Using Objects of Measurement to Detect Spreadsheet Errors

Michael J. Coblenz, Andrew J. Ko, and Brad A. Myers
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213 USA
mcoblenz@andrew.cmu.edu, ajko@cmu.edu, bam@cs.cmu.edu

Abstract

There are many common spreadsheet errors that traditional spreadsheet systems do not help users find. This paper presents a statically-typed spreadsheet language that adds additional information about the objects that the spreadsheet values represent. By annotating values with both units and labels, users denote both the system of measurement in which the values are expressed as well as the properties of the objects to which the values refer. This information is used during computation to detect some invalid computations and allow users to identify properties of the resulting values.

1. Introduction

In this paper, we introduce a new spreadsheet system called SLATE ("A Spreadsheet Language for Accentuating Type Errors"). The full SLATE system is described in [1]. By separating the unit from the object of measurement, and defining new semantics for spreadsheets so that both the unit and the object of measurement are taken into consideration, SLATE can help users find some kinds of errors in spreadsheets, which is important because 20% to 40% of spreadsheets contain errors [2].

By redefining the semantics of traditional spreadsheet operations, such as addition and multiplication, the system can generate additional information about results that reveals formula errors. For example, a user might mistakenly multiply pounds of apples by the price per pound of oranges. A traditional spreadsheet showing only values, as in Figure 1, would hide this error by displaying only the result in dollars. SLATE reveals the problem by showing that the result has properties of both apples and oranges, as shown in Figure 2.

Like SLATE, other systems have had the goal of improving on the spreadsheet paradigm. Apples and Oranges [3] is the most closely related work. In it,

Erwig and Burnett developed a unit system whereby the system inferred units for cells using a header cell inference algorithm [5]. The system flags cells if its inference algorithm suggests an error. Ahmad et al. developed a type system that uses a notion of header cells (similar to [3]) to define types, using is-a and has-a relationships, and requires users to specify the header cells [4]. However, it exposes complex unit information to users, and unlike SLATE, the information available to its unit system is limited to headers in spreadsheets.

Apples (per lb.)	Oranges (per lb.)
$0.45	$0.50

Fruit	Fruit Sold (lbs.)	Revenue
Apples	312	$140.40
Oranges	399	$179.55

Figure 1. An incorrect calculation in a spreadsheet.

Figure 2. The spreadsheet from Figure 1, using SLATE. The revenue for oranges is incorrect. SLATE computed the contents of the Revenue cells, including the labels.

3. An Example: Orchard Records

In Figure 1, a user attempted to calculate revenues for two types of fruit: apples and oranges. Instead of multiplying each weight of fruit by the corresponding cost, the user accidentally multiplied each weight by the cost per pound of apples. Conventional spreadsheets only display the result of the calculation, so the source of the error is not visible. Spreadsheets that consider only units would not reveal this error either, since both values under consideration have the same units: $ / lb. The mistake has been hidden, only to be found by a careful inspection of the formulas.

SLATE reveals errors by displaying additional information in the cells: in addition to displaying a unit, it displays a label, which is a list of attributes pertaining to the value in the cell. When displaying labels, SLATE encloses them in parentheses to visually separate them from the unit.

In Figure 2, the same calculation from Figure 1 is performed in SLATE. In the "Revenue" column, the amounts are treated as measurements of fruit. Cell B5 measures the cost of apples. Cell B6, however, appears to measure the cost of fruit that is simultaneously apples and oranges. This is obviously wrong; the user expected the cell to have only the attributes of oranges, since the calculation has nothing to do with apples. By computing and displaying these labels based on the labels the user entered for the original information, the system can help users detect otherwise hidden errors.

4. Language

In SLATE, every cell value has two corresponding attributes: a unit and a label. Units, such as meters or kilograms, capture information about the scale at which the measurement was taken and the dimensions of the measurement. SLATE adds *labels*, which define characteristics of the objects of measurement. A cell referring to 25 pounds of apples might read "25 lbs. (apples)". In this example, the label is (apples). A cell referring to apples picked in September might have the label "(apples, September)" because the value in the cell has characteristics of both apples and September.

SLATE must be customizable to work with many different kinds of objects. For example, a construction company expects a spreadsheet to understand plywood and concrete; a farm expects it to understand fruit and vegetables. Thus, each spreadsheet refers to a unit context and a label context. The unit context defines the base units: the primitive units that, when multiplied, form other units.

The label context forms the core of our research. The structure of the label context reflects the observation that many real-world concepts and objects are hierarchical. Operations are defined on the labels so that generalizations up the hierarchy take place where appropriate. Therefore, the label context is a tree, where each node is a particular concept. There is an edge from node n_1 to node n_2 if objects of type n_2 have all of the properties of objects of type n_1. For example, in the small label context in Figure 3, there is an edge from "Apples" to "Red Delicious" because red delicious are a kind of apple. The `Something` node represents the most general type of object; it is named as such to warn the user of a potentially dangerous generalization.

Expressions (other than those that have errors in evaluation) may be added or subtracted if and only if they have equivalent *units*. This restriction maintains the standard interpretation of units. Because units express the measurement system, permitting these operations for values of different units would result in nonsense. To determine the *label* of the result of an addition or subtraction operation, SLATE derives the least general label that includes all of the properties in both of the operands.

For example, using the context in Figure 3:

```
5 lbs. (apples) + 2 lbs. (oranges) = 7 lbs. (fruit)
```

because apples and oranges are both fruit. Similarly, in a context where September has been defined:

```
5 lbs. (apples, September) + 2 lbs. (oranges,
    September) = 7 lbs (fruit, September).
```

Notice how apples and oranges are combined into fruit, but because September is in both labels, it is copied into the final label.

The goal of this choice of definition of addition and subtraction is for the labels to succinctly express the properties of the resulting object. Addition can be thought of as a union of sets of objects. The label of the result expresses the properties that are guaranteed of an arbitrary element of the result. The only properties that can be guaranteed are those that are shared among all the objects in the set.

When two non-error values are multiplied or divided, the result's unit is the product or quotient, respectively, of the units of the operands. Errors
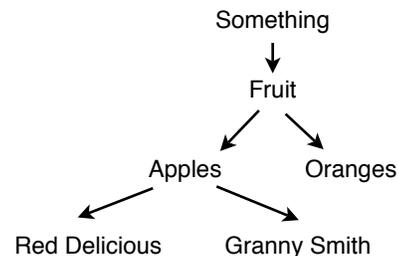

Figure 3. A small label context.

propagate: if an operand has an error in evaluation, so does the result. Multiplication and division of *labels* are defined so that the result label includes all of the properties of the operands' labels. The resulting label is the union of the operands' labels, with ancestors of ancestor-descendent pairs removed. For example:

```
2 qty. (apples) * $0.50 / qty. (apples) =
           $1.00 (apples)

2 qty. (apples) * $0.50 / qty. (oranges) =
           $1.00 (apples, oranges)
```

The goal of the choice to include *all* the properties of the operands was to reveal errors, rather than hiding them. Errors are highlighted because labels unexpectedly contain irrelevant properties. An alternative is to include only the properties common to *both* operands, similar to addition and subtraction, above. This represents a tradeoff between recording the history of the computation—perhaps resulting in large, unreadable labels—or erasing it, and hiding potential errors. However, spreadsheets are not frequently used to compute products of large sets of items, and chains of computation tend to be relatively short, so the large, unreadable case is not common.

Note that we chose different definitions for multiplication and division of labels than for addition and subtraction. It is common in spreadsheets to add values with disparate objects of measurement, and showing all the labels for all these cases would be too hard to read. However, this means that some reference errors in summations will not be highlighted by SLATE.

5. User Interface Issues

One central factor that determines SLATE's effectiveness is the ease with which users may enter labels. If users choose not to enter labels, the system does not provide benefits. Currently, users must type units and labels as text. This has the advantage of permitting users to enter data with only a keyboard. A disadvantage, however, is that users must remember to type them correctly, and learn the syntax of units and labels.

It would be advantageous to have a GUI tool for entering units and labels. Units could be entered with a sequence of pop-up menus, with one per base unit, with text boxes for exponents. Since the label context is a hierarchy, it would be logical to choose nodes in a manner analogous to other hierarchical systems, such as file systems.

Creating the unit and label contexts is somewhat more complex. Fortunately, not all users need to do this; it suffices for one expert user to create contexts for a large group of users with similar needs. Users could use a hierarchical editor, for example, to create label contexts.

Header inference techniques from Apples and Oranges [3] could be adopted in SLATE: the system could infer some label information from the headers. Also, it is important to evaluate SLATE with user studies; this is vital both for demonstrating SLATE's ability to help users detect errors. The system must also be tested with large contexts and data sets to show its effectiveness with large amounts of data.

6. Conclusions

SLATE represents a new approach to spreadsheet formula error detection. By considering objects of measurement in addition to units of measurement, it is possible to detect errors that would be hidden in other systems. Although user tests are required to determine whether it will be effective in practice, SLATE has the potential to uncover a new class of errors.

7. Acknowledgments

8. References

[1] Coblenz, Michael. "Using Objects of Measurement to Detect Spreadsheet Errors", Technical Report CMU-CS-05-150, School of Computer Science, Carnegie Mellon University, Pittsburgh, Pennsylvania, July 2005

[2] Panko, Raymond R. "What We Know About Spreadsheet Errors", *Journal of End User Computing*, Idea Group Publishing, Spring 1998, pp. 15-21.

[3] Erwig, Martin; and Burnett, Margaret. "Adding Apples and Oranges", *4th Int. Symp. on Practical Aspects of Declarative Languages*, Portland, Oregon, LNCS 2257, pp. 173-191, 2002

[4] Y. Ahmad, T. Antoniu, and S. Goldwater. "A Type System for Statically Detecting Spreadsheet Errors", IEEE Conf. Automated Soft. Engr., pp. 174-183, 2003.

[5] Abraham, Robin and Erwig, Martin. "Header and Unit Inference for Spreadsheets Through Spatial Analyses", *IEEE Symposium on Visual Languages and Human-Centric Computing*, Rome, Italy, September 2004, pp. 165-172.