Week 04 Report (4 Aug, 9 Aug) - Guangyao Xu

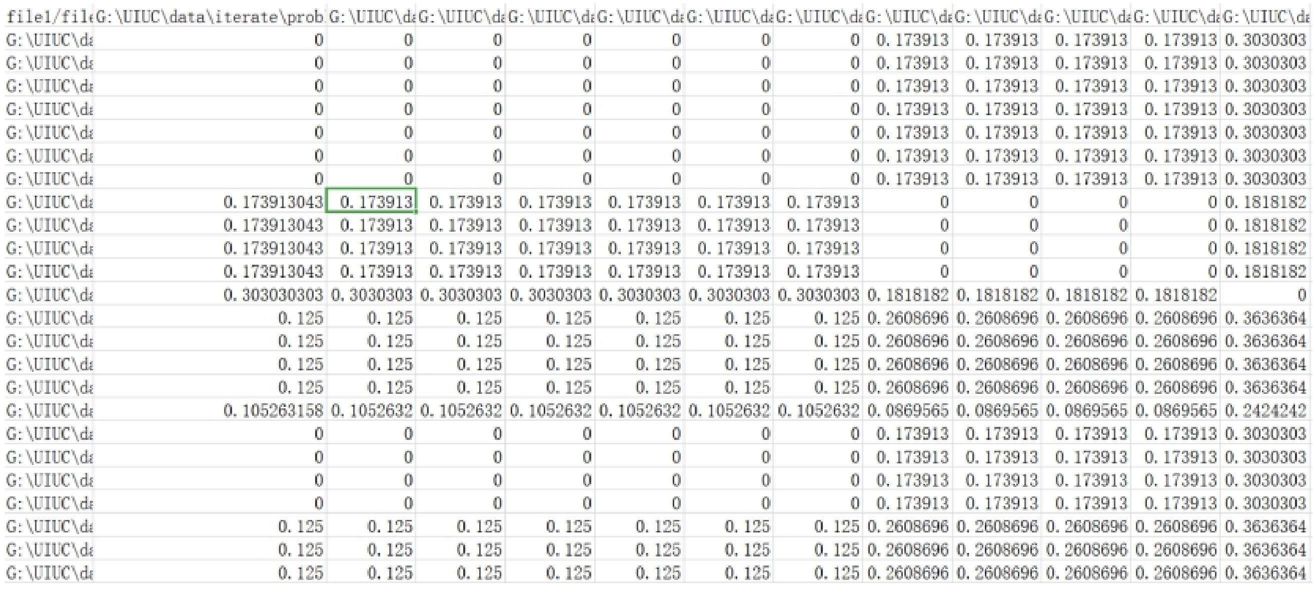
1、Work

1.1 generate edit distance data set from the existing data set 'iterate'

I tried to generate whole graphs of these data set last week. And this week, I tried to generated the CFG,DD and CD separately from the whole graph. Thus, I got the similarity of not only the whole graphs, but also the CFG,DD and CD graphs. You can find all of these result on my github project : https://github.com/BryceTsui/UIUCSummerResearch/tree/master/GED/result .

For more further data analysis, I also transfer the result matrix to csv format. So that you can use excel or other application to open and analyse it. The csv data is also in the link above.

The first raw and column represent the path of the files where the graphs generated from.



file1/fil	G:\UIUC\data\iterate\prob	G:\UIUC\da	G:\UIUC\da	G:\UIUC\da	G:\UIUC\da	G:\UIUC\da	G:\UIUC\da	G:\UIUC\da	G:\UIUC\da	G:\UIUC\da	G:\UIUC\da	G:\UIUC\da
G:\UIUC\da	0	0	0	0	0	0	0	0.173913	0.173913	0.173913	0.173913	0.3030303
G:\UIUC\da	0	0	0	0	0	0	0	0.173913	0.173913	0.173913	0.173913	0.3030303
G:\UIUC\da	0	0	0	0	0	0	0	0.173913	0.173913	0.173913	0.173913	0.3030303
G:\UIUC\da	0	0	0	0	0	0	0	0.173913	0.173913	0.173913	0.173913	0.3030303
G:\UIUC\da	0	0	0	0	0	0	0	0.173913	0.173913	0.173913	0.173913	0.3030303
G:\UIUC\da	0	0	0	0	0	0	0	0.173913	0.173913	0.173913	0.173913	0.3030303
G:\UIUC\da	0	0	0	0	0	0	0	0.173913	0.173913	0.173913	0.173913	0.3030303
G:\UIUC\da	0.173913043	0.173913	0.173913	0.173913	0.173913	0.173913	0.173913	0	0	0	0	0.1818182
G:\UIUC\da	0.173913043	0.173913	0.173913	0.173913	0.173913	0.173913	0.173913	0	0	0	0	0.1818182
G:\UIUC\da	0.173913043	0.173913	0.173913	0.173913	0.173913	0.173913	0.173913	0	0	0	0	0.1818182
G:\UIUC\da	0.173913043	0.173913	0.173913	0.173913	0.173913	0.173913	0.173913	0	0	0	0	0.1818182
G:\UIUC\da	0.303030303	0.3030303	0.3030303	0.3030303	0.3030303	0.3030303	0.3030303	0.1818182	0.1818182	0.1818182	0.1818182	0
G:\UIUC\da	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.2608696	0.2608696	0.2608696	0.2608696	0.3636364
G:\UIUC\da	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.2608696	0.2608696	0.2608696	0.2608696	0.3636364
G:\UIUC\da	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.2608696	0.2608696	0.2608696	0.2608696	0.3636364
G:\UIUC\da	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.2608696	0.2608696	0.2608696	0.2608696	0.3636364
G:\UIUC\da	0.105263158	0.1052632	0.1052632	0.1052632	0.1052632	0.1052632	0.1052632	0.0869565	0.0869565	0.0869565	0.0869565	0.2424242
G:\UIUC\da	0	0	0	0	0	0	0	0.173913	0.173913	0.173913	0.173913	0.3030303
G:\UIUC\da	0	0	0	0	0	0	0	0.173913	0.173913	0.173913	0.173913	0.3030303
G:\UIUC\da	0	0	0	0	0	0	0	0.173913	0.173913	0.173913	0.173913	0.3030303
G:\UIUC\da	0	0	0	0	0	0	0	0.173913	0.173913	0.173913	0.173913	0.3030303
G:\UIUC\da	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.2608696	0.2608696	0.2608696	0.2608696	0.3636364
G:\UIUC\da	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.2608696	0.2608696	0.2608696	0.2608696	0.3636364
G:\UIUC\da	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.2608696	0.2608696	0.2608696	0.2608696	0.3636364

t

I also output the time cost of each comparison. The result is as follow:

CFG time cost: 89.88861560505973 s

CD: time cost: 202.91443248247444 s

DD:1893.38469750431 s

1.2 extract opcode from the label of nodes

To compare the graphs better, I extract opcodes from the existing graphs. And I found they have such opcodes mainly:

{'', 'add', 'srem', 'sub', 'ret', 'store', 'getelementptr', 'br', 'alloca', 'icmp', 'mul', 'sdiv', 'load', '*', 'sext', 'bitcast', 'trunc', 'call'}

* is defined by myself to represent some special operator codes.

1.3 select example of edit distance

To make it more clear, I select two examples to illustrate the efficency of this algorithm.

https://github.com/BryceTsui/UIUCSummerResearch/blob/master/iterate/problems/checksum/08c7ea4ac39aa6a5ab206393bb4412de9d2c365ecdda9c1b391be963c1811014ed23d2722d7433b8e8a95305eee314d39da4950f31e01f9147f90af91a5c433a/006/checksum.c

and

https://github.com/BryceTsui/UIUCSummerResearch/blob/master/iterate/problems/checksum/08c7ea4ac39aa6a5ab206393bb4412de9d2c365ecdda9c1b391be963c1811014ed23d2722d7433b8e8a95305eee314d39da4950f31e01f9147f90af91a5c433a/015/checksum.c

These two codes have slight difference, and they have the same graphs of CFG,DD and CD. It means though they may look different in the code level, they are actually the same in the structure level (i.e CFG, DD and CD). And the edit distance of them is 0. It means the similarity of them is (1-0.0) = 1.0 (100%)

https://github.com/BryceTsui/UIUCSummerResearch/blob/master/iterate/problems/checksum/1b31fa5c50f7725ce468ebf24282f2d080a83aed87e4ee35522ae7710c8e0136bc263cc460b8ec7bf2c3519cb59af4a138e114d36541515b2609ab56ad2b8ee9/004/checksum.c

and

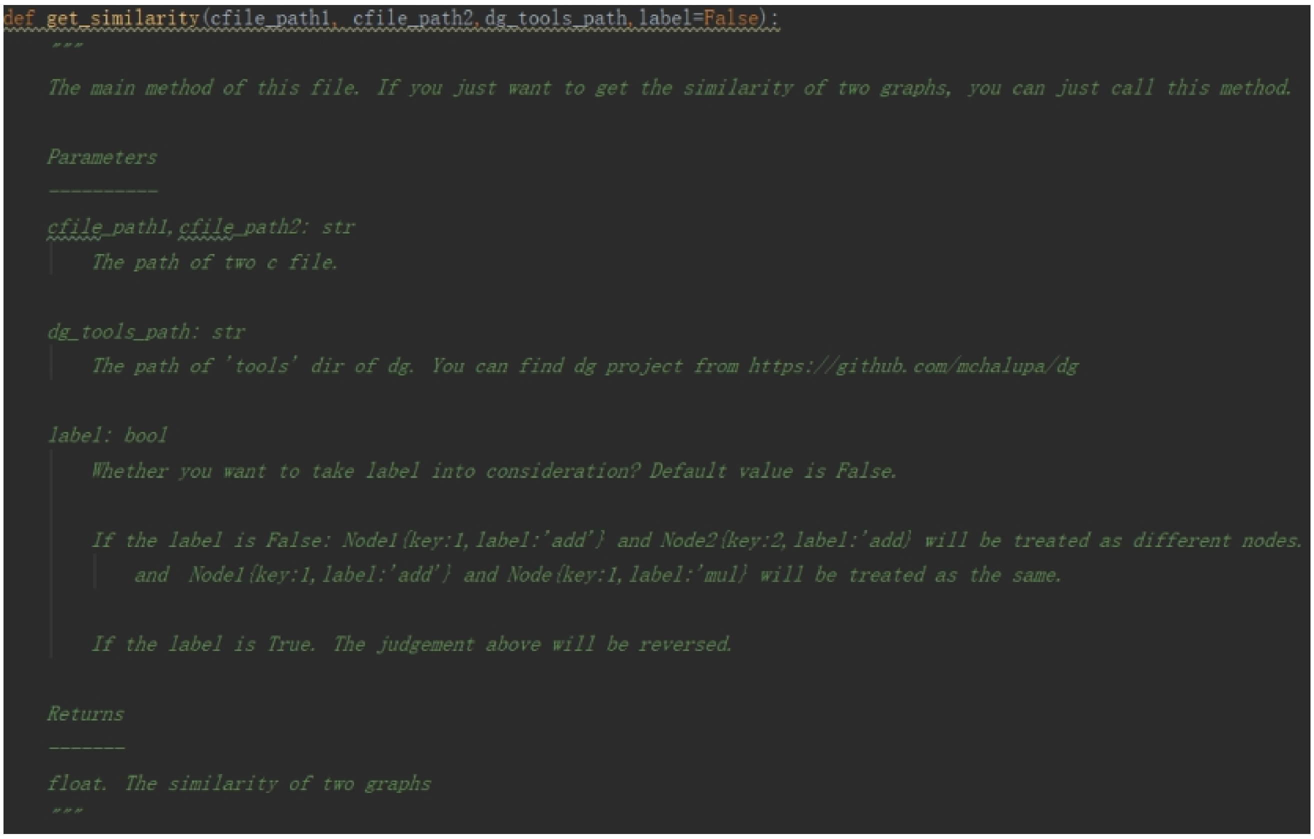
https://github.com/BryceTsui/UIUCSummerResearch/blob/master/iterate/problems/checksum/2c1556672751734adf9a561fbf88767c32224fca14a81e9d9c719f18d0b21765038acc16ecd8377f74d4f43e8c844538161d869605e3516cf797d0a6a59f1f8e/002/checksum.c

These two graphs are most different, because they have worst similarity (0.64).

1.4 standardize the implement of edit distance and write document for this API

To make it more clear, I write a standardized API program with detailed document. You can find this API from:

https://github.com/BryceTsui/UIUCSummerResearch/blob/master/GED/GraphsSimilarity.py


```
def get_similarity(cfile_path1, cfile_path2,dg_tools_path,label=False):
    """
    The main method of this file. If you just want to get the similarity of two graphs, you can just call this method.

    Parameters
    ----------
    cfile_path1,cfile_path2: str
        The path of two c file.

    dg_tools_path: str
        The path of 'tools' dir of dg. You can find dg project from https://github.com/mchalupa/dg

    label: bool
        Whether you want to take label into consideration? Default value is False.

        If the label is False: Node1{key:1,label:'add'} and Node2{key:2,label:'add} will be treated as different nodes.
            and  Node1{key:1,label:'add'} and Node{key:1,label:'mul} will be treated as the same.

        If the label is True. The judgement above will be reversed.

    Returns
    -------
    float. The similarity of two graphs
    """
```

You'd wouder just call get_similarity() method above and not try to call other methods of these program which is used to generate the similarity. If you want to modify the raw code, I also provide detailed annotation for you to read.

1.5 Implement two ways to calculate the similarity

I write two methods to calculate the similarity with_label or without_label.

Label of nodes look like this :

```
label="main::
%3 = alloca i32, align 4"

This method will return alloca.

label="main::
store i32 0, i32* %1, align 4"
```

I write a method to extract main operator code from the label, like alloca or store.

If you want to take label into consideration and use label to identify each node, you can set the label param to True. Then, get_similarity)() will call this method:

```
def getEditDistanceWithLabel(graph1,graph2):
    """
    get edit distance with consideration of label. It means this method considers the label of
each node and
    calculate the edit distance to transfer graph1 to graph2 with the same corresponding label.

    Example:

    node1 : key:1 label:'add', node2: key:2, label'add'. This method with treat these two nodes
as the same.
     If you want method considering these two nodes as different nodes, you need to use the
getEditDistanceWithoutLabel

    Parameters
    ----------
    graph1,graph2: networkx.Graph, networkx.DiGraph or networkx.MultiDiGraph
        Two graphs you want to get the edit distance between them.

    Returns
    -------
    float. representing the edit distance of this two graphs
    """
```

Otherwise, you can set label param to False, and I will call these method；

```
The main method of this file. If you just want to get the similarity of two graphs, you can just
call this method.

Parameters
----------
cfile_path1,cfile_path2: str
    The path of two c file.

dg_tools_path: str
    The path of 'tools' dir of dg. You can find dg project from https://github.com/mchalupa/dg

label: bool
    Whether you want to take label into consideration? Default value is False.

    If the label is False: Node1{key:1,label:'add'} and Node2{key:2,label:'add} will be treated
as different nodes.
        and  Node1{key:1,label:'add'} and Node{key:1,label:'mul} will be treated as the same.

    If the label is True. The judgement above will be reversed.

Returns
-------
float. The similarity of two graphs
"""
```