# FlushFill

Transparent Program Synthesis in Excel

Justin A. Middleton
North Carolina State University, Raleigh, USA
jamiddl2@ncsu.edu

April 30, 2016

**Abstract**

Program synthesis in Excel grants spreadsheet users a powerful way of transforming input without much work. However, users have no insight into what programs are being made and, therefore, they can't apply automatically generated solutions to situations or learn programs on their own. My work here is to meet the program synthesizing capabilities of current spreadsheets but make the resulting program accessible to the user, both in visibility and in comprehensibility. In the process, I evaluate a number of approaches to the this problem of transparent synthesis, ranging from the algorithms of FlashFill itself to principles of planning.

## 1 Introduction

Millions of programmers rely on spreadsheet programs, like Microsoft Excel, to make sense of their data, and, as part of some of my other ongoing research outside of class, Im looking for ways to make their lives easier. Already shipped with Excel is a feature called Flash Fill  given a set of inputs and a few sample outputs, it tries to create a program which captures the pattern of transformation which maps the set of inputs to outputs and then applies it to the rest of the incomplete values. However, it does this opaquely; users dont have insight into the program created, which impairs deep comprehension and reusability of the program. To be clearer, an example from the paper of FlashFill [1] asks about creating a program which captures the following two columns:

I took these words straight from my 582 pitch – no fears, will change.

| Input | Output |
|---|---|
| John DOE 3 Data [TS]865-000-0000 - - 453442-00 06-23-2009 | 865-000-0000 |
| A FF MARILYN 30S 865-000-0030 4535871-00 07-07-2009 | 865-000-0030 |
| A GEDA-MARY 100MG 865-001-0020 - - 5941-00 06-23-2009 | 865-001-0020 |

What needs to happen is for a program to discover the correct patterns and manipulations to take the string in the input column and change it to that in the output. In this case, there are many acceptable ways to do this: finding three sets of hyphen-separated numbers with digits in quantities of 3-3-4; finding some hyphen-connected numbers beginning with 865; etc.

# 2 Approach

## 2.1 Phase 1

The fact that the first prototype of FlashFill is about 5000 lines of code [1] was a good cue to me that a full repeat of the code would go out of my time. Even so, the algorithm as in that paper offered a great starting point for the program-synthesizing novice that I am. To start, then, I chose a plan with these few guidelines:

1. **A console application in C#.** Visual Studio has templates for Excel add-ins, which would be great to use if not for the extra time it would take to learn. Instead, I chose to pipe into the console some CSV files mimicking spreadsheet columns. If the algorithm worked out, then, hopefully, a move to an Excel add-in (which accepts C#) would not be too far out.

2. **None of the Usability extensions.** Let's be clear: I aim to prove merely that it's possible, not that I can snap my fingers and make a robust interface for semester project.

3. **Use the examples from paper for tests.** They're already written! It's a good way to save me some time.

4. **Don't worry about planning right now.** Proving I can do it one way should be enough for the first phase. Once I can do that, then I can move on.

#### 2.1.1 Major Wrinkles

P1 **A Lack of Loops** The FlashFill algorithm as described in [1] defines a use for looping over strings. Excel has no function to do the same task.

S1.1 *Fudge it with natural language* Rather than trying to do looping a dynamic number of times, just explain what a function needs to do with words like "Loop for as many times it has a substring" or something.

S1.2 *Go VBA instead.* Try to accomplish the same task using VBA lingo over pure Excel functions.

P2 **A Rejection of Regular Expressions** Don't think I can use these in standard Excel functions either.

S2.1 *Go VBA instead.* Since I think can. Plus, you can still use regular Excel functions in VBA[1].

P3 **Multi-Column input ambiguity.** It's not clear how the algorithm handles input when it spans columns.

S3.1 *For now, handle only one column input.*

S3.2 *Combine input columns into one string.*

## References

[1] Sumit Gulwani. Automating string processing in spreadsheets using input-output examples. In *ACM SIGPLAN Notices*, volume 46, pages 317–330. ACM, 2011.

[1] https://msdn.microsoft.com/en-us/library/office/hh211481%28v=office.14%29.aspx