취약코드 패턴을 이용한 취약점 분석

About me

- @LeaveRet
- BoB 3기
  - 교육단계 – pintool을 이용한 취약점 분석 자동화
  - 경연단계 – 프리서버 카운터어택
- 카이스트 SysSec 인턴 (2015.06.22~2015.08.20)
- @SEWorks (2015.08.24 ~)

주제 선정이유

- BoB 3기 취약점분석 트랙

주제 선정이유

- BoB 3기 취약점분석 트랙
  - Taint analysis를 이용한 취약점 검출
  - Z3를 이용한 잠재적 취약점 탐색
- Taint propagation analysis
  - 모든 대입연산, 산술연산 등 을 데이터 형에 맞게 다 처리해 줘야함
  - Pintool 특성상 처리코드는 최소화 해야함
- Z3
  - 모든 산술연산 인스트럭션들을 z3에 맞게 수식으로 변환해줘야함

주제 선정이유

- Taint propagation analysis
  - 구현하기 어렵다
- Fuzzing
  - 과연 내 퍼져가 취약점을 찾을 수 있을까
- Symbolic execution
  - ㄷ ㄷ ㄷ ㄷ
- Concolic execution
  - ㄷ ㄷ ㄷ ㄷ ㄷ ㄷ ㄷ ㄷ ㄷ ㄷ ㄷ ㄷ ㄷ ㄷ ㄷ

주제 선정이유

- 왜 사람은 취약점을 잘 찾는데
- 컴퓨터가 알아서 찾게하려면 어려울까

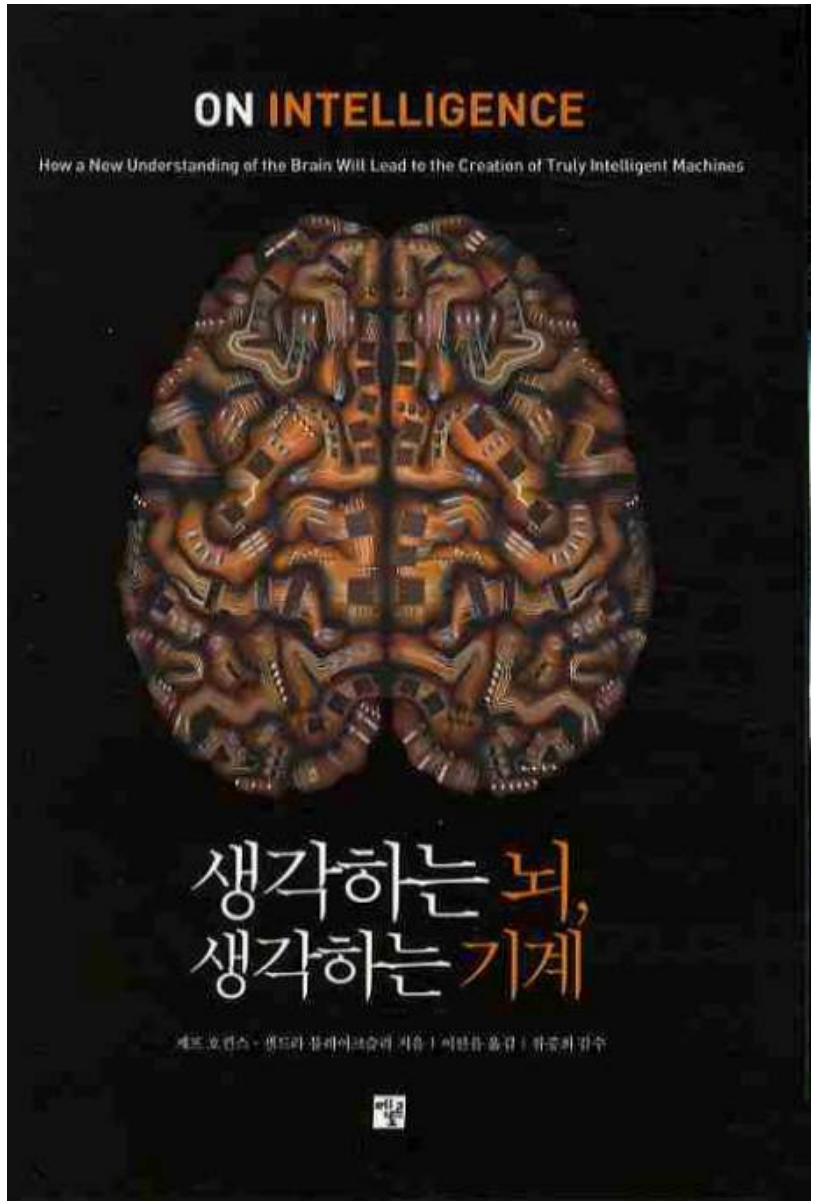
- → 사람은 취약점을 어떻게 찾을까?

첫 생각

- 인공 신경망으로 취약점을 찾아볼까?

- 어떻게 찾지..

뇌 과학 공부...

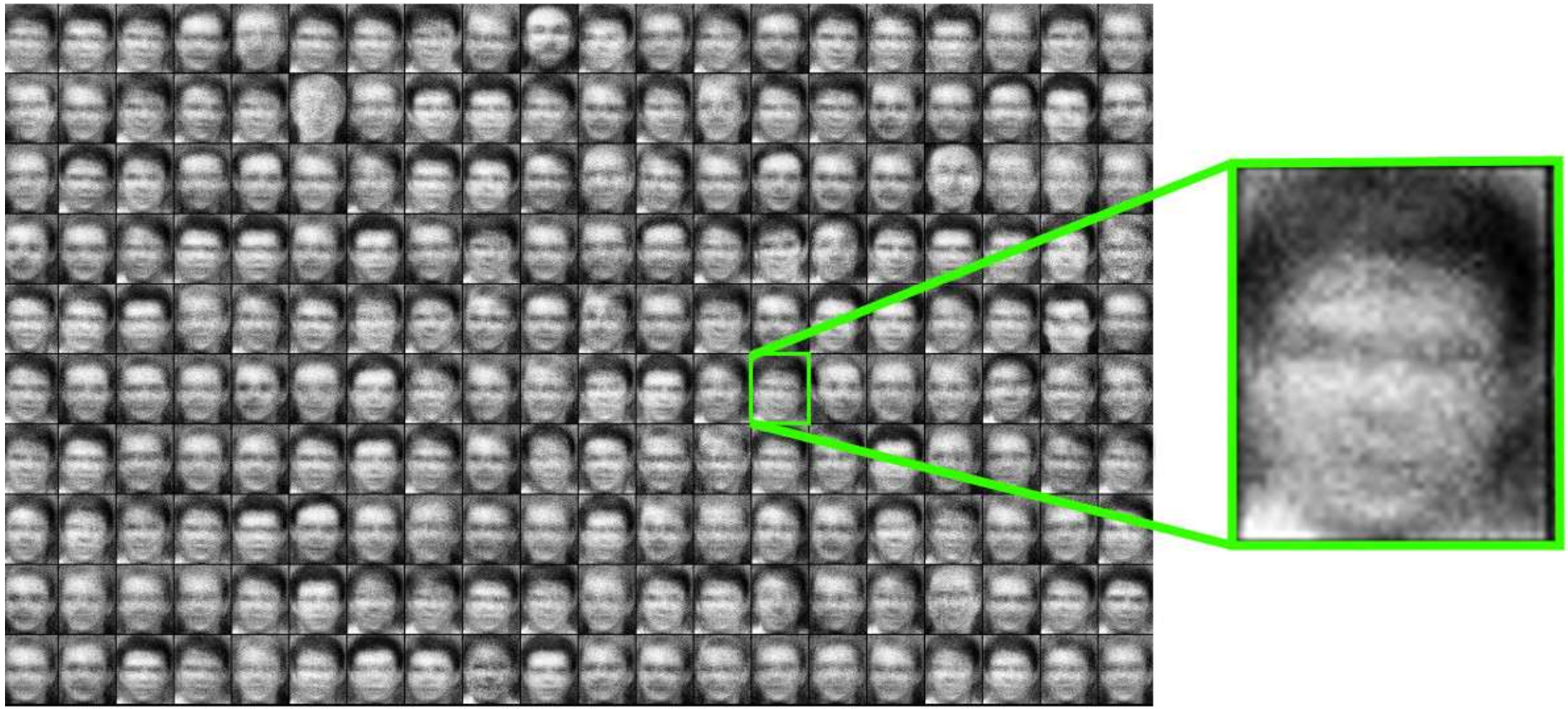

ON INTELLIGENCE
How a New Understanding of the Brain Will Lead to the Creation of Truly Intelligent Machines
생각하는 뇌,
생각하는 기계

사람 뇌의 특징

- 추상화
- 계층화
- .....

사람 뇌의 특징

- 추상화



프로그램의 추상화?

- 바이너리
  - Control flow graph?
  - 라이브러리 함수 사용패턴?
  - Binary visualization?

Control flow graph

- 너무 추상적임

라이브러리 함수 사용패턴

- Strip된 바이너리
- 커스텀 라이브러리

Binary visualization

- BoB 3기 취약점분석 트랙 심준보 멘토님 과제
- 활용할 방법이 있을 듯한데 아직 안해봄

오픈소스!

- 모든 심볼 제공
- 코딩 컨벤션 존재
- 복&붙 코드 많음
- 자세한 취약점 정보 제공
- 유명한 오픈소스 프로젝트
  - 10개~80개정도 cve 존재
- 오픈소스 취약점 발견 시 위험도 높음
  - Ex) libjpeg, ffmpeg, libavi 등

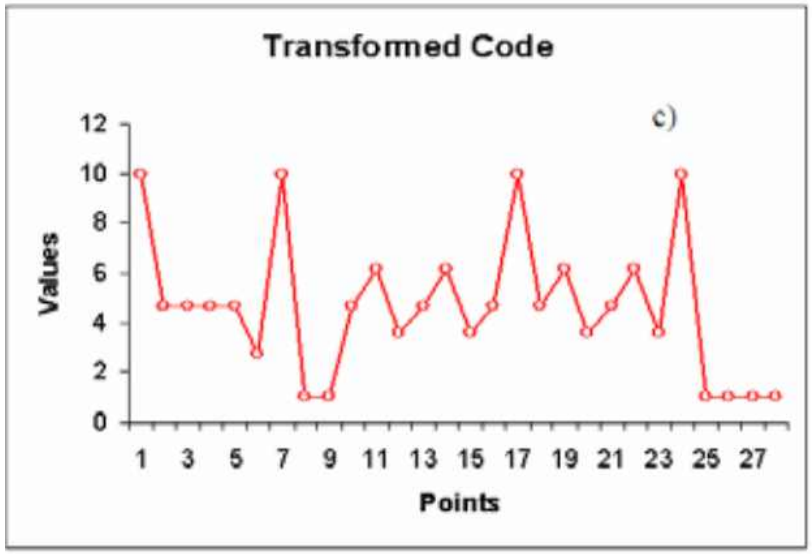
소스코드 패턴 매칭

- 표절 탐지

- Code Similarity on High Level Programs

```
public void Resize(int factor)
{
    Bitmap input = this.input;
    int i = 0; int j = 0;
    Bitmap output = new Bitmap(input.Width * factor, input.Height * factor);

    for (i = 0; i < input.Size.Width; i++)
    {
        for (j = 0; j < input.Size.Height; j++)
        {
            Color pixel = input.GetPixel(i, j);
            for (int x = 0; x < factor; x++)
            {
                for (int y = 0; y < factor; y++)
                    output.SetPixel(i * factor + x, j * factor + y, pixel);
            }
        }
    }
}
```

a)



- http://arxiv.org/ftp/arxiv/papers/0710/0710.5547.pdf

소스코드 패턴 매칭

- 소스코드 추상화
  - 토큰화 if(i<10) if(b == 5)
  - 그래프화
- 추상화된 소스코드간의 비교

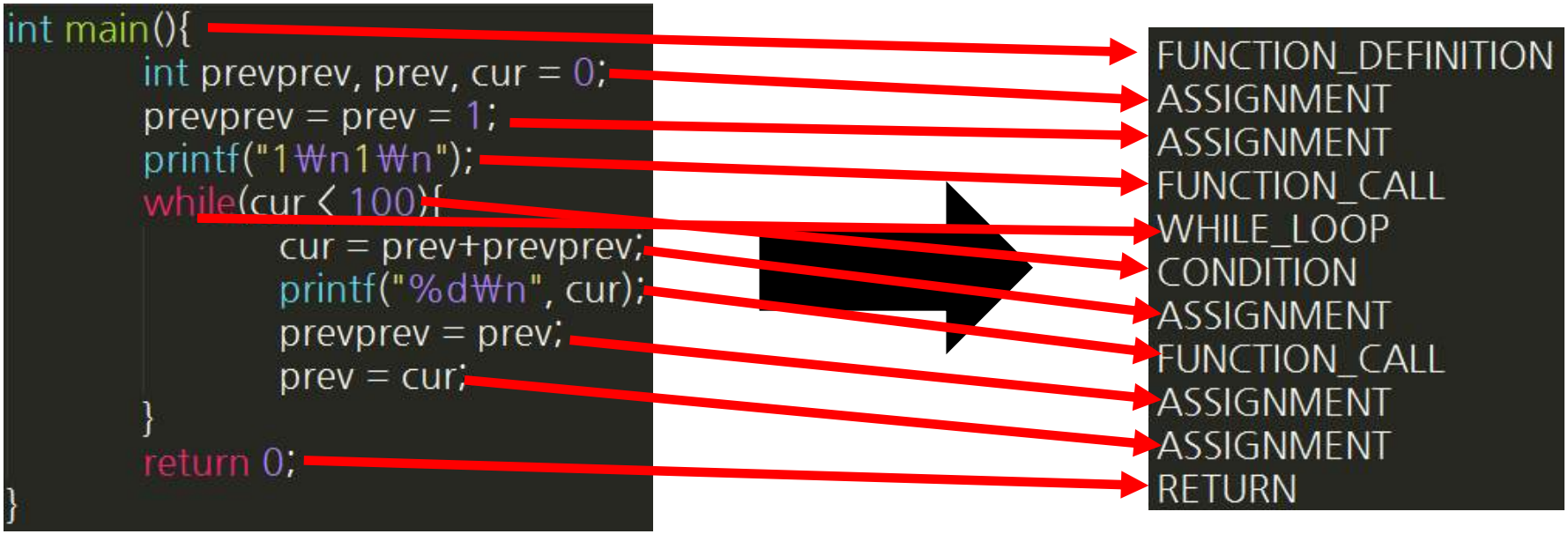
  → 그래프 비교 알고리즘
- Dynamic Time Warping 알고리즘 사용함

4. 기존에 음성데이터의 비교나 또는 어떤 모션 비교에 이 알고리즘을 많이 사용하여 비교하게 됩니다.

이 알고리즘에 가장 큰 특징은 다른 유사도 패턴 알고리즘보다 구현하기 쉽다는 장점이 있어서 패턴 알고리즘을 공부해보고 싶은 분들은
이 알고리즘을 처음으로 공부해보는것도 좋을거 같습니다.

소스코드 추상화

- 함수내의 모든 코드를 카테고리화

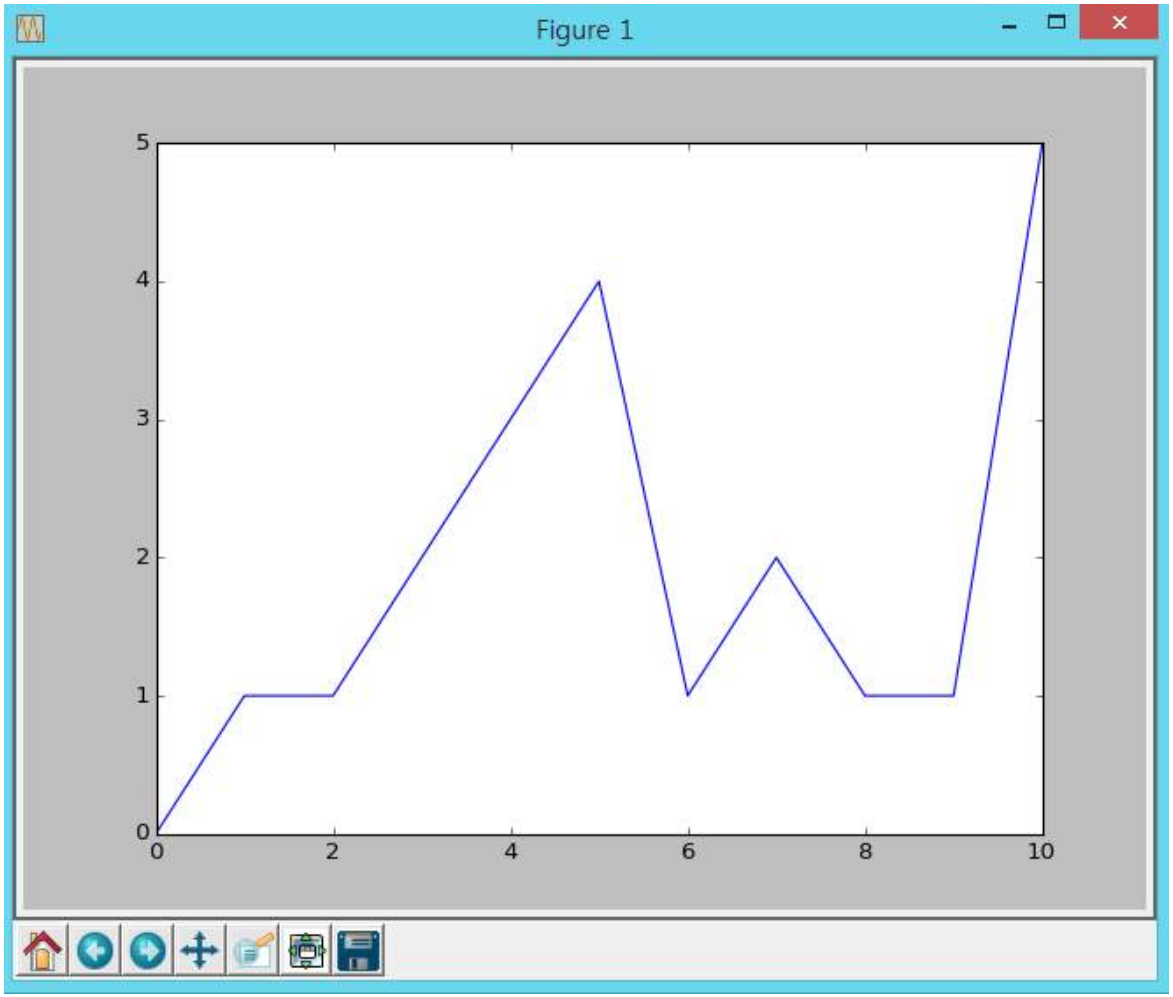


소스코드 추상화

- 각 토큰별로 임의의 숫자 배정(비슷한 토큰이 비슷한 숫자일수록 좋음)

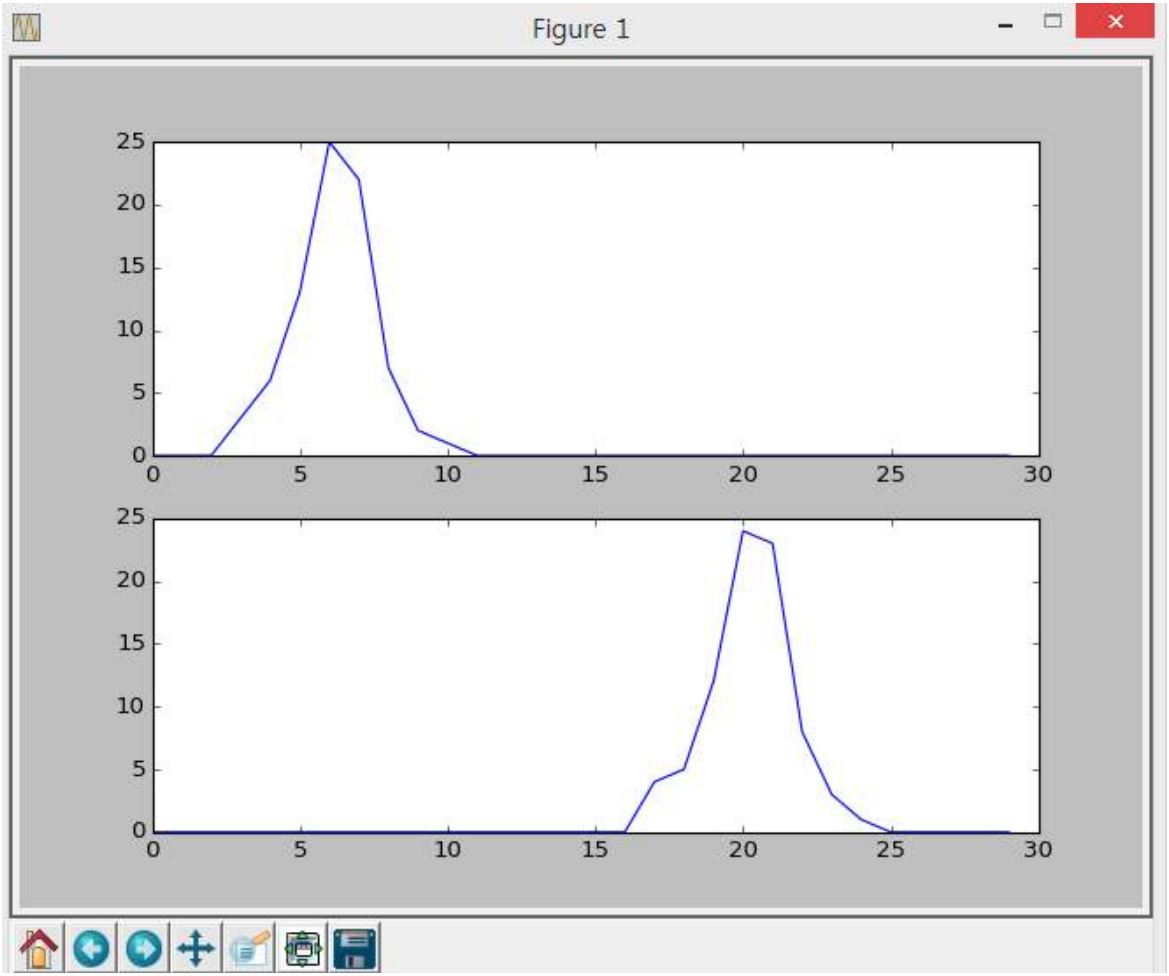
```
tokens = ["FUNCTION_DEFINITION", "ASSIGNMENT", "FUNCTION_CALL", "WHILE_LOOP"
fibonacci = """FUNCTION_DEFINITION
ASSIGNMENT
ASSIGNMENT
FUNCTION_CALL
WHILE_LOOP
CONDITION
ASSIGNMENT
FUNCTION_CALL
ASSIGNMENT
ASSIGNMENT
RETURN""".split("₩n")

fib_array = []
for i in fibonacci:
    fib_array.append(tokens.index(i))

import matplotlib.pyplot as plt
plt.plot(fib_array)
plt.show()
```

소스코드 추상화

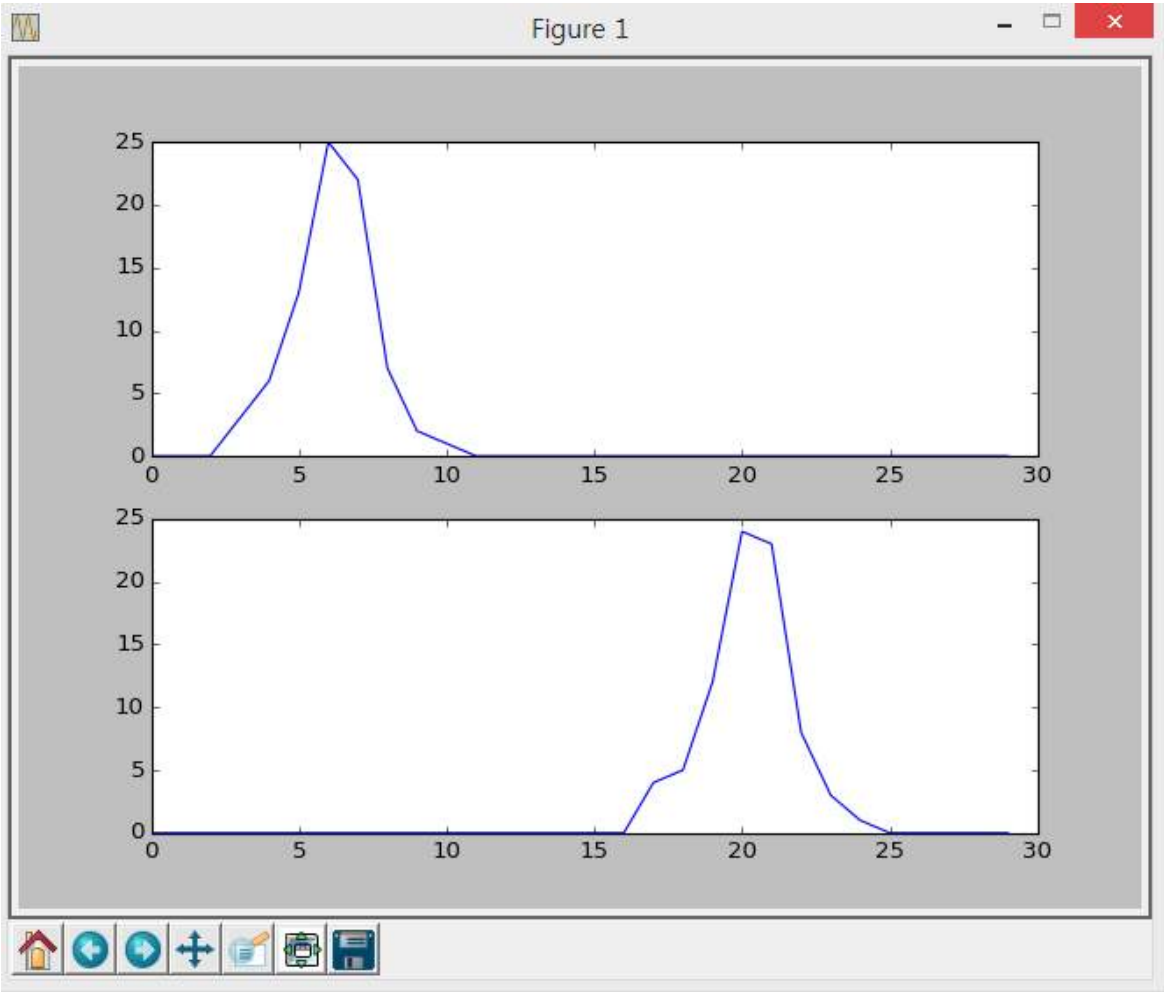


추상화된 소스코드 간의 비교

- Dynamic Time Warping

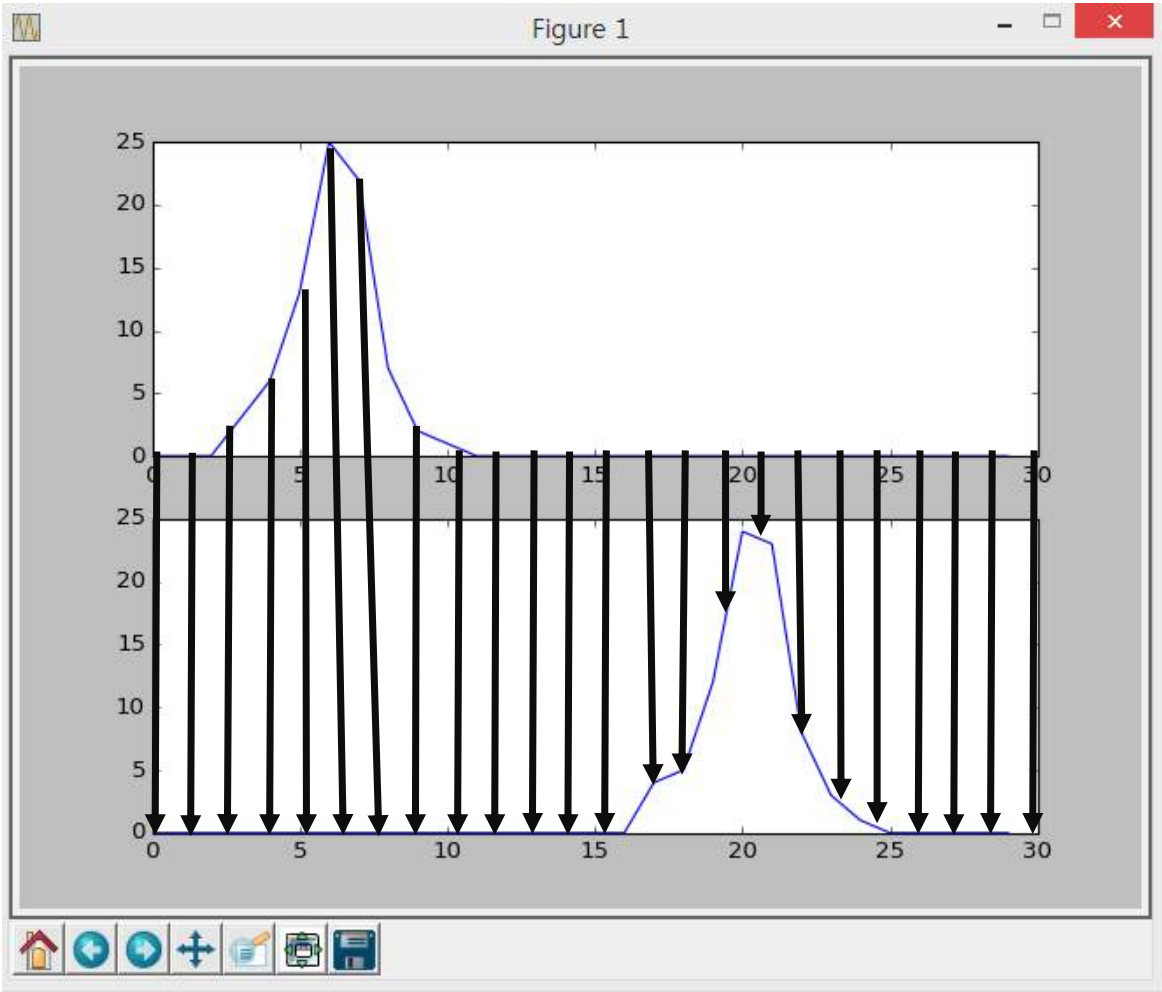


- 0 0 0 3 6 13 25 22 7 2 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
- 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 4 5 12 24 23 8 3 1 0 0 0 0 0



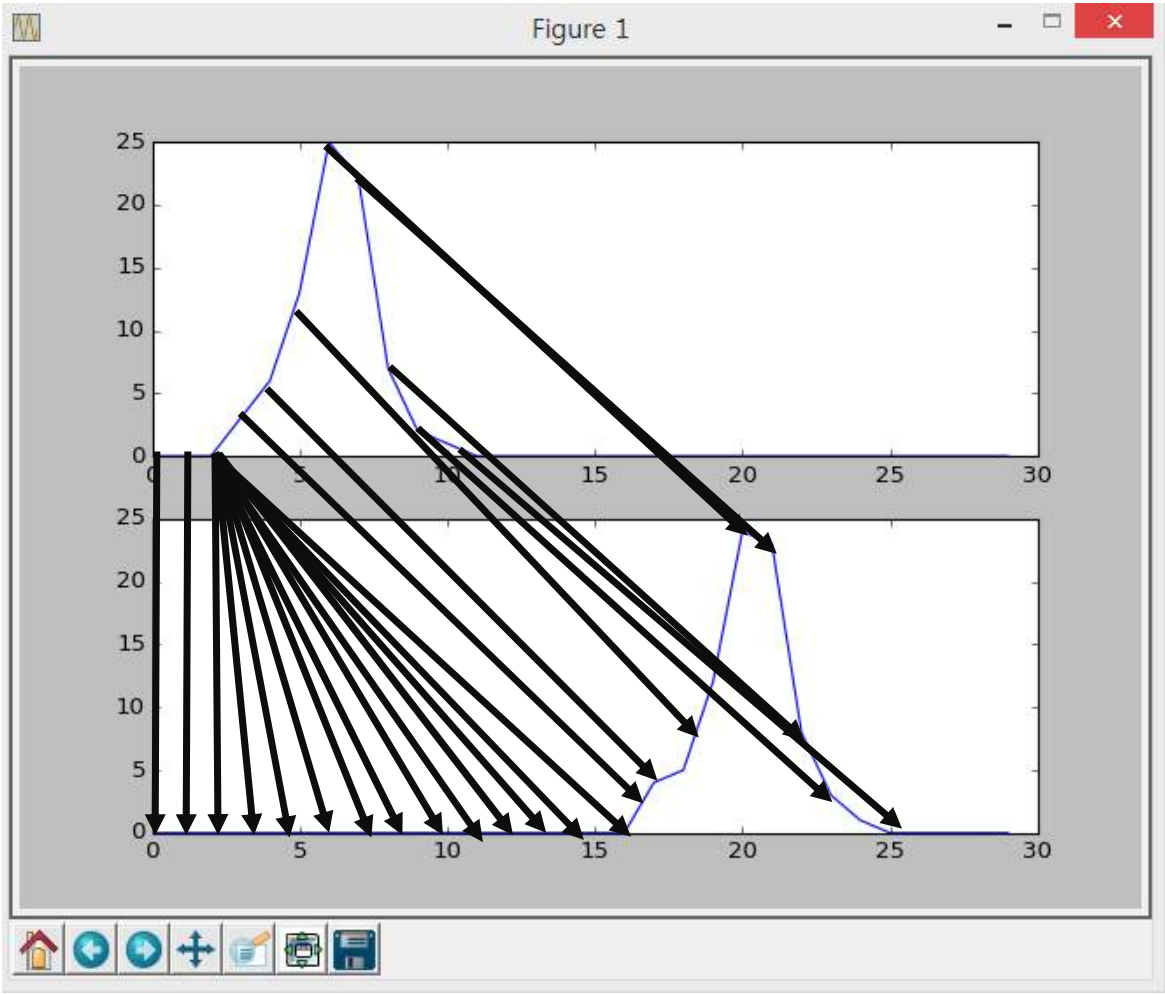
- 0 0 0 3 6 13 25 22 7 2 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
- 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 4 5 12 24 23 8 3 1 0 0 0 0 0

- Euclidean Norm
- = sqrt(3**2+6**2+13**2) ....
- = sqrt(2741)
- = 52.3245.....



- 0 0 0 3 6 13 25 22 7 2 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
- 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 4 5 12 24 23 8 3 1 0 0 0 0 0

- Dynamic Time Warping
- |3-4|+|6-5|+….+|1-1|
- = 7



- 실제코드의 그래프화

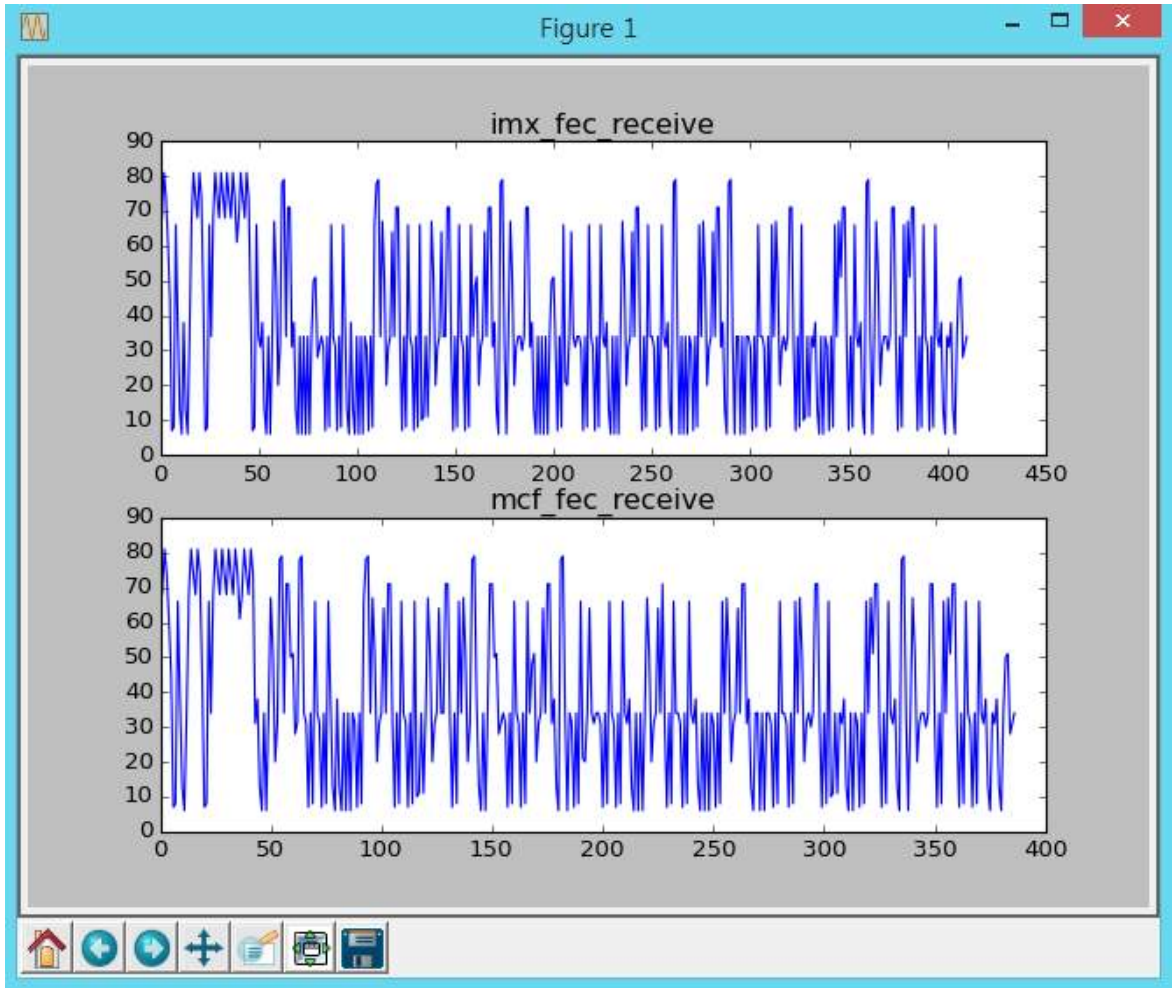
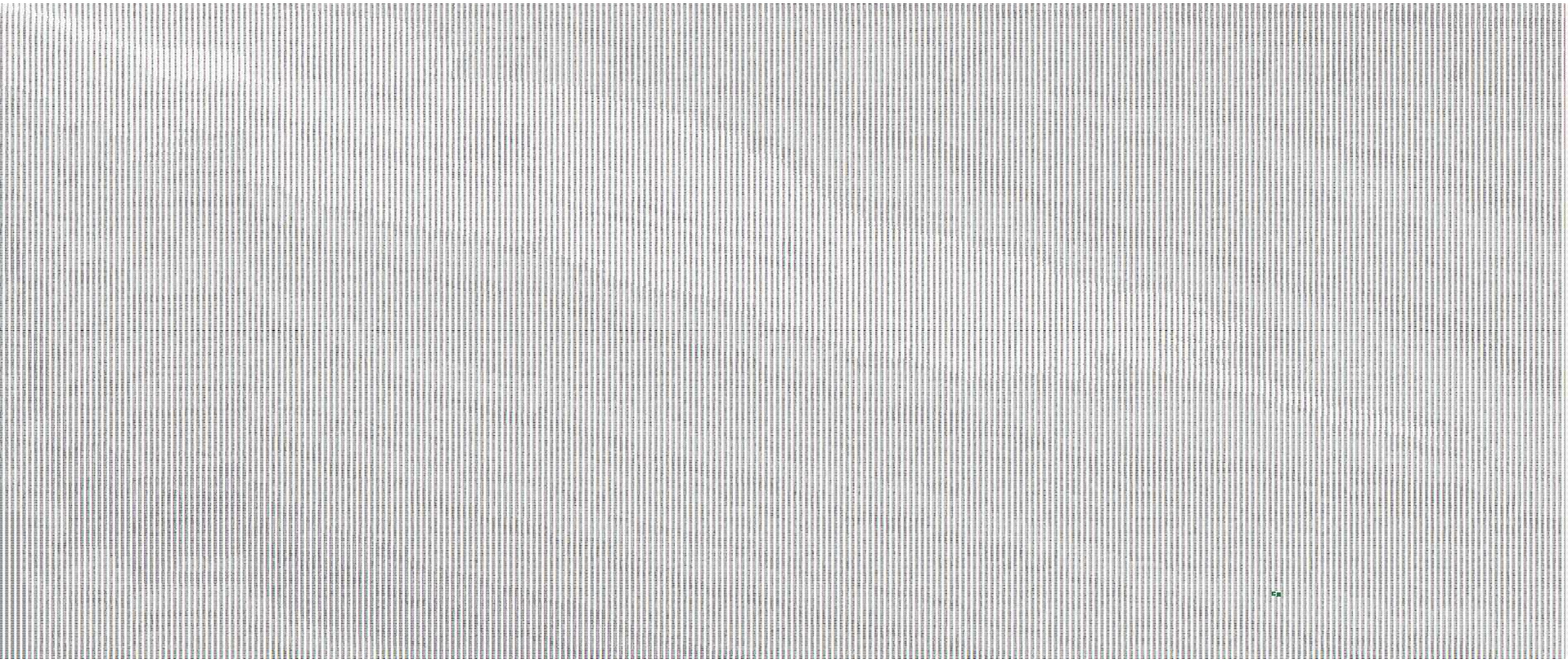
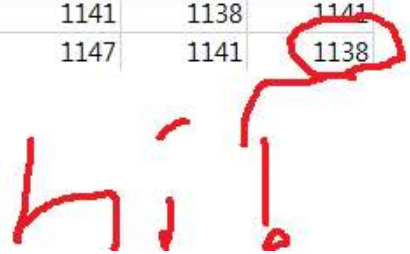


Figure 1
imx_fec_receive
mcf_fec_receive
0
10
20
30
40
50
60
70
80
90
0
50
100
150
200
250
300
350
400
450



1226	1233	1251	1276	1279	1279	1286	1304	1329	1332	1351	1371	1374	1374	1377
1250	1257	1239	1240	1267	1291	1310	1292	1293	1320	1363	1395	1392	1398	1401
1199	1203	1224	1252	1240	1243	1247	1268	1296	1293	1309	1326	1332	1335	1335
1219	1229	1208	1210	1236	1259	1273	1252	1254	1280	1322	1352	1346	1355	1361
1205	1233	1261	1268	1242	1271	1287	1305	1312	1286	1296	1311	1349	1381	1387
1141	1145	1166	1194	1194	1197	1201	1222	1250	1250	1266	1283	1289	1292	1292
1138	1145	1163	1188	1191	1191	1198	1216	1241	1244	1263	1283	1286	1286	1289
1145	1138	1163	1195	1192	1198	1191	1216	1248	1245	1256	1269	1279	1286	1290
1163	1163	1138	1145	1166	1184	1209	1191	1198	1219	1256	1294	1284	1297	1307
1188	1195	1145	1138	1166	1191	1216	1198	1191	1219	1263	1301	1306	1309	1325
1191	1192	1166	1166	1138	1141	1145	1166	1194	1191	1207	1224	1230	1233	1233
1191	1198	1184	1191	1141	1138	1145	1163	1188	1191	1210	1227	1227	1227	1230
1198	1191	1209	1216	1145	1145	1138	1163	1195	1192	1203	1216	1226	1233	1231
1216	1216	1191	1198	1166	1163	1163	1138	1145	1166	1203	1241	1231	1244	1252
1241	1248	1198	1191	1194	1188	1195	1145	1138	1166	1210	1248	1253	1256	1272
1244	1245	1219	1219	1191	1191	1192	1166	1166	1138	1154	1171	1177	1180	1180
1263	1256	1256	1263	1207	1210	1203	1203	1210	1154	1138	1139	1161	1180	1196
1283	1269	1294	1301	1224	1227	1216	1241	1248	1171	1139	1138	1161	1181	1197
1286	1279	1284	1306	1230	1227	1226	1231	1253	1177	1161	1161	1138	1141	1147
1286	1286	1297	1309	1233	1227	1233	1244	1256	1180	1180	1181	1141	1138	1141
1289	1290	1307	1325	1233	1230	1231	1252	1272	1180	1196	1197	1147	1141	1138



돌리는중...

```
[*] Start
[*] walking directory [ D:/study/qemu-2.5.0-rc1/ ]
Elapsed 0 seconds
[*] processing 6431 c files
[1/6431] D:/study/qemu-2.5.0-rc1/accel.c
[2/6431] D:/study/qemu-2.5.0-rc1/aio-posix.c
[3/6431] D:/study/qemu-2.5.0-rc1/aio-win32.c
[4/6431] D:/study/qemu-2.5.0-rc1/arch_init.c
[5/6431] D:/study/qemu-2.5.0-rc1/async.c
[6/6431] D:/study/qemu-2.5.0-rc1/balloon.c
[7/6431] D:/study/qemu-2.5.0-rc1/block.c
[8/6431] D:/study/qemu-2.5.0-rc1/blockdev-nbd.c
[9/6431] D:/study/qemu-2.5.0-rc1/blockdev.c
[10/6431] D:/study/qemu-2.5.0-rc1/blockjob.c
[11/6431] D:/study/qemu-2.5.0-rc1/bootdevice.c
[12/6431] D:/study/qemu-2.5.0-rc1/bt-host.c
[13/6431] D:/study/qemu-2.5.0-rc1/bt-vhci.c
[14/6431] D:/study/qemu-2.5.0-rc1/cpu-exec-common.c
[15/6431] D:/study/qemu-2.5.0-rc1/cpu-exec.c
[16/6431] D:/study/qemu-2.5.0-rc1/cpus.c
[17/6431] D:/study/qemu-2.5.0-rc1/cputlb.c
[18/6431] D:/study/qemu-2.5.0-rc1/device-hotplug.c
[19/6431] D:/study/qemu-2.5.0-rc1/device_tree.c
[20/6431] D:/study/qemu-2.5.0-rc1/disas.c
[21/6431] D:/study/qemu-2.5.0-rc1/dma-helpers.c
[22/6431] D:/study/qemu-2.5.0-rc1/dump.c
[23/6431] D:/study/qemu-2.5.0-rc1/exec.c
[24/6431] D:/study/qemu-2.5.0-rc1/gdbstub.c
```

```
[6420/6431] D:/study/qemu-2.5.0-rc1/util/qemu-progress.c
[6421/6431] D:/study/qemu-2.5.0-rc1/util/qemu-sockets.c
[6422/6431] D:/study/qemu-2.5.0-rc1/util/qemu-thread-posix.c
[6423/6431] D:/study/qemu-2.5.0-rc1/util/qemu-thread-win32.c
[6424/6431] D:/study/qemu-2.5.0-rc1/util/qemu-timer-common.c
[6425/6431] D:/study/qemu-2.5.0-rc1/util/rcu.c
[6426/6431] D:/study/qemu-2.5.0-rc1/util/readline.c
[6427/6431] D:/study/qemu-2.5.0-rc1/util/rfifolock.c
[6428/6431] D:/study/qemu-2.5.0-rc1/util/throttle.c
[6429/6431] D:/study/qemu-2.5.0-rc1/util/timed-average.c
[6430/6431] D:/study/qemu-2.5.0-rc1/util/unicode.c
[6431/6431] D:/study/qemu-2.5.0-rc1/util/uri.c
Elapsed 79 seconds
[*] Tokenizing codes
[Tokenizing function 1/45607] in2_ri2
[Tokenizing function 2/45607] omap_tap_init
[Tokenizing function 3/45607] gen_lswi
[Tokenizing function 4/45607] ath5k_get_rate_pcal_data
[Tokenizing function 5/45607] virtio_balloon_get_features
[Tokenizing function 6/45607] flash_detect_legacy
[Tokenizing function 7/45607] __udelay
[Tokenizing function 8/45607] do_i2c_mm
[Tokenizing function 9/45607] fw_cfg_boot_set
[Tokenizing function 10/45607] ide_transfer_start
```

QEMU에 돌려본 결과

```
Checking similar functions for [ne2000_receive]

[('ne2000_receive', 386), ('flash_erase_2', 2860), ('scc_setup_dma', 2866)]
FUNC_NAME       DTW_DISTANCE    FILE_PATH
ne2000_receive  386     D:/study/qemu-2.5.0-rc1/hw/net/ne2000.c
flash_erase_2   2860    D:/study/qemu-2.5.0-rc1/roms/u-boot/board/amcc/yucca/flash.c
scc_setup_dma   2866    D:/study/qemu-2.5.0-rc1/roms/u-boot/board/micronas/vct/scc.c
intel_hda_xfer  2916    D:/study/qemu-2.5.0-rc1/hw/audio/intel-hda.c
alloc_f 2949    D:/study/qemu-2.5.0-rc1/qemu-io-cmds.c
lan9118_receive 2962    D:/study/qemu-2.5.0-rc1/hw/net/lan9118.c
mcf_fec_receive 2972    D:/study/qemu-2.5.0-rc1/hw/net/mcf_fec.c
virtio_blk_handle_scsi_req      2988    D:/study/qemu-2.5.0-rc1/hw/block/virtio-blk.c
virtio_net_query_rxfilter       2997    D:/study/qemu-2.5.0-rc1/hw/net/virtio-net.c
ensure_visible  3006    D:/study/qemu-2.5.0-rc1/roms/u-boot/board/netphone/phone_console.c
Checking similar functions for [mcf_fec_receive]

[('mcf_fec_receive', 0), ('imx_fec_receive', 1138), ('onenand_block_test', 2257)]
FUNC_NAME       DTW_DISTANCE    FILE_PATH
mcf_fec_receive 0       D:/study/qemu-2.5.0-rc1/hw/net/mcf_fec.c
imx_fec_receive 1138    D:/study/qemu-2.5.0-rc1/hw/net/imx_fec.c
onenand_block_test      2257    D:/study/qemu-2.5.0-rc1/roms/u-boot/common/cmd_onenand.c
vscsi_send_rsp  2302    D:/study/qemu-2.5.0-rc1/hw/scsi/spapr_vscsi.c
kvm_mips_restore_count  2329    D:/study/qemu-2.5.0-rc1/target-mips/kvm.c
ubifs_read_nnode        2333    D:/study/qemu-2.5.0-rc1/roms/u-boot/fs/ubifs/lpt.c
do_verify       2385    D:/study/qemu-2.5.0-rc1/roms/u-boot/drivers/usb/gadget/f_mass_storage.c
sopreprbuf      2391    D:/study/qemu-2.5.0-rc1/slirp/socket.c
yaffs_ecc_calc  2409    D:/study/qemu-2.5.0-rc1/roms/u-boot/fs/yaffs2/yaffs_ecc.c
search_bbt      2426    D:/study/qemu-2.5.0-rc1/roms/u-boot/drivers/mtd/nand/nand_bbt.c
```

mcf_fec_receive

- 컴파일 시 미포함
- ne2000_receive함수가 취약해 보임(실제로 취약함)

결론

- 패턴 매칭도 쓸만하다
- 보완하고 발전시키면 인공신경망에 적용할 수 있다
  → 좀더 재밌는 연구거리가 많다
- 다른 분야에서 뭘 하고 있는지 보면 재밌는게 많다

추가 연구 계획

- 컴파일된 바이너리
- Jaccard similarity coefficient algorithm
- Hausdorff similarity
- 인공신경망에 적용

참고자료 - 감사합니다

- http://www.covert.io/research-papers/security/Vulnerability%20Extrapolation%20-%20Assisted%20Discovery%20of%20Vulnerabilities%20using%20Machine%20Learning.pdf
- http://www.prosec-project.org/docs/2012-acsac.pdf
- http://arxiv.org/ftp/arxiv/papers/0710/0710.5547.pdf
- http://blog.secmem.org/593
- http://www.smallake.kr/?p=17918
- http://jeremykun.com/2012/07/25/dynamic-time-warping/