Introduction

In the ARC challenge DSL, transformations (xforms) map one grid to another. To formalize these transformations, we use **shape types** to describe the sets of grids that a transformation can operate on. This allows us to distinguish between non-parametric and parametric transformations, depending on whether the transformations are specific to particular grid types or generalized using type variables.

Shape Types

A **shape type** S denotes a set of grids. This can capture structure such as dimensions, content patterns, or other properties. For example:

- S could represent all grids of a certain size, like all 3×3 grids.
- S could also represent grids with a specific pattern, such as grids where all cells along the diagonal are the same.

Shape types include singleton types, denoting specific grids, and type variables `X` of kind `Grid` (for now).

Transformations

A **transformation** xf is a total function $Grid \to Grid$.

Specs

A **spec** $[S, T]$ denotes the set of transformations xf such that $\forall G \in Grid.G \in S \Rightarrow \text{xf}(G) \in T$

In addition, $[S1, T1] \wedge [S2, T2] \wedge [S3, T3]$ is also a spec, denoting the intersection.

Matching

A **spec** $[S, T]$ matches an example (I, O) if

1. $I \in S$
2. $\forall \text{xf}.\text{xf} \in [S, T] \Rightarrow \text{xf}(I) = O$

Intuitively, matching means that the spec admits the input I, and exactly specifies the behaviour on it.

Non-Parametric Matching

Given examples $I_i \to O_i$ for $i = 1..3$, the following spec matches all of them:

$$[I1, O1] \wedge [I2, O2] \wedge [I3, O3]$$

where we write `I` as the singleton shape type denoting exactly `I`.

Here's an example

```
def flip_2x2(grid):
    # Assumes grid is 2x2
    return [[grid[0][1], grid[0][0]],
            [grid[1][1], grid[1][0]]]
```

This transformation, `flip_2x2`, is specifically designed to work only for 2×2 grids. It does not generalize to other grid sizes.

Parametric Spec

Consider this example

```
def flip_grid(grid):
    # Flip any grid horizontally
    return [row[::-1] for row in grid]
```

This transformation, `flip_grid`, works for grids of any size by flipping each row horizontally.

In addition to the non-parametric spec above, this transformation satisfies spec $[X, Flip(X)]$ where:

- X represents any grid.
- Flip(X) denotes the grid X flipped horizontally.

Genericity of Specs

A spec spec1 is more specific than spec2 if it is a strict subset. In the examples above, the non-parametric spec is more specific than the parametric one.

Genericity of Transformation

A transformation xf1 is more generic than xf2 if it satisfies more generic specs. In the examples above, `flip_grid` is more generic than `flip_2x2`.