Understanding Misunderstandings in Source Code

Dan Gopstein
J. Iannacone, Y. Yan, L. DeLong,
Y. Zhuang, M. Yeh, J. Cappos

NYU, UCCS, PSU

atomsofconfusion.com

What is confusing?

- goto statements
- Hungarian notation
- Pointers vs References
- Single Entry, Single Exit

Who chose these?
Why do we know they are confusing?

Rob Pike on Pointers

Pointers have a bad reputation in academia, because they are considered too dangerous, dirty somehow. But I think they are powerful notation, which means they can help us express ourselves clearly.

Rob Pike - Notes on Programming in C

Rob Pike on Pointers

Pointers have a bad reputation in academia, because they are considered too dangerous, dirty somehow. But I think they are powerful notation, which means they can help us express ourselves clearly.

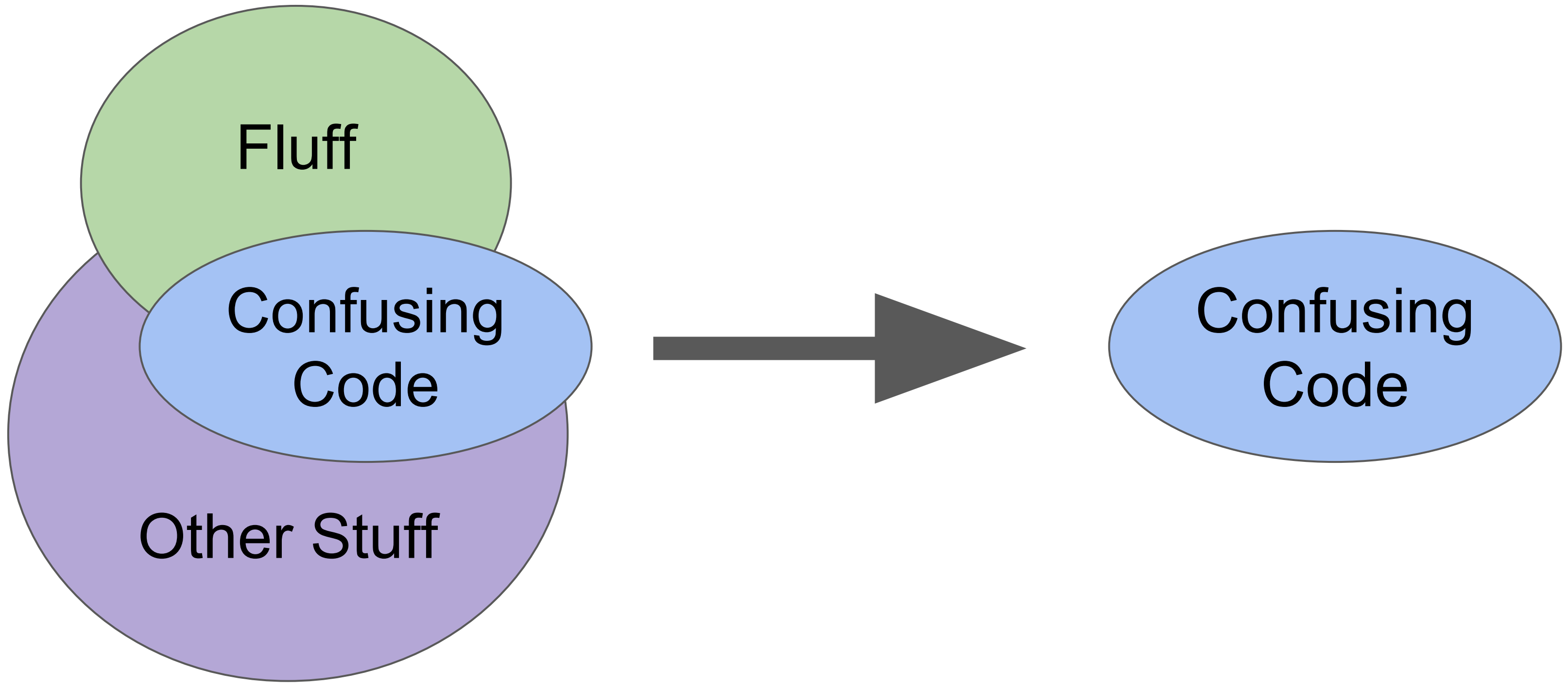
Rob Pike - Notes on Programming in C

Goal

A theory of confusion in software that is objective, rigorous, and empirical.

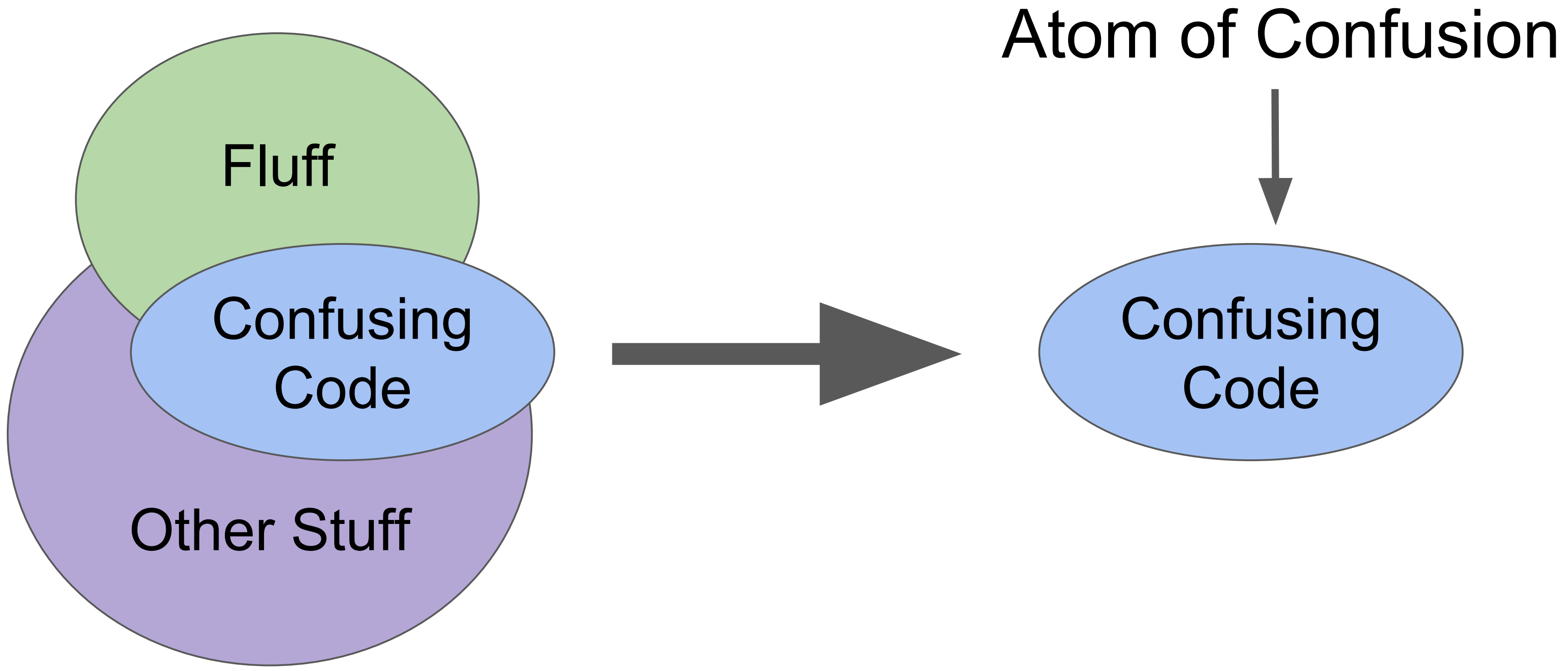
Atom of Confusion

The smallest piece of code that can cause confusion.



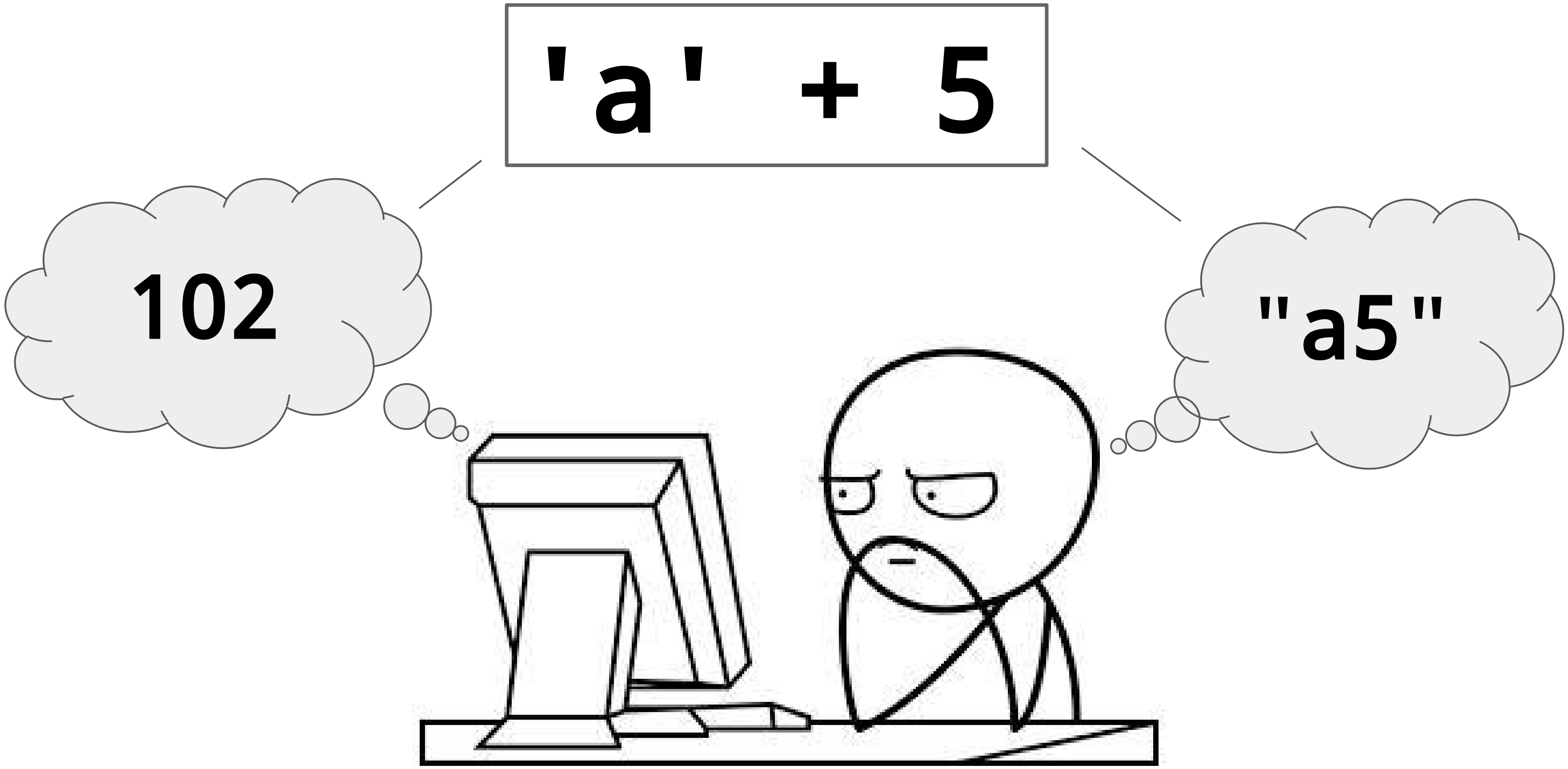
Atom of Confusion

The smallest piece of code that can cause confusion.



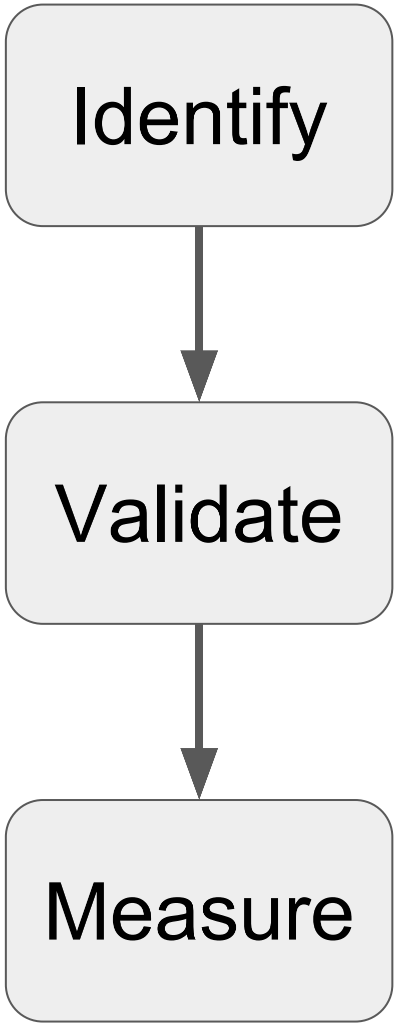
Confusion

When a person and a machine read the same piece of code, yet come to different conclusions about its output.



```
'a' + 5
```

How we objectively identified confusion

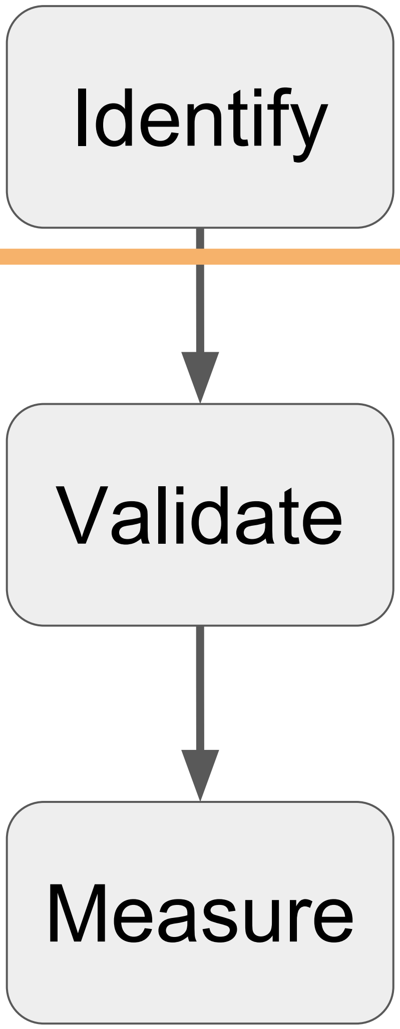


Find potentially confusing patterns

Evaluate whether programmers error while evaluating those patterns

Quantify the effect of removing confusing patterns from larger programs

How we objectively identified confusion



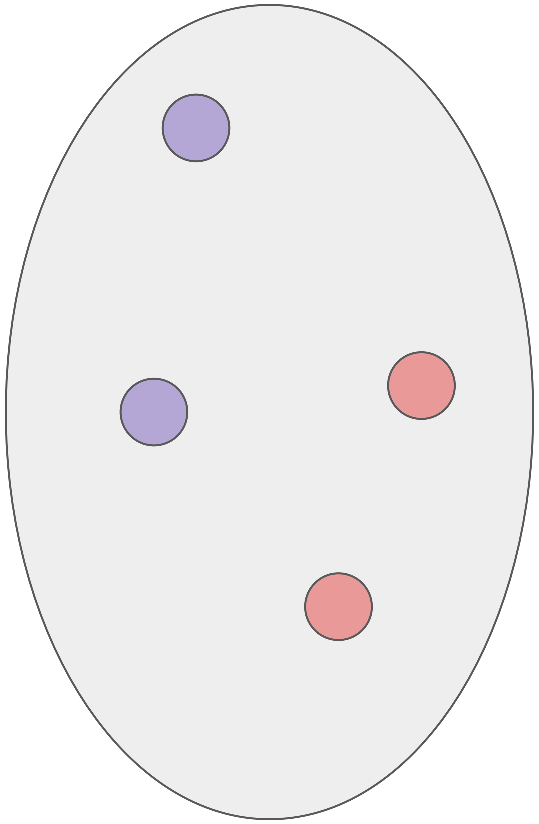
Find potentially confusing patterns

Evaluate whether programmers error while evaluating those patterns

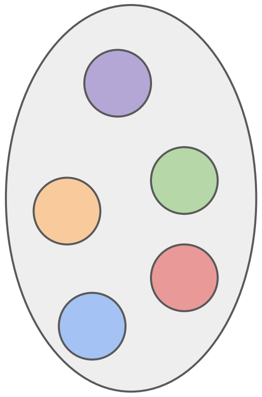
Quantify the effect of removing confusing patterns from larger programs

Comparison of places to look for atom candidates

Sparse and homogenous codebase



Dense and diverse codebase



International Obfuscated C Code Contest (IOCCC)

High density and wide variety of confusing code

```
                                                   extern int
                                                       errno
                                                         ;char
                                                            grrr
                             ;main(                           r,
  argv, argc )            int    argc                           ,
   r        ;           char *argv[];{int                     P( );
#define x  int i,       j,cc[4];printf("      choo choo\n"     ) ;
x  ;if    (P(  !        i              )        |  cc[  !      j ]
&  P(j    )>2  ?        j              :        i  ){*  argv[i++ +!-i]
;              for    (i=              0;;    i++                   );
_exit(argv[argc- 2    / cc[1*argc]|-1<<4 ]    ) ;printf("%d",P(""));}}
  P  (    a  )   char a   ;  {    a  ;   while(    a  >      "  B   "
  /* -    by E            ricM    arsh             all-      */);    }
```

Atom Candidates

Atom	Example
Change of Literal Encoding	`printf("%d", 013)`

Preprocessor in Statement	`int V1 = 1`
	`#define M1 1`
	`+1;`

Assignment as Value	`V1 = V2 = 3;`
Logic as Control Flow	`V1 && F2();`

Macro Operator Precedence	`#define M1 64-1`
	`2*M1`

Post-Increment /Decrement	`V1 = V2++;`
Type Conversion	`(double)(3/2)`

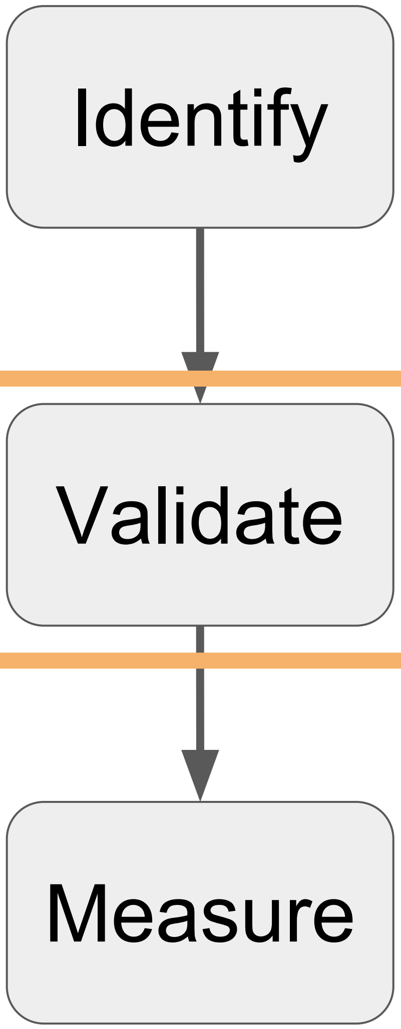
Atom	Example
Reversed Subscripts	`1["abc"]`
Conditional Operator	`V2 = (V1==3)?2:V2`
Comma Operator	`V3 = (V1+=1, V1)`
Pre-Increment /Decrement	`V1 = ++V2;`
Infix Operator Precedence	`0 && 1 || 2`
Omitted Curly Braces	`if (V) F(); G();`
Repurposed Variable	`argc = 7;`
Implicit Predicate	`if (4 % 2)`

Dead, Unreachable, Repeated	`V1 = 1;`
	`V1 = 2;`

Arithmetic as Logic	`(V1-3) * (V2-4)`
Pointer Arithmetic	`"abcdef"+3`

Constant Variables	`int V1 = 5;`
	`printf("%d", V1);`

How we objectively identified confusion



Find potentially confusing patterns

Evaluate whether programmers error while evaluating those patterns

Quantify the effect of removing confusing patterns from larger programs

Atom Removal Transformation

To replace code with functionally equivalent code, with the intent to reduce its level of confusion.

Example snippet question

What does this code output?

```
#define M1 64 - 1

void main(){

   int V1;

   V1 = M1 * 2;

   printf("%d\n", V1);

}
```

Example snippet question

What about this code?

```
void main(){
    int V1;
    V1 = 64 - 1 * 2;
    printf("%d\n", V1);
}
```

Example snippet question

Macro Operator Precedence

With Atom

```
#define M1 64 - 1

void main(){

    int V1;

    V1 = M1 * 2;

    printf("%d\n", V1);

}
```

Without Atom

```
void main(){

    int V1;

    V1 = 64 - 1 * 2;

    printf("%d\n", V1);

}
```

Experiment: Are atom candidates confusing?

- 11 person pilot
- 73 subjects
- 3 examples of each atom candidate
- Partial randomized counterbalanced design
- Analyzed with Durkalski adjusted McNemar test

Results

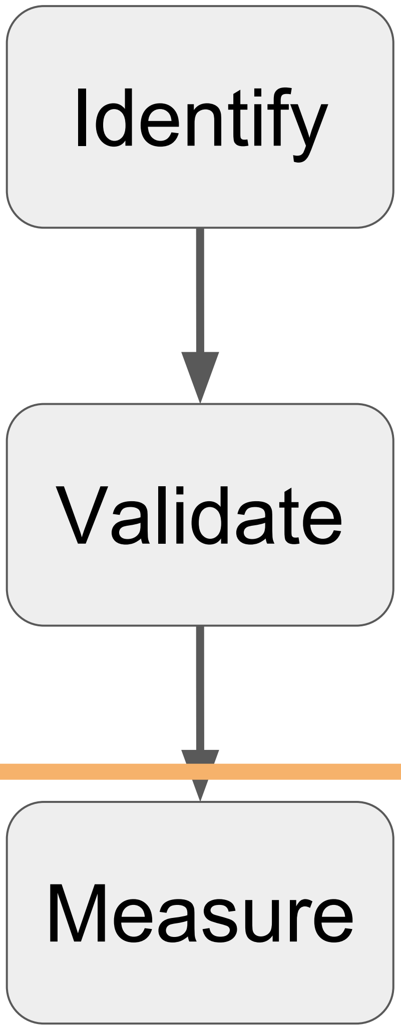
Atom	Effect	p-value
Change of Literal Encoding	0.60	2.93e-14
Preprocessor in Statement	0.47	8.53e-11
Assignment as Value	0.42	3.78e-10
Logic as Control Flow	0.41	5.62e-09
Macro Operator Precedence	0.36	1.77e-07
Post-Increment / Decrement	0.34	6.98e-08
Type Conversion	0.29	5.15e-07
Reversed Subscripts	0.23	1.52e-06

Atom	Effect	p-value
Conditional Operator	0.23	1.74e-05
Comma Operator	0.23	2.46e-04
Pre-Increment / Decrement	0.16	6.89e-04
Infix Operator Precedence	0.14	5.90e-05
Omitted Curly Braces	0.14	8.64e-03
Repurposed Variable	0.12	6.66e-03
Implicit Predicate	0.10	4.27e-03
~~Dead, Unreachable, Repeated~~	~~0.03~~	~~0.059~~
~~Arithmetic as Logic~~	~~0.03~~	~~0.248~~
~~Pointer Arithmetic~~	~~0.01~~	~~0.752~~
~~Constant Variables~~	~~0.00~~	~~1.000~~

Results

	Smallest Effect:	Largest Effect:
	Implicit Predicate	**Change of Literal Encoding**
	Difference in correct responses: 10%	Difference in correct responses: 60%
Atom	`if (4 % 2)`	`printf("%d", 013)`
No Atom	`if ((4 % 2) != 0)`	`printf("%d", 11)`

How we objectively identified confusion



Find potentially confusing patterns

Evaluate whether programmers error while evaluating those patterns

Quantify the effect of removing confusing patterns from larger programs

anonymous.c

First IOCCC winner
1984

```
int i;main(){for(;i["]<i;++i){
--i;}"];read('-'-'-',i+++"hell\
o, world!\n",'/'/'/'));}read(j
,i,p){write(j/p+p,i---j,i/i);}
```

Normalization

```
int i;main(){for(;i["]<i;++i){--i;}"];read('-'-'-',i+++"hell\
o, world!\n",'/'/'/'));}read(j,i,p){write(j/p+p,i---j,i/i);}
```

$\downarrow$

```
#include <stdio.h>
void F1(int V1, char *V2, int V3) {
  printf("a: %d %s %d\n", V1, V2, V3);
  int V4 = V1 / V3 + V3;
  char *V5 = V2-- - V1;
  int V6 = (int)V2 / (int)V2;
  printf("b: %d %s %d\n", V4, V5, V6);
}
int V7;
int main() {
  for (; V7["ab"];
       F1('a' - 'a',
          V7++ + "zy",
          'z' / 'z'))
    ;
  printf("c\n");
}
```

Measure confusion from atoms in bigger programs

Original:

```
int i;main(){for(;i["]<i;++i){--i;}"];read('-'-'-',i+++"hell\
o, world!\n",'/'/'/'));}read(j,i,p){write(j/p+p,i---j,i/i);}
```

Obfuscated

```
#include <stdio.h>
void F1(int V1, char *V2, int V3) {
  printf("a: %d %s %d\n", V1, V2, V3);
  int V4 = V1 / V3 + V3;
  char *V5 = V2-- - V1;
  int V6 = (int)V2 / (int)V2;
  printf("b: %d %s %d\n", V4, V5, V6);
}
int V7;
int main() {
  for (; V7["ab"];
       F1('a' - 'a',
          V7++ + "zy",
          'z' / 'z'))
    ;
  printf("c\n");
}
```

Clarified

```
#include <stdio.h>
void F1(int V1, char *V2, int V3) {
  printf("a: %d %s %d\n", V1, V2, V3);
  int V4 = (V1 / V3) + V3;
  char *V5 = V2 - V1;
  V2 = V2 - 1;
  int V6 = (int)V2 / (int)V2;
  printf("b: %d %s %d\n", V4, V5, V6);
}
int V7;
int main() {
  for (; "ab"[V7] != 0;) {
    F1(97 - 97,
       V7 + "zy",
       122 / 122);
    V7 = V7 + 1;
  }
  printf("c\n");
}
```

Impact Experiment

```
V1/V3+V3     =>   (V1/V3)+V3
```

```
V2--         =>   V2 = V2 - 1
```

```
V7["ab"]     =>   "ab"[V7]
```

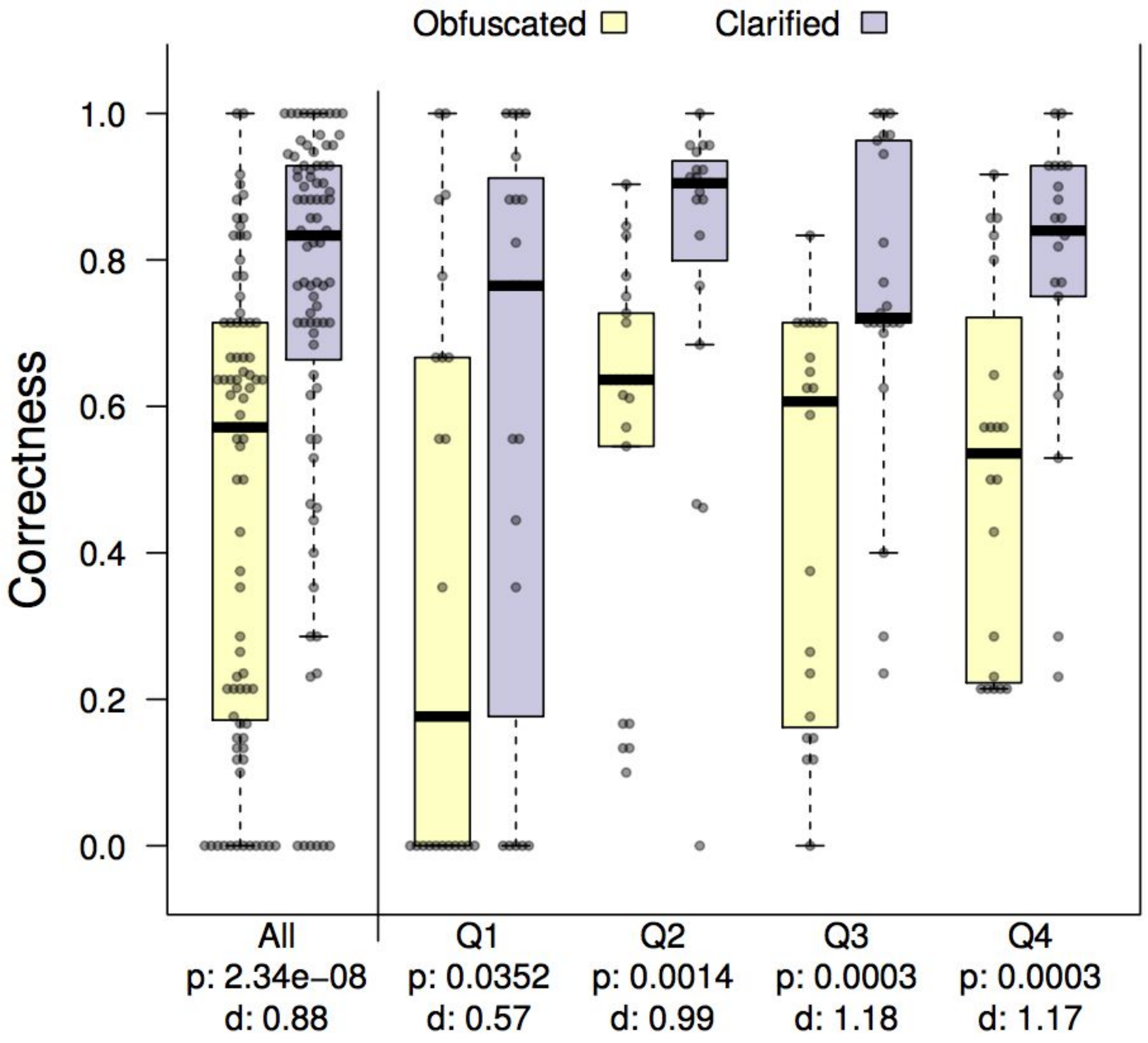
```
"ab"[V7]     =>   "ab"[V7] != 0
```

```
'z'          =>   122
```

Experiment: Impact of removing atoms from program

- 10 person pilot
- 43 subjects
- 4 programs (the normalized IOCCC winner from which atom candidates were derived)
- Partial randomized counterbalanced design
- Analyzed with t-test

Rates of correct output

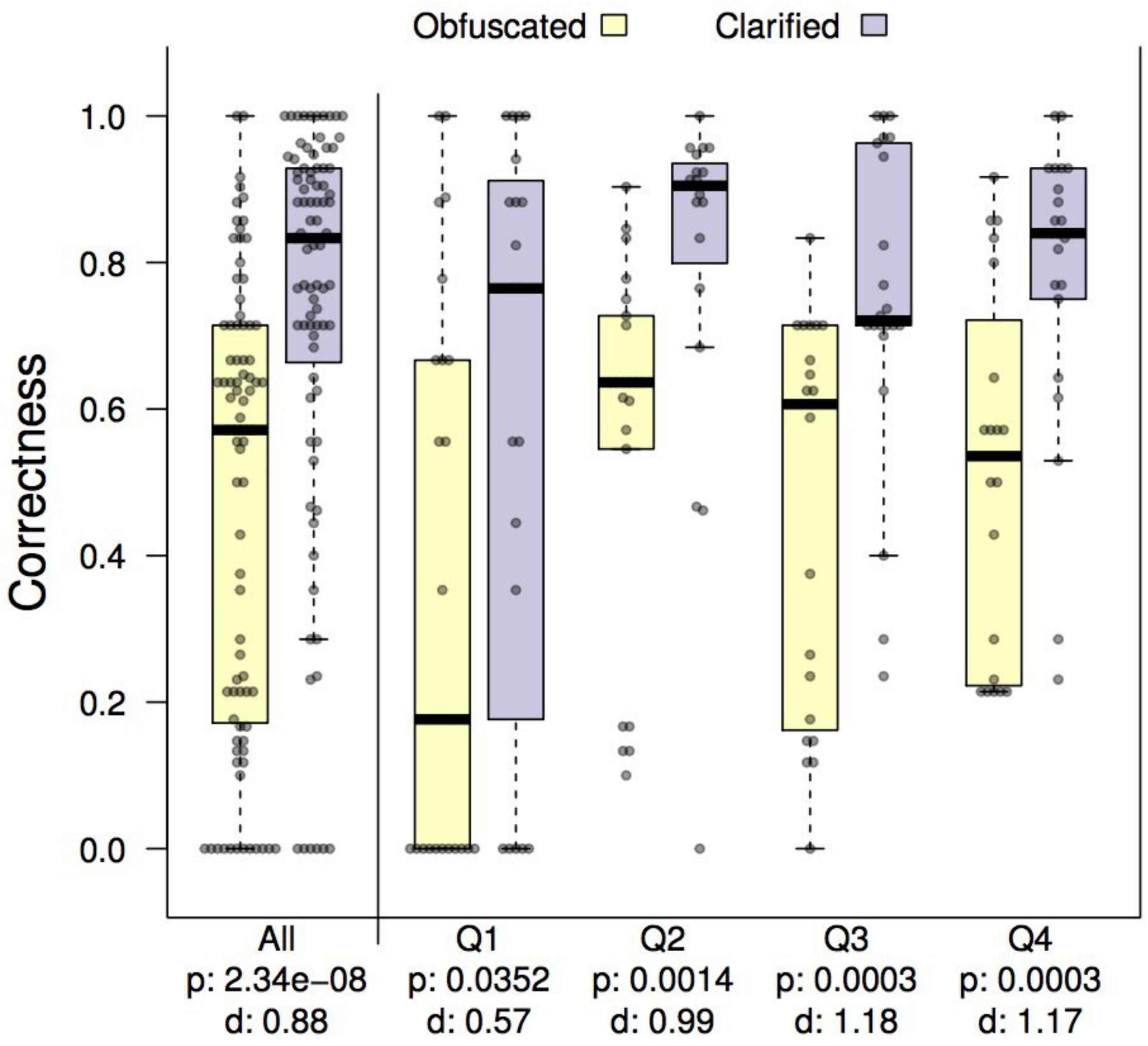


Further positive indicators

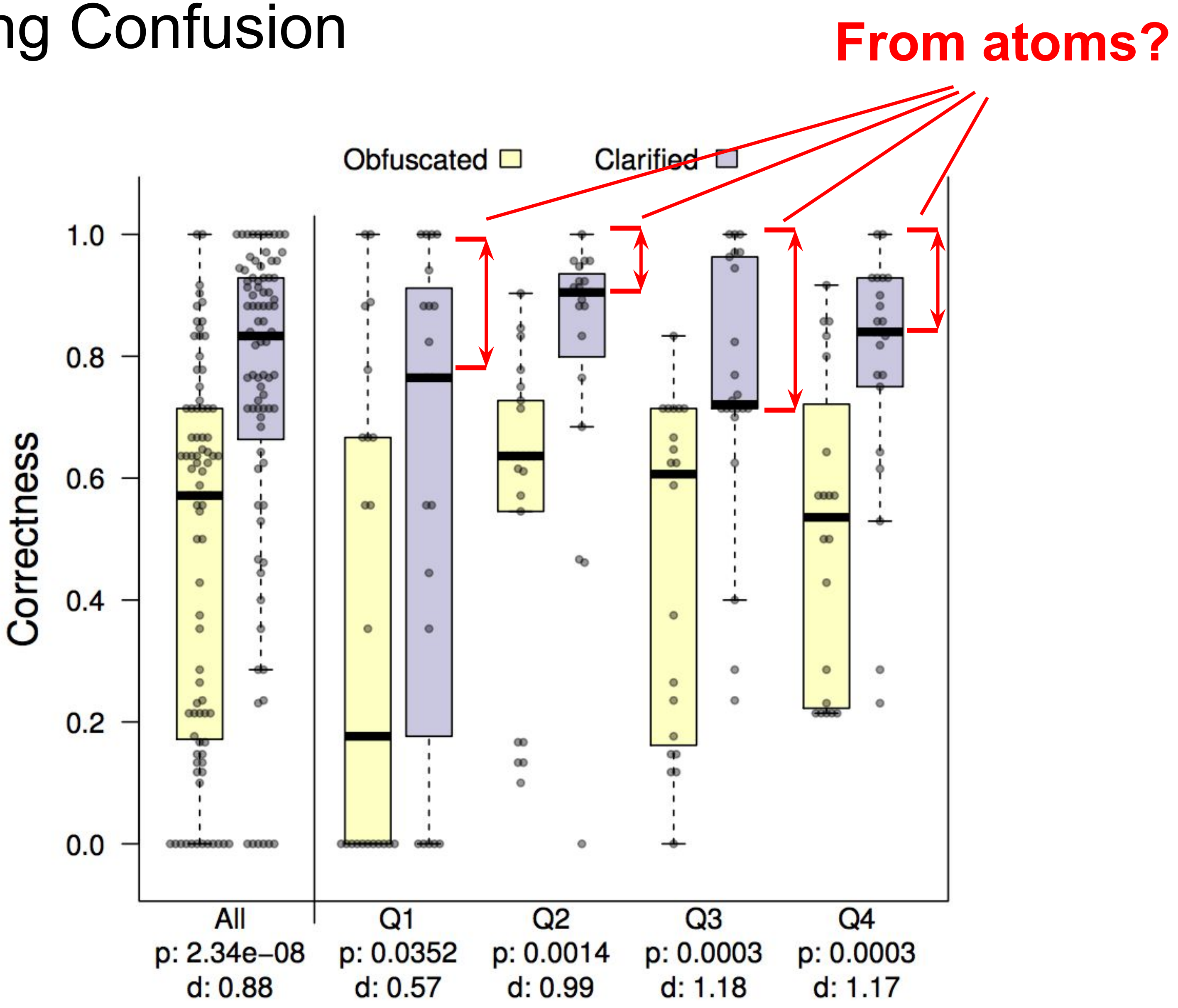
When atoms are removed

- People give up **1/4** as often
- People get lost **1/2** as often
- People write **1/3** more output
- People are **5x** more likely to be totally correct

Remaining Confusion



Remaining Confusion

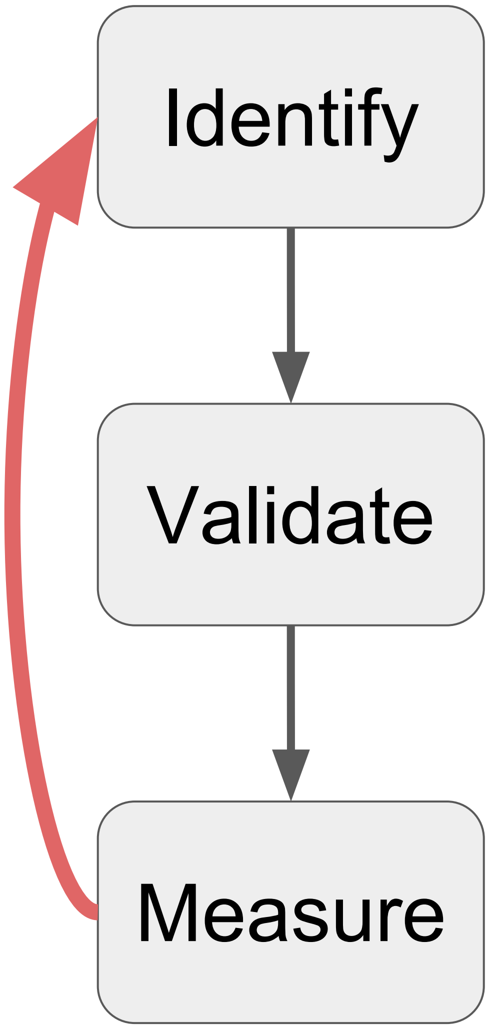


Remaining confusion (identifying false negatives)

What about confusion that remained?

- Static Integer Initialization to 0
- `"ab"[1]`
- `"ab"+1`

Our Work



Find atom candidates

Experiment with isolated snippets

Experiment with original corpus

Style Guides conflicting our findings

- Assignment as Value - **GNU**
- Pointer Arithmetic - **Rob Pike**
- Omitted Curly Braces - **Linux**, **NASA**
- Conditional Operator - **Kernighan and Pike**

GNU Coding Standards:

“Try to avoid assignments inside if-conditions (assignments inside while-conditions are ok).”

```
if (a = 0)
   ...
```

```
while (a = 0)
   ...
```

GNU Coding Standards:

“Try to avoid assignments inside if-conditions (assignments inside while-conditions are ok).”

$\varphi = 0.64$

```
if (a = 0)
  ...
```

$\varphi = 0.52$

```
while (a = 0)
  ...
```

Missing from Style Guides

Preprocessor in Statement

```
if (V1 < V2) {
  #define M1 1
  #define M2 2
}
```

Summary

- A method for quantitatively and objectively measuring misunderstanding of code
  - Extracted patterns from IOCCC winners
  - Objectively validate atom candidates (false positives)
  - Objectively measure impact of atoms in larger programs (false negatives)

- Findings conflict popular style guidelines

- All materials / data available

BOF

tonight @ 17:45

Room F0.530

- add to the dataset
- debate rigorous methodologies for creating such datasets
- discuss appropriate ways to analyze the dataset
- help to guide future data collection efforts
- get a head start on your own analysis using the data

All are welcome!

Thank You

Understanding Misunderstandings in Source Code

Dan Gopstein
J. Iannacone, Y. Yan, L. DeLong,
Y. Zhuang, M. Yeh, J. Cappos

NYU, UCCS, PSU

atomsofconfusion.com