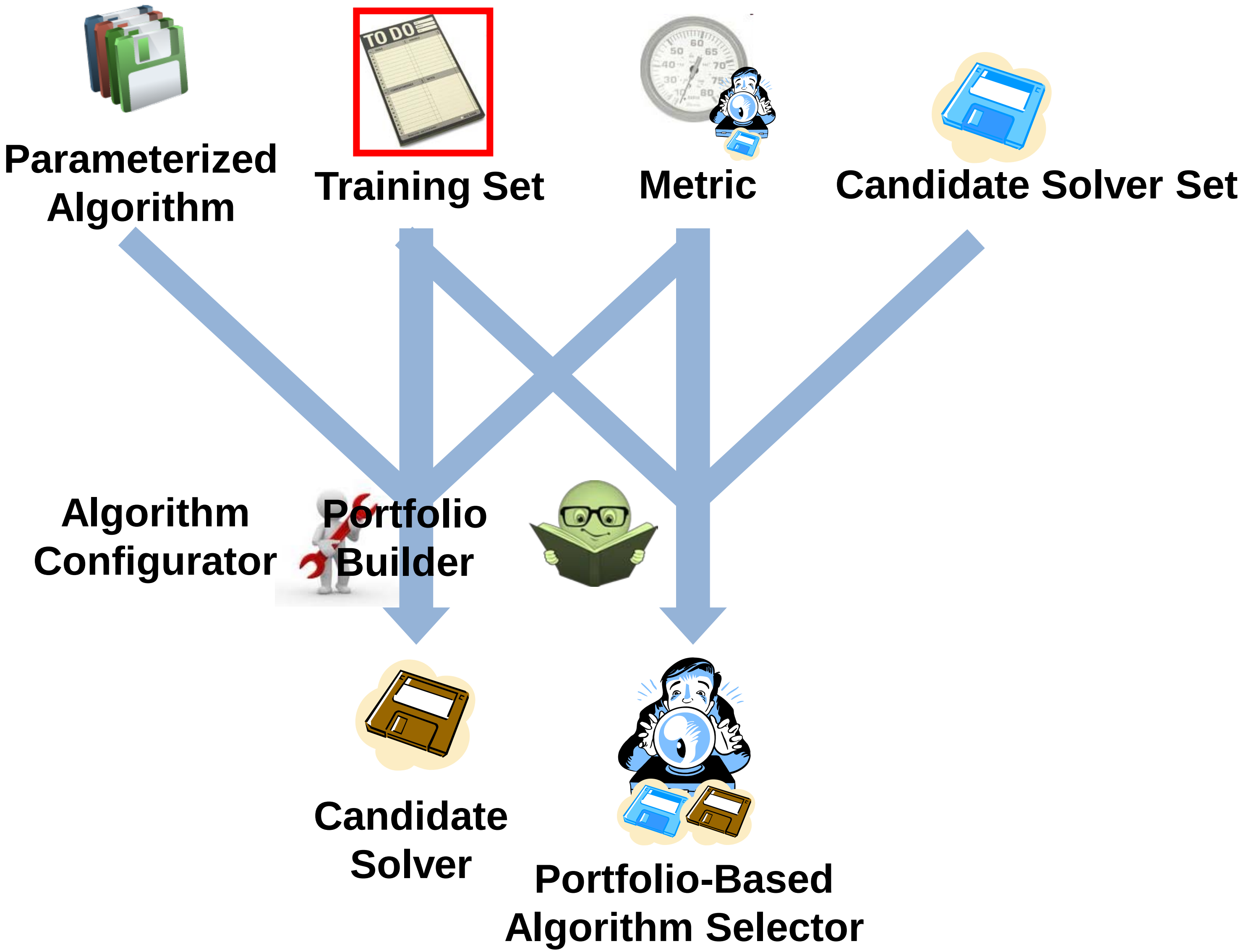
- **Avoid stagnation via a dynamic performance metric:**
  - return performance of *s* when *s* outperforms *P*
  - return performance of *P* otherwise
- **Intuitively:** *s* is scored for its **marginal contribution** to *P*
- **This metric is given to an off-the-shelf configurator**, which optimizes it to find a new configuration *s**
- **Thus, we retain the same core idea as "boosting"**
  - build a new algorithm that explicitly aims to **improve upon an existing portfolio**
- **Contrast with Stochastic Offline Programming:**
  - algorithms target **sets of instances having very different features**
  - these feature differences can be irrelevant to algorithm performance

Hydra Procedure: Iteration 1



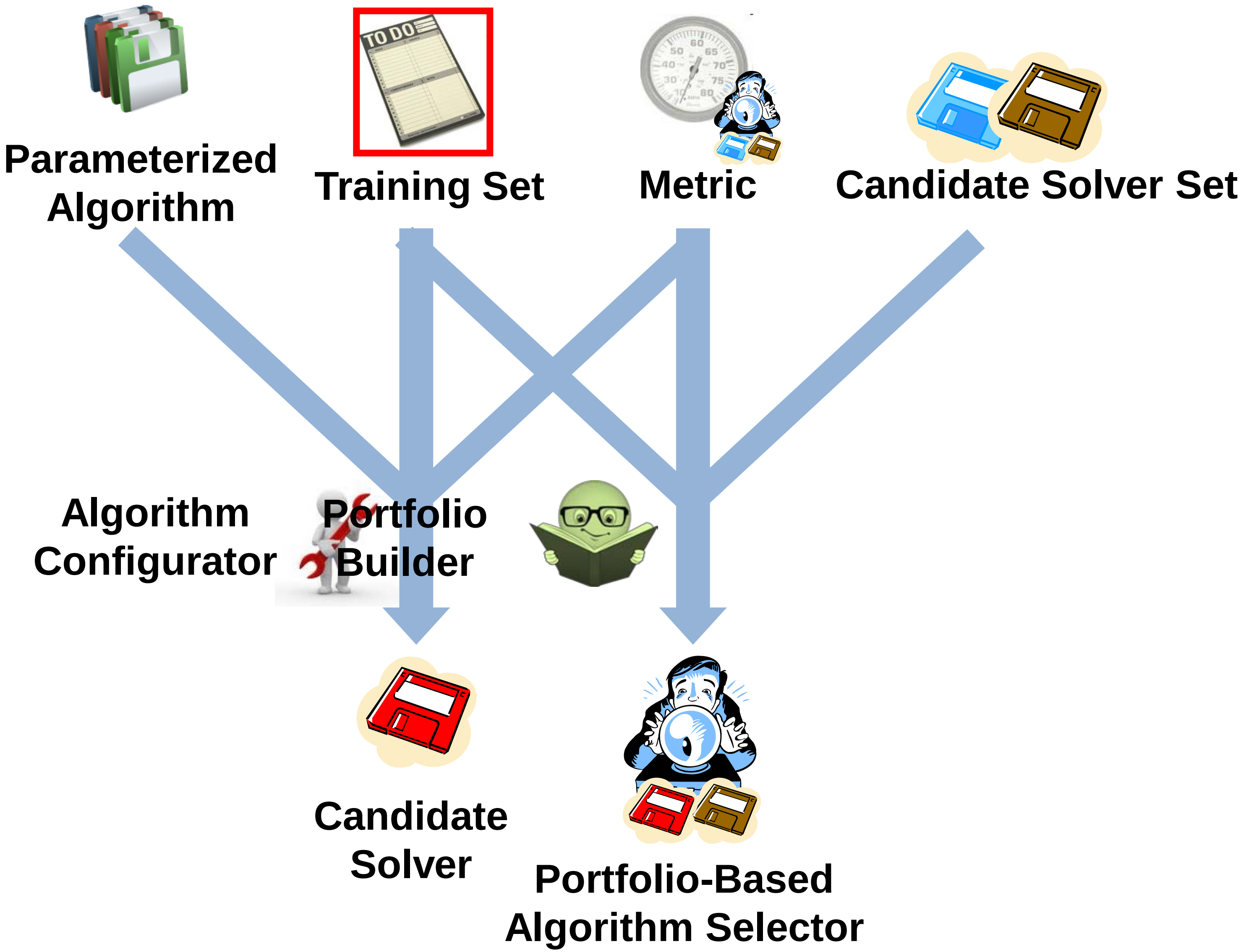
Xu, Hoos, Leyton-Brown. Hydra: Automatically Configuring Algorithms for Portfolio-Based Selection

Hydra Procedure: Iteration 2

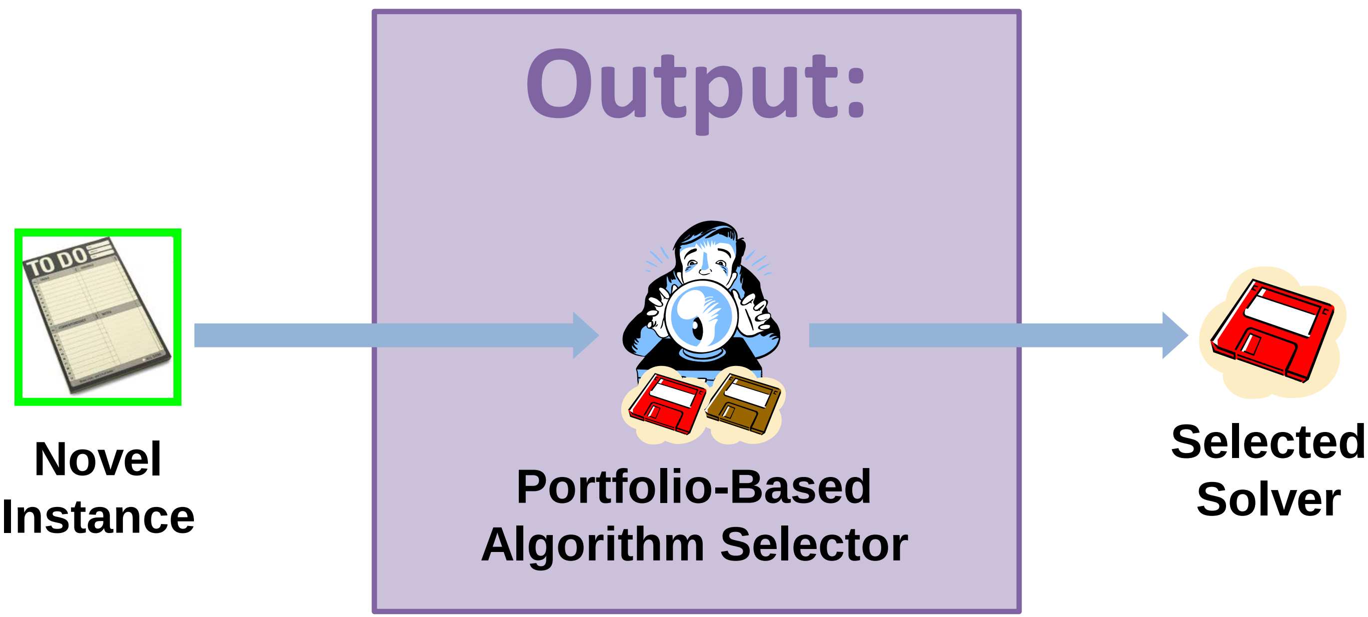


Xu, Hoos, Leyton-Brown. Hydra: Automatically Configuring Algorithms for Portfolio-Based Selection

Hydra Procedure: Iteration 3



Hydra Procedure: After Termination



Plan of This Talk

Background
- *SATzilla: Portfolio-Based Algorithm Selection*
- *SATenstein: Algorithm Configuration as Design*

Portfolio Synthesis
- *Related Work*
- *Hydra*

Experimental Results

Conclusions and Future Work

Problem Domain

- **Even though Hydra is most useful in other domains, we evaluated it on SAT.**

- **High bar for comparison**
  - strong state-of-the-art solvers
  - portfolio-based solvers already successful
  - ⇒ to be able to argue that Hydra does well, **we want to compare to a strong portfolio**

- **Pragmatic benefits**
  - a wide variety of interesting datasets
  - existing instance features
  - SATenstein is a suitable configuration target

Experimental Setup: Hydra's Inputs

- **Portfolio Builder:** SATzilla framework
  [Xu, Hutter, Hoos, Leyton-Brown, 2008]

- **Parameterized Solver:** SATenstein-LS
  [KhudaBukhsh, Xu, Hoos, Leyton-Brown, 2009]

- **Algorithm Configurator:** FocusedILS 2.3
  [Hutter, Hoos, Leyton-Brown, 2009]

- **Performance Metric:** Penalized average runtime (PAR)

- **Instance Sets:**
  - 2 from SATenstein paper
    [KhudaBukhsh, Xu, Hoos, Leyton-Brown, 2009]
  - 2 from previous SAT competitions

Experimental Setup: Challengers

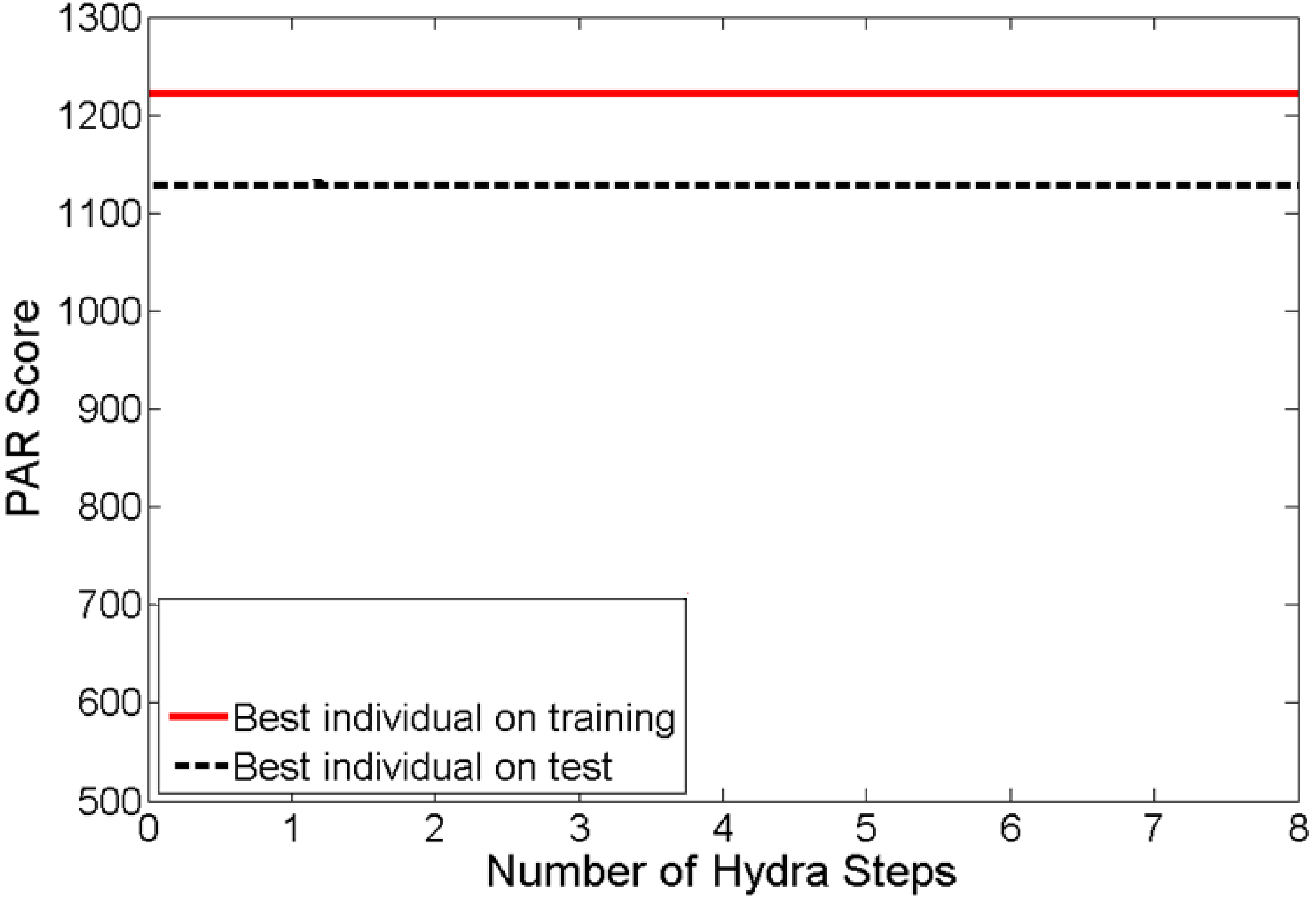
- **Individual state-of-the-art solvers**
  - 11 **manually-crafted** SLS solvers
    - all 7 SLS winners of any SAT competition 2002 – 2007
    - 4 other prominent solvers
  - 6 **SATenstein solvers**
- **Also considered portfolios of challengers**
  - used same **portfolio builder** (SATenstein)

Performance Summary

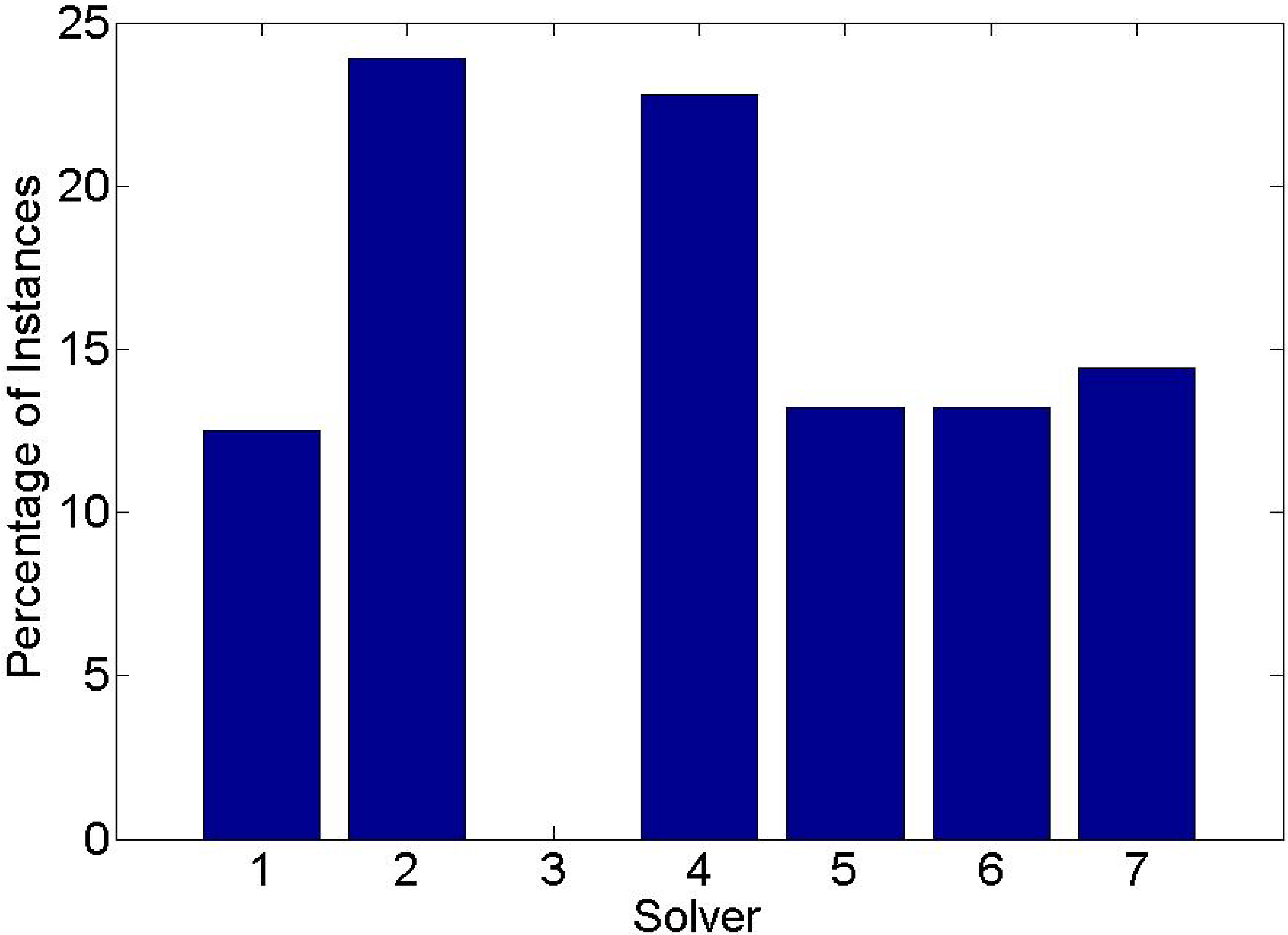
Solver	RAND	HAND	BM	INDU
Best Challenger (of 17)	1128.63	2960.39	224.53	11.89

** Statistically insignificant performance difference (sign rank test).*
Hydra's performance was significantly better in all other pairings.

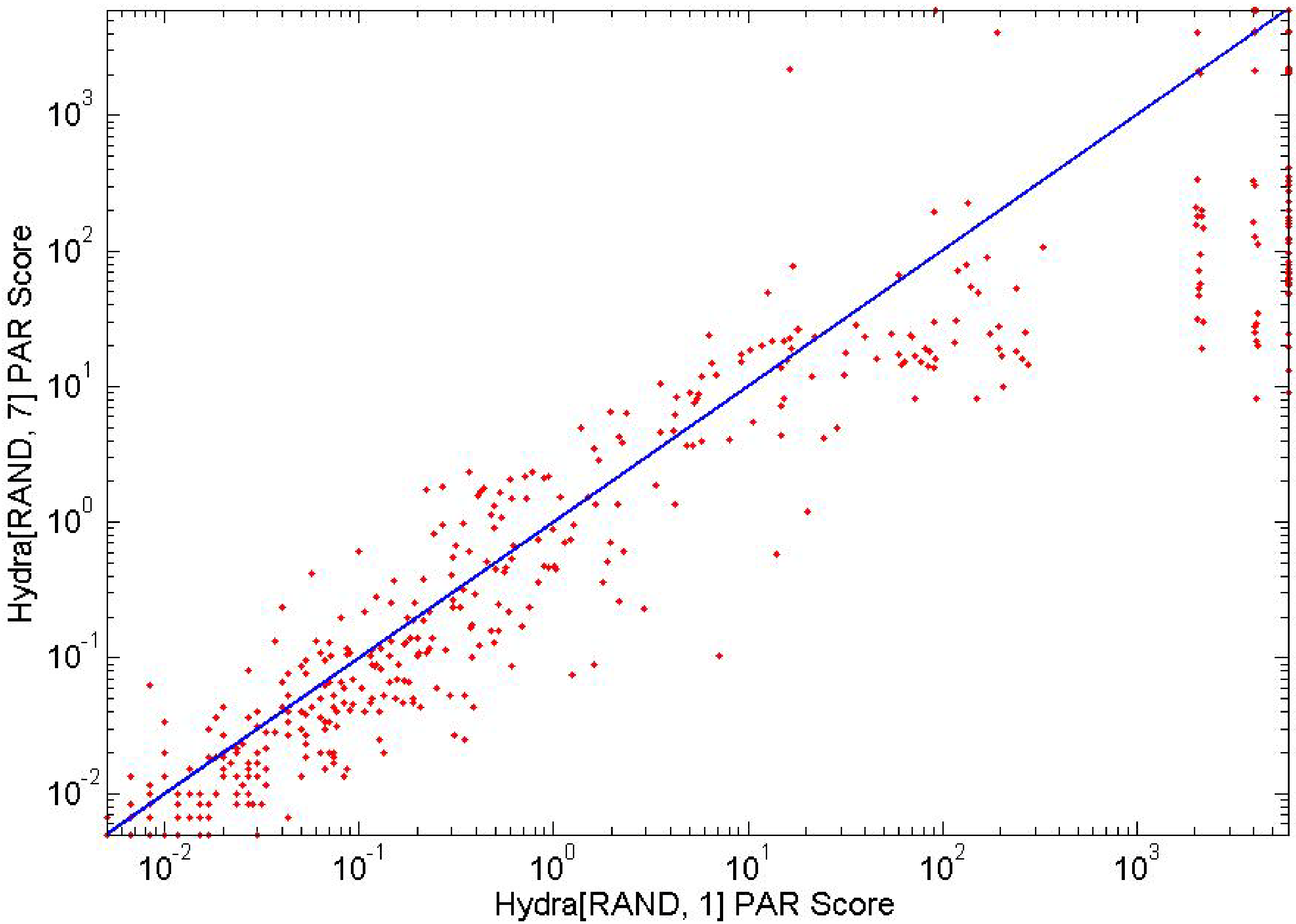
Performance Progress, RAND



Selection Percentages After 7 Iterations, RAND



Improvement After 7 Iterations, RAND



Conclusions

- **Hydra: an automatic design approach combining**
  - portfolio-based algorithm selection **(here: "SATzilla")**
  - automated algorithm configuration **(here: "SATenstein")**

- Completely automated
- Algorithm/configurator/portfolio-builder agnostic

- Most useful in domains where few strong solvers exist
- Nevertheless met or exceeded state-of-the-art performance on SLS for SAT in 4 domains

Thank You!