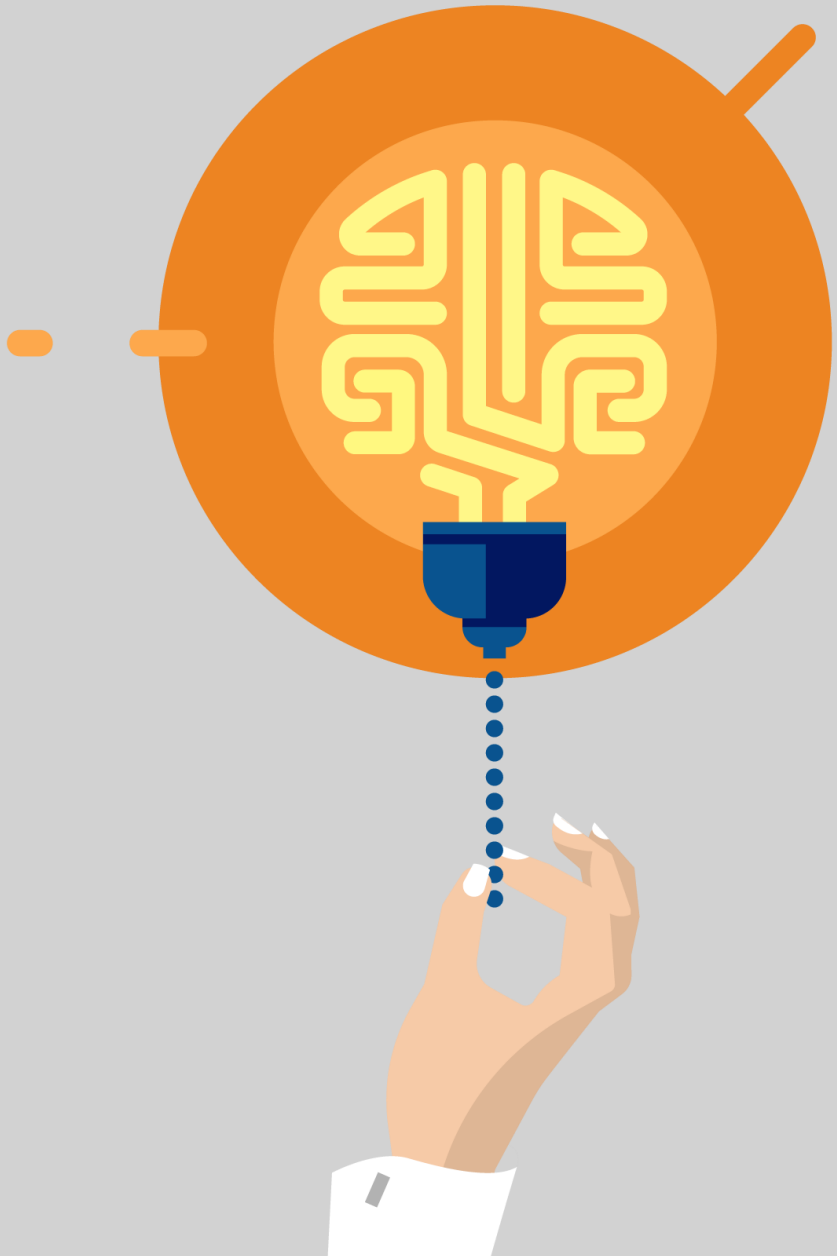
Microsoft

# Towards Learned Program Analyses with Machine Learning

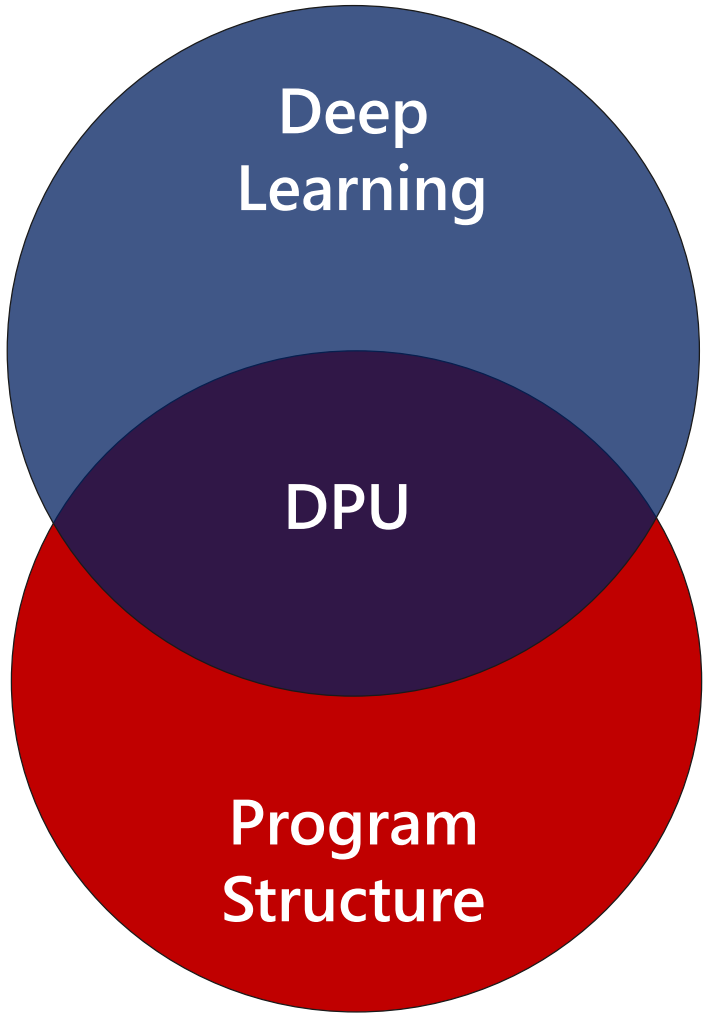
Miltos Allamanis

Microsoft Research Cambridge

Joint work with Earl T. Barr, Marc Brockschmidt, Santanu Dash, Mahmoud Khademi



# Deep Program Understanding




✓ Understands images/language/speech
✓ Finds patterns in noisy data

- Requires many samples
- Handling structured data is hard

✓ Interpretable
✓ Generalisation verifiable

- Manual effort
- Limited to specialists

Microsoft

# Source Code and Natural Language

# Code as…

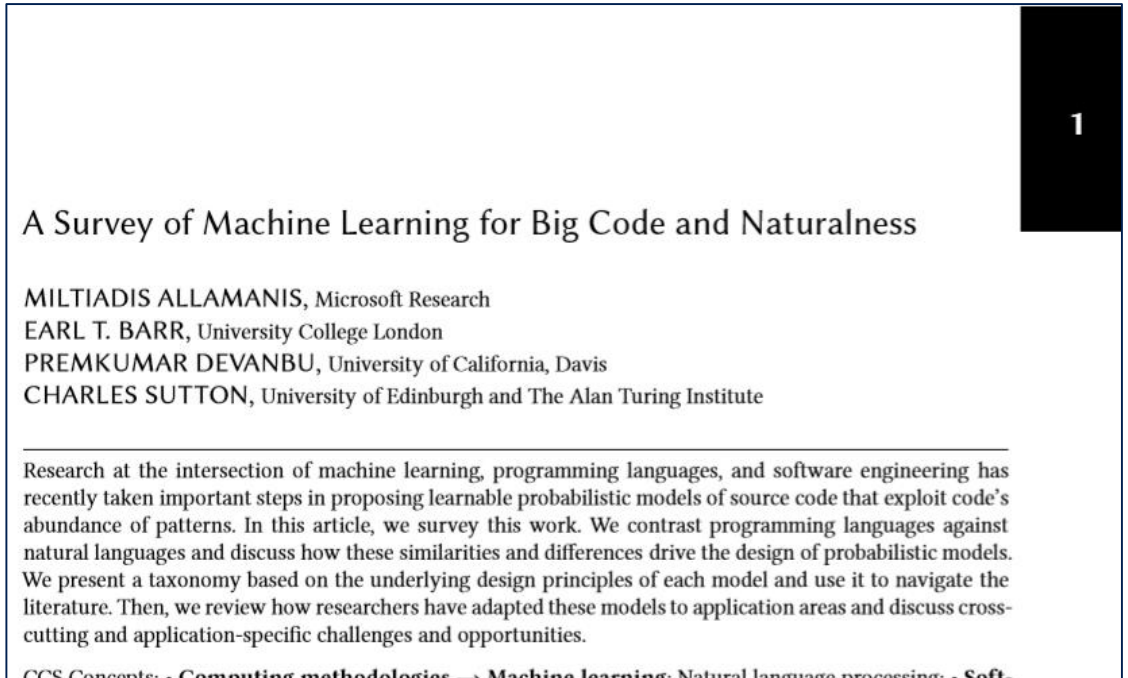
Data → Software Engineering (SE) Tools

Machine Learning (ML) component → Artificial Intelligence (AI) Tool

# Research in ML4Code

- Infer latent intent
- Ambiguous information

https://ml4code.github.io



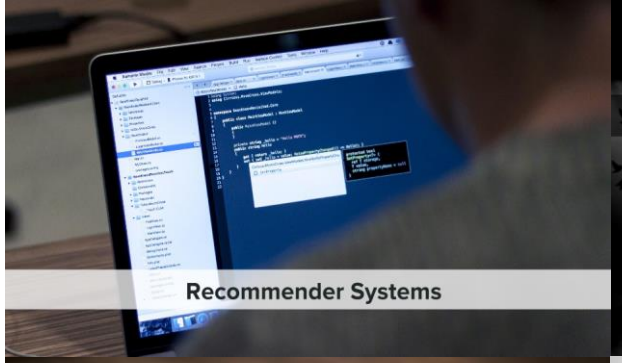
1

A Survey of Machine Learning for Big Code and Naturalness

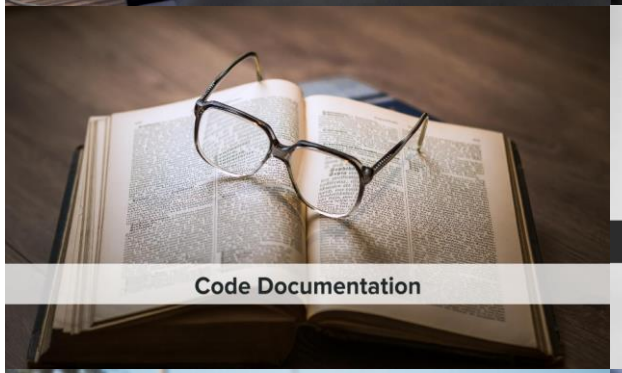
MILTIADIS ALLAMANIS, Microsoft Research
EARL T. BARR, University College London
PREMKUMAR DEVANBU, University of California, Davis
CHARLES SUTTON, University of Edinburgh and The Alan Turing Institute

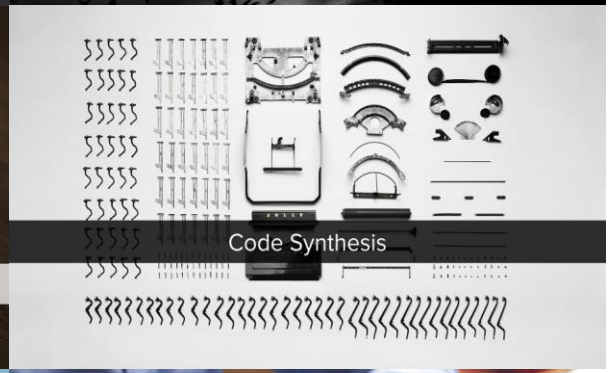
Research at the intersection of machine learning, programming languages, and software engineering has recently taken important steps in proposing learnable probabilistic models of source code that exploit code's abundance of patterns. In this article, we survey this work. We contrast programming languages against natural languages and discuss how these similarities and differences drive the design of probabilistic models. We present a taxonomy based on the underlying design principles of each model and use it to navigate the literature. Then, we review how researchers have adapted these models to application areas and discuss cross-cutting and application-specific challenges and opportunities.

CCS Concepts: • Computing methodologies → Machine learning; Natural language processing; • Soft-

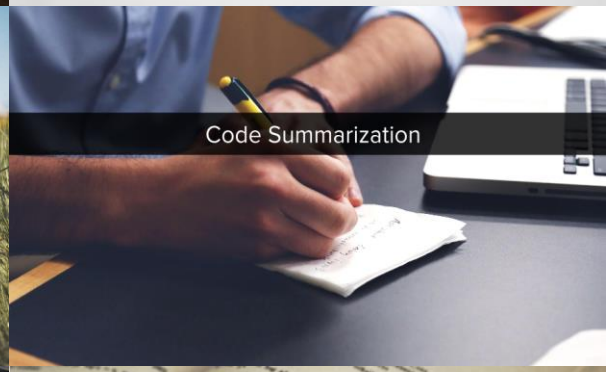


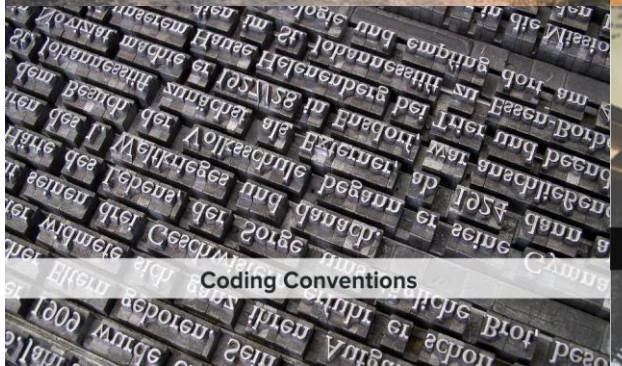


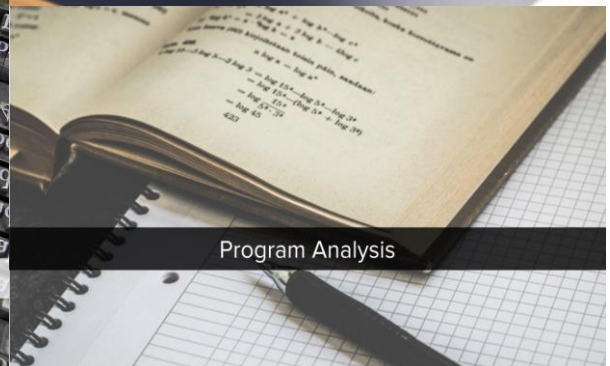






Microsoft

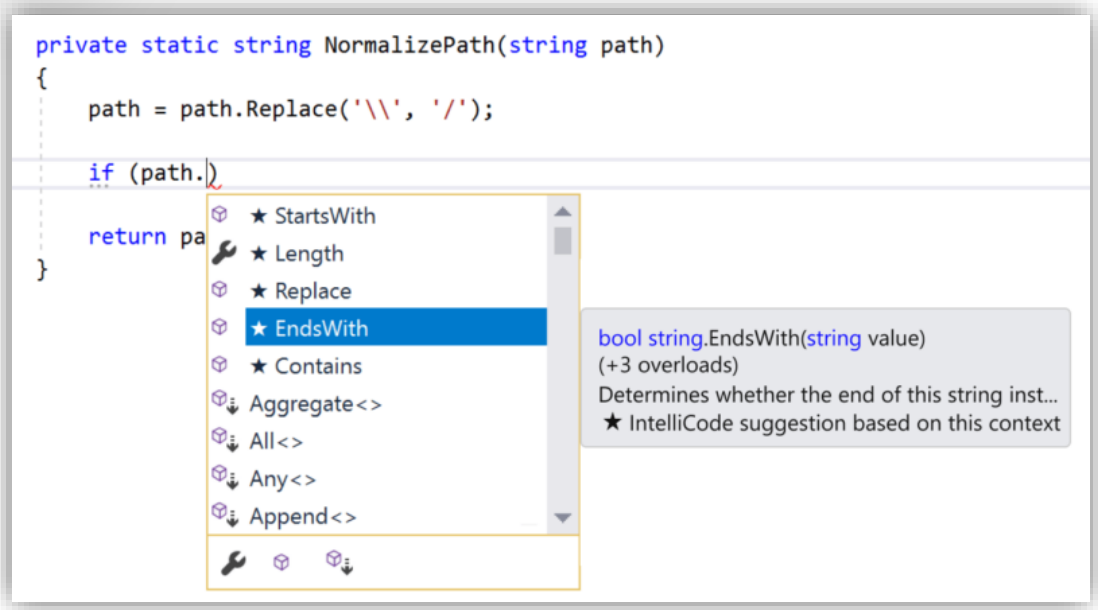
# Highlights: Program Analysis with Machine Learning

# Code Autocompletion

```
Text text = new Text(parent, SWT.NONE);
text.
```

setLayoutData(Object layoutData) : void - Control - 86%
setText(String string) : void - Text - 51%
addModifyListener(ModifyListener listener) : void - Text - 31%
getText() : String - Text - 12%
setEnabled(boolean enabled) : void - Control - 11%
setFont(Font font) : void - Control - 8%
Press '^Space' to show Adaptive Template Proposals (Code Recommenders)

http://www.eclipse.org/recommenders/




https://visualstudio.microsoft.com/services/intellicode/

# Variable Renaming

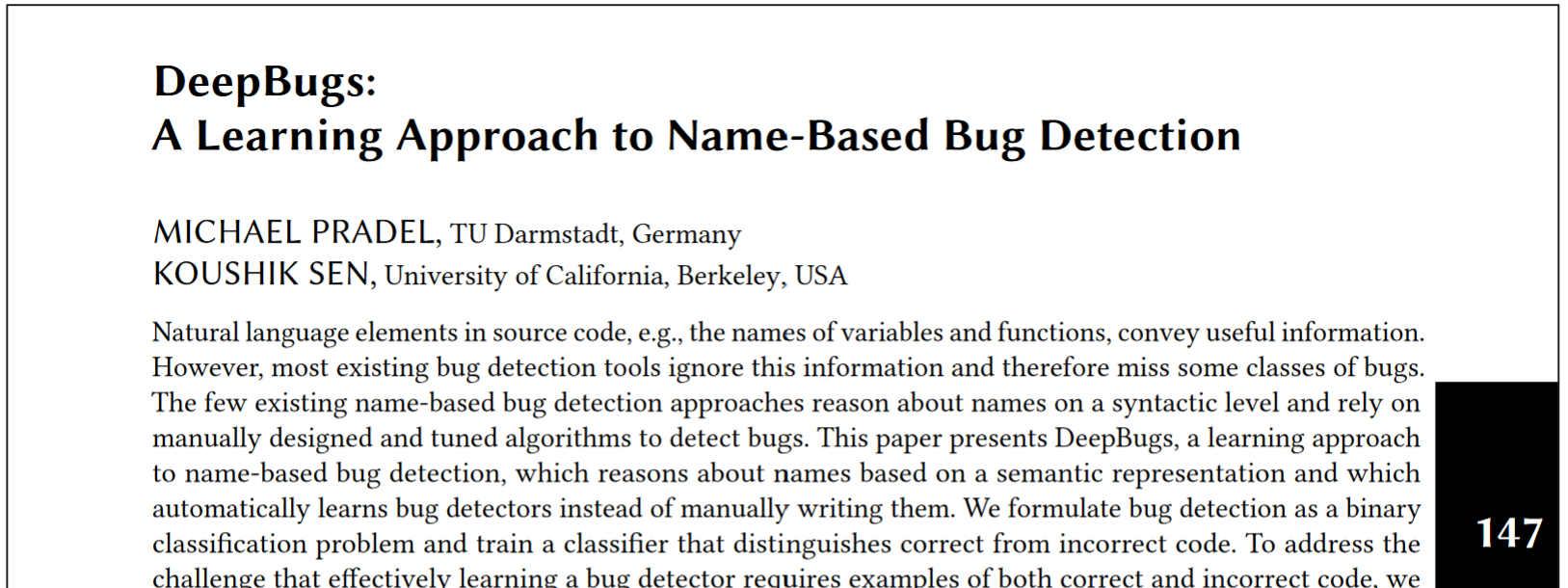
```
public class TextRunnerTest extends TestCase {
    void execTest(String testClass, boolean success) throws Exception {
        ...
        InputStream [redacted] = p.getInputStream();
        while ([redacted].read()) != -1);
        ...
    }
    ...
}
```

**Suggested Name**

`input (81.9%)`

# Argument Swapping

```
setSize(width, height) or setSize(height, width)

promise.done(res, err) or promise.done(err, res)
```



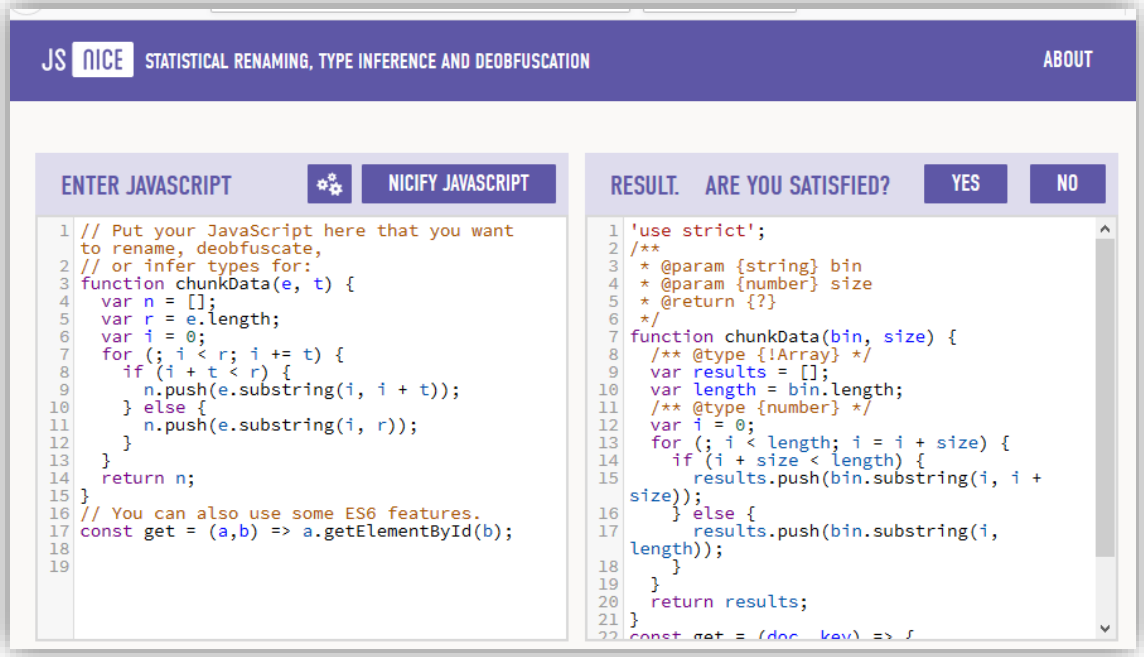
**DeepBugs:**
**A Learning Approach to Name-Based Bug Detection**

MICHAEL PRADEL, TU Darmstadt, Germany
KOUSHIK SEN, University of California, Berkeley, USA

Natural language elements in source code, e.g., the names of variables and functions, convey useful information. However, most existing bug detection tools ignore this information and therefore miss some classes of bugs. The few existing name-based bug detection approaches reason about names on a syntactic level and rely on manually designed and tuned algorithms to detect bugs. This paper presents DeepBugs, a learning approach to name-based bug detection, which reasons about names based on a semantic representation and which automatically learns bug detectors instead of manually writing them. We formulate bug detection as a binary classification problem and train a classifier that distinguishes correct from incorrect code. To address the challenge that effectively learning a bug detector requires examples of both correct and incorrect code, we

147

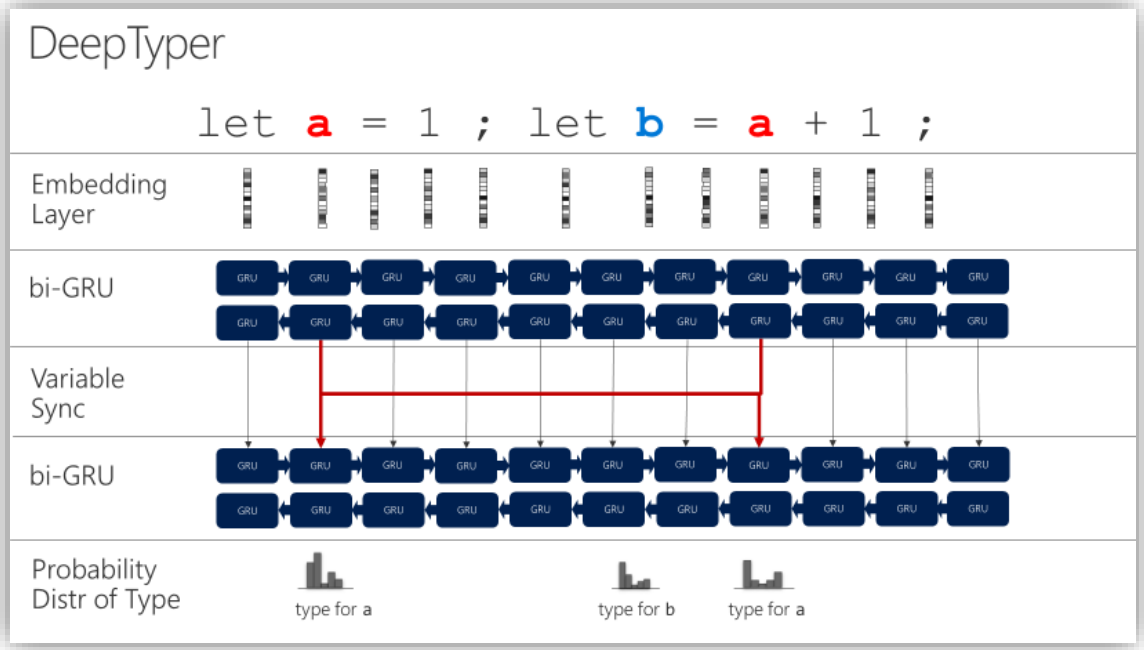
# Predicting Types




## Predicting Program Properties from Code

V. Raychev, M. Vechev, A. Krause. 2015

http://jsnice.org/




## Deep Learning Type Inference

V. Hellendoorn, C. Bird, E.T. Barr, M. Allamanis. 2018

Microsoft

- Variable Misuse
- Learning Nominal Type Refinements

# Variable Misuse

```
// Create or update the document.
var newDocument = await cosmosClient.UpsertDocumentAsync(cosmosDbCollectionUri, document);

if (updateRecord)
{
    logger.WriteLog($"Updated {existingDocument} to {newDocument}");
}
else
{
    logger.WriteLog($"Added {existingDocument}");
```

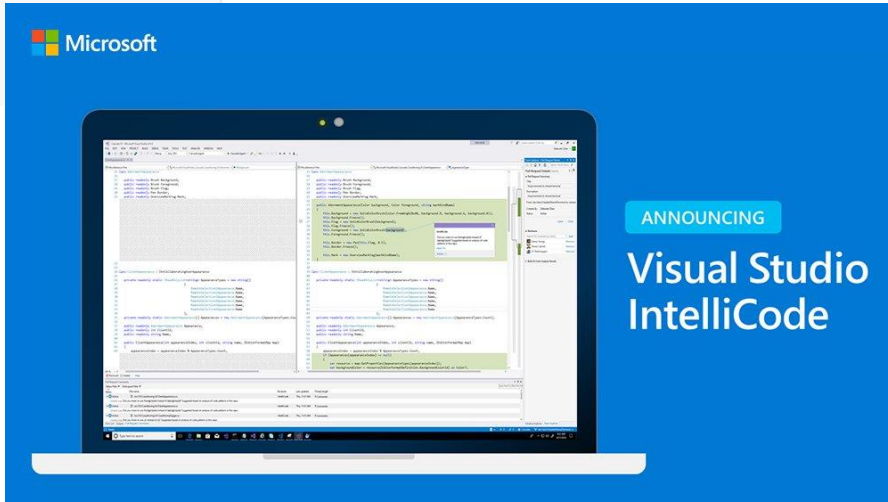
[redacted] Update 1 1 Resolved

Based on this repo's code patterns, did you intend to use 'newDocument' (confidence 92%) rather than 'existingDocument` (confidence 7%) here? Review is recommended by Research bot's Variable Misuse analysis.

JK [redacted]

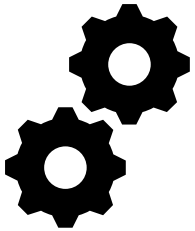
+1

Allamanis *et al.* "Learning to Represent Programs with Graphs". 2018



# Inferring Type Refinements

## Conceptual Types



## Defined Types

*"a password"* → `string password;`

*"a JSON string"* → `string data = Json.Load();`

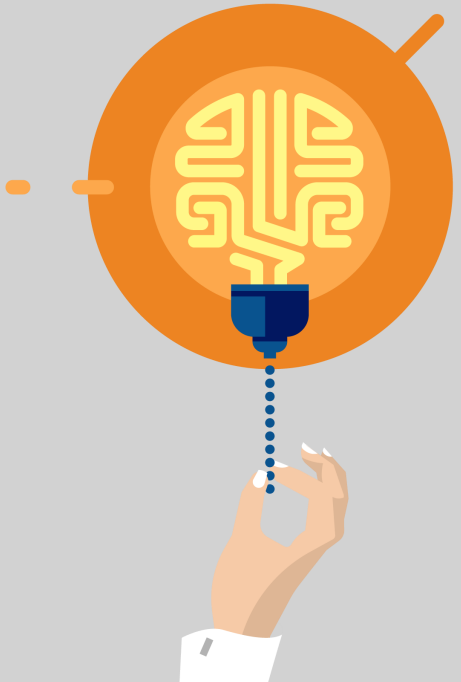
Latent; we don't observe the *conceptual* types.

Defined explicitly by the programmer.

Dash *et al.* 2018 "RefiNym: Using Names to Refine Types"

# Variable Misuse

with Graph Neural Networks



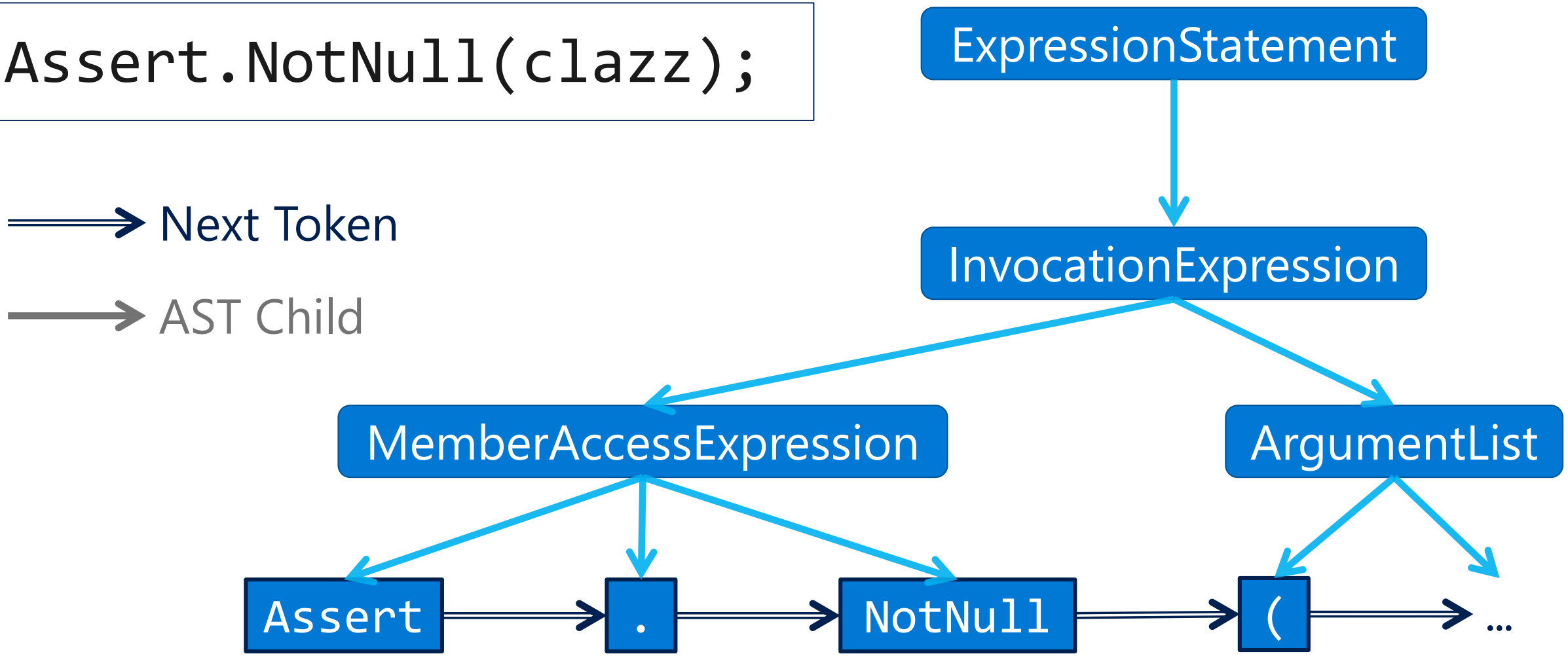
*Allamanis, Brockschmidt, Khademi. ICLR 2018*

# Programs as Graphs

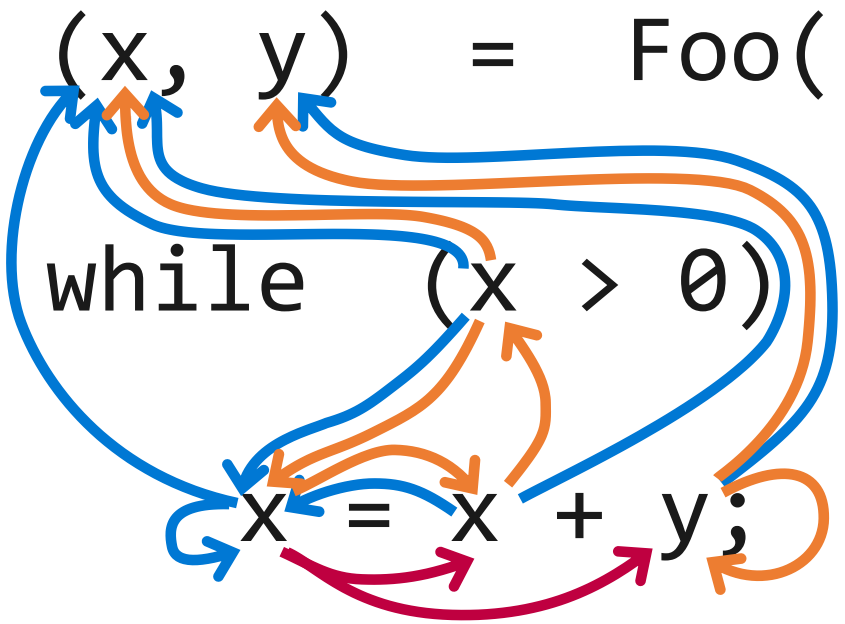
```
int SumPositive(int[] arr, int lim) {
  int sum = 0;
  for (int i = 0; i < lim; i++)
    if (arr[i] > 0)
        sum += arr[i];

  return sum;
}
```

# Programs as Graphs: Syntax

```
Assert.NotNull(clazz);
```



# Programs as Graphs: Data Flow



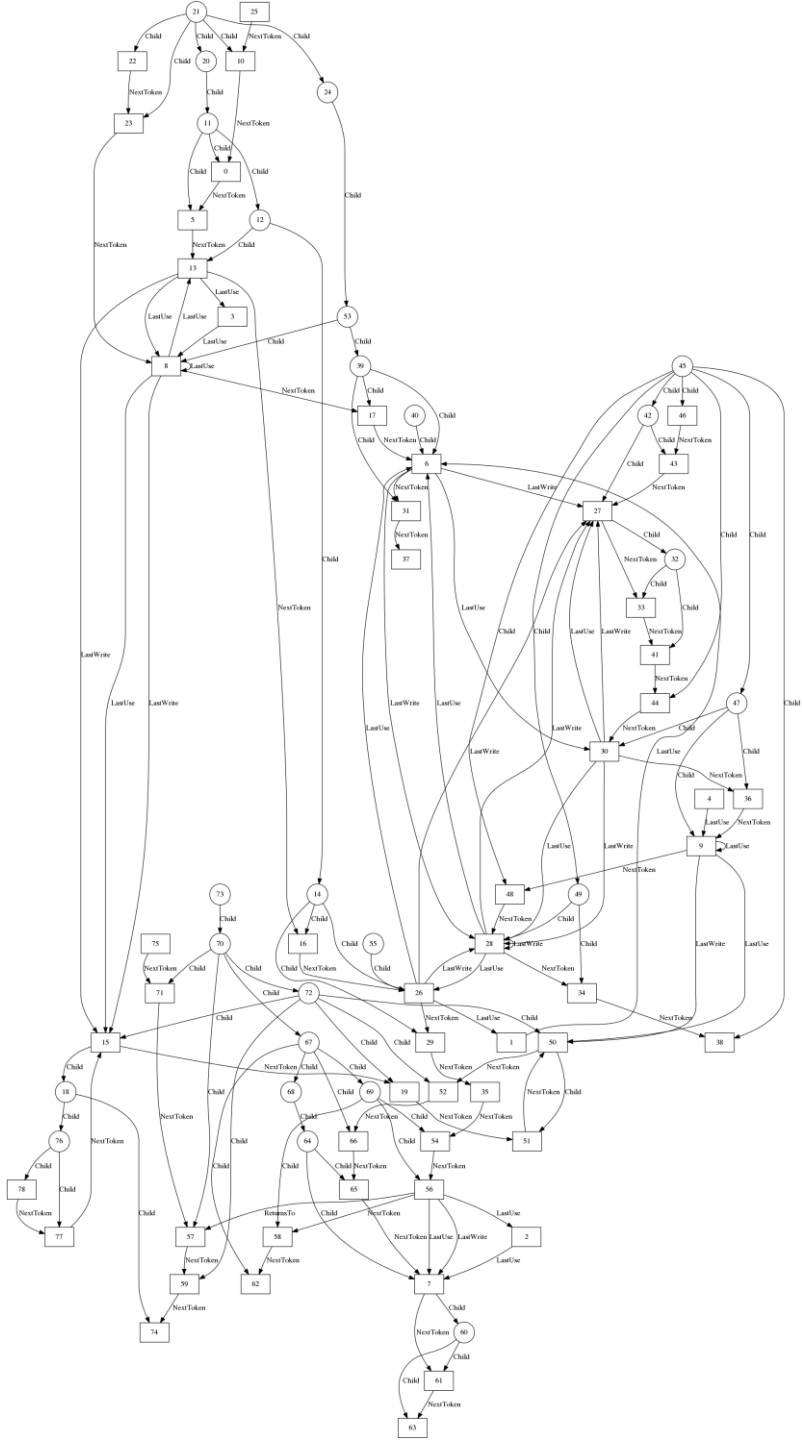

Last Write

Last Use

Computed From

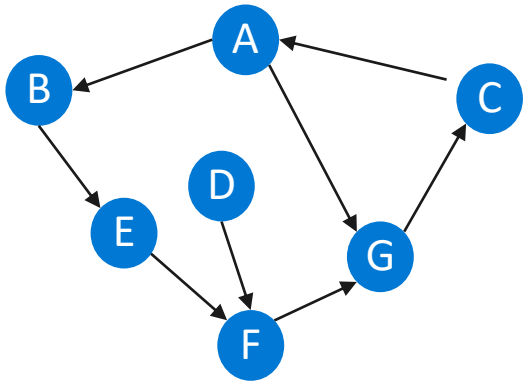
# Programs as Graphs

```
int SumPositive(int[] arr, int lim) {
  int sum = 0;
  for (int i=0; i<; ++)
    if ([]>0)
      +=[];
  return sum;
}
```

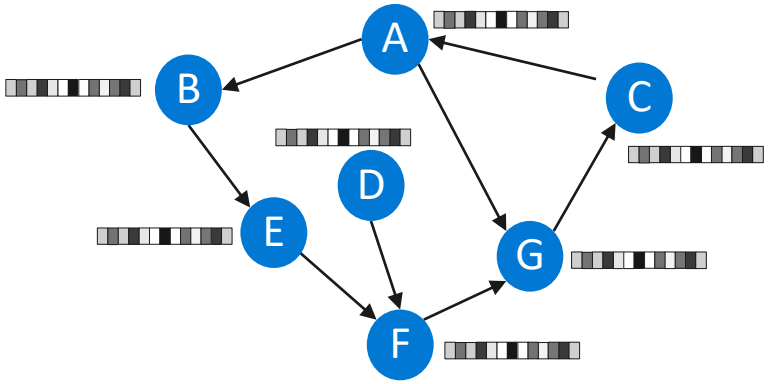


~900 nodes/graph  ~8k edges/graph

# Graph Neural Networks




Graph Representation
of Problem

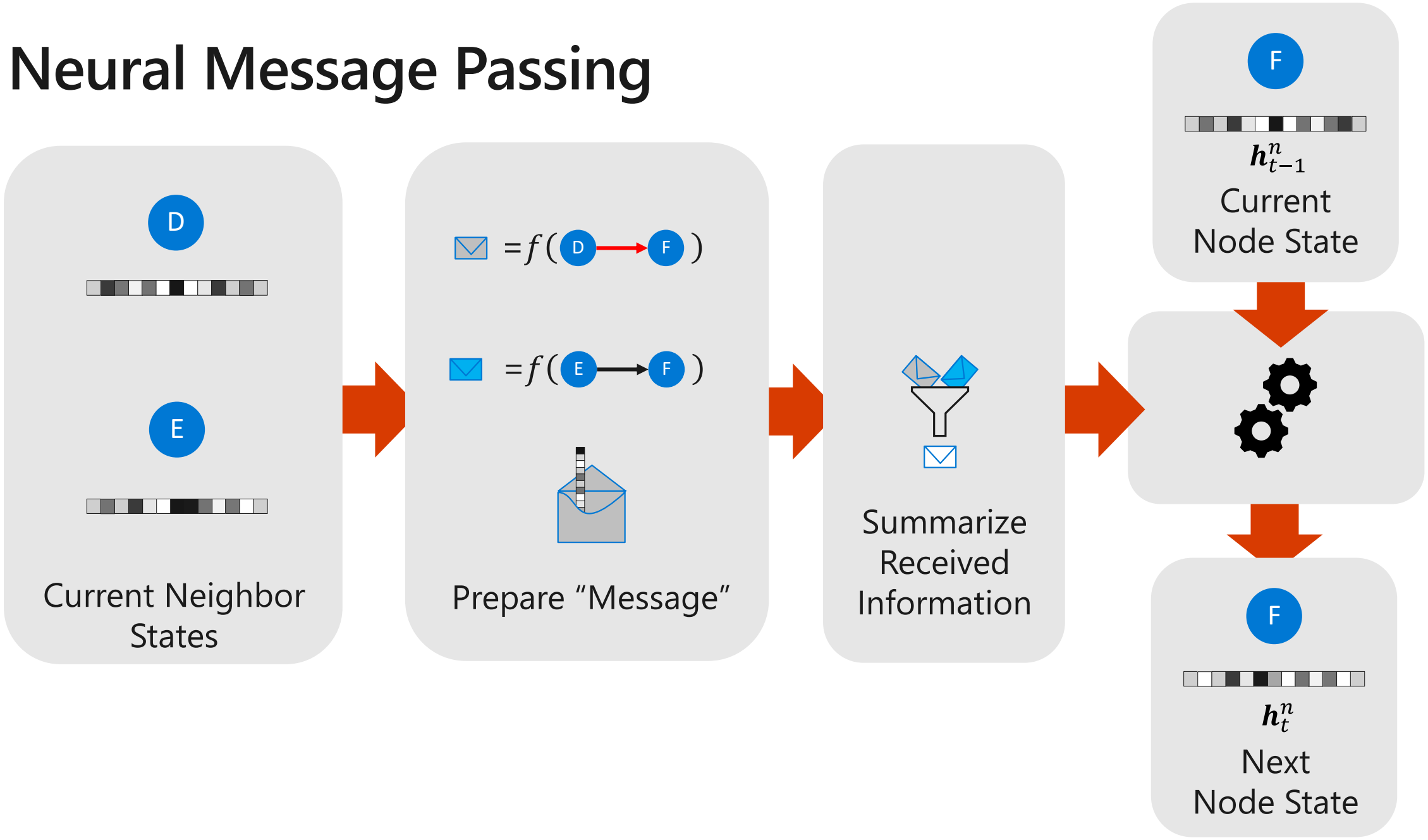



Initial Representation
of each node

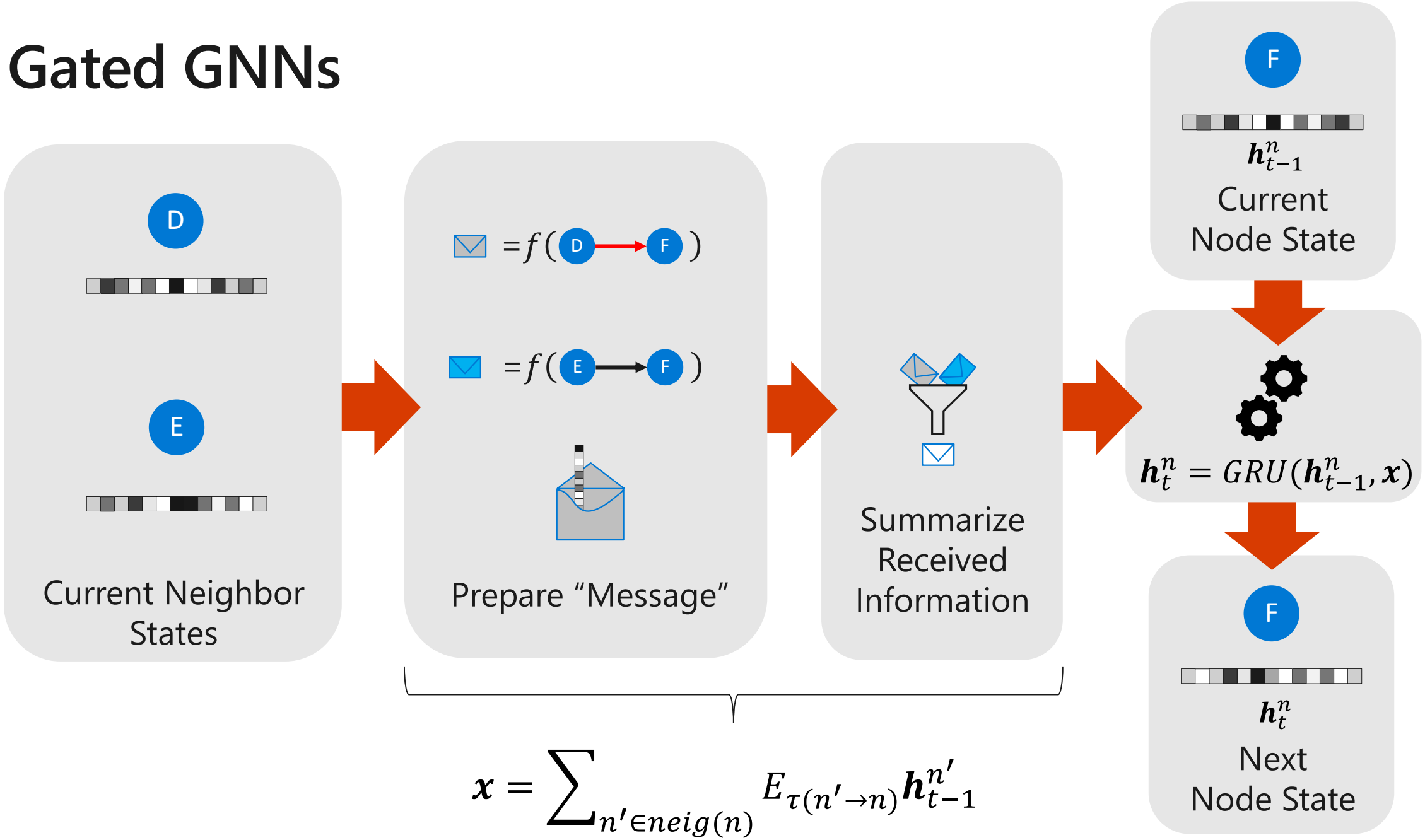
*Li et al (2015). Gated Graph Sequence Neural Networks.*

*Gilmer et al (2017). Neural Message Passing for Quantum Chemistry.*

# Neural Message Passing

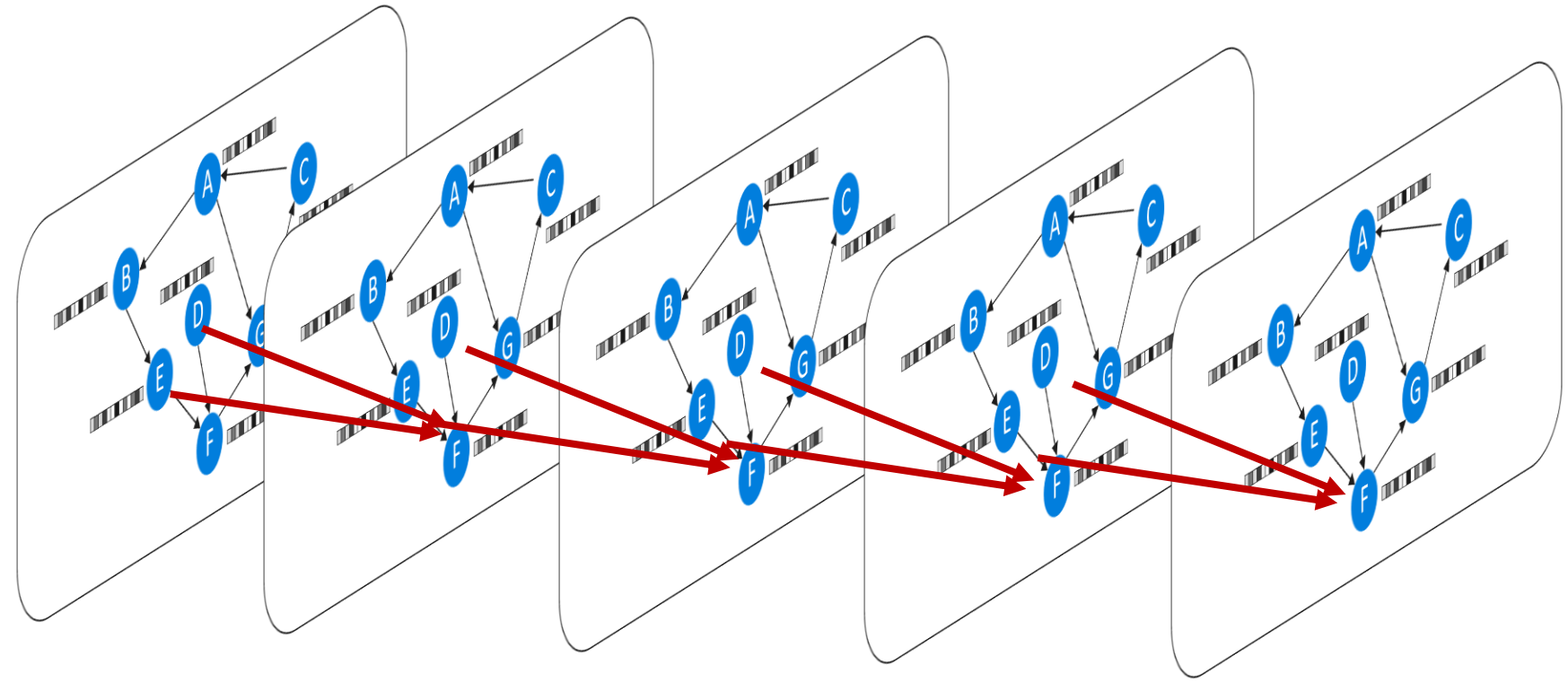


# Gated GNNs

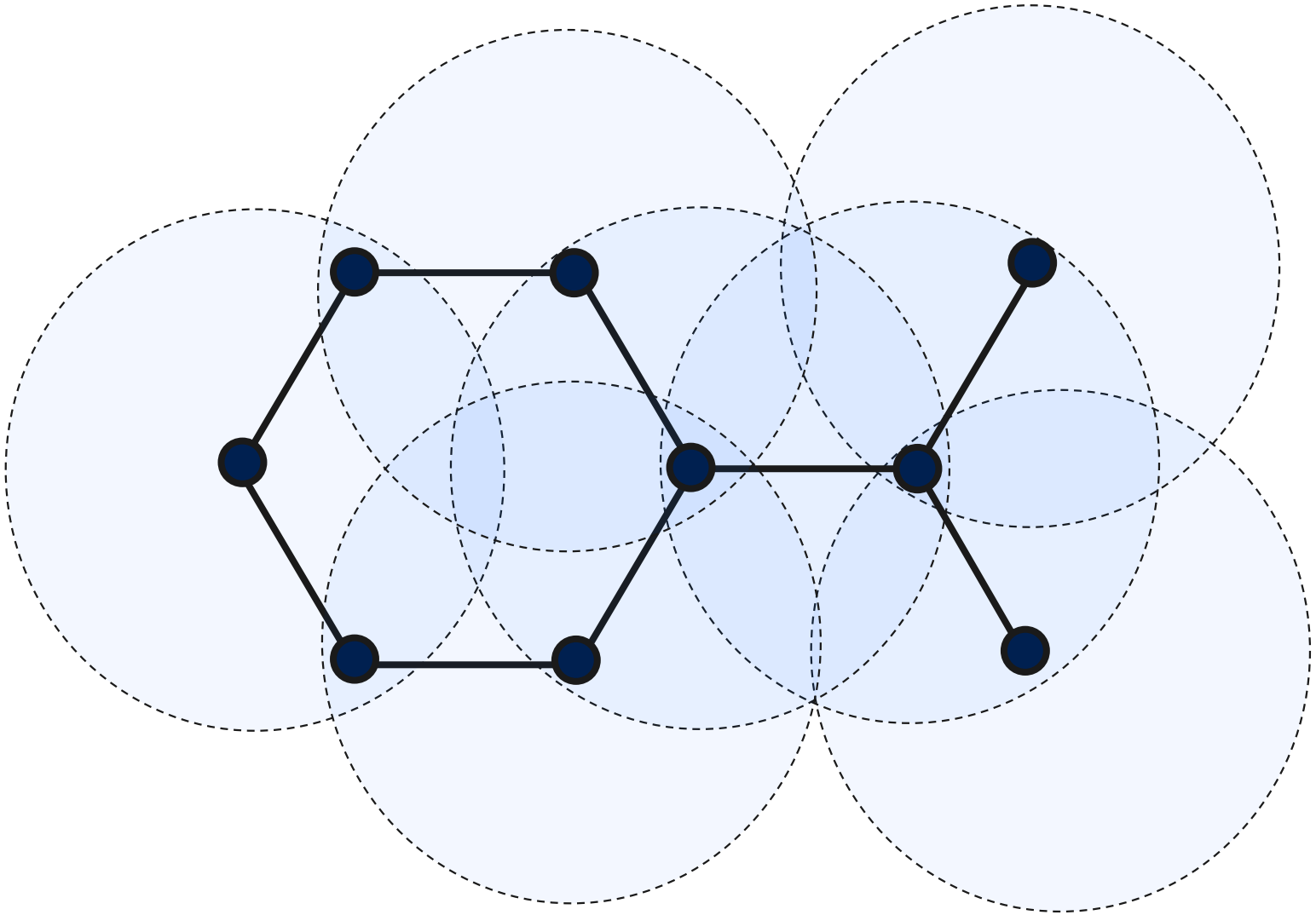



*Li et al (2015). Gated graph sequence neural networks.*

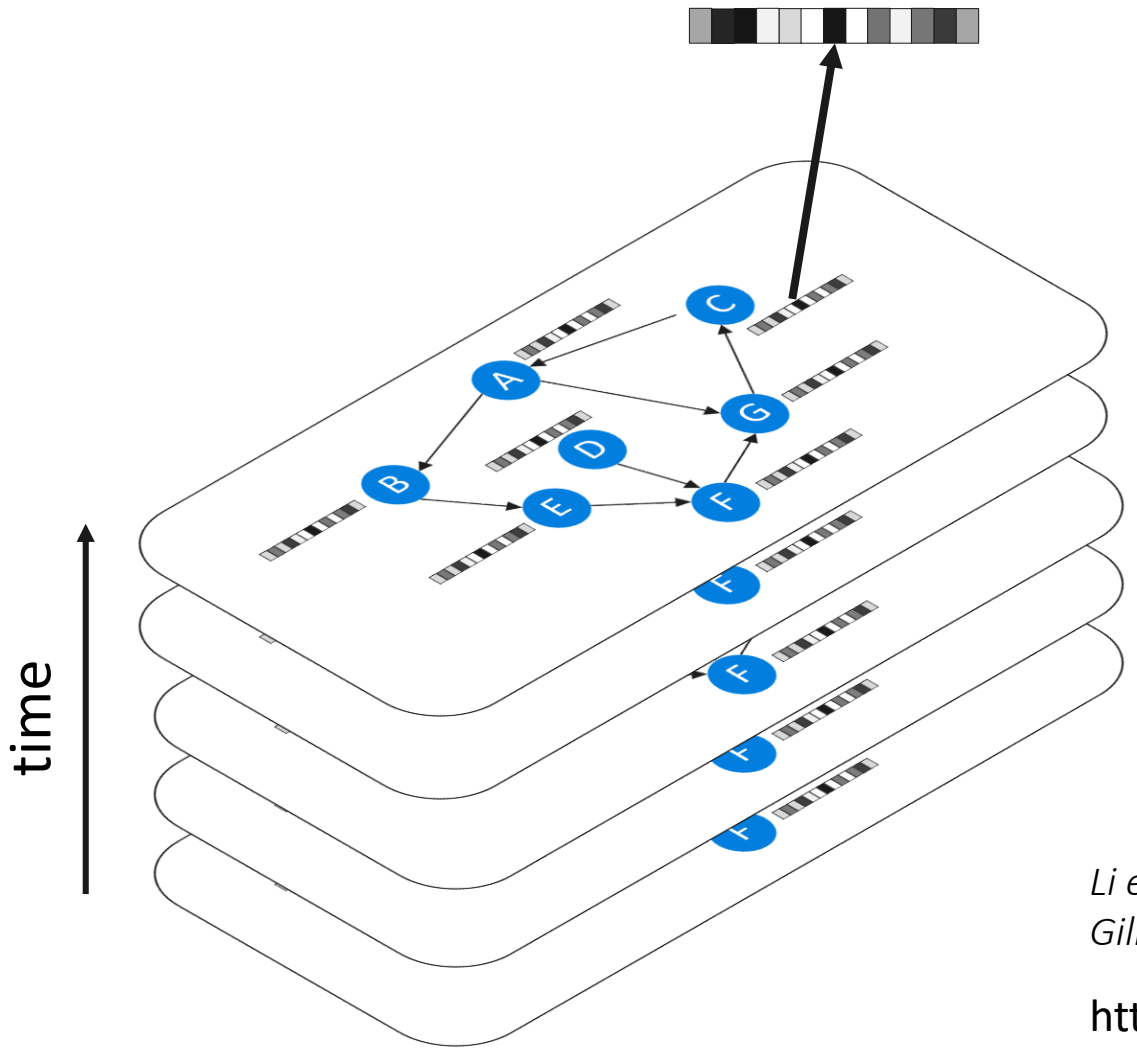
# Graph Neural Networks: Message Passing



# GNNs: Synchronous Message Passing (All-to-All)



# Graph Neural Networks: Output

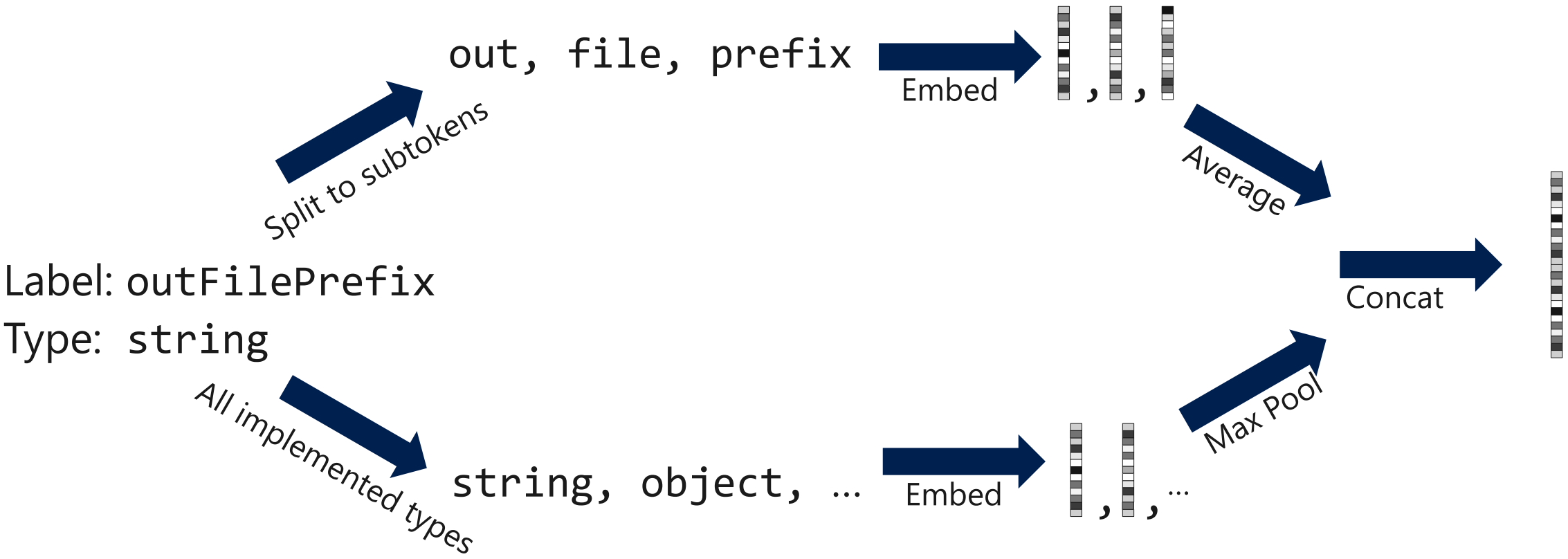



- node selection
- node classification
- graph classification

*Li et al (2015). Gated Graph Sequence Neural Networks.*
*Gilmer et al (2017). Neural Message Passing for Quantum Chemistry.*

https://github.com/Microsoft/gated-graph-neural-network-samples

# Initial Node Representations



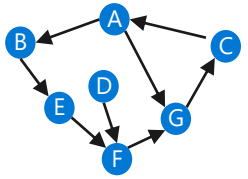
# Variable Misuse Task

```
var clazz=classTypes["Root"].Single() as JsonCodeGenerator.ClassType;
Assert.NotNull(clazz);

var first=classTypes["RecClass"].Single() as JsonCodeGenerator.ClassType;
Assert.NotNull(      );

Assert.Equal("string", first.Properties["Name"].Name);
Assert.False(clazz.Properties["Name"].IsArray);
```

Possible type-correct options: `clazz, first`

Not easy to catch with static analysis tools.

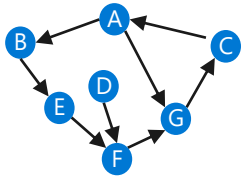
# Graph Representation for Variable Misuse




```
var clazz=classTypes["Root"].Single() as JsonCodeGenerator.ClassType;
Assert.NotNull(clazz);

var first=classTypes["RecClass"].Single() as JsonCodeGenerator.ClassType;
Assert.NotNull(      );

Assert.Equal("string", first.Properties["Name"].Name);
Assert.False(clazz.Properties["Name"].IsArray);
```

Possible type-correct options: `clazz, first`

# Graph Representation for Variable Misuse




```
var clazz=classTypes["Root"].Single() as JsonCodeGenerator.ClassType;
Assert.NotNull(clazz);

var first=classTypes["RecClass"].Single() as JsonCodeGenerator.ClassType;
Assert.NotNull( SLOT );

Assert.Equal("string", first.Properties["Name"].Name);
Assert.False(clazz.Properties["Name"].IsArray);
```

first

clazz

**Goal**: make the representation of SLOT as close as possible to the representation of the correct candidate node

$$f\left(\boldsymbol{h}_T^{SLOT}, \boldsymbol{h}_T^{first}\right) \gg f\left(\boldsymbol{h}_T^{SLOT}, \boldsymbol{h}_T^{clazz}\right)$$

# Dataset

2.9MLOC

GitHub

| Name | Git SHA | kLOCs | Slots | Vars | Description |
|---|---|---|---|---|---|
| Akka.NET | 719335a1 | 240 | 51.3k | 51.2k | Actor-based Concurrent & Distributed Framework |
| AutoMapper | 2ca7c2b5 | 46 | 3.7k | 10.7k | Object-to-Object Mapping Library |
| BenchmarkDotNet | 1670ca34 | 28 | 5.1k | 6.1k | Benchmarking Library |
| BotBuilder | 190117c3 | 44 | 6.4k | 8.7k | SDK for Building Bots |
| choco | 93985688 | 36 | 3.8k | 5.2k | Windows Package Manager |
| commandline† | 09677b16 | 11 | 1.1k | 2.3k | Command Line Parser |
| CommonMark.NET$^{Dev}$ | f3d54530 | 14 | 2.6k | 1.4k | Markdown Parser |
| Dapper | 931c700d | 18 | 3.3k | 4.7k | Object Mapper Library |
| EntityFramework | fa0b7ec8 | 263 | 33.4k | 39.3k | Object-Relational Mapper |
| Hangfire | ffc4912f | 33 | 3.6k | 6.1k | Background Job Processing Library |
| Humanizer† | cc11a77e | 27 | 2.4k | 4.4k | String Manipulation and Formatting |
| Lean† | f574bfd7 | 190 | 26.4k | 28.3k | Algorithmic Trading Engine |
| Nancy | 72e1f614 | 70 | 7.5k | 15.7 | HTTP Service Framework |
| Newtonsoft.Json | 6057d9b8 | 123 | 14.9k | 16.1k | JSON Library |
| Ninject | 7006297f | 13 | 0.7k | 2.1k | Code Injection Library |
| NLog | 643e326a | 75 | 8.3k | 11.0k | Logging Library |
| Opserver | 51b032e7 | 24 | 3.7k | 4.5k | Monitoring System |
| OptiKey | 7d35c718 | 34 | 6.1k | 3.9k | Assistive On-Screen Keyboard |
| orleans | e0d6a150 | 300 | 30.7k | 35.6k | Distributed Virtual Actor Model |
| Polly | 0afdbc32 | 32 | 3.8k | 9.1k | Resilience & Transient Fault Handling Library |
| quartznet | b33e6f86 | 49 | 9.6k | 9.8k | Scheduler |
| ravendb$^{Dev}$ | 55230922 | 647 | 78.0k | 82.7k | Document Database |
| RestSharp | 70de357b | 20 | 4.0k | 4.5k | REST and HTTP API Client Library |
| Rx.NET | 2d146fe5 | 180 | 14.0k | 21.9k | Reactive Language Extensions |
| scriptcs | f3cc8bcb | 18 | 2.7k | 4.3k | C# Text Editor |
| ServiceStack | 6d59da75 | 231 | 38.0k | 46.2k | Web Framework |
| ShareX | 718dd711 | 125 | 22.3k | 18.1k | Sharing Application |
| SignalR | fa88089e | 53 | 6.5k | 10.5k | Push Notification Framework |
| Wox | cdaf6272 | 13 | 2.0k | 2.1k | Application Launcher |

# Quantitative Results – Variable Misuse

| Accuracy (%) | BiGRU | BiGRU + Dataflow | GGNN |
|---|---|---|---|
| Seen Projects | 50.0 | 73.7 | **85.5** |

Seen Projects: 24 F/OSS C# projects (2060 kLOC): Used for train and test

3.8 type-correct alternative variables per slot (median 3, σ= 2.6)

# Quantitative Results – Variable Misuse

| Accuracy (%) | BiGRU | BiGRU + Dataflow | GGNN |
|---|---|---|---|
| Seen Projects | 50.0 | 73.7 | **85.5** |
| Unseen Projects | 28.9 | 60.2 | **78.2** |

Seen Projects: 24 F/OSS C# projects (2060 kLOC): Used for train and test
Unseen Projects: 3 F/OSS C# projects (228 kLOC): Used only for test
3.8 type-correct alternative variables per slot (median 3, $\sigma = 2.6$)

```
        // Create or update the document.
        var newDocument = await cosmosClient.UpsertDocumentAsync(cosmosDbCollectionUri, document);

        if (updateRecord)
        {
            logger.WriteLog($"Updated {existingDocument} to {newDocument}");
        }
        else
        {
            logger.WriteLog($"Added {existingDocument}");
```

[redacted] Update 1

1 Resolved

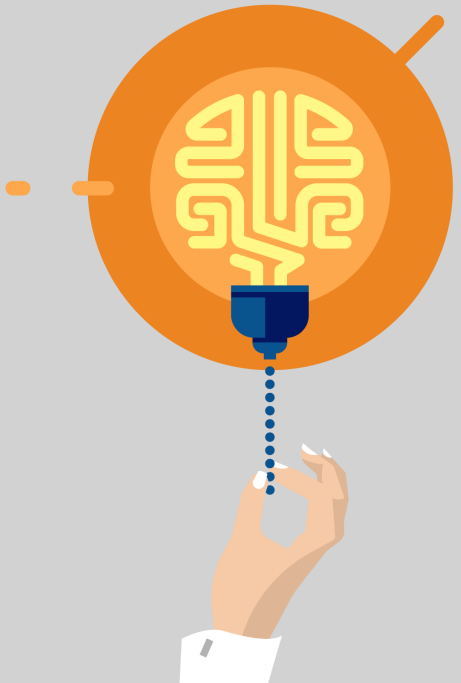
Based on this repo's code patterns, did you intend to use 'newDocument' (confidence 92%) rather than 'existingDocument` (confidence 7%) here? Review is recommended by Research bot's Variable Misuse analysis.

JK [redacted]

+1

# Refining Types

...using natural language.



*Dash, Allamanis, Barr. FSE 2018*

# Motivating Example – Primitive Obsession

```
def addToCart(cartId: String, productId: String, providerId: String)
```

```
addToCart(productId, providerId, cartId)
```

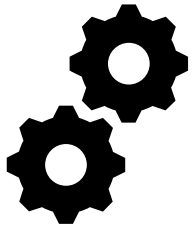
---

```
username := password
```

---

```
temperature + numOfOranges
```

# Conceptual Types



# Defined Types

*"a password"* → `string password;`

*"a JSON string"* → `string data = Json.Load();`

Latent; we don't observe the *conceptual* types.

Defined explicitly by the programmer.

# RefiNym: Nominal Refinements

```
string EncryptAndSignCookie(string cookieValue, FormsAuthenticationConfiguration config) {
        string encryptedCookie =
                config.CryptographyConfiguration.EncryptionProvider.Encrypt(cookieValue);

        var hmacBytes = GenerateHmac(encryptedCookie, config);
        string hmacString = Convert.ToBase64String(hmacBytes);
        return hmacString + encryptedCookie;
}
```

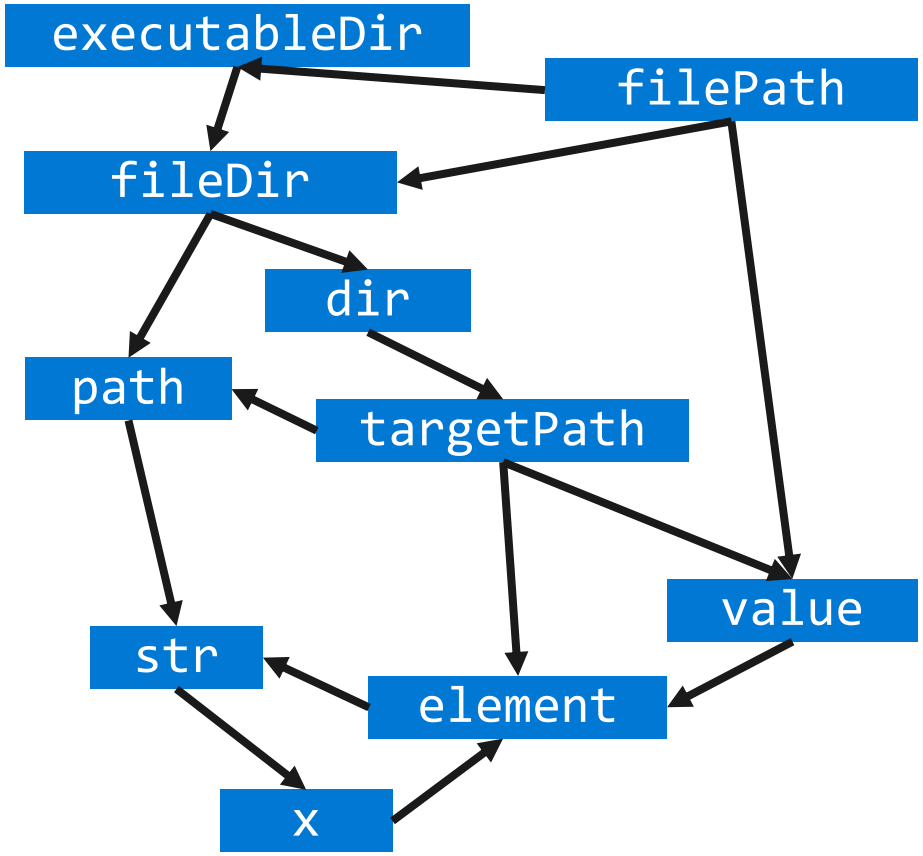
EncryptedString

UnencryptedString

EncryptedString

EncryptedString

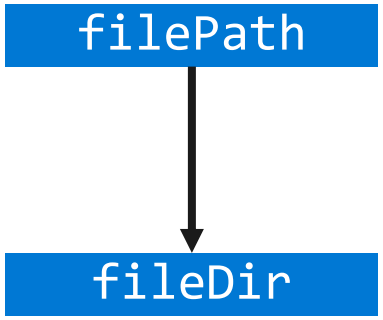
# Name Flow Graphs



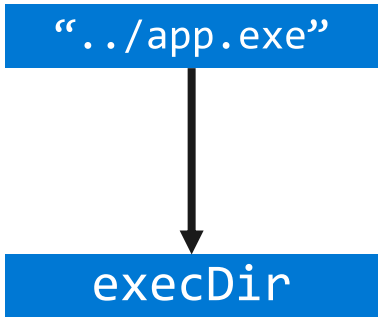

**Representation**:

- [Static] Data Flow
- Identifier Names

# Constructing Name Flows – Assignment

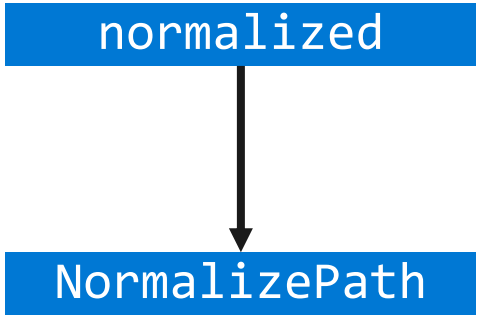



```
fileDir = filePath
```



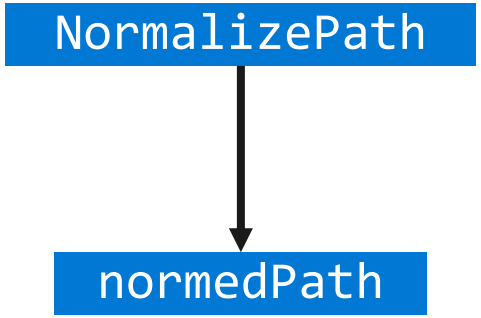

```
execDir = “../app.exe”
```

# Constructing Name Flows – Returns



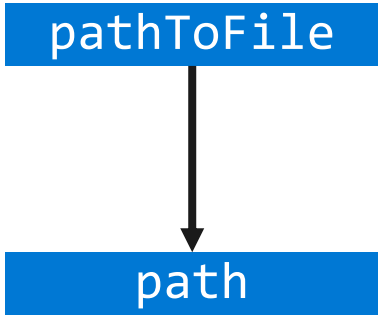

```
def NormalizePath(…){
    …

    return normalized;
}
```

# Constructing Name Flows – Function Calls




```
normedPath = NormalizePath(…)
```

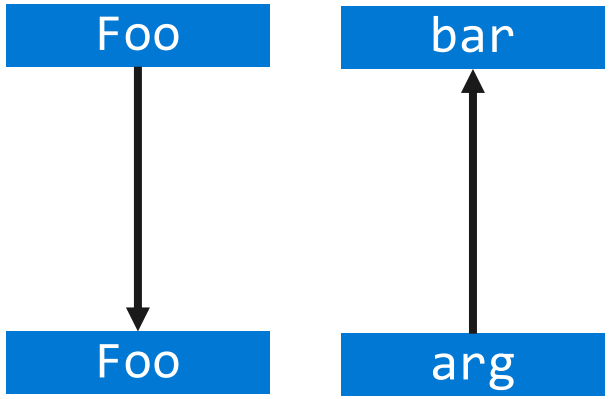
# Constructing Name Flows – Actuals to Formals




```
Exists(pathToFile)
```

```
def Exists(string path) { … }
```

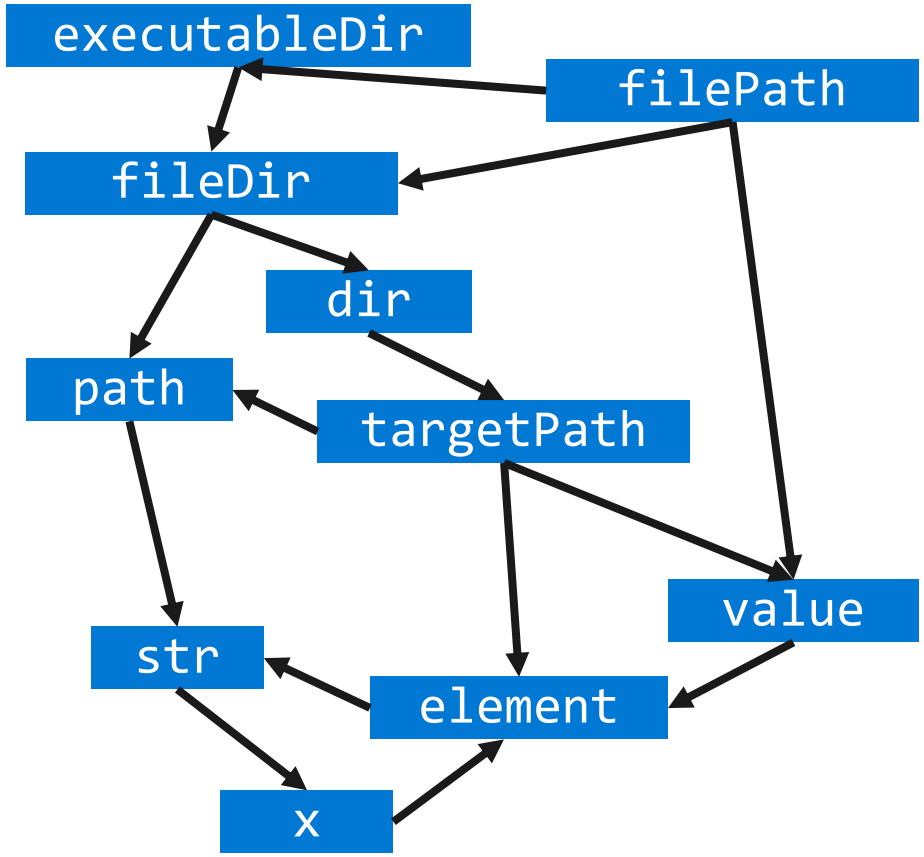
# Constructing Name Flows – Override




```
def override string Foo(string bar) { … }
```

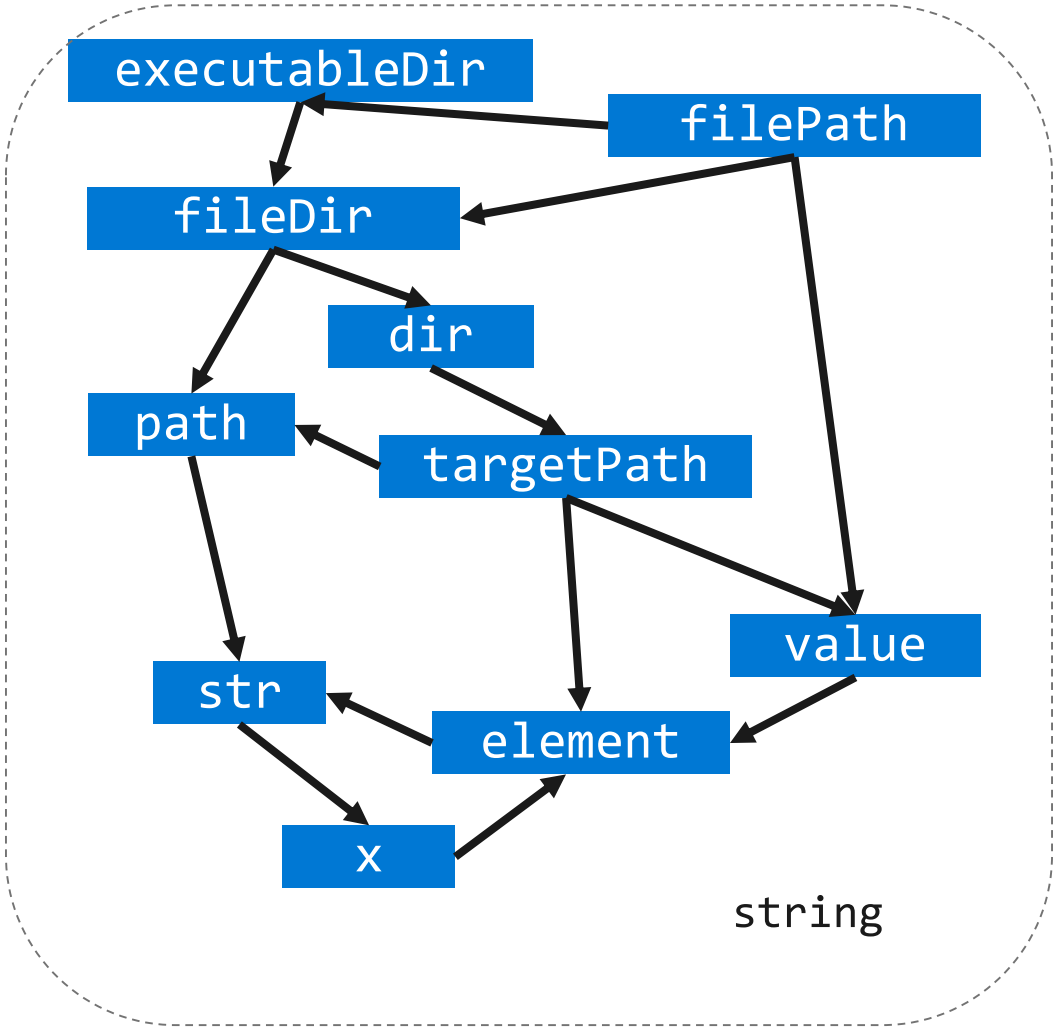
```
def string Foo(string arg) { … }
```

# Constructing Name Flows – Summary



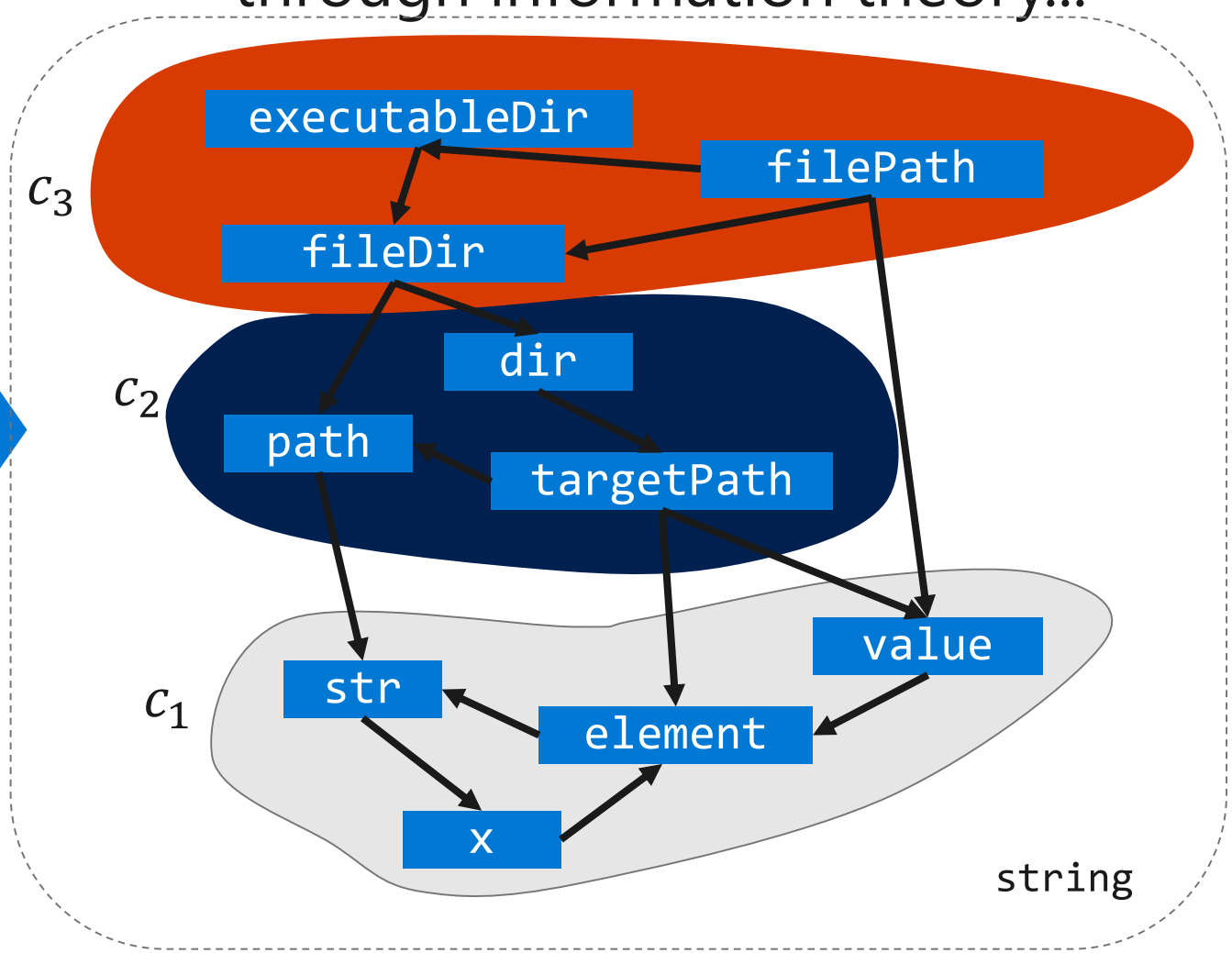

- Capture type-correct flows
- Capture names of variables/methods

# From Dataflow to Nominal Type Refinements




Dataflow of `string` variables/method and their names.

through information theory...




Nominal `string` refinements.

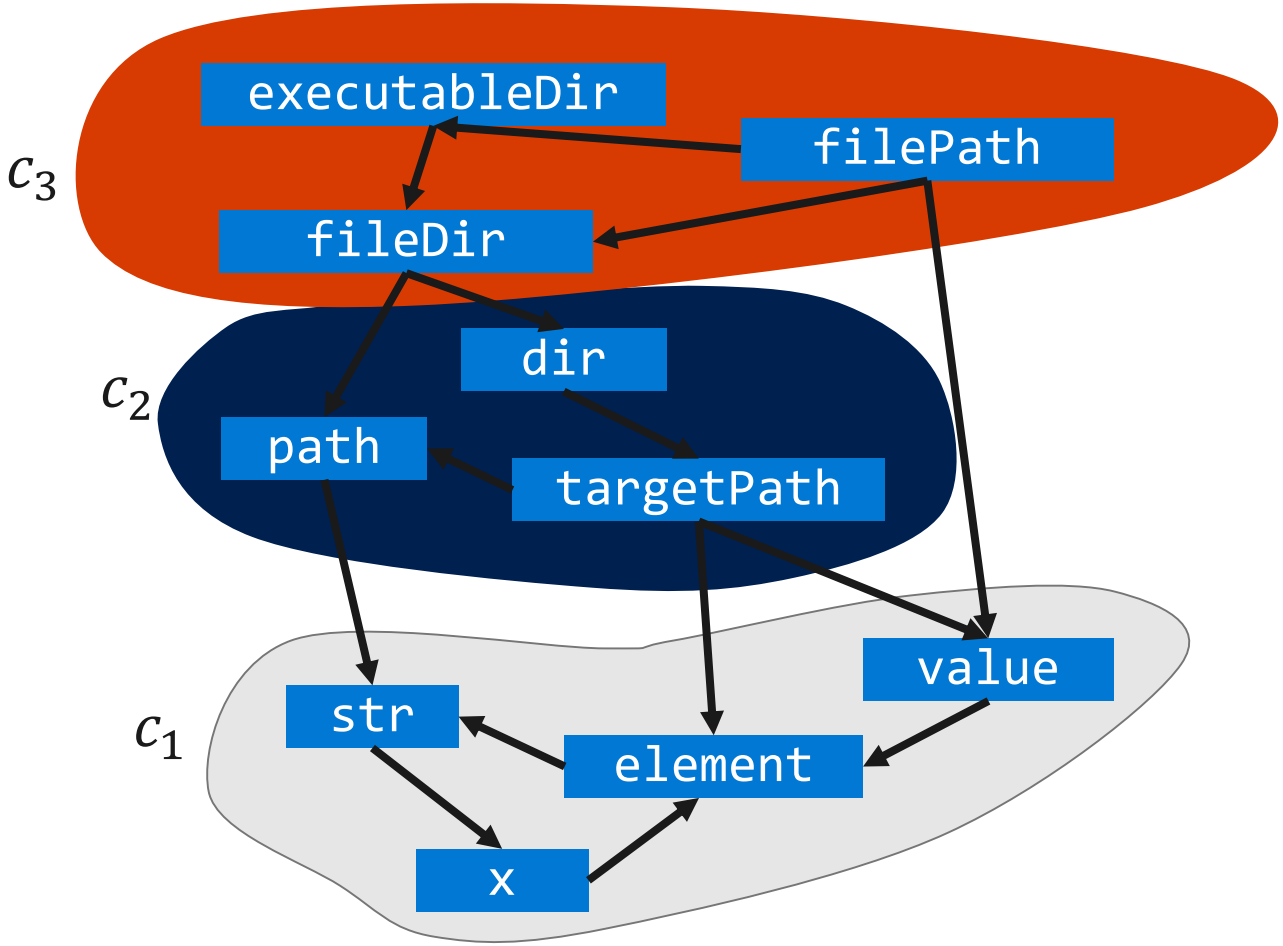
# Information-Theoretic Clustering

We want both

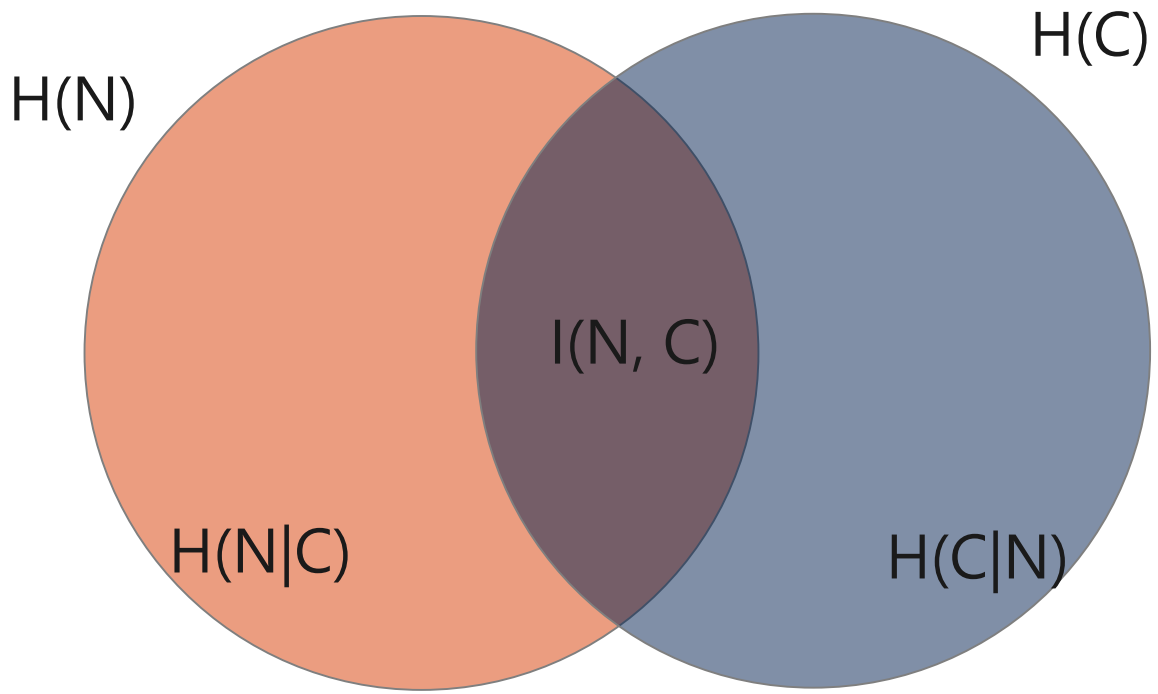
P(cluster|name)

P(name|cluster)

to have low entropy

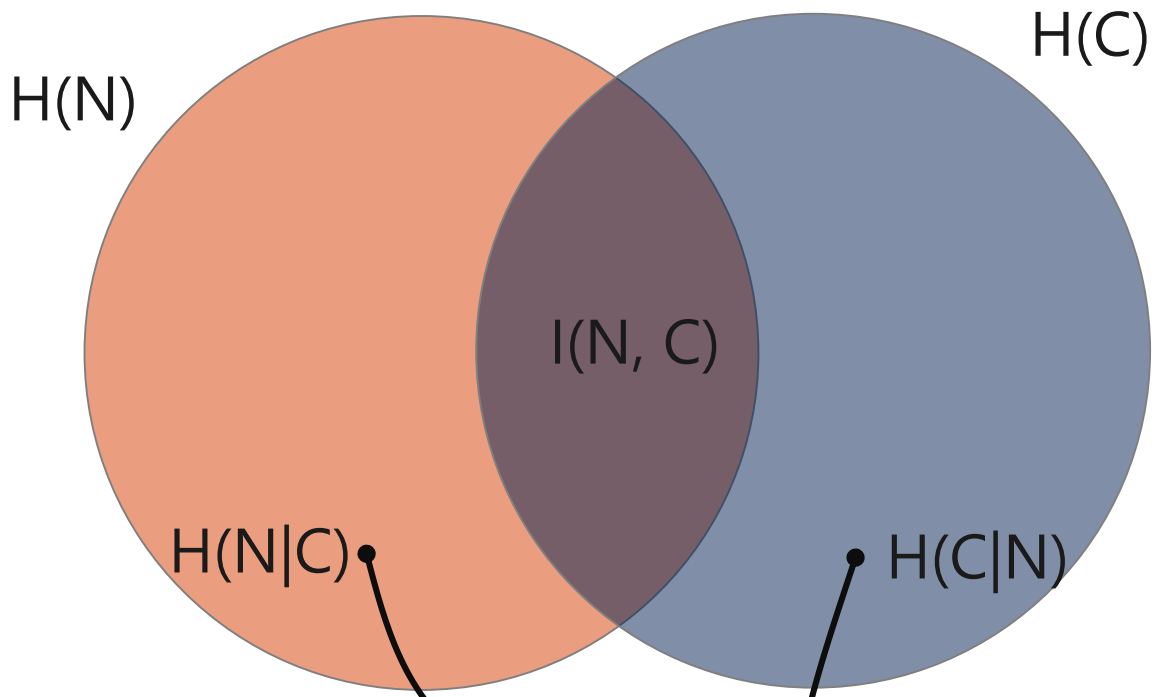


# Variation of Information



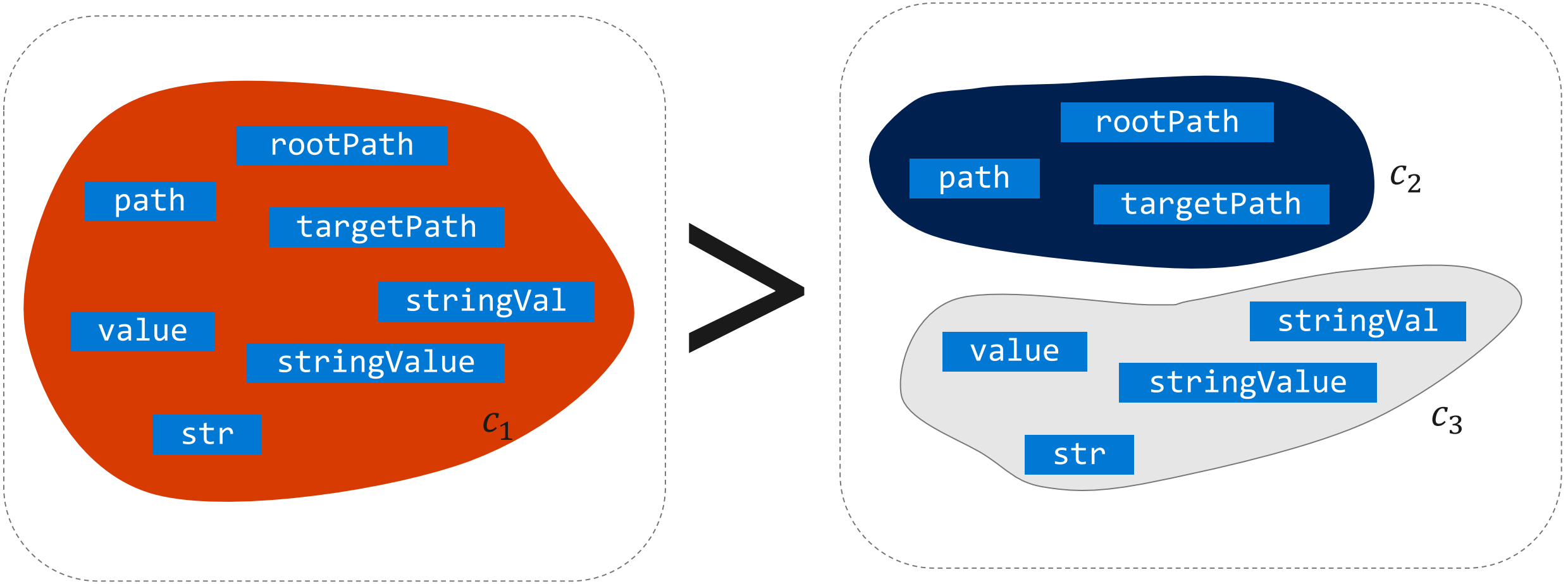

$$VI(N,C) = H(N|C) + H(C|N)$$
$$= H(C,N) - I(C,N)$$

# Variation of Information: Objective



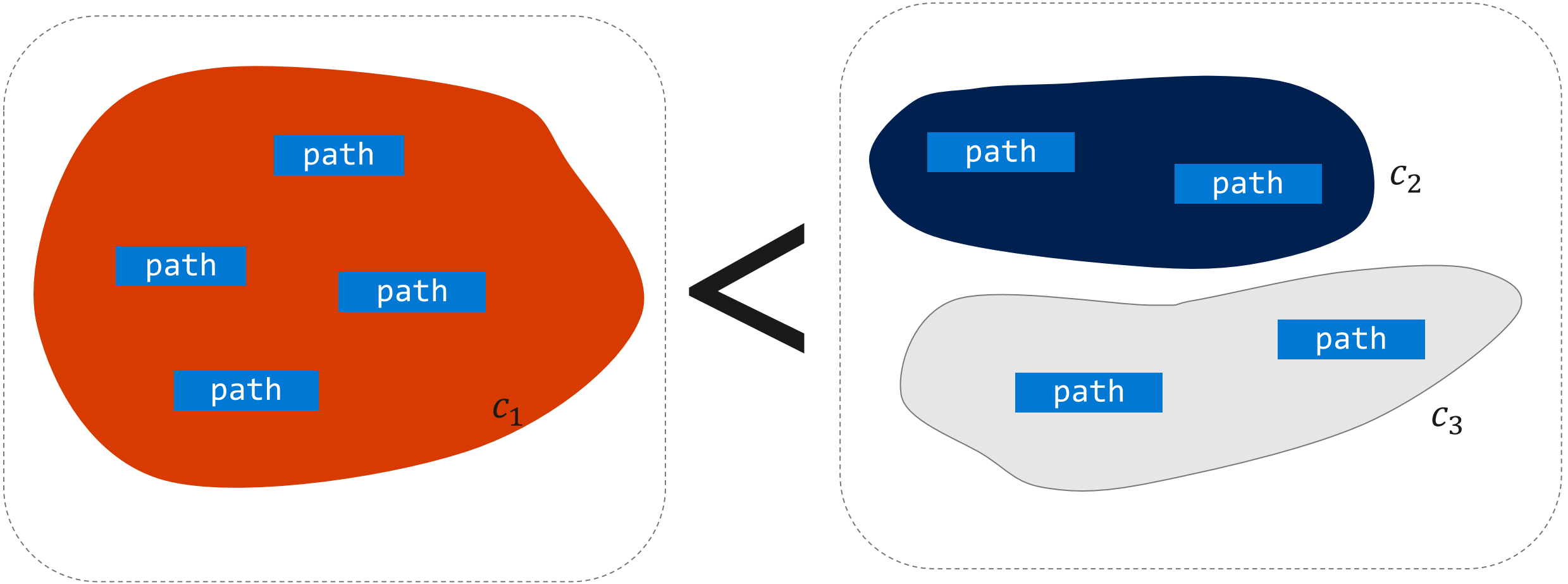

$$C^* = argmin_C VI(N, C)\ s.t. \exists R = (C, \subseteq)$$

# Variation of Information: Intuition




$$VI(N, \{c_1\}) > VI(N, \{c_2, c_3\})$$

# Variation of Information: Intuition

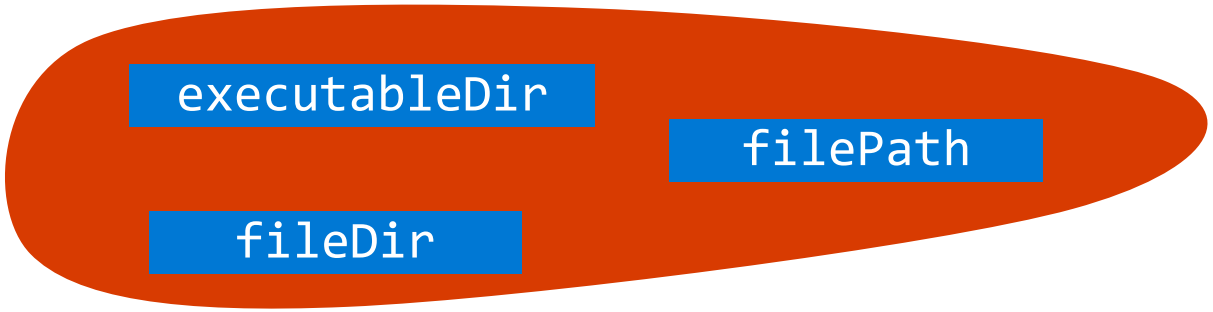



$$VI(N, \{c_1\}) < VI(N, \{c_2, c_3\})$$

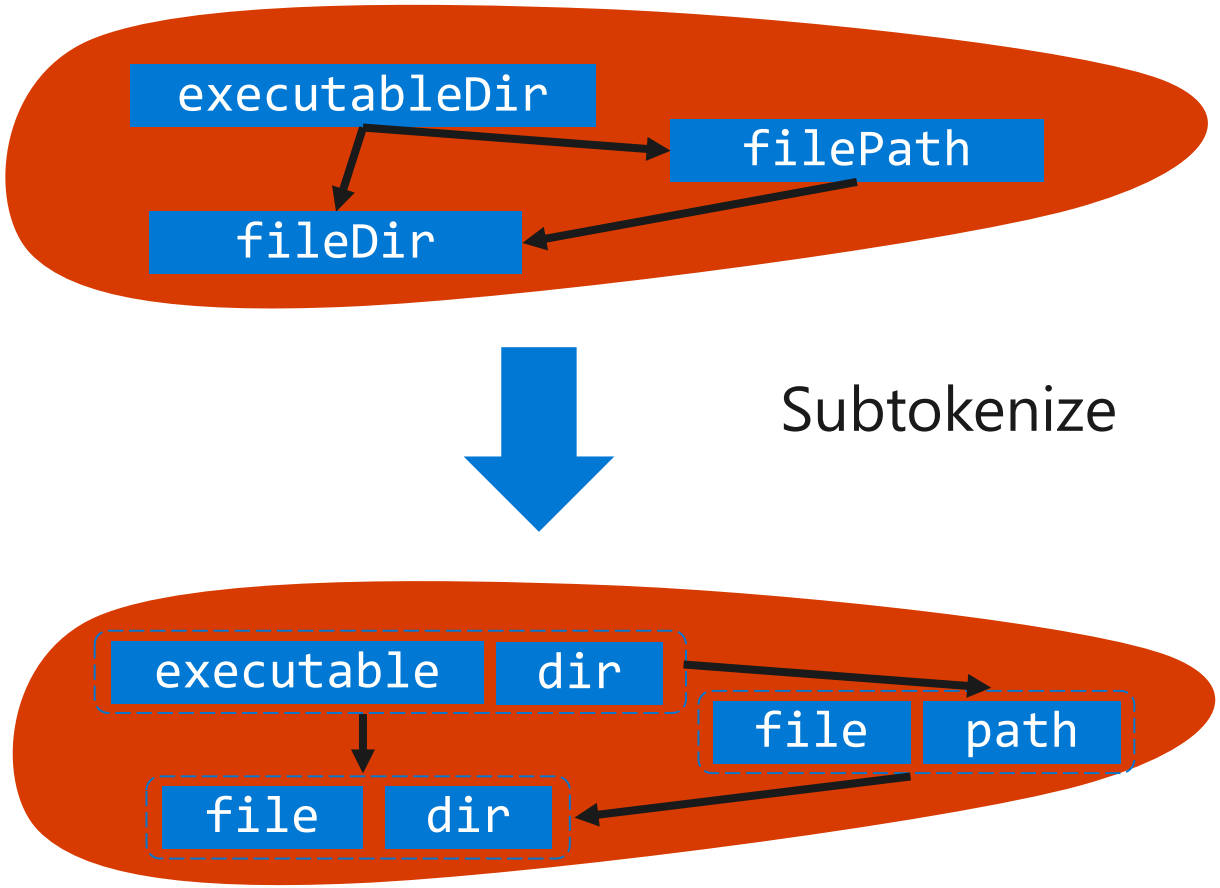
# Modeling Names

$$VI(N,C) = H(N|C) + H(C|N)$$
$$= H(N,C) - I(N,C)$$

$$H(N|C) = -\sum_{c \in C} P(c) \sum_{n \in N} P(n|c) \log P(n|c)$$



# Modeling Names – Subtokens

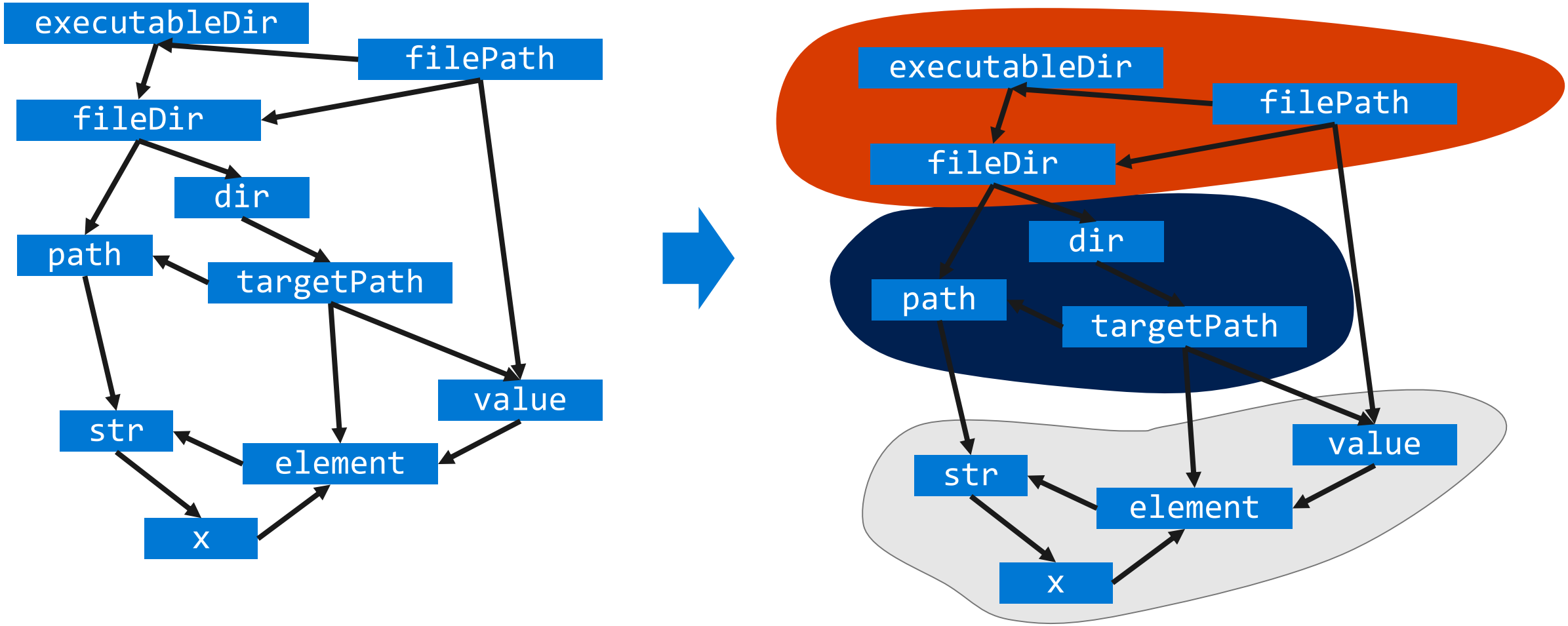



$$P_{emp}(\texttt{"dir"}|c) = \frac{2}{6}$$

- Define a multinomial probability over subtokens.
- Add a prior for smoothing:

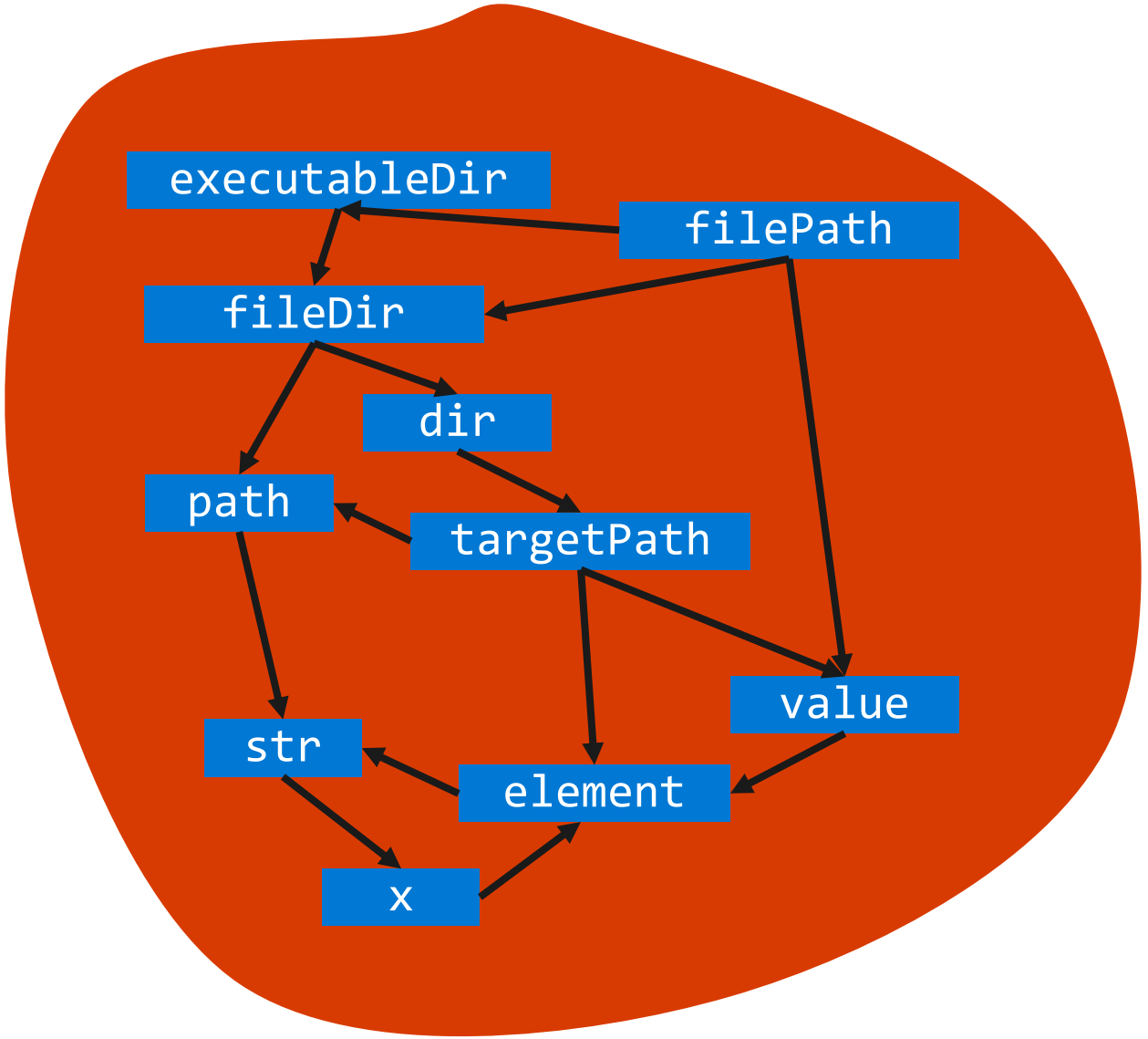
$$\mathrm{P(s|c)} = \frac{|s \in S_c| + aP_{all}(s)}{|S_c| + a}$$

# From Dataflow to Nominal Type Refinements

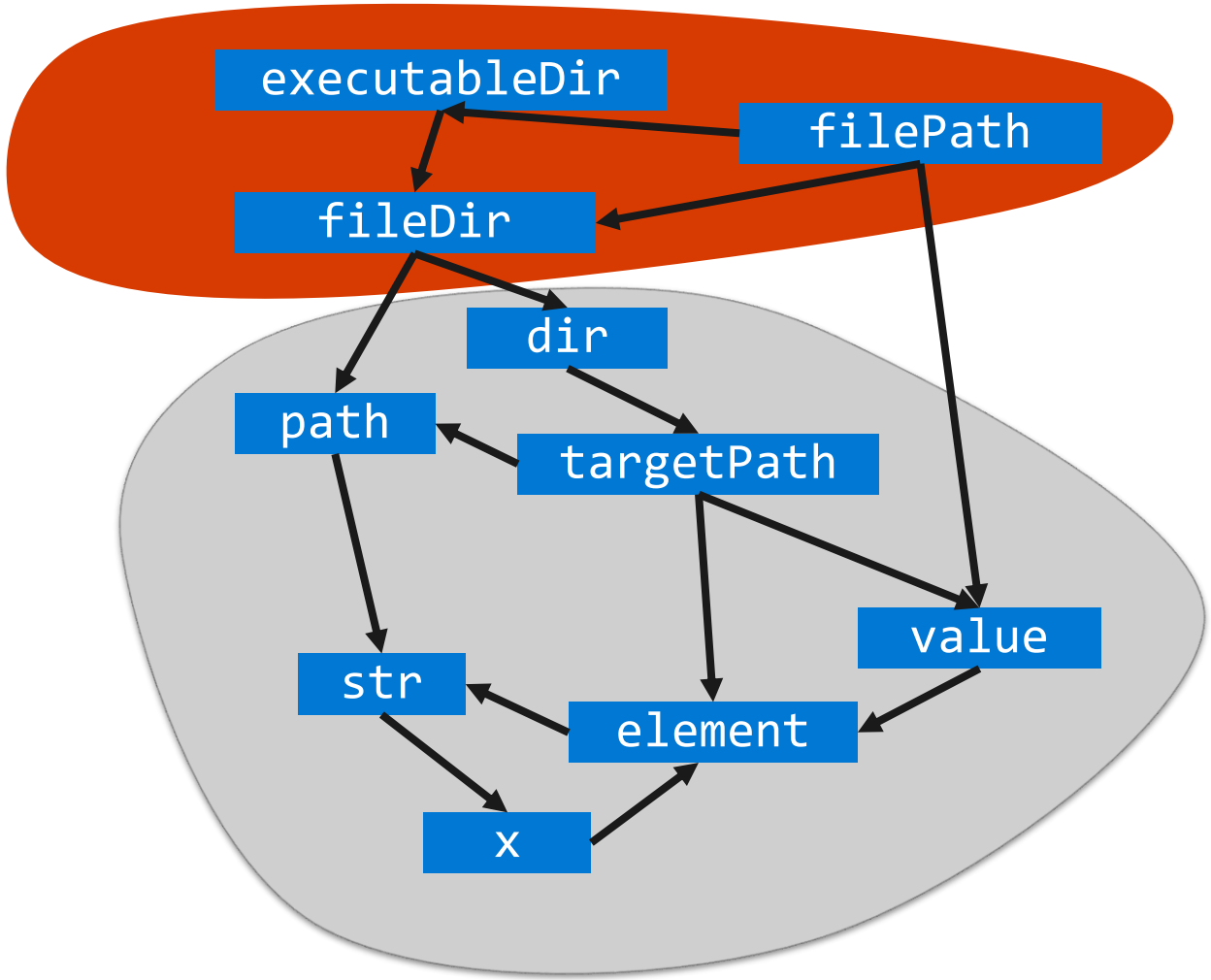



$$C^* = argmin_C VI(N, C) \;\; s.t. \exists R = (C, \subseteq)$$

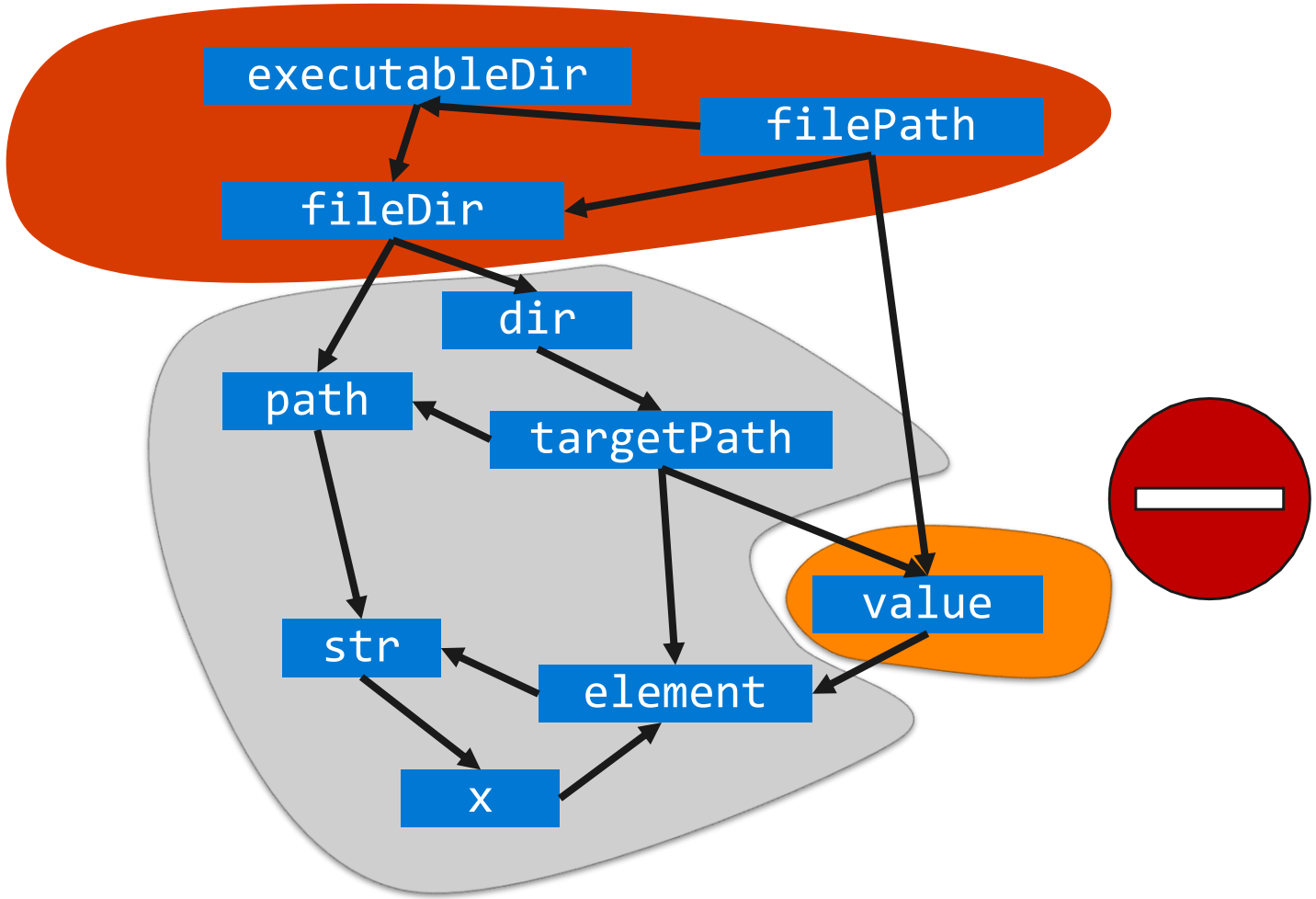
# Greedy Optimization of VI – Split



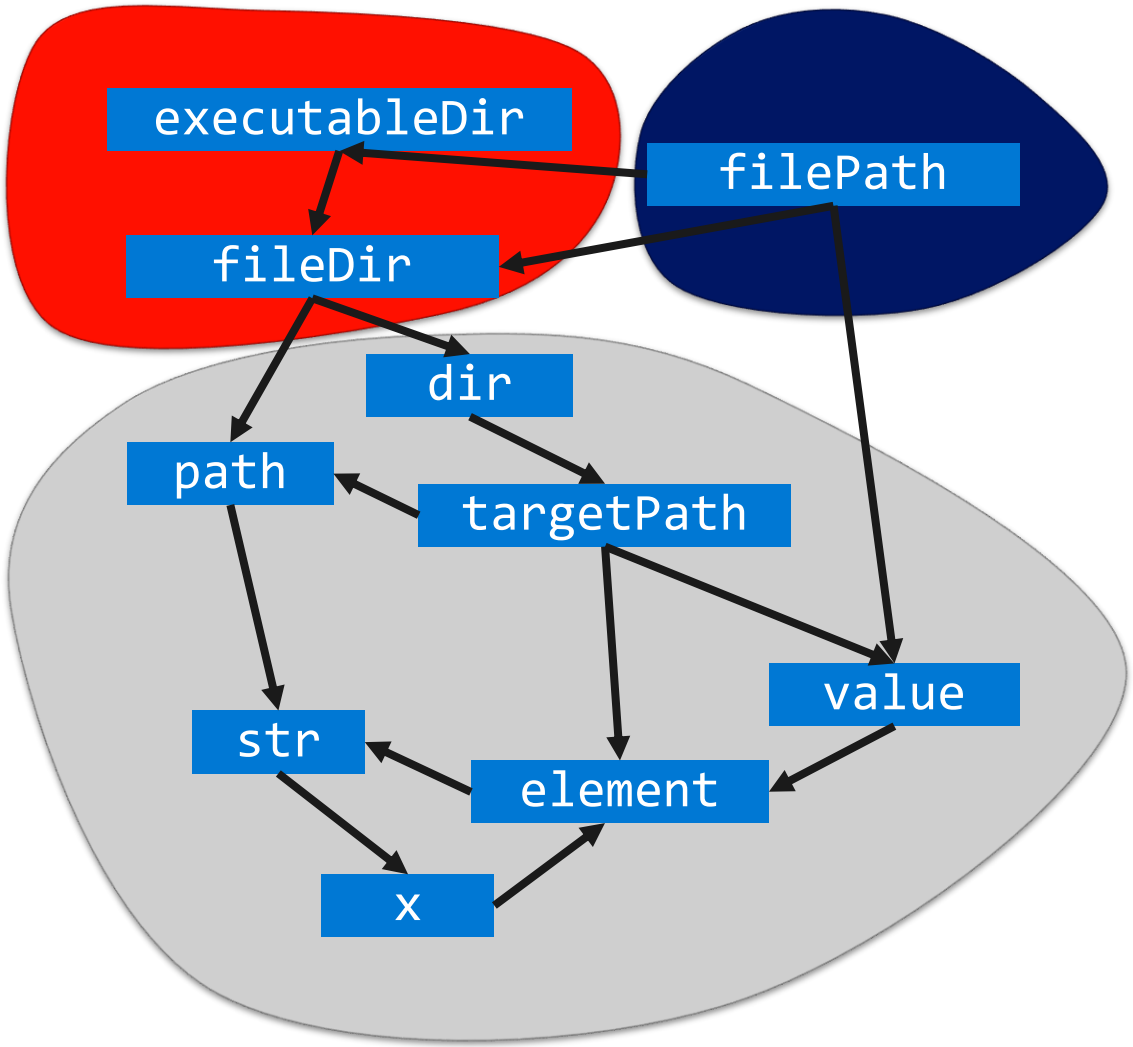
# Greedy Optimization of VI – Split



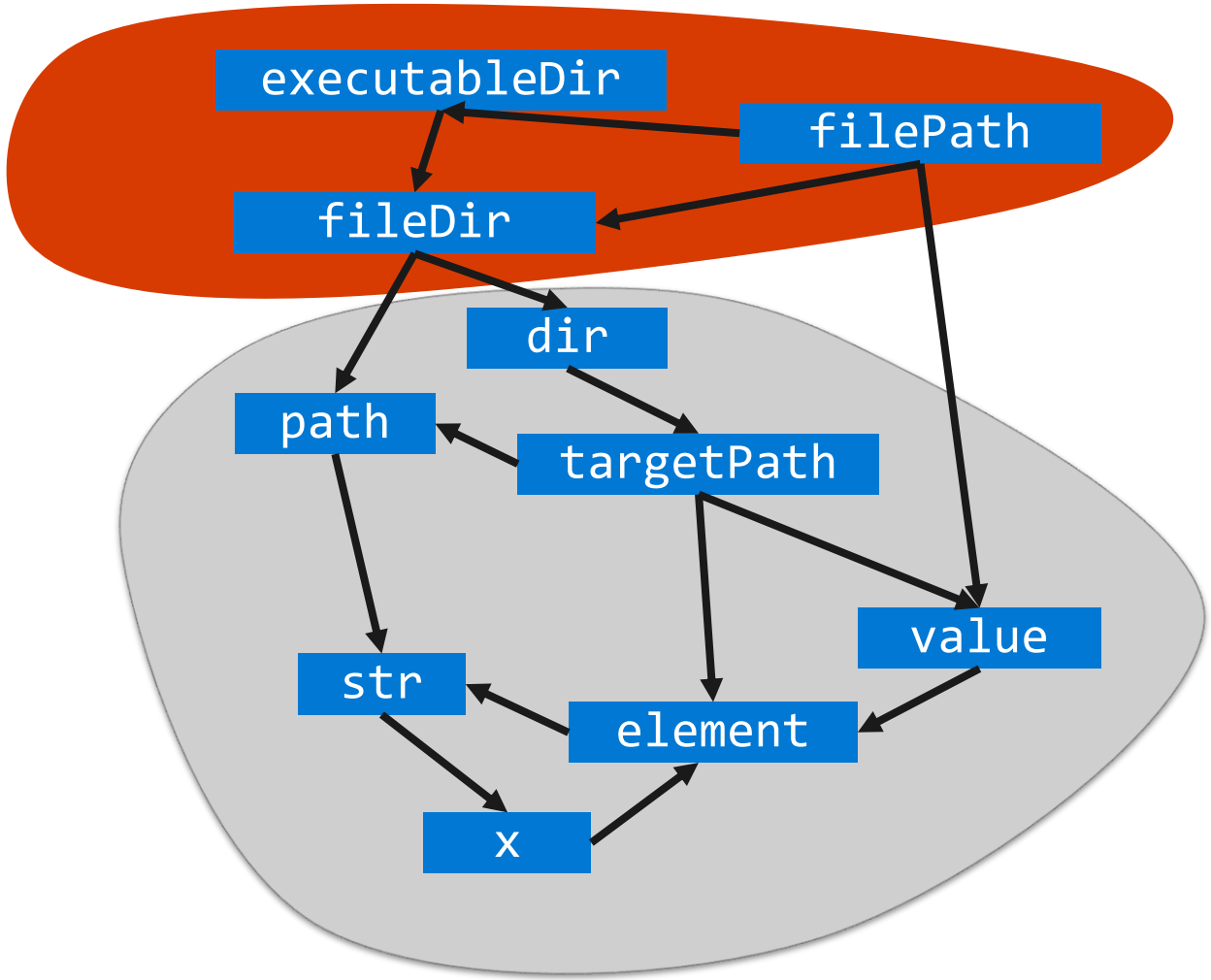
# Greedy Optimization of VI – Partial Order Point



# Greedy Optimization of VI – Merge



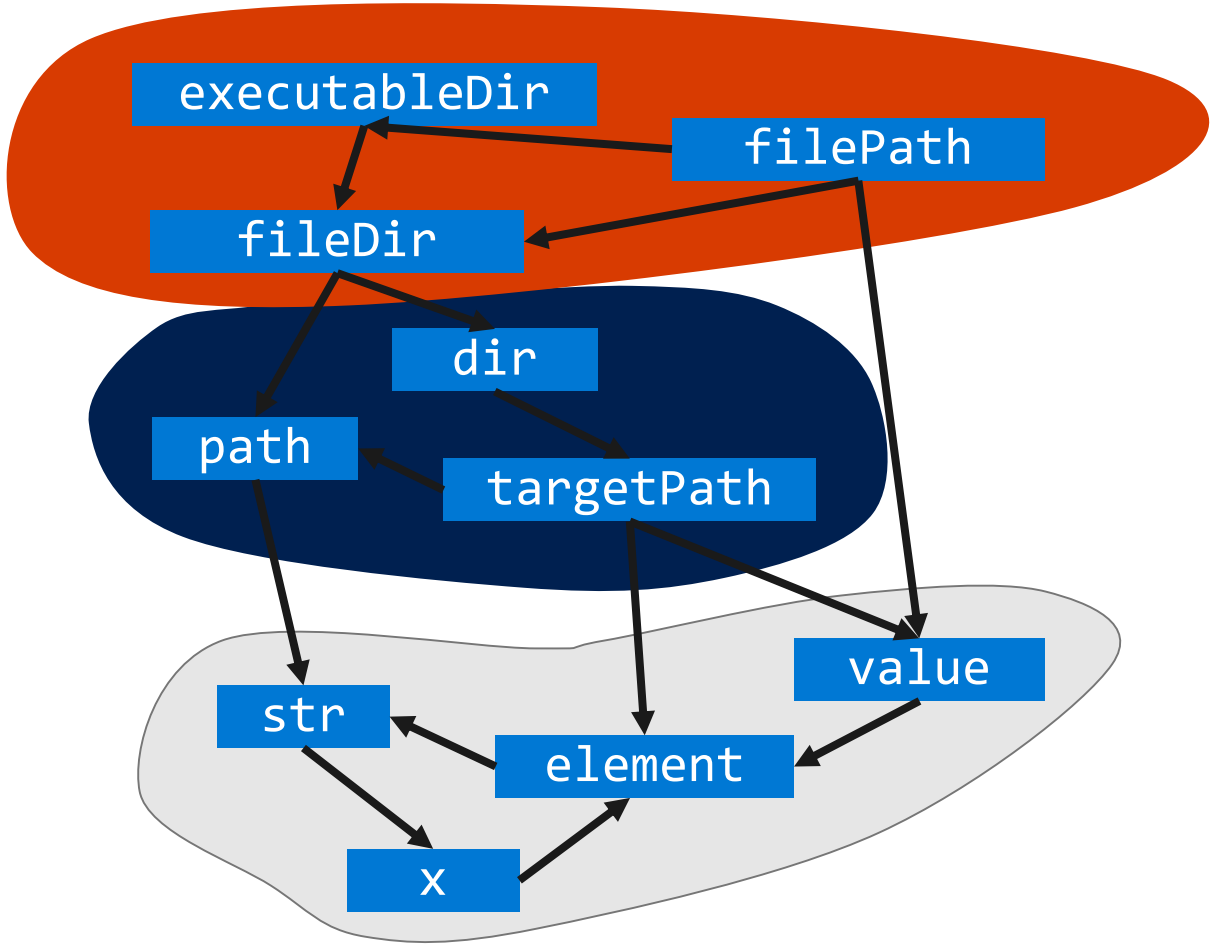
# Greedy Optimization of VI – Merge



| | Full name of nodes or constant values |
|---|---|
| 1 | path, Path, originalRequestPath, modifiedRequestPath, owinRequestPath, "/ emailConstraint/", contentPath, basePath, IViewEngineHost::ExpandPath, AspNetRootPathProvider::GetRootPath, "/", DiagnosticsConfiguration::GetNormalizedPath, NancyContext::ToFullPath, ModulePath |
| 2 | DefaultCulture, defaultCulture, cookieCulture, cultureLetters, name |
| 3 | earlyExitReason, "Requires Any Claim", "Requires Claims", "Requires Authentication", "Accept" |
| 4 | IObjectSerializer::Serialize, DefaultObjectSerializer::Serialize, JsonObject::ToString, SimpleJson::SerializeObject, HttpQsCollection::toString |
| 5 | method, Method, "PUT", "POST", "PATCH", "OPTIONS", "HEAD", "GET", "DELETE" |
| 6 | value, cookieValue, sourceString, "SomeValue", cookieValueEncrypted, attemptedValue, decryptedValue, defaultValue |
| 7 | password, "password", realPassword, plainText, Password |
| 8 | HttpUtility::UrlDecode, HttpUtility::UrlPathEncode, HttpUtility::UrlEncodeUnicode, redirectUrl, fallbackRedirectUrl, url, path |
| 9 | header, "Accept-Language", "Accept-Encoding", "Accept-Charset", "X-Custom", "Content-Disposition", "Vary", "Etag" |

| Full name of node or constant value in bepuphysics |
| --- |
| damping, SuspensionDamping, starchDamping, dampingConstant, angularDamping, LinearDamping |
| currentDistance, distance3, candidateDistance, pointDistance, distanceFromMaximum, grabDistance, VariableLinearSpeedCurve::GetDistance, tempDistance |
| goalVelocity, driveSpeed, GoalSpeed |
| minRadius, MinimumRadius, Radius, minimumRadiusA, WrappedShape::ComputeMinimumRadius, topRadius, MaximumRadius, graphicalRadius, TransformableShape::ComputeMaximumRadius |
| blendedCoefficient, KineticFriction, dynamicCoefficient,KineticBreakingFrictionCoefficient |
| angle, myMaximumAngle, MinimumAngle, currentAngle, MaximumAngle, steeringAngle, MathHelper::WrapAngle |
| targetHeight, Height, ProneHeight, crouchingHeight, standingHeight |
| Mass, effectiveMass, newMassA, newMass |
| M22, m11, M44, resultM44, M43, intermediate, m31, X, Y, Z |

# Interplay between types and names.




Joint work with S. Dash, E.T. Barr (UCL)

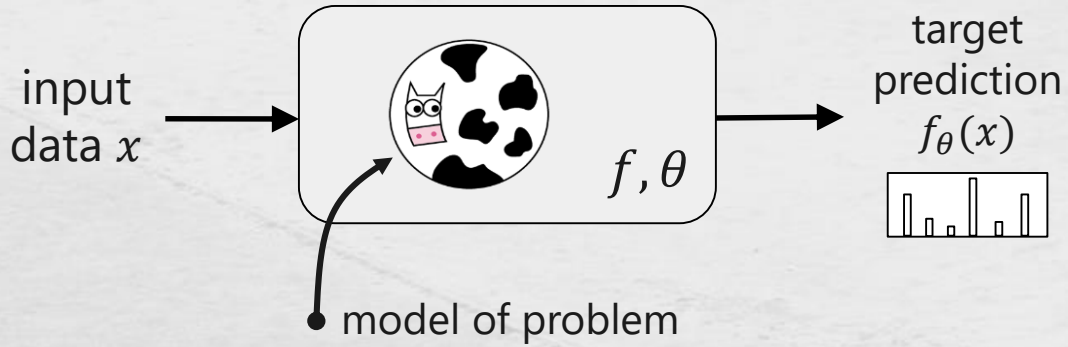
UI/UX

ML Capabilities

Metrics

Low resources

# Learning Signals




- Given dataset $\{(x_1, y_1), \ldots, (x_N, y_N)\}$
- Minimize Loss $\mathcal{L}(\theta) = \frac{1}{N}\sum_i L(f_\theta(x_i), y_i)$

# How to Tell a Compiler
# What We Think We Know?

Guy L. Steele Jr.
Software Architect, Oracle Labs

SPLASH-I Keynote
Friday, November 4, 2016

ORACLE

## This Talk Is an Essay (I Didn't Know Where It Would Go)

I started out wanting to tell things to a compiler (or IDE).

- Specifically, I want to tell a compiler far more than types.
- I thought the conclusion would be that compilers need theorem provers.

That's not a bad goal. But I have ended up wanting much more:

- I want a conversational partner that will track what I am doing.
- I want it to react to context and intention.
- I want it to give me relevant information.

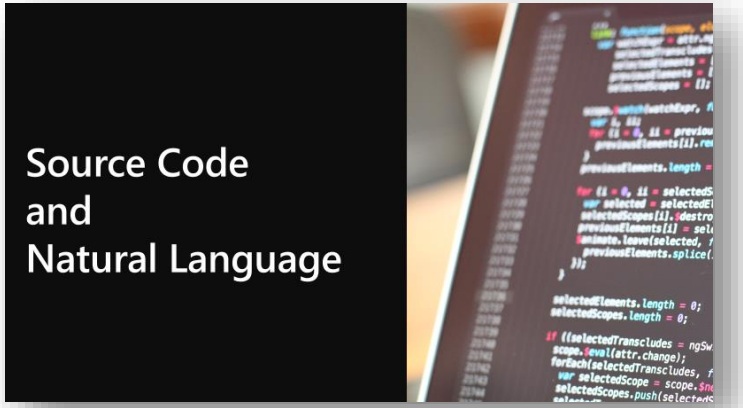
This is much harder than "Just the facts, Ma'am."

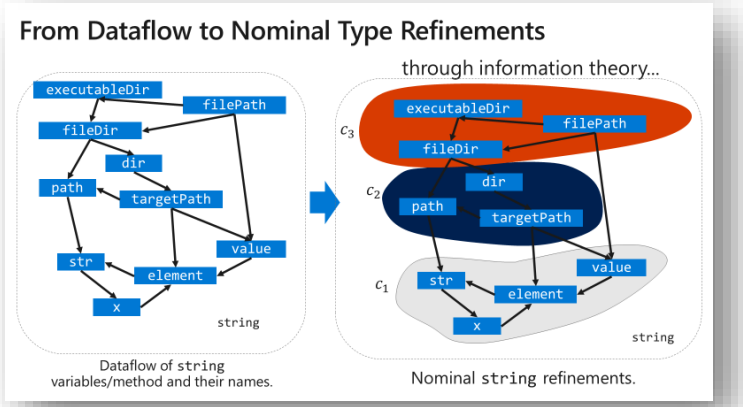
ORACLE

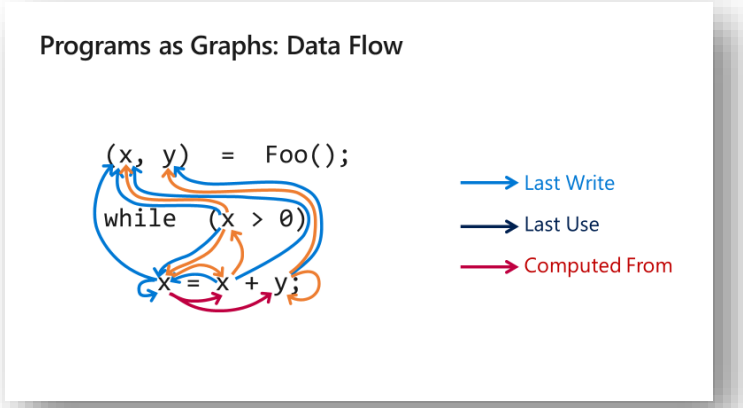




Slides at:
http://web.cs.ucdavis.edu/~su/SteeleSplash2016.pdf




# Towards Learned Program Analyses with Machine Learning

Reasoning over Rich Structures

Learning from Human Aspects of Code

miltos1

https://miltos.allamanis.com

Microsoft

Deep Program Understanding
Cambridge, UK