Hyperscan vs Regex

测试环境

```
- GCC, v7.5.0
- Ubuntu 18.04.5 LTS
- cmake v3.19.1
- cpu support AVX2 and AVX512
```

测试步骤

1. 编译hyperscan并且打开 AVX2 和 AVX512，参考docs

```
cmake -DBUILD_AVX512=on -DBUILD_AVX2=on --build ..
```

2. 从命令行输入正则匹配模板pattern和匹配文本file。

```
# hyperscan
./main $(pattern) $(file)

# regex
./mysimplegrep $(pattern) $(file)

# pattern: 正则匹配模板
# file: 匹配文本的路径
```

3. 分别记录hyperscan和regex匹配扫描的耗时，并在命令行打印匹配耗时和搜查结果的索引。

```
    clock_t start,end;

    // start the timer
    start=clock();

    // scan the text
    if (hs_scan(database, inputData, length, 0, scratch, eventHandler,
                pattern) != HS_SUCCESS) {
        fprintf(stderr, "ERROR: Unable to scan input buffer. Exiting.\n");
        hs_free_scratch(scratch);
        free(inputData);
        hs_free_database(database);
        return -1;
    }
```

```
    //close the timer
    end=clock();
    double time=(double)(end-start)/CLOCKS_PER_SEC;

    // print consumed time
    printf("%lf",time);
```

4. 更换匹配模板pattern重复步骤1-2

```
import time
import subprocess

# match patterns
patterns=['d.cker','bu.ld','i..ge','ports','net.ork','co..and','us.r','hos[a-
z]name','cont.iner','edg[a-z]x-network']
res_file=open("res.csv","w")
writer=csv.writer(res_file)

for i in range(len(patterns)):

    # execute hyperscan matching
    shell_command="./main %s ./test.yaml"%(patterns[i])
    child1=subprocess.Popen(shell_command,shell=True,stdout=subprocess.PIPE)
    time1=child1.stdout.read().decode()

    # execute regex matching
    shell_command="./mysimplegrep %s ./test.yaml"%(patterns[i])
    child2=subprocess.Popen(shell_command,shell=True,stdout=subprocess.PIPE)
    child2.wait()
    time2=child2.stdout.read().decode()
    res=[patterns[i],time1,time2]

    # record consumed time
    for i in res:
        res_file.write(str(i)+"|")

res_file.close()
```

测试结果

测试结果如下所示，hyperscan正则匹配的速度远大于regex，平均耗时约是regex的1/54，并且hyperscan支持并发匹配，在海量数据和高并发度匹配的场景下的优势巨大。

pattern	hyperscan	regex
d.cker	0.000210	0.010804
bu.ld	0.000193	0.010662
i..ge	0.000752	0.037291

pattern	hyperscan	regex
ports	0.000372	0.022707
net.ork	0.000678	0.039373
co..and	0.000389	0.038685
us.r	0.000462	0.026092
hos[a-z]name	0.000648	0.022948
cont.iner	0.000339	0.025170
edg[a-z]x-network	0.000651	0.021022