Introduction to Program Synthesis

Woosuk Lee

CSE9116

Hanyang University

Instructor

- Assistant professor @ Hanyang univ.
  (2018.9 ~ present)

- Postdoctoral researcher @ UPenn
  (2017.1 ~ 2018.8)
  (Advisor: Mayur Naik)

- Postdoctoral researcher @ Georgia Tech
  (2016.3 ~ 2017.1)
  (Advisor: Mayur Naik)

- Ph.D., Seoul Nat'l Univ. (2016. 2)
  (Advisor: Kwangkeun Yi)

Course Information

- CSE 9116 Program Synthesis

- Course website: http://psl.hanyang.ac.kr/courses/cse9116_2024s/

- Time & Place
  - Monday 14:00 - 16:00
  - Rm 416, 4th Eng Bldg.

- Office hours: Wednesday 10:00-12:00 @ Rm 403, 3rd Eng Bldg.

Evaluation

- Attendance 10%
- Student presentation 30%
  - 2 papers from the reading material
- Project 60%
  - Project proposal & final presentation
  - From project ideas or your own idea

Program Synthesis

- Automatically finding programs satisfying user intent

In form of constraints (e.g., input-output examples)

Specification

Program

- First mentioned by Alan Turing in his book *'Proposed Electronic Calculator'* (1945)

> *Instruction tables will have to be made up by mathematicians with computing experience and perhaps a certain puzzle-solving ability… This process of constructing instruction tables should be very fascinating. There need be no real danger of it ever becoming a drudge, for any processes that are quite mechanical may be turned over to the machine itself.* [1]

The holy grail of Computer Science

History

- 1950s: Synthesis: Compilers for *high-level languages* (e.g., Fortran)
- 1960 - 70s: logical formula ⇒ program (by transformation rules)
- 1970 - 90s: input-output examples ⇒ programs (programming-by-example, PBE)
  - Rather small LISP programs
  - Enumeration with heuristic, genetic programming
  - Bring inductive machine learning techniques into PBE

J. W. Backus et al., *The FORTRAN Automatic Coding System*, 1957

R. M. Burstall, John Darlington, *A Transformation System for Developing Recursive Programs*,

Z. Manna, R. Waldinger, Synthesis: Dreams => Programs

Phillip D. Summers, A Methodology for LISP Program Construction from Examples

Tessa Lau et al., Programming by Demonstration; An Inductive Learning Formulation

History

- 2000s: from user-provided *skeleton* (programs with holes)
  - Enabled efficient search due to limited search space
  - *counter-example guided inductive synthesis*
- 2010s:
  - Domain-specific synthesizers for commercial SW (e.g., string transformation from few input-output examples MS Excel)
  - Standard formulation SyGuS
  - Various *general-purpose* synthesizers

Armando Solar-Lezama, *Program Synthesis by Sketching*,

Sumit Gulwani, *Automating String Processing in Spreadsheets Using Input-output Examples*

Rajeev Alur et al., *Syntax-guided Synthesis*

Comparisons to Others (Synthesis vs. Compilation)

```
let rec insert x xs =
    match xs with
    | [] -> x :: []
    | h :: t ->
if x <= h then x :: xs
else h :: (insert x t)
```

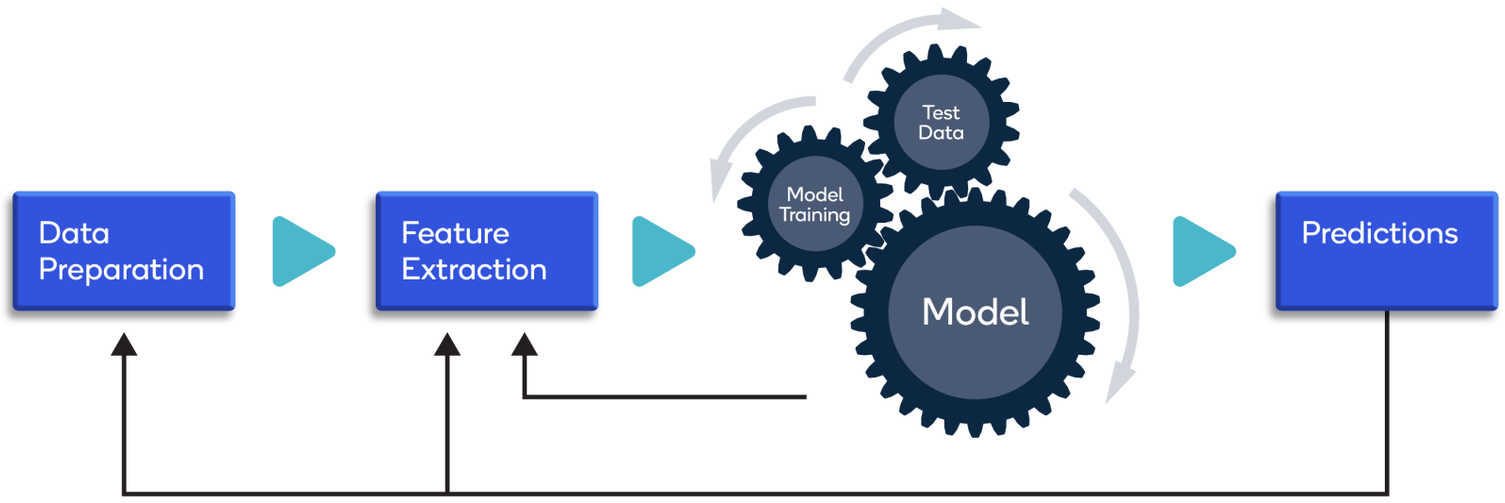
OCaml

```
append:
    push ebp
    mov ebp, esp
    push eax
    push ebx
    push len
    call malloc
    mov ebx, [ebp + 12]
    mov [eax + info], ebx
    mov dword [eax + next], 0
    mov ebx, [ebp + 8]
    cmp dword [ebx], 0
    je null_pointer
    mov ebx, [ebx]

next_element:
    cmp dword [ebx + next], 0
    je found_last
    mov ebx, [ebx + next]
    jmp next_element

found_last:
    push eax
    push addMes
    call puts
    add esp, 4
    pop eax
    mov [ebx + next], eax

go_out:
    pop ebx
    pop eax
    mov esp, ebp
    pop ebp
    ret 8

null_pointer:
    push eax
    push nullMes
    call puts
    add esp, 4
    pop eax
    mov [ebx], eax
    jmp go_out
```

Assembly

- **Commons**: High-level description ⇒ executable program
- **Diffs**:
  - Compilation: translation without discovering how to perform the task
  - Synthesis: discovery of how to perform the task

Comparisons to Others (Synthesis vs. Machine Learning)



- **Commons**: finding a function whose behavior matches with given data

- **Diffs**:

  - ML: tightly prescribed models(e.g., decision trees, neural nets), uninterpretable result, accepts noisy data

  - Synth: general classes of programs, interpretable, not robust to noise

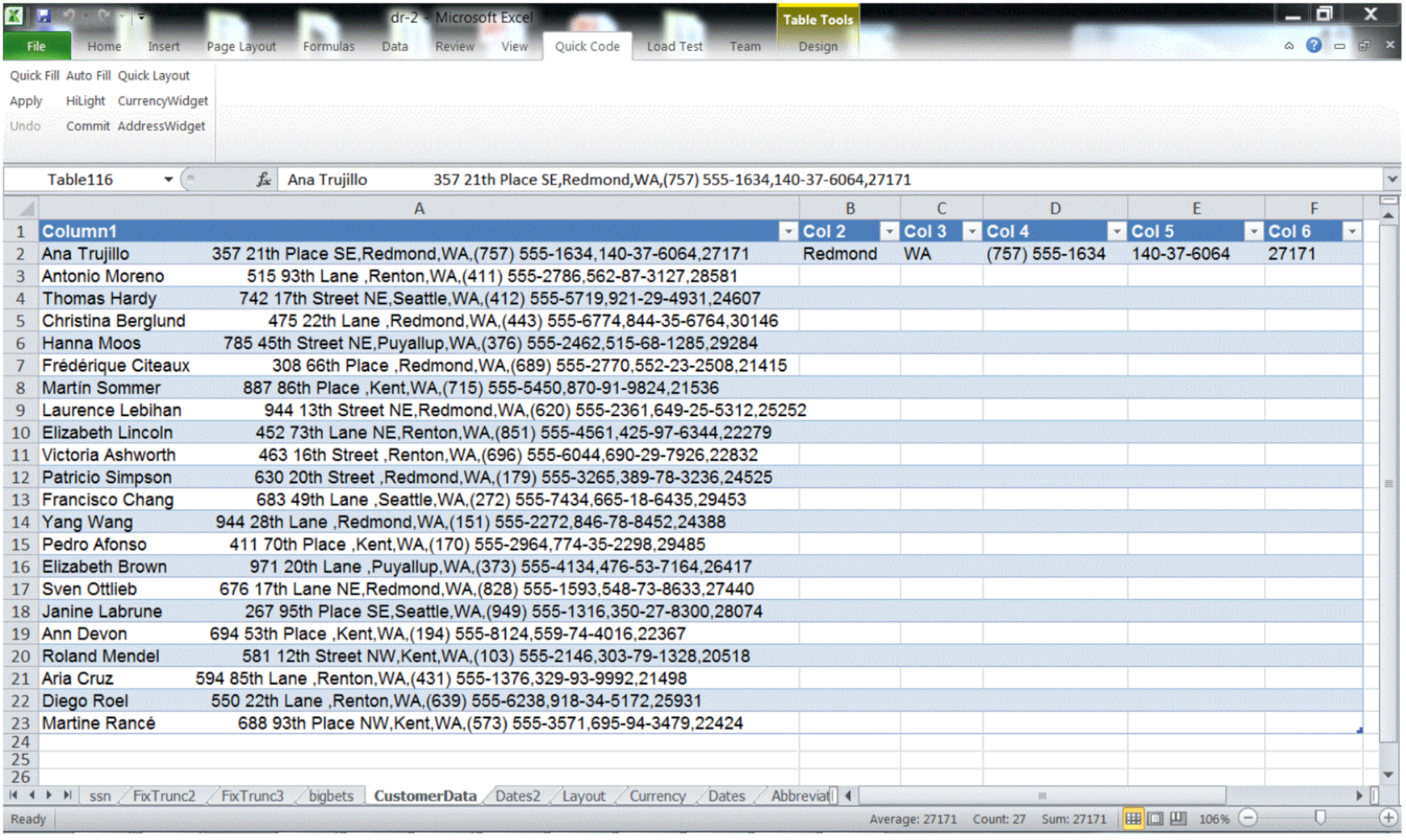
Comparisons to Others (Synthesis vs. Declarative Programming)

Example of Prolog program:

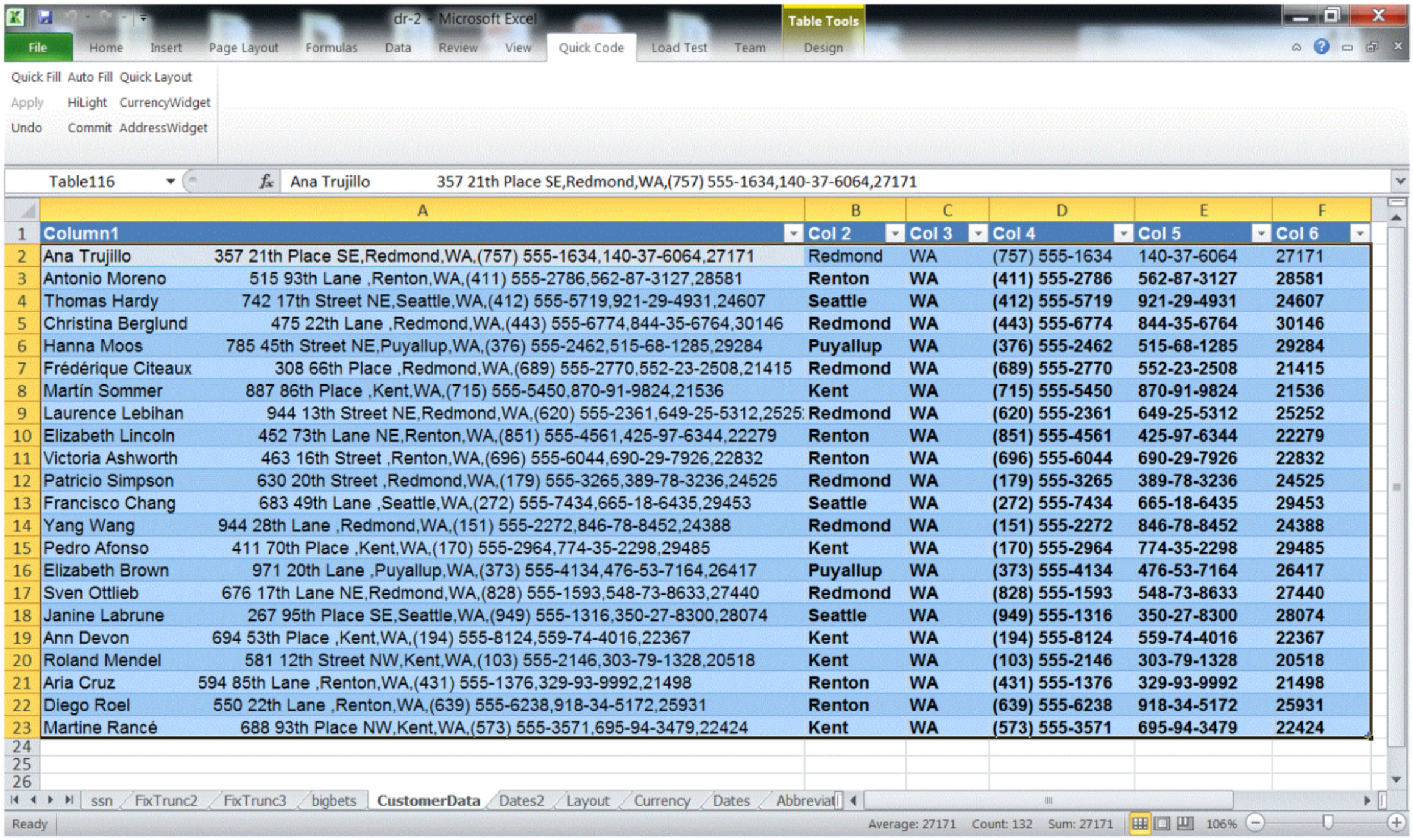
```
grandparent(X, Z) :- parent(X, Y), parent(Y, Z).
parent(X, Y) :- father(X, Y).
parent(X, Y) :- mother(X, Y).

mother(Mary, Stan). Father(Stan, Alice).
```

- **Commons**: about "what" rather than "how"
- **Diffs**:
  - Decl: spec should not be vague (e.g., IO examples X), result is slow when executed
  - Synth: spec may be vague, result is fast

Application — String Transformation (Excel FlashFill)



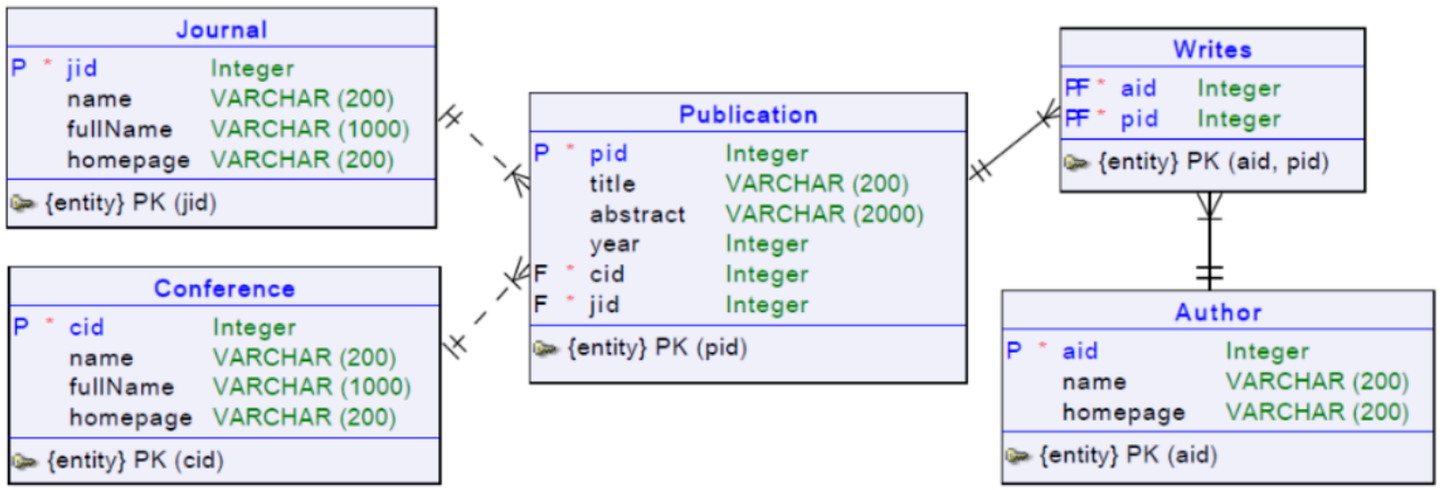
Application — String Transformation (Excel FlashFill)



	A	B	C	D	E	F
1	Column1	Col 2	Col 3	Col 4	Col 5	Col 6
2	Ana Trujillo 357 21th Place SE,Redmond,WA,(757) 555-1634,140-37-6064,27171	Redmond	WA	(757) 555-1634	140-37-6064	27171
3	Antonio Moreno 515 93th Lane ,Renton,WA,(411) 555-2786,562-87-3127,28581	Renton	WA	(411) 555-2786	562-87-3127	28581
4	Thomas Hardy 742 17th Street NE,Seattle,WA,(412) 555-5719,921-29-4931,24607	Seattle	WA	(412) 555-5719	921-29-4931	24607
5	Christina Berglund 475 22th Lane ,Redmond,WA,(443) 555-6774,844-35-6764,30146	Redmond	WA	(443) 555-6774	844-35-6764	30146
6	Hanna Moos 785 45th Street NE,Puyallup,WA,(376) 555-2462,515-68-1285,29284	Puyallup	WA	(376) 555-2462	515-68-1285	29284
7	Frédérique Citeaux 308 66th Place ,Redmond,WA,(689) 555-2770,552-23-2508,21415	Redmond	WA	(689) 555-2770	552-23-2508	21415
8	Martín Sommer 887 86th Place ,Kent,WA,(715) 555-5450,870-91-9824,21536	Kent	WA	(715) 555-5450	870-91-9824	21536
9	Laurence Lebihan 944 13th Street NE,Redmond,WA,(620) 555-2361,649-25-5312,2525	Redmond	WA	(620) 555-2361	649-25-5312	25252
10	Elizabeth Lincoln 452 73th Lane NE,Renton,WA,(851) 555-4561,425-97-6344,22279	Renton	WA	(851) 555-4561	425-97-6344	22279
11	Victoria Ashworth 463 16th Street ,Renton,WA,(696) 555-6044,690-29-7926,22832	Renton	WA	(696) 555-6044	690-29-7926	22832
12	Patricio Simpson 630 20th Street ,Redmond,WA,(179) 555-3265,389-78-3236,24525	Redmond	WA	(179) 555-3265	389-78-3236	24525
13	Francisco Chang 683 49th Lane ,Seattle,WA,(272) 555-7434,665-18-6435,29453	Seattle	WA	(272) 555-7434	665-18-6435	29453
14	Yang Wang 944 28th Lane ,Redmond,WA,(151) 555-2272,846-78-8452,24388	Redmond	WA	(151) 555-2272	846-78-8452	24388
15	Pedro Afonso 411 70th Place ,Kent,WA,(170) 555-2964,774-35-2298,29485	Kent	WA	(170) 555-2964	774-35-2298	29485
16	Elizabeth Brown 971 20th Lane ,Puyallup,WA,(373) 555-4134,476-53-7164,26417	Puyallup	WA	(373) 555-4134	476-53-7164	26417
17	Sven Ottlieb 676 17th Lane NE,Redmond,WA,(828) 555-1593,548-73-8633,27440	Redmond	WA	(828) 555-1593	548-73-8633	27440
18	Janine Labrune 267 95th Place SE,Seattle,WA,(949) 555-1316,350-27-8300,28074	Seattle	WA	(949) 555-1316	350-27-8300	28074
19	Ann Devon 694 53th Place ,Kent,WA,(194) 555-8124,559-74-4016,22367	Kent	WA	(194) 555-8124	559-74-4016	22367
20	Roland Mendel 581 12th Street NW,Kent,WA,(103) 555-2146,303-79-1328,20518	Kent	WA	(103) 555-2146	303-79-1328	20518
21	Aria Cruz 594 85th Lane ,Renton,WA,(431) 555-1376,329-93-9992,21498	Renton	WA	(431) 555-1376	329-93-9992	21498
22	Diego Roel 550 22th Lane ,Renton,WA,(639) 555-6238,918-34-5172,25931	Renton	WA	(639) 555-6238	918-34-5172	25931
23	Martine Rancé 688 93th Place NW,Kent,WA,(573) 555-3571,695-94-3479,22424	Kent	WA	(573) 555-3571	695-94-3479	22424
24						
25						
26						

Application – SQL Synthesizer (SQLizer)

Problem: "Find the number of papers in OOPSLA 2010"

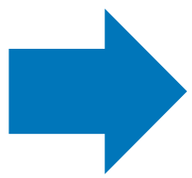


Output:

```
SELECT count(Publication.pid)
FROM Publication JOIN Conference ON Publication.cid = Conference.cid
WHERE Conference.name = "OOPSLA" AND Publication.year = 2010
```

Application — Optimization (STOKE)

```
 1 # gcc -03
 2
 3 movq rsi, r9
 4 movl ecx, ecx
 5 shrq 32, rsi
 6 andl 0xffffffff, r9d
 7 movq rcx, rax
 8 movl edx, edx
 9 imulq r9, rax
10 imulq rdx, r9
11 imulq rsi, rdx
12 imulq rsi, rcx
13 addq rdx, rax
14 jae .L0
15 movabsq 0x100000000, rdx
16 addq rdx, rcx
17 .L0:
18 movq rax, rsi
19 movq rax, rdx
20 shrq 32, rsi
21 salq 32, rdx
22 addq rsi, rcx
23 addq r9, rdx
24 adcq 0, rcx
25 addq r8, rdx
26 adcq 0, rcx
27 addq rdi, rdx
28 adcq 0, rcx
29 movq rcx, r8
30 movq rdx, rdi
```



```
 1 # STOKE
 2
 3 shlq 32, rcx
 4 movl edx, edx
 5 xorq rdx, rcx
 6 movq rcx, rax
 7 mulq rsi
 8 addq r8, rdi
 9 adcq 0, rdx
10 addq rdi, rax
11 adcq 0, rdx
12 movq rdx, r8
13 movq rax, rdi
```

Montgomery multiplication kernel from the OpenSSL RSA library. Compilations shown for gcc –O3 (left) and a stochastic optimizer (right).

http://stoke.stanford.edu

Application — Reverse Engineering

```
push   ebp
mov    ebp, esp
sub    esp, 0x10
mov    eax, dword ptr [ebp+0x0c]
mov    ecx, dword ptr [ebp+0x18]
push   ebx
mov    ebx, dword ptr [ebp+0x14]
sub    dword ptr [ebp+0x1c], ebx
mov    dword ptr [ebp+0x0c], eax
mov    eax, dword ptr [ebp+0x08]
push   esi
mov    esi, eax
push   edi
lea    edi, [eax+ecx]
sub    esi, ecx
lea    edx, [ebx-0x02]
sub    ecx, ebx
sub    ebx, edx
inc    eax
dec    dword ptr [ebp+0x10]
mov    dword ptr [ebp-0x0c], edi
mov    dword ptr [ebp-0x10], edx
mov    dword ptr [ebp+0x18], ecx
mov    dword ptr [ebp+0x14], ebx
js     0x024d993f
push   ebp
mov    ebp, esp
sub    esp, 0x10
mov    eax, dword ptr [ebp+0x0c]
mov    ecx, dword ptr [ebp+0x18]
push   ebx
mov    ebx, dword ptr [ebp+0x14]
sub    dword ptr [ebp+0x1c], ebx
mov    dword ptr [ebp+0x0c], eax
mov    eax, dword ptr [ebp+0x08]
push   esi
mov    esi, eax
push   edi
lea    edi, [eax+ecx]
sub    esi, ecx
lea    edx, [ebx-0x02]
sub    ecx, ebx
jns    0x024d9841
mov    edx, dword ptr [ebp-0x10]
movzx  ecx, byte ptr [eax-0x02]
add    edx, eax
mov    dword ptr [ebp+0x08], ecx
movzx  ecx, byte ptr [eax-0x01]
mov    dword ptr [ebp-0x08], edx
cmp    eax, edx
jnb    0x024d98e7
movzx  ebx, byte ptr [edi]
movzx  edx, byte ptr [eax]
add    ebx, edx
mov    dword ptr [ebp-0x04], edx
movzx  edx, byte ptr [esi]
lea    ebx, [ebx+ecx*4+0x04]
add    edx, ebx
add    edx, dword ptr [ebp+0x08]
mov    ebx, dword ptr [ebp+0x0c]
shr    edx, 0x03
mov    byte ptr [ebx], dl
movzx  edx, byte ptr [eax+0x01]
mov    ebx, dword ptr [ebp-0x04]
mov    dword ptr [ebp+0x08], edx
movzx  edx, byte ptr [edi+0x01]
add    edx, ecx
movzx  ecx, byte ptr [esi+0x01]
lea    edx, [edx+ebx*4+0x04]
mov    ebx, dword ptr [ebp+0x0c]
add    ecx, edx
mov    edx, dword ptr [ebp+0x08]
add    ecx, edx
shr    ecx, 0x03
mov    byte ptr [ebx+0x01], cl
movzx  edi, byte ptr [edi+0x02]
add    edi, dword ptr [ebp-0x04]
movzx  ebx, byte ptr [esi+0x02]
movzx  ecx, byte ptr [eax+0x02]
lea    edx, [edi+edx*4+0x04]
mov    edi, dword ptr [ebp-0x0c]
add    ebx, edx
mov    edx, dword ptr [ebp+0x0c]
add    ebx, ecx
```

Assembly code

```
#include <Halide.h>
#include <vector>
using namespace std;
using namespace Halide;

int main(){
  Var x_0;
  Var x_1;
  ImageParam input_1(UInt(8),2);
  Func output_1;
  output_1(x_0,x_1) =
    cast<uint8_t>((((((2+
      (2*cast<uint32_t>(input_1(x_0+1,x_1+1))) +
        cast<uint32_t>(input_1(x_0,  x_1+1)) +
        cast<uint32_t>(input_1(x_0+2,x_1+1)))
      >> cast<uint32_t>(2))) & 255));
  vector<Argument> args;
  args.push_back(input_1);
  output_1.compile_to_file("halide_out_0",args);
  return 0;
}
```

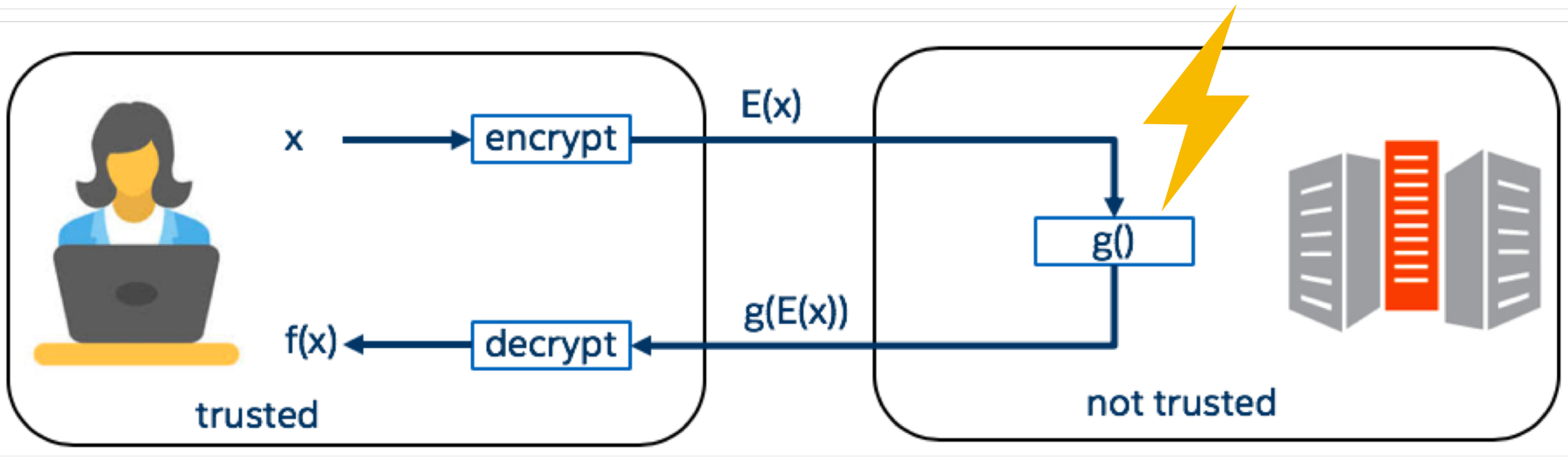
High-level DSL code

Lifting: legacy binary executable ⇒ high-level language program (for a maintanence purpose)

Shoaib Kamil, Alvin Cheung, Shachar Itzhaky, Armando Solar-Lezama, *Verified Lifting of Stencil Computations*

Mendis et al., Helium: Lifting High-Performance Stencil Kernels from Stripped x86 Binaries to Halide DSL Code

Application — Data Privacy



Accelerating *homomorphic encryption* applications

Lee et al., Optimizing homomorphic evaluation circuits via program synthesis and term rewriting, PLDI'20

Application — Program Repair

Bug

```
1  if (hbtype == TLS1_HB_REQUEST) {
2      ...
3      memcpy (bp, pl, payload);
4      ...
5  }
```

(a) The buggy part of the Heartbleed-vulnerable OpenSSL

Auto fix

```
1  if (hbtype == TLS1_HB_REQUEST
2      && payload + 18 < s->s3->rrec.length) {
3      /* receiver side: replies with TLS1_HB_RESPONSE */
4  }
```

(b) A fix generated by our tool, Angelix

≀|

Developer fix

```
1  if (1 + 2 + payload + 16 > s->s3->rrec.length)
2      return 0;
3  ...
4  if (hbtype == TLS1_HB_REQUEST) {
5      /* receiver side: replies with TLS1_HB_RESPONSE */
6  }
7  else if (hbtype == TLS1_HB_RESPONSE) {
8      /* sender side */
9  }
10 return 0;
```

(c) The developer-provided repair

Mechtaev et al., Angelix: Scalable Multiline Program Patch Synthesis via Symbolic Analysis, ICSE'16

Recent Two Directions

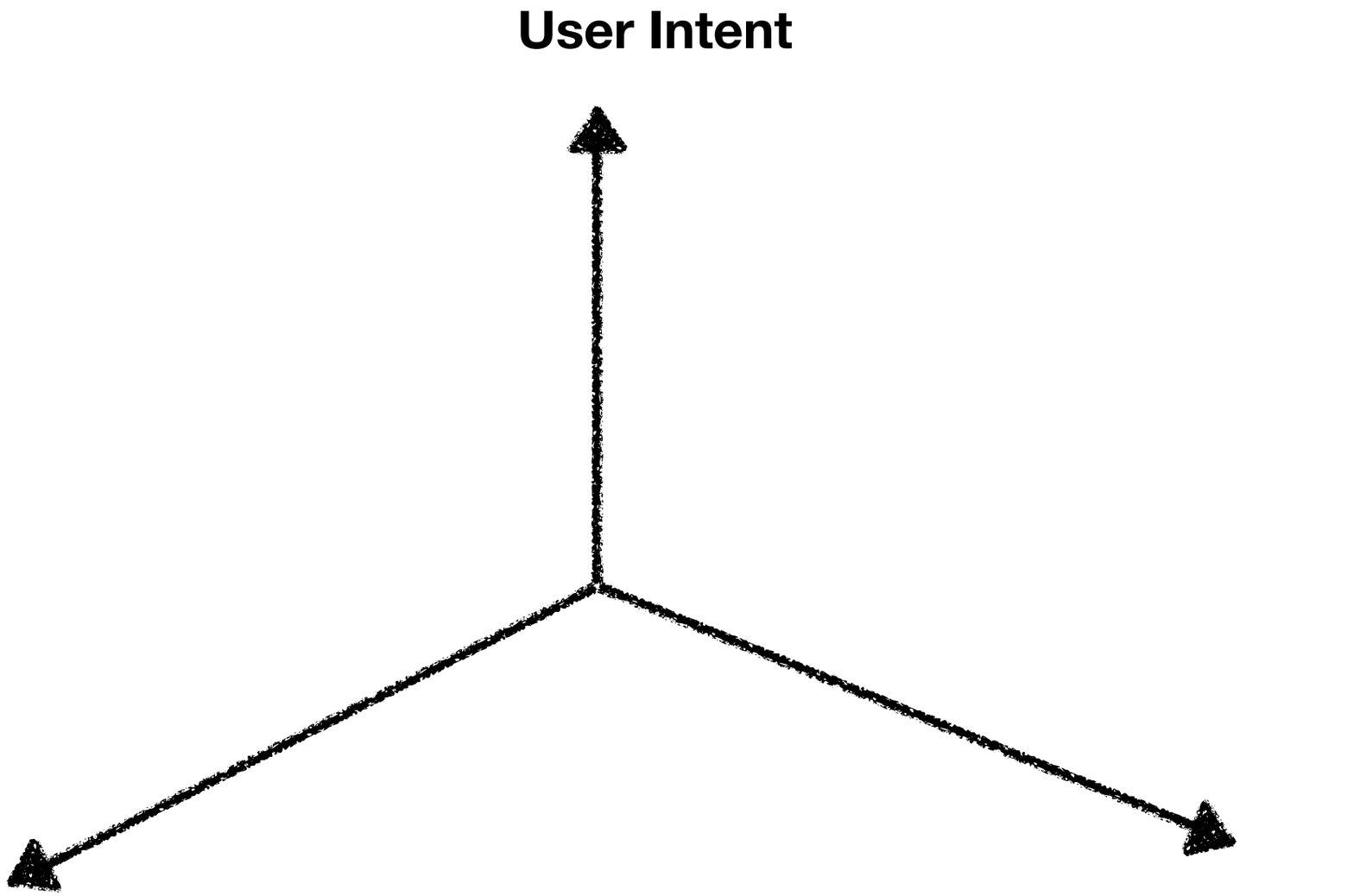
	Classical Program Synthesis	Neural Program Synthesis
Starting period	1960s	2010s after the rise of deep learning
Guarantee correctness?	**O**	**X**
Scalability	Low	High
Method	**Search algorithm**	**Neural net inference**
Example	Excel FlashFill	GitHub Copilot
Conferences	POPL, PLDI, CAV, ICSE, FSE, … (PL/SE)	NIPS, ICML, AAAI, … (ML)

Example of Neural Synthesis — GitHub Copilot

```
def solution(lst):
    """Given a non-empty list of integers, return the sum of all of the odd elements
    that are in even positions.

    Examples
    solution([5, 8, 7, 1]) ==>12
    solution([3, 3, 3, 3, 3]) ==>9
    solution([30, 13, 24, 321]) ==>0
    """
    return sum(lst[i] for i in range(0,len(lst)) if i % 2 == 0 and lst[i] % 2 == 1)
```

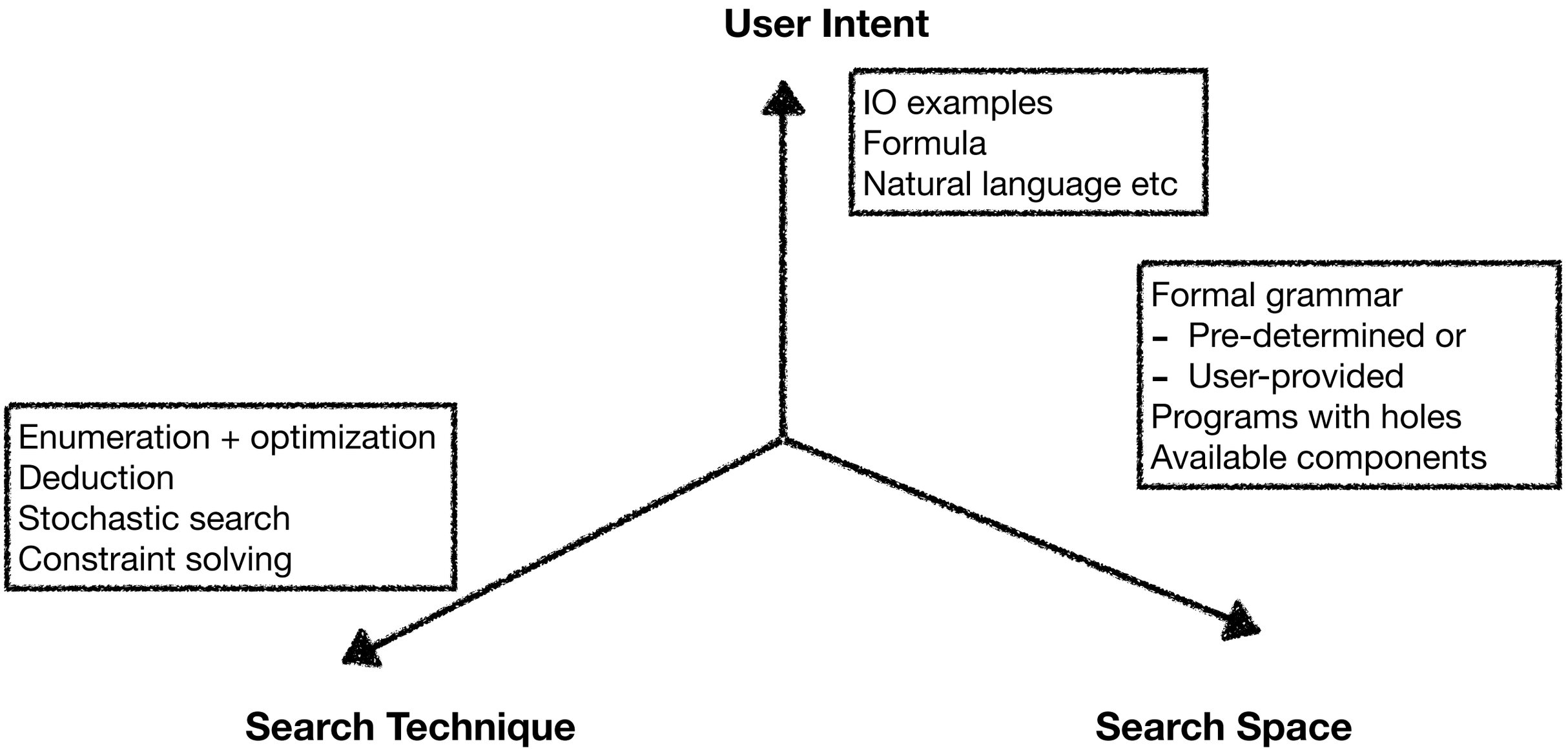
- Natural language description (+ IO examples) → Python code
- Based on OpenAI GPT-3
- May generate errors (syntax, type, runtime, functionality, etc)
- Program size ↑ → error rate ↑

Dimensions in Program Synthesis



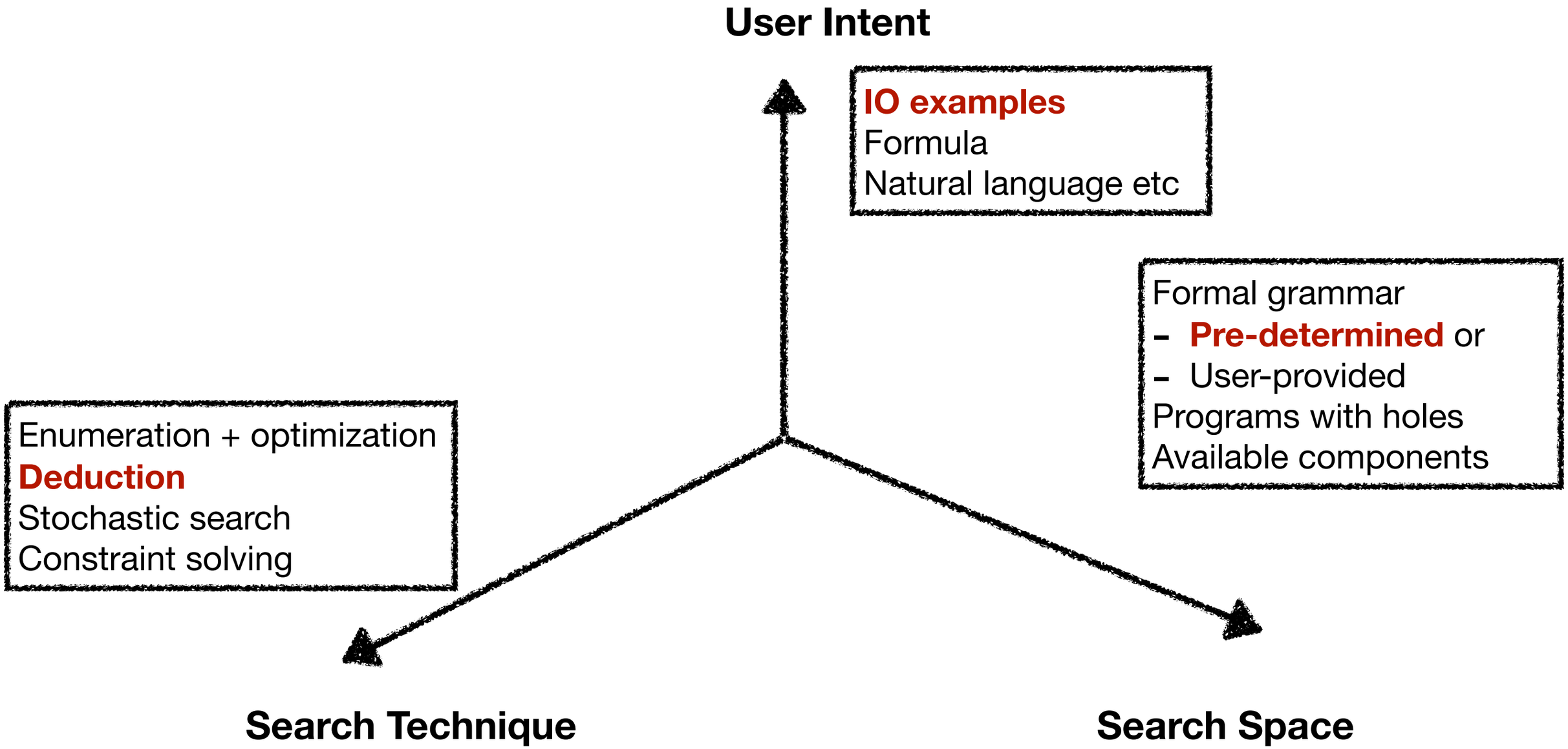
Dimensions in program synthesis, PPDP'10

Dimensions in Program Synthesis



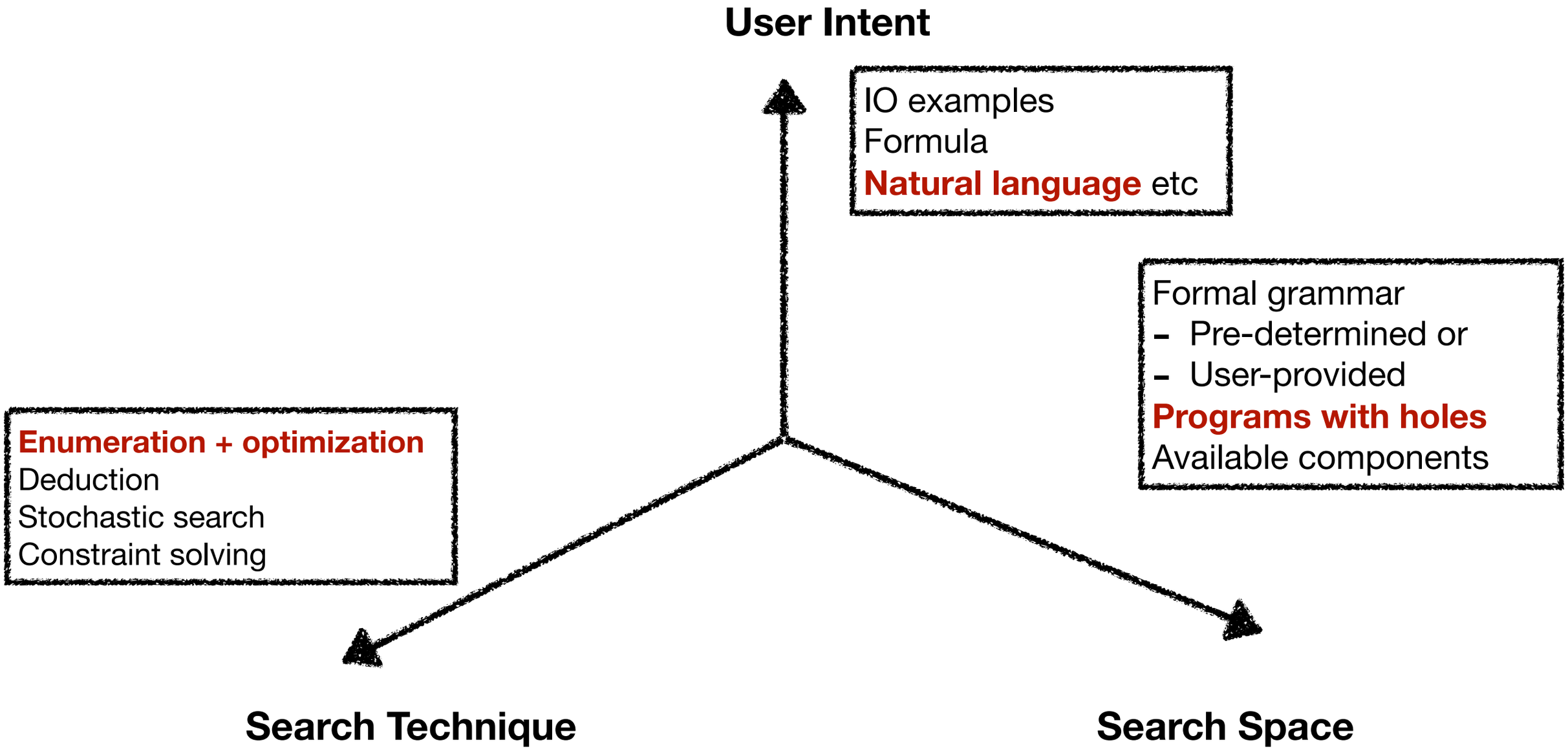
Dimensions in program synthesis, PPDP'10

FlashFill



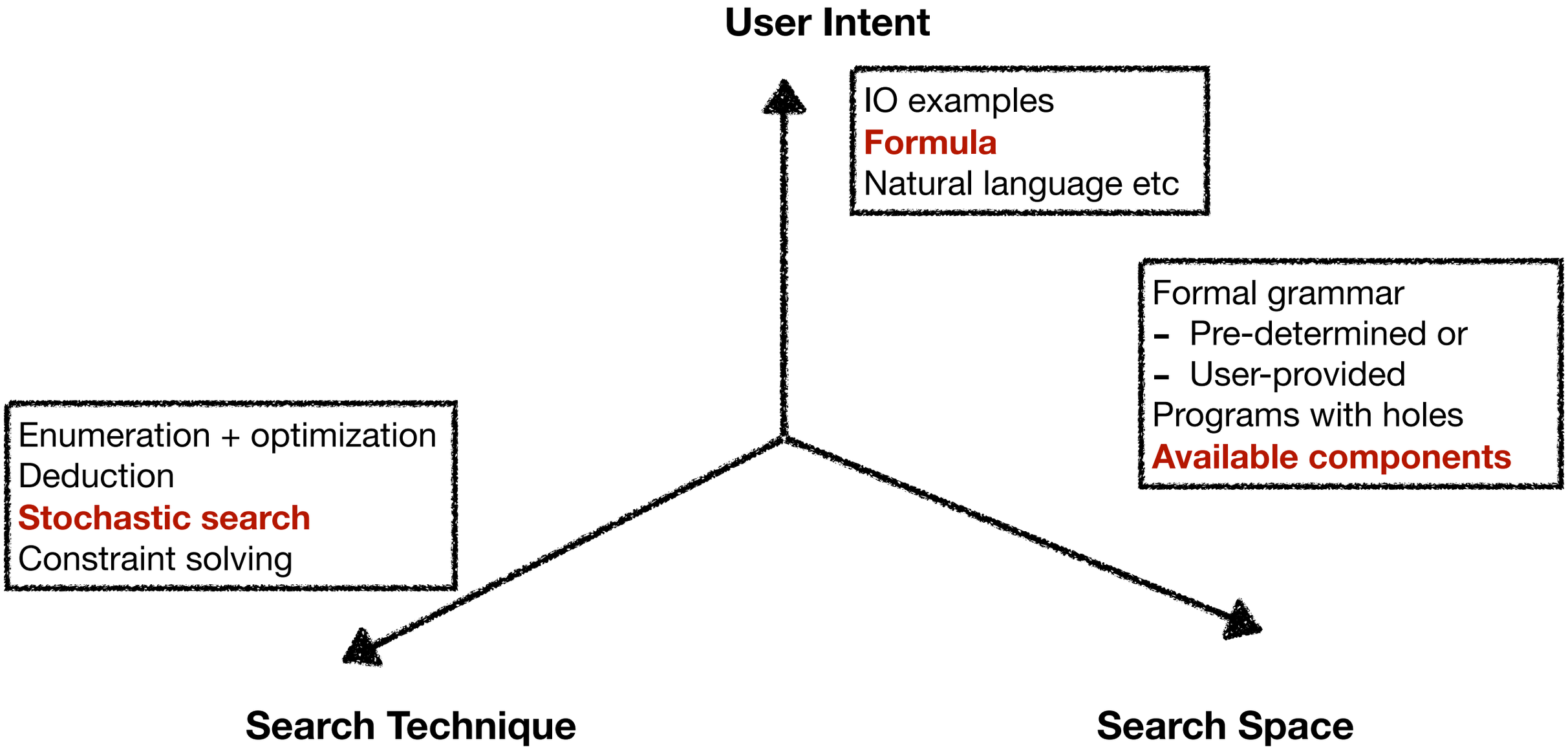
Dimensions in program synthesis, PPDP'10

SQLizer



Dimensions in program synthesis, PPDP'10

STOKE



Dimensions in program synthesis, PPDP'10

Observations on Various Synthesizers

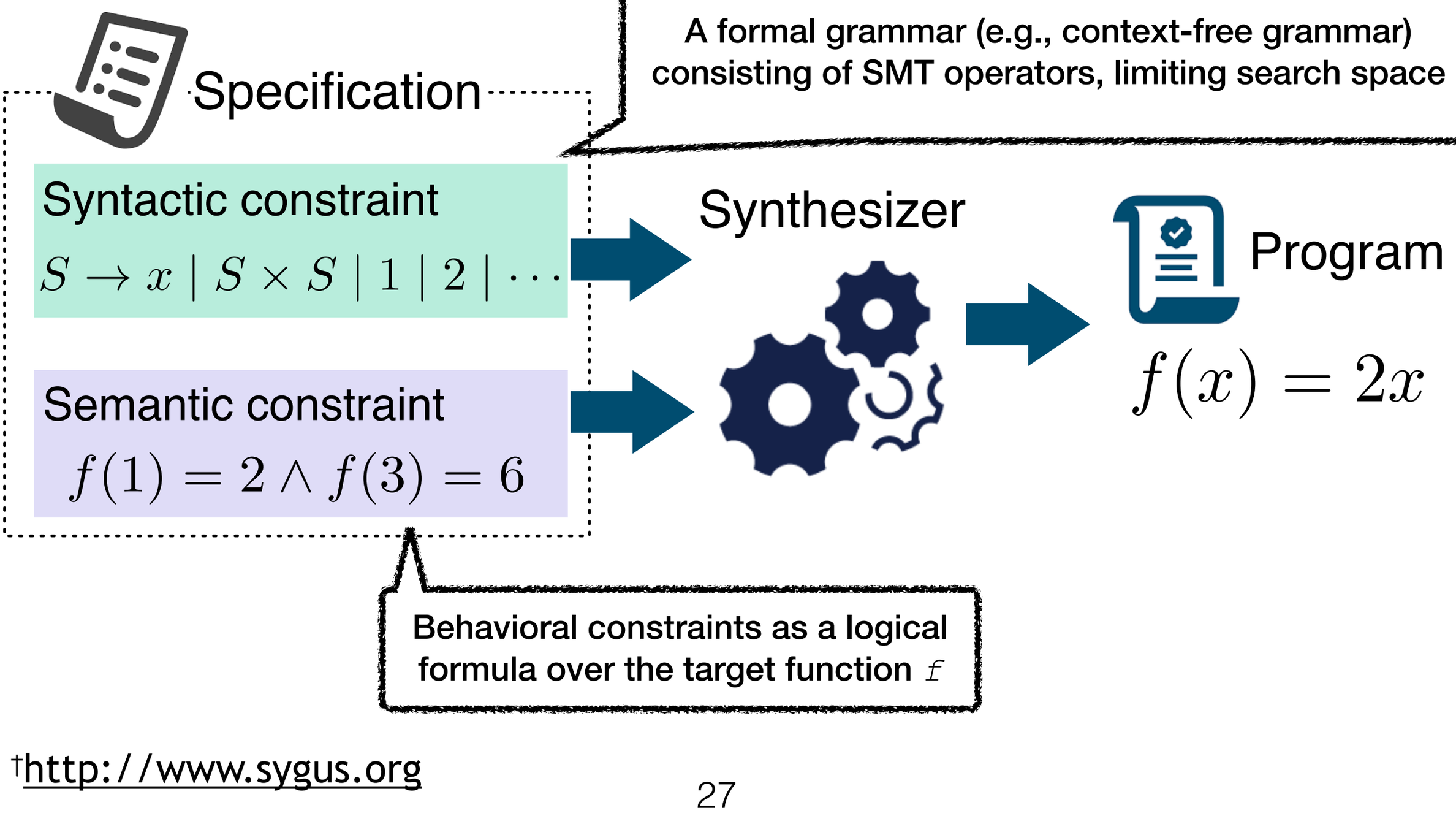
- Synthesis = finding a program in a space defined by a *context-free grammar*

- Correctness specification as a *first-order logic formula*

Standard Formulation

- Standardized formulation called Syntax-guided Synthesis (SyGuS)
- Facilitates study of *general-purpose* synthesis strategies
- Enables comparisons between various synthesizers (e.g., annual SyGuS competition)

Syntax-Guided Program Synthesis (SyGuS)†



†http://www.sygus.org