Let Me Do Your Homework For You

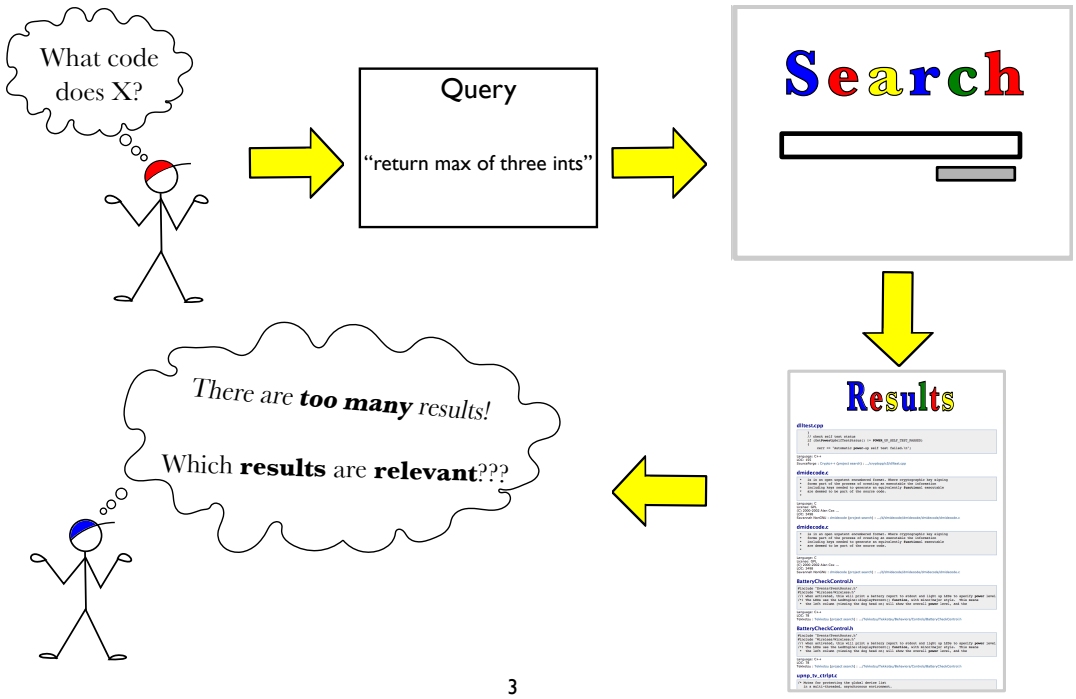
Kathryn T. Stolee
Assistant Professor of Software Engineering
Iowa State University

September 5, 2013

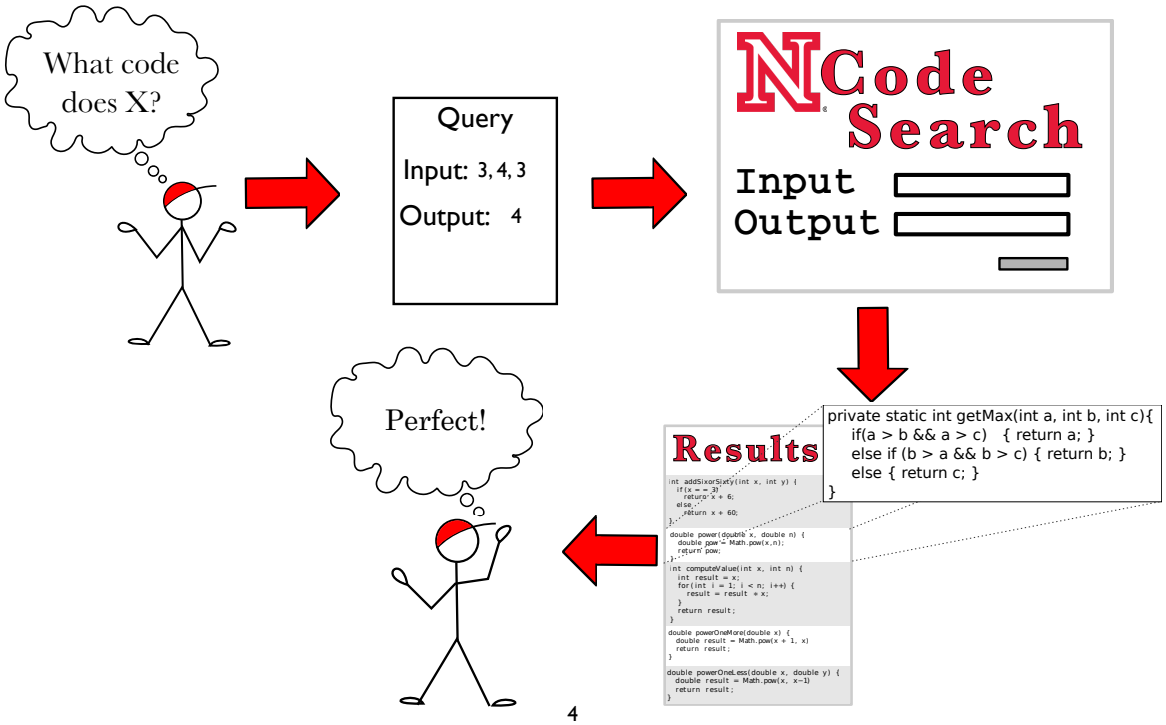
Homework Assignment

- Write a Java method that returns the maximum element among three integers

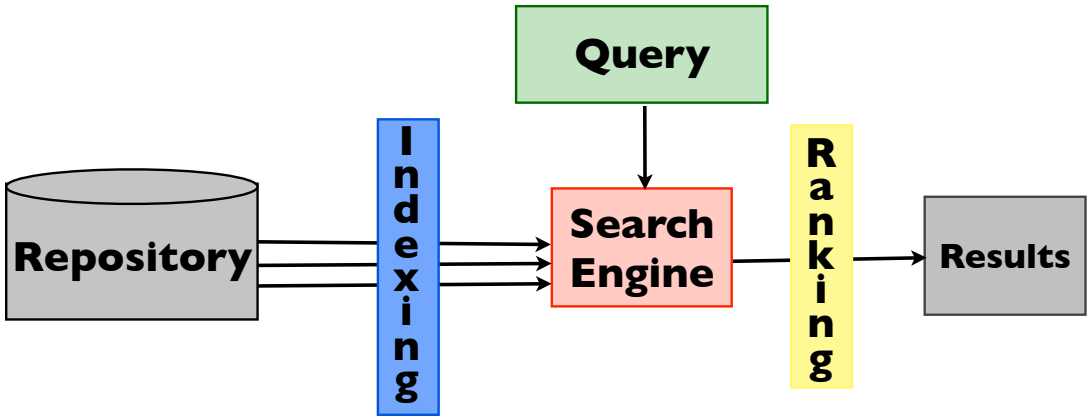
What Some People Do



A Better World...



How Does Search Work?



About Search

	Google	Satsy
Querying		
Indexing		
Matching		
Ranking		

About Search

	Google	Satsy
Querying	keywords	input/output example
Indexing		
Matching		
Ranking		

Working Example

- Write a Java method that returns the maximum element among three integers
  - Google Search: *"Java return maximum of three integers"*
  - Satsy:
    - Input: *3, 4, 3*
    - Output: *4*

About Search

	Google	Satsy
Querying	keywords	input/output example
Indexing		
Matching		
Ranking		

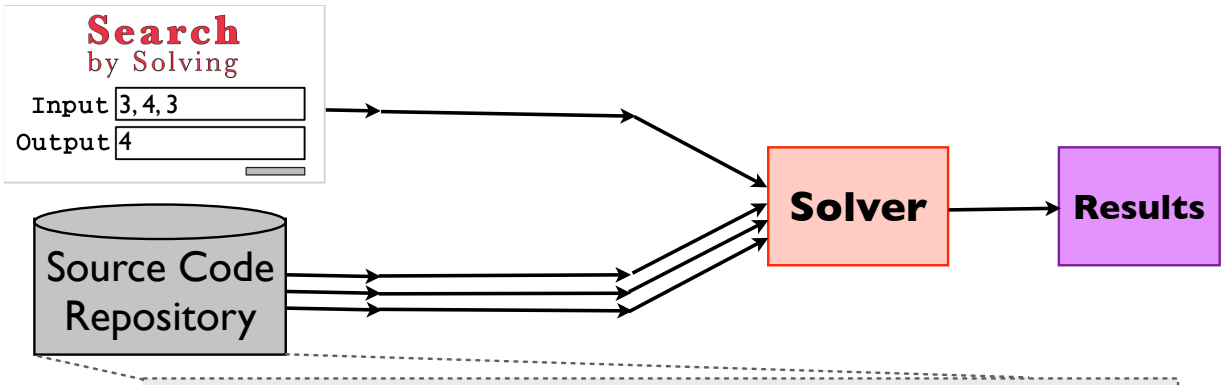
About Search

	Google	Satsy
Querying	keywords	input/output example
Indexing	words and their locations in web pages	code behavior
Matching		
Ranking		

About Search

	Google	Satsy
Querying	keywords	input/output example
Indexing	words and their locations in web pages	code behavior
Matching	text matching	Solver
Ranking		

Satsy: Behind the Scenes



Code Snippets:

```
private int getsum(int a, int b, int c){
  return a + b + c;
}

private boolean allPositive(int x, int y, int z){
  return x >= 0 && y >= 0 && z >= 0;
}

private boolean areEqual(String s, String t) {
  return s.toLowerCase().equals(t.toLowerCase());
}
```

```
private int getMax(int d, int e, int f){
  if(d > e && d > f) {
    return d;
  }else if (e > d && e > f) {
    return e;
  }else {
    return f;
  }
}
```

SMT Solvers

Satisfiability **M**odulo **T**heory solvers determine if a logical formula is satisfiable

Facts	Assertions
`a >= 0`	`(assert (>= a 0))`
`b = 2`	`(assert (= b 2))`
`c = 2`	`(assert (= c 2))`
`c = a * b`	`(assert (= ( * a b) c))`

Result: **sat** $a \mapsto 1$

SMT Solvers

Satisfiability **M**odulo **T**heory solvers determine if a logical formula is satisfiable

Facts	Assertions
`a >= 0`	`(assert (>= a 0))`
`b = ?`	**`(assert (= b ?))`**
`c = 2`	`(assert (= c 2))`
`c = a * b`	`(assert (= ( * a b) c))`

Result: **sat** $a \mapsto \mathbb{2} \wedge b \mapsto \mathbb{2}$

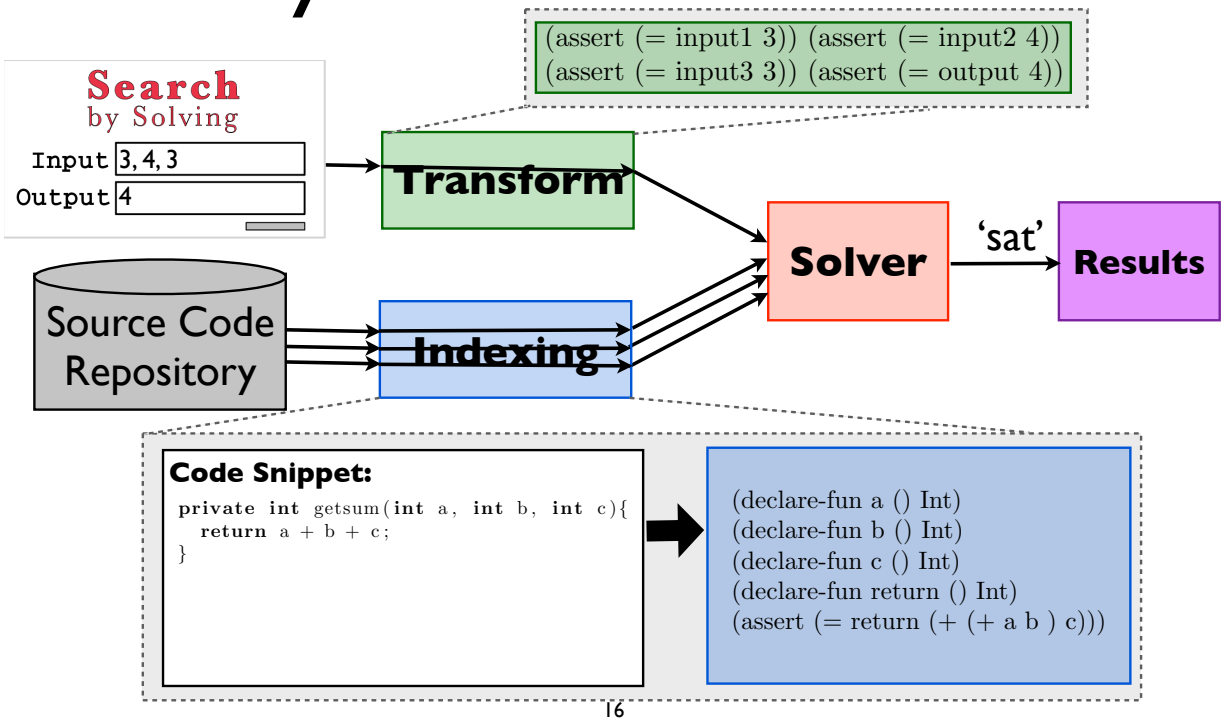
SMT Solvers

Satisfiability **M**odulo **T**heory solvers determine if a logical formula is satisfiable

Facts	Assertions
`a = 0`	`(assert (= a 0))`
`b = ?`	`(assert (= b ?))`
`c = 2`	`(assert (= c 2))`
`c = a * b`	`(assert (= ( * a b) c))`

Result: **unsat**

Satsy: Behind the Scenes



About Search

	Google	Satsy
Querying	keywords	input/output example
Indexing	words and their locations in web pages	code behavior
Matching	text matching	Solver
Ranking		

Indexing

Code Snippet:

```
private int getsum(int a, int b, int c){
  return a + b + c;
}
```

```
(declare-fun a () Int)
(declare-fun b () Int)
(declare-fun c () Int)
(declare-fun return () Int)
(assert (= return (+ (+ a b ) c)))
```

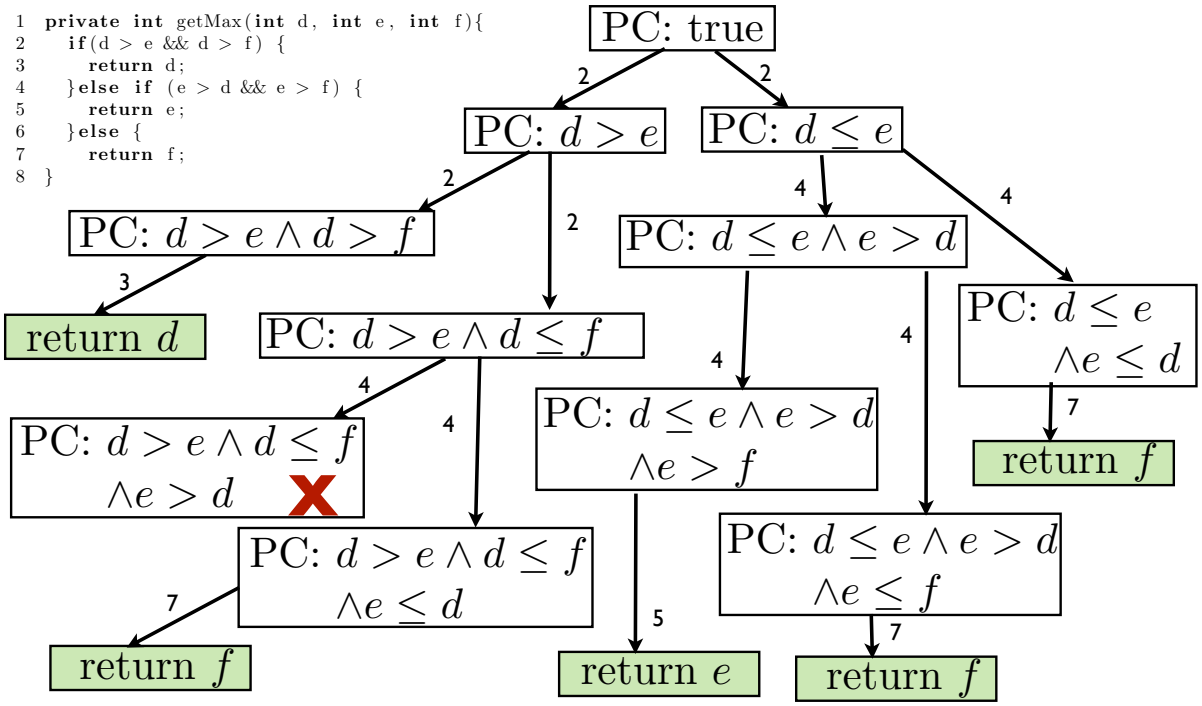
Code Snippet:

```
private int getMax(int d, int e, int f){
  if(d > e && d > f) {
    return d;
  }else if (e > d && e > f) {
    return e;
  }else {
    return f;
}
```

```
(declare-fun d () Int)
(declare-fun e () Int)
(declare-fun f () Int)
(declare-fun return () Int)
(assert (or
  (and (> d e) (> d f) (= return d))
  (and (> d e) (≤ d f) (≤ e d) (= return f))
  (and (≤ d e) (> e d) (> e f) (= return e))
  (and (≤ d e) (> e d) (≤ e f) (= return f))
  (and (≤ d e) (≤ e d) (= return f))))
```

Symbolic Execution

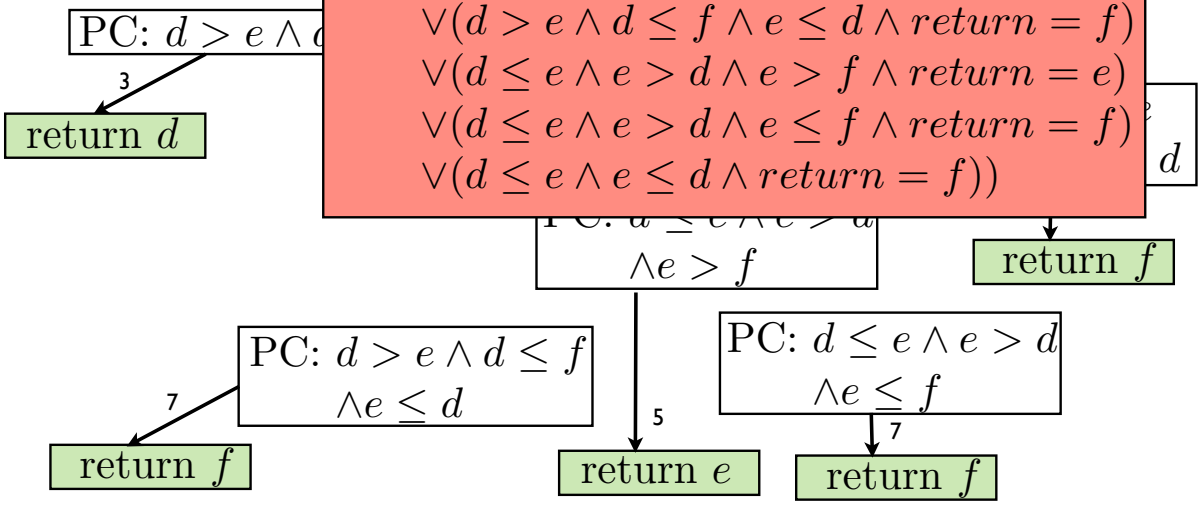
```
1 private int getMax(int d, int e, int f){
2   if(d > e && d > f) {
3     return d;
4   }else if (e > d && e > f) {
5     return e;
6   }else {
7     return f;
8 }
```



Symbolic Execution

```
1 private int getMax(int d, int e, int f){
2   if(d > e && d > f) {
3     return d;
4   }else if (e > d && e > f) {
5     return e;
6   }else {
7     return f;
8 }
```

$$P_{enc} = ((d > e \wedge d > f \wedge return = d) \vee (d > e \wedge d \leq f \wedge e \leq d \wedge return = f) \vee (d \leq e \wedge e > d \wedge e > f \wedge return = e) \vee (d \leq e \wedge e > d \wedge e \leq f \wedge return = f) \vee (d \leq e \wedge e \leq d \wedge return = f))$$



About Search

	Google	Satsy
Querying	keywords	input/output example
Indexing	words and their locations in web pages	code behavior
Matching	text matching	Solver
Ranking		

About Search

	Google	Satsy
Querying	keywords	input/output example
Indexing	words and their locations in web pages	code behavior
Matching	text matching	Solver
Ranking		

Matching

```
private int getsum(int a, int b, int c){
  return a + b + c;
}
```

```
(declare-fun a () Int)
(declare-fun b () Int)
(declare-fun c () Int)
(declare-fun return () Int)
(assert (= return (+ (+ a b ) c)))
```

Input	Output	Mapping
3, 4, 3	4	$a \mapsto 3 \wedge b \mapsto 4 \wedge c \mapsto 3 \wedge return \mapsto 4$
3, 0, 0	3	$a \mapsto 3 \wedge b \mapsto 0 \wedge c \mapsto 0 \wedge return \mapsto 3$

Result: sat

Matching

```
1 private int getMax(int d, int e, int f){
2   if(d > e && d > f) {
3     return d;
4   }else if (e > d && e > f) {
5     return e;
6   }else {
7     return f;
8 }
```

PC1: $d > e \wedge d > f \wedge return = d$
PC2: $d > e \wedge d \leq f \wedge e \leq d \wedge return = f$
PC3: $d \leq e \wedge e > d \wedge e > f \wedge return = e$
PC4: $d \leq e \wedge e > d \wedge e \leq f \wedge return = f$
PC5: $d \leq e \wedge e \leq d \wedge return = f$

Input	Output	Mapping
3, 4, 3	4	$d \mapsto 3 \wedge e \mapsto 4 \wedge f \mapsto 3 \wedge return \mapsto 4$

Result: **sat for PC3**

Matching

```
1 private int getMax(int d, int e, int f){
2   if(d > e && d > f) {
3     return d;
4   }else if (e > d && e > f) {
5     return e;
6   }else {
7     return f;
8 }
```

PC1: $d > e \wedge d > f \wedge return = d$
PC2: $d > e \wedge d \leq f \wedge e \leq d \wedge return = f$
PC3: $d \leq e \wedge e > d \wedge e > f \wedge return = e$
PC4: $d \leq e \wedge e > d \wedge e \leq f \wedge return = f$
PC5: $d \leq e \wedge e \leq d \wedge return = f$

Input	Output	Mapping
3, 4, 3	4	$d \mapsto 3 \wedge e \mapsto 4 \wedge f \mapsto 3 \wedge return \mapsto 4$
3, 0, 0	3	$d \mapsto 3 \wedge e \mapsto 0 \wedge f \mapsto 0 \wedge return \mapsto 3$

Result: **sat for PC3, PC1**

Matching

```
1 private int getMax(int d, int e, int f){
2   if(d > e && d > f) {
3     return d;
4   }else if (e > d && e > f) {
5     return e;
6   }else {
7     return f;
8 }
```

PC1: $d > e \wedge d > f \wedge return = d$
PC2: $d > e \wedge d \leq f \wedge e \leq d \wedge return = f$
PC3: $d \leq e \wedge e > d \wedge e > f \wedge return = e$
PC4: $d \leq e \wedge e > d \wedge e \leq f \wedge return = f$
PC5: $d \leq e \wedge e \leq d \wedge return = f$

Input	Output	Mapping
3, 4, 3	4	$d \mapsto 3 \wedge e \mapsto 4 \wedge f \mapsto 3 \wedge return \mapsto 4$
3, 0, 0	3	$d \mapsto 3 \wedge e \mapsto 0 \wedge f \mapsto 0 \wedge return \mapsto 3$
5, 5, 4	5	$d \mapsto 5 \wedge e \mapsto 5 \wedge f \mapsto 4 \wedge return \mapsto 5$

Result: **sat for PC3, PC1**

Why Not Execute the Programs?

```
1 private String getPart(String s, char c) {
2   int index = s.lastIndexOf(c);
3   return s.substring(0, index);
4 }
```

Input	Output	Result
“log.txt”	“log”	sat, $c \mapsto$ ‘.’

About Search

	Google	Satsy
Querying	keywords	input/output example
Indexing	words and their locations in web pages	code behavior
Matching	text matching	Solver
Ranking		

About Search

	Google	Satsy
Querying	keywords	input/output example
Indexing	words and their locations in web pages	code behavior
Matching	text matching	Solver
Ranking	PageRank algorithm	Partial Matching

Ranking with Partial Matches

Let P be a program
Let Q_P be the paths in a program
Let LS be a set of input/output examples
Let LS_{sat} be the set of input/output examples satisfied by a path in Q_P
Let Q_{sat} be the set of paths in a program that satisfy some $ls \in LS$

	$\|LS_{sat}\| = \varnothing$	$LS_{sat} \subset LS$	$LS_{sat} = LS$
$\|Q_{sat}\| = \varnothing$	–	–	–
$Q_{sat} \subset Q_P$	–	Under/Over	Over-Approx
$Q_{sat} = Q_P$	–	Under-Approx	Full Match

Ranking with Partial Matches

```
private int getsum(int a, int b, int c){
  return a + b + c;
}
```

Input	Output	Match
3, 4, 3	4	--
3, 0, 0	3	Under-Approx

	$\|LS_{sat}\| = \varnothing$	$LS_{sat} \subset LS$	$LS_{sat} = LS$
$\|Q_{sat}\| = \varnothing$	–	–	–
$Q_{sat} \subset Q_P$	–	Under/Over	Over-Approx
$Q_{sat} = Q_P$	–	Under-Approx	Full Match

Ranking with Partial Matches

```
1 private int getMax(int d, int e, int f){
2   if(d > e && d > f) {
3     return d;
4   }else if (e > d && e > f) {
5     return e;
6   }else {
7     return f;
8 }
```

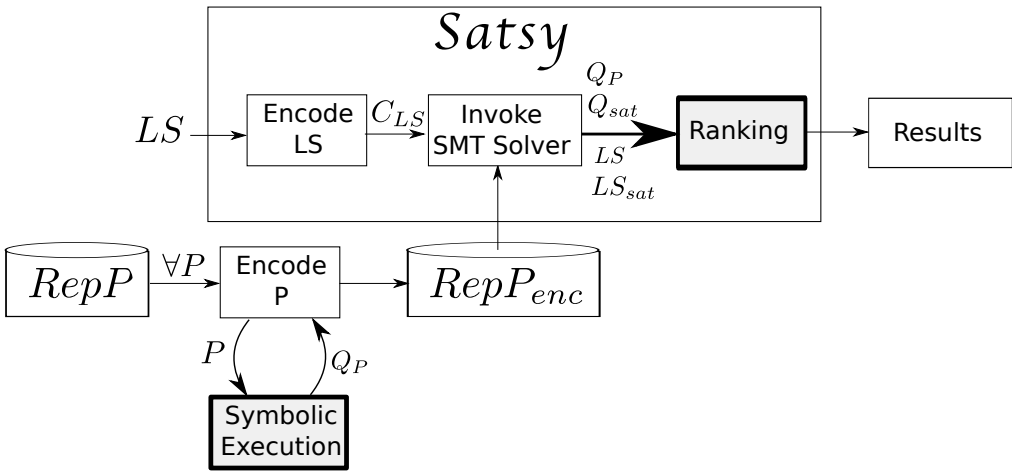
Input	Output	Path	Match
3, 4, 3	4	PC3	Over-Approx
3, 0, 0	3	PC1	Over-Approx
5, 5, 4	5	---	Under/Over-Approx

	$\|LS_{sat}\| = \varnothing$	$LS_{sat} \subset LS$	$LS_{sat} = LS$
$\|Q_{sat}\| = \varnothing$	–	–	–
$Q_{sat} \subset Q_P$	–	Under/Over	Over-Approx
$Q_{sat} = Q_P$	–	Under-Approx	Full Match

About Search

	Google	Satsy
Querying	keywords	input/output example
Indexing	words and their locations in web pages	code behavior
Matching	text matching	Solver
Ranking	PageRank algorithm	Partial Matching

Implementation



Language Support

⟨*statements*⟩ ::= (⟨*statement*⟩)*

⟨*statement*⟩ ::= ⟨*variableDeclaration*⟩ | ⟨*assignment*⟩ | ⟨*ifStatement*⟩ | ⟨*return*⟩

⟨*variableDeclaration*⟩ ::= ⟨*type*⟩ ⟨*ident*⟩ ';'

⟨*ifStatement*⟩ ::= 'if' '(' ⟨*expr*⟩ ')' '{' ⟨*statements*⟩ '}'
('else' 'if' '(' ⟨*expr*⟩ ')' '{' ⟨*statements*⟩ '}')*
('else' '{' ⟨*statements*⟩ '}')?

⟨*assignment*⟩ ::= ⟨*type*⟩? ⟨*ident*⟩ '=' ⟨*expr*⟩ ';'

⟨*return*⟩ ::= 'return' ⟨*expr*⟩ ';'

⟨*expr*⟩ ::= ⟨*logicalOrExpr*⟩

⟨*logicalOrExpr*⟩ ::= ⟨*logicalAndExpr*⟩ ('||' ⟨*logicalAndExpr*⟩)*

⟨*logicalAndExpr*⟩ ::= ⟨*equalityExpr*⟩ ('&&' ⟨*equalityExpr*⟩)?

⟨*equalityExpr*⟩ ::= ⟨*relationalExpr*⟩ (('!=' | '==') ⟨*relationalExpr*⟩)?

⟨*relationalExpr*⟩ ::= ⟨*addExpr*⟩ (('<' | '<=' | '>' | '>=') ⟨*addExpr*⟩)?

⟨*addExpr*⟩ ::= ⟨*multExpr*⟩ (('+' | '-') ⟨*multExpr*⟩)?

⟨*multExpr*⟩ ::= ⟨*primaryExpr*⟩ (('*' | '/' | '%') ⟨*primaryExpr*⟩)?

⟨*primaryExpr*⟩ ::= ⟨*invokeExpr*⟩ | ⟨*stringLit*⟩ | ⟨*charLit*⟩ | ⟨*integer*⟩ | ⟨*booleanLit*⟩
| '(' ⟨*expr*⟩ ')' | ⟨*ident*⟩

⟨*invokeExpr*⟩ ::= (⟨*ident*⟩ | ⟨*stringLit*⟩) (('.' ⟨*noarg*⟩ '(' ')')
| (⟨*onearg*⟩ '(' ⟨*expr*⟩ ')')
| (⟨*twoarg*⟩ '(' ⟨*expr*⟩ (',' ⟨*expr*⟩)? ')'))+

⟨*noarg*⟩ ::= 'length' | 'toLowerCase' | 'toUpperCase' | 'trim'

⟨*onearg*⟩ ::= 'charAt' | 'concat' | 'contains' | 'endsWith' | 'equals' | 'equalsIgnoreCase'
| 'indexOf' | 'lastIndexOf' | 'startsWith' | 'substring'

⟨*twoarg*⟩ ::= 'indexOf' | 'lastIndexOf' | 'replace'

⟨*type*⟩ ::= 'char' | 'String' | 'int' | 'boolean'

⟨*booleanLit*⟩ ::= 'true' | 'false'

⟨*integer*⟩ ::= ('0' | ('1'..'9') ('0'..'9')*);

- Next:
  - Loops
  - Nulls
  - Concurrency
  - Objects

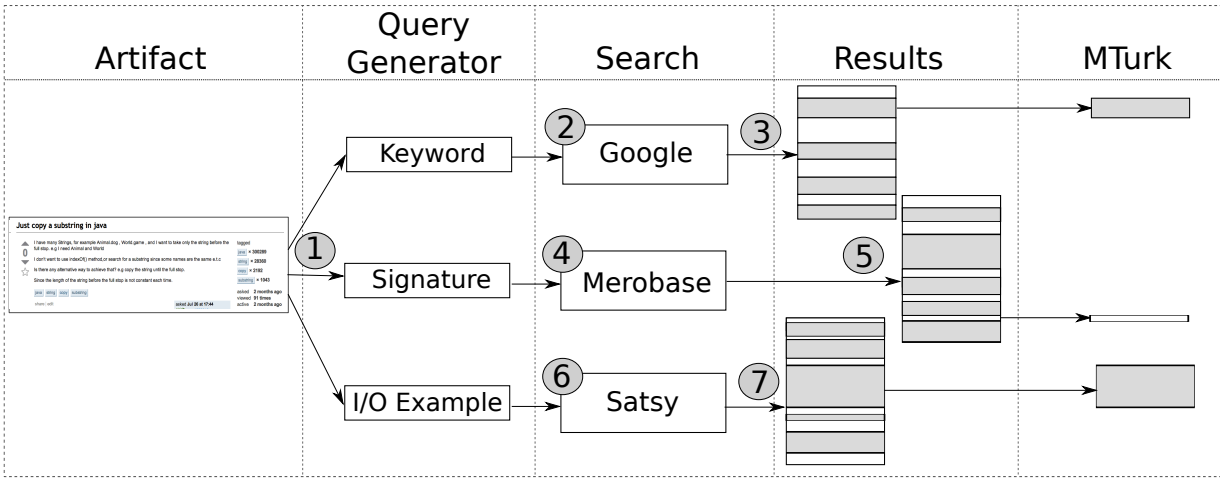
Evaluation

How do search results from **Satsy** compare to those found using **Google** and those found with the code-specific search engine, **Merobase**, from the *perspective of a programmer*?

Satsy Repository

- 8,430 methods scraped from GitHub
- 9,909 paths among the methods

Evaluation



8 artifacts * 3 query generators per artifact * 3 algorithms = 720 Results

Artifacts

Q	Description
1	Check if one string contains another, case insensitive
2	Capitalize the first letter of a string
3	Determine if a number is positive
4	Trim the file extension from a file name
5	Trim the last character from a string
6	Turn a string into a char
7	Determine if a character is numeric
8	Check if one string is a rotation of another (a rotation is when the first part of a string is spliced off and tacked onto the end)

Mechanical Turk Task

Programming Task: **Given a string, trim off the last character.**

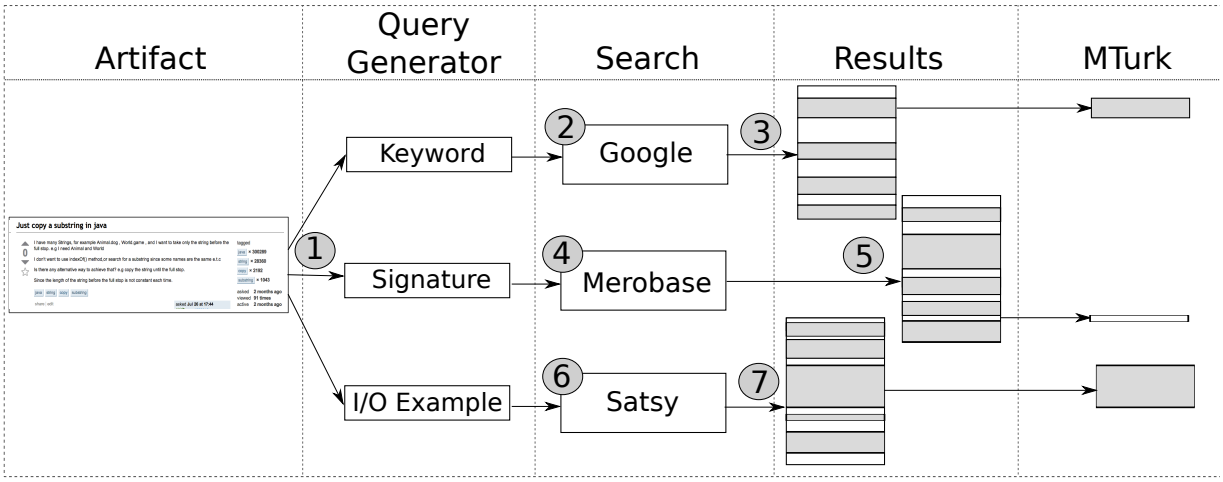
Code Snippet 1: Consider the following Java code:

```
boolean isRotation(String s1,String s2) {
return (s1.length() == s2.length()) && ((s1+s1).indexOf(s2) != -1);
}
```

1. Is this code *relevant* to the programming task (relevance means the source code can be easily adapted to solve the problem)?
   ◯ Yes ◯ No

   Why or why not? How could it be adapted? (requires 10+ word response)

Evaluation



Metric: P@10, relevance among top 10 results

Results

	P@10
Google	0.633
Merobase	0.375
Satsy	0.533

Impact of Specification Size on Relevance

	Number of Input/Output Pairs			
	1	2	3	4
Satsy	0.500	0.483	0.627	0.663
			Google	0.633

Ranking Revisited

	$\|LS_{sat}\| = \varnothing$	$LS_{sat} \subset LS$	$LS_{sat} = LS$
$\|Q_{sat}\| = \varnothing$	–	–	–
$Q_{sat} \subset Q_P$	–	Under/Over	Over-Approx
$Q_{sat} = Q_P$	–	Under-Approx	Full Match

Original Ranking: Full, Over, Under, Under/Over

Proposed Ranking: Over, Full, Under, Under/Over

Hypothetical Ranking

	P@10
Google	0.633
Merobase	0.375
Satsy	0.533
Satsy'	*0.653*

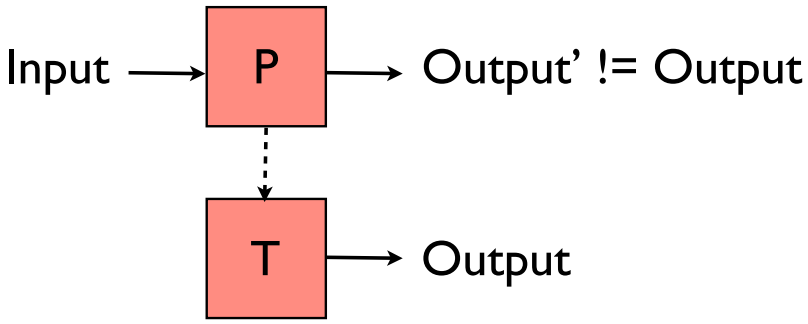
Conclusion

- More input/output examples lead to better results
- With smart ranking, Satsy could beat Google

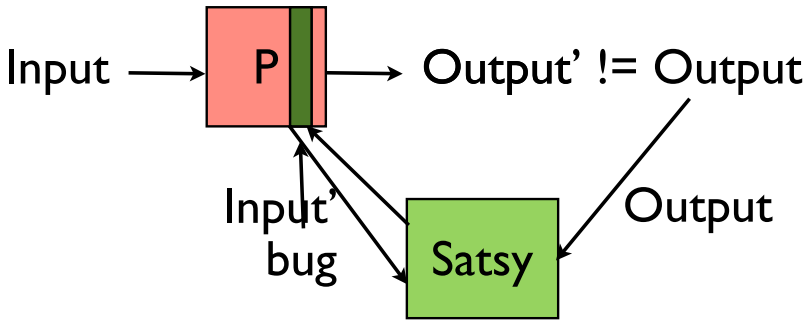
Where do we go from here?

- Increase language coverage
- Revise and evaluate ranking algorithms
- UI to facilitate user studies
  (senior design team, I'm talking to you!)
- Many potential directions.....

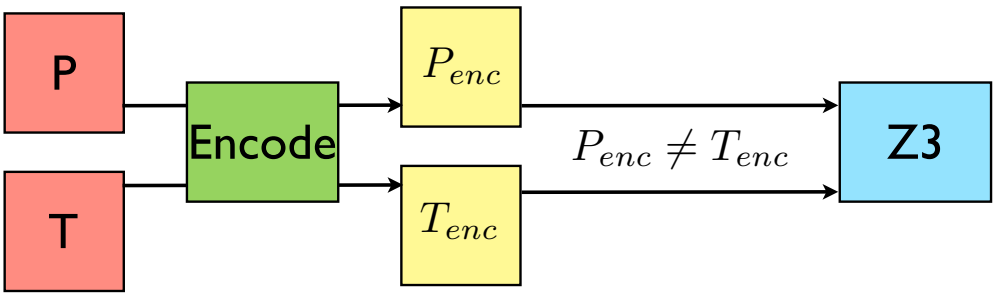
Program Synthesis



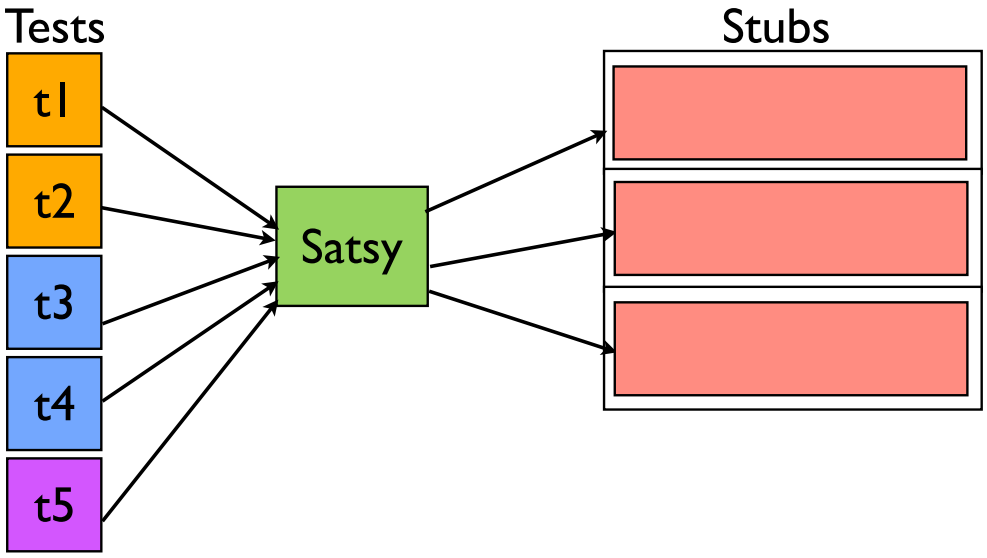
Automated Fault Fixing



Clone Refactoring



Test-Driven Development



Summary

- Introduced a code search approach using input/output
- Defined a notion of partial matching for ranking programs
- Evaluated the approach against Google and Merobase
- Discussed potential applications

Homework Assignment

- W
  ma
  ers

Satsy Can Help!

Acknowledgements

- Collaborators: Sebastian Elbaum and Matt Dwyer
- Funding:
  - Iowa State University
  - NSF SHF-1218265
  - NSF GRFP