SynthEdit: Format transformations by example using edit operations

Alex Bogatu, Norman W. Paton and Alvaro A.A. Fernandes, Nikolaos Konstantinou

School of Computer Science, University of Manchester

1. Introduction

- **Definition:** *Format transformations* is the sub-task of data wrangling that carries out changes to the representation of textual information, with a view to reducing inconsistencies.
- **Transformation scenario:** 1900s NY state governors:

Source	Target
Hugh Leo Carey (74-82)	Hugh L. Carey (1974-1982)
Gov. Jay Henry Lehman (33-42)	Jay H. Lehman (1933-1942)
Mario Matthew Cuomo (83-95)	
Gov. Martin Henry Glynn (13-15)	

Research objective
To develop a method for format transformations starting from given *input/output examples* that is (i) effective in transforming new strings (similar to the example input); (ii) scalable with the number of examples; and (iii) fully automated.

2. Related Work

Programming-by-Example synthesis algorithms

- FlashFill[1], BlinkFill[3]
- Spreadsheet-oriented: active user involvement
- Synthesis time exponential in the number of examples

Pattern enforcement and transformation tools

- Wrangler[2]
- Manual authoring of transformation scripts
- Expert-level knowledge about the language

3. Method

1. Extract regex-based tokens from each example instance:

Regex primitives
Number(**N**); Upper/Lower case(**U**/**L**); Alphabet(**A**); Alphanumeric(**Q**); Punctuation(**P**);

Source/Target: Hugh Leo Carey (74-82) | Hugh L. Carey (1974-1982)
Token-type repr.: **A A A P N P N P** | **A U P A P N P N P**

2. Generate edit operations converting source to target:

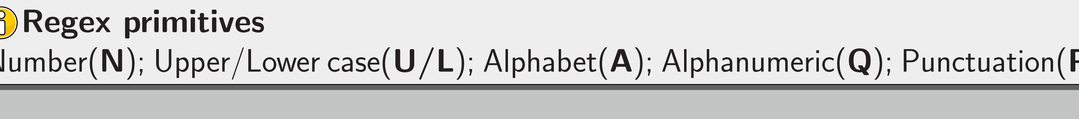
Edit operations
Insert(**INS**); Delete(**DEL**); Substitute(**SUB**);

SUB(A^s_0, A^t_0); **SUB**(A^s_1, U^t_0); **INS**(P^t_0); **SUB**(A^s_2, A^t_1); **SUB**(P^s_0, P^t_1);
SUB(N^s_0, N^t_0); **SUB**(P^s_1, P^t_2); **SUB**(N^s_1, N^t_1); **SUB**(P^s_2, P^t_3);

3. Express each target token as a string expression applied on some source token:

String expressions
Copy(**Copy**); Constant(**Const**); Substring(**Substr**); Concatenate(**Concat**);

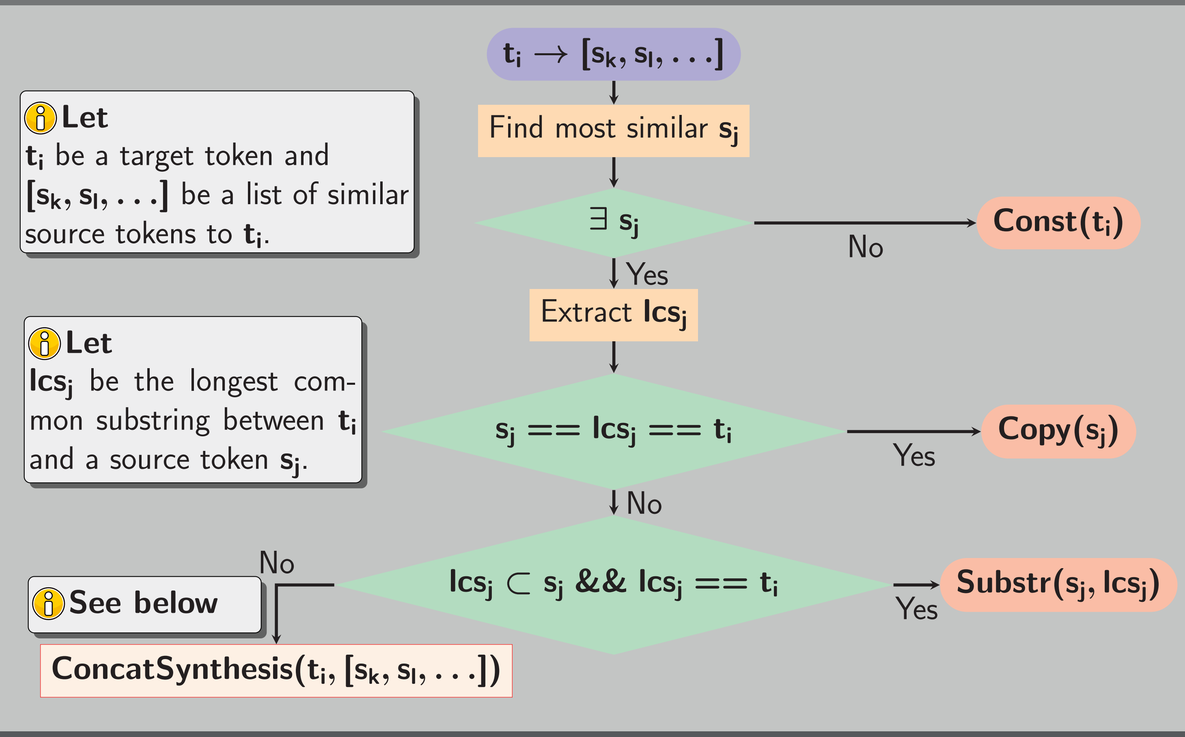
- For each target token, find all source tokens that are either a substring or a superstring of the target token (similar tokens).



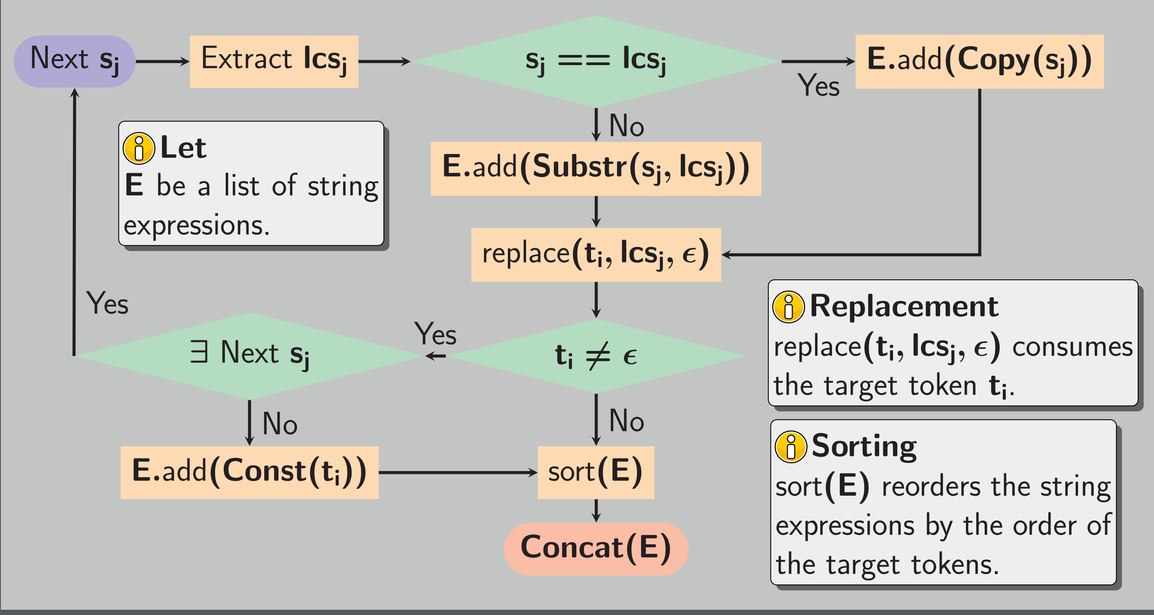
- For each pair of $<$ **target token, [list of similar tokens]** $>$, synthesize a new string expression.
- Final output example:

SUB(A^s_0, **Copy**(A^s_0)); **SUB**(A^s_1, **Substr**(A^s_1, 0, 1));
INS(**Const**(".")); **SUB**(A^s_2, **Copy**(A^s_2)); **SUB**(P^s_0, **Copy**(P^s_0));
SUB(N^s_0, **Concat**(**Const**("19"), **Copy**(N^s_0))); **SUB**(P^s_1, **Copy**(P^s_1));
SUB(N^s_1, **Concat**(**Const**("19"), **Copy**(N^s_1))); **SUB**(P^s_2, **Copy**(P^s_2));

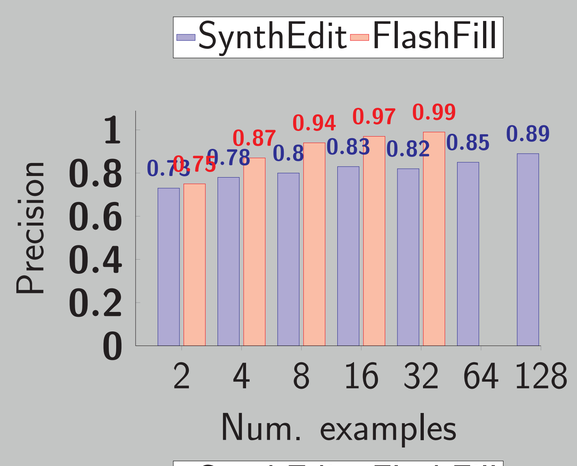
4. String expression synthesis

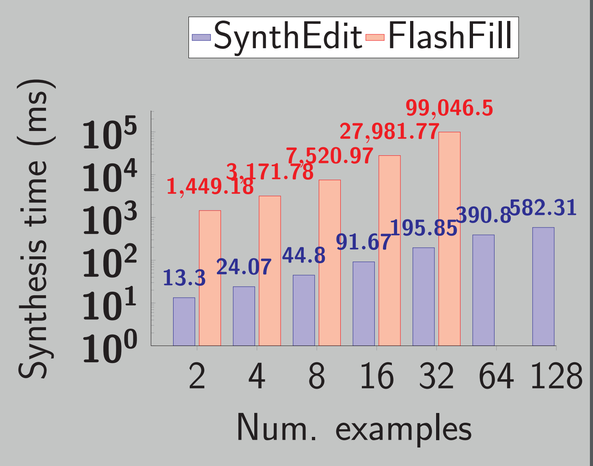


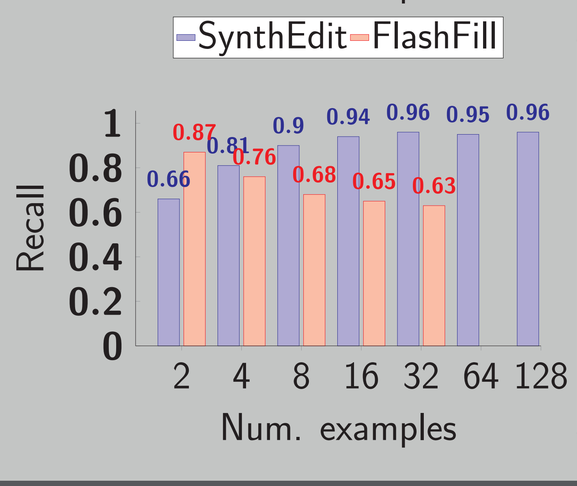
5. ConcatSynthesis



6. Evaluation: SynthEdit vs. FlashFill







SynthEdit

- performs a strict mapping between token-type representations and transformations $\Rightarrow$ it requires more examples and learns less general transformations.
- uses a simpler transformation language and it does not aim to exhaust the search space of transformations for each example instance $\Rightarrow$ it is more efficient.

7. Conclusions

- We propose a transformation language that uses *regex primitives*, *edit operations*, and *string expressions* to express format transformations.
- We propose a synthesis algorithm that, starting from a given set of *input/output examples*, automatically learns one or more transformations expressed using the mentioned language and consistent with the examples.
- Our proposed method is more efficient than the closest antagonist, while achieving better recall at the cost of slightly reduced precision.

References

[1] S. Gulwani. Automating string processing in spreadsheets using input-output examples. In *POPL*, pages 317–330, 2011.

[2] S. Kandel, A. Paepcke, J. M. Hellerstein, and J. Heer. Wrangler: interactive visual specification of data transformation scripts. In *CHI*, pages 3363–3372, 2011.

[3] R. Singh. Blinkfill: Semi-supervised programming by example for syntactic string transformations. *PVLDB*, 9(10):816–827, 2016.

This work is supported by the VADA Grant of the UK EPSRC.

alex.bogatu@manchester.ac.uk