Mining Chrome Repository

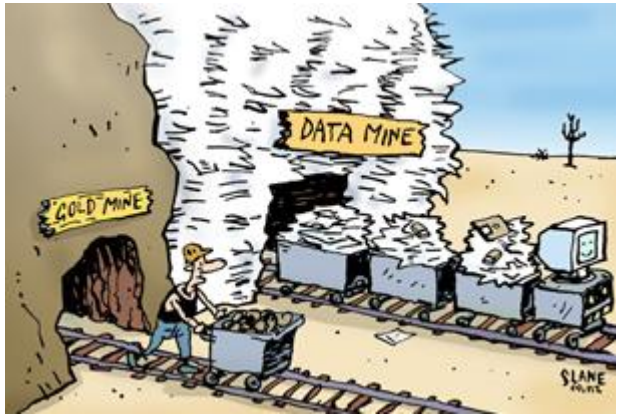
Group 17:

Cem Yusuf Aydoğdu

İrem Ertürk

İstanbul Technical University

May 15, 2016



Required Data Sets

- File Name List
  - with folders: *$ls -ld $(find .) | awk '{print $9}' | sed 's/^..//' | sed -n 'p;$='*
  - without folders: *$ls -ldp $(find .) | grep -v '/$' | awk '{print $9}' | sed 's/^..//'*
- Developer List
  - alphabetic developer names:

    $git log --pretty=format:"%cn" | sort | uniq
  - developer names and comit counts in numeric order:

    $git log --pretty=format:"%cn" | sort | uniq --count | sort -nr

Implementation Details: Top Developers

```
developer_list="440_F_DeveloperList_Numeric.txt"

declare -i total_commit
total_commit="$(git rev-list --all --count)"


findTopDevelopers(){ #$1=total_commit, $2=%x
        declare -i thresold
        declare -i temp
        temp=0
        thresold=total_commit*$2/100


        while IFS= read -r line  #while read line;
        do
                current=$(echo $line | awk '{print $1}')
                percent=$(echo "$(echo "($current*100)/$total_commit" | bc -l)" |
awk '{printf "%.10f\n",$1}')

                temp=$(($temp+$current))
                dev=$(echo $line | awk '{ for(i=2; i<NF; i++) printf "%s ", $i OFS;
if(NF) printf "%s",$NF; printf ORS}')
                echo -e $percent '\t' $dev

                if [ $temp -ge $thresold ]
                then
                        break
                fi
        done < "$developer_list"  #<<<"$(git log --pretty=format:\"%cn\" | sort |
uniq --count | sort -nr)"
}
```

Implementation Details: Top Edited Files

```
topfile_list="440_F_FilesCommitCounts_sorted.txt"
findTopFiles(){
        declare -i thresold
        declare -i temp
        temp=0
        thresold=total_commit*$2/100


        while IFS= read -r line
        do
                #echo $line
                #echo "$(git log --oneline -- $line | wc -l)" $line
                current=$(echo $line | awk '{print $1}')
                percent=$(echo "$(echo "($current*100)/$total_commit" | bc -l)" |
awk '{printf "%.10f\n",$1}')

                temp=$(($temp+$current))
                dev=$(echo $line | awk '{ for(i=2; i<NF; i++) printf "%s ", $i OFS;
if(NF) printf "%s",$NF; printf ORS}')
                echo -e $percent '\t' $dev

                if [ $temp -ge $thresold ]
                then
                        break
                fi

        done < "$topfile_list"

}
```

```
file_list="440_F_FileList.txt"

findCommitCountOfFiles(){

        while IFS= read -r line
        do
                #echo $line
                echo "$(git log --oneline -- $line | wc -l)" $line
        done < "$file_list"
}
```

Implementation Details: Create Adjacency Matrix

```
# print commiter names in the first line
declare -i index
index=0
while IFS= read -r name
do
        names["\"$name\""]=$index
        ((index++))
done < "$committer_list"
```

Map each developers name to an integer value and use it whenever the developer contibute the change in file.

```
initializeMatrix()
{
        local index=0
        local total=$(($row_number * $column_number))

        while [ "$index" -lt "$total" ]
        do
                AdjMatrix[$index]=0
                let "index += 1"
        done

}
```

To create the adjacency matrix for representing contributions to each files, we first create and initialize the Adjacency matrix with all zeros by using the initializeMatrix() function. Rows represent the files and the columns represents the developers.

Implementation Details: Create Adjacency Matrix

```
modifyValue()   ## $1=row $2=col $3=value
{
        local index=0

        let "index = $1*$column_number + $2"
        AdjMatrix["$index"]=$3
}
```

Modify value fuction takes the row, column and new_value as parameters. Because we store adjacency matrix in one one dimensional array rather than 2d array, we calculate the index by using row and column values. And change the value with one.

Implementation Details: Create Adjacency Matrix

```
while IFS= read -r item
do

        initializeMatrix

        while read n;
        do

            temp=${names[$n]}

             modifyValue 0 ${names[$n]} 1

        done <<<"$(git log  --pretty=format:\"%cn\" -- $item | sort | uniq )"

        printRow

done < "$file list"
```

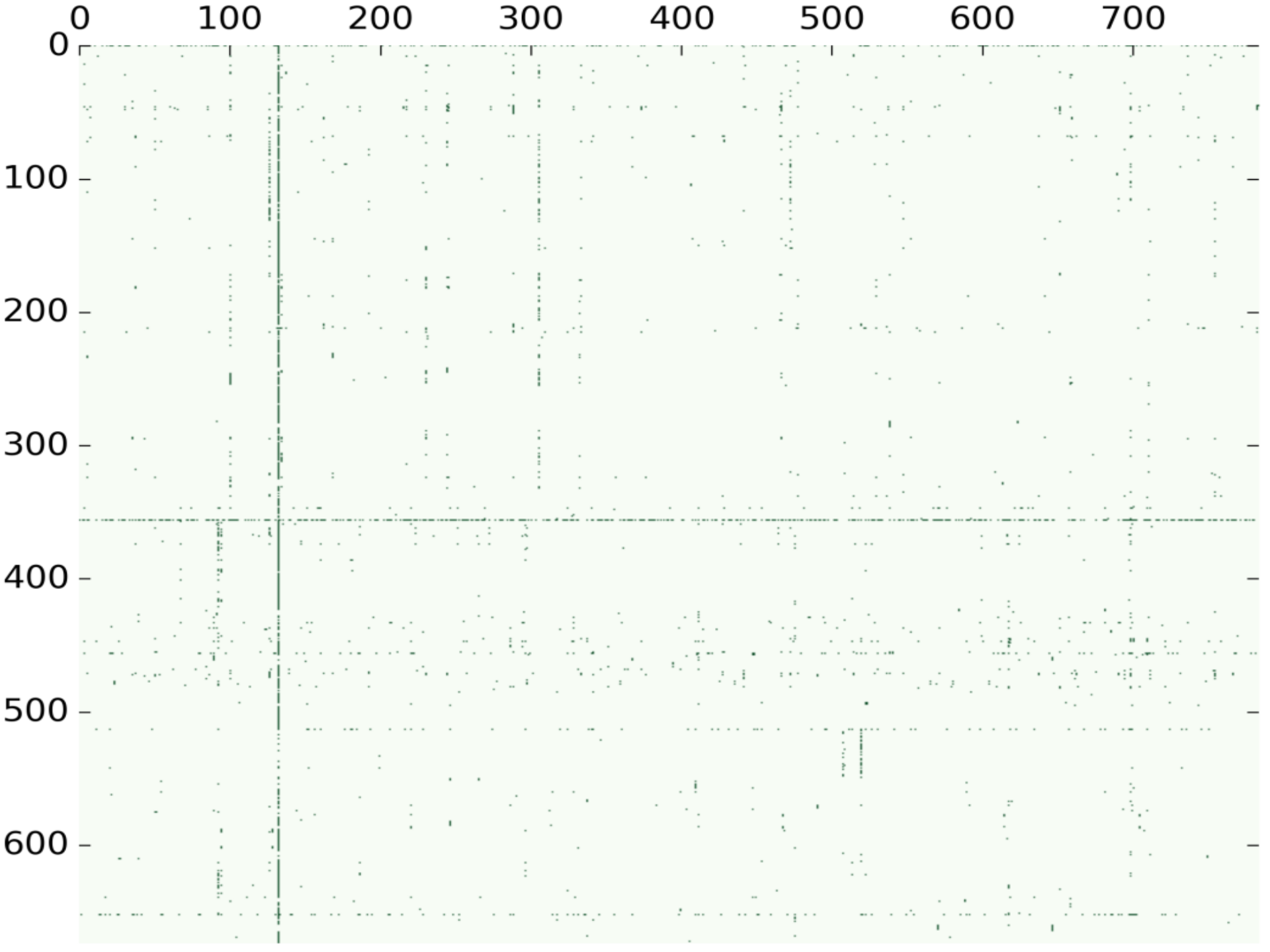
The main flow of algorithm is done here. The blue highlighted code loops through the "440_F_FileList.txt" file and for each file finds the file contributers. As you can see in the inner while loop we call the modifyValuefunction to change value in adjacency matrix. After all contributers of files modified in inner loop we print the rows.

Implementation Details: Matrix Visualization

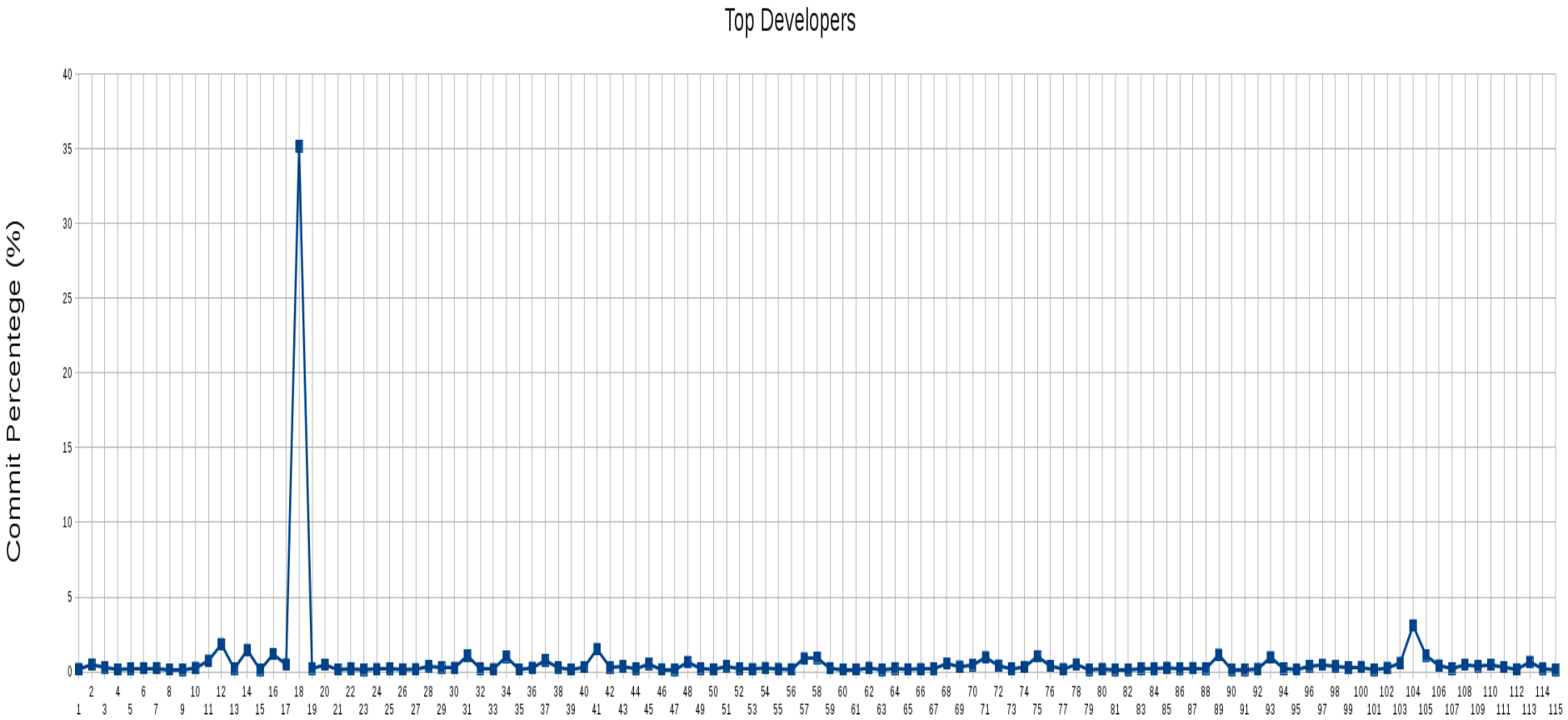
```
# -*- coding: utf-8 -*-
import matplotlib.pyplot as plt
import numpy as np
input_file="440_FO_Matrix.txt"
matrix=[]

# Read file into an array
with open(input_file) as f:
        for line in f:
                matrix.append(line.split())

# Copy matrix into numpy array
numpy_array=np.array(matrix, dtype=np.float)

# Plot numpy array
fig = plt.figure()
ax= fig.add_subplot(1,1,1)
ax.imshow(numpy_array, cmap=plt.cm.Greens, origin="lower")
ax.xaxis.tick_top()
ax.spines['top'].set_visible(False)
ax.spines['right'].set_visible(False)
ax.spines['bottom'].set_visible(False)
ax.spines['left'].set_visible(False)
plt.gca().invert_yaxis()

# Show and write
plt.show()
fig.savefig("adjMatrix_output.png", dpi=350)
```

At the last step of our project, we encounter with problem about visualization tools. First of all, there exists no download link in given SocialAction website. Then, we try to use Gephi social network visualization tool.

However, its comma separated format is not suitable for our adjacency matrix format. For that reason we write a python code which represents our adjacency matrix.

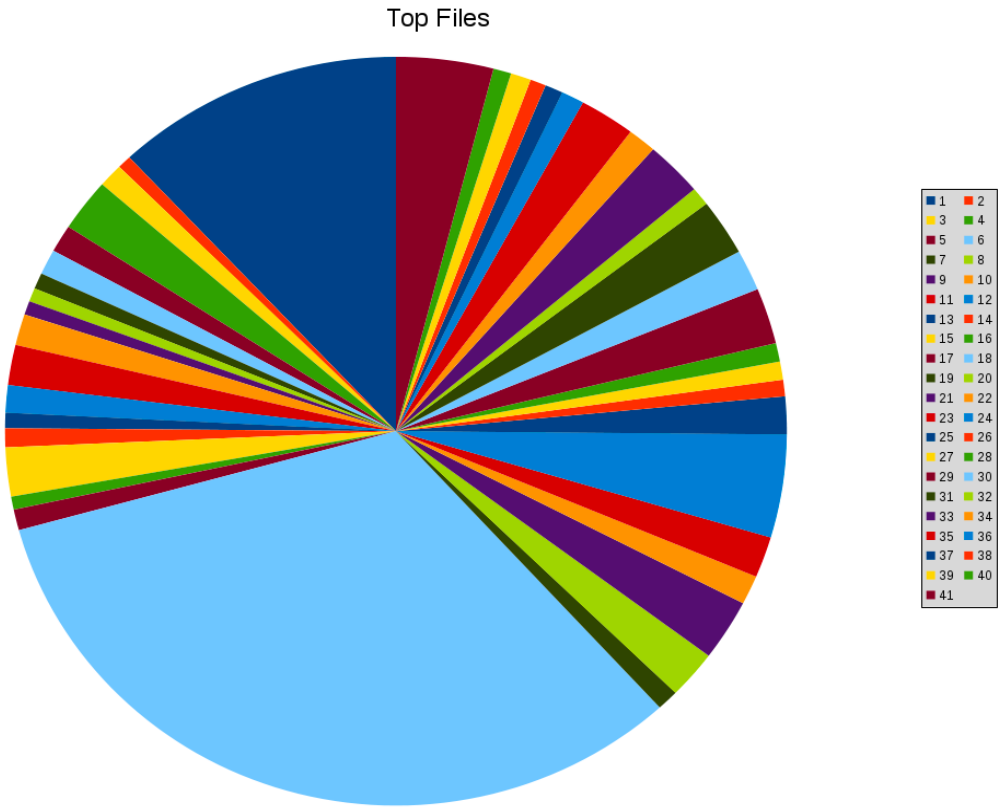

0
100
200
300
400
500
600
700
0
100
200
300
400
500
600

Results: Top Developers



35.1569671379	Commit bot
3.0934459335	thakis@chromium.org
1.8450523224	brettw@chromium.org
1.5145951900	ilevy@chromium.org
1.4503396365	bulach@chromium.org
1.1749586929	cjhopman@chromium.org

Results: Top Files



%	File
9.5557187443	all.gyp
0.4589682394	android/adb_install_apk.py
0.8169634661	android/buildbot/bb_device_status_check.py
1.817514228	android/buildbot/bb_device_steps.py
0.9454745732	android/buildbot/bb_run_bot.py
0.8720396549	android/envsetup.sh
0.5507618873	android/gyp/javac.py
0.4681476042	android/gyp/util/build_utils.py
0.477326969	android/PRESUBMIT.py
1.0739856802	android/provision_devices.py
1.386084083	android/pylib/device/device_utils.py
0.9454745732	android/pylib/gtest/filter/content_browsertests_disabled
0.5324031577	android/pylib/gtest/gtest_config.py
0.6517348999	android/pylib/perf/test_runner.py
1.689003121	android/test_runner.py
0.4497888746	android/tombstones.py
0.7159904535	build_config.h
26.0234991739	common.gypi
0.7159904535	config/android/config.gni
1.6614650266	config/android/internal_rules.gni
2.1112539012	config/android/rules.gni
1.0005507619	config/BUILDCONFIG.gn
1.4136221773	config/BUILD.gn
3.5524141729	config/compiler/BUILD.gn
1.3034697999	config/features.gni
0.5691206169	config/linux/BUILD.gn
0.6241968056	config/win/BUILD.gn
0.6425555352	get_landmines.py
1.9276666055	gn_migration.gypi
1.4136221773	gyp_chromium
1.946025335	install-build-deps.sh
0.6333761704	isolate.gypi
1.9093078759	java_apk.gypi
0.9454745732	java.gypi
1.8450523224	linux/system.gyp
0.7527079126	sanitizers/tsan_suppressions.cc
0.5966587112	toolchain/gcc_toolchain.gni
0.5232237929	toolchain/mac/BUILD.gn
0.6700936295	toolchain/win/BUILD.gn
0.5966587112	vs_toolchain.py
3.2127776758	whitespace_file.txt

Thank You For Listening