Readme for calculate part

This part is to calculate and analyse shortest route path using RTT measurements from RIPE Atlas. All relavant modules for this part is in “calculate” folder (OverlayAnalysis/calculate/.). We can execute `main.py` to realize all functions of the part calculate.

Requirements

- Python2.7
- Linux

Installation

```
sudo pip install pandas
sudo pip install mpld3
```

Usage

1. Main

The main script, `OverlayAnalysis/calculate/main.py` includes three modules which can realize all functions in this part. It combines all functions in rttShortest.py, rttCalculateToShow.py and rttDisplayMaskedArray.py modules and make essier to creat html files for graphs.

1.1 Main Input file

The `main.py` takes a csv file as its input, `OverlayAnalysis/data/dataID/dataID.csv`. This file is generated by RTT Measurements module. It contains all RTT values for all paires of couples and all times measures. With the following format:

Sondes	**0**	**1**	**2**	**3**	**…**
0	0	47	20	30	…
1	47	0	7	18	…
2	20	7	0	23	…
3	30	18	23	0	…
-------	----	----	----	----	----
0	0	47	20	30	…
1	47	0	7	18	…
2	20	7	0	23	…
3	30	18	23	0	…
-------	----	----	----	----	----

1.2 Execution main

Parameter:

- dataID : Thers is only parameter to change for diffferent dataID, dataID = “20160808_161300” (par default) in th head of main function. The dataID is the beginning time of the measurement.

- type of dataID : String

Run with Python2.7 (or higher): `OverlayAnalysis/calculate/main.py`.

1.3 Main output files

It will generate all graphs we need by call the display functions.

Exemples:

```
    showPathInformation(4, 2, informationDict, myRttDisplay.picturesPath, myRttDisplay.htmlPath)
    showPathInformation(13, 18, informationDict, myRttDisplay.picturesPath, myRttDisplay.htmlPath)#no valide measure for this link
    showPathInformation(19, 5, informationDict, myRttDisplay.picturesPath, myRttDisplay.htmlPath)
    showPathInformation(7, 6, informationDict, myRttDisplay.picturesPath, myRttDisplay.htmlPath)
    showPathInformation(3, 11, informationDict, myRttDisplay.picturesPath, myRttDisplay.htmlPath)#no valide measure for this link

    """generate fix graphs for app"""
    showMatrixMeanDelays(rtt3MeanDelay, rtt3MeanShortestDelay, myRttDisplay.nbProbes,myRttDisplay.picturesPath, my
RttDisplay.htmlPath)
    showMatrixDiffRtt(rtt3DiffPercentNew, myRttDisplay.nb
Probes, rtt3MaxDiffPercent, rtt3MinDiffPercent, rtt3MeanD
iffPercent, myRttDisplay.picturesPath, myRttDisplay.htmlP
ath)
    showMatrixDiffRttMinRttPathLength(rtt3ShortestAllData
TimeNew, rtt3AllDataNew, rtt3PathLengthNew, rtt3DiffPerce
ntNew,myRttDisplay.nbProbes, myRttDisplay.picturesPath, m
yRttDisplay.htmlPath)
    showMatrixCovRtt(rtt3ShortestAllDataTimeNew ,myRttDis
play.nbProbes , myRttDisplay.picturesPath, myRttDisplay.h
tmlPath)
    showHistoDifPercent(rtt3DiffPercentNew, "all data", m
yRttDisplay.nbProbes, myRttDisplay.picturesPath, myRttDis
play.htmlPath)
    showHistoPathLengthAllCouples(rtt3PathLengthNew, "all
 data", myRttDisplay.nbProbes, myRttDisplay.picturesPath,
 myRttDisplay.htmlPath)
    showCumulativenbTimesPercentDifRTT(rtt3DiffPercentNew
, myRttDisplay.nbProbes, myRttDisplay.picturesPath, myRtt
Display.htmlPath)
    showCumulativeNbcouplesPercentDifRTT(rtt3MeanDiffPerc
ent, rtt3MaxDiffPercent, rtt3MinDiffPercent, myRttDisplay
.nbProbes, myRttDisplay.picturesPath, myRttDisplay.htmlPa
th)

    """test chnageable graphs for app"""
    showDefaultTime(rtt3AllDataNew, 8, 1, myRttDisplay.pi
cturesPath, myRttDisplay.htmlPath)
    showHistoDefaultTime(rtt3AllDataNew, 8, 1, myRttDispl
ay.picturesPath, myRttDisplay.htmlPath)
    plotSrcFixedForVisu(rtt3MinDiffPercent, rtt3MeanDiffP
ercent, rtt3MaxDiffPercent, 8, myRttDisplay.nbProbes, myR
ttDisplay.picturesPath, myRttDisplay.htmlPath)

    ##shortest route graphs for one link
    showDefaultTimeShortestTime(rtt3AllDataNew, rtt3Short
estAllDataTimeNew, 8, 1, myRttDisplay.picturesPath, myRtt
Display.htmlPath)
    showPathLength(rtt3PathLengthNew, 8, 1, myRttDisplay.
picturesPath, myRttDisplay.htmlPath)
    showHistoPathLength(rtt3PathLengthNew, 8, 1, myRttDis
play.picturesPath, myRttDisplay.htmlPath)

```

and save them by html into `OverlayAnalysis/data/dataID/graphs/.` The graphs will be used once we run our web application.

2. Details of main

As main script `calculate/main.py` combines three modules functions and classes. First of them, RttShortest module is to calculate the shortest path and save relative data into csv files. Second of them, RttCalculateToShow is to creat json files in which we stock all data for genarating graphs we need. Last one, RttDisplayMaskedArray is to generate all graphs by using json data.

2.1 Calculate shortest path: RttShortest

2.1.1 input file

Csv file as main's input, `OverlayAnalysis/data/dataID/dataID.csv`. With the following format:

Sondes	**0**	**1**	**2**	**3**	**...**
0	0	47	20	30	...
1	47	0	7	18	...
2	20	7	0	23	...
3	30	18	23	0	...
-------	----	----	----	----	----
0	0	47	20	30	...
1	47	0	7	18	...
2	20	7	0	23	...
3	30	18	23	0	...
-------	----	----	----	----	----

2.1.2 Code in main.py

Code:

```
    # Calculate shortest time and information for shortest route path, all functions used are imported from rttShortest.py
    csvFilesList = [dataID +"/calculateData/ShortestPathLength.csv", dataID+"/calculateData/ShortestTime.csv", dataID+"/calculateData/informationDictResult.json"]
    filesExist = [f for f in csvFilesList if (os.path.isfile(f) and os.path.getsize(f)>0)]
    filesNoExist = list(set(filesExist)^set(csvFilesList))
    if filesNoExist:
        myRtt = RttShortest(dataID)
        myRtt.caculateAllData()
    else:
        print "csv files exist"
```

By these lines of code, it will justify if all output files for this part already exist. It will check fileNoExist if it is NONE, then it will calculate the shortest route path.

2.1.3 output file

`OverlayAnalysis/data/dataID/calculateData/ShortestPathLength.c` is to stock shortest path length, `OverlayAnalysis/data/dataID/calculateData/ShortestTime.csv` is to stock shortest path delay. Both of them respect the following form:

Sondes	**0**	**1**	**2**	**3**	**...**
0	0	47	20	30	...
1	47	0	7	18	...
2	20	7	0	23	...
3	30	18	23	0	...
-------	----	----	----	----	----
0	0	47	20	30	...
1	47	0	7	18	...
2	20	7	0	23	...
3	30	18	23	0	...
-------	----	----	----	----	----

`OverlayAnalysis/data/dataID/calculateData/informationDictResul`
is to stock every shortest route possible and their average value of time and average value of dfference between direct delay and shortest delay expressed by percentage. With the following format:

```
{
"1-2":
    [
    {"1-3-2": {"nbtimes":56, "meanTime": 125.973888(ms),
"meanDiffPercent": 25.88(%)}},
    {"1-8-2": {"nbtimes":56, "meanTime": 125.973888(ms),
"meanDiffPercent": 25.88(%)}},
      ...
    ]
"1-3":
    [
    {"1-2-3": {"nbtimes":56, "meanTime": 125.973888(ms),
"meanDiffPercent": 25.88(%)}},
    {"1-7-3": {"nbtimes":56, "meanTime": 125.973888(ms),
"meanDiffPercent": 25.88(%)}}
    ...
    ]
...
}
```

2.2 Calculate to show : RttCalculateToShow

This part is to calculate data for display functions. All data output will be stocked into json files.

2.2.1 input files

`OverlayAnalysis/data/dataID/calculateData/ShortestTime.csv` and `OverlayAnalysis/data/dataID/calculateData/ShortestPathLength.c`

2.2.2 Code in main

Code:

```
# calculate to show, calculation function used imported from rttCalculateToShow.py
    jsonfilesList = [dataID+'/output/AllData.json', dataID+"/output/ShortestTime.json", dataID+"/output/ShortestPathLength.json", dataID+'/output/DiffPercent.json', dataID+'/output/MaxdiffPercent.json', dataID+'/output/MindiffPercent.json', dataID+'/output/MeandiffPercent.json', dataID+'/output/MeanDelay.json', dataID+'/output/MeanShortestDelay.json', dataID+'/output/information.json' ]
    filesExist = [f for f in jsonfilesList if (os.path.isfile(f) and os.path.getsize(f)>0)]
    filesNoExist = list(set(filesExist)^set(jsonfilesList))
    if filesNoExist:
        myRttCalculateToShow = RttCalculateToShow(dataID)
        """transform all data into three dimensions array list as rtt2[index of time][src][dst]"""
        rtt2AllData = data3D(myRttCalculateToShow.allData, myRttCalculateToShow.nbprobes, myRttCalculateToShow.nbtimes)
        rtt2ShortestAllDataTime = data3D(myRttCalculateToShow.shortestAllDataTime, myRttCalculateToShow.nbprobes, myRttCalculateToShow.nbtimes)
        rtt2PathLength = data3D(myRttCalculateToShow.pathLength, myRttCalculateToShow.nbprobes, myRttCalculateToShow.nbtimes)
        """transform all data into three dimensions array
```

```
 list as rtt3[src][dst][index of time]"""
        rtt3AllData = data3D2(rtt2AllData, myRttCalculate
ToShow.nbprobes, myRttCalculateToShow.nbtimes)
        rtt3ShortestAllDataTime = data3D2(rtt2ShortestAll
DataTime, myRttCalculateToShow.nbprobes, myRttCalculateTo
Show.nbtimes)
        rtt3PathLength = data3D2(rtt2PathLength, myRttCal
culateToShow.nbprobes, myRttCalculateToShow.nbtimes)
        rtt3Difference=rtt3AllData-rtt3ShortestAllDataTim
e
        rtt3DiffPercent=getImprovement(rtt3Difference, rt
t3AllData)
        """get the masked arrays according to valid value
s, replace masked values by -99999"""
        rtt3AllDataNew=getMaskedArray(rtt3AllData, rtt3Al
lData,(0,2000))
        #rtt3DifferenceNew=getMaskedArray(rtt3AllData, rt
t3Difference,(0,2000))
        rtt3ShortestAllDataTimeNew=getMaskedArray(rtt3All
Data, rtt3ShortestAllDataTime,(0,2000))
        rtt3PathLengthNew=getMaskedArray(rtt3AllData, rtt
3PathLength,(0,2000))
        rtt3DiffPercentNew=getMaskedArray(rtt3AllData, rt
t3DiffPercent,(0,2000))
        """ Calculate maxValues and minValues of rtt3Diff
Percent """
        rtt3MaxdiffPercent = np.ma.amax(rtt3DiffPercentNe
w, axis=2)
```

```
        rtt3MindiffPercent = np.ma.amin(rtt3DiffPercentNe
w, axis=2)
        """calculate meanValues of rtt3DiffPercent """

        rtt3MeanDiffPercent = np.ma.mean(rtt3DiffPercentN
ew, axis=2)
        """ Calculate maxValues and minValues of rtt3AllD
ata """
        rtt3MaxDelay = np.ma.amax(rtt3AllDataNew, axis=2)
        rtt3MinDelay = np.ma.amin(rtt3AllDataNew, axis=2)
        """calculate meanValues of rtt3AllData """

        rtt3MeanDelay = np.ma.mean(rtt3AllDataNew, axis=2
)
        """ Calculate maxValues and minValues of rtt3Shor
testAllDataTimeNew """
        rtt3MaxShortestDelay = np.ma.amax(rtt3ShortestAll
DataTimeNew, axis=2)
        rtt3MinShortestDelay = np.ma.amin(rtt3ShortestAll
DataTimeNew, axis=2)
        """calculate meanValues of rtt3ShortestAllDataTim
eNew """
        rtt3MeanShortestDelay = np.ma.mean(rtt3ShortestAl
lDataTimeNew, axis=2)

        if dataID+'/output/information.json' in filesNoEx
ist:
            linkNoValide = []
            for src in range (myRttCalculateToShow.nbprob
es):
                for dst in range (myRttCalculateToShow.nb
probes):
                    if np.ma.count(rtt3AllDataNew[src][ds
t])==0 and src != dst:
                        linkNoValide.append(str(src)+"-"+
str(dst))

            """write data information (nbProbes, nbTimes)
 into a json file"""
            info = dict()
            info['nbProbes'] = myRttCalculateToShow.nbpro
bes
            info['nbTimes'] = myRttCalculateToShow.nbtime
s
            info['Total non-valid measures (%) '] = str(f
loat(np.ma.count_masked(rtt3AllDataNew))/(np.ma.count(rtt
3AllDataNew)+np.ma.count_masked(rtt3AllDataNew))*100)
            info['Mean Improvement (%) '] = str(getMeanIm
provement(rtt3DiffPercentNew))
            info['No valide measure links'] = linkNoValid
e
            print os.getcwd()
            with open(dataID +'/output/information.json',
 'w') as fp:
                json.dump(info, fp)
```

```
        """Save data into corresponding json files """
        if dataID+'/output/AllData.json' in filesNoExist:
            myRttCalculateToShow.rtt3AllDataFile.write_da
ta(rtt3AllDataNew)
        if dataID+'/output/ShortestTime.json' in filesNoE
xist:
            myRttCalculateToShow.rtt3ShortestTimeFile.wri
te_data(rtt3ShortestAllDataTimeNew)
        if dataID+'/output/ShortestPathLength.json' in fi
lesNoExist:
            myRttCalculateToShow.rtt3LengthPathFile.write
_data(rtt3PathLengthNew)
        #if dataID+'/output/Difference.json' in filesNoEx
ist:
            #myRttCalculateToShow.rtt3DifferenceFile.writ
e_data(rtt3DifferenceNew)
        if dataID+'/output/DiffPercent.json' in filesNoEx
ist:
            myRttCalculateToShow.rtt3DiffPercentFile.writ
e_data(rtt3DiffPercentNew)
        if dataID+'/output/MaxdiffPercent.json' in filesN
oExist:
            myRttCalculateToShow.rtt3MaxdiffPercentFile.w
rite_data(rtt3MaxdiffPercent)
        if dataID+'/output/MindiffPercent.json' in filesN
oExist:
            myRttCalculateToShow.rtt3MindiffPercentFile.w
rite_data(rtt3MindiffPercent)
```

```
        if dataID+'/output/MeandiffPercent.json' in files
NoExist:
            myRttCalculateToShow.rtt3MeandiffPercentFile.
write_data(rtt3MeanDiffPercent)
        if dataID+'/output/MeanDelay.json' in filesNoExis
t:
            myRttCalculateToShow.rtt3MeanShortestDelayFil
e.write_data(rtt3MeanDelay)
        if dataID+'/output/MeanShortestDelay.json' in fil
esNoExist:
            myRttCalculateToShow.rtt3MeanDelayFile.write_
data(rtt3MeanShortestDelay)
    else:
        print "json files exist"
```

2.2.3 Output files

All output files are json files. We can write our masked array directly into json files and read files return masked array by using JsonFile module, `OverlayAnalysis/calculate/fileTools/jsonfile.py`.

Output files name
`OverlayAnalysis/data/dataID/output/AllData.json`
`OverlayAnalysis/data/dataID/output/ShortestTime.json`
`OverlayAnalysis/data/dataID/output/ShortestPathLength.json`
`OverlayAnalysis/data/dataID/output/DiffPercent.json`

`OverlayAnalysis/data/dataID/output/MaxdiffPercent.json`
`OverlayAnalysis/data/dataID/output/MindiffPercent.json`
`OverlayAnalysis/data/dataID/output/MeandiffPercent.json`
`OverlayAnalysis/data/dataID/output/MeanShortestDelay.json`

Except these eight data files for display functions, another information is also one output of this part.

Information file
`OverlayAnalysis/data/dataID/output/information.json`, With the following format:
{"Total non-valid measures (%) ": “35.7749007937”, “nbProbes”: 20, "Mean Improvement (%) ": “8.29684412461”, “nbTimes”: 5040, “No valide measure links”: [“11-12”, “11-13”, “11-14”, “11-15”, “11-16”, “11-17”, “11-18”, “11-19”, “12-11”, “13-11”, “13-14”, “13-15”, “13-16”, “13-17”, “13-18”, “13-19”, “14-11”, “14-13”, “15-11”, “15-13”, “16-11”, “16-13”, “17-11”, “17-13”, “18-11”, “18-13”, “19-11”, “19-13”]}

2.3 display and generate html : RttDisplayPlugins

This part is to read all json files for display and save graphs into the corresponding directory.

2.3.1 input files

All input files are generated by two modules RttShortest and

RttCalculateToShow that are represented before.

Input files name
`OverlayAnalysis/data/dataID/calculateData/informationDictResu`
`OverlayAnalysis/data/dataID/output/AllData.json`
`OverlayAnalysis/data/dataID/output/ShortestTime.json`
`OverlayAnalysis/data/dataID/output/ShortestPathLength.json|`
`OverlayAnalysis/data/dataID/output/DiffPercent.json`
`OverlayAnalysis/data/dataID/output/MaxdiffPercent.json`
`OverlayAnalysis/data/dataID/output/MindiffPercent.json`
`OverlayAnalysis/data/dataID/output/MeandiffPercent.json`
`OverlayAnalysis/data/dataID/output/MeanShortestDelay.json`

2.3.2 Code in main.py

Code:

```
    #generate html graphs, display functions used importe
d from rttDisplayPlugins.py
    myRttDisplay = RttDisplay(dataID)
    print('Total non-valid measures (%) '+ myRttDisplay.i
nfo['Total non-valid measures (%) '])
    print('Mean Improvement (%) '+ myRttDisplay.info['Mea
n Improvement (%) '])
```

```
    """Load data from json files"""
    rtt3AllDataNew = myRttDisplay.rtt3AllDataFile.read_data()
    rtt3ShortestAllDataTimeNew = myRttDisplay.rtt3ShortestTimeFile.read_data()
    rtt3PathLengthNew = myRttDisplay.rtt3LengthPathFile.read_data()
    #rtt3DifferenceNew = myRttDisplay.rtt3DifferenceFile.read_data()
    rtt3DiffPercentNew = myRttDisplay.rtt3DiffPercentFile.read_data()
    """Load data from json files"""
    rtt3MaxDiffPercent = myRttDisplay.rtt3MaxdiffPercentFile.read_data()
    rtt3MinDiffPercent = myRttDisplay.rtt3MindiffPercentFile.read_data()
    rtt3MeanDiffPercent = myRttDisplay.rtt3MeandiffPercentFile.read_data()
    rtt3MeanDelay = myRttDisplay.rtt3MeanDelayFile.read_data()
    rtt3MeanShortestDelay = myRttDisplay.rtt3MeanShortestDelayFile.read_data()


    """Shortest path information plot"""
    with open (dataID +"/calculateData/informationDictResult.json", "r") as fs:
        print 'enter read'
        informationDict = json.load(fs)
    showPathInformation(4, 2, informationDict, myRttDisplay.picturesPath, myRttDisplay.htmlPath)
    showPathInformation(13, 18, informationDict, myRttDisplay.picturesPath, myRttDisplay.htmlPath)#no valide measure for this link
    showPathInformation(19, 5, informationDict, myRttDisplay.picturesPath, myRttDisplay.htmlPath)
    showPathInformation(7, 6, informationDict, myRttDisplay.picturesPath, myRttDisplay.htmlPath)
    showPathInformation(3, 11, informationDict, myRttDisplay.picturesPath, myRttDisplay.htmlPath)#no valide measure for this link

    """generate fix graphs for app"""
    showMatrixMeanDelays(rtt3MeanDelay, rtt3MeanShortestDelay, myRttDisplay.nbProbes,myRttDisplay.picturesPath, myRttDisplay.htmlPath)
    showMatrixDiffRtt(rtt3DiffPercentNew, myRttDisplay.nbProbes, rtt3MaxDiffPercent, rtt3MinDiffPercent, rtt3MeanDiffPercent, myRttDisplay.picturesPath, myRttDisplay.htmlPath)
    showMatrixDiffRttMinRttPathLength(rtt3ShortestAllDataTimeNew, rtt3AllDataNew, rtt3PathLengthNew, rtt3DiffPercentNew,myRttDisplay.nbProbes, myRttDisplay.picturesPath, myRttDisplay.htmlPath)
    showMatrixCovRtt(rtt3ShortestAllDataTimeNew ,myRttDisplay.nbProbes , myRttDisplay.picturesPath, myRttDisplay.h
tmlPath)
    showHistoDifPercent(rtt3DiffPercentNew, "all data", myRttDisplay.nbProbes, myRttDisplay.picturesPath, myRttDisplay.htmlPath)
    showHistoPathLengthAllCouples(rtt3PathLengthNew, "all data", myRttDisplay.nbProbes, myRttDisplay.picturesPath, myRttDisplay.htmlPath)
    showCumulativenbTimesPercentDifRTT(rtt3DiffPercentNew, myRttDisplay.nbProbes, myRttDisplay.picturesPath, myRttDisplay.htmlPath)
    showCumulativeNbcouplesPercentDifRTT(rtt3MeanDiffPercent, rtt3MaxDiffPercent, rtt3MinDiffPercent, myRttDisplay.nbProbes, myRttDisplay.picturesPath, myRttDisplay.htmlPath)


    """test chnageable graphs for app"""
    showDefaultTime(rtt3AllDataNew, 8, 1, myRttDisplay.picturesPath, myRttDisplay.htmlPath)
    showHistoDefaultTime(rtt3AllDataNew, 8, 1, myRttDisplay.picturesPath, myRttDisplay.htmlPath)
    plotSrcFixedForVisu(rtt3MinDiffPercent, rtt3MeanDiffPercent, rtt3MaxDiffPercent, 8, myRttDisplay.nbProbes, myRttDisplay.picturesPath, myRttDisplay.htmlPath)


    ##shortest route graphs for one link
    showDefaultTimeShortestTime(rtt3AllDataNew, rtt3ShortestAllDataTimeNew, 8, 1, myRttDisplay.picturesPath, myRttDisplay.htmlPath)
    showPathLength(rtt3PathLengthNew, 8, 1, myRttDisplay.picturesPath, myRttDisplay.htmlPath)
    showHistoPathLength(rtt3PathLengthNew, 8, 1, myRttDisplay.picturesPath, myRttDisplay.htmlPath)
    """test old function not for app"""
    plotSrcFixed(rtt3DiffPercentNew, rtt3MeanDiffPercent, 8, myRttDisplay.nbProbes, myRttDisplay.picturesPath, myRttDisplay.htmlPath)
```

2.3.3 Output files

All analysis graphs for all links and all measures are in the folder `OverlayAnalysis/data/dataID/graphs/.` The other graphs for one link or one origin are generated in the folder `OverlayAnalysis/data/dataID/graphs/generated/.`