Multilingual Code Co-evolution Using Large Language Models

Jiyang Zhang, Pengyu Nie, Junyi Jessy Li, Milos Gligoric

TEXAS
The University of Texas at Austin

FSE 2023

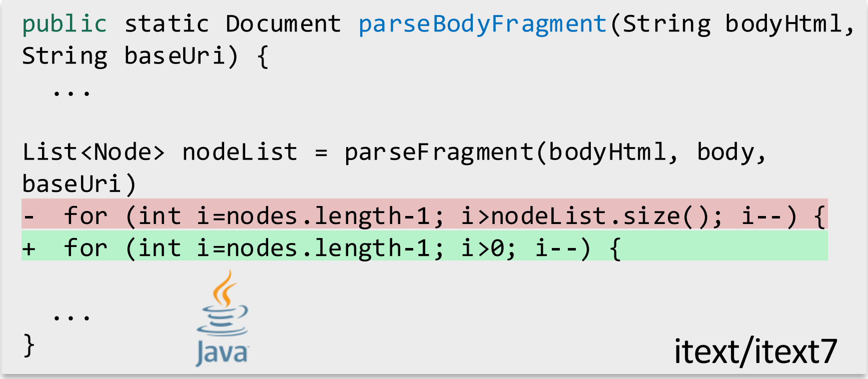
Supported by NSF
CCF-2107291
IIS-2145479
CCF-2217696
CCF-2313027

Software Co-evolution

- One Software could be implemented/provided in multiple programming languages (PLs)
  - MongoDB: PyMongo (Python), Mongoid (Ruby)
- Maintaining software across PLs is challenging
  - Software are constantly evolving and code change in *source* PL should be propagated timely to *target* PLs
  - Building rule-based systems requires manual work and expertise
  - Machine learning code translation models fail to precisely infer the project-specific data types or class names

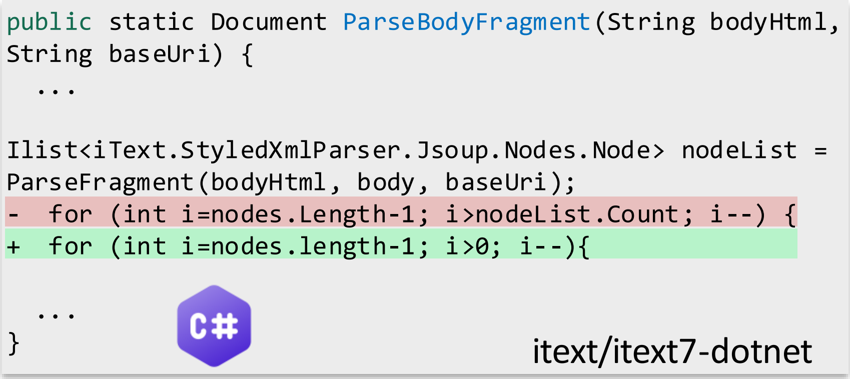
CODEDITOR

- Task: co-evolving software in different PLs
  - Updating code in target PLs based on changes made in source PL
- CODEDITOR: translate edits across PLs and perform the edits



```
public static Document parseBodyFragment(String bodyHtml,
String baseUri) {
  ...

List<Node> nodeList = parseFragment(bodyHtml, body,
baseUri)
-  for (int i=nodes.length-1; i>nodeList.size(); i--) {
+  for (int i=nodes.length-1; i>0; i--) {

  ...
}
```

itext/itext7



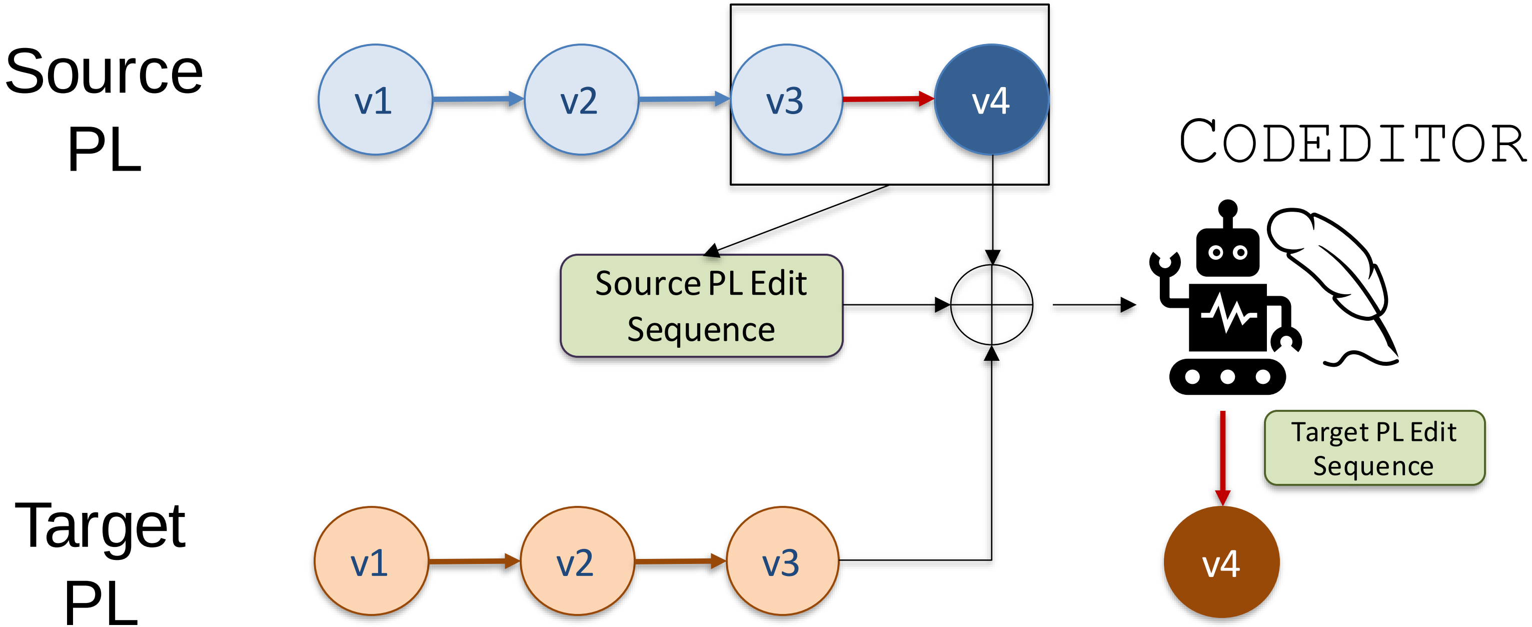
```
public static Document ParseBodyFragment(String bodyHtml,
String baseUri) {
  ...

Ilist<iText.StyledXmlParser.Jsoup.Nodes.Node> nodeList =
ParseFragment(bodyHtml, body, baseUri);
-  for (int i=nodes.Length-1; i>nodeList.Count; i--) {
+  for (int i=nodes.length-1; i>0; i--){

  ...
}
```

itext/itext7-dotnet

Our Contributions

- Propose a novel task of updating code in the target PL based on the changes made in the source PL
- Build a large language model to tackle this task: CODEDITOR
- Create the first dataset with aligned code changes between Java and C#
- Show our model significantly outperforms the existing ML-based code translation models

Overview



Source PL

Target PL

Source PL Edit Sequence

CODEDITOR

Target PL Edit Sequence

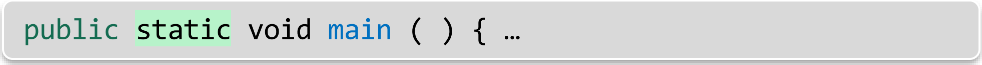
Edit Representation: Concise Edits[1]

- Insert
  - `<Insert>` [span of tokens] `<InsertEnd>`
- Delete
  - `<Delete>` [span of tokens] `<DeleteEnd>`
- Replace
  - `<ReplaceOld>` [span of old tokens] `<ReplaceNew>` [span of new tokens] `<ReplaceEnd>`

[1] Jiyang Zhang, Sheena Panthaplackel, Pengyu Nie, Junyi Jessy Li, and Milos Gligoric. 2022. CoditT5: Pretraining for Source Code and Natural Language Editing.

Edit Representation: Unambiguous Edits

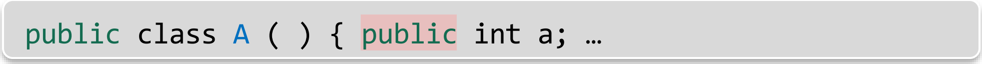
- Do not use Insert



```
public static void main ( ) { …
```

```
<ReplaceOldKeepBefore> public <ReplaceNewKeepBefore> public static
<ReplaceEnd>
```

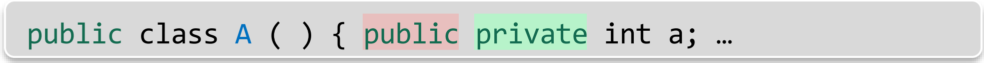
- Discard unclear Delete



```
public class A ( ) { public int a; …
```

```
<ReplaceOldKeepBefore> {public<ReplaceNewKeepBefore> {<ReplaceEnd>
```

- Discard unclear Replace



```
public class A ( ) { public private int a; …
```

```
<ReplaceOldKeepBefore> { public <ReplaceNewKeepBefore> { private
<ReplaceEnd>
```

Concise and Unambiguous Edits

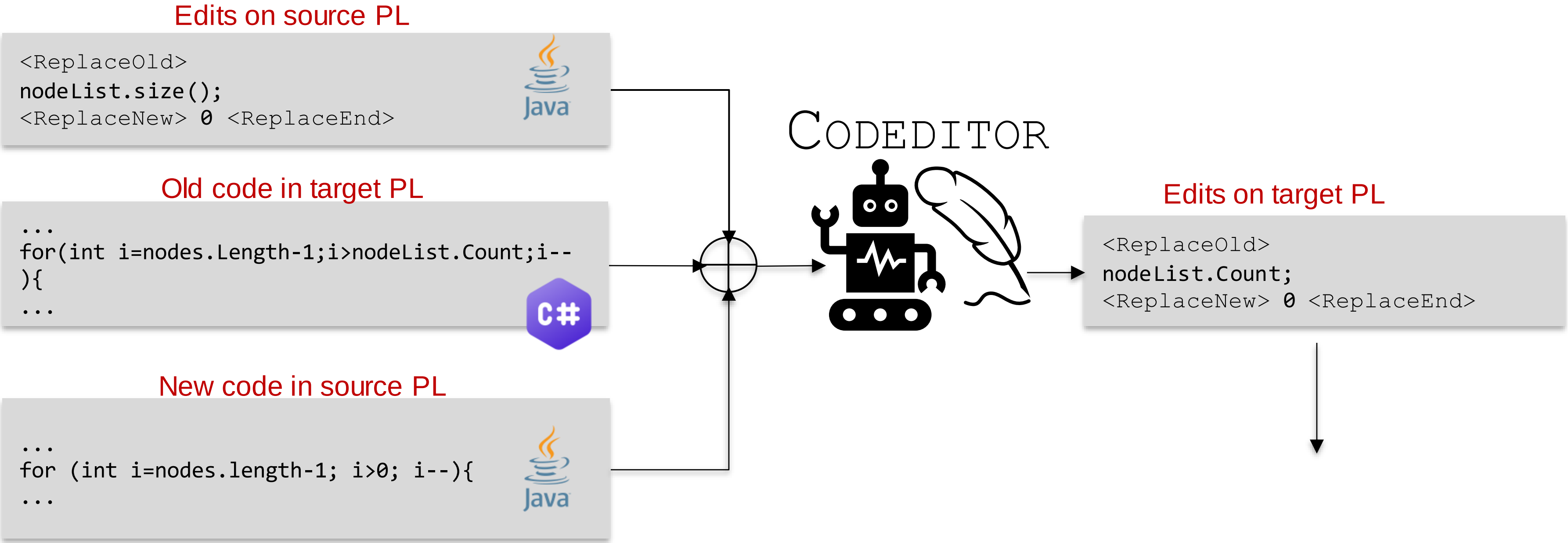
Edit Operation	Concise	Unambiguous

Insert	`<Insert>`	`<ReplaceKeepBefore>`
		`<ReplaceKeepAfter>`

Delete	`<Delete>`	`<Delete>`
		`<ReplaceKeepBefore>`
		`<ReplaceKeepAfter>`

Replace	`<Replace>`	`<Replace>`
		`<ReplaceKeepBefore>`
		`<ReplaceKeepAfter>`

Model Input and Output



Edits on source PL

```
<ReplaceOld>
nodeList.size();
<ReplaceNew> 0 <ReplaceEnd>
```

Old code in target PL

```
...
for(int i=nodes.Length-1;i>nodeList.Count;i--
){
...
```

New code in source PL

```
...
for (int i=nodes.length-1; i>0; i--){
...
```

Edits on target PL

```
<ReplaceOld>
nodeList.Count;
<ReplaceNew> 0 <ReplaceEnd>
```

Dataset

- 8 open-source Java and C# projects [1]
- 6.6 K parallel Java and C# code changes made by developers
  - Code changes in the paired C# method happen no later than 90 days of the Java change
  - Pair code changes by Jaccard Similarity
- Split dataset for training and evaluation based on time
- Task: J2CS and CS2J (not limited to Java and C#)

[1] Shuai Lu, Daya Guo, Shuo Ren, Junjie Huang, Alexey Svyatkovskiy, Ambrosio Blanco, Colin Clement, Dawn Drain, Daxin Jiang, Duyu Tang, et al. 2021. CodeXGLUE: A Machine Learning Benchmark Dataset for Code Understanding and Generation. arXiv preprint arXiv:2102.04664 (2021).

Baselines & Metrics

- Baselines:
  - CopyEdits
  - CodeT5
  - Codex
- Metrics (from 0 to 100):
  - xMatch: pct. of the predictions exactly matches the ground truths
  - SARI: edit actions overlap
  - BLEU, CodeBLEU: token-level overlap

Quantitative Results (J2CS)

	xMatch	SARI	BLEU	CodeBLEU
ML-Translator (CodeT5)	38.02	83.77	87.45	77.15
CopyEdits	38.21	76.92	90.29	91.34
CodeT5	60.41	80.11	90.00	76.63
Codex	48.84	72.80	80.71	59.63
CodEditor	**67.23**	**87.23**	**95.44**	**96.02**

Summary

- Formulate a new task: translation code changes across PLs
- CODEDITOR : a large language model that uses code change history and learns to make edits on other PLs
- Mine open-source repositories to collect more than 6K paired Java and C# changes
- Evaluate on this newly created dataset

Jiyang Zhang <jiyang.zhang@utexas.edu>