GENERALIZE

Constraining Program Synthesis Search

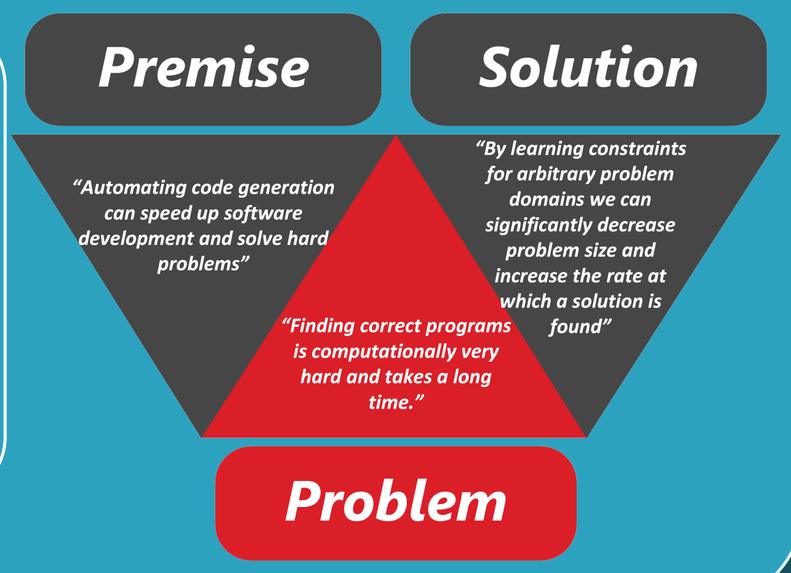
By: L.G. Kroes (L.G.Kroes@student.tudelft.nl)[1]

TU Delft

1. Introduction

Program Synthesis

"Automatic generation of code from some specification"



2. Research Questions

GENERALIZE	1	*Can we build a system to constrain unseen Domain Specific Languages for functional program settings?*
Constraints	A	*What constraints can be applied in a functional setting?*
Elicitation	B	*How do we elicit constraints for arbitrary problem settings?*
Application	C	*How do we apply constraints in more complex function sequences?*
Validation	D	*How do we verify the elicited constraints are valid?*
Verification	E	*How does a constrained system perform in comparison to an unconstrained system?*
Explanation	F	*Can we explain this result?*

3. Motivating Example

Routing problem:

Find any route to the ball

REMEMBER:

This is just a thought experiment! Actual applications are considerably more complex

a. How many possible routes are there between two points:

Infinitely many

b. How many optimal routes are there?:

$d = |x_a - x_b| + |y_a - y_b|$

Upper-bound : $O(2^d)$

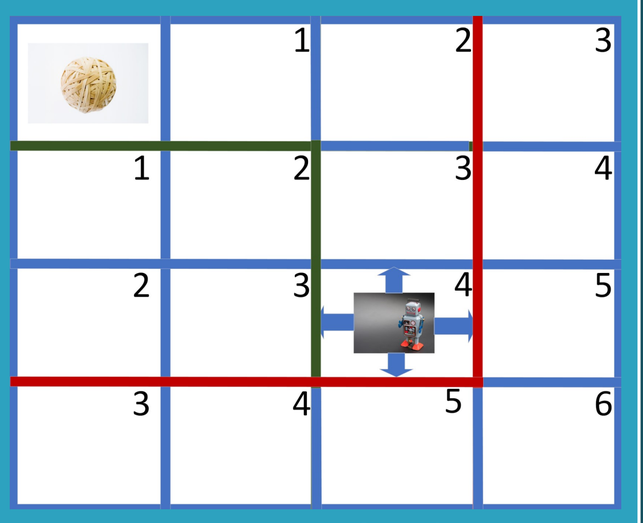
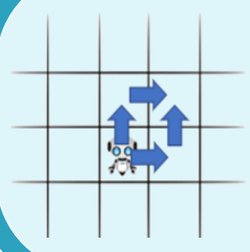
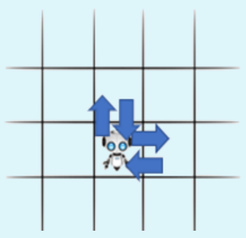


Figure 1: An instance of a robot routing problem.



Symmetry: *"Does order matter?"*



Reflection: *"do functions reverse each other?"*

4. Methodology

STEP 1:

Find independence and identity relations between functions using observational methods. Derive associated constraints

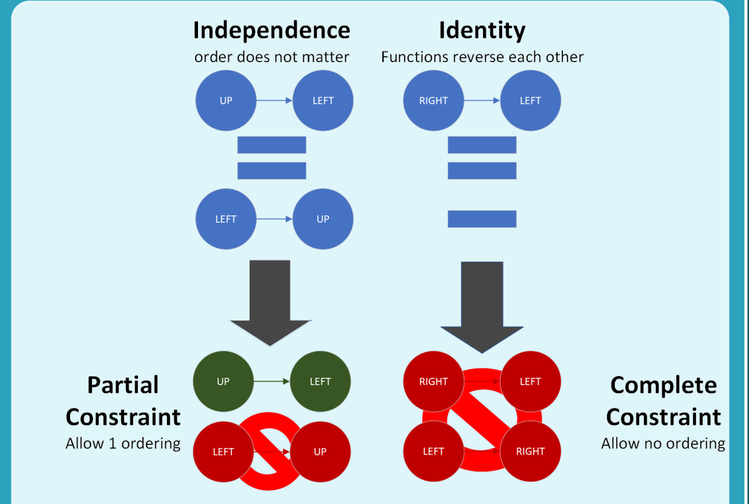


Figure 2: Properties and their implied constraints

Independence-Grouping

Find groups of indepedendent tokens to propagate complete and partial constraints

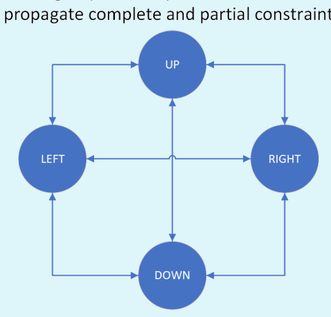


Figure 3: Grouped set of independent tokens (arrow iff an independence property holds between 2 functions)

STEP 2:

Group strongly connected clusters of independent functions to create more effective constraints

STEP 3:

Verify the elicited constraints using a genetic algorithm

Elicited constraints
Found by testing properties in a small set of test environments

Genetic Algorithm
Searches constraint space using techniques akin to natural selection

Verified constraints
Constraint assignment that has been found to improve search speed

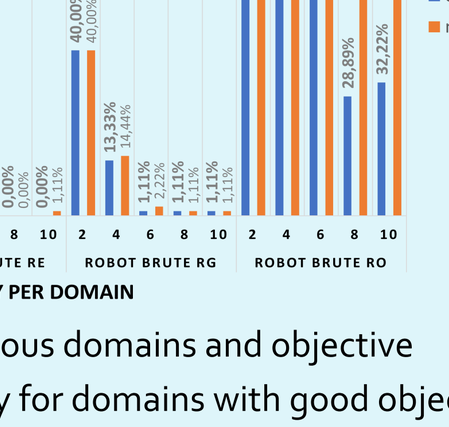


Figure 4: Validating constraints using a genetic algorithm

5. Results

Figure 5: Problem solving accuracy for various domains and objective functions. A thorough decrease in accuracy for domains with good objective functions is observed. A slight increase in accuracy is seen when no such function is available.

Two Points of failure

1. Constraints

- Relatively many good solutions are pruned
- Objective Functions rate states improperly in constrained search
- More complex functions means less possible solutions.

2. Genetic algorithm

- Runtime of candidate solutions was too low to signal issues with local minima
- An insufficient training set was used, harder problems should be included.

1. Supervised by: Dr. S Dumančić (s.dumancic@tudelft.nl)