Geometric Analogy A2 Report

Junho Park

Algorithm

There are three main functions that drive this program: makeMatch(), newGetTransformations(), and getTransformation2().

makeMatch() :

After parsing the information provided by A1.py, the program keeps track of each interpretation of a file in 8 different lists, A (for all the interpretations of A), B (for all the interpretations of B), C (for all the interpretations of C), and K1-K5 (for all the interpretations of each K). Afterwards, pairs of two of these lists are passed into the makeMatch() function in order to create all the possible pairs of shape associations between the two lists of interpretations passed in. For example:

Assume A has the shapes (p1,p2) and B has the shapes (p1,p2). makeMatch() would first create all possible pairings of the two, ([p1,p1],[p2,p2]) and ([p2,p1],[p1,p2]). Now if A had 3 interpretations, each with shapes (p1,p2), makeMatch() would create the association ([p1,p1],[p2,p2]) and ([p2,p1],[p1,p2]) for each interpretation for a total of 6 different matches. Afterwards, this function returns the list of all the different matchings and the program will store them in the list ABmatching. We would then repeat this process for CK1, CK2, CK3, CK4, CK5 storing all of them in their own lists analogously named with ABmatching

newGetTransformations() :

This function takes in the lists of interpretations and their attributes, as well as the matchings (ABmatching, CK1matching, CK2matching, CK3matching, CK4matching, CK5matching) provided by makeMatch(). This function finds the first order transformations between each set of matchings (A,B), (C,K1), (C,K2), (C,K3), (C,K4), and (C,K5). After it finds all the first order transformations between a set of matchings, the function will return a list of every single possible first order transformation for each set of matchings. Since the function returns every single first order transformation, we don't compute a cost for each transformation in this section as we have to test every first order transformation in our second order transformation anyways (this is to avoid the case where the cost of our first order transformation is very low, but the cost of the second order transformation using that first order transformation is very high). Every first order transformation will fall under one of four types of transformations:

Addt/delt describes transformations where a shapes was added or deleted from A to B or C to Kx

Change describes transformations related to size changes, and shape changes (like triangle to square).

Move describes transformations related to a translation across the graph, both horizontal and vertical.

Pos describes any transformation where attributes relative to other shapes within the file (like *left_of, right_of, above, below, inside, outside, overlap)* change compared to the other file. For example, if the matching *(p1,p1)* and *(p2,p2)* were given, A contained the attribute *left_of(p1,p2),* and B contained the attribute *right_of(p1,p2),* then the first order transformation *pos(left_of(p1,p2),right_of(p1,p2))* would be pushed into the final transformation list.

The function will return a list of every single possible transformations for a given set of matchings and the program will store it in another data structure. For the function, newGetTransformations(ABmatching, A, B), the list of all first order transformations will be returned and stored into the list ABtransforms. (CKxmatching, C, Kx)'s first order transformations will be stored in CKxtransforms for x = one through five.

getTransformations2() :

This function takes two sets of first order transformations (like ABtransforms and CK1transforms). We'll run through the code with these two lists of transformations.

First, all possible matchings of shapes between an entry in ABtransforms (one set of first order transformations) and an entry in CK1transforms (again, one set of first order transformations) are found. This is essentially the same process as the makeMatch() function for the initial shape interpretations.

Next, the function will compare the set of first order transformations from ABtransforms and the set of second order transformations from CK1transforms in order to find differences in the transformations between the two. Every time a first order transformation in ABtransforms doesn't have an equivalent first order transformation in CK1transforms, the function writes *remove('transformAB')* and every time a first order transformation in CK1transforms doesn't have an equivalent first order transformation in ABtransforms, the function writes *add('transformCK1')*. Every time the function removes or adds a transform, it also adds up a cost for each second order transformation. The costs for add and remove are the same, but for each first order transformation, the cost values are as follows:

Change in the type of shape (i.e. triangle to square) = 5,000 points

Adding or deleting a shape = 500 points

Position changes (any first order transformation starting with *pos*) = 50 points

Shape translations (any first order transformation starting with *move)* = 5 points

Size changes between shapes = 1 point

In the case that there is a tie between penalties, the lengths of the first order transformations are used as a tiebreaker. This helps us find the simplest first order transformation that leads to the simplest second order transformation. This function will return the penalty for the best second order transformation between AB and CK1, and we append it in our main function to a list named costs.

We run this function for each set of first order transformations that we have to find second order transformations for, so getTransformation(AB,CK1), getTransformation(AB,CK2), getTransformation(AB, CK3), getTransformation(AB,CK4), and getTransformation(AB,CK5). We will end up with the costs list containing the cost of the best second order transformation for each set of first order transformations with indices corresponding to the number after K.

In the end, we determine the lowest penalty within the costs list (with tiebreakers broken by each respective list of first order transformations) and the answer is returned in the form "K = x".

Problems

P1 :

```
pjunho@LAPTOP-JN87Q1L5:/mnt/c/Users/Junho/Documents/_School/COURSES/EECS492/a2$ python3 a2.py P1
A:
[[[['p1', '20.0', '30.0', '50.0', '90.0', '80.0', '30.0', '20.0', '30.0'], ['p2', '38.0', '40.0', '50.0', '70.0', '62.0'
, '40.0', '38.0', '40.0']], [['triangle', 'middle', 'center', 'outside', 'p2'], ['triangle', 'middle', 'center', 'inside
', 'p1']]]]


B:
[[[['p1', '20.0', '30.0', '50.0', '90.0', '80.0', '30.0', '20.0', '30.0']], [['triangle', 'middle', 'center']]]]


C:
[[[['c1', '50.0', '50.0', '30.0'], ['p1', '40.0', '40.0', '60.0', '40.0', '60.0', '60.0', '40.0', '60.0', '40.0', '40.0'
]], [['circle', 'middle', 'center', 'outside', 'p1'], ['square', 'middle', 'center', 'inside', 'c1']]]]


K4:
[[[['c1', '50.0', '50.0', '30.0']], [['circle', 'middle', 'center']]]]


T(T(A,B),T(C,K))
[]
T(A,B)
[[('p1', 'p1'), ('p2', 'del')], 'pos(outside(p1,p2),none(p1,del))', 'delt(p2)']
T(C,K)
[[('c1', 'c1'), ('p1', 'del')], 'pos(outside(c1,p1),none(c1,del))', 'delt(p1)']
K = 4
```

P2 :

```
pjunho@LAPTOP-JN87Q1L5:/mnt/c/Users/Junho/Documents/_School/COURSES/EECS492/a2$ python3 a2.py P2
A:
[[[['c1', '50.0', '75.0', '20.0'], ['c2', '50.0', '75.0', '10.0'], ['p1', '40.0', '15.0', '60.0', '15.0', '60.0', '35.0', '40.0',
'35.0', '40.0', '15.0']], [['circle', 'top', 'center', 'outside', 'c2', 'above', 'p1'], ['circle', 'top', 'center', 'inside', 'c1'
, 'above', 'p1'], ['square', 'bottom', 'center', 'below', 'c1', 'below', 'c2']]]]


B:
[[[['c1', '50.0', '25.0', '20.0'], ['c2', '50.0', '75.0', '10.0'], ['p1', '40.0', '15.0', '60.0', '15.0', '60.0', '35.0', '40.0',
'35.0', '40.0', '15.0']], [['circle', 'bottom', 'center', 'below', 'c2', 'outside', 'p1'], ['circle', 'top', 'center', 'above', 'c
1', 'above', 'p1'], ['square', 'bottom', 'center', 'inside', 'c1', 'below', 'c2']]]]


C:
[[[['c1', '50.0', '25.0', '5.0'], ['p1', '30.0', '50.0', '70.0', '50.0', '50.0', '95.0', '30.0', '50.0'], ['p2', '45.0', '65.0', '
55.0', '65.0', '55.0', '55.0', '45.0', '55.0', '45.0', '65.0']], [['circle', 'bottom', 'center', 'below', 'p1', 'below', 'p2'], ['
triangle', 'top', 'center', 'above', 'c1', 'above', 'p2', 'outside', 'p2'], ['square', 'top', 'center', 'above', 'c1', 'below', 'p
1', 'inside', 'p1']]]]


K4:
[[[['c1', '50.0', '25.0', '5.0'], ['p1', '30.0', '10.0', '70.0', '10.0', '50.0', '55.0', '30.0', '10.0'], ['p2', '45.0', '80.0', '
55.0', '80.0', '55.0', '70.0', '45.0', '70.0', '45.0', '80.0']], [['circle', 'bottom', 'center', 'inside', 'p1', 'below', 'p2'], [
'triangle', 'bottom', 'center', 'outside', 'c1', 'below', 'p2'], ['square', 'top', 'center', 'above', 'c1', 'above', 'p1']]]]


T(T(A,B),T(C,K))
["remove(['pos', 'none', 'c1', 'c2', 'below', 'c1', 'c2'])", "remove(['pos', 'none', 'c2', 'c1', 'above', 'c2', 'c1'])", "add(['po
s', 'above', 'p1', 'p2', 'below', 'p1', 'p2'])", "add(['pos', 'below', 'p2', 'p1', 'above', 'p2', 'p1'])"]
T(A,B)
[[('c1', 'c1'), ('c2', 'c2'), ('p1', 'p1')], 'move(top(c1),bottom(c1))', 'pos(outside(c1,c2),none(c1,c2))', 'pos(none(c1,c2),below
(c1,c2))', 'pos(inside(c2,c1),none(c2,c1))', 'pos(none(c2,c1),above(c2,c1))', 'pos(above(c1,p1),none(c1,p1))', 'pos(none(c1,p1),ou
tside(c1,p1))', 'pos(below(p1,c1),none(p1,c1))', 'pos(none(p1,c1),inside(p1,c1))']
T(C,K)
[[('c1', 'c1'), ('p1', 'p1'), ('p2', 'p2')], 'pos(below(c1,p1),none(c1,p1))', 'pos(none(c1,p1),inside(c1,p1))', 'move(top(p1),bott
om(p1))', 'pos(above(p1,c1),none(p1,c1))', 'pos(none(p1,c1),outside(p1,c1))', 'pos(above(p1,p2),below(p1,p2))', 'pos(outside(p1,p2
),none(p1,p2))', 'pos(below(p2,p1),above(p2,p1))', 'pos(inside(p2,p1),none(p2,p1))']
K = 3
```

P3 :

```
pjunho@LAPTOP-JN87Q1L5:/mnt/c/Users/Junho/Documents/_School/COURSES/EECS492/a2$ python3 a2.py P3
A:
[[['p1', '75.0', '70.0', '70.0', '60.0', '65.0', '50.0', '70.0', '50.0', '85.0', '50.0', '75.0', '70.0'], ['p2', '70.0', '60.0', '
70.0', '50.0', '70.0', '30.0', '20.0', '30.0', '20.0', '70.0', '70.0', '70.0', '70.0', '60.0']], [['triangle', 'top', 'right', 'ri
ght_of', 'p2', 'above', 'p2', 'overlap', 'p2'], ['rectangle', 'middle', 'left', 'left_of', 'p1', 'below', 'p1', 'overlap', 'p1']]]


B:
[[[['p1', '40.0', '55.0', '30.0', '35.0', '50.0', '35.0', '40.0', '55.0'], ['p2', '75.0', '30.0', '25.0', '30.0', '25.0', '70.0',
'75.0', '70.0', '75.0', '30.0']], [['triangle', 'bottom', 'left', 'left_of', 'p2', 'below', 'p2', 'inside', 'p2'], ['rectangle', '
middle', 'center', 'right_of', 'p1', 'above', 'p1', 'outside', 'p1']]]]


C:
[[['p1', '70.0', '70.0', '60.0', '70.0', '60.0', '60.0', '70.0', '60.0', '80.0', '60.0', '80.0', '70.0', '70.0', '70.0'], ['p2', '
10.0', '20.0', '5.0', '30.0', '5.0', '70.0', '10.0', '80.0', '70.0', '80.0', '70.0', '70.0', '70.0', '60.0', '70.0', '20.0', '10.0
', '20.0']], [['rectangle', 'top', 'right', 'right_of', 'p2', 'above', 'p2', 'overlap', 'p2'], ['scc', 'middle', 'left', 'left_of'
, 'p1', 'below', 'p1', 'overlap', 'p1']]]


K3:
[[[['p1', '10.0', '20.0', '5.0', '30.0', '5.0', '70.0', '10.0', '80.0', '70.0', '80.0', '70.0', '20.0', '10.0', '20.0'], ['p2', '1
5.0', '40.0', '15.0', '30.0', '35.0', '30.0', '35.0', '40.0', '15.0', '40.0']], [['scc', 'middle', 'left', 'right_of', 'p2', 'abov
e', 'p2', 'outside', 'p2'], ['rectangle', 'bottom', 'left', 'left_of', 'p1', 'below', 'p1', 'inside', 'p1']]]]


T(T(A,B),T(C,K))
["remove(['move', 'top', 'p1', 'bottom', 'p1'])", "remove(['move', 'right', 'p1', 'left', 'p1'])", "remove(['move', 'left', 'p2',
'center', 'p2'])", "add(['move', 'top', 'p1', 'bottom', 'p2'])", "add(['move', 'right', 'p1', 'left', 'p2'])"]
T(A,B)
[[('p1', 'p1'), ('p2', 'p2')], 'move(top(p1),bottom(p1))', 'move(right(p1),left(p1))', 'pos(right_of(p1,p2),left_of(p1,p2))', 'pos
(above(p1,p2),below(p1,p2))', 'pos(overlap(p1,p2),inside(p1,p2))', 'move(left(p2),center(p2))', 'pos(left_of(p2,p1),right_of(p2,p1
))', 'pos(below(p2,p1),above(p2,p1))', 'pos(overlap(p2,p1),outside(p2,p1))']
T(C,K)
[[('p2', 'p1'), ('p1', 'p2')], 'pos(left_of(p2,p1),right_of(p1,p2))', 'pos(below(p2,p1),above(p1,p2))', 'pos(overlap(p2,p1),outsid
e(p1,p2))', 'move(top(p1),bottom(p2))', 'move(right(p1),left(p2))', 'pos(right_of(p1,p2),left_of(p2,p1))', 'pos(above(p1,p2),below
(p2,p1))', 'pos(overlap(p1,p2),inside(p2,p1))']
K = 3
```

P4 :

```
pjunho@LAPTOP-JN87Q1L5:/mnt/c/Users/Junho/Documents/_School/COURSES/EECS492/a2$ python3 a2.py P4
A:
[[['c1', '50.0', '50.0', '10.0'], ['p1', '20.0', '30.0', '50.0', '90.0', '80.0', '30.0', '20.0', '30.0']], [['circle', 'middle',
'center', 'inside', 'p1'], ['triangle', 'middle', 'center', 'outside', 'c1']]]


B:
[[['c1', '50.0', '75.0', '20.0'], ['p1', '38.0', '20.0', '50.0', '50.0', '62.0', '20.0', '38.0', '20.0']], [['circle', 'top', 'ce
nter', 'above', 'p1'], ['triangle', 'bottom', 'center', 'below', 'c1']]]


C:
[[['c1', '50.0', '50.0', '20.0'], ['p1', '40.0', '60.0', '60.0', '60.0', '60.0', '40.0', '40.0', '40.0', '40.0', '60.0']], [['cir
cle', 'middle', 'center', 'outside', 'p1'], ['square', 'middle', 'center', 'inside', 'c1']]]


K2:
[[['c1', '50.0', '20.0', '10.0'], ['p1', '30.0', '90.0', '70.0', '90.0', '70.0', '50.0', '30.0', '50.0', '30.0', '90.0']], [['cir
cle', 'bottom', 'center', 'below', 'p1'], ['square', 'top', 'center', 'above', 'c1']]]


T(T(A,B),T(C,K))
["remove(['change', 'small', 'c1', 'large', 'c1'])", "remove(['move', 'middle', 'c1', 'top', 'c1'])", "add(['change', 'small', 'p1
', 'large', 'p1'])", "add(['move', 'middle', 'p1', 'top', 'p1'])"]
T(A,B)
[[('c1', 'c1'), ('p1', 'p1')], 'change(small(c1),large(c1))', 'move(middle(c1),top(c1))', 'pos(inside(c1,p1),none(c1,p1))', 'pos(n
one(c1,p1),above(c1,p1))', 'change(large(p1),small(p1))', 'move(middle(p1),bottom(p1))', 'pos(outside(p1,c1),none(p1,c1))', 'pos(n
one(p1,c1),below(p1,c1))']
T(C,K)
[[('c1', 'c1'), ('p1', 'p1')], 'change(large(c1),small(c1))', 'move(middle(c1),bottom(c1))', 'pos(outside(c1,p1),none(c1,p1))', 'p
os(none(c1,p1),below(c1,p1))', 'change(small(p1),large(p1))', 'move(middle(p1),top(p1))', 'pos(inside(p1,c1),none(p1,c1))', 'pos(n
one(p1,c1),above(p1,c1))']
K = 2
```

P5 :

```
pjunho@LAPTOP-JN87Q1L5:/mnt/c/Users/Junho/Documents/_School/COURSES/EECS492/a2$ python3 a2.py P5
A:
[[[['d1', '50.0', '80.0'], ['p1', '40.0', '40.0', '60.0', '40.0', '60.0', '20.0', '40.0', '20.0', '40.0', '40.0']], [['dot', 'top'
, 'center', 'above', 'p1'], ['square', 'bottom', 'center', 'below', 'd1']]]]


B:
[[[['p1', '40.0', '40.0', '60.0', '40.0', '60.0', '20.0', '40.0', '20.0', '40.0', '40.0']], [['square', 'bottom', 'center']]]]


C:
[[[['d1', '30.0', '50.0'], ['p1', '60.0', '60.0', '80.0', '60.0', '80.0', '40.0', '60.0', '40.0', '60.0', '60.0']], [['dot', 'midd
le', 'left', 'left_of', 'p1'], ['square', 'middle', 'right', 'right_of', 'd1']]]]


K5:
[[[['p1', '60.0', '60.0', '80.0', '60.0', '80.0', '40.0', '60.0', '40.0', '60.0', '60.0']], [['square', 'middle', 'right']]]]


T(T(A,B),T(C,K))
["remove(['pos', 'below', 'p1', 'd1', 'none', 'p1', 'del'])", "add(['pos', 'right_of', 'p1', 'd1', 'none', 'p1', 'del'])"]
T(A,B)
[[('p1', 'p1'), ('d1', 'del')], 'pos(below(p1,d1),none(p1,del))', 'delt(d1)']
T(C,K)
[[('p1', 'p1'), ('d1', 'del')], 'pos(right_of(p1,d1),none(p1,del))', 'delt(d1)']
K = 5
```