

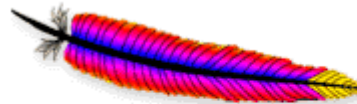
Typestate-Based Semantic Code Search Over Partial Programs

Alon Mishne^{*}, Sharon Shoham^{**}, Eran
Yahav^{*}

^{*} Technion - Israel Institute of Technology

^{**} Tel Aviv-Yafo Academic College

Components Are Everywhere



Apache Commons



Components are Accessed Via APIs

```
class File {  
    File(String);  
    void open();  
    void close();  
    byte read();  
    void write(byte);  
}
```

Components are Accessed Via APIs

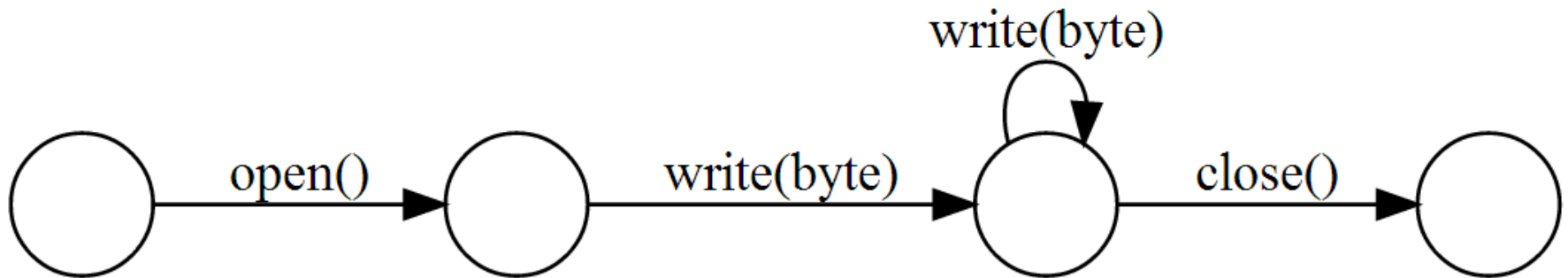
```
class File {  
    File(String);  
    void open();  
    void close();  
    byte read();  
    void write(byte);  
}
```

```
void writeBuff(File f, byte[] buff) {  
    f.open();  
    for (byte b : buff) f.write(b);  
    f.close();  
}
```

Components are Accessed Via APIs

```
class File {  
  File(String);  
  void open();  
  void close();  
  byte read();  
  void write(byte);  
}
```

```
void writeBuff(File f, byte[] buff) {  
  f.open();  
  for (byte b : buff) f.write(b);  
  f.close();  
}
```



There Are Many Available Examples



Code^{beta}

java.sql.Connection preparedStatement

Search

Filter Code Results

Definitions:

Method (151)

Projects:

ALT Linux (24.688)

Code Results

Results 1 - 10 of about 1,092,647 results for

Keep current filters: (Deselect all Filters)

File: [PreparedStatement.java](#)

Proje

```
9 public class PreparedStatement extends Task<PrepareSta  
10  
11 @Override
```

1,092,647

There Are Many Available Examples

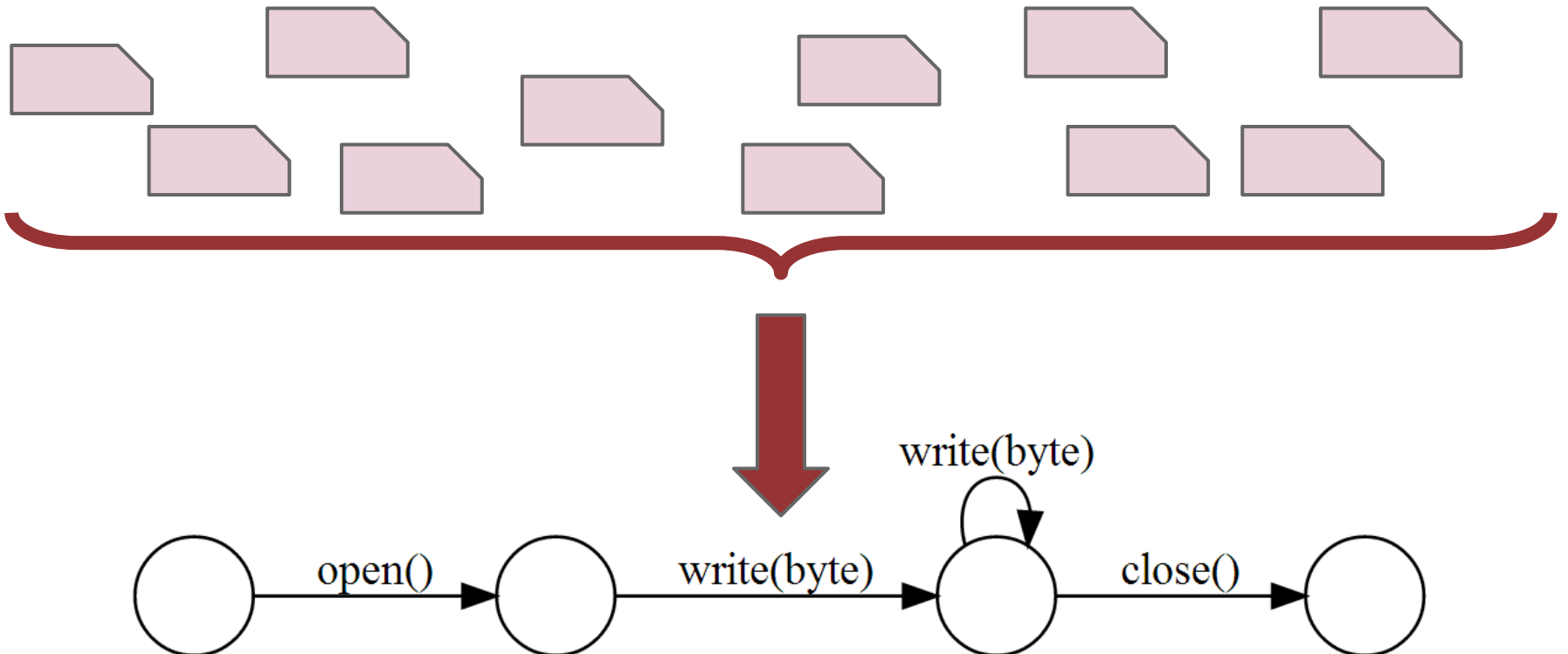
The screenshot shows the hloh Code search interface. At the top left is the hloh logo and 'Code beta' text. A search bar contains the text 'java.sql.Connection preparedStatement' and a 'Search' button is to its right. Below the search bar is a 'Filter Code Results' sidebar on the left with sections for 'Definitions:' (Method (151)) and 'Projects:' (ALT Linux (24.688)). The main 'Code Results' area shows a search for 'java.sql.Connection preparedStatement' with 'Results 1 - 10 of about 1,092,647 results found'. A red box highlights this result count, and a red arrow points from the number '1,092,647' below to the box. Below the search bar, there is a checkbox 'Keep current filters: (Deselect all Filters)'. The main content area shows a file 'PrepareStatement.java' with code snippets: '9 public class PreparedStatement extends Task<PreparedStatement>', '10', and '11 @Override'.

- Partial
- Noisy
- Which one?

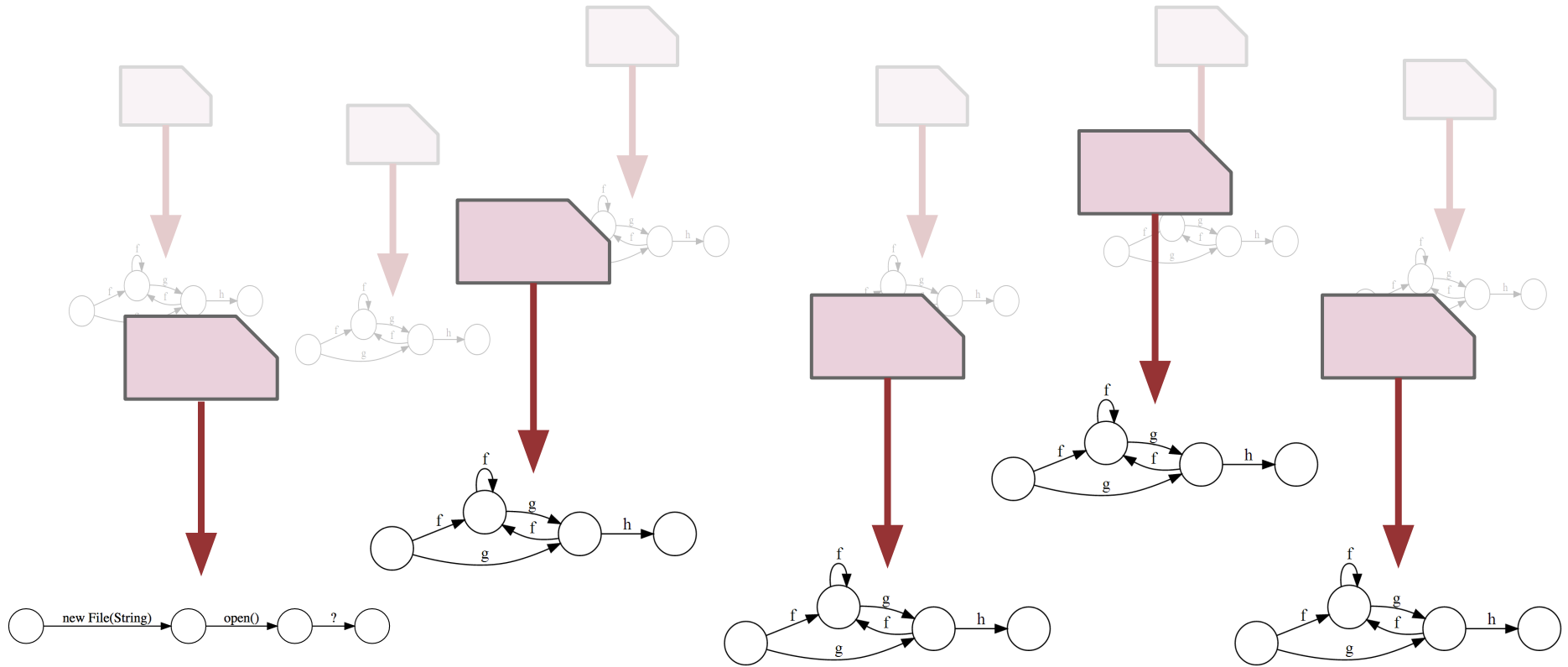
1,092,647

Challenge

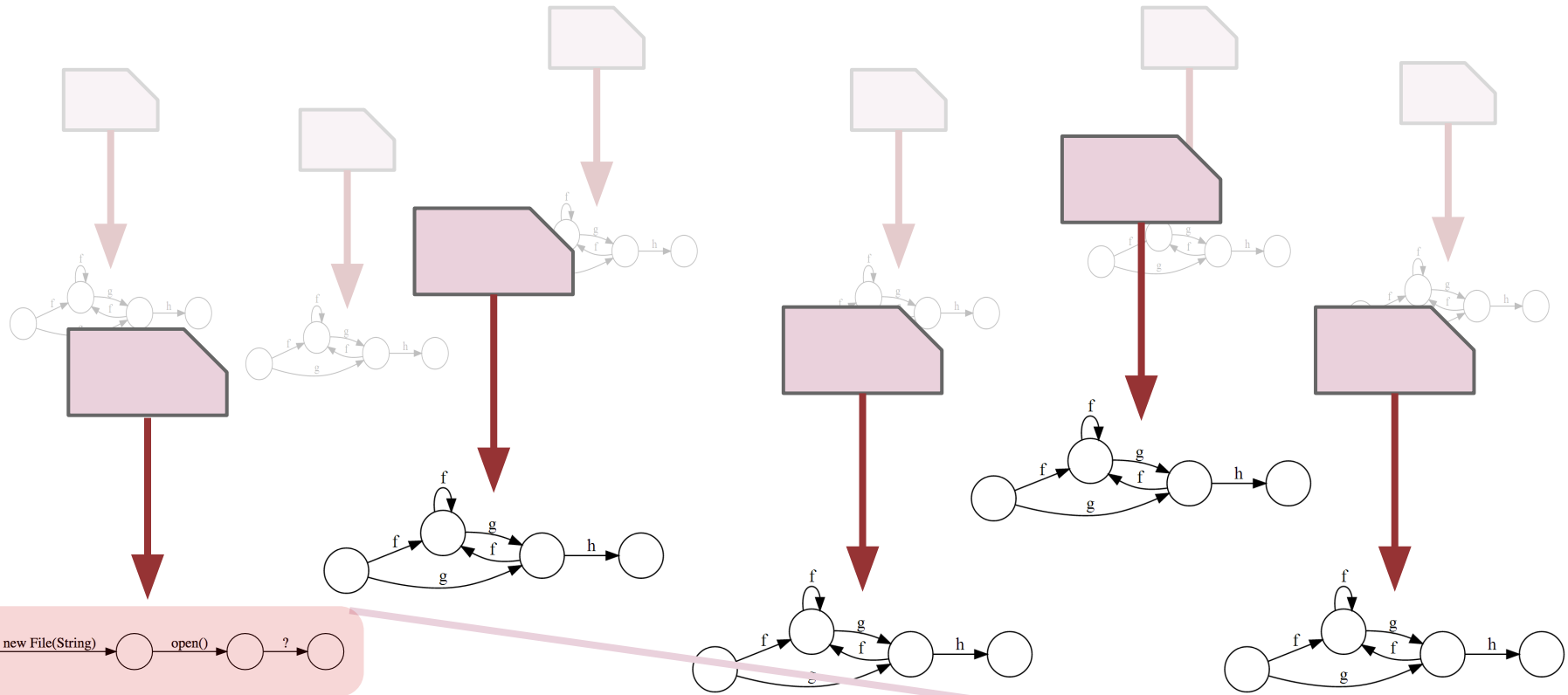
How do you leverage the huge number of examples for **understanding** how to use the API?



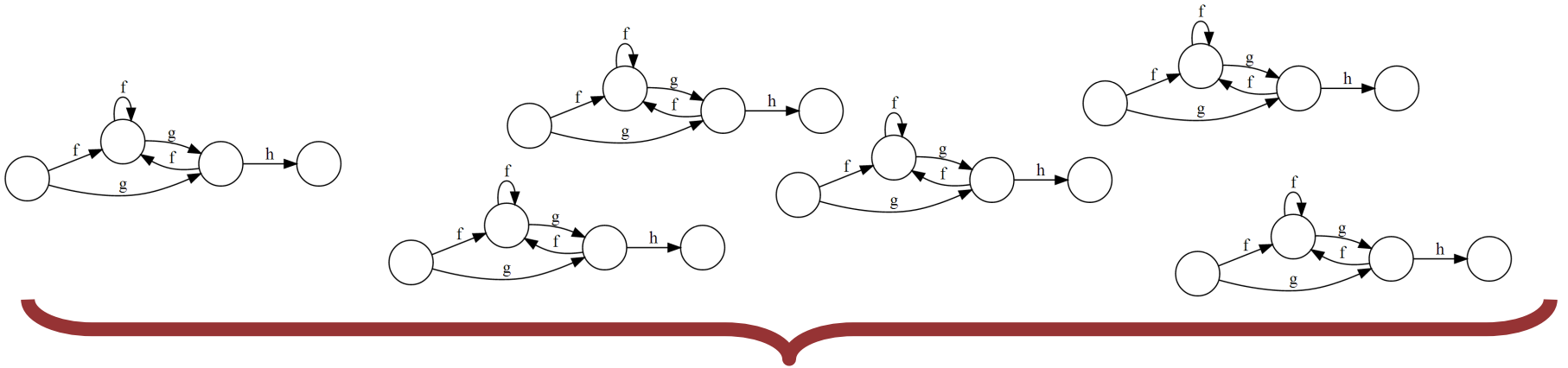
Partial Specification from each Snippet



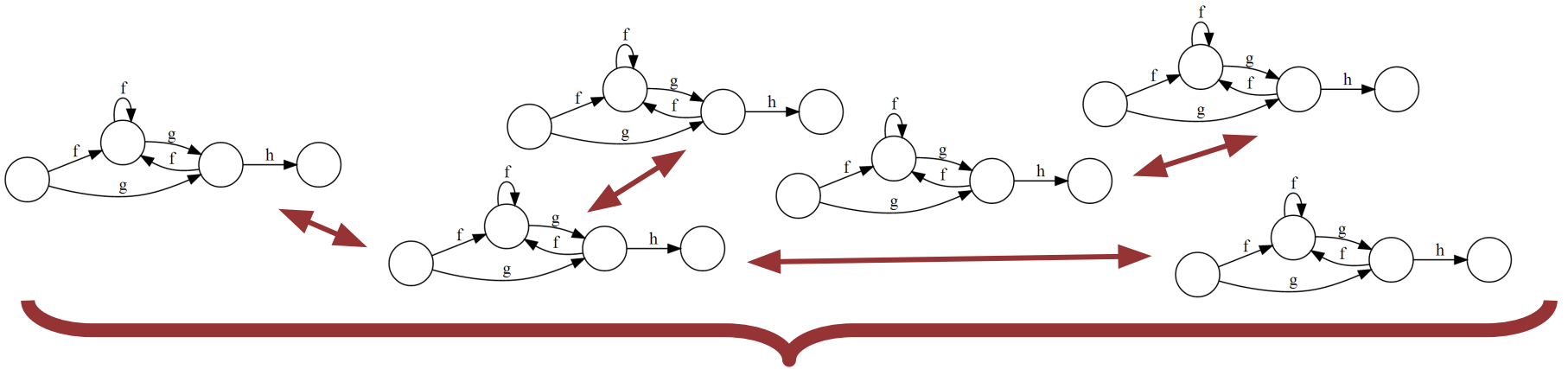
Partial Specification from each Snippet



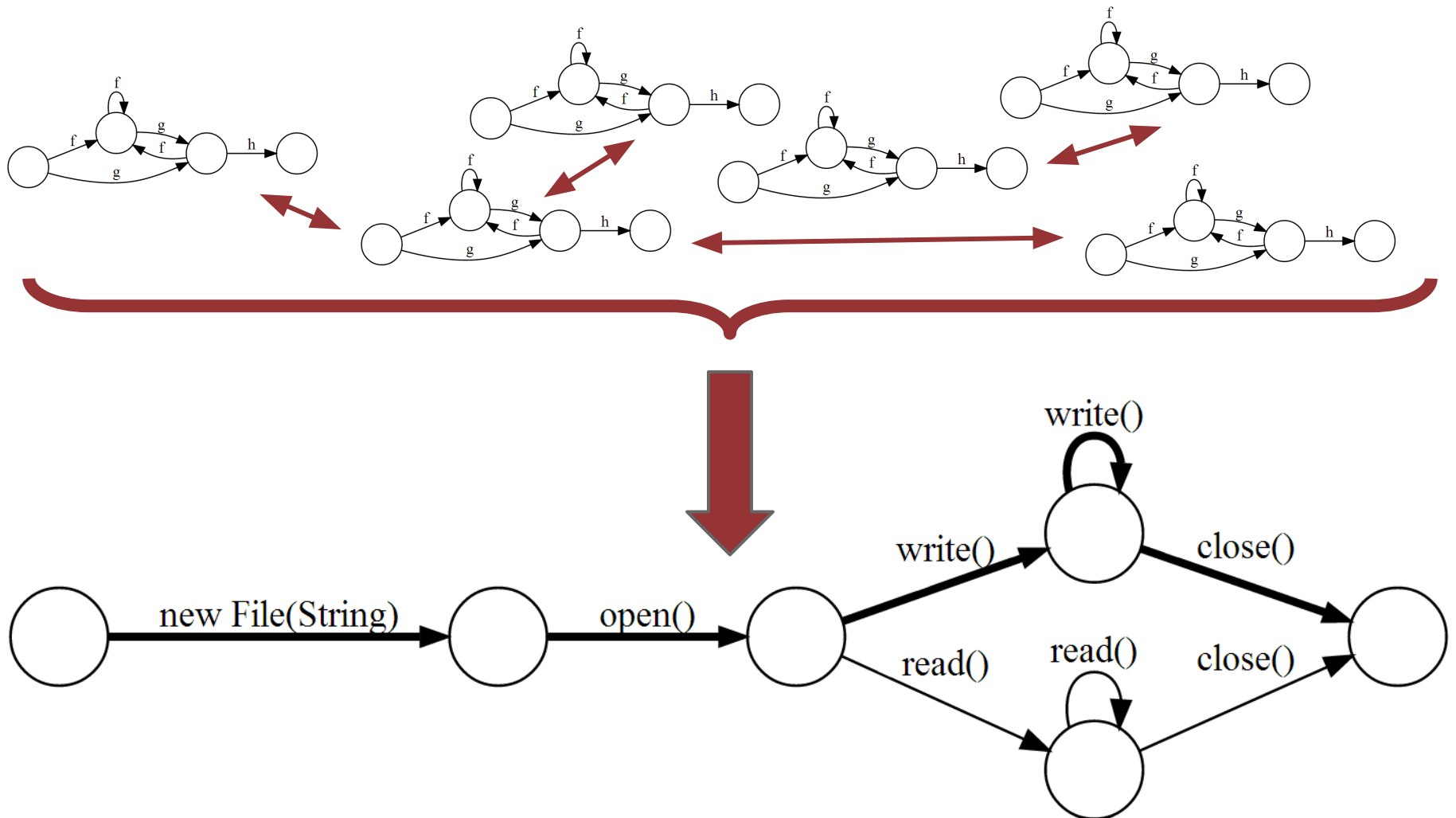
Consolidation Into Larger Specifications



Consolidation Into Larger Specifications



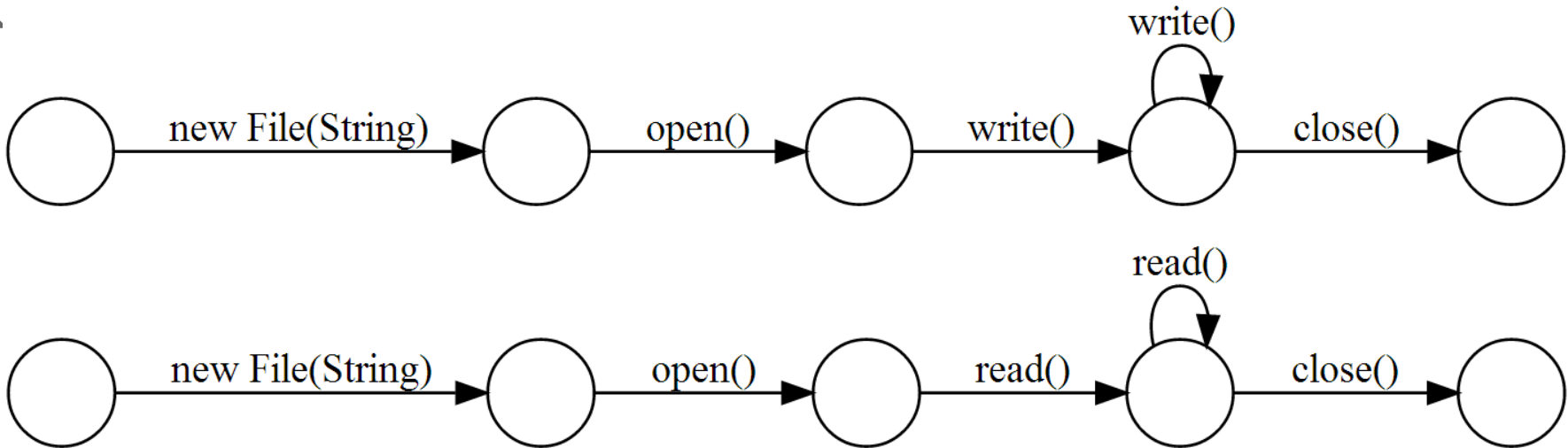
Consolidation Into Larger Specifications



Query Against The Consolidated Specification

```
File f = ?  
f.open();  
f.?
```

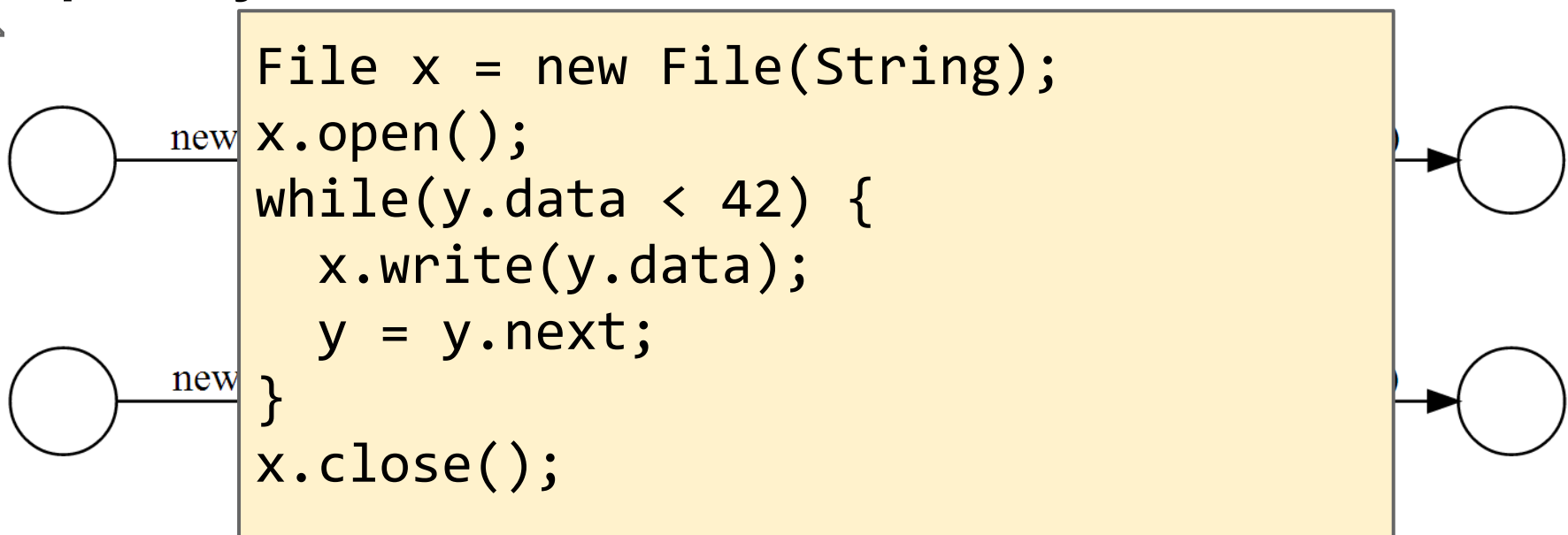
Popularity



Query Against The Consolidated Specification

```
File f = ?  
f.open();  
f.?
```

Popularity

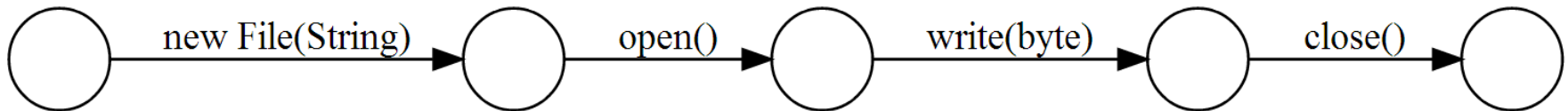


Details

1. **Extracting partial specification**
 - Data representation
 - Dealing with missing information
2. **Consolidating partial specifications**
 - Completing missing information
 - Tracking popularity
3. **Query matching**
 - Matching queries to specifications
 - Ranking by popularity
 - Getting back code examples

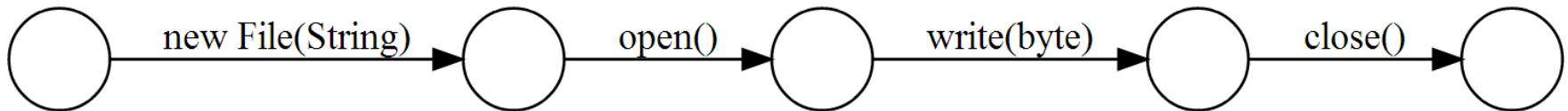
Client-Side Static Analysis

```
1 File f = new File(filename);  
2 f.open();  
3 f.write(...);  
4 f.close();
```



Client-Side Static Analysis

```
1 File f = new File(filename);  
2 f.open();  
3 f.write(...);  
4 f.close();
```



- Handles branches and loops
- Interprocedural
- Alias-aware with points-to information

Abstraction

```
while(...) {  
    File f = new File(filename);  
    f.open();  
    while(...) f.write(...);  
    f.close();  
}
```

Abstraction

```
while(...) {  
  File f = new File(filename);  
  f.open();  
  while(...) f.write(...);  
  f.close();  
}
```

Unbounded
sequence of
events

Abstraction

```
while(...) {  
  File f = new File(filename);  
  f.open();  
  while(...) f.write(...);  
  f.close();  
}
```

Unbounded
number of
objects

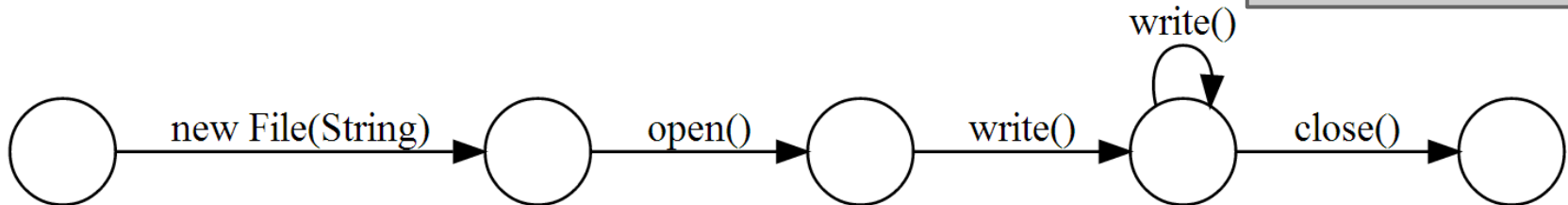
Unbounded
sequence of
events

Abstraction

```
while(...) {  
  File f = new File(filename);  
  f.open();  
  while(...) f.write(...);  
  f.close();  
}
```

Unbounded
number of
objects

Unbounded
sequence of
events



- Represent unbounded sequence of events using finite-state automaton
- Represent unbounded number of concrete objects using abstract objects

Dealing With Multiple Related Objects

```
class FileManager {  
    FileManager();  
    File getFile(String);  
}
```

```
class File {  
    void open();  
    void close();  
    byte read();  
    void write(byte);  
}
```

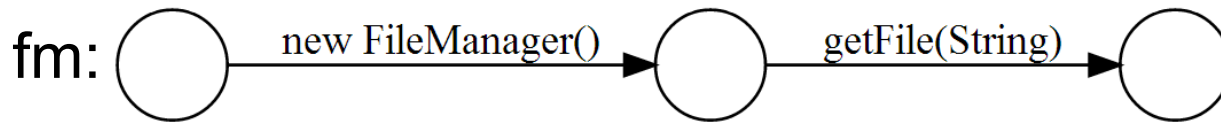
```
File getOpenFile(String filename) {  
    FileManager fm =  
        new FileManager();  
    File f = fm.getFile(filename);  
    f.open();  
    return f;  
}
```

Dealing With Multiple Related Objects

```
class FileManager {  
    FileManager();  
    File getFile(String);  
}
```

```
class File {  
    void open();  
    void close();  
    byte read();  
    void write(byte);  
}
```

```
File getOpenFile(String filename) {  
    FileManager fm =  
        new FileManager();  
    File f = fm.getFile(filename);  
    f.open();  
    return f;  
}
```

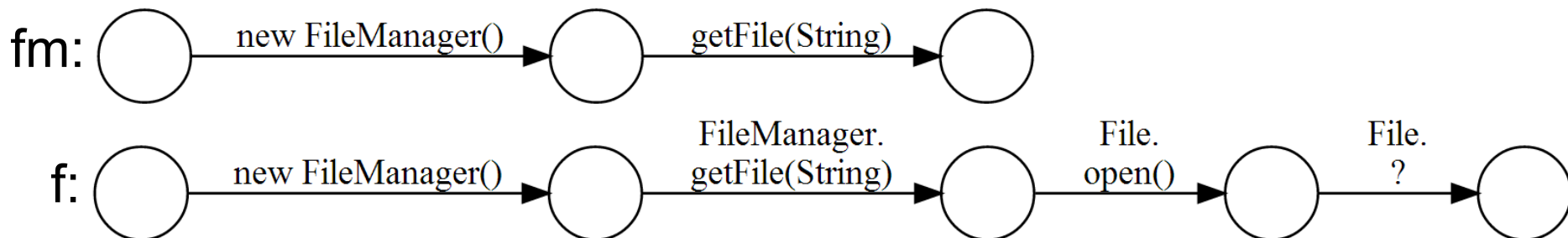


Dealing With Multiple Related Objects

```
class FileManager {  
    FileManager();  
    File getFile(String);  
}
```

```
class File {  
    void open();  
    void close();  
    byte read();  
    void write(byte);  
}
```

```
File getOpenFile(String filename) {  
    FileManager fm =  
        new FileManager();  
    File f = fm.getFile(filename);  
    f.open();  
    return f;  
}
```

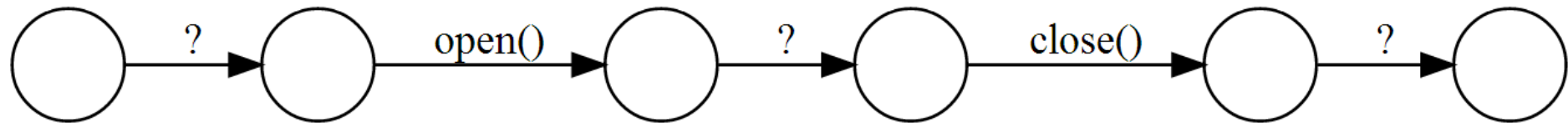


Dealing With Missing Information

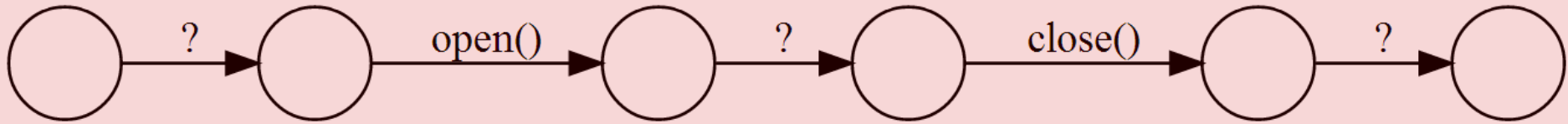
```
void process(File f) {  
    f.open();  
    doSomething(f);  
    f.close();  
}
```

Dealing With Missing Information

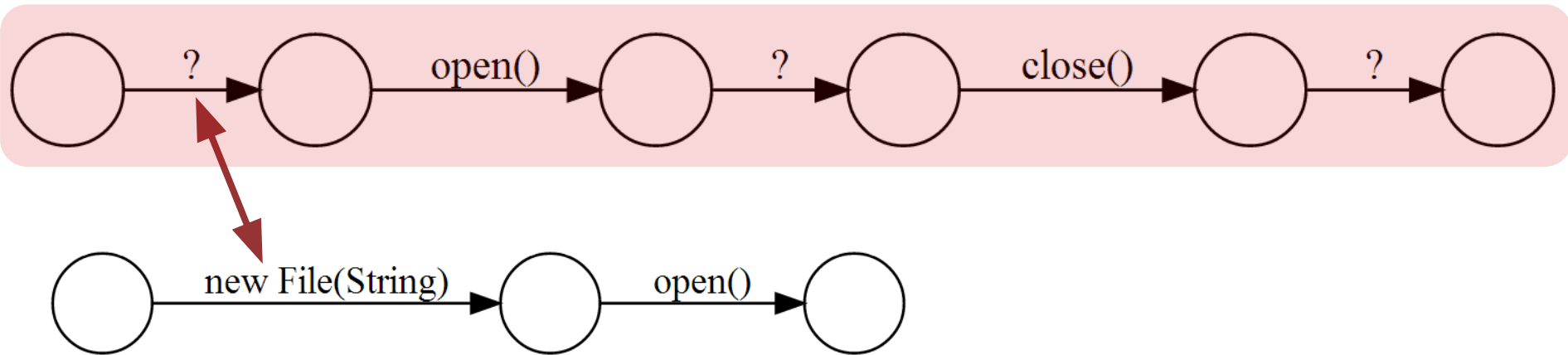
```
void process(File f) {  
  f.open();  
  doSomething(f);  
  f.close();  
}
```



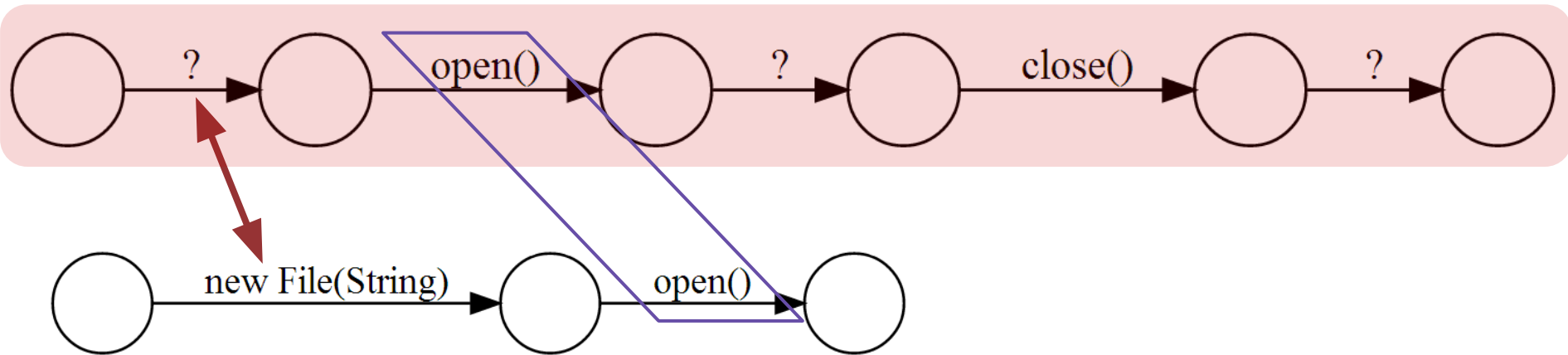
Unknown Elimination



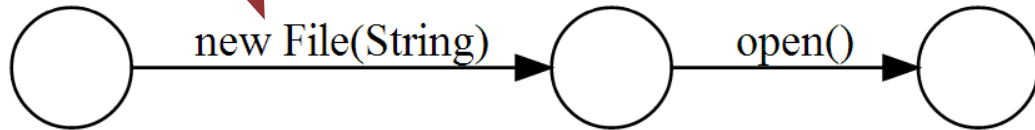
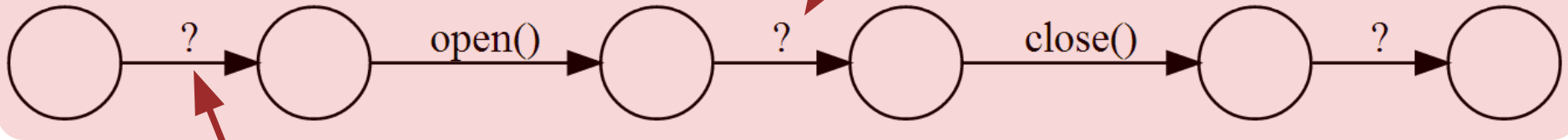
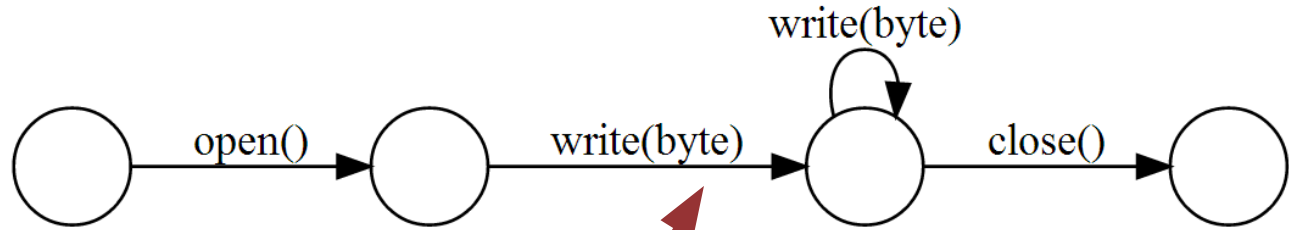
Unknown Elimination



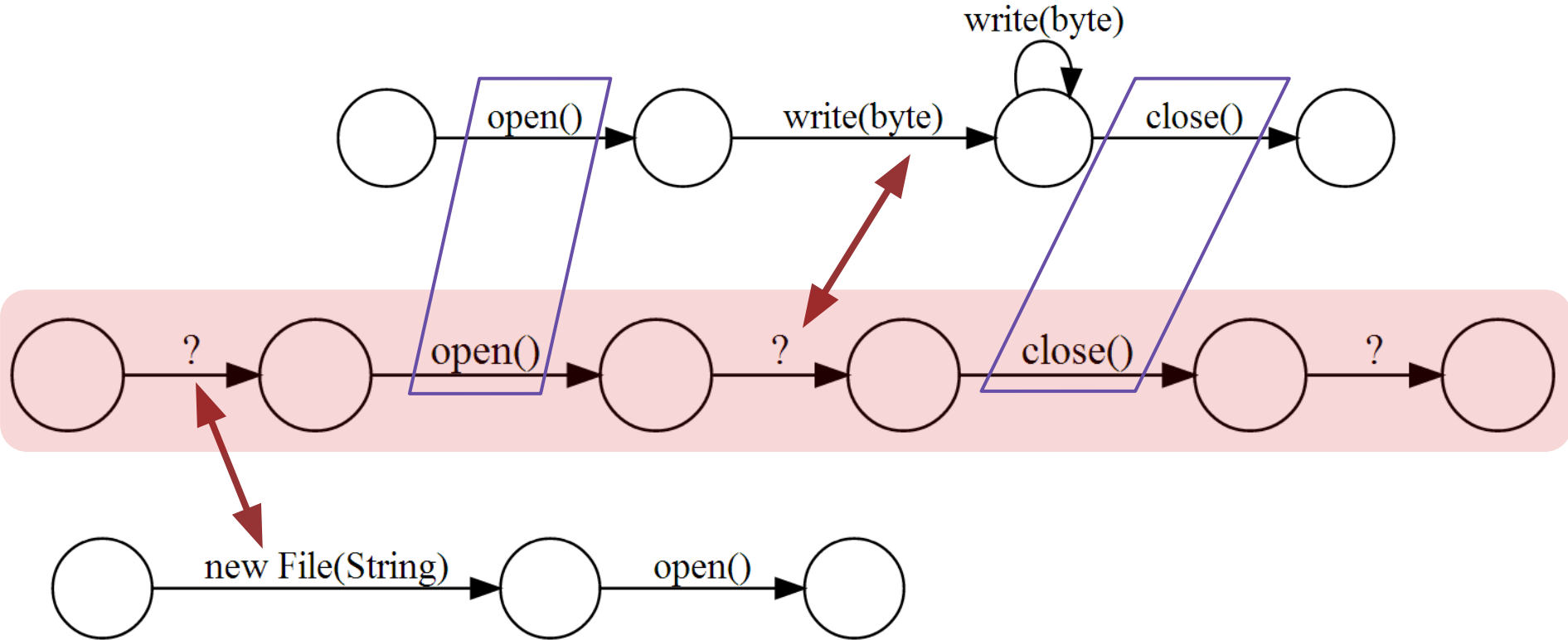
Unknown Elimination



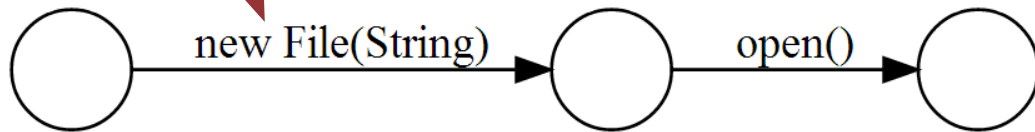
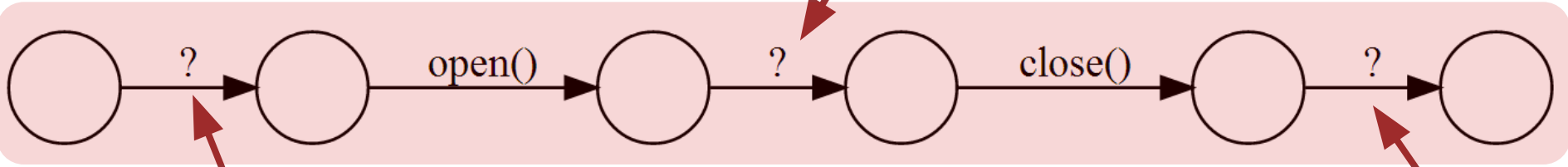
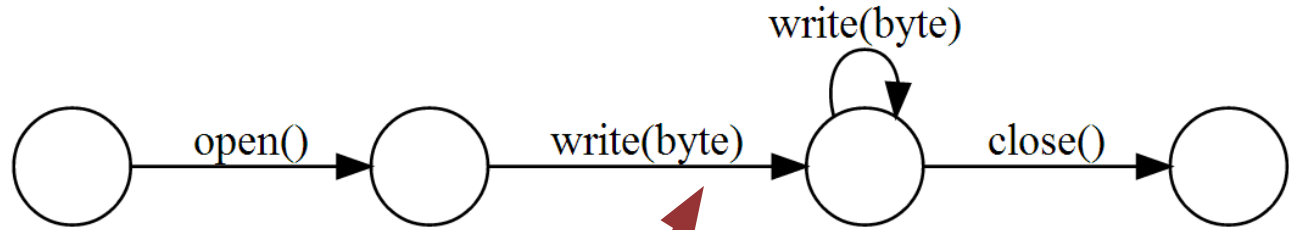
Unknown Elimination



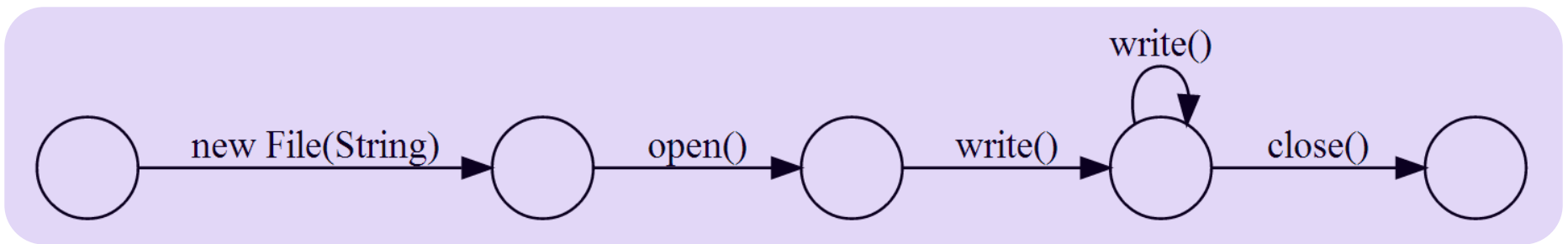
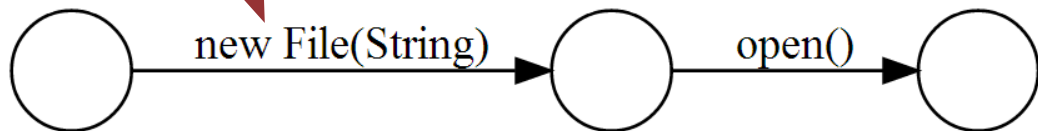
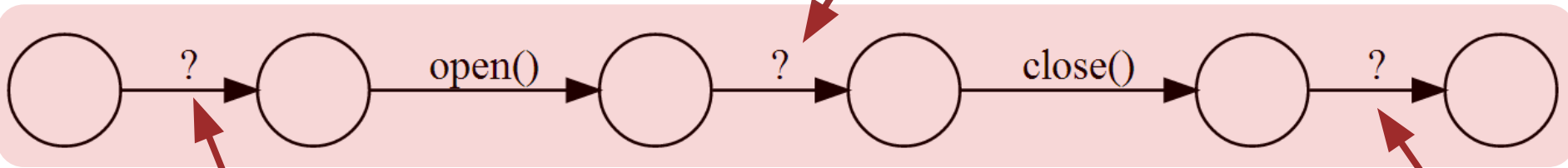
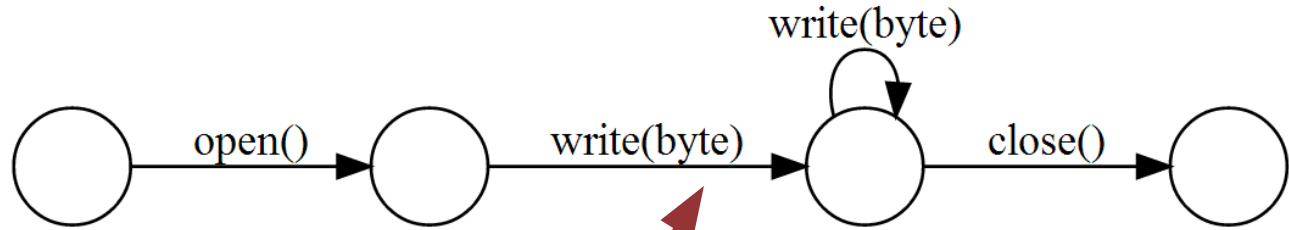
Unknown Elimination



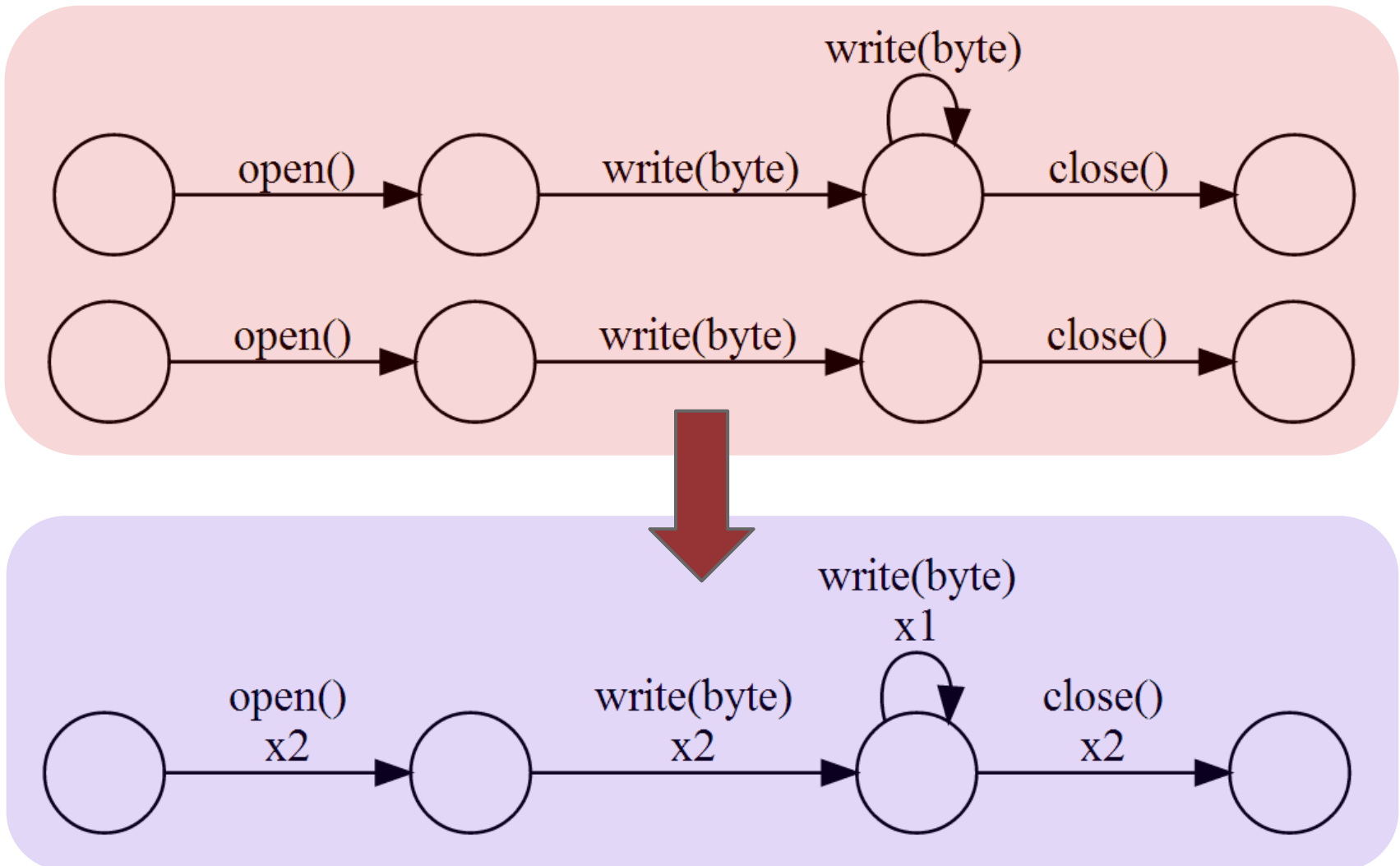
Unknown Elimination



Unknown Elimination



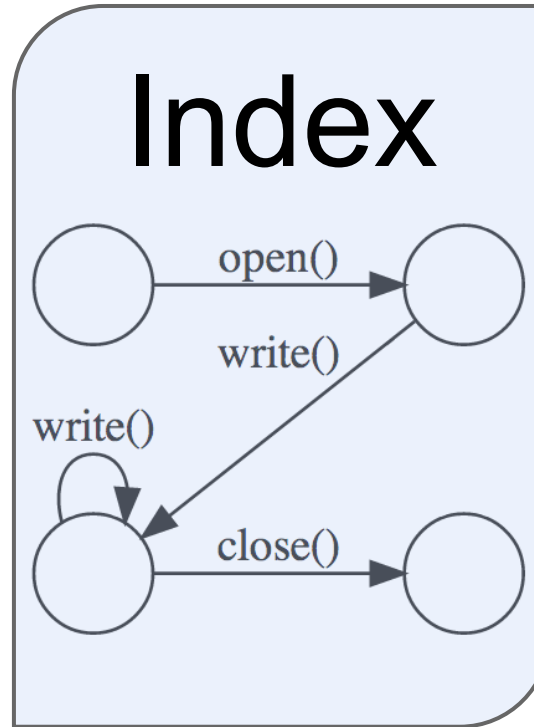
Grouping and Merging



Matching Queries With Examples

Query

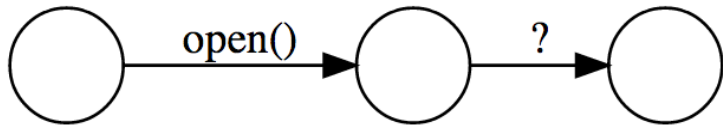
```
f.open();  
f.?
```



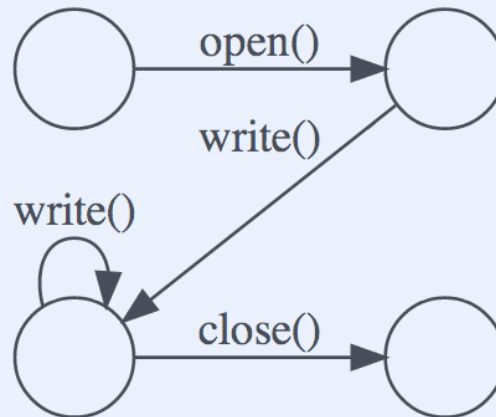
Matching Queries With Examples

Query

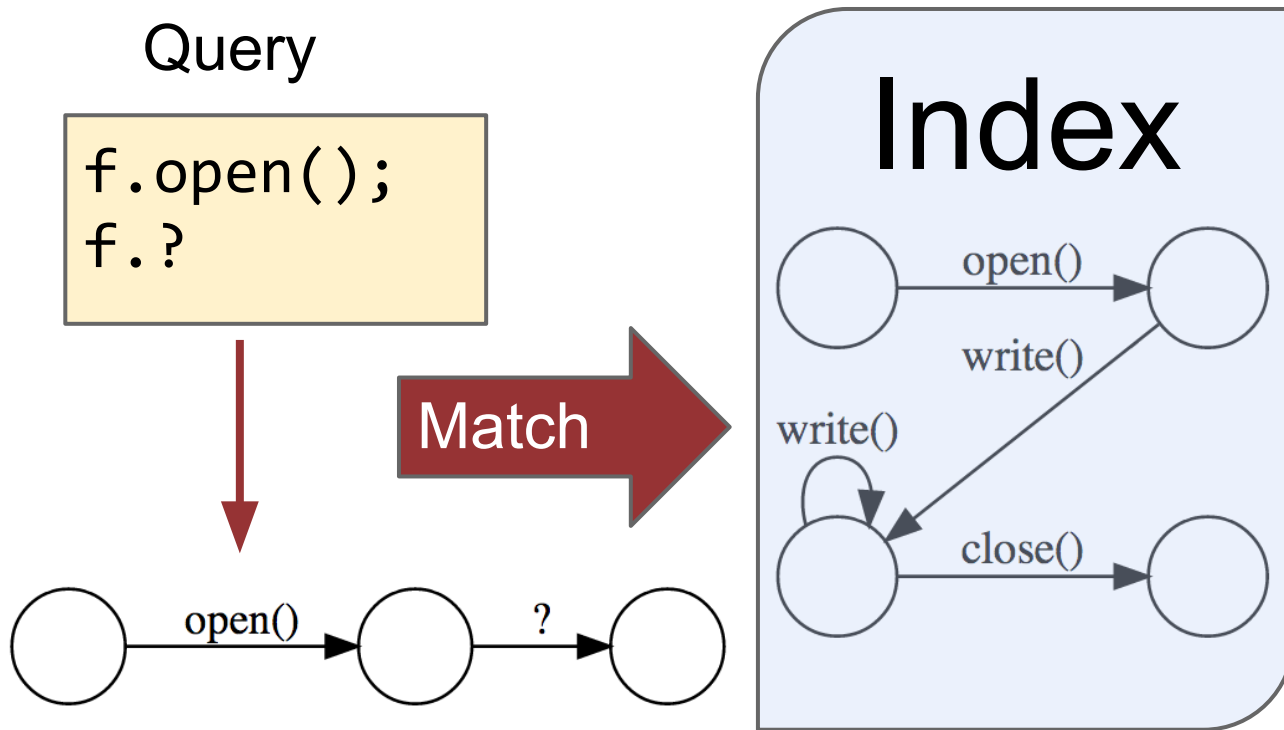
```
f.open();  
f.?
```



Index

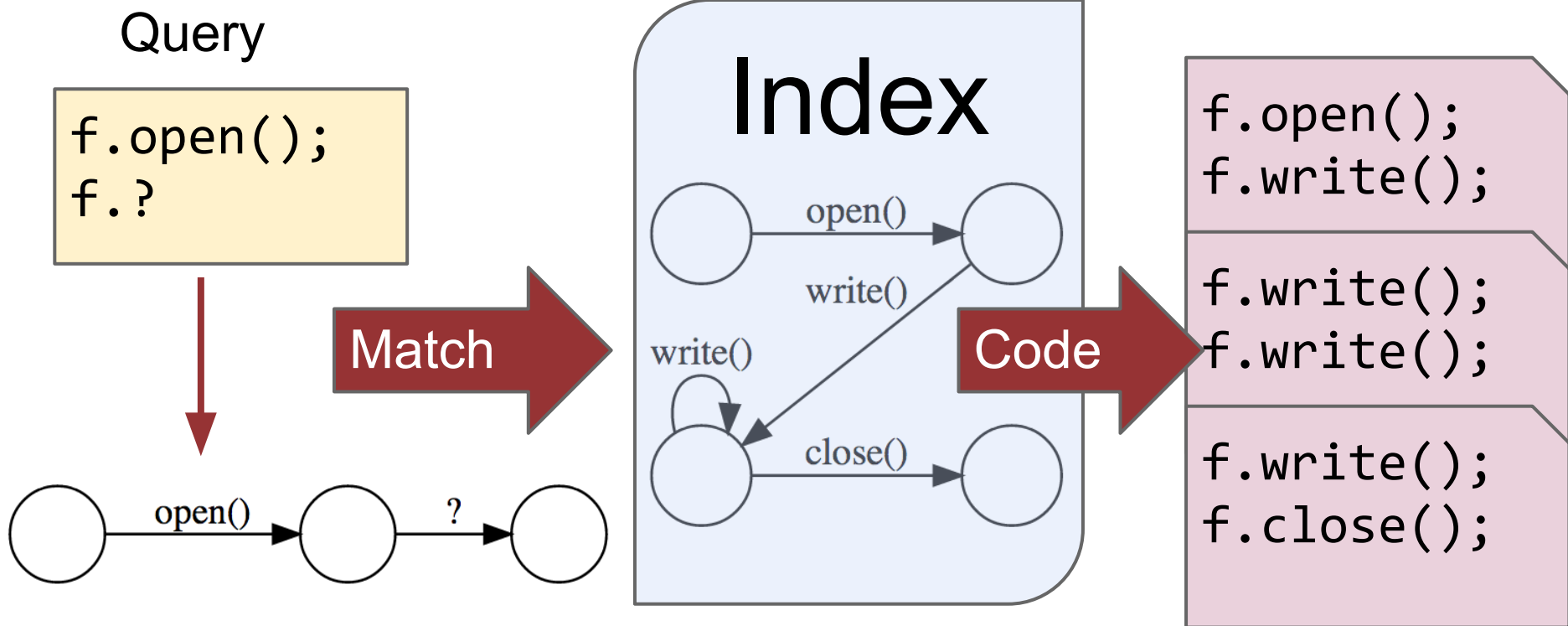


Matching Queries With Examples



Match: A relaxed form of automata inclusion algorithm, taking unknown edges into account

Matching Queries With Examples



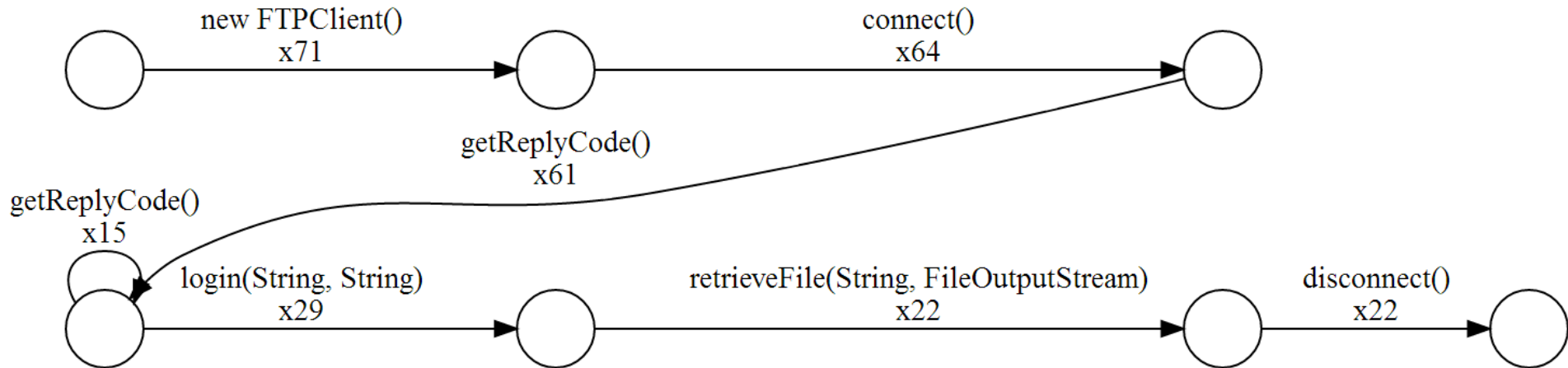
Match: A relaxed form of automata inclusion algorithm, taking unknown edges into account

Evaluation

- Indexed 8 popular APIs, with thousands of examples each
 - Indexing takes hours
 - Querying takes seconds (can be optimized further)
- Consolidation is effective
 - Found long sequences that cannot be found using individual examples
- Ranking is effective
 - Examples covering full tutorial scenario found in the top matches
- Creation context is important in practice
 - Many APIs require sequences over multiple objects

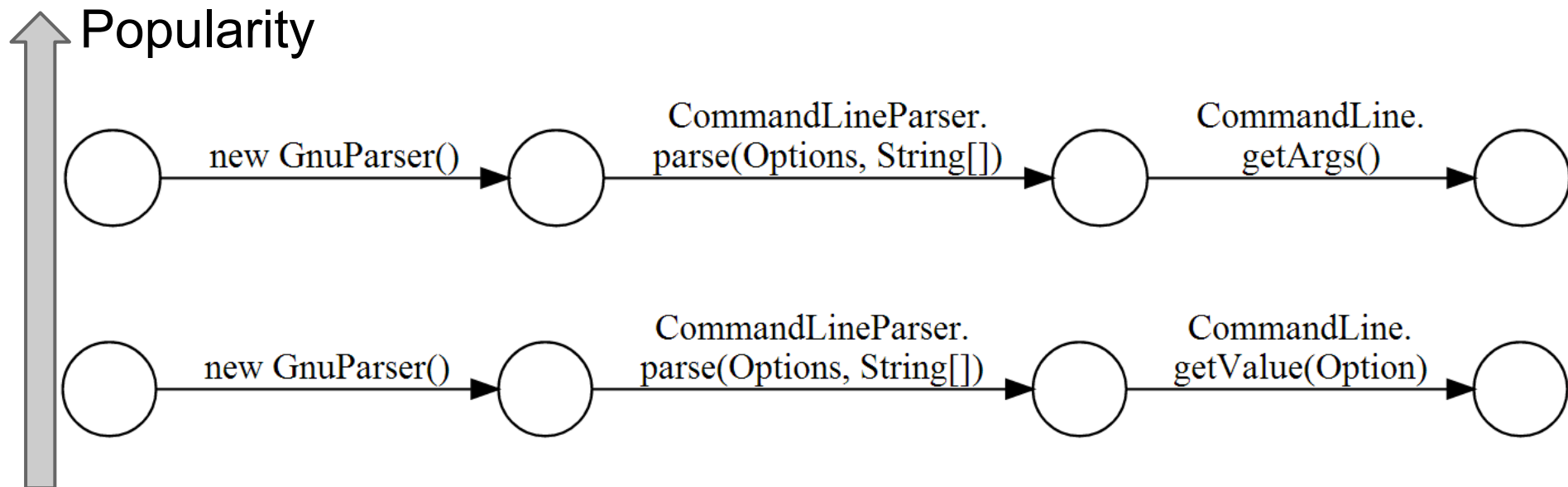
Example Result: Complete Sequence

```
FTPClient c = ?;  
c.login("username", "password");  
c.?  
c.retrieveFile("file path", out);
```



Example Result: Query Over Multiple Types

```
GnuParser p = new GnuParser();  
p.?  
String s = p.?
```



Conclusions

- Leverages the **vast** number of available examples
- Builds **consolidated and distilled** API usage specifications from the many partial examples
- Automatically presents the most **popular** sequences that still **match** the query

It Works and is Open-Source!

This work is implemented in an open-source tool called **PRIME**, available on sourceforge, at:

priming.sf.net

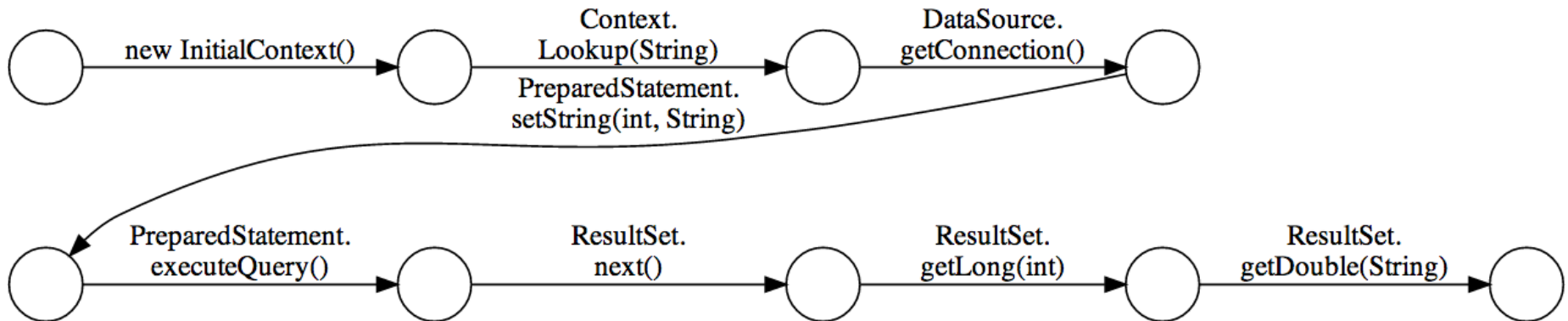
Backup

Closest Related Work

- Zhong's MAPO
 - Only simple sequences (no loops)
 - Can't handle partial code (needs compilation)
 - More and better results for same API
 - Noisy since it doesn't track per-object
- Thummalapenta and Xie's PARSEWeb
 - Can handle partial
 - Only object types
 - AST-based
 - Only how to get one type from another.
- Mandelin's Jungloids
 - Only type safety, no typestate
 - We cover (some of) their examples

Example Result: Long Query

```
InitialContext ic =  
new InitialContext();  
ic.?.  
PreparedStatement ps = ic.?.  
ResultSet rs = ps.executeQuery();  
rs.?.  
double d = rs.getDouble("column");
```



Additional Applications



PRIME
Index

Additional Applications

```
f.close();  
f.open();
```

Verify

**PRIME
Index**

```
Warning:  
open() is rare  
after close()
```

Additional Applications

