

~~FlashMeta~~ Microsoft PROSE SDK: A Framework for Inductive Program Synthesis

Oleksandr Polozov
University of Washington

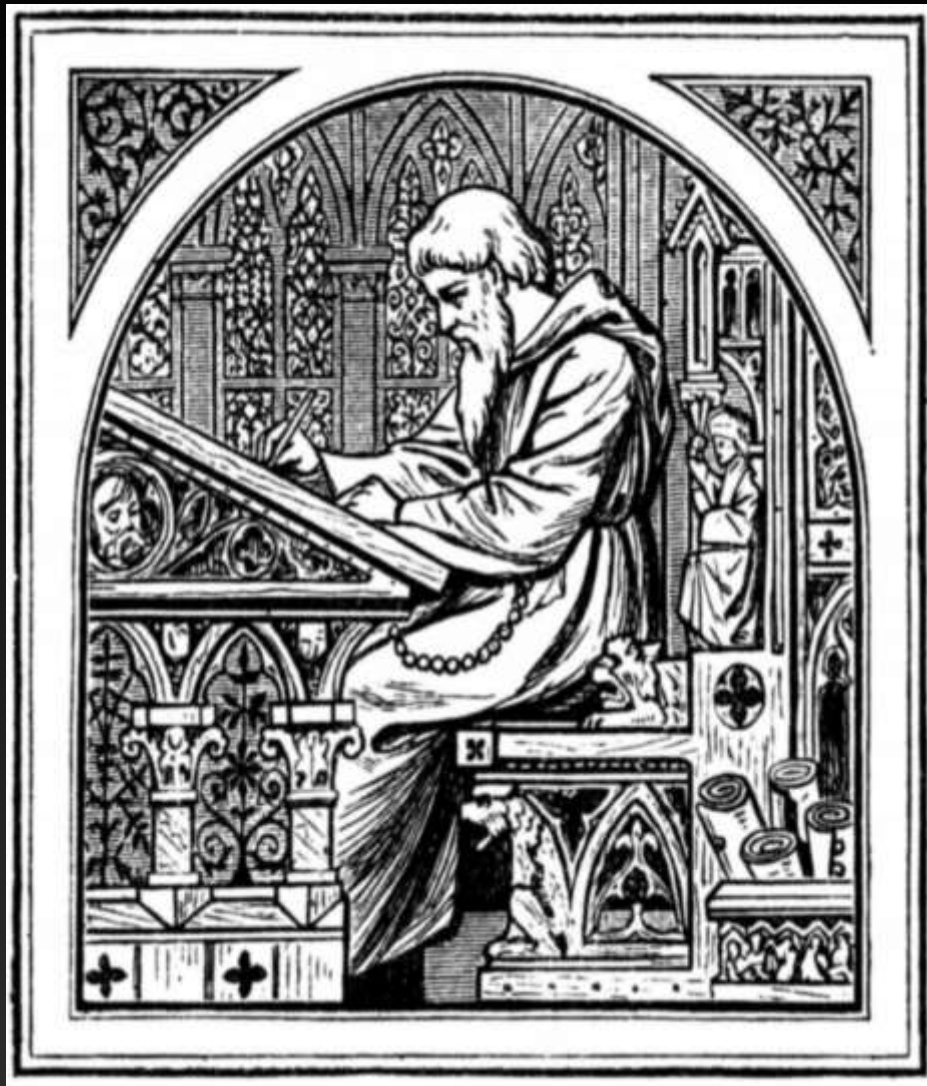
Sumit Gulwani
Microsoft Research

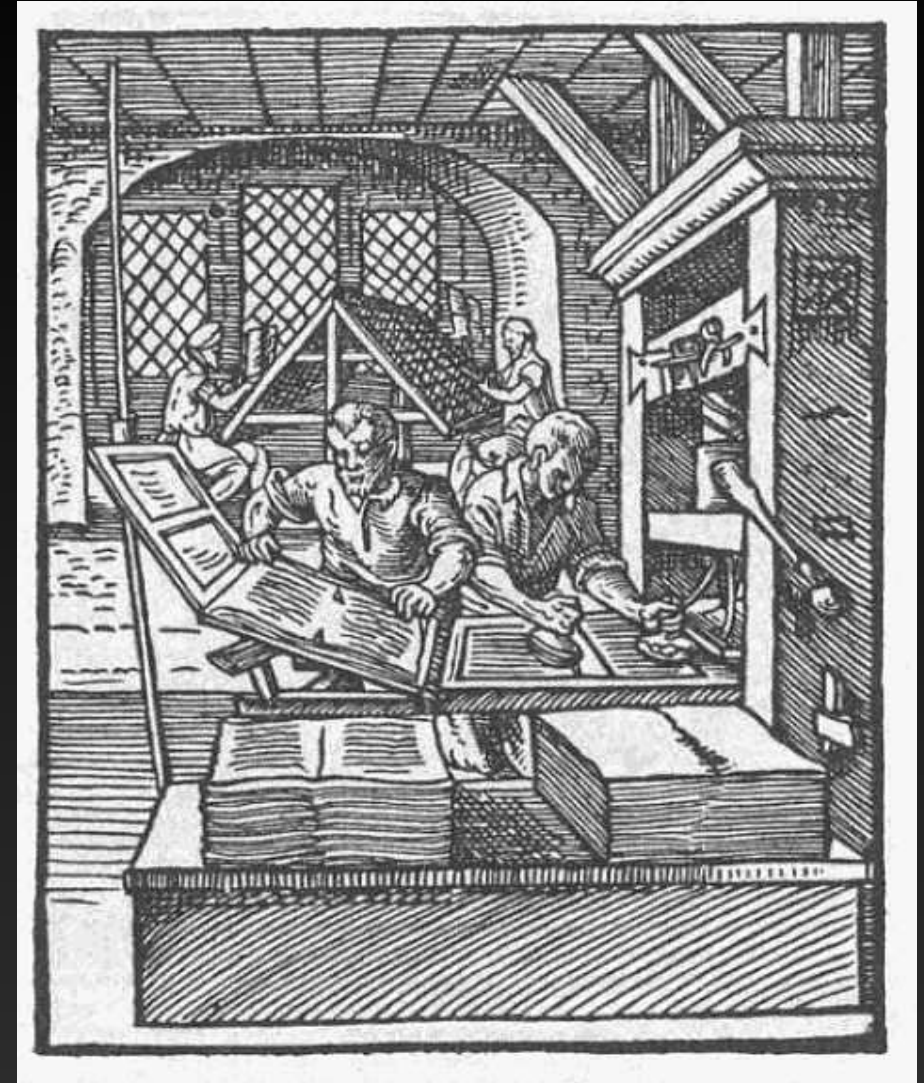
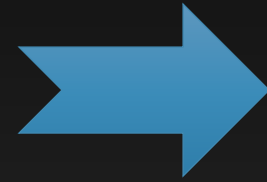


Microsoft
Research

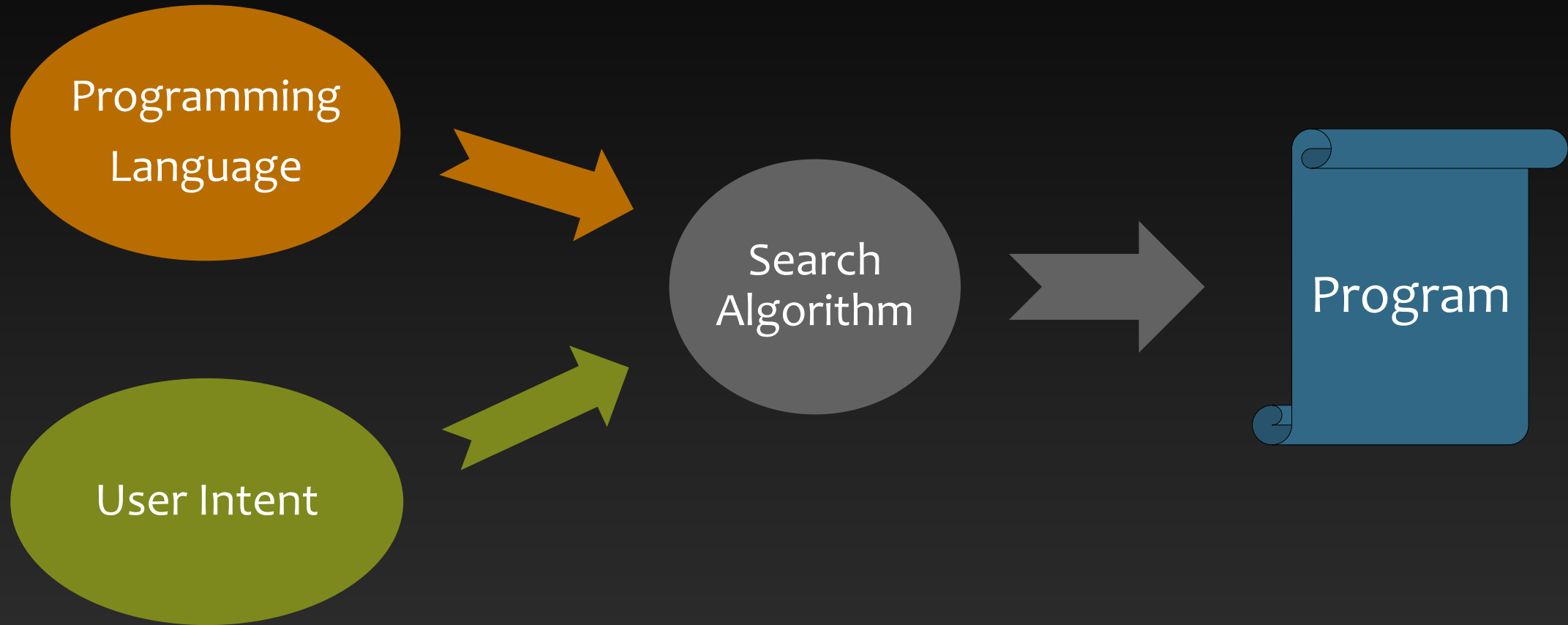
Why do people create frameworks?

Industrialization (a.k.a. “Tech Transfer”)





Program Synthesis: “The Ultimate Dream” of CS



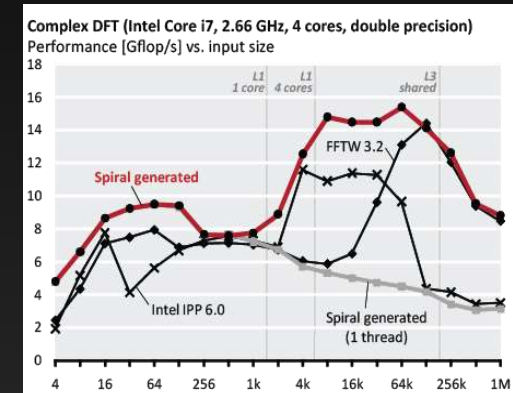
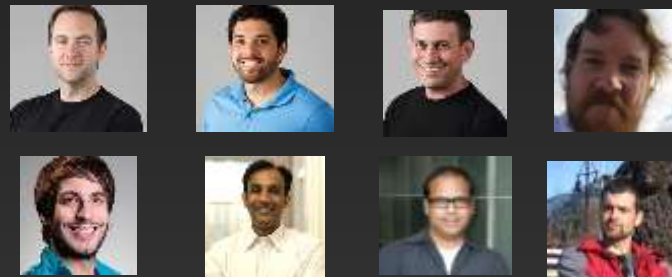
Industrialization Time?



Flash Fill (2010-2012)



Trifacta (2012-2015)

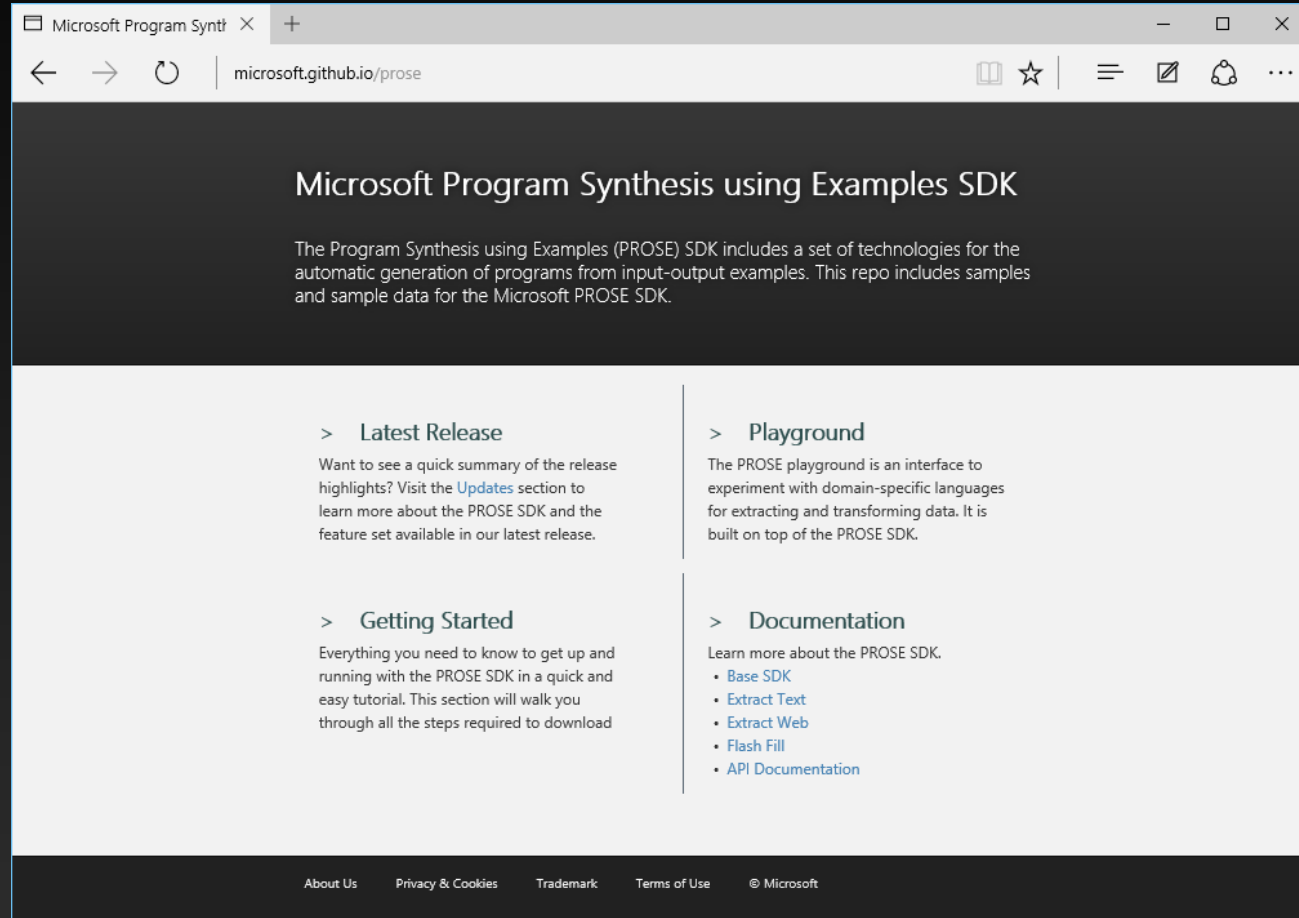


SPIRAL (2000-2015)



+114
more

Microsoft Program Synthesis using Examples SDK



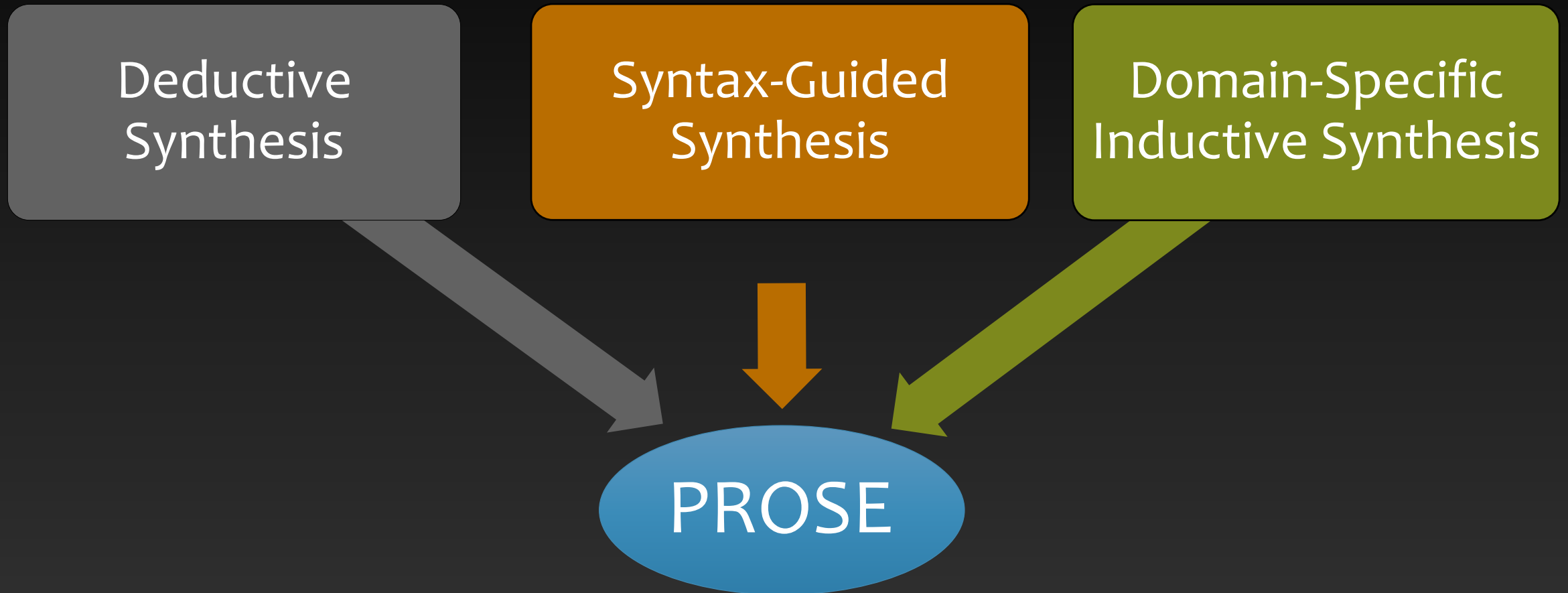
The screenshot shows a web browser window with the URL microsoft.github.io/prose. The page title is "Microsoft Program Synthesis using Examples SDK". The main content area has a dark header with the title and a paragraph: "The Program Synthesis using Examples (PROSE) SDK includes a set of technologies for the automatic generation of programs from input-output examples. This repo includes samples and sample data for the Microsoft PROSE SDK." Below this, there are four sections arranged in a 2x2 grid, each with a right-pointing chevron icon:

- > Latest Release**
Want to see a quick summary of the release highlights? Visit the [Updates](#) section to learn more about the PROSE SDK and the feature set available in our latest release.
- > Playground**
The PROSE playground is an interface to experiment with domain-specific languages for extracting and transforming data. It is built on top of the PROSE SDK.
- > Getting Started**
Everything you need to know to get up and running with the PROSE SDK in a quick and easy tutorial. This section will walk you through all the steps required to download
- > Documentation**
Learn more about the PROSE SDK.
 - [Base SDK](#)
 - [Extract Text](#)
 - [Extract Web](#)
 - [Flash Fill](#)
 - [API Documentation](#)

The footer contains links for "About Us", "Privacy & Cookies", "Trademark", "Terms of Use", and "© Microsoft".

<https://microsoft.github.io/prose>

Shoulders of Giants



Shoulders of Giants

Deductive
Synthesis

- + No invalid candidates \implies fast
- [Usually] complete specs
- Domain axiomatization

Püschel et al. [IEEE '05]
Panchekha et al. [PLDI '15]
Manna, Waldinger [TOPLAS '80]

PROSE

Shoulders of Giants

Syntax-Guided
Synthesis

Alur et al. [FMCAD '13]



PROSE

- + Shrinks the search space
- + Generic algorithms

- No domain-specific insights
- Limited to SMT-LIB

Shoulders of Giants

- + Arbitrarily complex DSLs
- + Input/output examples
- 1-2 person-years (PhD)
- One-off

Domain-Specific
Inductive Synthesis

Lau et al. [ICML '00]
Gulwani [POPL '10] etc.
Feser et al. [PLDI '15]

PROSE

Shoulders of Giants

“Divide & Conquer”

Deductive
Synthesis



Search
Algorithm

“Search over a DSL”

Syntax-Guided
Synthesis



Programming
Language

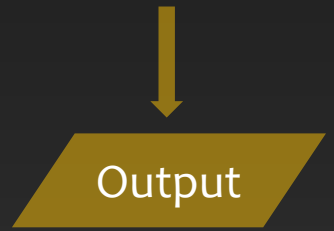
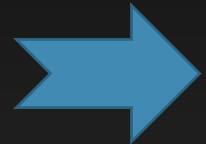
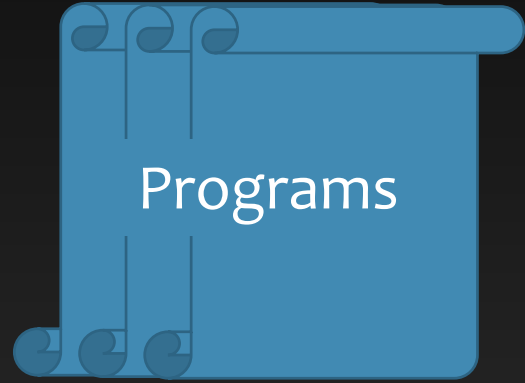
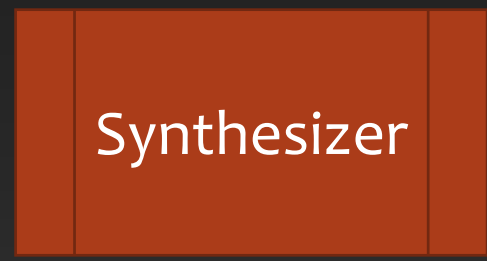
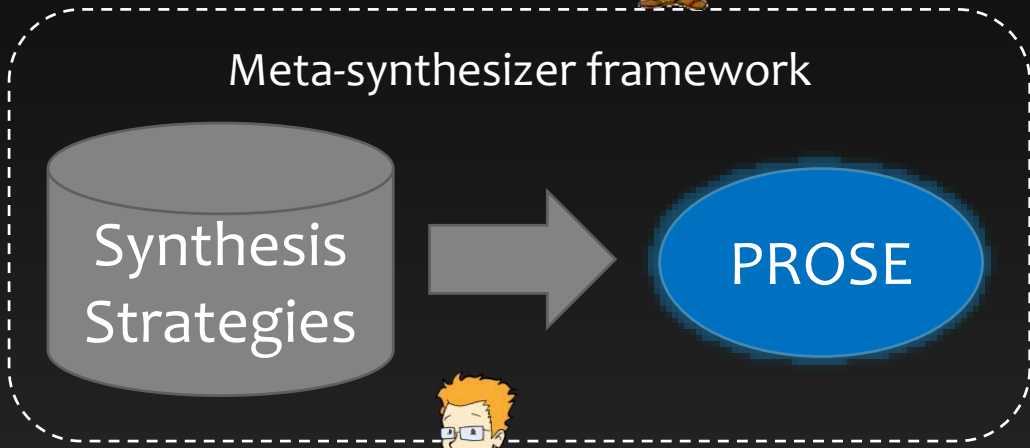
“Learn from examples”

Domain-Specific
Inductive Synthesis



User
Intent

PROSE





Domain-Specific Language

FlashFill (portion) as a PROSE DSL

```
string output(string[] inputs) :=  
  | ConstantString(s)  
  | let string x = std.list.Kth(inputs, k) in  
    Substring(x, positionPair(x));  
  
Tuple<int, int> positionPair(string s) :=  
  std.Pair(positionIn(s), positionIn(s));  
  
int positionIn(string s) := AbsolutePosition(s, k)  
  | RegexPosition(s, std.Pair(r, r), k);  
  
const int k;      const RegularExpression r;  const string s;
```

DSL design = Art + *Lots* of iterations



Inductive Specification

Input-Output Examples

input state σ

\Rightarrow

output value out

"206-279-6261"

\Rightarrow

"(206) 279-6261"

"415.413.0703"

\Rightarrow

"(415) 413-0703"

"(646) 408 6649"

\Rightarrow

"(646) 408-6649"

When one example is too many

[1] A. Ahmed, A. W. Appel, C. D. Richards, K. N. Swadi, G. Tan, and D. C. Wang. Semantic foundations for typed assembly languages. *ACM Trans. Program. Lang. Syst.*, 32(3), 2010.

[2] A. W. Appel. *Program Logics for Certified Compilers*. Cambridge University Press, 2014.

[3] A. W. Appel and S. Blazy. Separation logic for small-step Cminor. In *TPHOLs*, volume 4732 of LNCS, pages 5–21. Springer, 2007.

[4] A. W. Appel and D. A. McAllester. An indexed model of recursive types for foundational proof-carrying code. *ACM Trans. Program. Lang. Syst.*, 23(5):657–683, 2001.

[5] Y. Bertot. Structural abstract interpretation: A formal study using Coq. In *Language Engineering and Rigorous Software Development, LerNet Summer School*, pages 153–194. Springer, 2008.

[6] B. Blanchet, P. Cousot, R. Cousot, J. Feret, L. Mauborgne, A. Mine, D. Monniaux, and X. Rival. A static analyzer for large safety-critical software. In *PLDI*, pages 196–207. ACM, 2003.

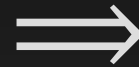
[7] S. Blazy, Z. Dargaye, and X. Leroy. Formal verification of a C compiler front-end. In *Formal Methods*, volume 4085 of LNCS, pages 460–475. Springer, 2006.

[8] S. Blazy, V. Laporte, A. Maroneze, and D. Pichardie. Formal verification of a C value analysis based on abstract interpretation. In *SAS*, volume 7935 of LNCS, pages 324–344. Springer, 2013.

[9] S. Boldo and G. Melquiond. Flocq: A unified library for proving floating-point algorithms in Coq. In *ARITH*, pages 243–252. IEEE, 2011.

[10] T. Braibant, J.-H. Jourdan, and D. Monniaux. Implementing and reasoning about hash-consed data structures in Coq. *J. Autom. Reasoning*, 53(3):271–304, 2014.

[11] D. Cachera, T. P. Jensen, D. Pichardie, and V. Rusu. Extracting a data flow analyser in constructive logic. *Theor. Comput. Sci.*, 342(1):56–78, 2005.



List of Authors	Publication Year
A. Ahmed, A. W. Appel, C. D. Richards, K. N. Swadi, G. Tan, and D. C. Wang	2010
A. W. Appel	2014
A. W. Appel and S. Blazy	2007
A. W. Appel and D. A. McAllester	2001
Y. Bertot	2008
B. Blanchet, P. Cousot, R. Cousot, J. Feret, L. Mauborgne, A. Mine, D. Monniaux, and X. Rival	2003
S. Blazy, Z. Dargaye, and X. Leroy	2006
S. Blazy, V. Laporte, A. Maroneze, and D. Pichardie	2013
S. Boldo and G. Melquiond	2011
T. Braibant, J.-H. Jourdan, and D. Monniaux	2014
D. Cachera, T. P. Jensen, D. Pichardie, and V. Rusu	2005
A. Chlipala	2008
S. Cho, J. Kang, J. Choi, C.-K. Hur, and K. Yi	2013
P. Cousot and R. Cousot	1977
P. Cousot and R. Cousot	1979
P. Cousot, R. Cousot, J. Feret, L. Mauborgne, A. Mine, and X. Rival	2009
P. Cousot, R. Cousot, J. Feret, L. Mauborgne, A. Mine, D. Monniaux, and X. Rival	2006
A. Foulhe, D. Monniaux, and M. Perin	2013
A. Foulhe and S. Boulme	2014
D. Greenaway, J. Andronick, and G. Klein	2012

Inductive Specification

input state σ

\Rightarrow

output constraint $\varphi(\text{out})$

- [1] A. Ahmed, A. W. Appel, C. D. Richards, K. N. Swadi, G. Tan, and D. C. Wang. Semantic foundations for typed assembly languages. *ACM Trans. Program. Lang. Syst.*, 32(3), 2010.
- [2] A. W. Appel. *Program Logics for Certified Compilers*. Cambridge University Press, 2014.
- [3] A. W. Appel and S. Blazy. Separation logic for small-step Cminor. In *TPHOLS*, volume 4732 of LNCS, pages 5–21. Springer, 2007.
- [4] A. W. Appel and D. A. McAllester. An indexed model of recursive types for foundational proof-carrying code. *ACM Trans. Program. Lang. Syst.*, 23(5):657–683, 2001.
- [5] Y. Bertot. Structural abstract interpretation: A formal study using Coq. In *Language Engineering and Rigorous Software Development*, LerNet Summer School, pages 153–194. Springer, 2008.
- [6] B. Blanchet, P. Cousot, R. Cousot, J. Feret, L. Mauborgne, A. Mine, D. Monniaux, and X. Rival. A static analyzer for large safety-critical software. In *PLDI*, pages 196–207. ACM, 2003.
- [7] S. Blazy, Z. Dargaye, and X. Leroy. Formal verification of a C compiler front-end. In *Formal Methods*, volume 4085 of LNCS, pages 460–475. Springer, 2006.
- [8] S. Blazy, V. Laporte, A. Maroneze, and D. Pichardie. Formal verification of a C value analysis based on abstract interpretation. In *SAS*, volume 7935 of LNCS, pages 324–344. Springer, 2013.
- [9] S. Boldo and G. Melquiond. Flocq: A unified library for proving floating-point algorithms in Coq. In *ARITH*, pages 243–252. IEEE, 2011.

\Rightarrow

$\text{out} \ni ["2010", "2014", \dots]$

Inductive Specification

input state σ

\Rightarrow

output constraint $\varphi(\text{out})$



Examples are ambiguous!

[1] A. Ahmed, A. W. Appel, C. D. Richards, K. N. Swadi, G. Tan, and D. C. Wang. Semantic foundations for typed assembly languages. ACM Trans. Program. Lang. Syst., 32(3), 2010.
[2] A. W. Appel. Program Logics for Certified Compilers. Cambridge University Press, 2014.

From:

... and up to 10^{20} more candidates

all lines ending with “Number ◦ Dot”

“Space ◦ Number ◦ Dot”

starting with “Word ◦ Space ◦ CamelCase”

Extract:

the first “Number” before a “Dot”

the last “Number” before a “Dot”

the last “Number” before a “Dot ◦ LineBreak”

the last “Number”

text between the last “Space” and the last “Dot”

the first “Comma ◦ Space” and the last “Dot ◦ LineBreak”

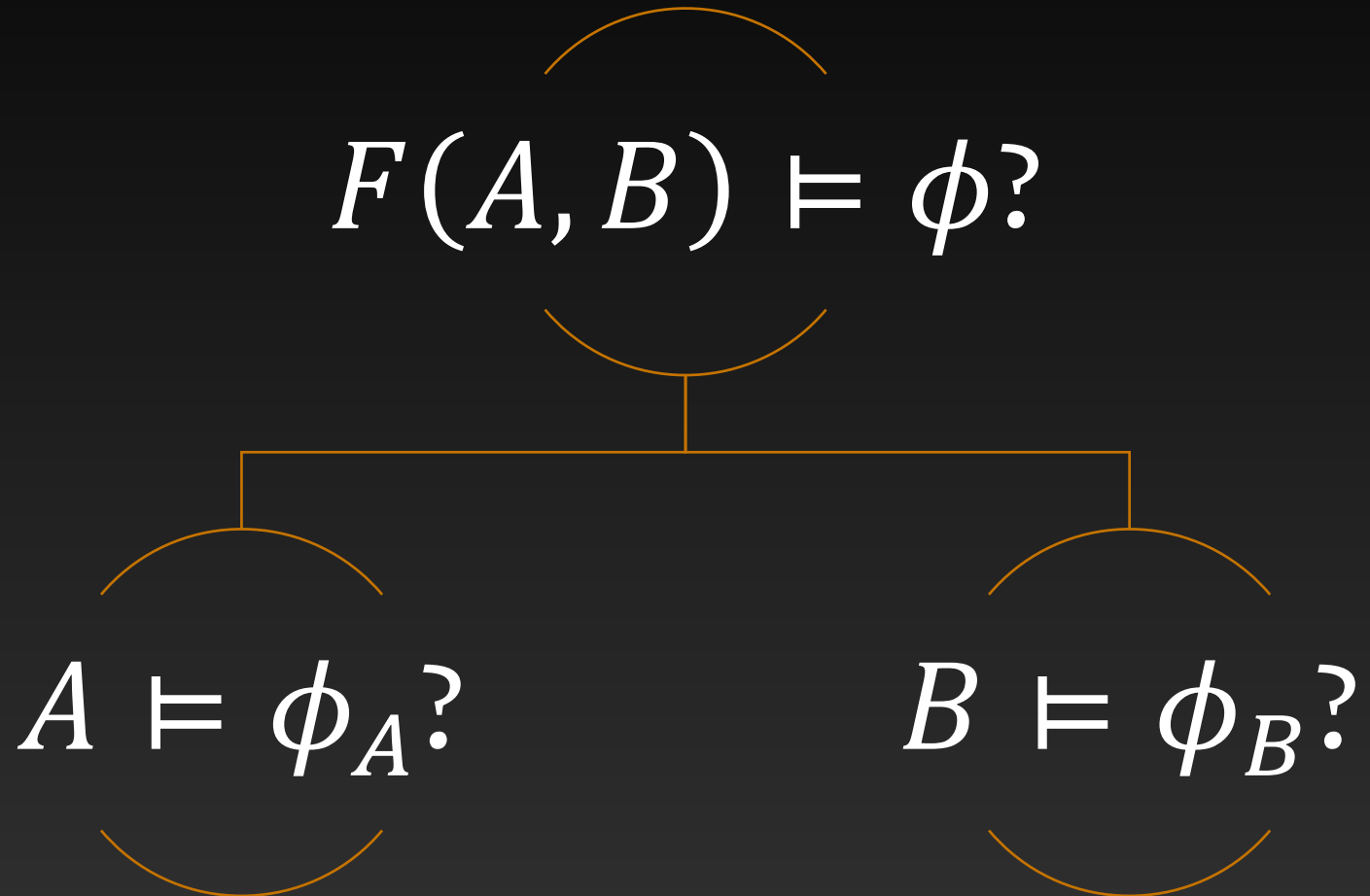
One program is insufficient.

Program Set \Rightarrow Ranking
(Version Space Algebra) User interaction
Runtime correction
...



Synthesis Strategy

Observation 1: Inverse Semantics



Concat(F, E)

$$\varphi: \begin{cases} \text{"Kathleen S. Fisher"} \Rightarrow \text{"Dr. Fisher"} \\ \text{"Bill Gates, Sr."} \Rightarrow \text{"Dr. Gates"} \end{cases}$$

$\exists E$: Concat(F, E) satisfies φ if and only if F satisfies _____ ?

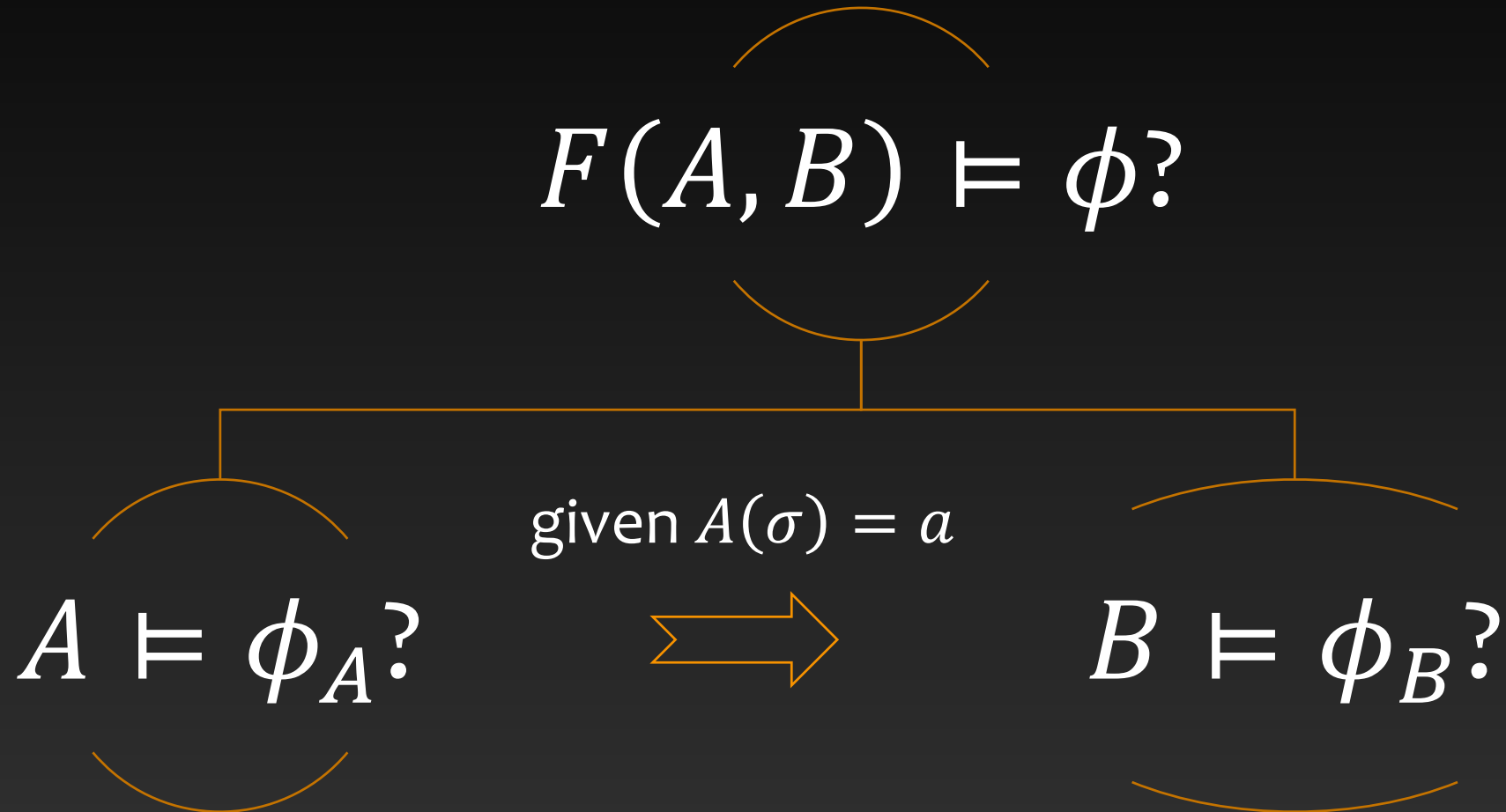
$$\varphi_f: \begin{cases} \text{"Kathleen S. Fisher"} \Rightarrow \text{"D"} \vee \text{"Dr"} \vee \text{"Dr."} \vee \text{"Dr. "} \vee \text{"Dr. F"} \vee \dots \\ \text{"Bill Gates, Sr."} \Rightarrow \text{"D"} \vee \text{"Dr"} \vee \text{"Dr."} \vee \text{"Dr. "} \vee \text{"Dr. G"} \vee \dots \end{cases}$$

$\exists F$: Concat(F, E) satisfies φ if and only if E satisfies _____ ?



F and E are not independent!

Observation 2: Skolemization



Concat(F, E)

$$\varphi: \begin{cases} \text{"Kathleen S. Fisher"} \Rightarrow \text{"Dr. Fisher"} \\ \text{"Bill Gates, Sr."} \Rightarrow \text{"Dr. Gates"} \end{cases}$$

$\exists E$: Concat(F, E) satisfies φ if and only if F satisfies _____?

$$\varphi_f: \begin{cases} \text{"Kathleen S. Fisher"} \Rightarrow \text{"D"} \vee \text{"Dr"} \vee \text{"Dr."} \vee \text{"Dr. "} \vee \text{"Dr. F"} \vee \dots \\ \text{"Bill Gates, Sr."} \Rightarrow \text{"D"} \vee \text{"Dr"} \vee \text{"Dr."} \vee \text{"Dr. "} \vee \text{"Dr. G"} \vee \dots \end{cases}$$

Given an output of F , Concat(F, E) satisfies φ if and only if E satisfies _____?

$$F = \begin{cases} \text{"Kathleen S. Fisher"} \Rightarrow \text{"Dr. "} \\ \text{"Bill Gates, Sr."} \Rightarrow \text{"Dr. "} \end{cases} \Rightarrow \varphi_E: \begin{cases} \text{"Kathleen S. Fisher"} \Rightarrow \text{"Fisher"} \\ \text{"Bill Gates, Sr."} \Rightarrow \text{"Gates"} \end{cases}$$

Inverse Semantics + Skolemization = Witness Function

Witness function: $\varphi \mapsto \varphi_F$

$\exists E$: $\text{Concat}(F, E)$ satisfies φ if and only if F satisfies _____ ?

Conditional witness function: $(\varphi \mid F(\sigma) = f) \mapsto \varphi_E$

Given an output of F , $\text{Concat}(F, E)$ satisfies φ if and only if E satisfies _____ ?



Domain-Specific

Modular

No synthesis reasoning

Enable efficient deduction

Results

Unifies 10+ prior POPL/PLDI/... papers

- Lau, T., Domingos, P., & Weld, D. S. (2000). Version Space Algebra and its Application to Programming by Demonstration. In *ICML* (pp. 527–534).
- Kitzelmann, E. (2011). A combined analytical and search-based approach for the inductive synthesis of functional programs. *KI-Künstliche Intelligenz*, 25(2), 179–182.
- Gulwani, S. (2011). Automating string processing in spreadsheets using input-output examples. In *POPL* (Vol. 46, p. 317).
- Singh, R., & Gulwani, S. (2012). Learning semantic string transformations from examples. *VLDB*, 5(8), 740–751.
- Andersen, E., Gulwani, S., & Popovic, Z. (2013). A Trace-based Framework for Analyzing and Synthesizing Educational Progressions. In *CHI* (pp. 773–782).
- Yessenov, K., Tulsiani, S., Menon, A., Miller, R. C., Gulwani, S., Lampson, B., & Kalai, A. (2013). A colorful approach to text processing by example. In *UIST* (pp. 495–504).
- Le, V., & Gulwani, S. (2014). FlashExtract : A Framework for Data Extraction by Examples. In *PLDI* (p. 55).
- Barowy, D. W., Gulwani, S., Hart, T., & Zorn, B. (2015). FlashRelate: Extracting Relational Data from Semi-Structured Spreadsheets Using Examples. In *PLDI*.
- Kini, D., & Gulwani, S. (2015). FlashNormalize : Programming by Examples for Text Normalization. *IJCAI*.
- Osera, P.-M., & Zdancewic, S. (2015). Type-and-Example-Directed Program Synthesis. In *PLDI*.
- Feser, J., Chaudhuri, S., & Dillig, I. (2015). Synthesizing Data Structure Transformations from Input-Output Examples. In *PLDI*.
- ...

Program Synthesis meets Software Engineering

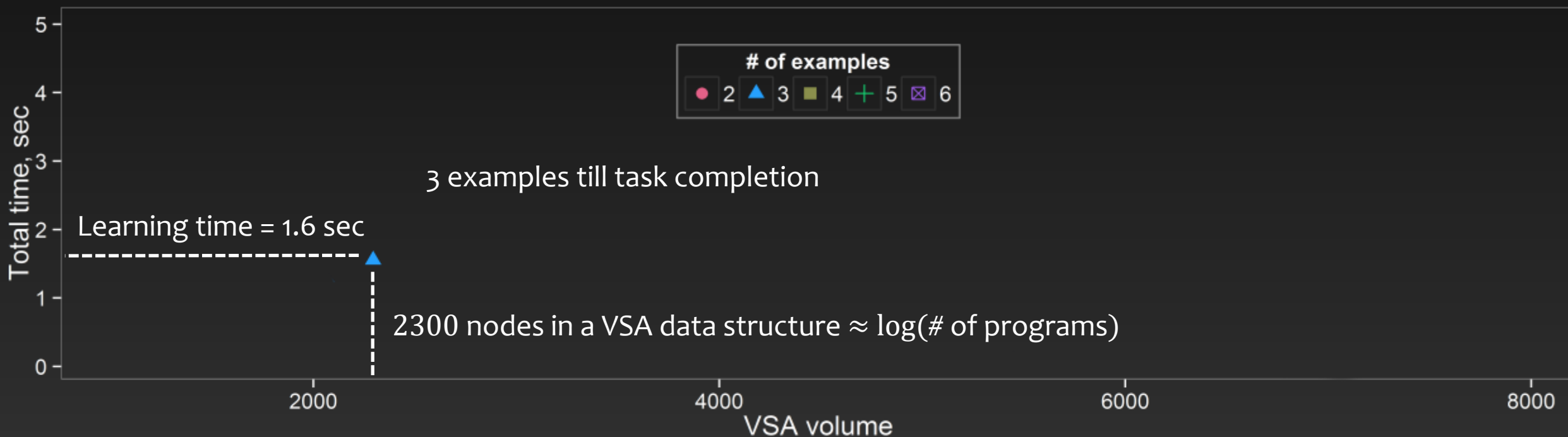
Project	Reference	Lines of Code		Development Time	
		Original	PROSE	Original	PROSE
Flash Fill	POPL 2010	12K	3K	9 months	1 month
Text Extraction	PLDI 2014	7K	4K	8 months	1 month
Text Normalization	IJCAI 2015	17K	2K	7 months	2 months
Spreadsheet Layout	PLDI 2015	5K	2K	8 months	1 month
Web Extraction	—	—	2.5K	—	1.5 months

Performance: 0.5 – 3X Original

More general \Rightarrow **Slower**

Algorithmic advances \Rightarrow **Faster**

Example: FlashExtract

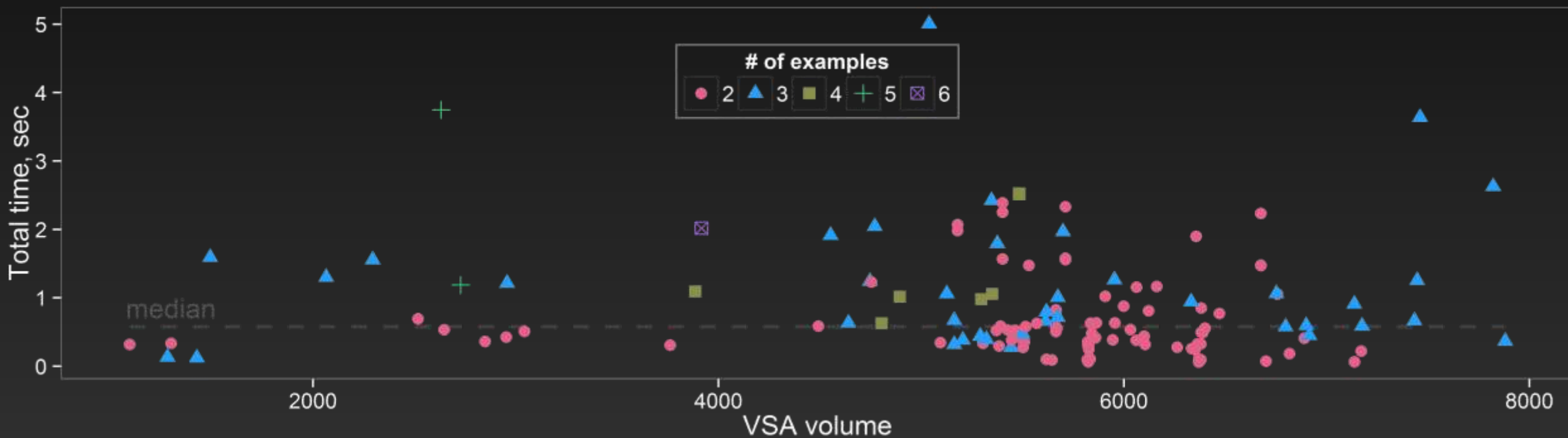


Performance: 0.5 – 3X Original

More general \Rightarrow **Slower**

Algorithmic advances \Rightarrow **Faster**

Example: FlashExtract





Applications

Email Parsing in Cortana

Junk | Sweep Move to Categories ...

All

3:38 PM Promotions

3:14 PM

3:01 PM

2:42 PM Newsletters

Alaska Airlines 454 Departs in 3 weeks
Confirmation #X322G14

SEA LAX

5:30 AM Wednesday 10/4/2015
Seattle
Terminal - Gate -
Seat -

8:10 AM Wednesday 10/4/2015
Los Angeles
Terminal - Gate -

[View in Calendar](#)

Alaska Airlines 1021 LAX → LIH Oct 4th, 2015 @ 9:15 AM

Tue 10/13/2015, 2 hrs 40 min

17801 International Blvd, Seattle, WA 98158

Flight to Los Angeles
Alaska Airlines AS454
Conf #X322G14

Status: Check-in is now possible

SEA LAX

5:30 am	8:10 am
Oct 13	Oct 13
Seattle	Los Angeles
Terminal D, Gate 2	Terminal 6, Gate 66
Seat 22A	

[View Email](#) [Check-in](#)

ConvertFrom-String in PowerShell

```
PS C:\> $template = '@'

  1      {Time*:1} ms      <1 ms      <1 ms      cusred024ca901-tengige0-007-13.network.microsoft.com [{IP:10.31.196.2}]
 10      {Time*:2} ms      1 ms       1 ms       {IP:104.44.81.80}
 12      2 ms          2 ms       1 ms       a-0001.a-msedge.net [204.79.197.200]

'@

PS C:\> tracert bing.com | ConvertFrom-String -TemplateContent $template

Time IP
---- --
1     10.31.196.2
1     10.37.1.174
1     10.37.66.201
1     10.37.44.94
1     10.37.67.230
1     10.37.45.69
1     131.107.202.162
2     131.107.200.18
1     207.46.36.105
2     104.44.81.80
1     10.201.196.145
1     204.79.197.200
```

Research:

<https://microsoft.github.io/prose>

Play:

<https://microsoft.github.io/prose/demo>

Contact:

prose-contact@microsoft.com

See our demo @ MSR table:

Thank you!

Questions?

The screenshot shows the PROSE Demo web application interface. The main content is a table with columns: Super Bowl, Year, Month, Team1, Team2, and Location. The table lists Super Bowls from I to XLIX. The interface includes a navigation bar with 'Log out' and 'About' buttons, and a sidebar with various tool icons.

Super Bowl	Year	Month	Team1	Team2	Location
Super Bowl I	1957	January	Green Bay Packers 01	15-10	Kansas City Chiefs 01
Super Bowl II	1958	January	Green Bay Packers 02	13-14	Oakland Raiders 01
Super Bowl III	1959	January	New York Jets 01	16-7	Indianapolis Colts 01
Super Bowl IV	1970	January	Kansas City Chiefs 02	23-7	Minnesota Vikings 01
Super Bowl V	1971	January	Indianapolis Colts 02	16-13	Dallas Cowboys 01
Super Bowl VI	1972	January	Dallas Cowboys 02	24-3	Miami Dolphins 01
Super Bowl VII	1973	January	Atlanta Falcons 01	14-7	Washington Redskins 01
Super Bowl VIII	1974	January	Minnesota Vikings 02	34-7	Minnesota Vikings 02
Super Bowl IX	1975	January	Pittsburgh Steelers 01	16-6	Minnesota Vikings 03
Super Bowl X	1976	January	Pittsburgh Steelers 02	21-17	Dallas Cowboys 02
Super Bowl XI	1977	January	Oakland Raiders 02	32-14	Minnesota Vikings 04
Super Bowl XII	1978	January	Dallas Cowboys 03	27-10	Denver Broncos 01
Super Bowl XIII	1979	January	Pittsburgh Steelers 03	35-31	Dallas Cowboys 03
Super Bowl XIV	1980	January	Pittsburgh Steelers 04	31-19	St. Louis Rams 01
Super Bowl XV	1981	January	Oakland Raiders 03	27-10	Philadelphia Eagles 01
Super Bowl XVI	1982	January	San Francisco 49ers 01	28-21	Cincinnati Bengals 01
Super Bowl XVII	1983	January	Washington Redskins 02	37-17	Miami Dolphins 02
Super Bowl XVIII	1984	January	Oakland Raiders 04	38-9	Washington Redskins 02
Super Bowl XIX	1985	January	San Francisco 49ers 02	38-9	Miami Dolphins 03
Super Bowl XX	1986	January	Chicago Bears 01	46-10	New England Patriots 01
Super Bowl XXI	1987	January	New York Giants 01	39-20	Denver Broncos 02
Super Bowl XXII	1988	January	Washington Redskins 03	42-10	Denver Broncos 03
Super Bowl XXIII	1989	January	San Francisco 49ers 03	20-16	Cincinnati Bengals 02
Super Bowl XXIV	1990	January	San Francisco 49ers 04	55-10	Denver Broncos 04
Super Bowl XXV	1991	January	New York Giants 02	20-19	Buffalo Bills 01
Super Bowl XXVI	1992	January	Washington Redskins 04	37-24	Buffalo Bills 02
Super Bowl XXVII	1993	January	Dallas Cowboys 04	52-17	Buffalo Bills 03
Super Bowl XXVIII	1994	January	Dallas Cowboys 05	30-13	Buffalo Bills 04
Super Bowl XXIX	1995	January	San Francisco 49ers 05	49-26	San Diego Chargers 01
Super Bowl XXX	1996	January	Dallas Cowboys 06	27-17	Pittsburgh Steelers 05
Super Bowl XXXI	1997	January	Green Bay Packers 03	35-21	New England Patriots 02
Super Bowl XXXII	1998	January	Denver Broncos 05	31-24	Green Bay Packers 04
Super Bowl XXXIII	1999	January	Denver Broncos 06	34-19	Atlanta Falcons 01
Super Bowl XXXIV	2000	January	St. Louis Rams 02	23-16	Tennessee Titans 01
Super Bowl XXXV	2001	January	Baltimore Ravens 01	34-7	New York Giants 02
Super Bowl XXXVI	2002	February	New England Patriots 01	20-17	St. Louis Rams 02
Super Bowl XXXVII	2003	January	Tampa Bay Buccaneers 01	48-21	Oakland Raiders 05
Super Bowl XXXVIII	2004	February	New England Patriots 02	32-29	Cleveland Panthers 01
Super Bowl XXXIX	2005	February	New England Patriots 03	24-21	Philadelphia Eagles 02
Super Bowl XL	2006	February	Pittsburgh Steelers 06	21-10	Seattle Seahawks 01
Super Bowl XLI	2007	February	Indianapolis Colts 03	28-23	Chicago Bears 02

