Inductive Program Synthesis for ARC

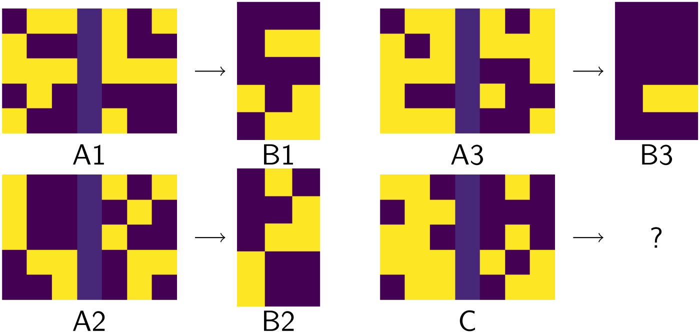
Aditya Challa

28th September 2022

Overview

1. Decoupling elements of ARC task
   - What is the ARC challenge?
   - Issues with ARC challenge?
   - Decoupling ARC tasks
2. Can ILP help? - Böhm Jacopini theorem
3. Introducing IPARC Challenge
4. Demo of the GitHub repository

ARC Task



An example task from the ARC (taken from `https://samacquaviva.com/LARC/explore/`). The task to be solved is of the form "if A1:B1 and A2:B2 and A3:B3 then C: ?".

- The operation is to take the intersection of left and right region.

Issues with ARC

  - Nothing is known about the data.
  - No background knowledge is provided.
  - What would such a function look like.
- In short, ARC is too open-ended.

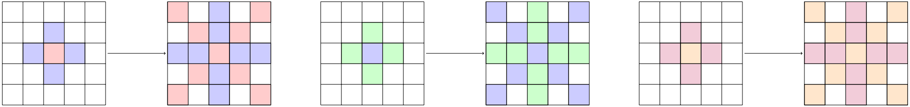
Decopuling ARC tasks

We premise that ARC tasks consists of three parts-

- Color Change
- Grid Resize
- Structure

Decopuling ARC tasks

- Color Change



  - ARC tasks are Color-Permulation-Equivariant
  - That is, if f is an answer and P denotes the permutation of colors, then,

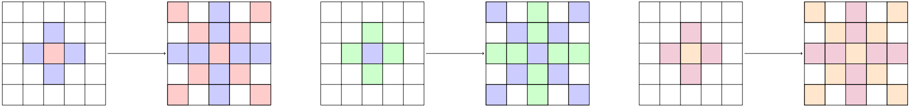
$$f(P(.)) = P(f(.))$$

Decopuling ARC tasks

- Resizing Grids
  - Standard way to deal with this is to assume infinite grids.
  - However, we still need to know which part to focus on. So, we need a `Crop` function!

Decopuling ARC tasks

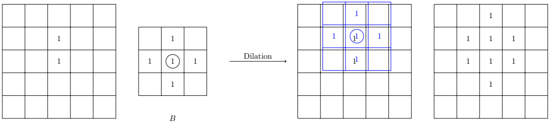
- Structural Changes



- Trickiest of them all and least understood aspect of ARC.
- We use ideas from Mathematical Morphology (MM) to understand this.

Decopuling ARC tasks

- Mathematical Morphology Operators - Dilation.



X

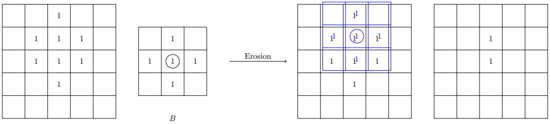
Place the center of B at each 1 pixel of X. and take the union

$X \oplus B$

- The dilation operator takes a pixel and transforms it to a set, coincidentally called structuring element.
- If there are multiple pixels, take the union.

Decopuling ARC tasks

- Mathematical Morphology Operators - Erosion.



X

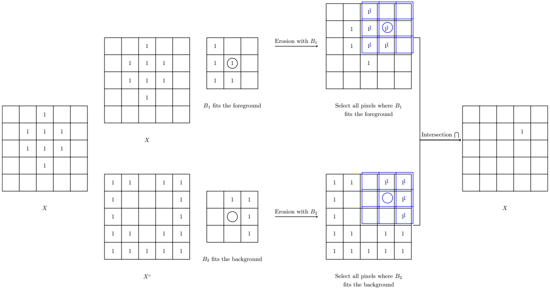
Place the center of B at each 1 pixel of X.
and take all the pixels for which B_x "fits"

$X \ominus B$

- Informally, it selects only those pixels at which, structuring element is contained within the set.

Decopuling ARC tasks

- Mathematical Morphology Operators - Hit-Or-Miss.


X
X
X^c
B_1 fits the foreground
B_2 fits the background
Erosion with B_1
Erosion with B_2
Select all pixels where B_1 fits the foreground
Select all pixels where B_2 fits the background
Intersection $\bigcap$
X

  - Selects those pixels which fit a pattern. Essentially an intersection of erosions for foreground and background.

- As you can intuitively imagine, these three would form the basis of identifying the structure of the ARC task.

Decopuling ARC tasks

- Putting the basic elements together
  - Let $I^k_{m,n}$ denote an ARC image. m, n denotes the size, k denotes the number of colors.
  - Each color can also be seen as an binary image!

    $$I^k_{m,n} = \langle I^{(1)}_{m,n}, I^{(2)}_{m,n}, \cdots , I^{(k)}_{m,n} \rangle$$

    where $I^{(1)}_{m,n}$ is binary image corresponding to color k
  - Let us call, $\mathcal{L}^k_{m,n} = \{I^k_{m,n}\}$

Decopuling ARC tasks

- Putting the basic elements together
  - Consider only color and structure for the time-being.

$$\begin{tikzcd}
\tilde{I}^{k}_{m,n} & & = \Big\langle & \tilde{I}^{(1)}_{m,n} & \tilde{I}^{(2)}_{m,n} & \tilde{I}^{(3)}_{m,n} & \cdots & \tilde{I}^{(k)}_{m,n} & \Big\rangle \\
 & := & & & & \boxed{\text{Color Change}} \arrow[ull] \arrow[ul] \arrow[u] \arrow[urr] & & & \\
I^{k}_{m,n} \arrow[uu, "\Phi"] & & = \Big\langle & I^{(1)}_{m,n} \arrow[urr, "\phi^{(1)}"] & I^{(2)}_{m,n} \arrow[ur, "\phi^{(2)}"] & I^{(3)}_{m,n} \arrow[u, "\phi^{(3)}"] & \cdots & I^{(k)}_{m,n} \arrow[ull, "\phi^{(k)}"'] & \Big\rangle
\end{tikzcd}$$

- The structure part operates on the individual binary images.
- The color change resolves the color operations by mapping $\{0, 1\}^k$ to one-hot vectors.
- Resizing operators, `Pad` and `Crop` allows us to increase the size of the image and focus on particular region.

Decopuling ARC tasks

- Adding memory to the program.
  - The memory can be modelled using the `Endomorphism` functions

$$\oplus(I^k_{m,n}) = I^k_{m,n} \cup \Phi(I^k_{m,n})$$
$$\otimes(I^k_{m,n}) = I^k_{m,n} \cap \Phi(I^k_{m,n})$$

- Let the set of functions which are a composition of these elements be Σ.

Decopuling ARC tasks

- Are these sufficient??

Proposition

If $I^k_{m_1,n_1}, \tilde{I}^k_{m_2,n_2}$ *are from* $\bigcup \mathcal{L}^k_{m,n}$ *and* $I^k_{m_1,n_1}$ *is non-empty, then there exists a sequence of functions* $\langle f_1, f_2, \dots, f_N \rangle$ *s.t. each* $f_i \in \Sigma$ *and* $\tilde{I}^k_{m_2,n_2} = f_N(f_{N-1} \cdots (f_1(I^k_{m_1,n_1})) \cdots)$.

Decopuling ARC tasks

- A small catch!

Proposition

Given the set of functions Σ_1 *there exists an ARC-like task* T *that contains elements* $(I_1, \tilde{I}_1)$, $(I_2, \tilde{I}_2)$, *s.t.* $\tilde{I}_1 =_{\Sigma_1} f(I_1)$, *and* $\tilde{I}_2 =_{\Sigma_1} g(I_2)$ *but there is no* h *s.t.* $\tilde{I}_1 =_{\Sigma_1} h(I_1)$ *and* $\tilde{I}_2 =_{\Sigma_1} h(I_2)$.

- There seems to be LOGIC missing the framework!

ILP and Böhm Jacopini theorem

ILP formulation:

Given:	The set of basic functions B as background knowledge, and examples E of input-output images,
Find:	a program H s.t.$B \bigwedge H \vDash E$.

ILP and Böhm Jacopini theorem

ILP formulation:

Structured Program Theorem The following three control structures are sufficient to construct any computable function:

```
  Sequence   newp(X,Y,NewP):- NewP(X,Y).
 Selection   if_else(X,Y,Q,NewP1,NewP2):- (Q(X) ->
             NewP1(X,Y); NewP2(X,Y)).
Iteration    repeat_k(X,Y,K,NewP):-
              (K > 0 -> (NewP(X,Y1), K1 is K - 1,
             repeat_k(Y1,Y,K1,NewP)); Y=X).
```

- Does there exist an ILP engine capable of this?

IPARC Challenge

- IPARC - Simple dataset to test an ILP engine capable of Program Synthesis

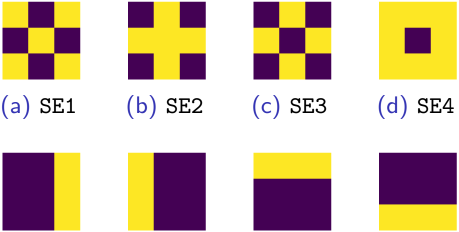
Three Categories:

- Category A requires finding a sequence which explains the examples in E.
- Category B requires finding incorporating the Sequence/Selection/Conditional program synthesis into the ILP engine.
- Category C asks - Would program synthesis be any simpler if snapshots are given?

- We use the framework we described above to construct these programs.

IPARC Challenge

- Searching for Structuring Elements



(a) SE1 (b) SE2 (c) SE3 (d) SE4

(e) SE5 (f) SE6 (g) SE7 (h) SE8

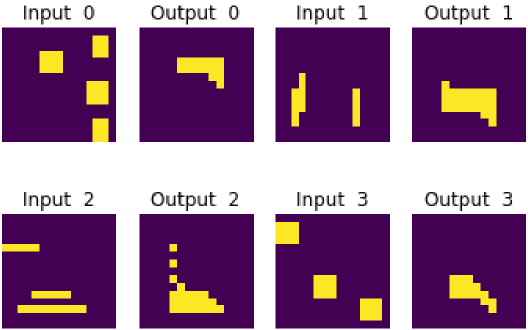
- Specific Structuring elements (SE) to search.

IPARC Challenge

- Category A (Simple)
  - Aim is to find a sequence of operators which explain the input-output pairs.
    `Dilation SE*` → `Dilation SE*` → `Dilation SE*` → `Dilation SE*` → `Erosion SE*` → `Erosion SE*` → `Erosion SE*` → `Erosion SE*`

IPARC Challenge

- Category A (Simple)

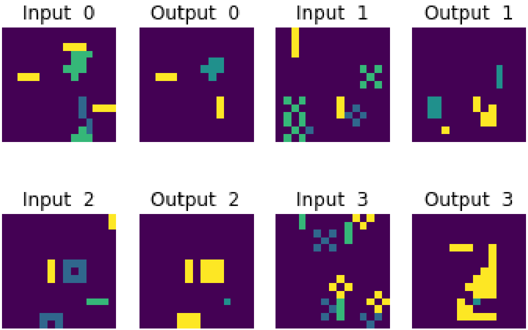


IPARC Challenge

- Category A (Hard)
  - Recall that each color can be operated on separately and are combined using a color-change rule.
  - The aim is to find a sequence for each different color `Dilation SE*` → `Dilation SE*` → `Dilation SE*` → `Dilation SE*` → `Erosion SE*` → `Erosion SE*` → `Erosion SE*` → `Erosion SE*`
  - And identify a color change rule.
  - To simplify we stick to 3 colors.

IPARC Challenge

- Category A (Hard)

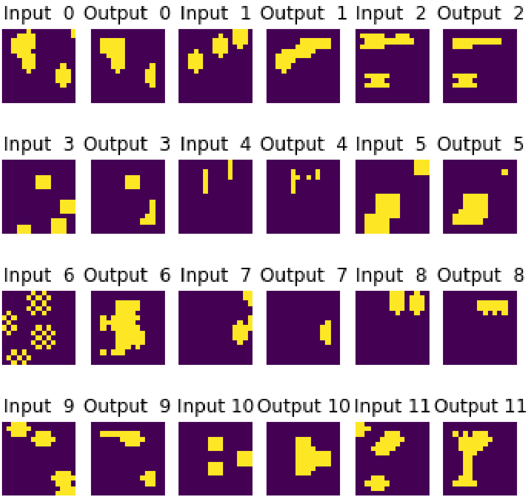


IPARC Challenge

- Category B - Sequence
  - Invent a sequence predicate which are common across tasks.
  - Example, Two tasks
    `Dilation SE5` → `Dilation SE7` → `Erosion SE5` → `Erosion SE7` → `Dilation SE6` → `Dilation SE7` → `Erosion SE6` → `Erosion SE7`
    and
    `Dilation SE7` → `Dilation SE7` → `Erosion SE7` → `Erosion SE7` → `Dilation SE6` → `Dilation SE7` → `Erosion SE6` → `Erosion SE7`
    Should identify
    `Dilation SE6` → `Dilation SE7` → `Erosion SE6` → `Erosion SE7`
    as common predicate.

IPARC Challenge

- Category B (Sequence)

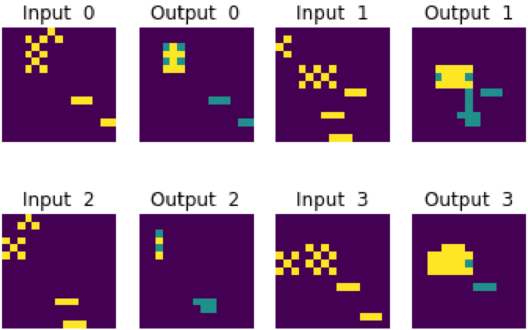


IPARC Challenge

- Category B - Selection
  - Recall `Hit-or-Miss` selects pixels corresponding to a pattern. This is used to simulate the `IF` condition.
  - So, Given a binary image these tasks require the ILP engine to -
    - (a) Select the pixels which match a pattern and assign these a new color.
    - (b) Operate on the two sets of pixels with two distinct sequences.
    - (c) Use a color-change to resolve conflicts

IPARC Challenge

- Category B (Selection)

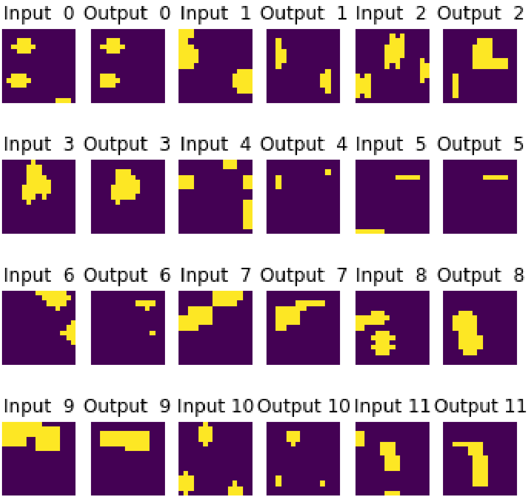


IPARC Challenge

- Category B - Iteration
  - Invent a `Iterate_k` predicate which are common across tasks.
  - Two subtasks with common iterate predicate should add the iterate to the background knowledge

IPARC Challenge

- Category B (Iteration)

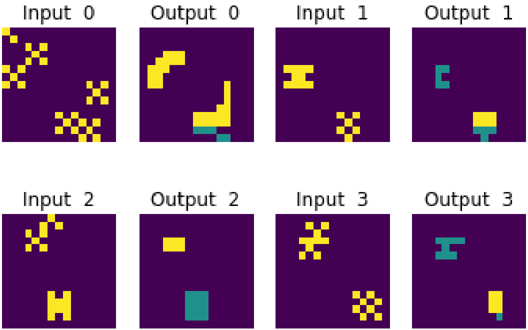


IPARC Challenge

- Category B - Hard
  - The hard task combines all three - Sequence, Selection and Iteration
  - Given a binary image, these tasks ILP engine to -
    (a) Select the pixels which match a pattern (using `Hit-or-Miss`) and assign these a new color.
    (b) Operate on one set of pixels with a sequence.
    (c) Operate on one set of pixels using an iteration.
    (d) Use a color-change to resolve conflicts.

IPARC Challenge

- Category B (Hard)



IPARC Challenge

- Category C
  - Would intermediate snapshots help solve the task?

Materials and Evaluation

- Background Knowledge
  - ops - `Dilation, Erosion, HitOrMiss, ChangeColor`
  - se - `SE1-SE8`
  - `ChangeColor` use an mapping in the form of array, $C : \{0, 1\}^k \rightarrow \{0, 1, \cdots, k\}$

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 2 \end{bmatrix} \tag{1}$$

Materials and Evaluation

- Understanding color (band)
  - `Dilation, Erosion` can operate on an individual color (a.k.a band).
  - `HitOrMiss` creates a new color (band) for the selected pixels.
  - `ChangeColor` works across colors (bands).

Materials and Evaluation

- Data
  - Use `.json` files to store the data
  - Solutions are provided in both `.txt` format and `.json` format.

Materials and Evaluation

- Testing a program
  - A program can be tested using `TestSequence.py`
  - each op is written as `<band>-<op>-<se>-<iterate>`
  - `<band>` specifies the band on which the operator is applied. Ignored not work for `HitorMiss` and `ChangeColor`.
  - `<op>` specifies the operation.
  - `<se>` denotes a structuring element `SE1-SE8` or array for `ChangeColor`
  - `<iterate>` defines the number of times this operator is repeated.

Materials and Evaluation

- Testing a program (Example)

```
python TestSequence.py ./Dataset/CatB_Hard/Task000.json
1-HitOrMiss-SE8-1 1-Dilation-SE6-2 1-Erosion-SE6-2
2-Dilation-SE8-1 2-Dilation-SE7-1 2-Dilation-SE5-1
2-Dilation-SE7-1 2-Erosion-SE7-1 2-Erosion-SE5-1
2-Erosion-SE7-1
1-ChangeColor-[[0,0,0],[0,1,2],[1,0,1],[1,1,2]]-1
```

Materials and Evaluation

- Evaluation:
  - How many of the examples does the program explain?
  - How long is the program?
  - Is there any additional information used?

Demo

DEMO of the GitHub Repo.