# Utilising Uncertainty for Efficient Learning of Likely-Admissible Heuristics

Ofir Marom, Benjamin Rosman
*University of the Witwatersrand*

## Previous Work

Handcrafting heuristics is prohibitive. Alterative: **learn heuristics from data with Machine Learning.**

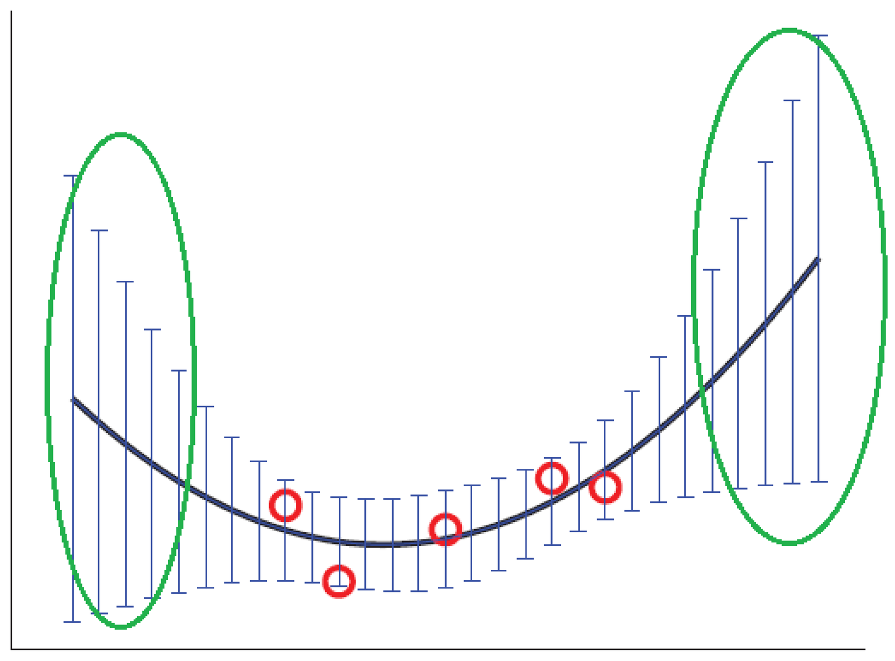
*Bootstrap Learning Heuristics*

initialise weak heuristic $h$
**repeat**
  generate training tasks randomly
  try solve each task in time-limit
  **if** can't solve enough **then**
    increase time-limit
  **else**
    update $h$ with plans of solved tasks
  **end if**
**until** forever

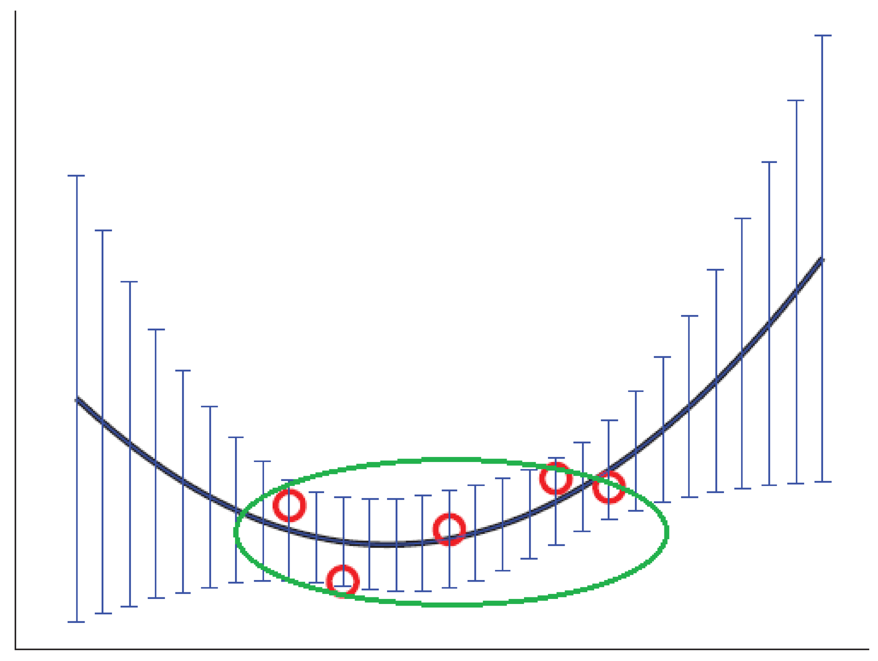
Shortcomings:

- inefficient to train (random task generation)
- high suboptimality (train on non-optimal plans)

## Our Contribution



**Epistemic Uncertainty**: lack of training data



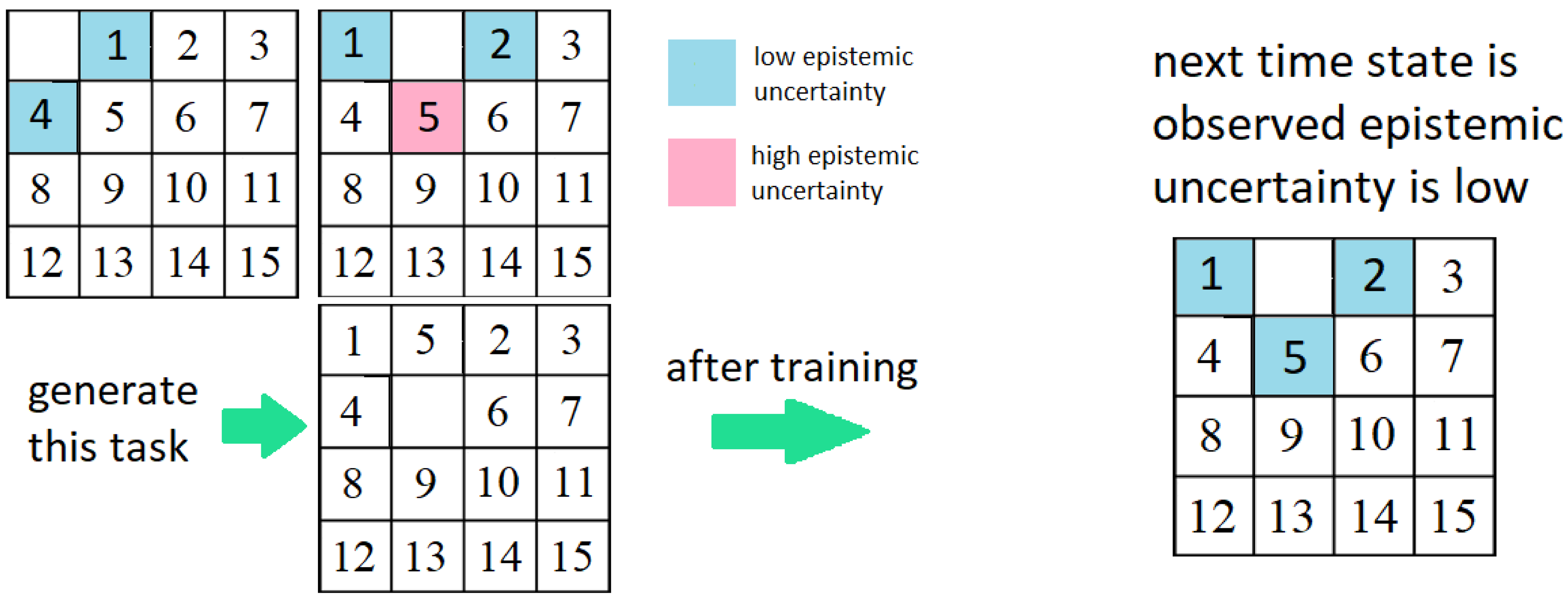
**Aleatoric Uncertainty**: variation in training data

Used to:

- systematically explore task-space (more efficient than random tasks)
- plan with likely-admissible heuristic (train on likely-optimal plans)

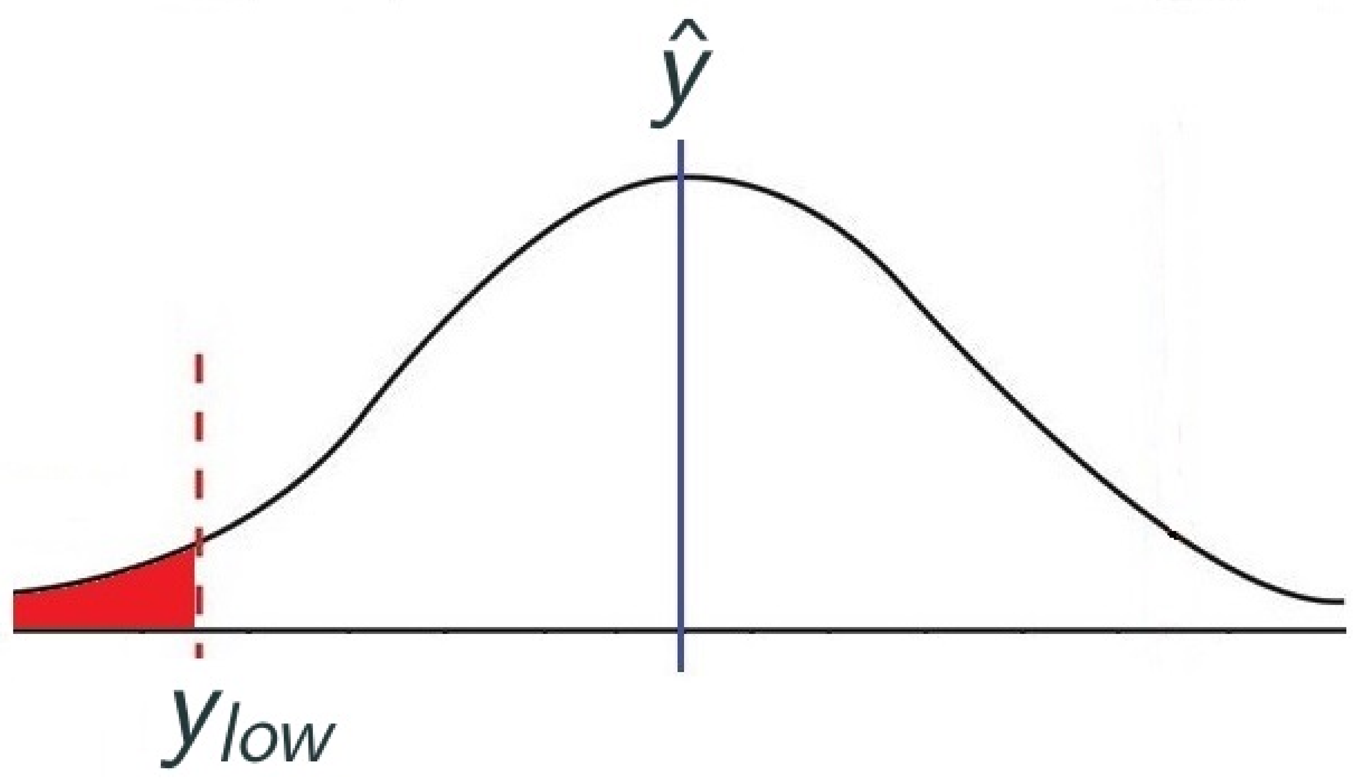
## Efficient Task Generation

- start at goal state
- seek state with high epistemic uncertainty
- generate task with this start state
- solve and learn from plan




## Reduce Suboptimality

- plan with conservative $y_{low}$
- i.e. $P(y_{low} \leq y) = 0.95$ (likely-admissible heuristic)
- produces likely-optimal plans
- mitigates compounding errors during training

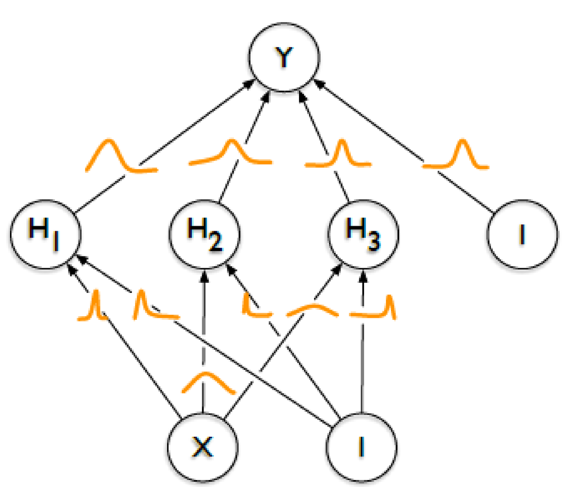



## Our Algorithm

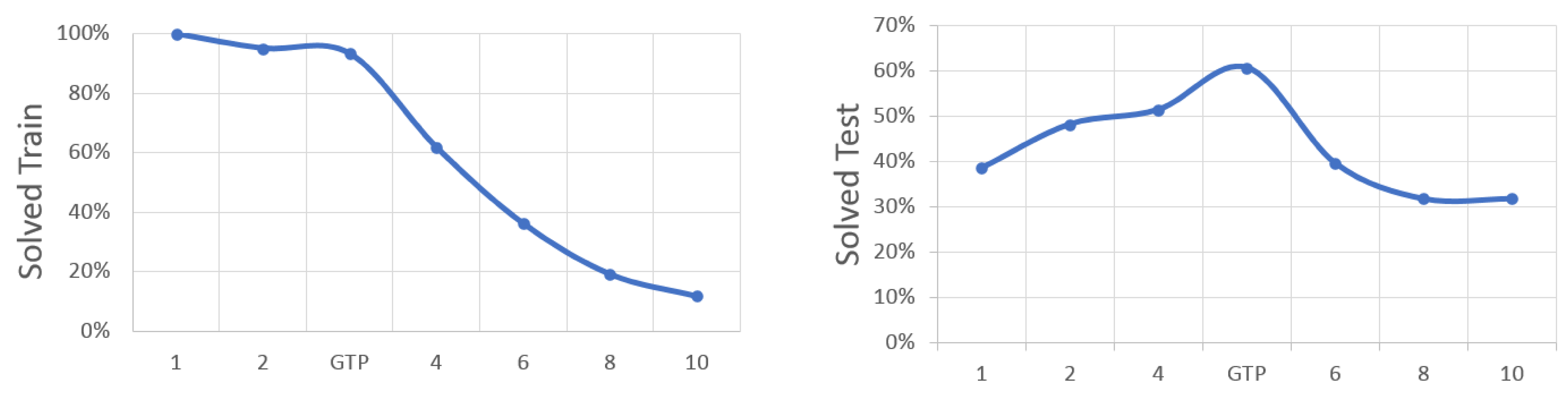
*Efficiently Learn Likely-Admissible Heuristic*

initialise weak heuristic $h$
**repeat**
  generate training tasks efficiency
  try solve each task with $y_{low}$
  **if** can't solve enough **then**
    make $y_{low}$ less conservative
  **else**
    update $h$ with plans of solved tasks
  **end if**
**until** forever

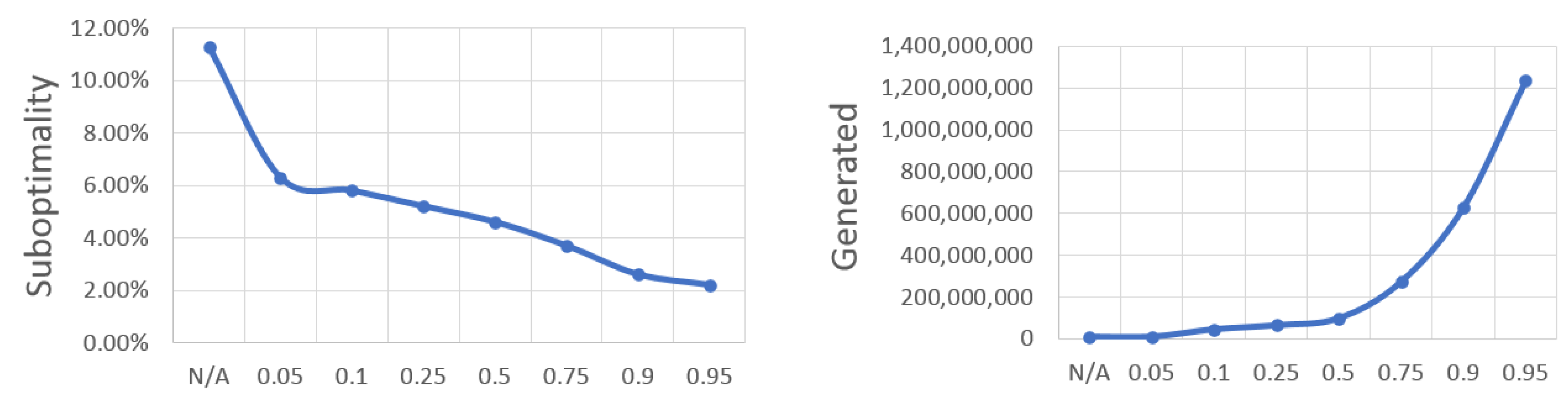
How to model these uncertainties? We use Weight Uncertainty Neural Networks.




## Results




Efficiency results for 15-puzzle




Suboptimality results for 24-puzzle