Anytime Program Synthesis for the BEN ARC Solver

Program synthesis can be framed as a search problem. Large search spaces require efficient algorithms to make synthesis feasible. One proposed approach is Divide, Align and Conquer (DA&C). DA&C-based solvers are predominantly greedy in their decision-making, and the role of backtracking in this context has largely not yet been explored.

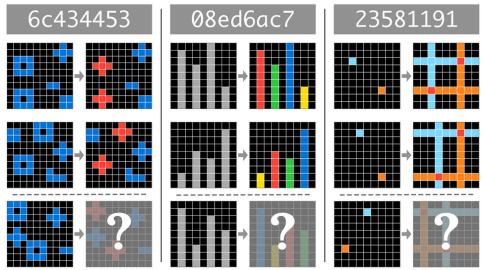
Jakub Florek[1], Sebastijan Dumančić[2], Dekel Zak[3]

1. J.M.Florek@student.tudelft.nl
2. Responsible Professor
3. Supervisor

TU Delft

Herb.jl

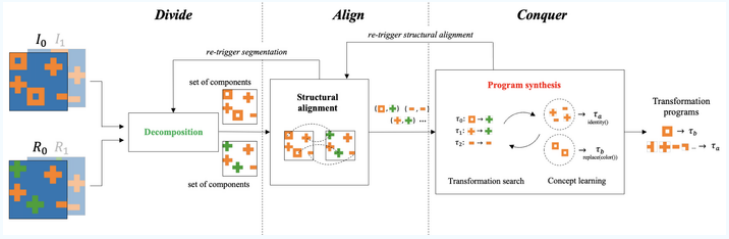
1. Background



- **Abstraction and Reasoning Corpus (ARC)**
  - Grid-based transformation tasks
  - Defined by input-output examples.
  - Challenging for AI
- **Divide, Align, and Conquer (DA&C)**
  - **Divide:** Split input/output
  - **Align:** Match corresponding objects
  - **Conquer:** Synthesize transformations
- **BEN solver:** DA&C solver. Structured synthesis pipeline. Allows limited backtracking in case of critical failures.

2.Research Question

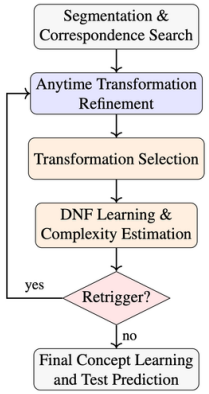
- **Conflict-Driven Clause Learning (CDCL)** in related search domains.
- **Idea:** Use internal solver metrics to improve the choices.
- **Problem:** What part of the solver suffers the most from poor choices?



- **Research Question: To what extent can internal solver metrics be used to guide retriggerable transformation search within the BEN pipeline?**

3.Methods

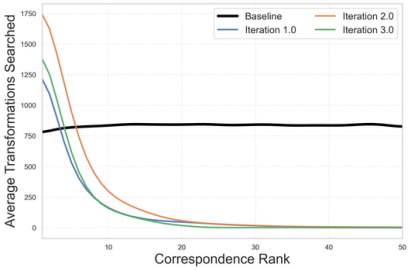
- **Anytime Transformation Refinement:** From a **one-shot** exhaustive search into a **resumable, anytime refinement** process. Store intermediate **search states** and incrementally improve transformations.
- **Weighted Round-Robin Scheduling:** The scheduler dynamically allocates refinement **budgets** across correspondences. It prioritizes **high-ranking** correspondences while preserving exploration in **lower ranks**.
- **Heuristic-Driven Retriggering:** Threshold-based **metrics** evaluate **transformation quality** at the end of each refinement cycle. Retriggers if the current result shows excessive **overfitting** or **high structural complexity**.
- **Adaptive Search Focus:** This approach shifts the synthesis objective from simply finding **any valid program** to **progressively improving** the generality of transformations.



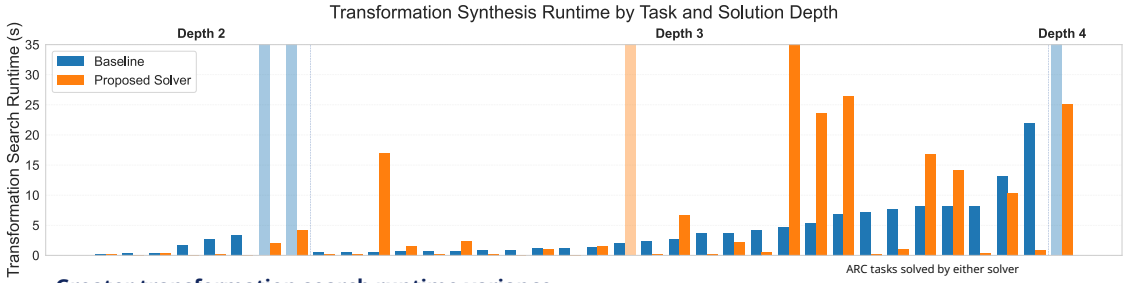
4.Experiments

Configuration	Outcome	Tasks	Avg.Full Time(s)	Avg.Search Time(s)	Avg.Explored Transforms	Avg.Overfit Ratio($\rho_{overfit}$)	Avg.DNF Comp(ρ_{DNF})
BaselineBEN	Solved	33	31.15	3.89	7624.70	0.07	0.94
	TimedOut	80	92.95	72.15	115400.61	–	–
	Failed	283	50.71	21.85	34837.81	0.60	1.48
ProposedSolver	Solved	35	41.42	7.17	6553.14	0.04	0.97
	TimedOut	146	161.05	117.21	5608.19	–	–
	Failed	219	153.68	64.03	14036.73	0.54	0.78

- **Similar solve rate**
- **Less visited transformations, comparable runtime**
- **Change in threshold metrics**



- **Vastly different search profiles**
- **More effort at the head**
- **Dynamic vs static correspondence ranks**



- **Greater transformation search runtime variance**
- **3 new solved tasks**
- **Much deeper search, same runtime limit**

5.Conclusion

- **Adaptive Synthesis:** Reformulated transformation synthesis for the BEN solver into a **resumable, anytime search process** with dynamically allocated computation. Altered search distribution.
- **Performance Impact:** Does not increase the **aggregate solve rate** on the ARC benchmark. Reduces the number of **explored candidate transformations** and enables the solver to discover more **unique solutions**.
- **Key Insight:** Transformation synthesis is likely not the primary bottleneck in the current BEN architecture, pointing toward earlier pipeline stages, as key areas for future improvement.