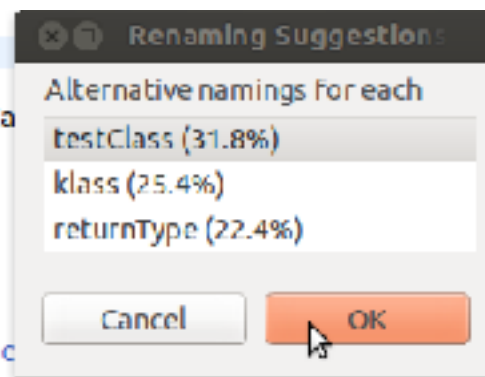


```
private Test testCaseForClass(Class<?> each) {
    if (TestCase.class.isAssignableFrom(each)) {
        return new TestSuite(each.asSubclass(TestCase));
    } else {
        return warning(each.getCanonicalName() + "
    }
}
```



```
for (Iterator iter=$methodInvoc;
    iter.hasNext(); )
{$BODY$}
```

Statistical Analysis of Computer Program Text

Charles Sutton
University of Edinburgh
& The Alan Turing Institute

Companion Site: <http://bit.ly/sutton-nlpswe>



THE UNIVERSITY of EDINBURGH
informatics

**The
Alan Turing
Institute**

Microsoft
Research

EPSRC
Engineering and Physical Sciences
Research Council

Source code is a means of human communication

```
try{  
    Node $name=$methodInvoc();  
    $BODY$  
}finally{  
    $(Transaction).finish();  
}
```

```
public static final  
String $name = $StringLit;
```



Over **20 billion** lines of open source code online

Implicit knowledge about how to write code

- Uses common libraries
- Avoids common bugs
- Easy to read and maintain

Source code as a means of human communication

Perhaps PL text has NL-style regularities

Statistical NLP techniques for
identifying patterns in PL text

Every SWE activity can benefit from NLP+ML

- Defining requirements
- Architecting
- Implement systems
- Reading
- Navigation
- Maintenance
- Optimising performance
- Validation
- Refactoring
- Porting

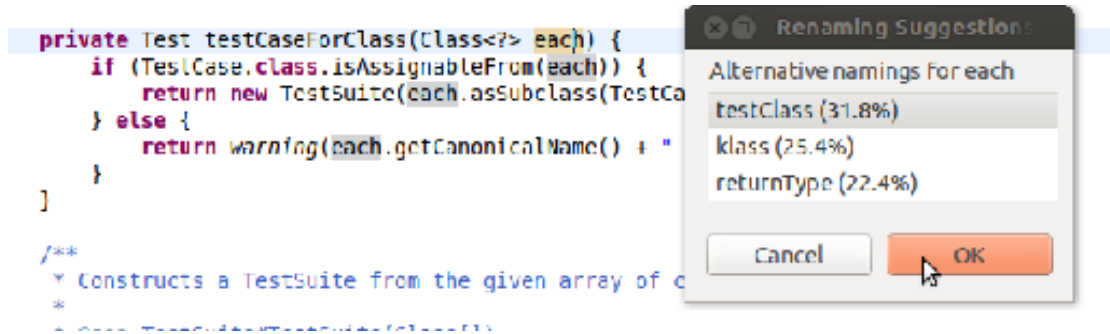
Every NLP problem has SWE analogue

- Spelling correction
- Finding co-locations
- Summarisation
- Generation
- Machine Translation
- Question Answering
- Semantic Parsing
- Semantic Entailment
- Information Extraction
- Information Retrieval
- Grounding Semantics
- Statistical Parsing (!)



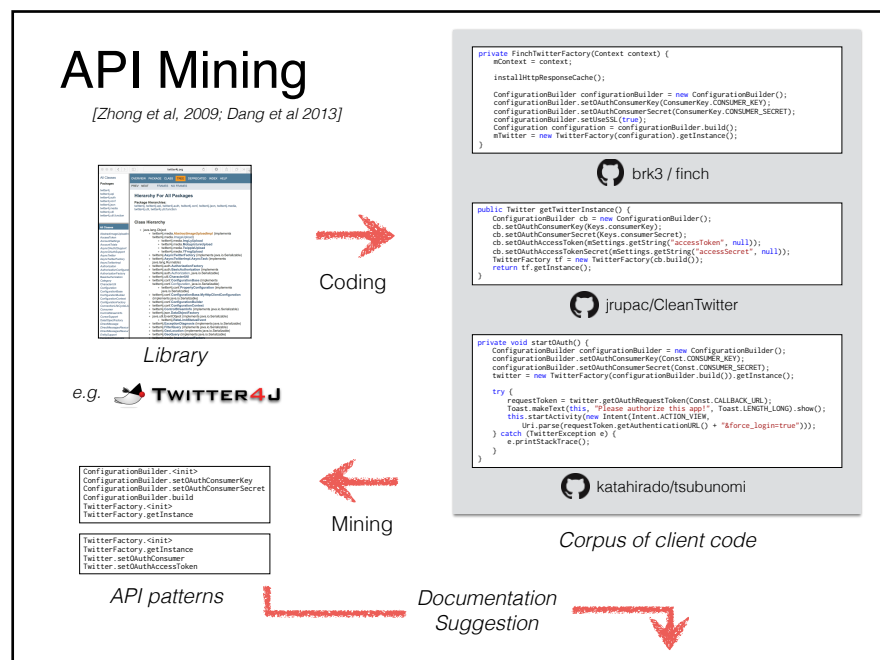
```
while (($String) = $(BufferedReader).
    readLine()) != null) {
    $BODY$
}
```

```
while (($String) = $(BufferedReader).
    readLine()) != null) {
    $BODY$
}
```

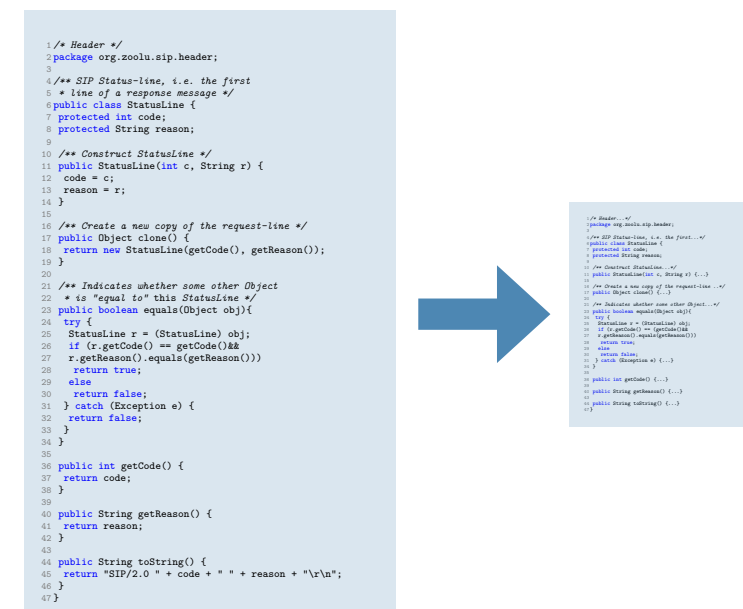


Learning coding conventions

Mining idioms (probabilistic grammars)



(structure learning)



Code summarization (topic models)



Learning Natural Coding Conventions

A **coding convention** is a syntactic constraint beyond that imposed by the language grammar

Coding Conventions

Developers care

- Create style guides
- Enforce during code reviews

Research in SWE

- *Boogerd and Moonen, 2008*
- *Caprile and Tonella, 2000*
- *Takang, 1996*



gofmt

indent



Importance of Conventions

	Code Review Discussions
Conventions	38%
Naming	24%
Formatting	9%

Study at Microsoft:

169 code reviews with 1,093 discussion threads.

Where conventions come from?

- Too many to agree explicitly
- Instead arise *implicitly*
- *Soft* constraints (mores) rather than hard constraints (laws)



New developers don't know
about implicit conventions

Coding convention inference problem:

Learn conventions from examples
of conventional code

junit/src/test/java/junit/tests/runner/TextRunnerTest.java

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

Suggest
alternate
names

input
InputStream
is
stream

Score by ngram model
and threshold

input (81.93%)

Language Models for Source Code

Probability distribution over token sequences:

$$P(t_0 \dots t_M) = \prod_{m=0}^M P(t_m | t_{m-1} \dots t_{m-n+1})$$

Consider naive estimator:

$$P(t_m | t_{m-1} \dots t_{m-n+1}) = \frac{\text{count}(t_m \dots t_{m-n+1})}{\text{count}(t_{m-1} \dots t_{m-n+1})}$$

In Naturalize : Choose the name other programmers use in similar contexts

junit/src/test/java/junit/tests/runner/TextRunnerTest.java

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

Suggest
alternate
names

input
InputStream
is
stream

Score by ngram model
and threshold

input (81.93%)

$$P(t_m | t_M \dots t_{m+1}, t_{m-1} \dots t_{m-n+1})$$

Learning Formatting Conventions

```
5      @Override public void
6      write(int arg0) throws IOException {
7      }
8  }
```

```
5  INDENT1n3s @ SPACE0 ID SPACE1s public SPACE1s void
6  INDENT1n0 ID SPACE0 ( SPACE0 ID SPACE1s ID SPACE0 ) SPACE1s
  throws SPACE1s ID SPACE1s {
7  INDENT1n0 }
8  INDENT1n-3s }
```


Evaluation Methodology

Automatic evaluation:

- Top 10 Java projects on GitHub
- Perturb existing code
- Measure: does Naturalize retrieve ground truth.

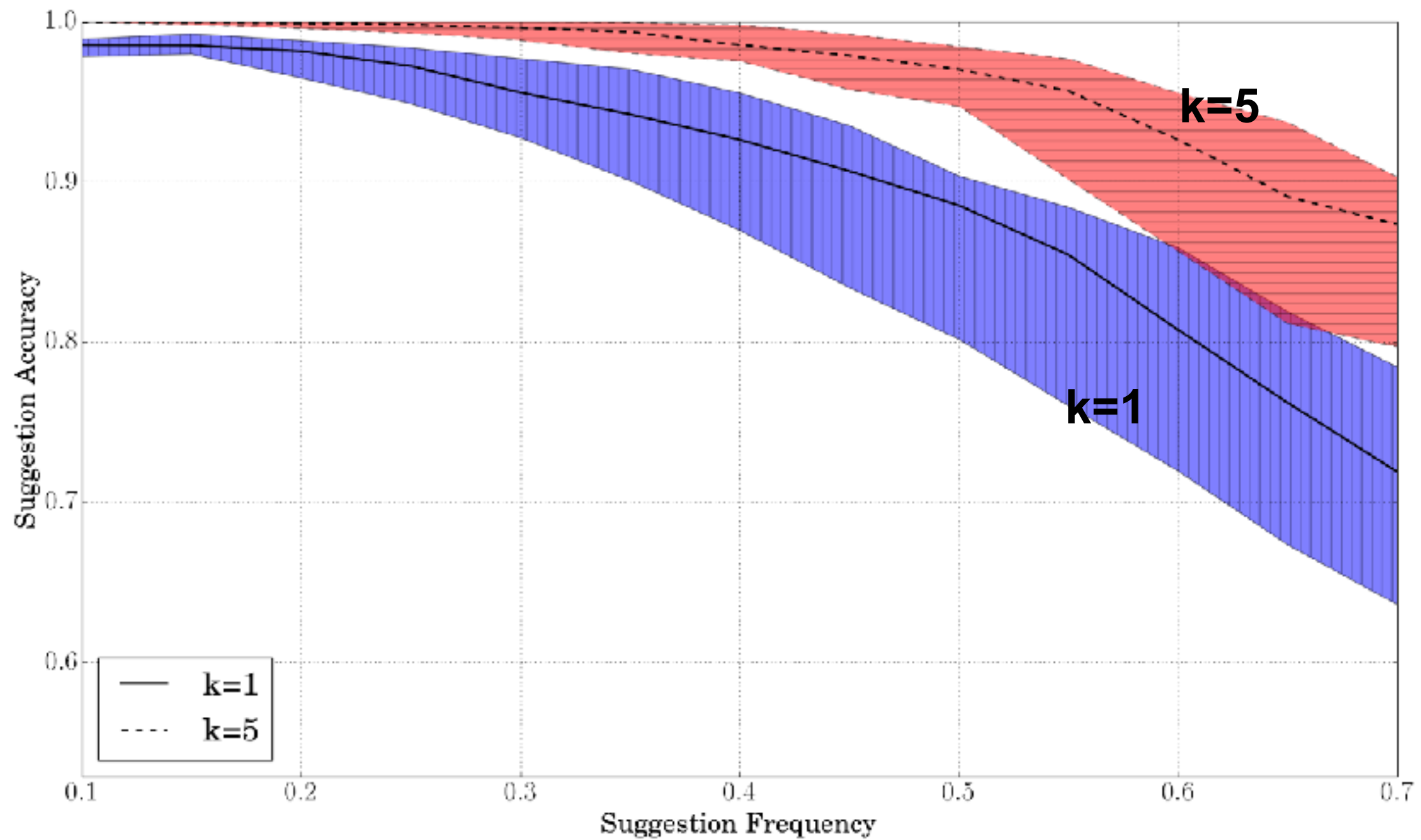
```
ForkJoinTask<?> XYZZY;  
if (task instanceof ForkJoinTask<?>)  
    XYZZY = (ForkJoinTask<?>) task;  
else  
    XYZZY = new ForkJoinTask.AdaptedRunnableAction(task);  
externalPush(XYZZY);
```



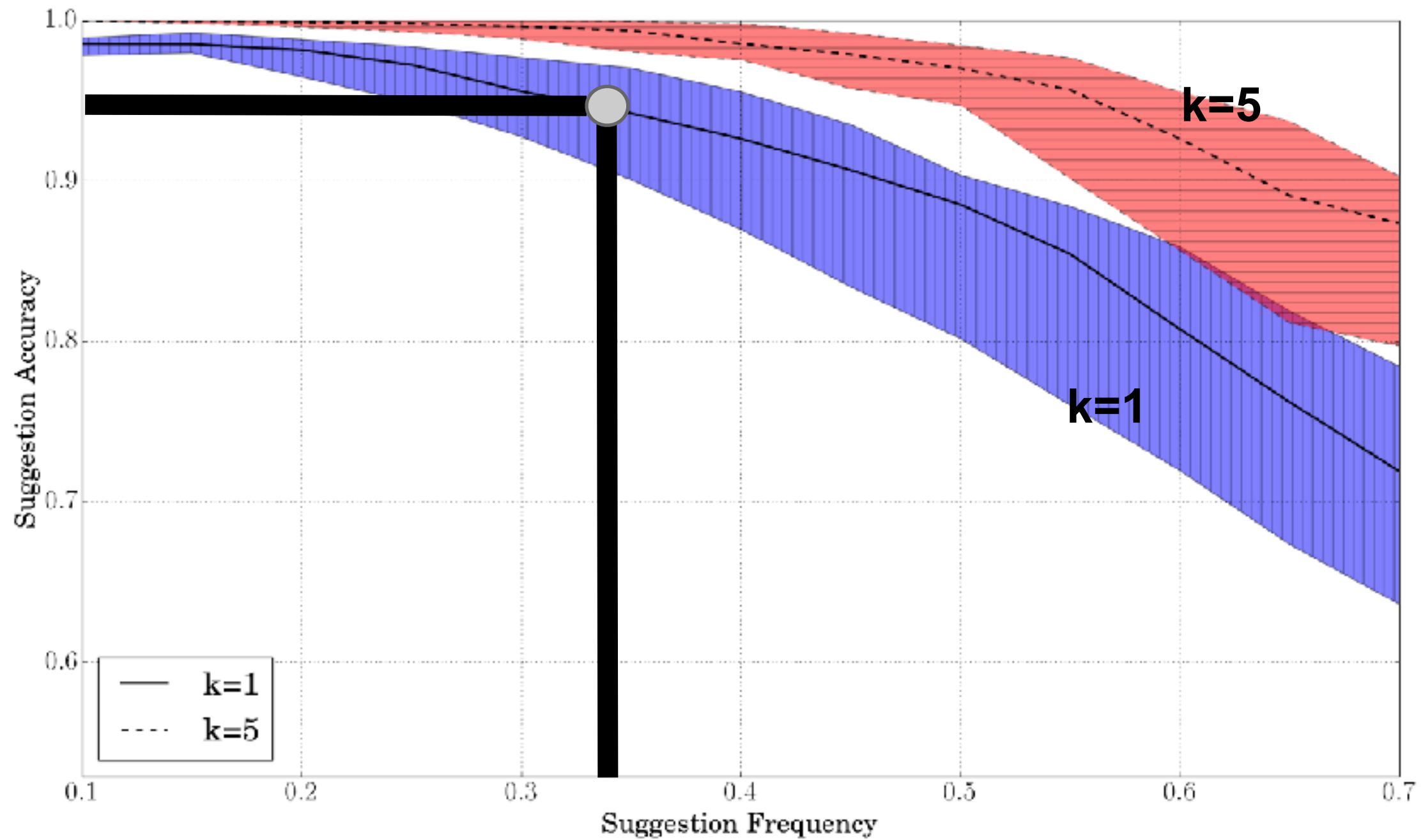
Naturalize

1. job (30%)
2. task (20%)
3. tsk (15%)

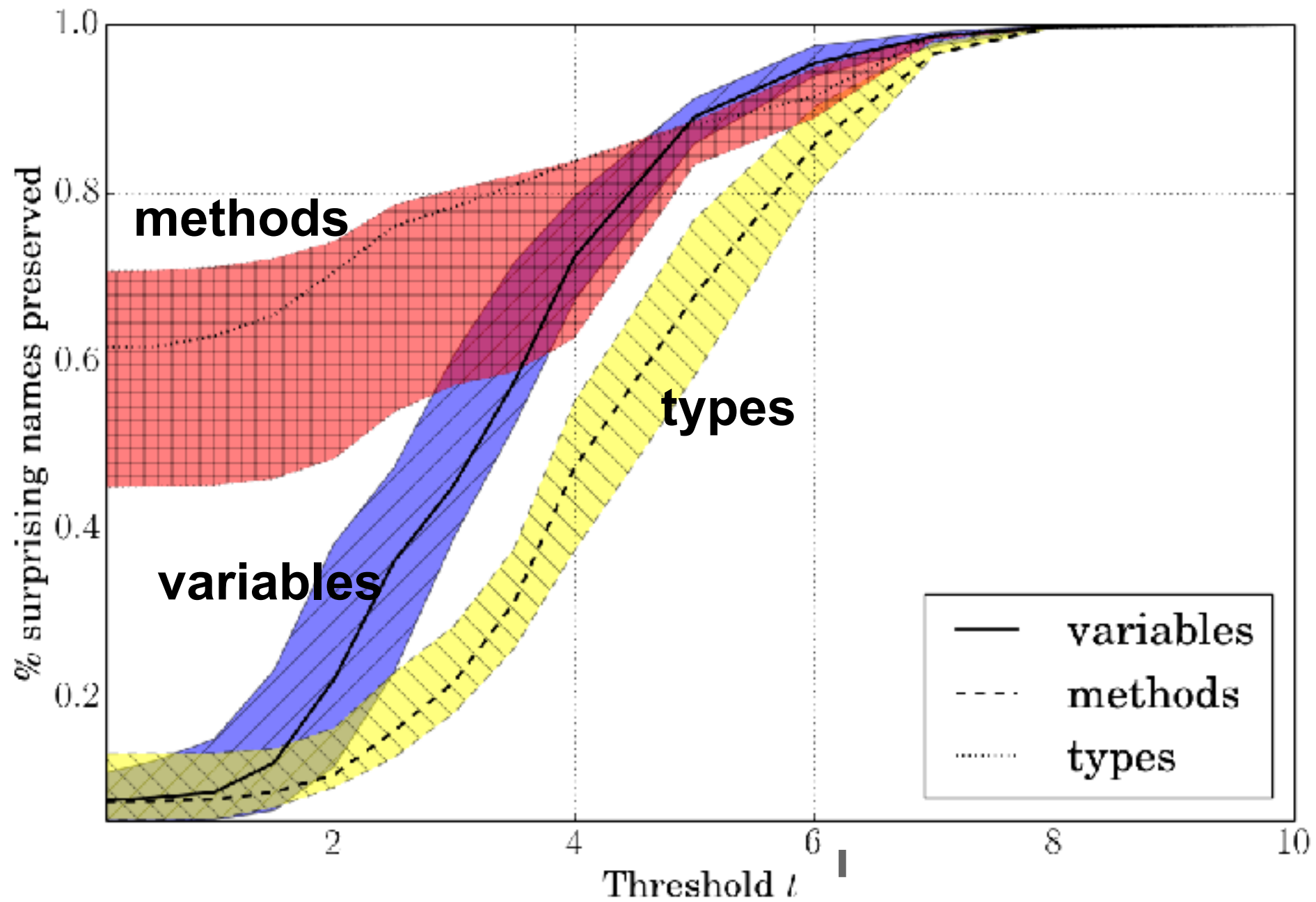
Variable Renaming



Variable Renaming



All names go to i ? No!



GitHub

This screenshot shows a GitHub pull request interface. At the top, a commit history section lists three commits by user 'mallamanis' on 10 Feb: 'Renamed the ImmutableBlobContainer container to blobContainer.', 'Renamed XContentParser.Token named "t" to "token".', and 'Renamed ClusterBlocks variable named "block" to "token".'. Below this, a section shows 'javanna' adding 'enhancement' labels 'v1.2.0' and 'v2.0.0' on 7 Apr. A comment from 'javanna' on 7 Apr says 'Merged, thanks!'. At the bottom, a red circle icon indicates the pull request was closed by 'javanna' on 7 Apr.

This screenshot shows a GitHub pull request interface. At the top, a commit history section lists five commits by user 'mallamanis' on 26 Feb: 'Renamed "1" (15.46%) to "index" (31.34%). The naturalize tool detected...', 'Renamed "value" (18.55%) to "scalar" (51.13%). The naturalize tool...', 'Minor changes in JavaDoc', 'Renamed "vector" (7.51%) to "point" (64.23%). The naturalize tool...', and 'Renamed "scale" (24.82%) to "scaleXY" (75.98%). The naturalize tool...'. Below this, a comment from 'sinistersnare' on 26 Feb says 'Very interesting project, and thanks for the contribution! I've always wanted to spend a good amount of time using findbugs with libgdx, there's a ton of recommendations it provides.'. A comment from 'badlogic' on 26 Feb says 'Wow, that's a pretty cool tool!'. Below the comments, a red circle icon indicates the pull request was closed by 'badlogic' on 26 Feb. A green circle icon indicates the pull request was reopened by 'badlogic' on 26 Feb. At the bottom, a purple circle icon indicates the pull request was merged by 'badlogic' on 26 Feb, merging commit '3d5848f' into 'libgdx:master' from 'mallamanis:master'. A 'Revert' button is visible at the bottom right.

18 patches for 5 open source projects:
14 accepted - 4 still waiting

Extensions / future work

- Neural network language models

[Allamanis, Barr, Bird and Sutton, 2015]

- Method and class naming
 - Convolutional attention mechanism

[Allamanis, Peng and Sutton, 2016]

- Future work
 - Longer distance context
 - Code semantics
 - LSTMs

Mining Idioms from Code

```
while (($String) = $(BufferedReader).  
    readLine()) != null) {  
    $BODY$  
}  
  
while (($String) = $(BufferedReader).  
    readLine()) != null) {  
    $BODY$  
}
```



A code idiom is a syntactic code fragment that recurs frequently across software projects and has a single semantic purpose.

What are Code Idioms? Example

Looping through lines of a `BufferedReader`

```
while (($String) = $(BufferedReader).  
    readLine()) != null) {  
    $BODY$  
}
```

Idioms Contain Metavariables

Looping through lines of a `BufferedReader`

```
while (($(String) = $(BufferedReader).  
    readLine())) != null) {  
    $BODY$  
}
```

Metavariables

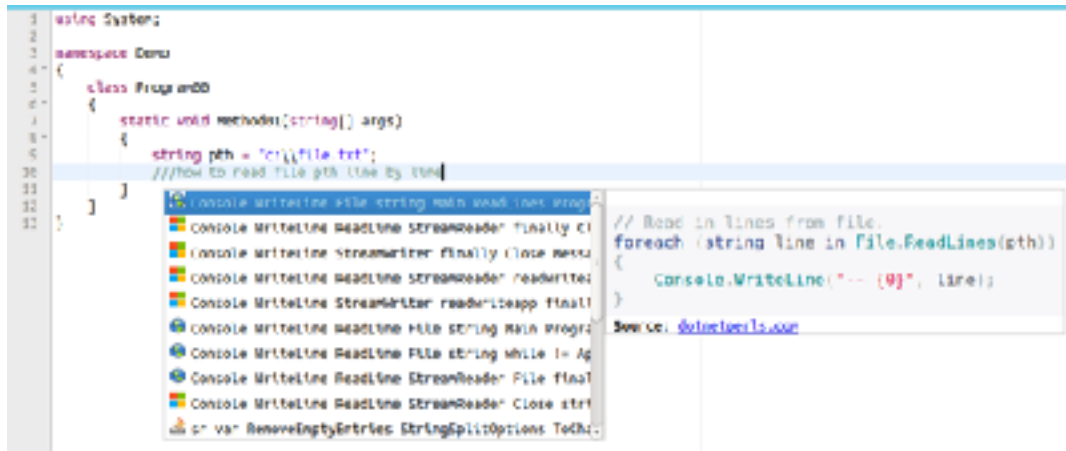
Idioms Contain Gaps

Looping through lines of a `BufferedReader`

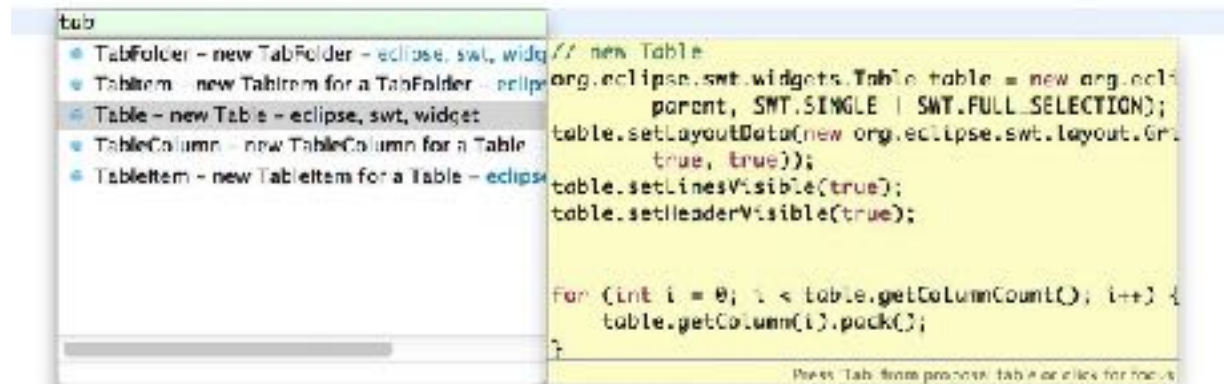
```
while (($String) = $(BufferedReader).  
    readLine()) != null) {  
    $BODY$  
}
```

└──────────▶ gap (non-terminal)

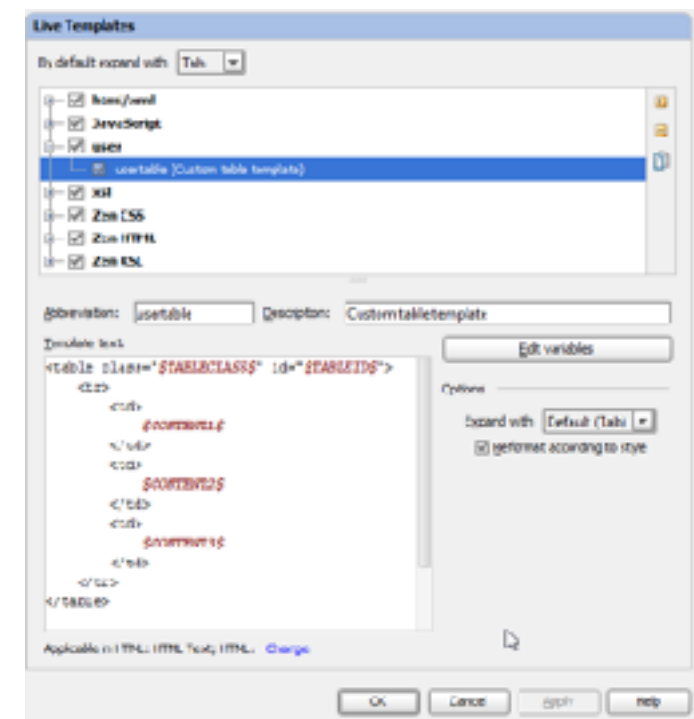
Idiom-Related Tools



Microsoft Visual Studio Code Assistant



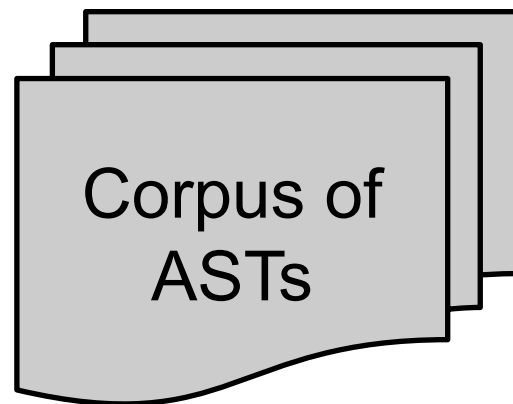
Eclipse SnipMatch



IntelliJ IDEA live templates

The Idiom Mining Problem

HAGGIS



Idioms

```
try{
  Node $name=$methodInvoc();
  $BODY$
}finally{
  $(Transaction).finish();
}

Location.distanceBetween(
  $(Location).getLatitude(),
  $(Location).getLongitude(),
  $...);

Document doc=Jsoup.connect(URL).
  userAgent("Mozilla").
  header("Accept","text/html").
  get();

Toast.makeText(this,
  $stringLiteral,Toast.LENGTH_SHORT)
  .show()

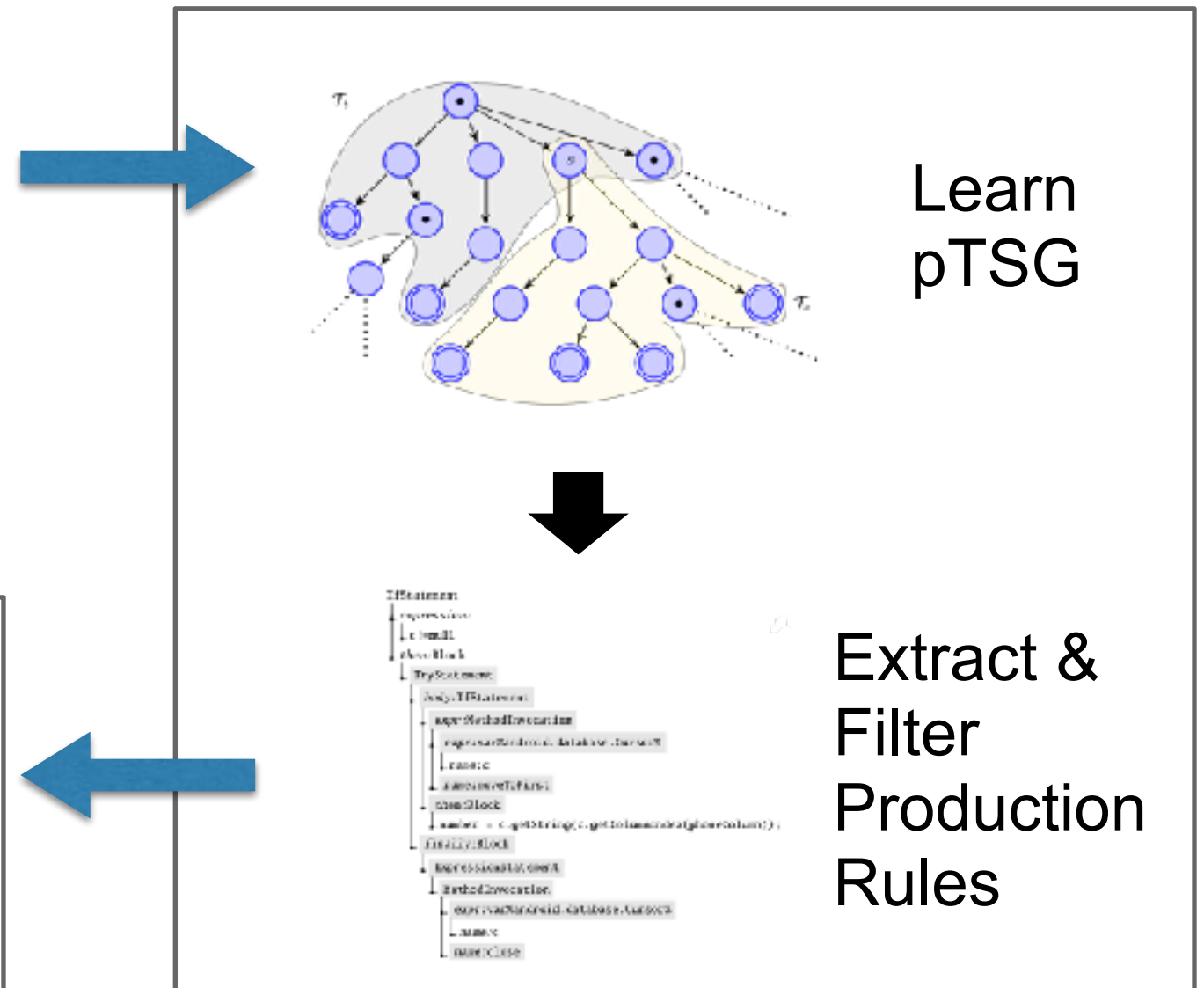
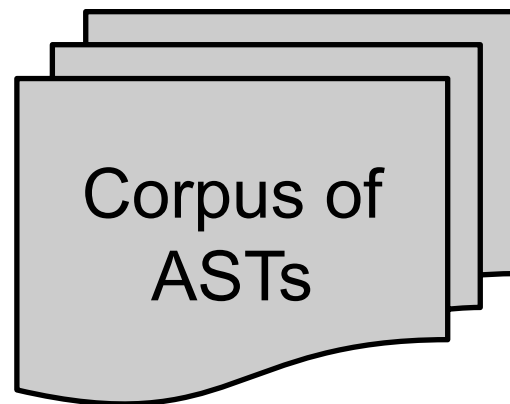
while (($String) = $(BufferedReader).
  readLine()) != null) {
  $BODY$
}
```



Holistic, Automatic Gathering of
Grammatical Idioms from Software

The Idiom Mining Problem

HAGGIS



Idioms

```
try{
    Node $name=$methodInvoc();
    $BODY$
}finally{
    $(Transaction).finish();
}

Location.distanceBetween(
    $(Location).getLatitude(),
    $(Location).getLongitude(),
    $...);

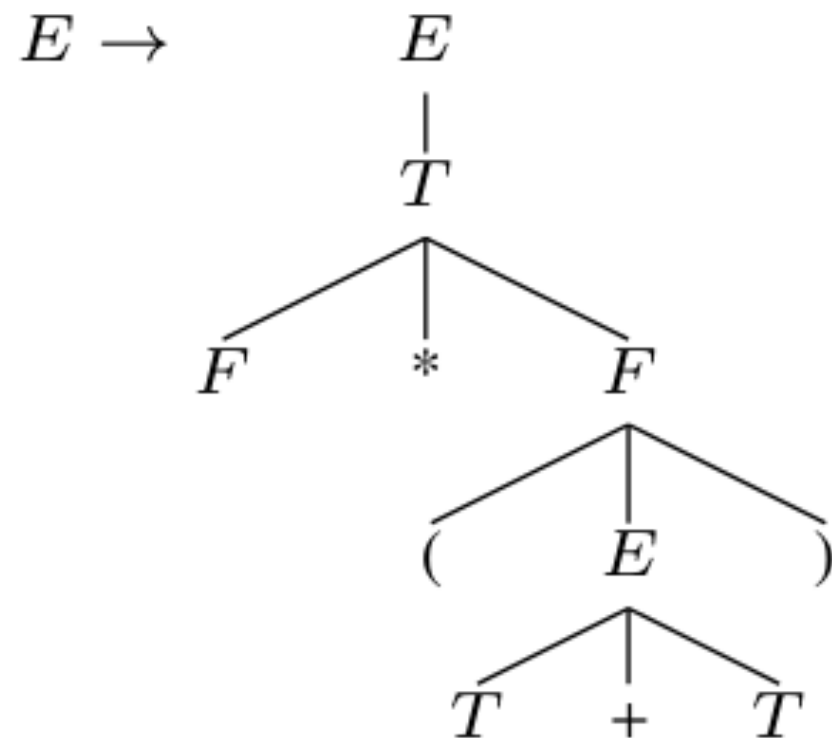
Document doc=Jsoup.connect(URL).
    userAgent("Mozilla").
    header("Accept","text/html").
    get();

Toast.makeText(this,
    $stringLit,Toast.LENGTH_SHORT)
    .show()

while (($String) = $(BufferedReader).
    readLine()) != null {
    $BODY$
}
```

Holistic, Automatic Gathering of Grammatical Idioms from Software

Probabilistic TSGs



Probability: 0.3

$$E \rightarrow E + E$$

Probability: 0.4

$$E \rightarrow T$$

Probability: 0.3

Given a CFG and corpus,
Infer elementary trees and
their probabilities

[Joshi and Schabes, 1997]

[Cohn et al, 2010]

[Post, and Gildea, 2009]

Inferring TSGs

Maximum likelihood maximizes:

$$P(T_1 \dots T_N | \theta)$$

θ : pTSG rules

- Selects the rules that best explain the corpus
- Problem: Overfitting

Inferring TSGs

Using Bayes Rule:

$$P(\theta | T_1 \dots T_N)$$

Posterior Distribution

θ : pTSG rules

Approximate using Markov Chain Monte Carlo

Type-based MCMC: [Liang, Jordan, Klein 2010]

Random code

```
public class JavaProjectionCalculator {
    private boolean enableCollapsing;
    public void setCollapsing(boolean collapseOn){
        enableCollapsing=collapseOn;
    }
    public Map findAnnotations(IJavaElement parentElement){
        try {
            Throwable result=new HashMap();
            findAnnotations((double)com.google.common.base.Preconditions,result);
            return result;
        } catch (JavaModelException e) { }
        return true;
    }
    private TSGNode findAnnotations(ProjectionAnnotation annotation, TableColumn
result) throws JavaModelException {
        int nextId;
        int elemType=elem.getElementType();
        Set regions=null;
        try {
            regions=computeProjections(owner);
        } catch ( RuntimeException e) {
            e.printStackTrace();
            throw e;
        }
        if (elem instanceof IParent) {
            IJavaElement[] children=((IParent)owner).getChildren();
            for (int fromPosition=0; i < children.length; i++) {
                IJavaElement aChild=children[i];
                Set childRegions=findAnnotations(aChild,result);
            }
        }
    }
}
```

Evaluation

- Qualitative analysis
- Precision and coverage in held out set
- External evaluation: StackOverflow
- Idioms and the real world: Eclipse SnipMatch

Projects Dataset

Name	Forks	Stars	Files	Description
arduino	2633	1533	180	Electronics Prototyping
atmosphere	1606	370	328	WebSocket Framework
bigbluebutton	1018	1761	760	Web Conferencing
elasticsearch	5972	1534	3525	REST Search Engine
grails-core	936	492	831	Web App Framework
hadoop	756	742	4985	Map-Reduce Framework
hibernate	870	643	6273	ORM Framework
libgdx	2903	2342	1985	Game Dev Framework
netty	2639	1090	1031	Net App Framework
storm	1534	7928	448	Distributed Computation
vert.x	2739	527	383	Application platform
voldemort	347	1230	936	NoSQL Database
wildfly	1060	1040	8157	Application Server

Library Dataset

Package Name	Files	Description
android.location	1262	Android location API
android.net.wifi	373	Android WiFi API
com.rabbitmq	242	Messaging system
com.spatial4j	65	Geospatial library
io.netty	65	Network app framework
opennlp	202	NLP tools
org.apache.hadoop	8467	Map-Reduce framework
org.apache.lucene	4595	Search Server
org.elasticsearch	338	REST Search Engine
org.eclipse.jgit	1350	Git implementation
org.hibernate	7822	Persistence framework
org.jsoup	335	HTML parser
org.mozilla.javascript	1002	JavaScript implementation
org.neo4j	1294	Graph database
twitter4j	454	Twitter API

Mined Idioms (General Java)

Iterate through the elements of an Iterator

```
for (Iterator iter=$methodInvoc;  
    iter.hasNext(); )  
    {$BODY$}
```

Looping through lines from a BufferedReader

```
while (($String) = $(BufferedReader).  
    readLine()) != null) {  
    $BODY$  
}
```

Creating a logger for a class

```
private final static Log $name=  
    LogFactory.getLog($type.class);
```

Defining a String constant

```
public static final  
    String $name = $StringLit;
```

Mined Idioms (Library-Specific)

Database transaction in node4j

```
try{
    Node $name=$methodInvoc();
    $BODY$
}finally{
    $(Transaction).finish();
}
```

Get the distance between two points in Android

```
Location.distanceBetween(
    $(Location).getLatitude(),
    $(Location).getLongitude(),
    $...);
```

Get an HTML Document in jsoup

```
Document doc=Jsoup.connect(URL).
    userAgent("Mozilla").
    header("Accept","text/html").
    get();
```

Show a small popup in Android

```
Toast.makeText(this,
    $stringLiteral,Toast.LENGTH_SHORT)
    .show()
```

Idioms in StackOverflow



Test Corpus	Coverage	Precision
Stack Overflow	31%	67%
PROJECTS	22%	50%

Mined idioms are more common in example code

Eclipse SnipMatch

Currently contains ~100 human-created code snippets
(Eclipse Recommenders Project)



We submitted 44 snippets, of which:

- 19 already in SnipMatch
- 5 accepted
- 4 unsupported by tool
- 1 rejected as a bad practice
- 15 still waiting

Why patterns in software?

Orthogonal interfaces

Tools that “do one thing well” need to be combined well

Surface-semantic correspondence

Semantics available from *glancing* rather than *reading*

```
void addOne (int[] arr) {  
    for (int i = 0; i < arr.length; i++) {  
        arr[i] += 1;  
    }  
}
```

Natural code: Code with
good correspondence?

```
void foo (int[] bar) {  
    int baz = 0;  
    while (true) {  
        bar[baz] = bar[baz] + 1;  
        baz = baz + 1;  
        if (baz > bar.length) break;  
    }  
}
```

API Mining

[Zhong et al, 2009; Dang et al 2013]



Library

e.g. **TWITTER4J**

```
ConfigurationBuilder.<init>()
ConfigurationBuilder.setAuthConsumerKey()
ConfigurationBuilder.setAuthConsumerSecret()
ConfigurationBuilder.build()
TwitterFactory.<init>()
TwitterFactory.getInstance()
```

API patterns

Coding

Mining

Documentation
Suggestion

```
private FinchTwitterFactory(Context context) {
    mContext = context;
    installHttpCache();
    ConfigurationBuilder configurationBuilder = new ConfigurationBuilder();
    configurationBuilder.setAuthConsumerKey(ConsumerKey.CONSUMER_KEY);
    configurationBuilder.setAuthConsumerSecret(ConsumerKey.CONSUMER_SECRET);
    configurationBuilder.setUseSSL(true);
    Configuration configuration = configurationBuilder.build();
    mTwitter = new TwitterFactory(configuration).getInstance();
}

public Twitter getTwitterInstance() {
    ConfigurationBuilder cb = new ConfigurationBuilder();
    cb.setAuthConsumerKey(Keys.consumerKey);
    cb.setAuthConsumerSecret(Keys.consumerSecret);
    cb.setOAuthAccessToken(settings.getString("accessToken", null));
    cb.setOAuthAccessSecret(settings.getString("accessSecret", null));
    TwitterFactory tf = new TwitterFactory(cb.build());
    return tf.getInstance();
}

private void startOAuth() {
    ConfigurationBuilder configurationBuilder = new ConfigurationBuilder();
    configurationBuilder.setAuthConsumerKey(Const.CONSUMER_KEY);
    configurationBuilder.setAuthConsumerSecret(Const.CONSUMER_SECRET);
    twitter = new TwitterFactory(configurationBuilder.build()).getInstance();

    try {
        requestToken = twitter.getOAuthRequestToken(Const.CALLBACK_URL);
        Toast.makeText(this, "Please authorize this app!", Toast.LENGTH_LONG).show();
        this.startActivity(new Intent(Intent.ACTION_VIEW,
            Uri.parse(requestToken.getAuthenticationURL() + "&force_login=true")));
    } catch (TwitterException e) {
        e.printStackTrace();
    }
}
```

brk3 / finch

jrupac/CleanTwitter

katahirado/tsubunomi

Corpus of client code

API Mining from Github

Modern development is layers of libraries

Average Java file on Github:

Imports from **2.1** packages outside project

45% of files import an external package

(Not counting `java.*` `javax.*` `sun.*`)

Github Java corpus (Allamanis and Sutton, 2013)

13000+ projects with at least one fork, 2M+ Java files

<http://groups.inf.ed.ac.uk/cup/javaGithub/>

(heuristic analysis)

API Mining

[Zhong et al, 2009; Dang et al 2013]




Library

e.g.  **TWITTER4J**

```
ConfigurationBuilder.<init>
ConfigurationBuilder.setOAuthConsumerKey
ConfigurationBuilder.setOAuthConsumerSecret
ConfigurationBuilder.build
TwitterFactory.<init>
TwitterFactory.getInstance
```

```
TwitterFactory.<init>
TwitterFactory.getInstance
Twitter.setOAuthConsumer
Twitter.setOAuthAccessToken
```

API patterns


Coding



API Mining

Documentation
Suggestion

```
private FinchTwitterFactory(Context context) {
    mContext = context;

    installHttpResponseBodyCache();

    ConfigurationBuilder configurationBuilder = new ConfigurationBuilder();
    configurationBuilder.setOAuthConsumerKey(ConsumerKey.CONSUMER_KEY);
    configurationBuilder.setOAuthConsumerSecret(ConsumerKey.CONSUMER_SECRET);
    configurationBuilder.setUseSSL(true);
    Configuration configuration = configurationBuilder.build();
    mTwitter = new TwitterFactory(configuration).getInstance();
}
```


 brk3 / finch

```
public Twitter getTwitterInstance() {
    ConfigurationBuilder cb = new ConfigurationBuilder();
    cb.setOAuthConsumerKey(Keys.consumerKey);
    cb.setOAuthConsumerSecret(Keys.consumerSecret);
    cb.setOAuthAccessToken(mSettings.getString("accessToken", null));
    cb.setOAuthAccessTokenSecret(mSettings.getString("accessSecret", null));
    TwitterFactory tf = new TwitterFactory(cb.build());
    return tf.getInstance();
}
```

 jrupac/CleanTwitter

```
private void startOAuth() {
    ConfigurationBuilder configurationBuilder = new ConfigurationBuilder();
    configurationBuilder.setOAuthConsumerKey(Const.CONSUMER_KEY);
    configurationBuilder.setOAuthConsumerSecret(Const.CONSUMER_SECRET);
    twitter = new TwitterFactory(configurationBuilder.build()).getInstance();

    try {
        requestToken = twitter.getOAuthRequestToken(Const.CALLBACK_URL);
        Toast.makeText(this, "Please authorize this app!", Toast.LENGTH_LONG).show();
        this.startActivity(new Intent(Intent.ACTION_VIEW,
            Uri.parse(requestToken.getAuthenticationURL() + "&force_login=true")));
    } catch (TwitterException e) {
        e.printStackTrace();
    }
}
```

 katahirado/tsubunomi

Corpus of client code



Frequent Sequence Mining

Return all patterns with \geq given support

*[Agrawal and Srikant, 1995;
Wang and Han, 2004]*

Support of pattern: Number of database sequences that contain it

b d b a f e c

b c e a

e d a f c

a e f b

b d a e f c

Database of sequences



d a f c

b a f c

a e

b e

e c

...

Sequence patterns

(e.g. minimum support = 3)

Problem: Frequent can be trivial!

Fundamental Pathologies

Truncation

d a f c

Real pattern



a c

Could be returned
(more frequent!)

Spurious correlation

Support(**a**) = 90%

Support(**d**) = 90%

... but independent ...



d a

Pattern at 81%
min_support

Freerider

a f c real pattern

Support(**d**) = 90%

... but independent ...



a d f c

for high enough
min_support

Effect: Redundant
list of patterns

For API Mining...

```
TwitterFactory.<init>  
TwitterFactory.getInstance
```

```
TwitterFactory.<init>  
Twitter.setOAuthConsumer
```

```
Status.getUser  
Status.getText
```

```
auth.AccessToken.<init>  
Twitter.setOAuthAccessToken
```

```
TwitterFactory.<init>  
TwitterFactory.getInstance  
Twitter.setOAuthConsumer  
Twitter.setOAuthAccessToken
```

```
TwitterFactory.getInstance  
Twitter.setOAuthConsumer
```

```
TwitterFactory.<init>  
TwitterFactory.getInstance  
Twitter.setOAuthConsumer
```

```
TwitterFactory.<init>  
Twitter.setOAuthAccessToken
```

```
TwitterFactory.<init>  
TwitterFactory.getInstance  
Twitter.setOAuthAccessToken
```

```
TwitterFactory.getInstance  
Twitter.setOAuthAccessToken
```

```
TwitterFactory.<init>  
Twitter.setOAuthConsumer  
Twitter.setOAuthAccessToken
```

*Top 10 API patterns
from pure sequence
mining (BIDE)*

Previous Approach: Cluster before/after

[Zhong et al, 2009; Dang et al 2013]

Interesting Sequence Mining

define a goodness measure on a *set* of patterns

Minimum description length

[Vreeken et al, 2011; Tatti and Vreeken, 2012; Lam et al 2014]

Use patterns to define a compression algorithm for database

Search for patterns that best compress

Probabilistic methods

[Fowkes and Sutton, KDD 2016, PKDD 2016]

Use patterns to define a probability distribution over database

Search for patterns that maximise database probability

(actually isomorphic; see MacKay, 2003)

Sequences more meaningful, less redundant

Probabilistic Sequence Mining

[Fowkes and Sutton, KDD 2016]

Define a distribution $P(\text{ database } | \text{ patterns })$

\mathcal{I}

[b c e]	: 0.1, 0.6
[d f]	: 0.7, 0.3
[d f]	: 0.8, 0.2
[e f]	: 0.8 , 0.1

*Sequence patterns
(with probabilities)*



Sample

Inclusion variables:

z_1 : **1**

z_2 : **1**

z_3 : **1**

z_4 : **0**

Interleave
randomly



$P(X, z | \mathcal{I})$

probability of generating X, z
from this process

X **b** **d** **c** **e** **d** **f** **f**

Sampled database sequence

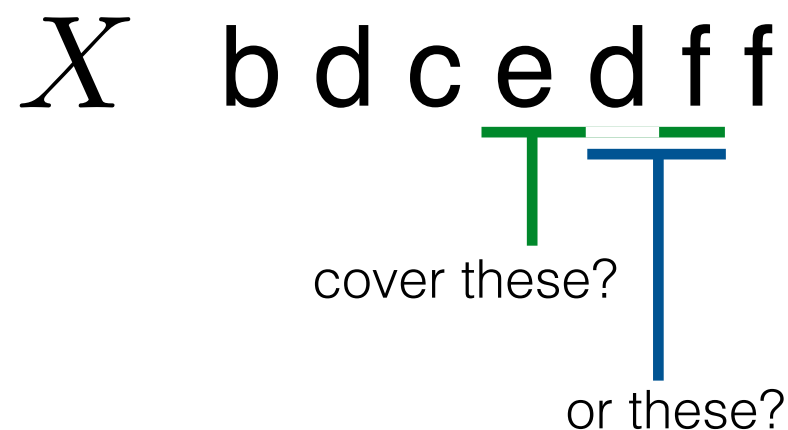
Probabilistic Sequence Mining

[Fowkes and Sutton, KDD 2016]

Model:

$$p(X, \mathbf{z} | \mathbf{\Pi}) = \frac{1}{|\mathcal{P}|} \prod_{S \in \mathcal{I}} \prod_{m=0}^{|\pi_S|-1} \pi_{S_m}^{[z_S=m]}$$

Inference: Determine $z | X, \mathcal{I}$



[b c e]	: 0.1, 0.6
[d f]	: 0.7, 0.3
[d f]	: 0.8, 0.2
[e f]	: 0.8, 0.1

Use greedy algorithm to

$$\max_z \log p(z | X, \mathcal{I})$$

(extension of weighted set cover)

Probabilistic Sequence Mining

[Fowkes and Sutton, KDD 2016]

Output of inference

b d c e d f f
e e d f f f
d f d d f f

z	[b c e]	[d f]	[d f]	[e f]
	1	1	1	0
	0	1	0	1
	0	1	1	1

Learning step: Infer \mathcal{I}

Update probabilities
(average of z)

Propose new patterns

Add to model

See if probability increases

\mathcal{I}

[b c e] : 0.3, 0.7
[d f] : 0.0, 1.0
[d f] : 0.7, 0.3
[e f] : 0.3, 0.7

Formally: Structural Expectation Maximization

Probabilistic API Miner (PAM)

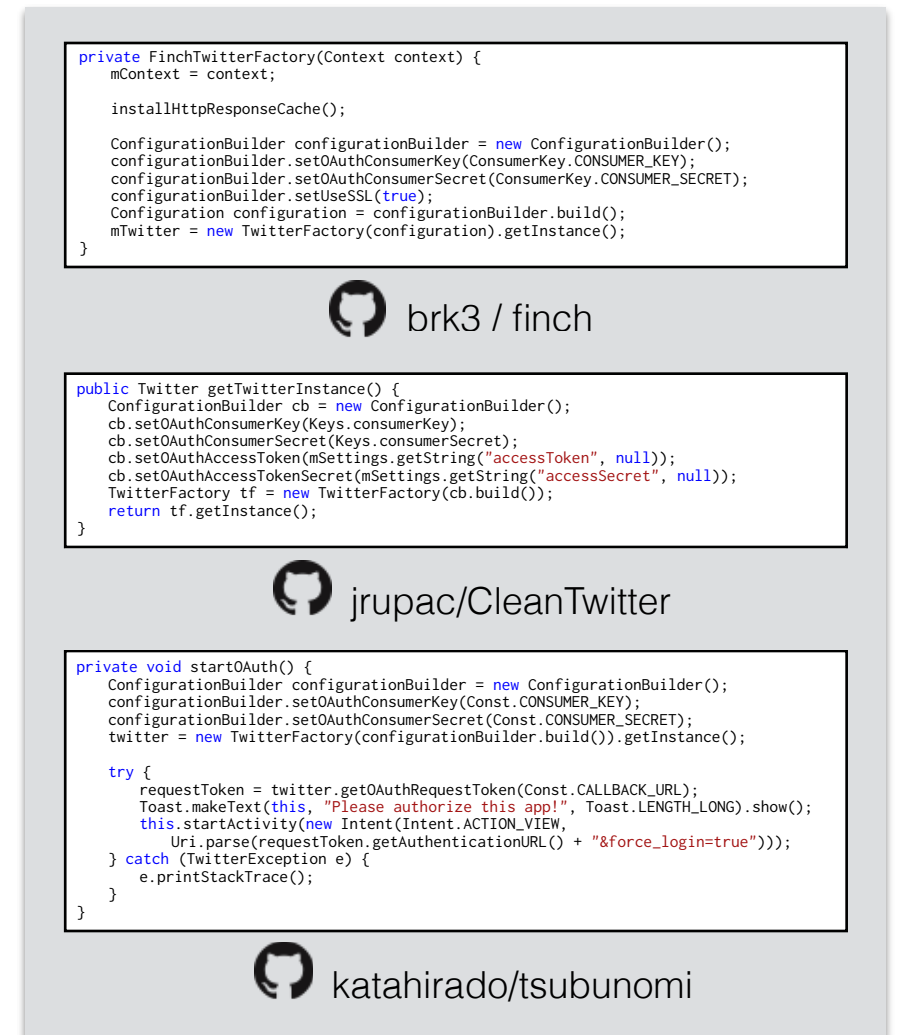
Interesting sequence mining for API mining

```
ConfigurationBuilder.<init>
ConfigurationBuilder.setOAuthConsumerKey
ConfigurationBuilder.setOAuthConsumerSecret
ConfigurationBuilder.setUseSSL
ConfigurationBuilder.build
TwitterFactory.<init>
TwitterFactory.getInstance
```

```
ConfigurationBuilder.<init>
ConfigurationBuilder.setOAuthConsumerKey
ConfigurationBuilder.setOAuthConsumerSecret
ConfigurationBuilder.setOAuthAccessToken
ConfigurationBuilder.setOAuthAccessTokenSecret
ConfigurationBuilder.build
TwitterFactory.<init>
TwitterFactory.getInstance
```

```
ConfigurationBuilder.<init>
ConfigurationBuilder.setOAuthConsumerKey
ConfigurationBuilder.setOAuthConsumerSecret
ConfigurationBuilder.build
TwitterFactory.<init>
TwitterFactory.getInstance
TwitterFactory.getOAuthRequestToken
RequestToken.getAuthenticationURL
```

Sequence database



```
private FinchTwitterFactory(Context context) {
    mContext = context;

    installHttpResponseBodyCache();

    ConfigurationBuilder configurationBuilder = new ConfigurationBuilder();
    configurationBuilder.setOAuthConsumerKey(ConsumerKey.CONSUMER_KEY);
    configurationBuilder.setOAuthConsumerSecret(ConsumerKey.CONSUMER_SECRET);
    configurationBuilder.setUseSSL(true);
    Configuration configuration = configurationBuilder.build();
    mTwitter = new TwitterFactory(configuration).getInstance();
}

public Twitter getTwitterInstance() {
    ConfigurationBuilder cb = new ConfigurationBuilder();
    cb.setOAuthConsumerKey(Keys.consumerKey);
    cb.setOAuthConsumerSecret(Keys.consumerSecret);
    cb.setOAuthAccessToken(mSettings.getString("accessToken", null));
    cb.setOAuthAccessTokenSecret(mSettings.getString("accessSecret", null));
    TwitterFactory tf = new TwitterFactory(cb.build());
    return tf.getInstance();
}

private void startOAuth() {
    ConfigurationBuilder configurationBuilder = new ConfigurationBuilder();
    configurationBuilder.setOAuthConsumerKey(Const.CONSUMER_KEY);
    configurationBuilder.setOAuthConsumerSecret(Const.CONSUMER_SECRET);
    twitter = new TwitterFactory(configurationBuilder.build()).getInstance();

    try {
        requestToken = twitter.getOAuthRequestToken(Const.CALLBACK_URL);
        Toast.makeText(this, "Please authorize this app!", Toast.LENGTH_LONG).show();
        this.startActivity(new Intent(Intent.ACTION_VIEW,
            Uri.parse(requestToken.getAuthenticationURL() + "&force_login=true")));
    } catch (TwitterException e) {
        e.printStackTrace();
    }
}
```

brk3 / finch

jrupal/CleanTwitter

katahirado/tsubunomi

Corpus

*Probabilistic
sequence mining*

Data

Target projects: 17 Java libraries, all that:

- Library source on Github

- Library in top 1000 Github projects

- Called by >50 other methods on Github

- At least 10k lines of **example**/ code

- Total: Over 300k lines of example code

Client methods: all that called any targets

- 967 client projects

- Total: Over 4M lines of client code

Experimental Questions

Quality

- Match to “held-out” client code

- Match to examples from library developers

- Measure: sequence overlap, precision, recall

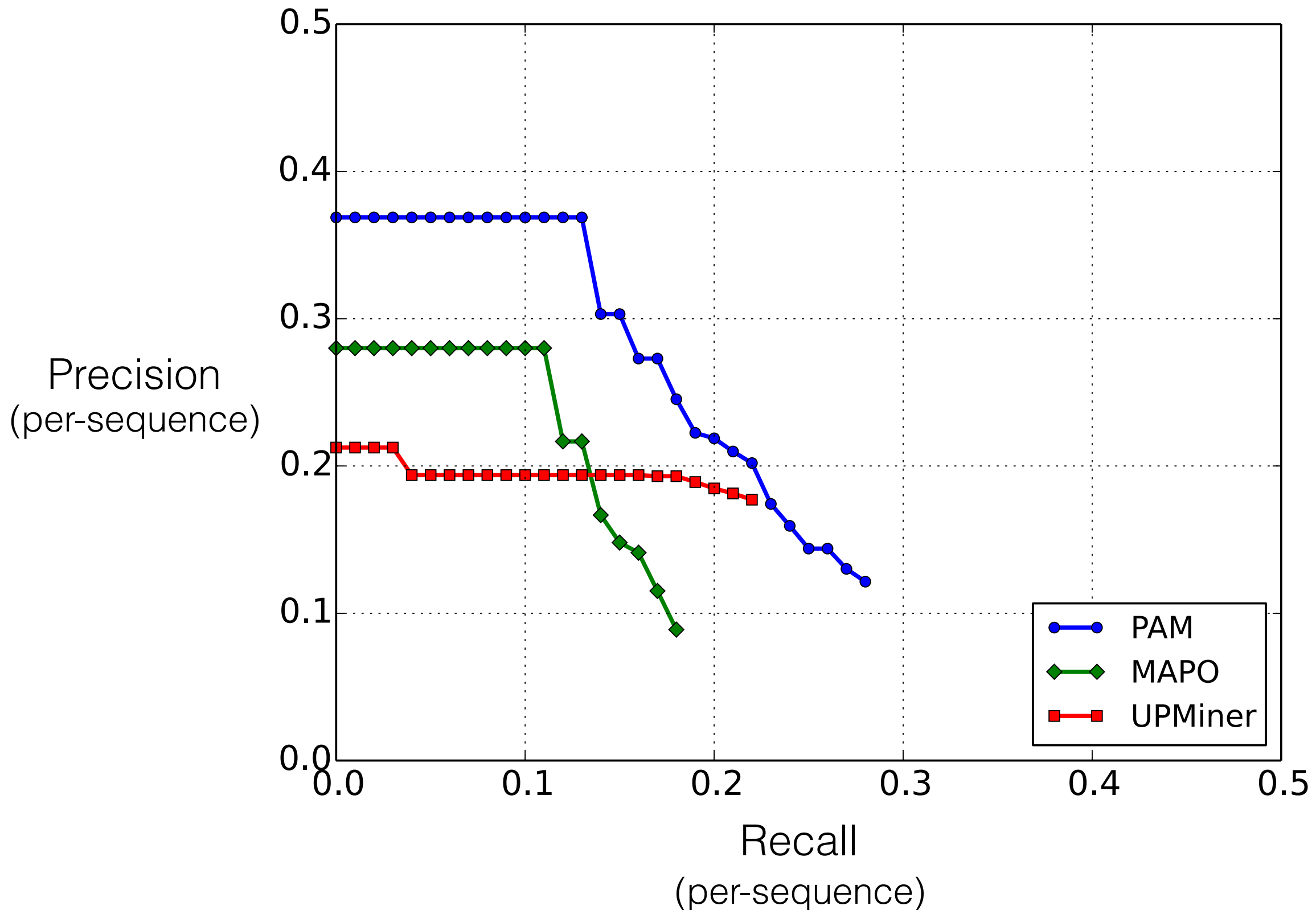
Redundancy

- Why? Ease of use, diversity

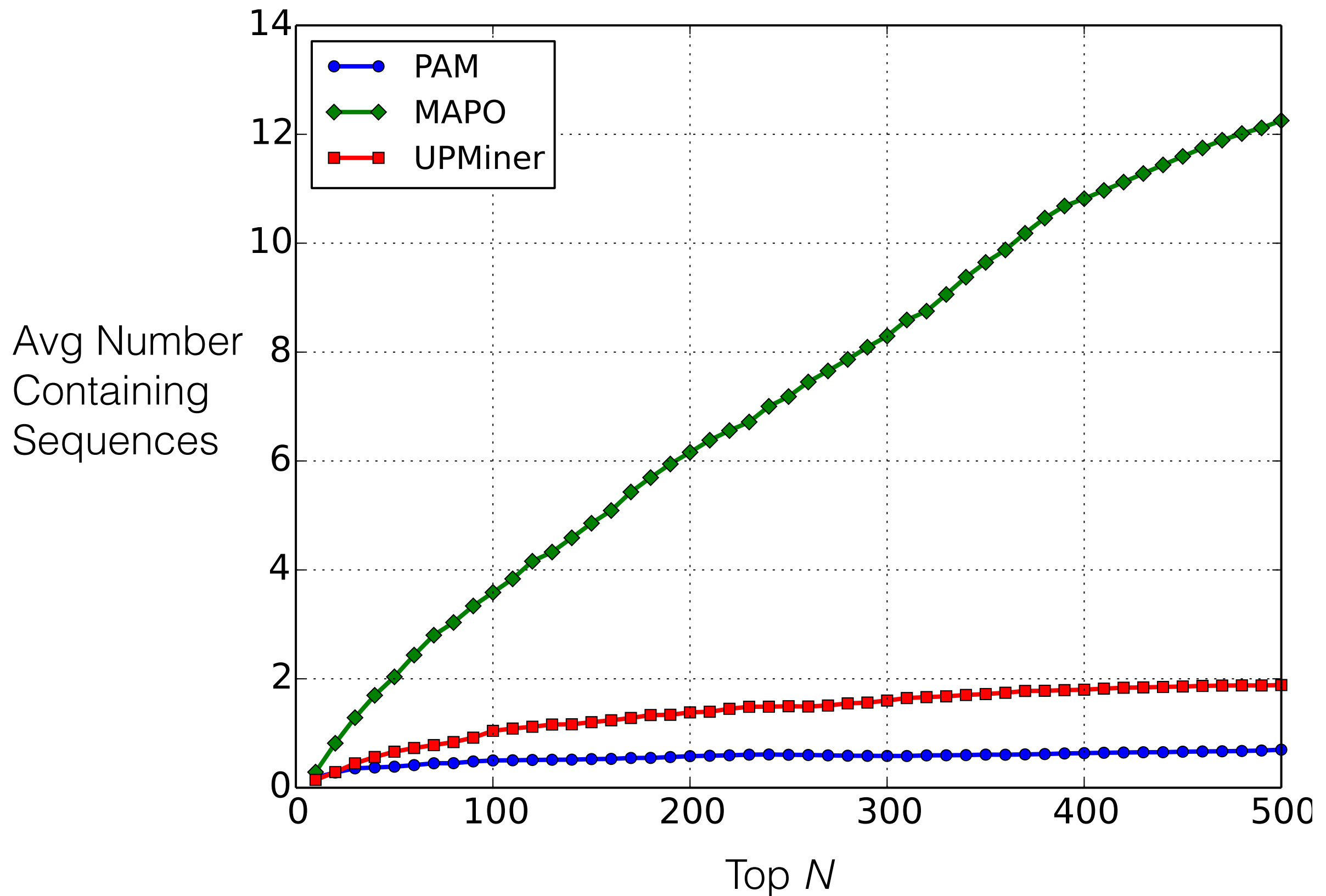
- Measure: number of containing sequences

All results averaged over the 17 libraries

Handwritten Examples



Redundancy



Example: twitter4j

PAM

MAPO

[Zhong et al, '09]

UPMiner

[Wang et al, '13]

TwitterFactory.<init>
TwitterFactory.getInstance

TwitterFactory.<init>
TwitterFactory.getInstance

TwitterFactory.<init>
TwitterFactory.getInstance

TwitterFactory.<init>
TwitterFactory.getInstance
Twitter.setOAuthConsumer
Twitter.setOAuthAccessToken

Status.getUser
Status.getText

TwitterFactory.getInstance
Twitter.setOAuthConsumer

Status.getUser
Status.getText

ConfigurationBuilder.<init>
ConfigurationBuilder.build

TwitterFactory.<init>
TwitterFactory.getInstance
Twitter.setOAuthConsumer

AccessToken.getToken
AccessToken.getTokenSecret

ConfigurationBuilder.<init>
TwitterFactory.<init>

Status.getUserStatus.getText

ConfigurationBuilder.<init>
ConfigurationBuilder.build
TwitterFactory.<init>
TwitterFactory.getInstance

ConfigurationBuilder.<init>
ConfigurationBuilder.setOAuthConsumerKey

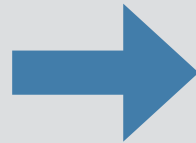
Twitter.setOAuthConsumer
Twitter.setOAuthAccessToken

■ : two main types of twitter initialization call

```

1 /* Header */
2 package org.zoolu.sip.header;
3
4 /** SIP Status-line, i.e. the first
5  * line of a response message */
6 public class StatusLine {
7     protected int code;
8     protected String reason;
9
10    /** Construct StatusLine */
11    public StatusLine(int c, String r) {
12        code = c;
13        reason = r;
14    }
15
16    /** Create a new copy of the request-line */
17    public Object clone() {
18        return new StatusLine(getCode(), getReason());
19    }
20
21    /** Indicates whether some other Object
22     * is "equal to" this StatusLine */
23    public boolean equals(Object obj){
24        try {
25            StatusLine r = (StatusLine) obj;
26            if (r.getCode() == getCode() &&
27                r.getReason().equals(getReason()))
28                return true;
29            else
30                return false;
31        } catch (Exception e) {
32            return false;
33        }
34    }
35
36    public int getCode() {
37        return code;
38    }
39
40    public String getReason() {
41        return reason;
42    }
43
44    public String toString() {
45        return "SIP/2.0 " + code + " " + reason + "\r\n";
46    }
47 }

```



```

1 /* Header... */
2 package org.zoolu.sip.header;
3
4 /** SIP Status-line, i.e. the first... */
5 public class StatusLine {
6     protected int code;
7     protected String reason;
8
9     /** Construct StatusLine... */
10    public StatusLine(int c, String r) {...}
11
12    /** Create a new copy of the request-line ... */
13    public Object clone() {...}
14
15    /** Indicates whether some other Object... */
16    public boolean equals(Object obj){
17        try {
18            StatusLine r = (StatusLine) obj;
19            if (r.getCode() == getCode() &&
20                r.getReason().equals(getReason()))
21                return true;
22            else
23                return false;
24        } catch (Exception e) {...}
25    }
26
27    public int getCode() {...}
28
29    public String getReason() {...}
30
31    public String toString() {...}
32 }

```

Code summarisation

```

1 /* Header */
2 package org.zoolu.sip.header;
3
4 /** SIP Status-line, i.e. the first
5  * line of a response message */
6 public class StatusLine {
7     protected int code;
8     protected String reason;
9
10    /** Construct StatusLine */
11    public StatusLine(int c, String r) {
12        code = c;
13        reason = r;
14    }
15
16    /** Create a new copy of the request-line */
17    public Object clone() {
18        return new StatusLine(getCode(), getReason());
19    }
20
21    /** Indicates whether some other Object
22     * is "equal to" this StatusLine */
23    public boolean equals(Object obj){
24        try {
25            StatusLine r = (StatusLine) obj;
26            if (r.getCode() == getCode() &&
27                r.getReason().equals(getReason()))
28                return true;
29            else
30                return false;
31        } catch (Exception e) {
32            return false;
33        }
34    }
35
36    public int getCode() {
37        return code;
38    }
39
40    public String getReason() {
41        return reason;
42    }
43
44    public String toString() {
45        return "SIP/2.0 " + code + " " + reason + "\r\n";
46    }
47 }

```

statusline.java from BigBlueButton

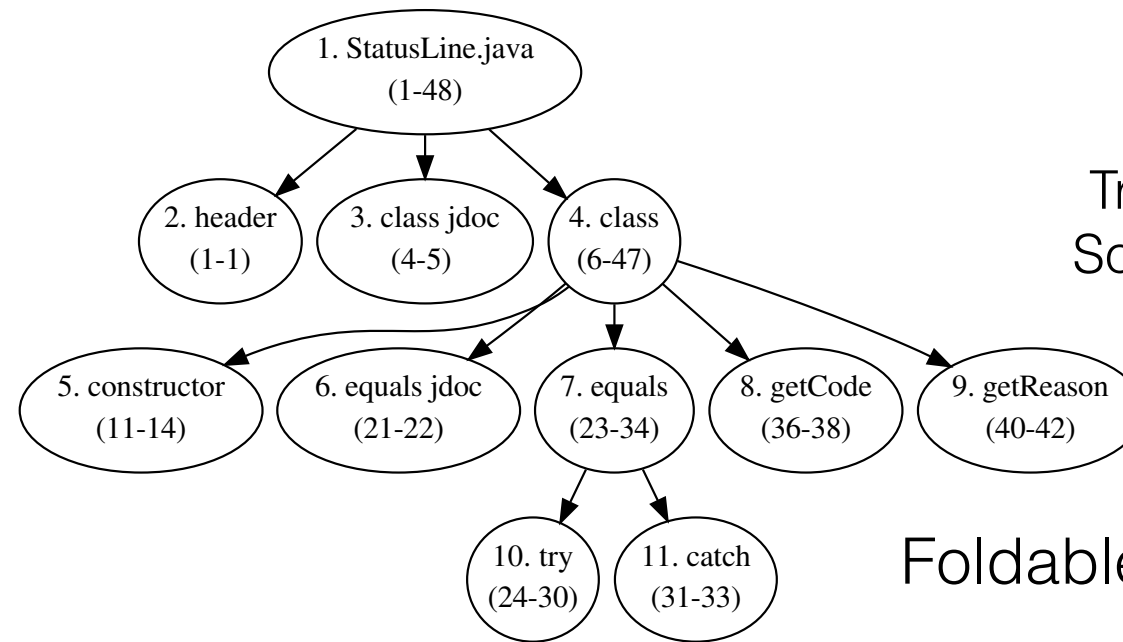
```

1 /* Header...*/
2 package org.zoolu.sip.header;
3
4 /** SIP Status-line, i.e. the first...*/
5 public class StatusLine {
6     protected int code;
7     protected String reason;
8
9
10    /** Construct StatusLine...*/
11    public StatusLine(int c, String r) {...}
12
13
14    /** Create a new copy of the request-line ...*/
15    public Object clone() {...}
16
17
18    /** Indicates whether some other Object...*/
19    public boolean equals(Object obj){
20        try {
21            StatusLine r = (StatusLine) obj;
22            if (r.getCode() == (getCode() &&
23                r.getReason().equals(getReason()))
24                return true;
25            else
26                return false;
27        } catch (Exception e) {...}
28    }
29
30    public int getCode() {...}
31
32    public String getReason() {...}
33
34    public String toString() {...}
35 }

```

TASSAL

Tree-based Autofolding
Software Summarization
Algorithm



Foldable Tree

```
1/* Header */
2package org.zoolu.sip.header;
3
4/** SIP Status-line, i.e. the first
5 * line of a response message */
6public class StatusLine {
7    protected int code;
8    protected String reason;
9
10    /** Construct StatusLine */
11    public StatusLine(int c, String r) {
12        code = c;
13        reason = r;
14    }
15
16    /** Create a new copy of the request-line */
17    public Object clone() {
18        return new StatusLine(getCode(), getReason());
19    }
20
21    /** Indicates whether some other Object
22     * is "equal to" this StatusLine */
23    public boolean equals(Object obj){
24        try {
25            StatusLine r = (StatusLine) obj;
26            if (r.getCode() == getCode() &&
27                r.getReason().equals(getReason()))
28                return true;
29            else
30                return false;
31        } catch (Exception e) {
32            return false;
33        }
34    }
35
36    public int getCode() {
37        return code;
38    }
39
40    public String getReason() {
41        return reason;
42    }
43
44    public String toString() {
45        return "SIP/2.0 " + code + " " + reason + "\r\n";
46    }
47}
```



File



Bag of "words"



TopicSum

[Haghighi and Vanderwende, 2009]



Optimization

match unfolded nodes
to file topic

src	connection	"File" topic
bean	spring	"Project" topic
get	value	"Java" topic

Example topics

Background	Project		File	
	spring-framework	bigbluebutton	DataSourceUtils	Qualsp
get	bean	sip	connection	lsp
string	org	org	con	j
value	test	log	holder	constants
name	context	it	source	k
type	springframework	event	data	ld8
object	exception	gnu	synchronizati on	mode
i	request	listener	isolation	tmp

Developer Study

	Conciseness	Usefulness
Gold	3.34	3.33
TASSAL	3.27	3.18
Javadocs	3.07	2.69
Shallowest	2.97	2.50
Largest	3.08	2.67

Six developers. Avg 4 years industry experience. 1...5 Likert scale

Statistical Analysis of Computer Program Text

Charles Sutton, University of Edinburgh

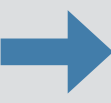
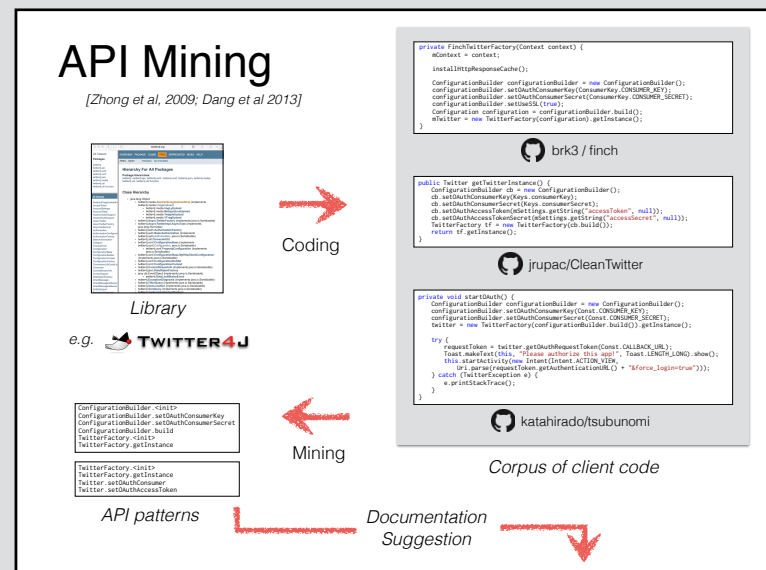


Companion web site

<http://bit.ly/sutton-nlpswe>

Our other related research

<https://mast-group.github.io>



```
1: // Header
2: package org.xmllib.sip.header;
3:
4: /** SIP Status-line, i.e. the first
5:  * line of a response message */
6: public class StatusLine {
7:     protected int code;
8:     protected String reason;
9:
10:    /** Construct StatusLine */
11:    public StatusLine(int c, String r) {
12:        code = c;
13:        reason = r;
14:    }
15:
16:    /** Create a new copy of the request-line */
17:    public Object clone() {
18:        return new StatusLine(getCode(), getReason());
19:    }
20:
21:    /** Indicates whether some other Object
22:     * is "equal to" this StatusLine */
23:    public boolean equals(Object obj) {
24:        try {
25:            StatusLine r = (StatusLine) obj;
26:            if (r.getCode() == getCode() &&
27:                r.getReason().equals(getReason()))
28:                return true;
29:            else
30:                return false;
31:        } catch (Exception e) {
32:            return false;
33:        }
34:    }
35:
36:    public int getCode() {
37:        return code;
38:    }
39:
40:    public String getReason() {
41:        return reason;
42:    }
43:
44:    public String toString() {
45:        return "SIP/2.0 " + code + " " + reason + "\r\n";
46:    }
47:}
```

Thanks!

- Miltiadis Allamanis, MSR
- Hao Peng, U Washington
- Jaroslav Fowkes, Oxford
- Razvan Ranca
- Mirella Lapata, UoE
- Pankajan Chantirasagaran
- Christian Bird, MSR
- Earl Barr, UCL

EPSRC
Engineering and Physical Sciences
Research Council

Microsoft
Research