Fuzzy Edit Sequences in Genetic Improvement

Aymeric Blot

University College London

GI@ICSE’19 — 28 May 2019
Fuzzy Edit Sequences (in GI)

“Edit Sequences” (“Patches”)

▶ Mutant source code representation

“Semantics”

▶ Add significance to individual edits/mutations
▶ Guide the GI search process to preserve meaning

“Fuzzy”

✗ Fuzzing

▶ Fuzz testing
▶ Automatic random test input generation

✓ Fuzzy Matching

▶ Fuzzy string searching
▶ Approximate string matching
Edits ...
Edit Sequences ...

... a: foo();
b: bar();
c: baz();
...

... a: // empty
b: bar();
c: baz();
...

d(a)

... a: // empty
b: baz();
b: bar();
c: baz();
...

i(b,c)

d(a)i(b,c)
Edit Sequences are Great!

**Very flexible**
- Easy to generate
- Easy to mutate
- Easy to crossover

**Sparse**
- Not source code
- Can be broken down
- *Close* to human understanding

**But**
- Focus on practical modification
- Mechanical
Edit Sequences are Great?

Very flexible
- Easy to generate
- Easy to mutate
- Easy to crossover

Sparse
- Not source code
- Can be broken down
- *Close* to human understanding

But
- Focus on practical modification
- Mechanical
Edit Sequences in Practice

Edits as triplets \((op, t, i)\)

- **op**: mutation
- **t**: target location
- **i**: ingredient location (optional)

List of triplets

```
"i" 1 2 "d" 2 ∅ "r" 3 40 ...
```

Triplet of lists

```
"i" "d" "r" ...
1 2 3 ...
2 ∅ 40 ...
```
Edit Context

\[
\begin{align*}
\text{a: } & x++; \\
\text{b: } & f(x); \\
\text{z: } & x = 7;
\end{align*}
\]

\[
\text{∅} \quad \text{swp}(a, b) \\
\text{r}(a, z) \quad \text{r}(b, z) \quad \text{swp}(a, b) \quad \text{r}(b, z)
\]

\[
\begin{align*}
\text{a: } & f(x); \\
\text{b: } & x++; \\
\text{z: } & x = 7;
\end{align*}
\]

\[
\begin{align*}
\text{a: } & f(x); \\
\text{b: } & x = 7; \\
\text{z: } & x = 7;
\end{align*}
\]

\[
\begin{align*}
\text{a: } & x++; \\
\text{b: } & x = 7; \\
\text{z: } & x = 7;
\end{align*}
\]

\[
\begin{align*}
\text{a: } & x = 7; \\
\text{b: } & f(x); \\
\text{z: } & x = 7;
\end{align*}
\]
Location vs Content

Motivation
Could we use content instead of/in addition to location?

- Instead of? No. Content is not unique.
- In addition to? Yes?

Examples
- Location:
  - “delete ‘line 17’”
  - “replace ‘line 12’ by ‘line 42’”
- Content:
  - “delete ‘i++;’”
  - “replace ‘x = 0;’ by ‘x = y;’”
- Both:
  - “delete ‘i++;’ at ‘line 17’”
  - “replace ‘x = 0;’ at ‘line 12’ by ‘x = y;’ at ‘line 42’”
Edit Semantics

Problems

- Edits are only tied to location
- Context depends on previous edits
- Context changes are ignored

Definition Proposition

Edit semantic:
Meaning in terms of content + location

\[ r(10,40) \text{ “replace ‘x++’ at ‘line 10’ with ‘x = 7’ at ‘line 40’”} \]
\[ \neq r(10,40) \text{ “replace ‘f(x)’ at ‘line 10’ with ‘x = 7’ at ‘line 40’”} \]
Edit Context Changes

Patches

- patch 1: `edit1edit2edit3`
- patch 2: `edit4edit5`

During mutation

- Deletion (middle): patch 1 → `edit2edit3`
- Insertion (middle): patch 1 → `edit1edit2edit6edit3`
- Permutation: patch 1 → `edit1edit3edit2`
- Any target but the last edit

During crossover

- Concatenation: `edit1edit2edit3edit4edit5`
- Any use of external data
# Edit Sequences with Content

Edits as tuples \((\text{op}, \text{tl}, \text{il}, \text{tc}, \text{ic})\)

- **op**: mutation
- **tl**, **tc**: target **location** and **content**
- **il**, **ic**: ingredient **location** and **content** (optional)

## List of tuples

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
</table>
| "i" | 1 | 2 | "foo();" | "bar();"
| "d" | 2 | ∅ | "bar();" | ∅
| "r" | 3 | 40 | "BUG();" | "FIX()"

...
Conflict Management

At creation
(op, a, b) becomes (op, a, b, α, β)

\[ \alpha = f(a), \beta = f(b), f: \text{lookup function} \]

At application
Compare again \( \alpha \) to \( f(a) \), \( \beta \) to \( f(b) \)

⚠️ Conflict! e.g., \( \alpha \neq f(a) \) or \( f(a) \) fails

- Location match: \((a, \alpha) \rightarrow (a, \alpha')\)
- Content match: \((a, \alpha) \rightarrow (a', \alpha)\)
- Weak match: \((a, \alpha) \rightarrow (a', \alpha')\)
Conflict Resolution Strategies

Example: Concatenation crossover

- edit1edit2 + edit3edit4

⚠ Conflict between edits 1 and 3! (e.g., identical)

Resolution strategies

- **Ignore**: edit1edit2edit3edit4
- **Discard**: edit1edit2edit4
- **Repair**: edit1edit2edit3’edit4

Exploration vs exploitation

- Strategies are not mutually exclusives
- Multiple repairs can be found (and ranked)
Fuzzy Matching

For content
- String edit distance
- Tree edit distance
- Tokenization? (lexical analysis)

For location
- Euclidean distance
- Path length
- Height to target ancestor
- Specific nodes in path
Possible Further Steps?

Content repair?
▶ \((\text{op, } a, b, \alpha, \beta) \rightarrow (\text{op, } a, b, \alpha', \beta) \rightarrow (\text{op, } a, b, \alpha', \beta')\)
   with \(\beta' = \text{repair}(\beta, \alpha, \alpha')\)

Approximate location?
▶ (“replace every between”, \(a \rightarrow a', b, \alpha, \beta\))
▶ (“replace every inside”, \(a, b, \alpha, \beta\))

Partial edits?
▶ (“replace”, \(a, b, \text{“□ < □”}, \text{“□ <= □”}\))
▶ (“delete”, □, ∅, “assert(□)”, ∅)
Final Words

Edit sequences are great

- Very flexible
- Sparse
- Too mechanical?

Content based semantic

- More guidance
- More diversity
- Yet to be investigated
Selected References


