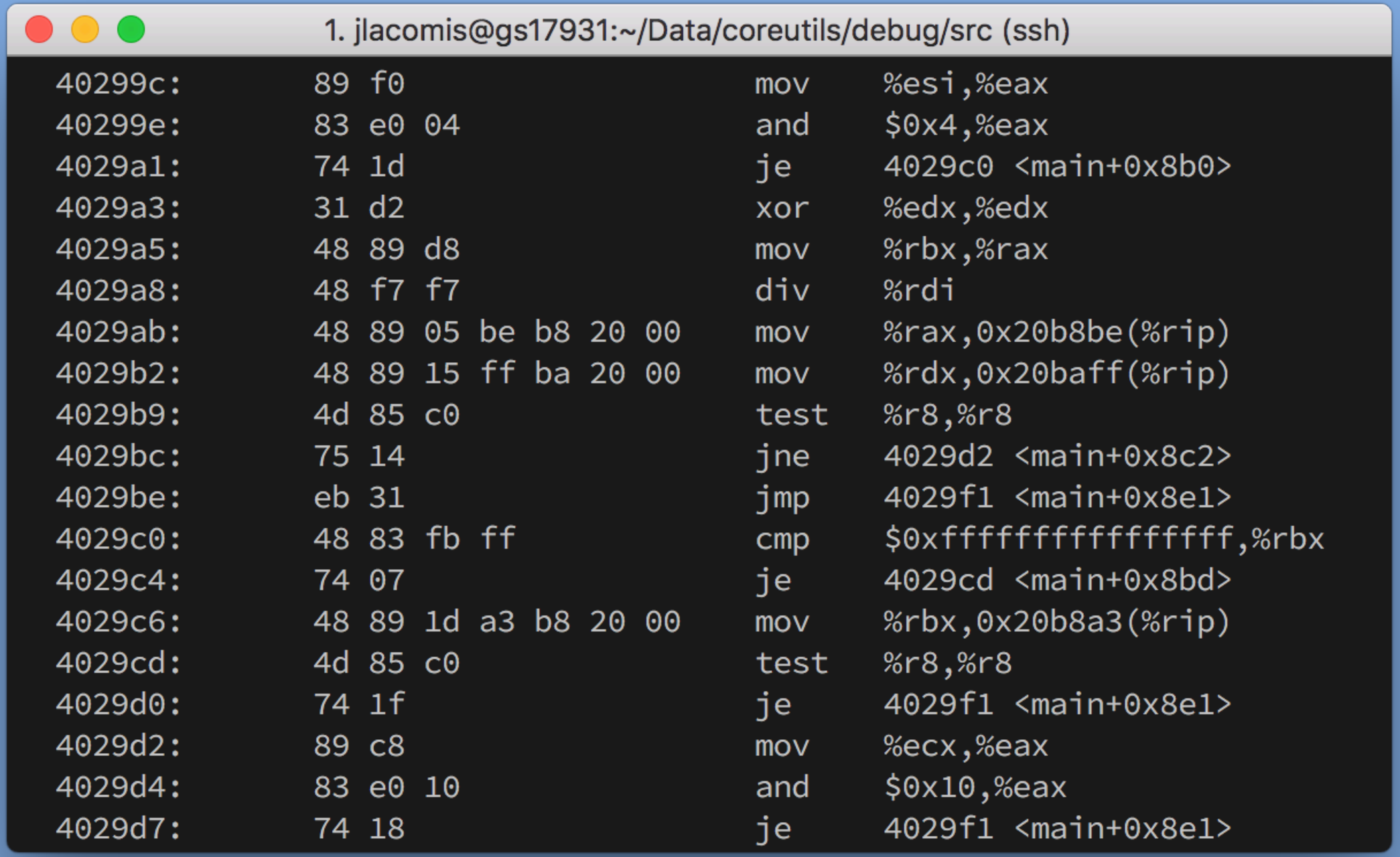
Constraint-Guided Statistical Type Reconstruction for Decompilation

Jeremy Lacomis
Carnegie Mellon University

STRUDEL
SOCIO-TECHNICAL RESEARCH USING DATA EXCAVATION LAB

squaresLab
Software Quality in Real Evolving Systems

Software Engineering Institute
Carnegie Mellon


1. jlacomis@gs17931:~/Data/coreutils/debug/src (ssh)
40299c: 89 f0 mov %esi,%eax
40299e: 83 e0 04 and $0x4,%eax
4029a1: 74 1d je 4029c0 <main+0x8b0>
4029a3: 31 d2 xor %edx,%edx
4029a5: 48 89 d8 mov %rbx,%rax
4029a8: 48 f7 f7 div %rdi
4029ab: 48 89 05 be b8 20 00 mov %rax,0x20b8be(%rip)
4029b2: 48 89 15 ff ba 20 00 mov %rdx,0x20baff(%rip)
4029b9: 4d 85 c0 test %r8,%r8
4029bc: 75 14 jne 4029d2 <main+0x8c2>
4029be: eb 31 jmp 4029f1 <main+0x8e1>
4029c0: 48 83 fb ff cmp $0xffffffffffffffff,%rbx
4029c4: 74 07 je 4029cd <main+0x8bd>
4029c6: 48 89 1d a3 b8 20 00 mov %rbx,0x20b8a3(%rip)
4029cd: 4d 85 c0 test %r8,%r8
4029d0: 74 1f je 4029f1 <main+0x8e1>
4029d2: 89 c8 mov %ecx,%eax
4029d4: 83 e0 10 and $0x10,%eax
4029d7: 74 18 je 4029f1 <main+0x8e1>

Reverse Engineering

26 Feb 2013 | 14:00 GMT

The Real Story of Stuxnet

How Kaspersky Lab tracked down the malware that stymied Iran's nuclear-fuel enrichment program

By **David Kushner**

Computer cables snake across the floor. Cryptic flowcharts are scrawled across various whiteboards adorning the walls. A life-size Batman doll stands in the hall. This office might seem no different than any other geeky workplace, but in fact it's the front line of a war—a cyberwar, where most battles play out not in remote jungles or deserts but in

2014 IEEE International Conference on Software Maintenance and Evolution

Reverse Engineering PL/SQL Legacy Code: An Experience Report

Martin Habringer
voestalpine Stahl GmbH
4020 Linz, Austria
martin.habringer@voestalpine.com

Michael Moser and Josef Pichler
Software Analytics and Evolution
Software Competence Center Hagenberg GmbH
4232 Hagenberg, Austria
michael.moser@scch.at, josef.pichler@scch.at

***Abstract*—The reengineering of legacy code is a tedious endeavor. Automatic transformation of legacy code from an old technology to a new one preserves potential problems in legacy code with respect to obsolete, changed, and new business cases. On the other hand, manual analysis of legacy code without assistance of original developers is time consuming and error-prone. For the purpose of reengineering PL/SQL legacy code in the steel making domain, we developed tool support for the reverse engineering of PL/SQL code into a more abstract and comprehensive representation. This representation then serves as input for stakeholders to manually analyze legacy code, to identify obsolete and missing business cases, and, finally, to support the re-implementation of a new system. In this paper we briefly introduce the tool and present results of reverse engineering PL/SQL legacy code in the steel making domain. We show how stakeholders are supported in analyzing legacy code by means of general-purpose analysis techniques combined with domain-specific representations and conclude with some of the lessons learned.**

***Keywords*—reverse engineering; program comprehension; source code analysis**

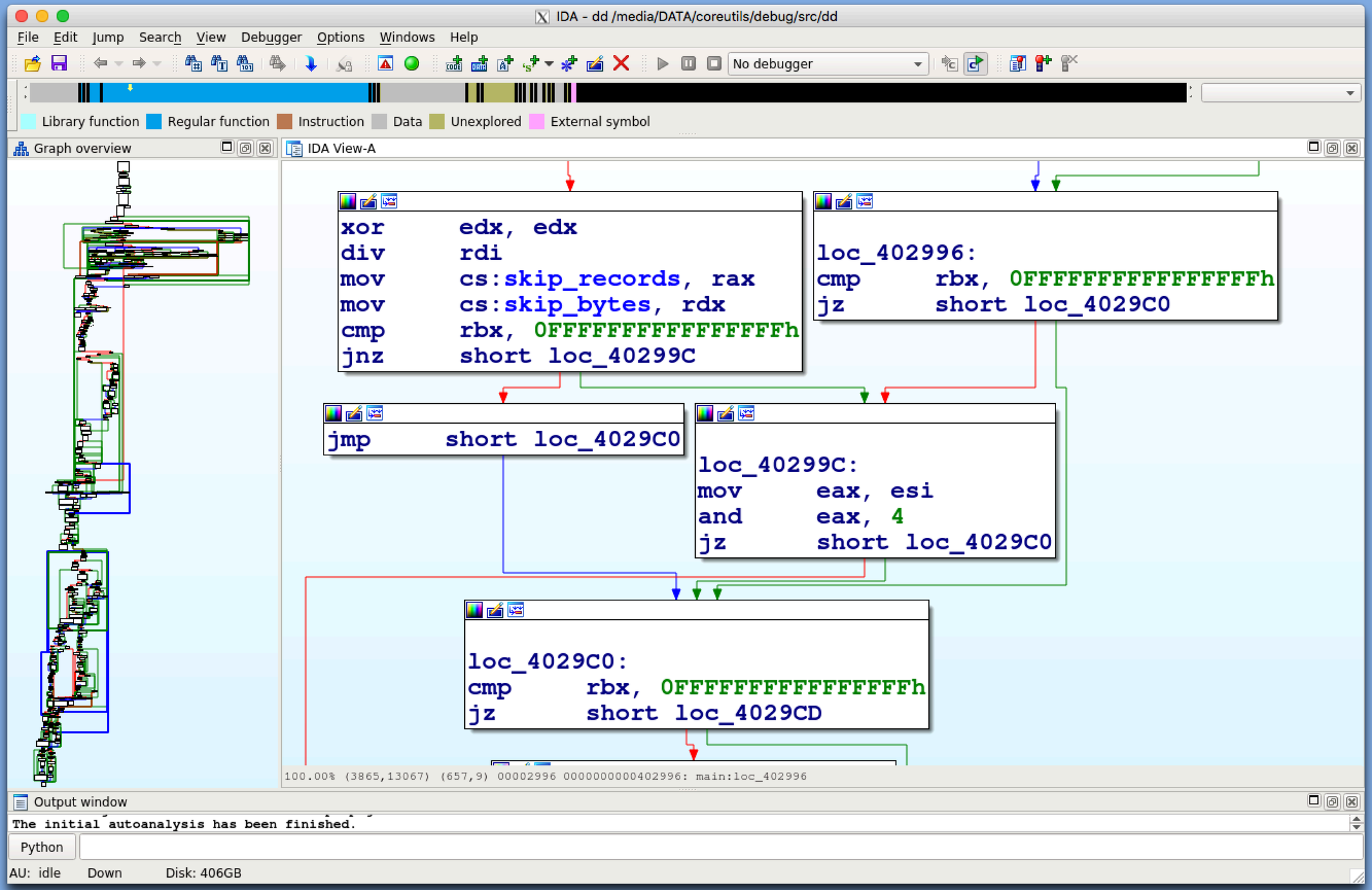
I. Introduction

- Changes in business cases over the last years were not reflected in verification logic of the legacy code.
- For a new production plant, additional requirements must be incorporated.
- The maintenance of the legacy programs was complicated by the retirement of original developers.
- Legacy code is not extensible in a safe and reliable way.
- Stakeholders estimated high effort for manual analysis of the legacy code.

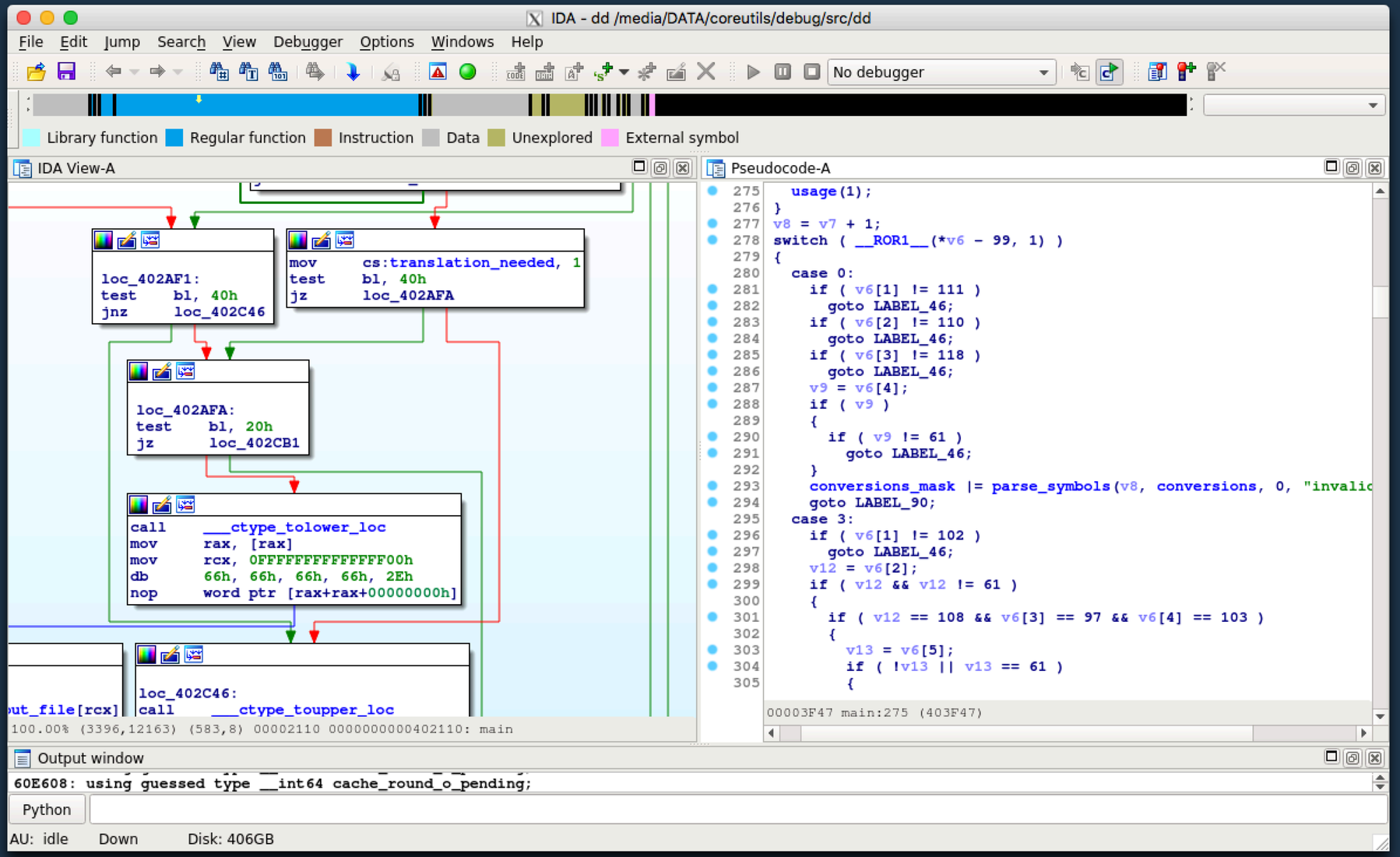
The goal for the reverse engineering tool was to support stakeholders to comprehend the verification logic implemented in the legacy programs. Whereas, comprehension requires that stakeholders can (1) *identify* the business cases currently checked by the software as well as that stakeholders are able to (2) *extend* the verification logic with respect to new requirements.

The contributions of this paper are:

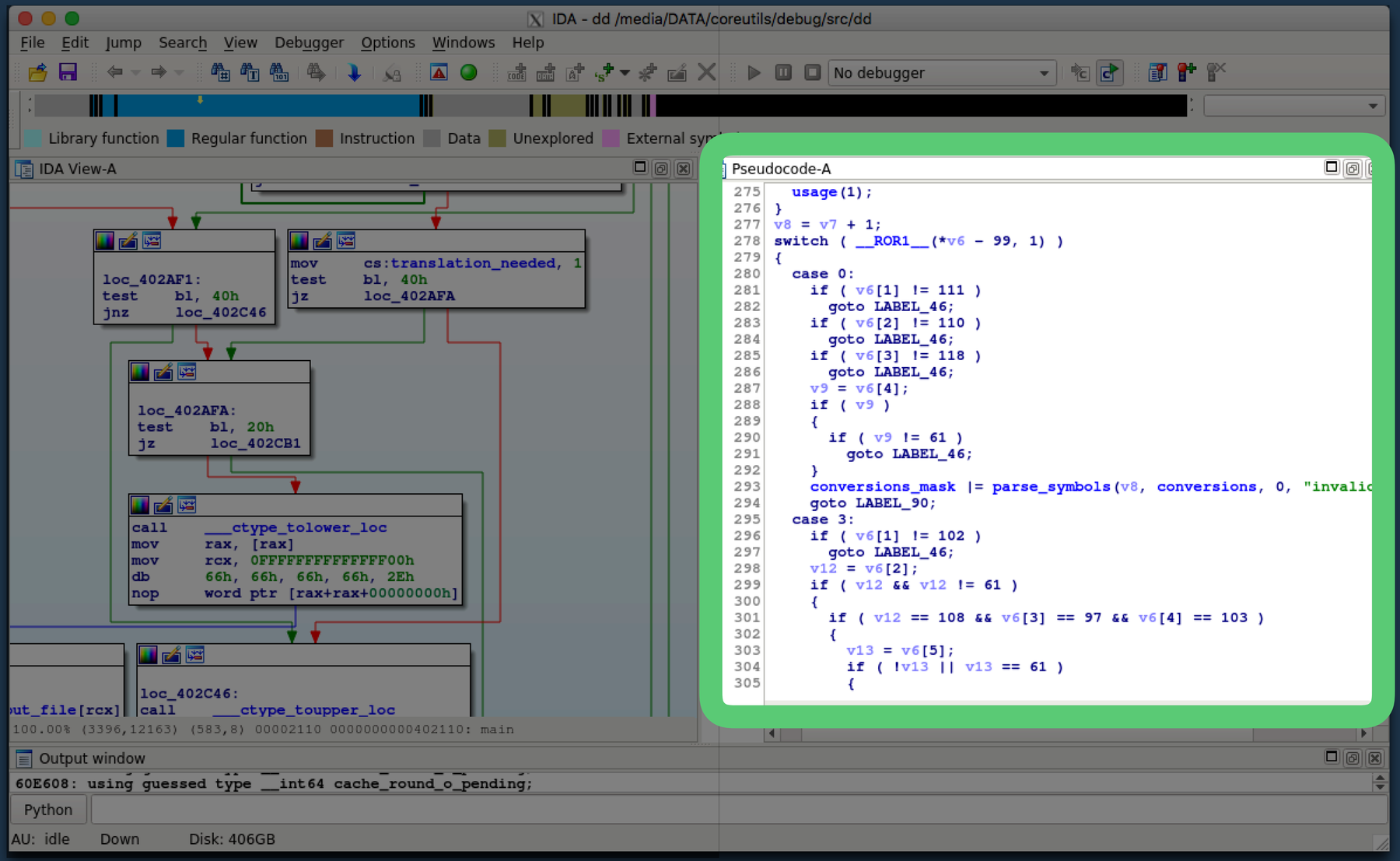
1) Presenting requirements for a reverse engineering tool aimed at PL/SQL legacy code.


IDA - dd /media/DATA/coreutils/debug/src/dd
File Edit Jump Search View Debugger Options Windows Help
No debugger
Library function
Regular function
Instruction
Data
Unexplored
External symbol
Graph overview
IDA View-A
xor edx, edx
div rdi
mov cs:skip_records, rax
mov cs:skip_bytes, rdx
cmp rbx, 0FFFFFFFFFFFFFFFFh
jnz short loc_40299C
loc_402996:
cmp rbx, 0FFFFFFFFFFFFFFFFh
jz short loc_4029C0
jmp short loc_4029C0
loc_40299C:
mov eax, esi
and eax, 4
jz short loc_4029C0
loc_4029C0:
cmp rbx, 0FFFFFFFFFFFFFFFFh
jz short loc_4029CD
100.00% (3865,13067) (657,9) 00002996 0000000000402996: main:loc_402996
Output window
The initial autoanalysis has been finished.
Python
AU: idle Down Disk: 406GB

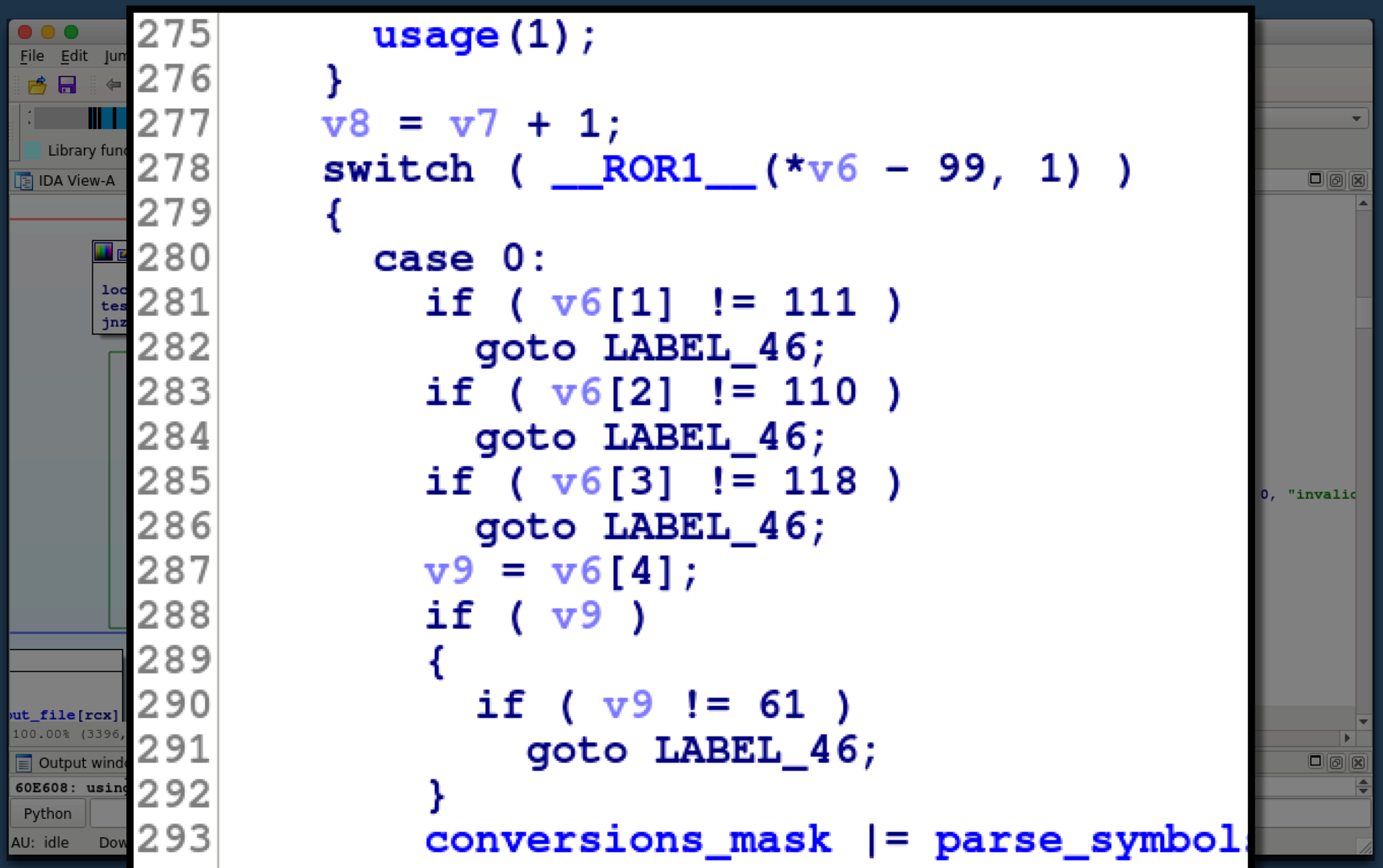
Decompiler


IDA - dd /media/DATA/coreutils/debug/src/dd
File Edit Jump Search View Debugger Options Windows Help
No debugger
Library function Regular function Instruction Data Unexplored External symbol
IDA View-A
loc_402AF1:
test bl, 40h
jnz loc_402C46
mov cs:translation_needed, 1
test bl, 40h
jz loc_402AFA
loc_402AFA:
test bl, 20h
jz loc_402CB1
call ___ctype_tolower_loc
mov rax, [rax]
mov rcx, 0FFFFFFFFFFFFFF00h
db 66h, 66h, 66h, 66h, 2Eh
nop word ptr [rax+rax+00000000h]
loc_402C46:
call ___ctype_toupper_loc
100.00% (3396,12163) (583,8) 00002110 0000000000402110: main
Pseudocode-A
275 usage(1);
276 }
277 v8 = v7 + 1;
278 switch (__ROR1__(*v6 - 99, 1))
279 {
280 case 0:
281 if (v6[1] != 111)
282 goto LABEL_46;
283 if (v6[2] != 110)
284 goto LABEL_46;
285 if (v6[3] != 118)
286 goto LABEL_46;
287 v9 = v6[4];
288 if (v9)
289 {
290 if (v9 != 61)
291 goto LABEL_46;
292 }
293 conversions_mask |= parse_symbols(v8, conversions, 0, "invalic
294 goto LABEL_90;
295 case 3:
296 if (v6[1] != 102)
297 goto LABEL_46;
298 v12 = v6[2];
299 if (v12 && v12 != 61)
300 {
301 if (v12 == 108 && v6[3] == 97 && v6[4] == 103)
302 {
303 v13 = v6[5];
304 if (!v13 || v13 == 61)
305 {
00003F47 main:275 (403F47)
Output window
60E608: using guessed type __int64 cache_round_o_pending;
Python
AU: idle Down Disk: 406GB

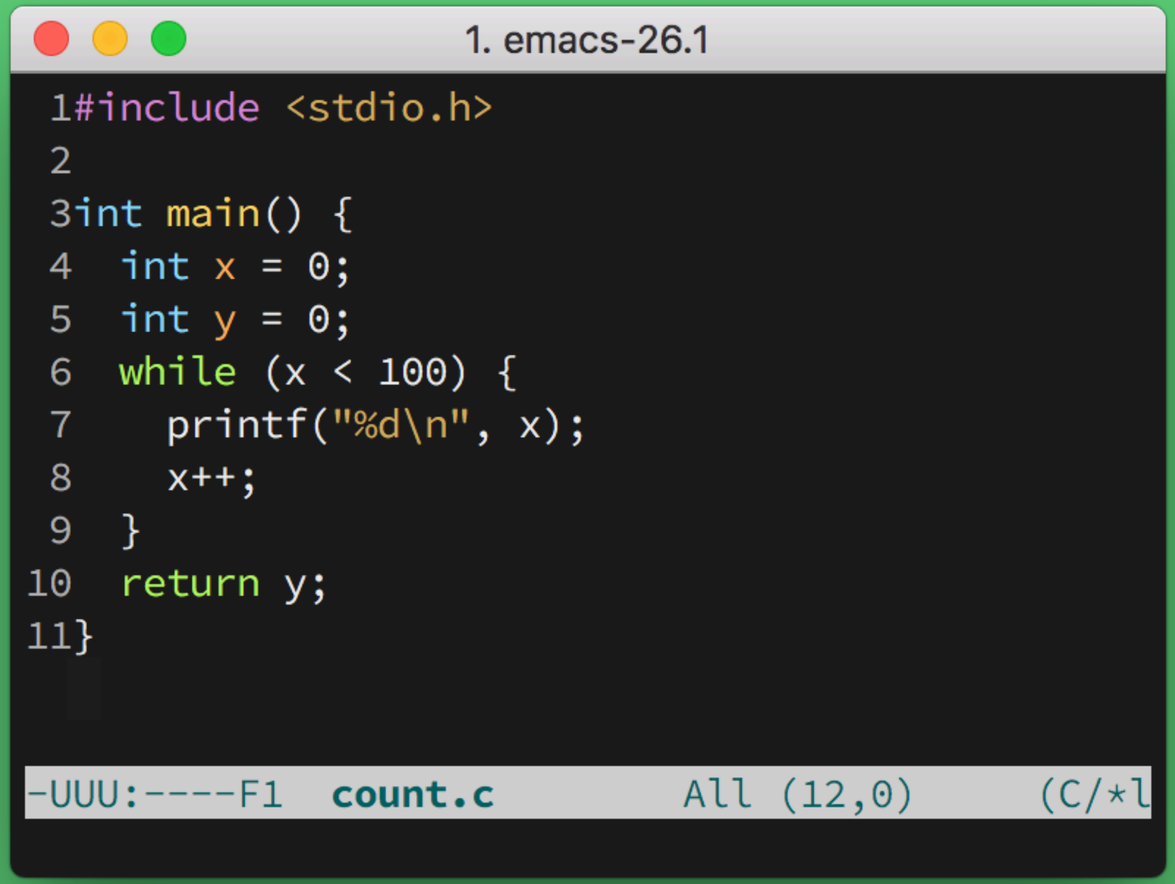
Decompiler

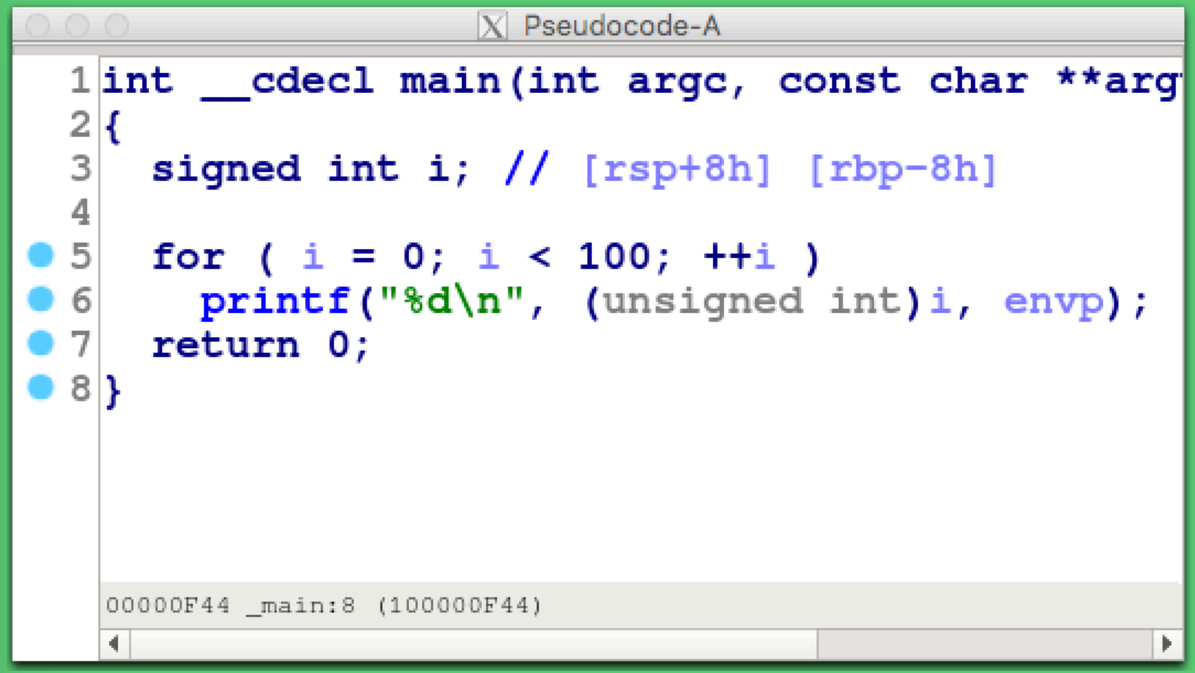

IDA - dd /media/DATA/coreutils/debug/src/dd
File Edit Jump Search View Debugger Options Windows Help
No debugger
Library function Regular function Instruction Data Unexplored
IDA View-A
loc_402AF1:
test bl, 40h
jnz loc_402C46
mov cs:translation_needed, 1
test bl, 40h
jz loc_402AFA
loc_402AFA:
test bl, 20h
jz loc_402CB1
call ___ctype_tolower_loc
mov rax, [rax]
mov rcx, 0FFFFFFFFFFFFFF00h
db 66h, 66h, 66h, 66h, 2Eh
nop word ptr [rax+rax+00000000h]
loc_402C46:
call ___ctype_toupper_loc
100.00% (3396,12163) (583,8) 00002110 0000000000402110: main
Pseudocode-A
275 usage(1);
276 }
277 v8 = v7 + 1;
278 switch (__ROR1__(*v6 - 99, 1))
279 {
280 case 0:
281 if (v6[1] != 111)
282 goto LABEL_46;
283 if (v6[2] != 110)
284 goto LABEL_46;
285 if (v6[3] != 118)
286 goto LABEL_46;
287 v9 = v6[4];
288 if (v9)
289 {
290 if (v9 != 61)
291 goto LABEL_46;
292 }
293 conversions_mask |= parse_symbols(v8, conversions, 0, "invalic
294 goto LABEL_90;
295 case 3:
296 if (v6[1] != 102)
297 goto LABEL_46;
298 v12 = v6[2];
299 if (v12 && v12 != 61)
300 {
301 if (v12 == 108 && v6[3] == 97 && v6[4] == 103)
302 {
303 v13 = v6[5];
304 if (!v13 || v13 == 61)
305 {
Output window
60E608: using guessed type __int64 cache_round_o_pending;
Python
AU: idle Down Disk: 406GB

Decompiler


File Edit Jun
Library func
IDA View-A
Output wind
60E608: using
Python
AU: idle
275 usage(1);
276 }
277 v8 = v7 + 1;
278 switch (__ROR1__(*v6 - 99, 1))
279 {
280 case 0:
281 if (v6[1] != 111)
282 goto LABEL_46;
283 if (v6[2] != 110)
284 goto LABEL_46;
285 if (v6[3] != 118)
286 goto LABEL_46;
287 v9 = v6[4];
288 if (v9)
289 {
290 if (v9 != 61)
291 goto LABEL_46;
292 }
293 conversions_mask |= parse_symbol

Decompilers *don't* Recover Code!





Compilation Loses Information

Comments:

```
/* This is the functionality
 * you're looking for! */
```

Loop Constructs:

```
while (x < 100) {...}

for (i = 0; i < 100; ++i) {...}
```

Variable Names:

```
int width, length;
double volume;
char *user_id;
```

User-Defined Types:

```
typedef struct {
  int x;
  int y;
} point;
```

Compilation Loses Information

Comments:

```
/* This is the functionality
 * you're looking for! */
```

Loop Constructs:

```
while (x < 100) {...}

for (i = 0; i < 100; ++i) {...}
```

Variable Names:

```
int width, length;
double volume;
char *user_id;
```

User-Defined Types:

```
typedef struct {
  int x;
  int y;
} point;
```

Compilation Loses Information

Comments:

```
/* This is the functionality
 * you're looking for! */
```

Loop Constructs:

```
while (x < 100) {...}

for (i = 0; i < 100; ++i) {...}
```

Variable Names:

```
int width, length;
double volume;
char *user_id;
```

User-Defined Types:

```
typedef struct {
  int x;
  int y;
} point;
```

Types Improve Variable Renaming

```
unsigned int width, total, attendance;
size_t name_length;
```

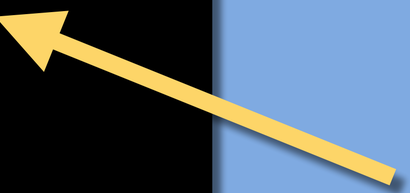
Meaningful Variable Names for Decompiled Code: A Machine Translation Approach,
A. Jaffe, J. Lacomis, E. J. Schwartz, C. Le Goues, and B. Vasilescu, in *ICPC*, 2018

Types Can Guide Understanding

```
double func(int *a1, int *a2) {
  double v1, v2;

  v1 = pow((*a1 - *a2), 2);
  v2 = pow((a1[1] - a2[1]), 2);

  return sqrt(v1 + v2);
}
```

Types Can Guide Understanding

```
double func(int *a1, int *a2) {
  double v1, v2;

  v1 = pow((*a1 - *a2), 2);
  v2 = pow((a1[1] - a2[1]), 2);

  return sqrt(v1 + v2);
}
```



`int` pointer arguments

Types Can Guide Understanding

```
double func(int *a1, int *a2) {
  double v1, v2;

  v1 = pow((*a1 - *a2), 2);
  v2 = pow((a1[1] - a2[1]), 2);

  return sqrt(v1 + v2);
}
```

Dereference and do some math

Types Can Guide Understanding

```
double func(int *a1, int *a2) {
  double v1, v2;

  v1 = pow((*a1 - *a2), 2);
  v2 = pow((a1[1] - a2[1]), 2);

  return sqrt(v1 + v2);
}
```

Index like an array?!?

Types Can Guide Understanding

```
typedef struct {
  int x;
  int y;
} point;

double func(point *a1, point *a2) {
  double v1, v2;

  v1 = pow((a1->x - a2->x), 2);
  v2 = pow((a1->y - a2->y), 2);

  return sqrt(v1 + v2);
}
```

Types Can Guide Understanding

```
double func(int *a1, int *a2) {
  double v1, v2;

  v1 = pow((*a1 - *a2), 2);
  v2 = pow((a1[1] - a2[1]), 2);

  return sqrt(v1 + v2);
}
```

```
typedef struct {
  int x;
  int y;
} point;

double func(point *a1, point *a2) {
  double v1, v2;

  v1 = pow((a1->x - a2->x), 2);
  v2 = pow((a1->y - a2->y), 2);

  return sqrt(v1 + v2);
}
```

Types Can Guide Understanding

```
double func(int *a1, int *a2) {
  double v1, v2;

  v1 = pow((*a1 - *a2), 2);
  v2 = pow((a1[1] - a2[1]), 2);

  return sqrt(v1 + v2);
}
```

```
typedef struct {
  int x;
  int y;
} point;

double func(point *a1, point *a2) {
  double v1, v2;

  v1 = pow((a1->x - a2->x), 2);
  v2 = pow((a1->y - a2->y), 2);

  return sqrt(v1 + v2);
}
```

Types Can Guide Understanding

```
double func(int *a1, int *a2) {
  double v1, v2;

  v1 = pow((*a1 - *a2), 2);
  v2 = pow((a1[1] - a2[1]), 2);

  return sqrt(v1 + v2);
}
```

```
typedef struct {
  int x;
  int y;
} point;

double func(point *a1, point *a2) {
  double v1, v2;

  v1 = pow((a1->x - a2->x), 2);
  v2 = pow((a1->y - a2->y), 2);

  return sqrt(v1 + v2);
}
```

Types Can Guide Understanding

```
double func(int *a1, int *a2) {
  double v1, v2;

  v1 = pow((*a1 - *a2), 2);
  v2 = pow((a1[1] - a2[1]), 2);

  return sqrt(v1 + v2);
}
```

```
typedef struct {
  int x;
  int y;
} point;

double func(point *a1, point *a2) {
  double v1, v2;

  v1 = pow((a1->x - a2->x), 2);
  v2 = pow((a1->y - a2->y), 2);

  return sqrt(v1 + v2);
}
```

Statistical: Select Types Using ML

```
double func(int *a1, int *a2) {
  double v1, v2;

  v1 = pow((*a1 - *a2), 2);
  v2 = pow((a1[1] - a2[1]), 2);

  return sqrt(v1 + v2);
}
```

```
typedef struct {
  int x;
  int y;
} point;

double func(point *a1, point *a2) {
  double v1, v2;

  v1 = pow((a1->x - a2->x), 2);
  v2 = pow((a1->y - a2->y), 2);

  return sqrt(v1 + v2);
}
```

Statistical: Select Names Using ML

```
double func(int *a1, int *a2) {
  double v1, v2;

  v1 = pow((*a1 - *a2), 2);
  v2 = pow((a1[1] - a2[1]), 2);

  return sqrt(v1 + v2);
}
```

```
double func(int *pnt1, int *pnt2) {
  double v1, v2;

  v1 = pow((*a1 - *a2), 2);
  v2 = pow((a1[1] - a2[1]), 2);

  return sqrt(v1 + v2);
}
```

Constraint-Guided: Restrict to Compatible Types

```
double func(int *a1, int *a2) {
  double v1, v2;

  v1 = pow((*a1 - *a2), 2);
  v2 = pow((a1[1] - a2[1]), 2);

  return sqrt(v1 + v2);
}
```

```
typedef struct {
  int x;
  int y;
} point;
```

```
double func(point *a1, point *a2) {
  double v1, v2;

  v1 = pow((a1->x - a2->x), 2);
  v2 = pow((a1->y - a2->y), 2);

  return sqrt(v1 + v2);
}
```

Type Constraints

```
typedef struct {
  int x;
  int y;
} point;

sizeof(char);     // 1

sizeof(int);      // 4
sizeof(float);    // 4

sizeof(point);    // 8
sizeof(int[2]);   // 8
```

Type Constraints

```
typedef struct {
  int x;
  int y;
} point;

sizeof(char);     // 1

sizeof(int);      // 4
sizeof(float);    // 4

sizeof(point);    // 8
sizeof(int[2]);   // 8
```

Type Constraints

```
typedef struct {
  int x;
  int y;
} point;

sizeof(char);     // 1

sizeof(int);      // 4
sizeof(float);    // 4

sizeof(point);    // 8
sizeof(int[2]);   // 8
```

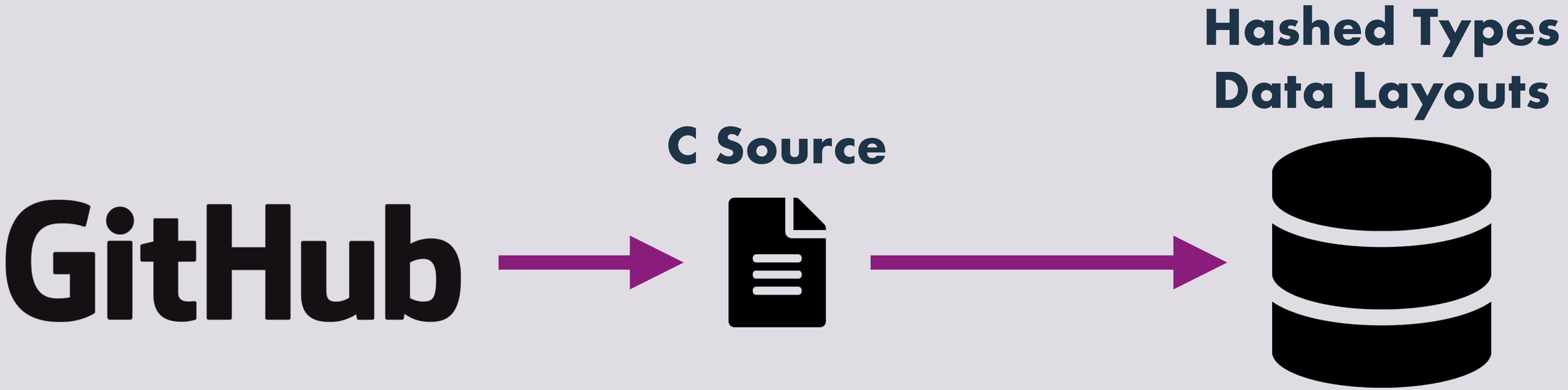
Type Constraints

```
typedef struct {
  int x;
  int y;
} point;

sizeof(char);     // 1

sizeof(int);      // 4
sizeof(float);    // 4

sizeof(point);    // 8
sizeof(int[2]);   // 8
```

Type Constraints

```
typedef struct {
  int x;
  int y;
} point;

sizeof(char);     // 1

sizeof(int);      // 4
sizeof(float);    // 4

sizeof(point);    // 8
sizeof(int[2]);   // 8
```

Mining Types



Meaningful Variable Names for Decompiled Code: A Machine Translation Approach,
A. Jaffe, J. Lacomis, E. J. Schwartz, C. Le Goues, and B. Vasilescu, in *ICPC*, 2018

Statistical: Type Assignment using ML

```
double func(int *a1, int *a2) {
  double v1, v2;

  v1 = pow((*a1 - *a2), 2);
  v2 = pow((a1[1] - a2[1]), 2);

  return sqrt(v1 + v2);
}
```

```
typedef struct {
  int x;
  int y;
} point;

double func(point *a1, point *a2) {
  double v1, v2;

  v1 = pow((a1->x - a2->x), 2);
  v2 = pow((a1->y - a2->y), 2);

  return sqrt(v1 + v2);
}
```

Previous Work: n-gram Models

I → am → very → tired
I → am → very → banana

Previous Work: n-gram Models

I → am → very → tired
I → am → very → banana

int → main → (→ int → argc
int → main → (→ int → banana

Missing Context

Decompiler Output

```
double func(int *a1, int *a2) {
  double v1, v2;

  v1 = pow((*a1 - *a2), 2);
  v2 = pow((a1[1] - a2[1]), 2);

  return sqrt(v1 + v2);
}
```

Original Code

```
double func(point *a1, point *a2) {
  double v1, v2;

  v1 = pow((a1->x - a2->x), 2);
  v2 = pow((a1->y - a2->y), 2);

  return sqrt(v1 + v2);
}
```

Missing Context

Decompiler Output

```
double func(int *a1, int *a2) {
  double v1, v2;

  v1 = pow((*a1 - *a2), 2);
  v2 = pow((a1[1] - a2[1]), 2);

  return sqrt(v1 + v2);
}
```

Original Code

```
double func(point *a1, point *a2) {
  double v1, v2;

  v1 = pow((a1->x - a2->x), 2);
  v2 = pow((a1->y - a2->y), 2);

  return sqrt(v1 + v2);
}
```

Missing Context

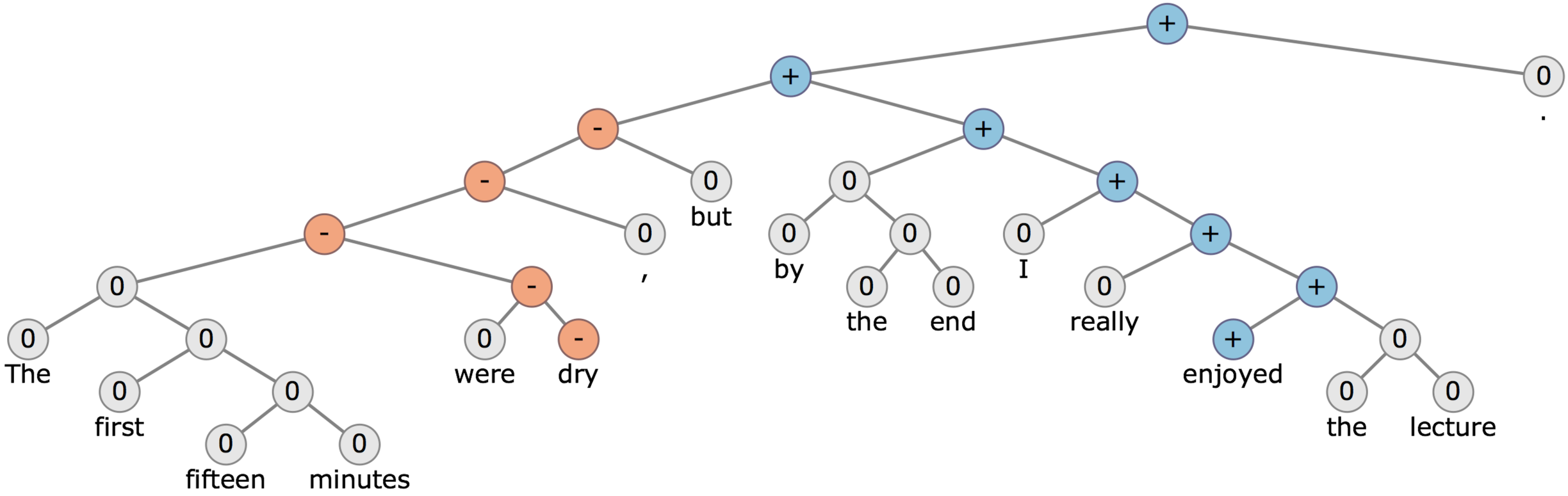
Decompiler Output

```
double func(int *a1, int *a2) {
  double v1, v2;

  v1 = pow((*a1 - *a2), 2);
  v2 = pow((a1[1] - a2[1]), 2);

  return sqrt(v1 + v2);
}
```

Original Code

```
double func(point *a1, point *a2) {
  double v1, v2;

  v1 = pow((a1->x - a2->x), 2);
  v2 = pow((a1->y - a2->y), 2);

  return sqrt(v1 + v2);
}
```

Inspiration: Text Classification



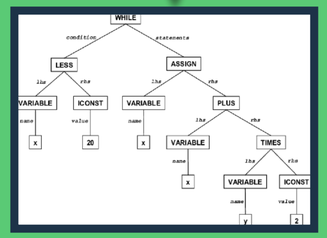
Compilation

Source Code

```
double func(int *a1, int *a2) {...}
```

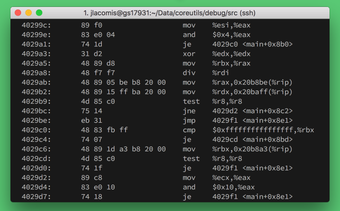
Parse

Abstract Syntax Tree



Generate Assembly

Assembly Code



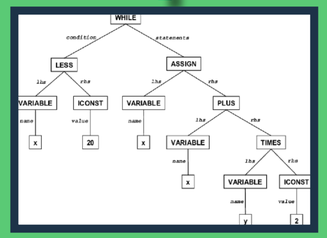
Decompilation

Decompiled Code

```
double func(int *a1, int *a2) {...}
```

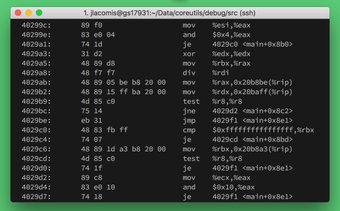
Generate Code

Abstract Syntax Tree



Generate Tree

Disassembled Code



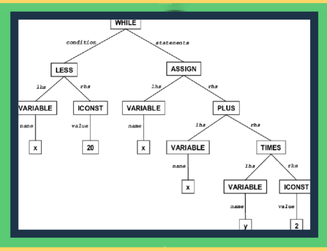
Decompilation

Decompiled Code

```
double func(int *a1, int *a2) {...}
```

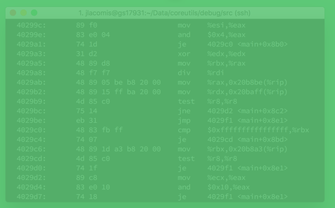
Generate Code

Abstract Syntax Tree



Generate Tree

Disassembled Code



Strategy

Decompiler Output

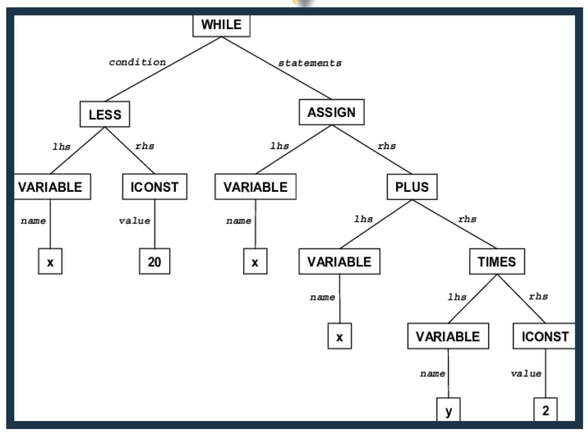
```
double func(int *a1, int *a2) {...}
```

Strategy

Decompiler Output

```
double func(int *a1, int *a2) {...}
```

Parse

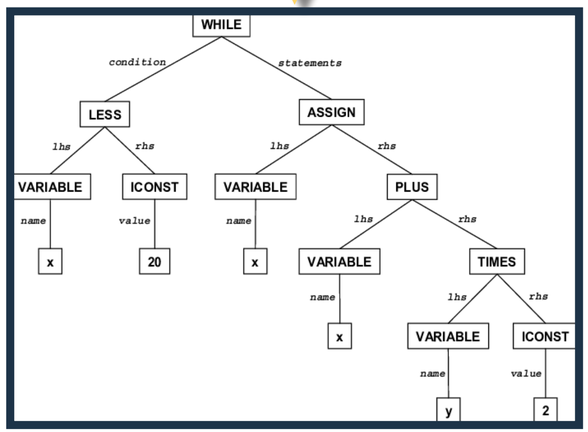


Strategy

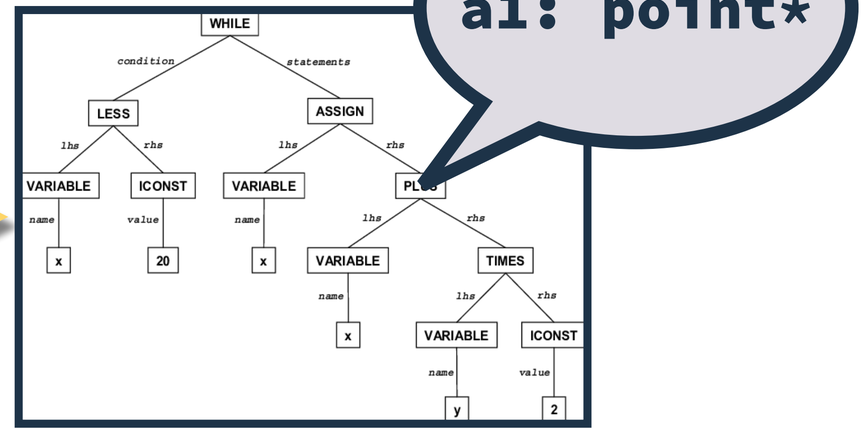
Decompiler Output

```
double func(int *a1, int *a2) {...}
```

Parse



Tagger

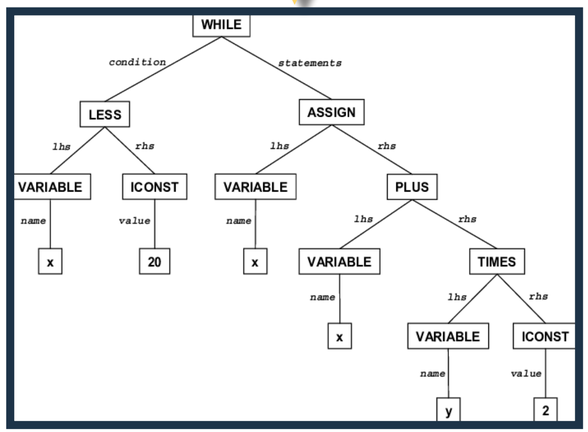


Strategy

Decompiler Output

```
double func(int *a1, int *a2) {...}
```

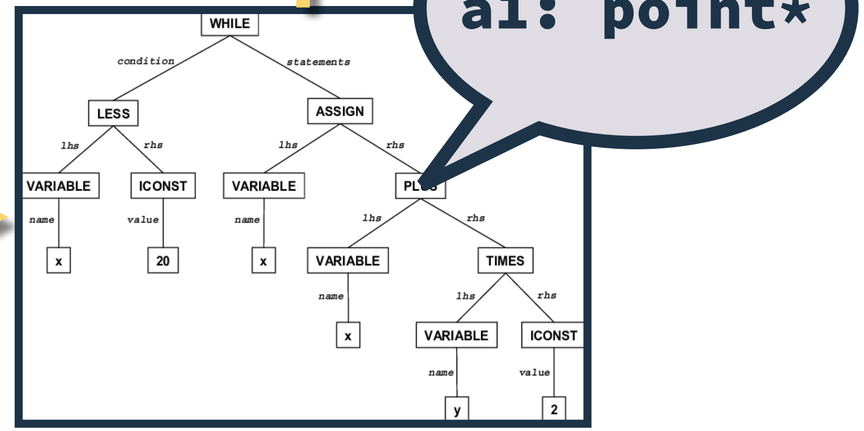
Parse



Tagger

New Output

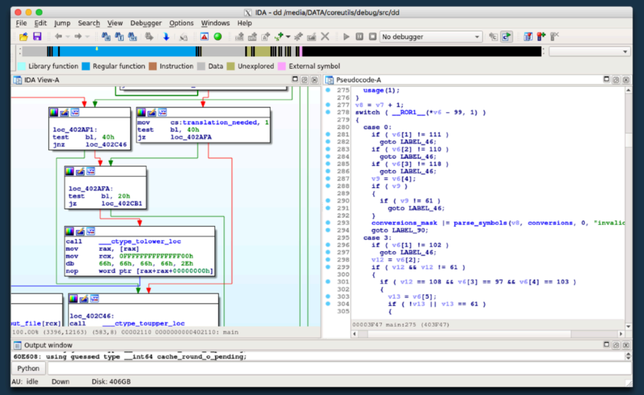
```
double func(point *a1, int *a2) {...}
```



a1: point*

Conclusion

Decompiler



Compilation Loses Information

Comments:

```
/* This is the functionality
 * you're looking for! */
```

Loop Constructs:

```
while (x < 100) {...}

for (i = 0; i < 100; ++i) {...}
```

Variable Names:

```
int width, length;
double volume;
char *user_id;
```

User-Defined Types:

```
typedef struct {
  int x;
  int y;
} point;
```

Type Constraints

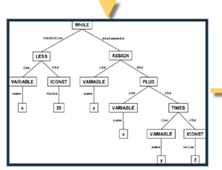
```
typedef struct {
  int x;
  int y;
} point;

sizeof(char);     // 1

sizeof(int);      // 4
sizeof(float);    // 4

sizeof(point);    // 8
sizeof(int[2]);   // 8
```

Strategy

Decompiler Output

```
double func(int *a1, int *a2) {...}
```

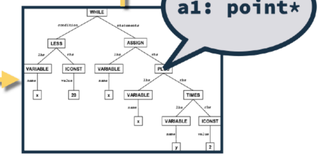
Parse



Tagger

New Output

```
double func(point *a1, int *a2) {...}
```



Contact Me

Jeremy Lacomis

jlacomis@cmu.edu

@jlacomis

jeremylacomis.com