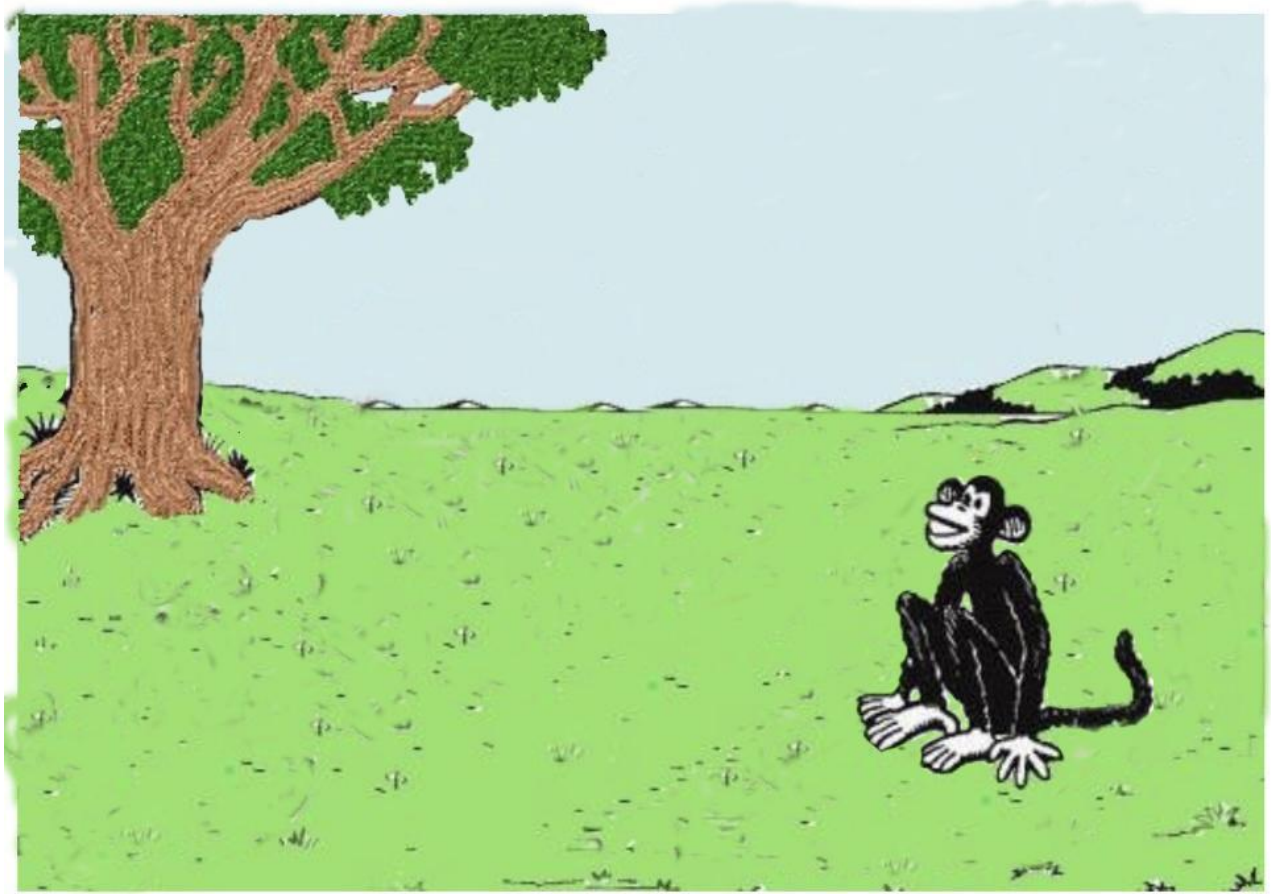
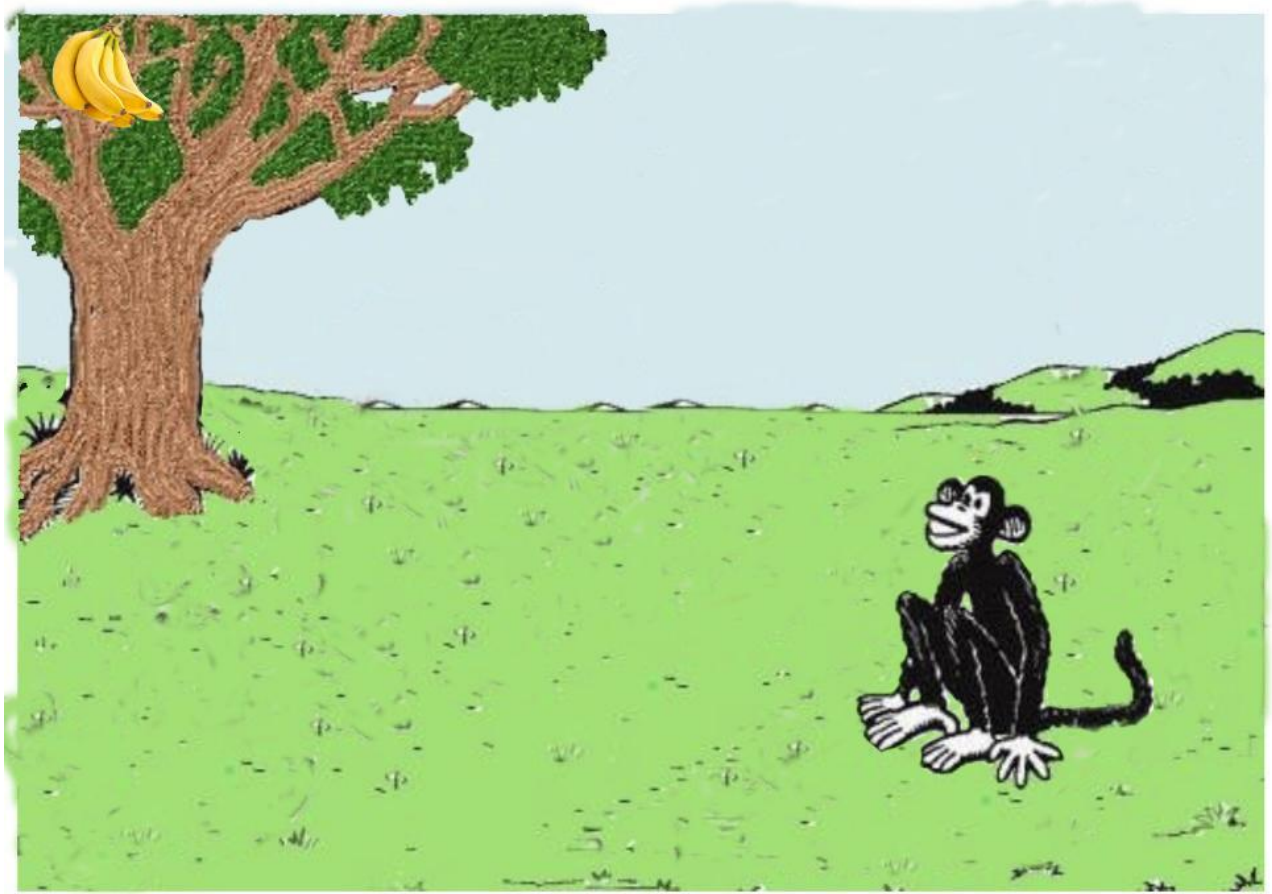
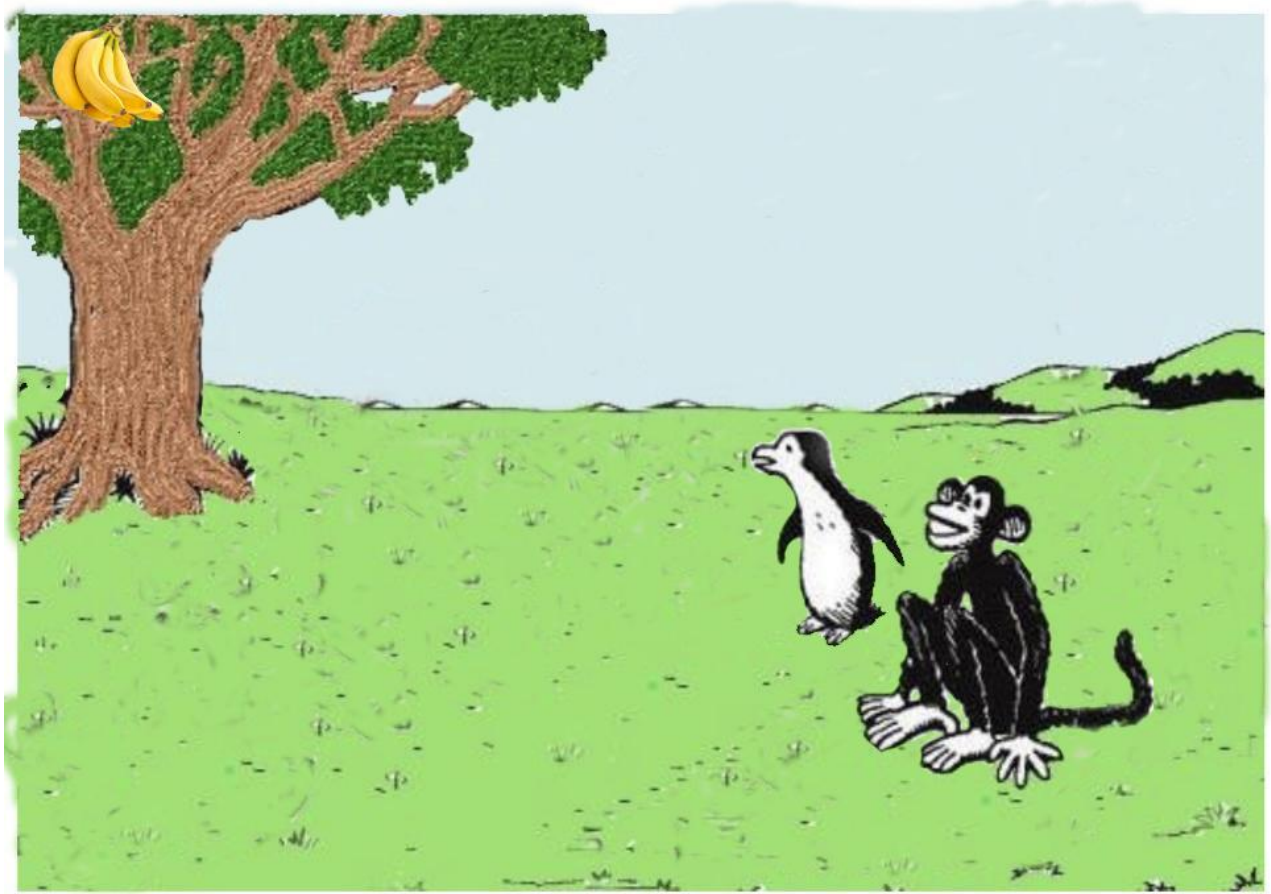
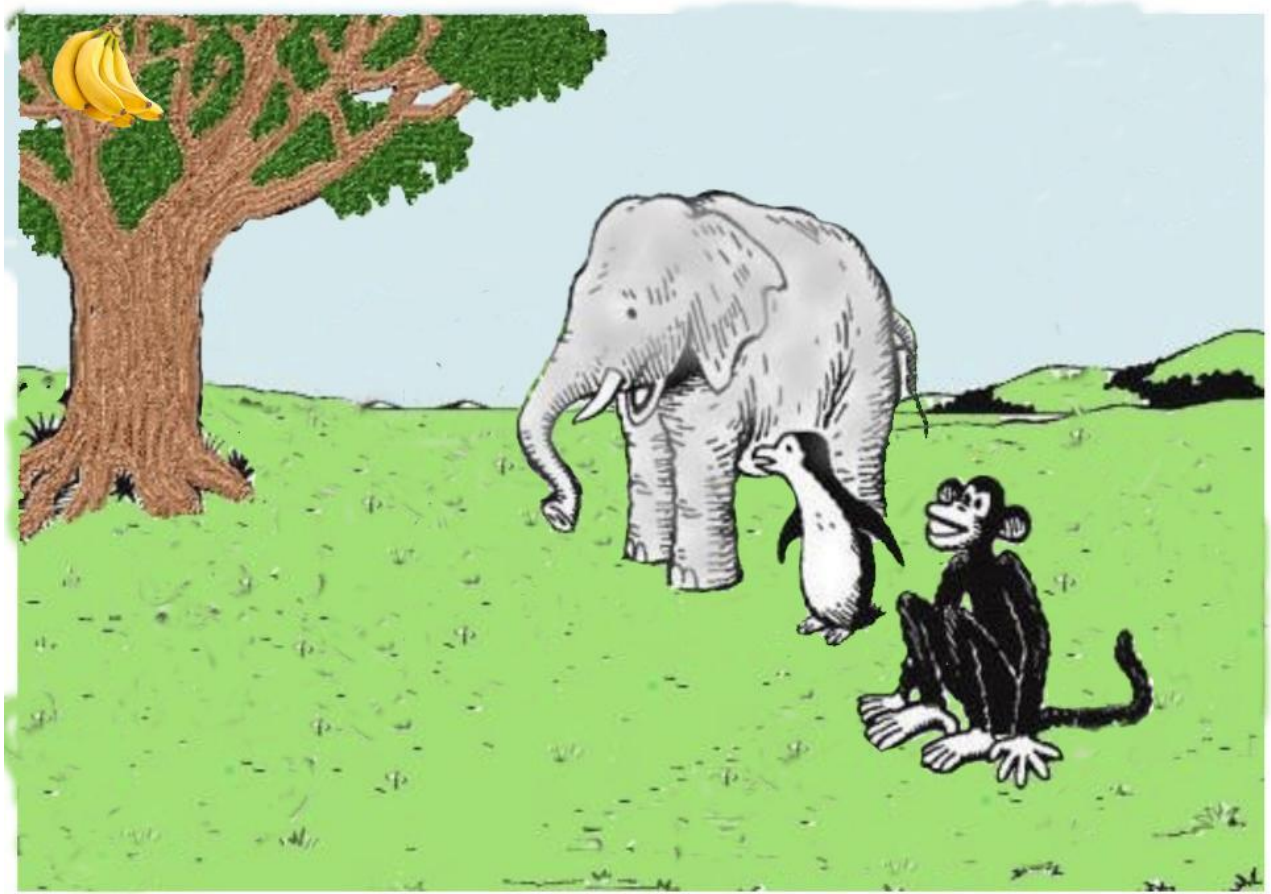
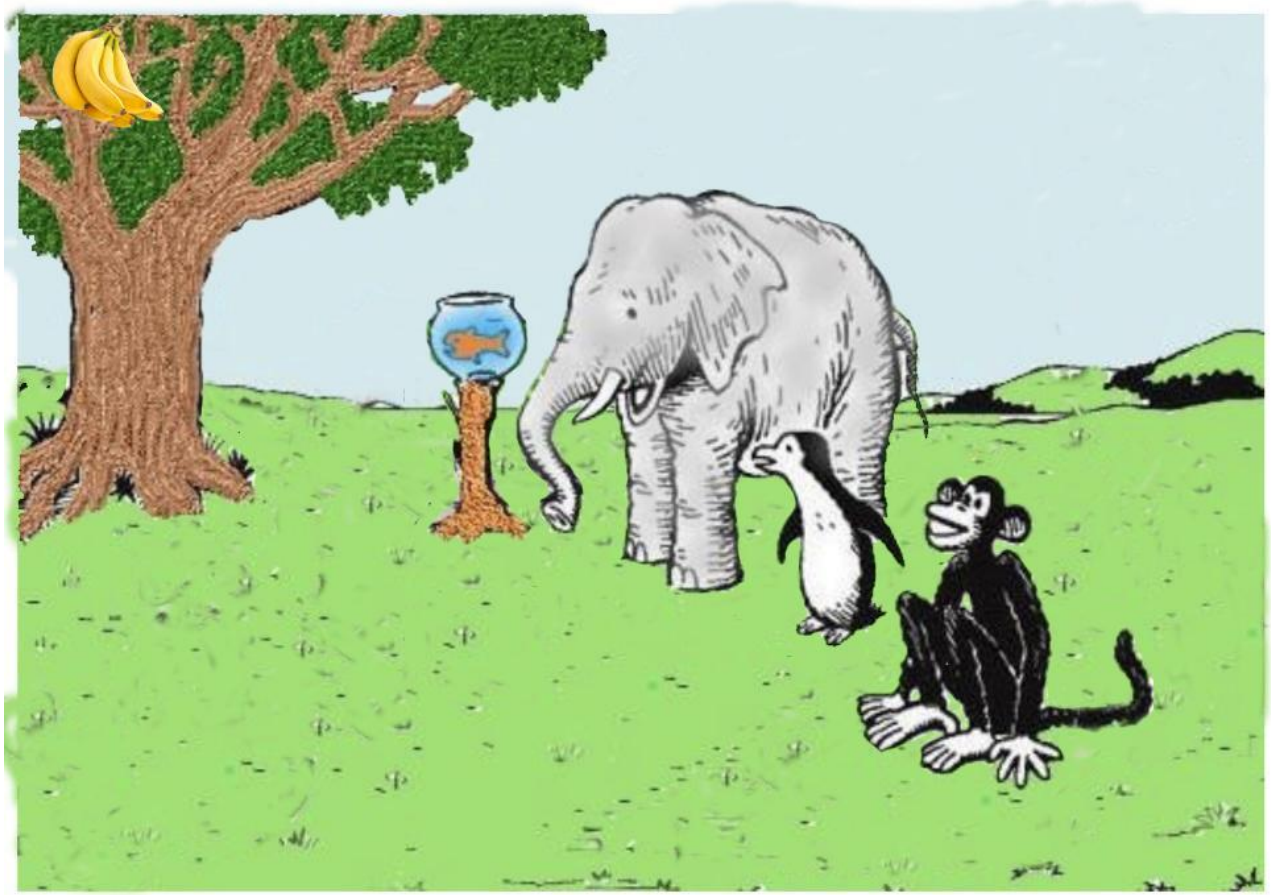










Decompiler benchmarking

- Inleiding compilatie/decompilatie/benchmarken (Jaap)
- Het probleem van benchmarken en onze aanpak (Kesava)
- Implementatie/resultaten (Reijer)

Decompilatie

```
#include <stdio.h>
void wachtopchar(){
   getchar();
}
int main(){
   // melden dat we er zijn!
   printf("Hello, world!\n");
   // even wachten...
   wachtopchar();
   // en leeftijd teruggeven
   return 47;
}
```

```
F3 0F 1E FA 48 83 EC 08 48 8B 05
D9 2F 00 00 48 85 C0 74 02 FF D0
48 83 C4 08 C3 00 00 00 00 00 FF
35 E2 2F 00 00 FF 25 E4 2F 00 00
0F 1F 40 00 FF 25 E2 2F 00 00 68
00 00 00 00 E9 E0 FF FF FF FF 25
B2 2F 00 00 66 90 00 00 00 00 00
00 00 00 F3 0F 1E FA 31 ED 49 89
D1 5E 48 89 E2 48 83 E4 F0 50 54
45 31 C0 31 C9 48 8D 3D D1 00 00
00 FF 15 63 2F 00 00 F4 66 2E 0F
1F 84 00 00 00 00 00 48 8D 3D A9
2F 00 00 48 8D 05 A2 2F 00 00 48
39 F8 74 15 48 ...
```

Decompilatie

```
#include <stdio.h>
void wachtopchar(){
   getchar();
}
int main(){
   // melden dat we er zijn!
   printf("Hello, world!\n");
   // even wachten...
   wachtopchar();
   // en leeftijd in dagen
   return 47*365+12;
}
```

```
F3 0F 1E FA 48 83 EC 08 48 8B 05
D9 2F 00 00 48 85 C0 74 02 FF D0
48 83 C4 08 C3 00 00 00 00 00 FF
35 E2 2F 00 00 FF 25 E4 2F 00 00
0F 1F 40 00 FF 25 E2 2F 00 00 68
00 00 00 00 E9 E0 FF FF FF FF 25
B2 2F 00 00 66 90 00 00 00 00 00
00 00 00 F3 0F 1E FA 31 ED 49 89
D1 5E 48 89 E2 48 83 E4 F0 50 54
45 31 C0 31 C9 48 8D 3D D1 00 00
00 FF 15 63 2F 00 00 F4 66 2E 0F
1F 84 00 00 00 00 00 48 8D 3D A9
2F 00 00 48 8D 05 A2 2F 00 00 48
39 F8 74 15 48 ...
```

1 What 2 Why 3 Issues

Decompilatie

```
#include <stdio.h>
void wachtopchar(){
   getchar();
}
int main(){
   // melden dat we er zijn!
   printf("Hello, world!\n");
   // even wachten...
   wachtopchar();
   // en leeftijd in dagen
   return 47*365+12;
}
```

```
F3 0F 1E FA 48 83 EC 08 48 8B 05
D9 2F 00 00 48 85 C0 74 02 FF D0
48 83 C4 08 C3 00 00 00 00 00 FF
35 E2 2F 00 00 [illegible] E4 2F 00 00
0F 1F 40 00 [illegible] 25 E[illegible] 2F 00 00 68
00 00 00 00 E9 E0 F[illegible] FF FF FF 25
B2 2F 00 00 66 90 [illegible] 00 00 00 00
00 00 00 F3 0F 1[illegible] FA 31 ED 49 89
D1 5E 48 89 E2 48 83 E4 F0 50 54
45 31 C0 31 C9 4[illegible] 8D 3D D1 00 00
00 FF 15 63 2F 00 00 F4 66 2E 0F
1F 84 00 00 00 00 00 48 8D 3D A9
2F 00 00 48 8D 05 A2 2F 00 00 48
39 F8 74 15 48 ...
```

1 What 2 Why 3 Issues

Decompilatie

```
#include <stdio.h>
void wachtopchar(){
   getchar();
}
int main(){
   // melden dat we er zijn!
   printf("Hello, world!\n");
   // even wachten...
   wachtopchar();
   // en leeftijd in dagen
   return 47*365+12;
}
```

```
2F 6C 69 62
36 34 2F 6C
64 2D 6C 69
6E 75 78 2D
78 38 36 2D
36 34 2E 73
6F 2E 32 00
```

Decompilatie

```
#include <stdio.h>
void wachtopchar(){
   getchar();
}
int main(){
   // melden dat we er zijn!
   printf("Hello, world!\n");
   // even wachten...
   wachtopchar();
   // en leeftijd in dagen
   return 47*365+12;
}
```

```
2F 6C 69 62    /lib
36 34 2F 6C    64/l
64 2D 6C 69    d-li
6E 75 78 2D    nux-
78 38 36 2D    x86-
36 34 2E 73    64.s
6F 2E 32 00    o.2\0

/lib64/ld-linux-x86-64.so.2
```

1 What 2 Why 3 Issues

Decompilatie

```
#include <stdio.h>
void wachtopchar(){
   getchar();
}
int main(){
   // melden dat we er zijn!
   printf("Hello, world!\n");
   // even wachten...
   wachtopchar();
   // en leeftijd in dagen
   return 47*365+12;
}
```

```
F3 0F 1E FA 48 83 EC 08 48 8B 05
D9 2F 00 00 48 85 C0 74 02 FF D0
48 83 C4 08 C3 00 00 00 00 00 FF
35 E2 2F 00 00 FF 25 E4 2F 00 00
0F 1F 40 00 FF 25 E2 2F 00 00 68
00 00 00 00 E9 E0 FF FF FF FF 25
B2 2F 00 00 66 90 00 00 00 00 00
00 00 00 F3 0F 1E FA 31 ED 49 89
D1 5E 48 89 E2 48 83 E4 F0 50 54
45 31 C0 31 C9 48 8D 3D D1 00 00
00 FF 15 63 2F 00 00 F4 66 2E 0F
1F 84 00 00 00 00 00 48 8D 3D A9
2F 00 00 48 8D 05 A2 2F 00 00 48
39 F8 74 15 48 ...
```

1 What 2 Why 3 Issues

Decompilatie

```
void functie1(){
   functie2();
}
int main(){

   functie4("Hello, world!\n");

   functie1();

   return 17532;
}
```

```
F3 0F 1E FA 48 83 EC 08 48 8B 05
D9 2F 00 00 48 85 C0 74 02 FF D0
48 83 C4 08 C3 00 00 00 00 00 FF
35 E2 2F 00 00 FF 25 E4 2F 00 00
0F 1F 40 00 FF 25 E2 2F 00 00 68
00 00 00 00 E9 E0 FF FF FF FF 25
B2 2F 00 00 66 90 00 00 00 00 00
00 00 00 F3 0F 1E FA 31 ED 49 89
D1 5E 48 89 E2 48 83 E4 F0 50 54
45 31 C0 31 C9 48 8D 3D D1 00 00
00 FF 15 63 2F 00 00 F4 66 2E 0F
1F 84 00 00 00 00 00 48 8D 3D A9
2F 00 00 48 8D 05 A2 2F 00 00 48
39 F8 74 15 48 ...
```

1 What 2 Why 3 Issues

Decompilatie

```
#include <stdio.h>
void wachtopchar(){
   getchar();
}
int main(){
   // melden dat we er zijn!
   printf("Hello, world!\n");
   // even wachten...
   wachtopchar();
   // en leeftijd in dagen
   return 47*365+12;
}
```

```
F3 0F 1E FA 48 83 EC 08 48 8B 05
D9 2F 00 00 48 85 C0 74 02 FF D0
48 83 C4 08 C3 00 00 00 00 00 FF
35 E2 2F 00 00 FF 25 E4 2F 00 00
0F 1F 40 00 FF 25 E2 2F 00 00 68
00 00 00 00 E9 E0 FF FF FF FF 25
B2 2F 00 00 66 90 00 00 00 00 00
00 00 00 F3 0F 1E FA 31 ED 49 89
D1 5E 48 89 E2 48 83 E4 F0 50 54
45 31 C0 31 C9 48 8D 3D D1 00 00
00 FF 15 63 2F 00 00 F4 66 2E 0F
1F 84 00 00 00 00 00 48 8D 3D A9
2F 00 00 48 8D 05 A2 2F 00 00 48
39 F8 74 15 48 ...
```

1 What 2 Why 3 Issues

Decompilatie

```
#include <stdio.h>
void wachtopchar(){
   getchar();
}
int main(){
   // melden dat we er zijn!
   printf("Hello, world!\n");
   // even wachten...
   wachtopchar();
   // en leeftijd in dagen
   return 47*365+12;
}
```

```
F3 0F 1E FA 48 83 EC 08 48 8B 05
D9 2F 00 00 48 85 C0 74 02 FF D0
48 83 C4 08 C3 00 00 00 00 00 FF
35 E2 2F 00 00 FF 25 E4 2F 00 00
0F 1F 40 00 FF 25 E2 2F 00 00 68
00 00 00 00 E9 E0 FF FF FF FF 25
B2 2F 00 00 66 90 00 00 00 00 00
00 00 00 F3 0F 1E FA 31 ED 49 89
D1 5E 48 89 E2 48 83 E4 F0 50 54
45 31 C0 31 C9 48 8D 3D D1 00 00
00 FF 15 63 2F 00 00 F4 66 2E 0F
1F 84 00 00 00 00 00 48 8D 3D A9
2F 00 00 48 8D 05 A2 2F 00 00 48
39 F8 74 15 48 ...
```

```
main = print "hello, world"
```

1 What 2 Why 3 Issues

Decompilatie

```
#include <stdio.h>
void wachtopchar(){
   getchar();
}
int main(){
   // melden dat we er zijn!
   printf("Hello, world!\n");
   // even wachten...
   wachtopchar();
   // en leeftijd in dagen
   return 47*365+12;
}
```

```
F3 0F 1E FA 48 83 EC 08 48 8B 05
D9 2F 00 00 48 85 C0 74 02 FF D0
48 83 C4 08 C3 00 00 00 00 00 FF
35 E2 2F 00 00 FF 25 E4 2F 00 00
0F 1F 40 00 FF 25 E2 2F 00 00 68
00 00 00 00 E9 E0 FF FF FF FF 25
B2 2F 00 00 66 90 00 00 00 00 00
00 00 00 F3 0F 1E FA 31 ED 49 89
D1 5E 48 89 E2 48 83 E4 F0 50 54
45 31 C0 31 C9 48 8D 3D D1 00 00
00 FF 15 63 2F 00 00 F4 66 2E 0F
1F 84 00 00 00 00 00 48 8D 3D A9
2F 00 00 48 8D 05 A2 2F 00 00 48
39 F8 74 15 48 ...
```

C

source: <1kb, binary: 16kb

decompiler output: 3kb, 124 regels

```
main = print "hello, world"
```

Haskell

source: <1kb, binary: 921kb (58×)

decompiler output: 1.536kb (512×), 42.618 regels (343×)

1 What 2 Why 3 Issues

Decompilatie

```c
#include <stdio.h>

int main(){
   for (int a=0;a<5;a++){
      printf("Hello, world!\n");
   }
   return 0;
}
```

```c
#include <stdio.h>

int main(){
   printf("Hello, world!\n");
   printf("Hello, world!\n");
   printf("Hello, world!\n");
   printf("Hello, world!\n");
   printf("Hello, world!\n");
   return 0;
}
```

Decompilatie

```
#include <stdio.h>

int main(){
   for (int x=0;x<100;x++){
      for (int y=0;y<100;y++){
         printf("%d %d\n",x,y);
         if (getchar()==-1){ goto end; }
      }
   }

end:
   return 45;
}
```

Decompilatie

```
          ...
          mov  dword ptr [rbp - 8], 0      # x=0
.LBB0_1:  cmp  dword ptr [rbp - 8], 100    # while (x<100) {
          jge  .LBB0_10
          mov  dword ptr [rbp - 12], 0     #    y=0
.LBB0_3:  cmp  dword ptr [rbp - 12], 100   #    while (y<100) {
          jge  .LBB0_8
           printf();
          call  getchar@PLT                #    if (getchar()==-1) { goto end; }
          cmp  eax, -1
          jne  .LBB0_6
          jmp  .LBB0_11
.LBB0_6:  jmp  .LBB0_7                     #    //compleet nutteloze jump!
.LBB0_7:  mov  eax, dword ptr [rbp - 12]   #    y++
          add  eax, 1
          mov  dword ptr [rbp - 12], eax
          jmp  .LBB0_3                     #    }
.LBB0_8:  jmp  .LBB0_9                     #   //compleet nutteloze jump
.LBB0_9:  mov  eax, dword ptr [rbp - 8]    # x++
          add  eax, 1
          mov  dword ptr [rbp - 8], eax
          jmp  .LBB0_1                     # }
.LBB0_10: jmp  .LBB0_11
.LBB0_11: mov  eax, 45                     # return 45
```

1 What 2 Why 3 Issues

Decompilatie

```
          ...
          mov  dword ptr [rbp - 8], 0       # x=0
.LBB0_1:  cmp  dword ptr [rbp - 8], 100     # while (x<100) {
          jge  .LBB0_10
          mov  dword ptr [rbp - 12], 0      #    y=0
.LBB0_3:  cmp  dword ptr [rbp - 12], 100    #    while (y<100) {
          jge  .LBB0_8
           printf();
          call  getchar@PLT                 #    if (getchar()==-1) { goto end; }
          cmp  eax, -1
          jne  .LBB0_6
          jmp  .LBB0_11
.LBB0_6:  jmp  .LBB0_7                      #    //compleet nutteloze jump!
.LBB0_7:  mov  eax, dword ptr [rbp - 12]    #    y++
          add  eax, 1
          mov  dword ptr [rbp - 12], eax
          jmp  .LBB0_3                      #    }
.LBB0_8:  jmp  .LBB0_9                      #   //compleet nutteloze jump
.LBB0_9:  mov  eax, dword ptr [rbp - 8]     # x++
          add  eax, 1
          mov  dword ptr [rbp - 8], eax
          jmp  .LBB0_1                      # }
.LBB0_10: jmp  .LBB0_11
.LBB0_11: mov  eax, 45                      # return 45
```

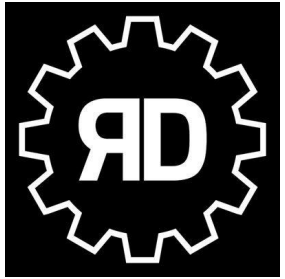
1 What 2 Why 3 Issues

Decompilatie

```
          ...
          mov   dword ptr [rbp - 8], 0       # x=0
.LBB0_1:  cmp   dword ptr [rbp - 8], 100     # while (x<100) {
          jge   .LBB0_10
          mov   dword ptr [rbp - 12], 0      #    y=0
.LBB0_3:  cmp   dword ptr [rbp - 12], 100    #    while (y<100) {
          jge   .LBB0_8
           printf();
          call  getchar@PLT                  #    if (getchar()==-1) { goto end; }
          cmp   eax, -1
          jne   .LBB0_6
          jmp   .LBB0_11  # <--- deze wil je juist als goto (be)houden!
.LBB0_6:  jmp   .LBB0_7                      #    //compleet nutteloze jump!
.LBB0_7:  mov   eax, dword ptr [rbp - 12]    #    y++
          add   eax, 1
          mov   dword ptr [rbp - 12], eax
          jmp   .LBB0_3                      #    }
.LBB0_8:  jmp   .LBB0_9                      #   //compleet nutteloze jump
.LBB0_9:  mov   eax, dword ptr [rbp - 8]     # x++
          add   eax, 1
          mov   dword ptr [rbp - 8], eax
          jmp   .LBB0_1                      # }
.LBB0_10: jmp   .LBB0_11
.LBB0_11: mov   eax, 45                      # return 45
```

1 What 2 Why 3 Issues

Decompilatie

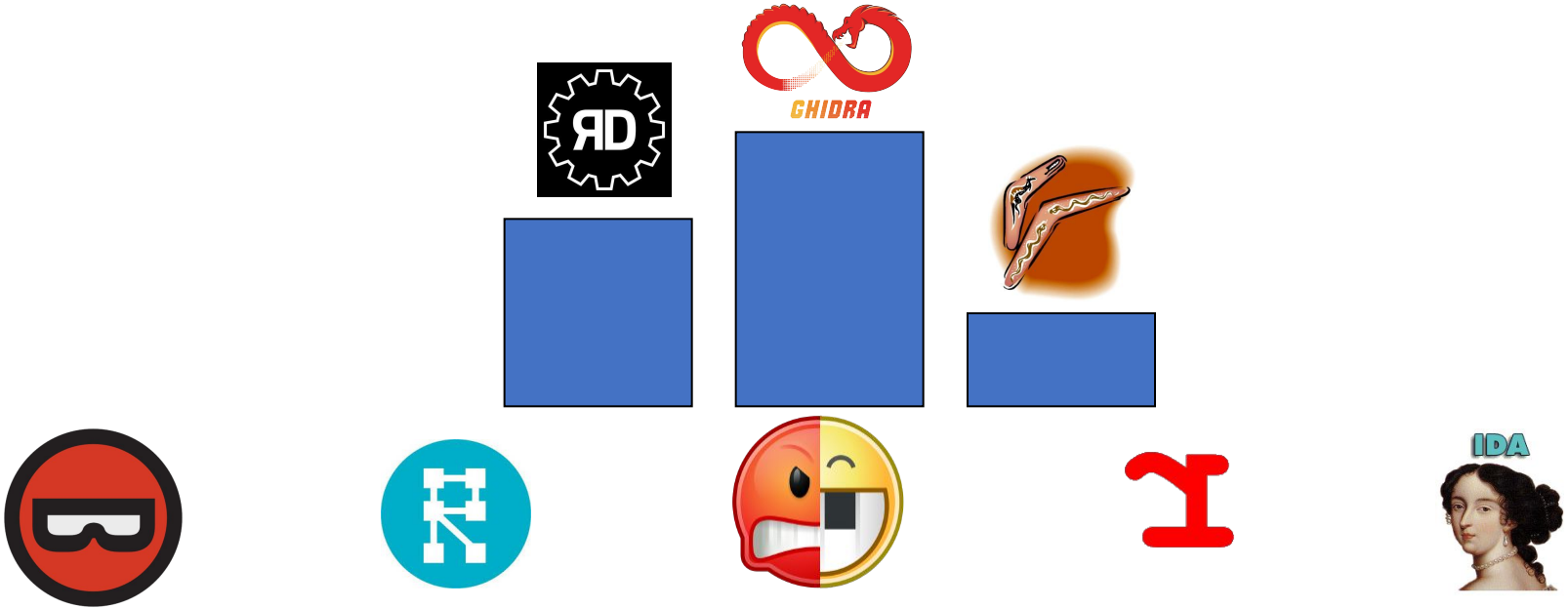


1 What

2 Why

3 Issues

Decompilatie



1 What 2 Why 3 Issues

Ontwerp

Uitgangspunt

een benchmark
die decompilers een score toekent
voor de kwaliteit van de gedecompileerde code

Ontwerp

Blackbox testen

programma $\xrightarrow{\text{decompilatie}}$ code $\xrightarrow{\text{(her)compilatie}}$ programma’

1 Ideeën

2 Deelonderzoeken

Ontwerp

een benchmark
die decompilers een score toekent
voor de kwaliteit van de gedecompileerde code

en die hedendaagse decompilers kan onderscheiden

Ontwerp

C code analyseren

1 Ideeën

2 Deelonderzoeken

Compile and decompile | ready

flags: -O0 -shared

- [x] Use code marker
- [x] Show assembly
- [x] Show ghidra.bat
- [ ] Show console output
- [ ] Show LLVM IR
- [x] Show retdec.bat

search

```c
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 typedef struct {
5     int i;
6     float f;
7 } S;
8
9
10 int global_builtin;
11 S global_struct;
12 int global_array[3];
13 int* global_ptr;
14
15
16 void init_globals() {
17     const int global_ptr_size = 5;
18     global_ptr = malloc(sizeof(int) * global_ptr_size);
19     for (int i=0; i<global_ptr_size; i++)
20         global_ptr[i] = i*i;
21
22     global_builtin = 50;
23
24     global_struct = (S){50, .5};
25
26     global_array[0] = 1;
27     global_array[1] = 2;
28     global_array[2] = 3;
29 }
30
31
32 int main() {
33     init_globals();
34
35     //use local variables
36     int local_builtin;
37     S local_struct;
38     int local_array[3];
39     int* local_ptr;
40
41     const int local_ptr_size = 5;
42     local_ptr = malloc(sizeof(int) * local_ptr_size);
43     for (int i=0; i<local_ptr_size; i++)
44         local_ptr[i] = i*i;
45
46     local_builtin = 50;
47
48     local_struct = (S){50, .5};
49
50     local_array[0] = 1;
51     local_array[1] = 2;
52     local_array[2] = 3;
53
54     printf("8270329");
55     return local_struct.i;
56 }
57
58 /*
```

assembly | ready

search

```asm
67      .long   0x3f000000                      # float 0.5
68      .text
69      .globl  main
70      .p2align  4, 0x90
71 main:                                        # @main
72 .seh_proc main
73 # %bb.0:
74      push rbp
75      .seh_pushreg rbp
76      sub  rsp, 96
77      .seh_stackalloc 96
78      lea  rbp, [rsp + 96]
79      .seh_setframe rbp, 96
80      .seh_endprologue
81      call __main
82      mov  dword ptr [rbp - 4], 0
83      call init_globals
84      mov  dword ptr [rbp - 44], 5
85      mov  ecx, 20
86      call malloc
87      mov  qword ptr [rbp - 40], rax
88      mov  dword ptr [rbp - 48], 0
89 .LBB1_1:                                     # =>This Inner Loop
   Header: Depth=1
90      cmp  dword ptr [rbp - 48], 5
91      jge  .LBB1_4
92 # %bb.2:                                     #   in Loop:
   Header=BB1_1 Depth=1
93      mov  edx, dword ptr [rbp - 48]
94      imul edx, dword ptr [rbp - 48]
95      mov  rax, qword ptr [rbp - 40]
96      movsxd   rcx, dword ptr [rbp - 48]
97      mov  dword ptr [rax + 4*rcx], edx
98 # %bb.3:                                     #   in Loop:
   Header=BB1_1 Depth=1
99      mov  eax, dword ptr [rbp - 48]
100     add  eax, 1
101     mov  dword ptr [rbp - 48], eax
102     jmp  .LBB1_1
103 .LBB1_4:
104     mov  dword ptr [rbp - 8], 50
105     mov  dword ptr [rbp - 56], 50
106     movss    xmm0, dword ptr [rip + .LCPI1_0] # xmm0 = mem[0],
   zero,zero,zero
107     movss    dword ptr [rbp - 52], xmm0
108     mov  rax, qword ptr [rbp - 56]
109     mov  qword ptr [rbp - 16], rax
110     mov  dword ptr [rbp - 28], 1
111     mov  dword ptr [rbp - 24], 2
112     mov  dword ptr [rbp - 20], 3
113     lea  rcx, [rip + .L.str]
114     call printf
115     mov  eax, dword ptr [rbp - 16]
116     add  rsp, 96
117     pop  rbp
118     ret
119     .seh_endproc
120                                             # -- End function
121     .def printf;
122     .scl 3;
123     .type    32;
124     .endef
125     .p2align  4, 0x90                       # -- Begin
   function printf
126 printf:                                     # @printf
```

ghidra.bat | ready

search

```ghidra
572     if (pcVar1 == (FARPROC)0x0) goto LAB_259b31523;
573   }
574   (*pcVar1)(&DAT_259b3c000,&obj);
575 LAB_259b31523:
576   atexit((_func_5014 *)&__gcc_deregister_frame);
577   return;
578 }
579
580
581
582 // WARNING: Globals starting with '_' overlap smaller symbols at
    the same address
583
584 void init_globals(void)
585
586 {
587   int local_8;
588
589                    // 0x15a0  5  init_globals
590   global_ptr = malloc(0x14);
591   for (local_8 = 0; local_8 < 5; local_8 = local_8 + 1) {
592     *(int *)((longlong)global_ptr + (longlong)local_8 * 4) =
    local_8 * local_8;
593   }
594   _global_builtin = 0x32;
595   _global_struct = 0x3f00000000000032;
596   _global_array = 1;
597   _DAT_259b3f0b8 = 2;
598   _DAT_259b3f0bc = 3;
599   return;
600 }
601
602
603
604 int __cdecl main(int _Argc,char **_Argv,char **_Env)
605
606 {
607   void *pvVar1;
608   int local_38;
609
610                    // 0x1650  6  main
611   __main();
612   init_globals();
613   pvVar1 = malloc(0x14);
614   for (local_38 = 0; local_38 < 5; local_38 = local_38 + 1) {
615     *(int *)((longlong)pvVar1 + (longlong)local_38 * 4) =
    local_38 * local_38;
616   }
617   printf("8270329");
618   return 0x32;
619 }
620
621
622
623 int __cdecl printf(char *_Format,...)
624
625 {
626   int iVar1;
627   FILE *pFVar2;
628   undefined1 (*in_RDX) [10];
629   undefined8 in_R8;
630   undefined8 in_R9;
631   undefined1 (*local_res10) [10];
632   undefined8 local_res18;
633   undefined8 local_res20;
```

retdec.bat | ready

search

```retdec
479
480 // Address range: 0x259b314b0 - 0x259b31557
481 int64_t __gcc_register_frame(void) {
482     int64_t * moduleHandle = GetModuleHandleA(
    "libgcc_s_dw2-1.dll"); // 0x259b314c7
483     int64_t func = 0x259b314a0; // 0x259b314d3
484     if (moduleHandle != NULL) {
485         // 0x259b314d5
486         g17 = (int64_t)LoadLibraryA("libgcc_s_dw2-1.dll");
487         GetProcAddress(moduleHandle, "__register_frame_info");
488         func = (int64_t)GetProcAddress(moduleHandle,
    "__deregister_frame_info");
489     }
490     // 0x259b31523
491     g2 = func;
492     return atexit(0x259b31560);
493 }
494
495 // Address range: 0x259b31560 - 0x259b3159e
496 int64_t __gcc_deregister_frame(void) {
497     // 0x259b31560
498     if (g17 == 0) {
499         // 0x259b31598
500         return g2;
501     }
502     // 0x259b31589
503     return FreeLibrary((int64_t *)g17);
504 }
505
506 // Address range: 0x259b315a0 - 0x259b31647
507 int64_t init_globals(void) {
508     int64_t * mem = malloc(20); // 0x259b315b1
509     *(int64_t *)0x259b3f0a0 = (int64_t)mem;
510     *(int32_t *)mem = 0;
511     for (int64_t i = 1; i < 5; i++) {
512         int32_t v1 = i;
513         *(int32_t *)(4 * i + *(int64_t *)0x259b3f0a0) = v1 * v1;
514     }
515     // 0x259b315f8
516     *(int32_t *)0x259b3f0a8 = 50;
517     __asm_movss_26(__asm_movss(0x3f000000));
518     *(int64_t *)0x259b3f0ac = 50;
519     *(int32_t *)0x259b3f0b4 = 1;
520     *(int32_t *)&g18 = 2;
521     *(int32_t *)&g19 = 3;
522     return 50;
523 }
524
525 // Address range: 0x259b31650 - 0x259b316fe
526 int main(int argc, char ** argv) {
527     // 0x259b31650
528     __main();
529     init_globals();
530     int64_t mem = (int64_t)malloc(20); // 0x259b31677
531     for (int64_t i = 0; i < 5; i++) {
532         int32_t v1 = i;
533         *(int32_t *)(4 * i + mem) = v1 * v1;
534     }
535     // 0x259b316b1
536     __asm_movss_26(__asm_movss(0x3f000000));
537     int64_t v2; // 0x259b31650
538     printf("8270329", 16, v2, v2);
539     return 50;
540 }
541
```

Ontwerp

Codemarkers

Ground truth

```
void functie1() {
    printf("begin functie1");
    int i = 1;
    printf("while loop");
    while (i == 1) {
        ...
```

Decompiler output

```
void function_401000() {
    DWORD v1;
    v1 = 1;
    function_4011a0("begin functie1");
    function_4011a0("while loop");
    while (v1 == 1) {
        ...
```

1 Ideeën

2 Deelonderzoeken

Ontwerp

Optimalisatie tegengaan

Ground truth

```c
int functie1() {
    int result = 0;
    printf("for loop");
    for (int i=0; i<10; i++) {
        result += i;
    }
    return result;
}
```

Assembly

```asm
push    rax
lea     rdi, [rip + .L.str]
xor     eax, eax
call    printf@PLT
mov     eax, 45
pop     rcx
ret
```

Decompiler output

```c
int function_401000() {
    printf("for loop");
    return 45;
}
```

1 Ideeën

2 Deelonderzoeken

Ontwerp

Optimalisatie tegengaan

- IO gebruiken

```
while (true) {
    getchar();
    ...
}
```

```
int i = 10;
// hier berekeningen met i
fwrite(&i, sizeof(int), 1, stdout);
```

1 Ideeën

2 Deelonderzoeken

Ontwerp

Atomisch testen

Ontwerp

Atomisch testen

- functies
- control flow
- datastructuren

Ontwerp

Ground truth

```
void use_double(double* d);

void functie_1() {
    double d = 1.0;
    use_double(&d);

    d = d * 2;
    use_double(&d);
}
```

Assembly

```
push    rbx
sub     rsp, 16
movabs  rax, 4607182418800017408
mov     qword ptr [rsp + 8], rax
lea     rbx, [rsp + 8]
mov     rdi, rbx
call    use_double@PLT
movsd   xmm0, qword ptr [rsp + 8]
addsd   xmm0, xmm0
movsd   qword ptr [rsp + 8], xmm0
mov     rdi, rbx
call    use_double@PLT
add     rsp, 16
pop     rbx
ret
```

Decompiler output

```
void function_403a60() {
    char local[16];
    *(int64_t*)(local) = 4607182418800017408;
    function_403bf0(local);
    *(double*)(local) = *(double*)(local) * 2;
    function_403bf0(local);
}
```

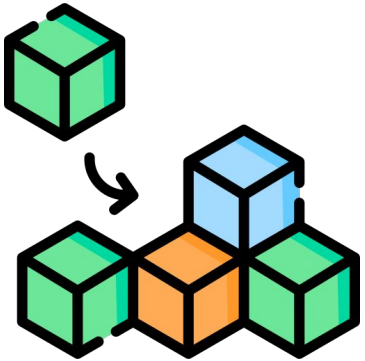
1 Ideeën

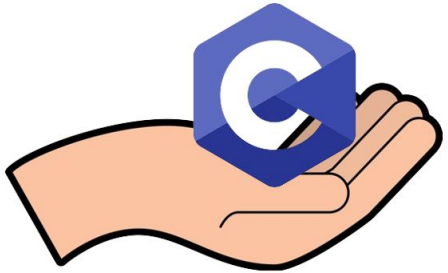
2 Deelonderzoeken

Implementatie

deb'm

1 Generate 2 Assess 3 Report





Implementatie

1  4  4  4 

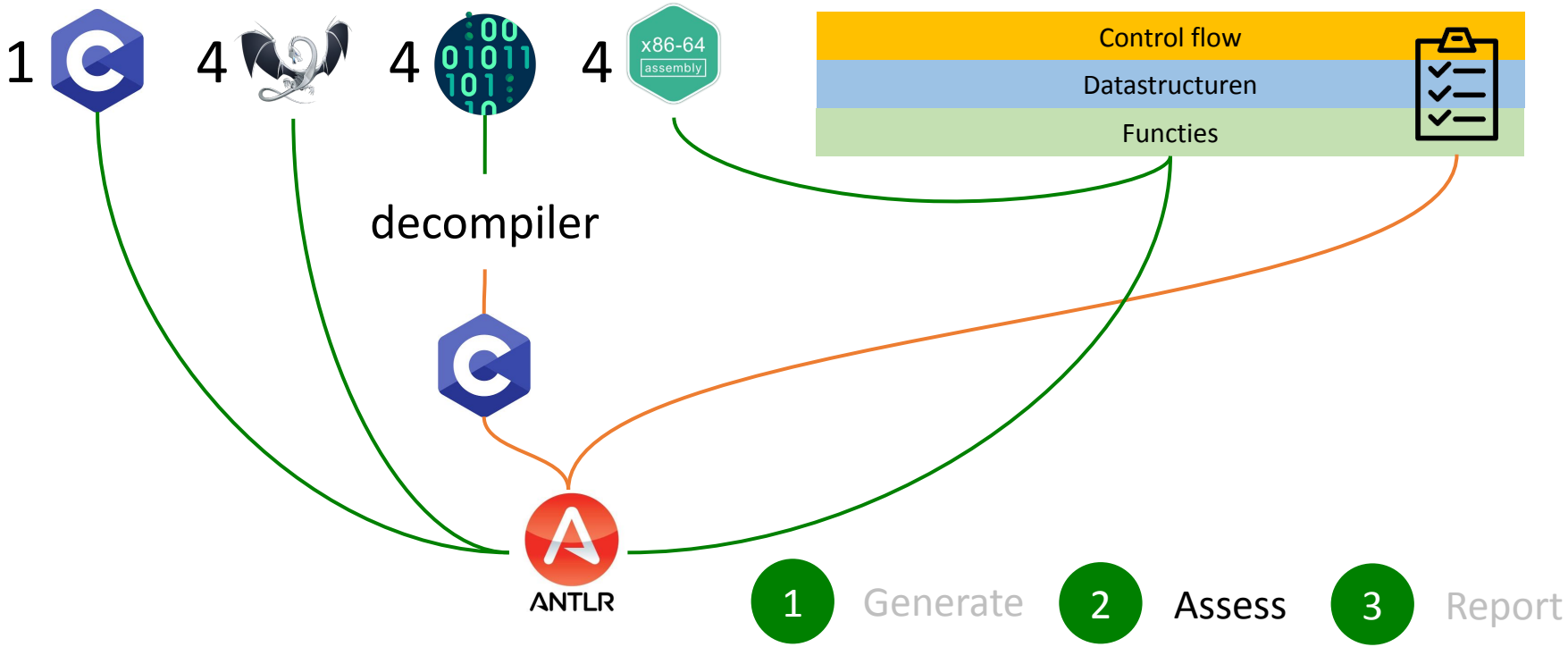
75 C sources = 300 binaries = 1 container

Control flow | Datastructuren | Functies

```c
#include <stdio.h>

struct DS_struct_1 {
  int i;
  struct DS_struct_1* next;
};

int FF_function_1(int x, DS_struct_1 y){
  printf("FF_function_1");
  FF_function_1(3, y);
  while(int i = 0; i < 5; i++){
    printf("begin-loop 1");
    y->i++;
    printf("end-loop 1");
  }
  return 3;
}

int main(){
  struct DS_struct_1* my_struct;
  FF_function_1(3, my_struct);
  return 0;
}
```

1 Generate 2 Assess 3 Report

Implementatie



Implementatie

Code markers

```
__CM_printf("c5852db2-7acb-cba3-7f81-e7ef3cd1d3b8FF>>ID:4f2,function:423,location:START,CHECKSUM:3E50");
```

1 Generate 2 Assess 3 Report

Implementatie

Control flow

Ground truth

```
int i = 14;
__marker("before while loop");
while(i > 7){
  __marker("while loop body start");
  getchar();
  i--;
}
__marker("after while loop");
```

Decompiler output

```
int v1 = 14;
function_010("before while loop");
while(true){
  function_010("while loop body start");
  getchar();
  v1--;
  if(v1 <= 7){
    goto _LAB134;
  }
}
_LAB134:
function_010("after while loop");
```

1 Generate 2 Assess 3 Report

Implementatie

Datastructuren

Ground truth

```
int aantalMensen;
int totaalInkomen;

float gemiddeldInkomen = totaalInkomen /
aantalMensen;
```

Decompiler output

```
float v2;
int v1;
float v3;
v3 = v1 / v2;
```

`typedef int a;`
 	
`a * b;`
	`int a;`
	`int b;`
	`a * b;`

1 Generate 2 Assess 3 Report

Implementatie

Datastructuren

Ground truth

```
int aantalMensen;
__marker("ID:100,%p", &aantalMensen);
int totaalInkomen;
__marker("ID:101,%p", &totaalInkomen);

float gemiddeldInkomen = totaalInkomen /
aantalMensen;
__marker("ID:102,%p", &gemiddeldInkomen);
```

Decompiler output

```
float v2;
int v1;
function_010("ID:100,%p", &v1);
function_010("ID:101,%p", &v2);
float v3;
v3 = v1 / v2;
function_010("ID:102,%p", &v3);
```

Implementatie

Datastructuren

Ground truth

```
int aantalMensen;
__marker("ID:100,%p", &aantalMensen);
int totaalInkomen;
__marker("ID:101,%p", &totaalInkomen);

float gemiddeldInkomen = totaalInkomen /
aantalMensen;
__marker("ID:102,%p", &gemiddeldInkomen);
```

Decompiler output

```
float v2;
int v1;
function_010("ID:100,%p", &v1);
function_010("ID:101,%p", &v2);
float v3;
v3 = v1 / v2;
function_010("ID:102,%p", &v3);
```

Marker ID	Origineel	Na decompilatie
100	int	int
101	int	float
102	float	float

1 Generate 2 Assess 3 Report

Implementatie

Functies

Ground truth

```
float berekenGemiddeldInkomen(int[] x){

}

int berekenTotaalInkomen(){

}
```

Decompiler output

```
type_4 function_01400004(){

}

type_4 function_01400308(int * a1){
  int uVar1;
  type_4 *puVar3;

  uVar1 = *a1;
}
```

1 Generate 2 Assess 3 Report

Implementatie

Functies

Ground truth

```
float berekenGemiddeldInkomen(int[] x){
  __marker("berekenGemiddeldInkomen");
}

int berekenTotaalInkomen(){
  __marker("berekenTotaalInkomen");
}
```

Decompiler output

```
type_4 function_01400004(){
  function_010("berekenTotaalInkomen");
}

type_4 function_01400308(int * a1){
  int uVar1;
  type_4 *puVar3;

  uVar1 = *a1;
  function_010("berekenGemiddeldInkomen");
}
```

1 Generate 2 Assess 3 Report

Implementatie

Functies

Ground truth (x64 assembly)

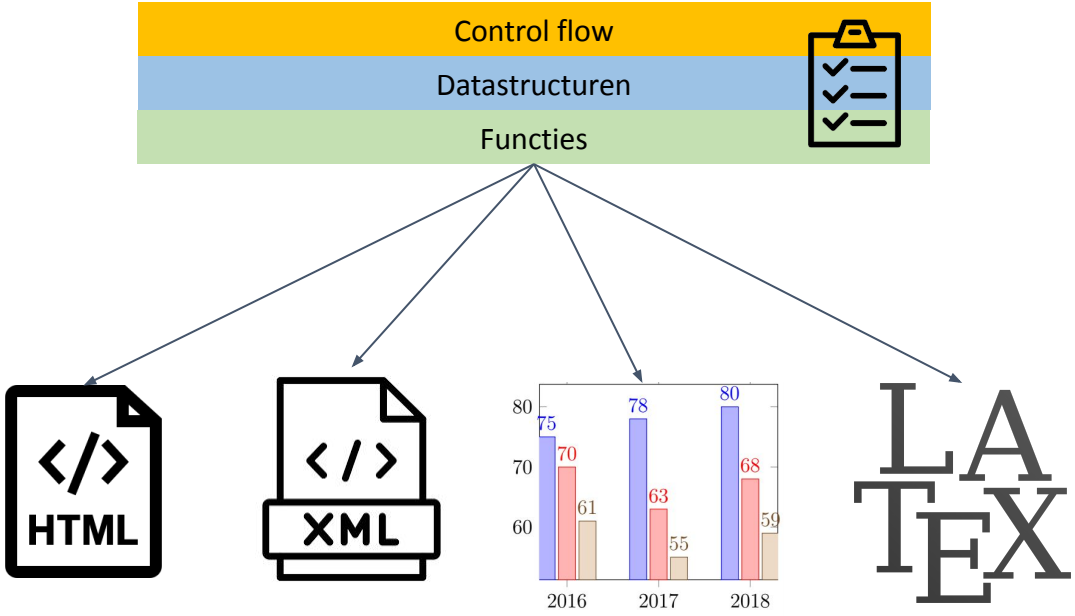
```
_berekenGemiddeldInkomen:
.seh_proc func_1
pushq %rsi
.seh_pushreg %rsi
subq $112, %rsp
.seh_stackalloc 112
.seh_endprologue
movq %rcx, %rsi
movq %rsi, 88(%rsp)
movq %rdx, 80(%rsp)
movl %r8d, 44(%rsp)
movb %r9b, 43(%rsp)
leaq .L.str.88(%rip), %rcx
callq __marker
```

Decompiler output

```
type_4 function_01400004(){
  function_010("berekenTotaalInkomen");
}

type_4 function_01400308(int * a1){
  int uVar1;
  type_4 *puVar3;

  uVar1 = *a1;
  function_010("berekenGemiddeldInkomen");
}
```

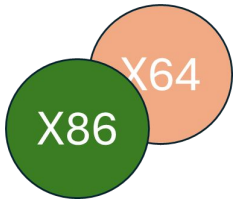
1 Generate 2 Assess 3 Report

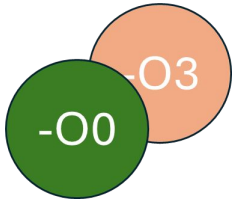
Implementatie



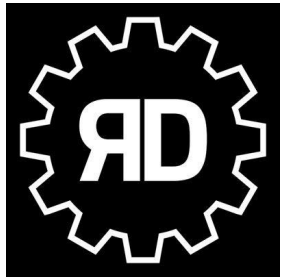
1 Generate 2 Assess 3 Report

Resultaten





- Overall winner



- Snel
- Als enige functie start gemist
- 3x zoveel goto’s

- Als enige variadic function gevonden (x64)

In een notendop

- Uitdagingen bij decompilatie
- Uitdagingen bij benchmarken
- Implementatie: deb’m
  - Generate (C code -> compiler)
  - Assess (decompiler -> parsen -> assessen -> resultaten)
  - Report
- Resultaten