Algorithm Selection for Optimal Multi-Agent Path finding

Omri Kaduri, Eli Boyarski, Roni Stern

Ben Gurion University of the Negev, Israel
Palo Alto Research Center (PARC), USA

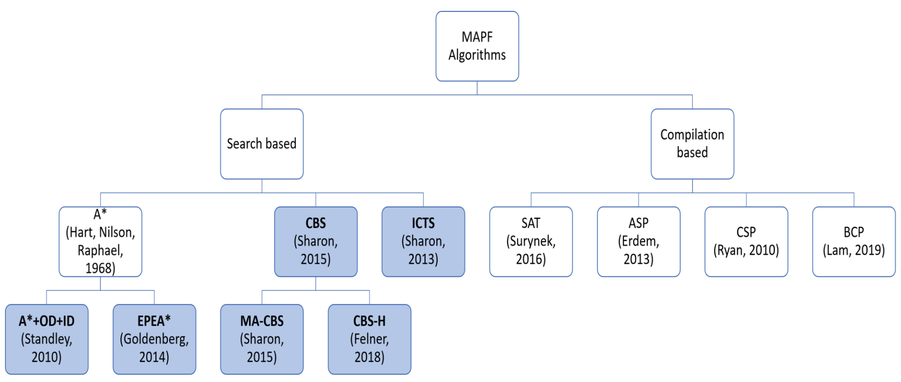
1. MULTI-AGENT PATH FINDING (MAPF)

- **Input**
  1. A map with N locations
  2. Set of N agents with start and goal locations
- **Output** - Joint plan for all agents to reach their goals without collisions
- **Assumptions**
  1. Discrete time
  2. Each agents performs one action (move or wait) at each time
  3. A collision occurs when 2 agents in the same vertex/edge at the same time
- **Objective-** Minimize the sum of costs (SOC)

2. OPTIMAL SOLVERS FOR CLASSICAL MAPF

- Optimal solvers for classical MAPF can be divided to two main branches:
  - Search-based algorithms
  - Compilation-based algorithms

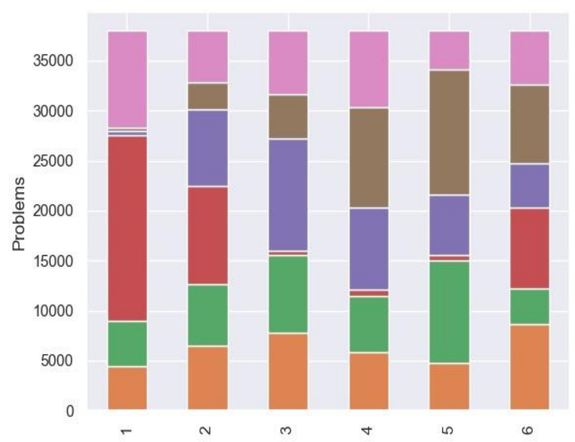
Our focus is on search-based algorithms, and specifically, we have experimented with the algorithms bolded at the next figure:



3. MOTIVATION

For each algorithm, we recorded the time it took to solve each MAPF problem.

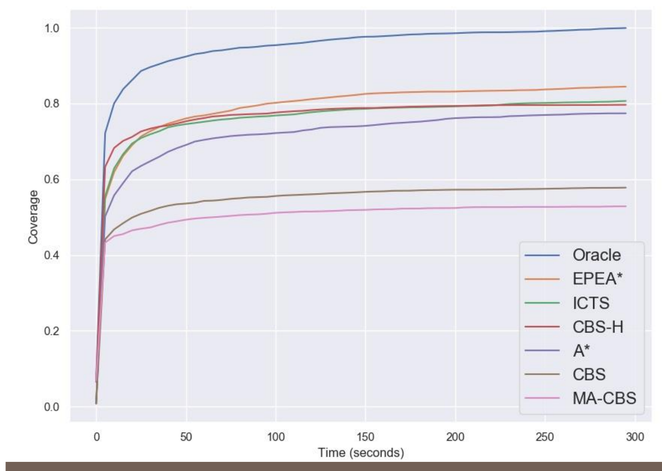
- Using the recorded running times for each algorithm over a wide range of problems, we can see for each algorithm, how many problems it was the 1st to solve, 2nd to solve and so on until the last column, which shows how many problems each algorithm was the slowest to solve.



- Interestingly, we can see different behavior between different algorithms. CBS-H, for example, is either the first or second to solve, or the last to solve.

3. MOTIVATION – CONT.

When comparing the coverage (I.e., the ratio of problems solved under a set timeout) of different algorithms to an oracle (optimal selection of algorithm for each problem), we see that **the oracle is significantly better than every single algorithm!**



4. FEATURE EXTRACTION

In-order to use Machine Learning models for Algorithm Selection (AS) in MAPF, we need to extract features from the MAPF problem. We have experimented the 2 main approaches:

- **Hand-crated feature extraction**
  - **Grid-features** - #rows, #cols, obs. density ratio
  - **Grid-agents features -** #Agents, branching factor, sparsity ratio
  - **Agent-agent features**

- **Convert MAPF to an image**
  - Based on Sigurdson et al, 2019
  - Blocked/unblocked cells => Black/White
  - Source/target cells => Green/Red

5. REGRESSION VS CLASSIFICATION

Algorithm selection, which shown to be effective at other NP-Hard tasks (e.g., SAT, Planning), might be considered to be either a regression or a classification problem.

- **Classification:** predict the fastest algorithm for each problem
  - **Pros**: Directly learn an algorithm selection model, jointly on all solvers
  - **Cons**: Choosing the second-fastest algorithm is as bad as last algorithm
- **Regression:** predict the runtime of each algorithm, then choose best
  - **Pros**: Might better model the "hardness" of problem for each solver
  - **Cons**: Not directly an algorithm selection, noisy target, censored data

Task/ Representation	**Regression**	**Classification**
Hand-crafted Features	XGBoost Regression	XGBoost Classification
MAPF-as-image	CNN Regression	CNN Classification

6. EXPERIMENTS AND RESULTS

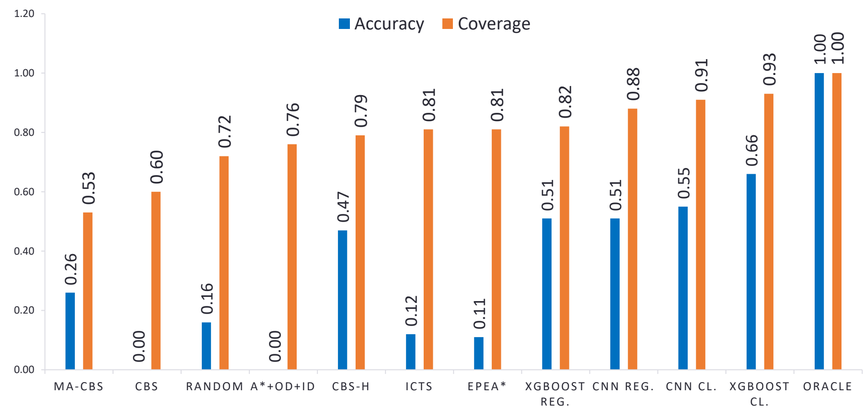
Dataset

- MovingAI (Stern et al., 2019) dataset, 28 grids with 25 scenario files
- Each scenario solved incrementally from 2 agents until unsolved under 5 minutes
- Problems where no algorithm solved are discarded
- 30/70 split to train/test, **grouped by scenario**

Metrics

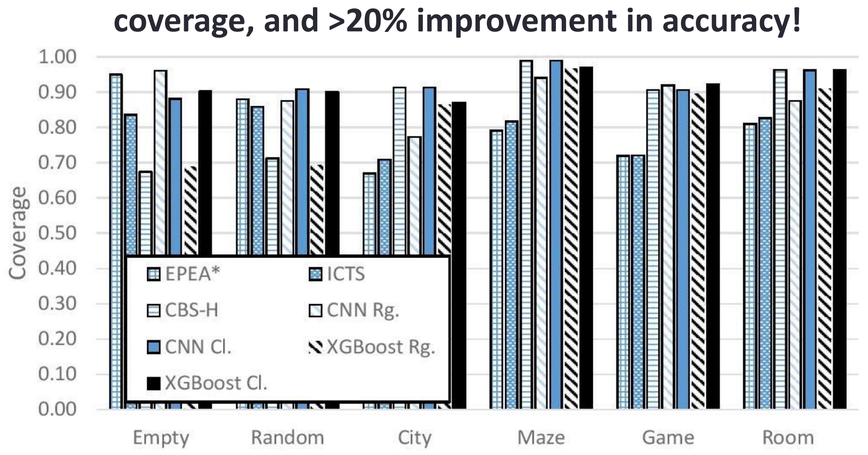
- **Accuracy –** Prob. the fastest algorithm is chosen
- **Coverage - %** problems. solved under the timeout
- **Total runtime -** time needed to solve all problems

- Baselines
  - Random selection
  - Choose one algorithm for all problems



Key points:

- All algorithm selection models out-performed every single solver
- Our best performing algorithm selection model – based on XGBoost with Classification task, **achieves more than 11% improvement in coverage, and >20% improvement in accuracy!**



7. FUTURE WORK AND CONCLUSION

Future work can involve:

- Adding compilation-based solvers
- Generalizing to new unseen maps
- Use an algorithm selection model to **understand the hardness of MAPF problems** - what makes a problem hard for each solver?
- Online selection, where trying to solve a sub-problem under a small amount of time and use the information from that experience as a feature, is an interesting future line of work.

Conclusion:

- **Algorithm selection is effective for optimal MAPF**
- Multiple learning-based methods are able to tackle algorithm selection
  - Hand-crafted features with classification performs best
- Our code and dataset are **publicly available**!

https://github.com/OmriKaduri/MAPF-Classification