Bidirectional Spreadsheet Formulas

Nuno Macedo Hugo Pacheco Nuno Sousa Alcino Cunha

VL/HCC 2014
August 1, Melbourne, Australia

Spreadsheet Formulas

- spreadsheet *formulas* define computations and are at the core of spreadsheets;
- allow the calculation of values at an *target* cell from a set of *source* cells.

A	B	C = A + B
10	5	15

Spreadsheet Formulas

- spreadsheet *formulas* define computations and are at the core of spreadsheets;
- allow the calculation of values at an *target* cell from a set of *source* cells.

A	B	C = A + B
10	10	15

Spreadsheet Formulas

- spreadsheet *formulas* define computations and are at the core of spreadsheets;
- allow the calculation of values at an *target* cell from a set of *source* cells.

A	B	C = A + B
10	10	20

Spreadsheet Formulas

- in some scenarios it is helpful to update the *target* cell and have the changes reflected in the *source* cells
  - interchangeable formats;
  - reverse formulas;
  - fix errors;
  - "what-if" scenarios;

A	B	C = A + B
10	5	15

- not supported natively by spreadsheet systems.

Spreadsheet Formulas

- in some scenarios it is helpful to update the *target* cell and have the changes reflected in the *source* cells
  - interchangeable formats;
  - reverse formulas;
  - fix errors;
  - "what-if" scenarios;

A	B	C = A + B
10	5	20

- not supported natively by spreadsheet systems.

Spreadsheet Formulas

- in some scenarios it is helpful to update the *target* cell and have the changes reflected in the *source* cells
  - interchangeable formats;
  - reverse formulas;
  - fix errors;
  - "what-if" scenarios;

A	B	C = A + B
10	10	20

- not supported natively by spreadsheet systems.

Example

- profit forecasting example:

=RIGHT(#B2;4) =D2+E2 =IF(H2>0;#H2;"Loss")

	A	B	C	D	E	F	G	H	I
1	id	Name	Ref.	Cost	Taxes	Profit %	T. Cost	Profit €	Print
2	1	TV LCD Ref. 5555	5555	50,00 €	3,00 €	1,4	53,00 €	21,20 €	21,20 €
3	2	Blu-ray Player Ref. 1231	1231	20,00 €	2,00 €	1,5	22,00 €	11,00 €	11,00 €
4	3	Digital Camera Ref. 4235	4235	5,00 €	1,00 €	0,5	6,00 €	- 3,00 €	Loss
5	4	GPS Navigator Ref. 3468	3468	24,00 €	5,00 €	2	29,00 €	29,00 €	29,00 €

=#F2*G2-G2

Design Principles

- spreadsheets are widely used by *non-proficient* users;
- focus on seamless integration:
  - *intuitive* to traditional users;
  - *conservative* as to not affect standard behavior;
  - *transparent* to be predictable to the user.

Challenges

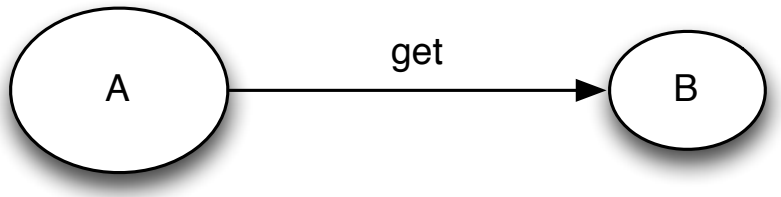
- formulas are typically *non-injective*;
- multiple ways to define backward transformations;
- to uphold the previous principles the user should be able to *control* and *inspect* the chosen one.

Bidirectional Transformation

- data transformation abounds in software engineering;
- often performed in both ways: maintainability problem;
- extensive work has been done on *bidirectional transformation*;
- a *single artifact* specifies both forward and backward transformations.

Lenses

- *lenses* are one of the most popular approaches;
- designed for asymmetrical scenarios;



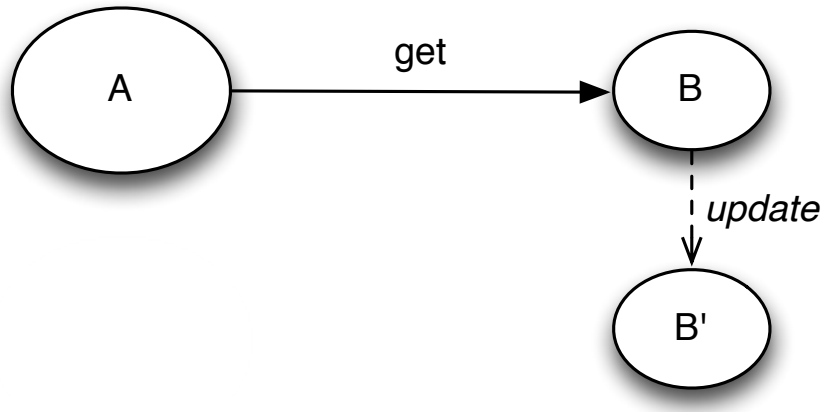
$$\text{get}\ (\text{put}\ (v, s)) = v \qquad \text{(Acceptable)}$$

$$\text{put}\ (\text{get}\ s, s) = s \qquad \text{(Stable)}$$

- the transformation get entails the putback put.

Lenses

- *lenses* are one of the most popular approaches;
- designed for asymmetrical scenarios;



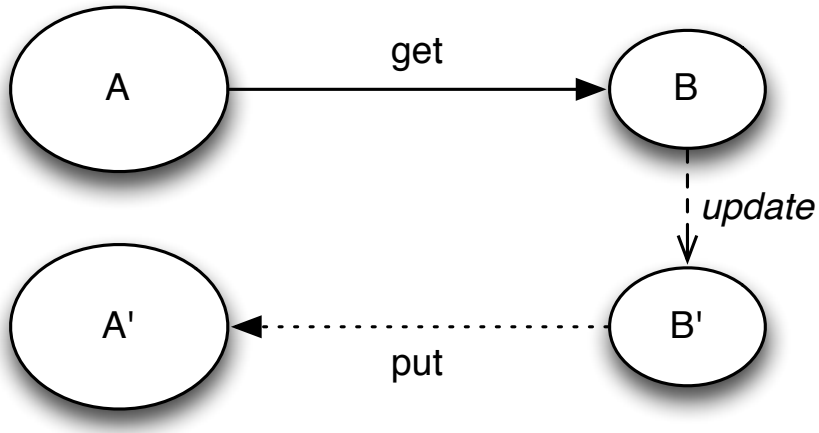
$$\text{get}\ (\text{put}\ (v, s)) = v \qquad \text{(ACCEPTABLE)}$$

$$\text{put}\ (\text{get}\ s, s) = s \qquad \text{(STABLE)}$$

- the transformation get entails the putback put.

Lenses

- *lenses* are one of the most popular approaches;
- designed for asymmetrical scenarios;



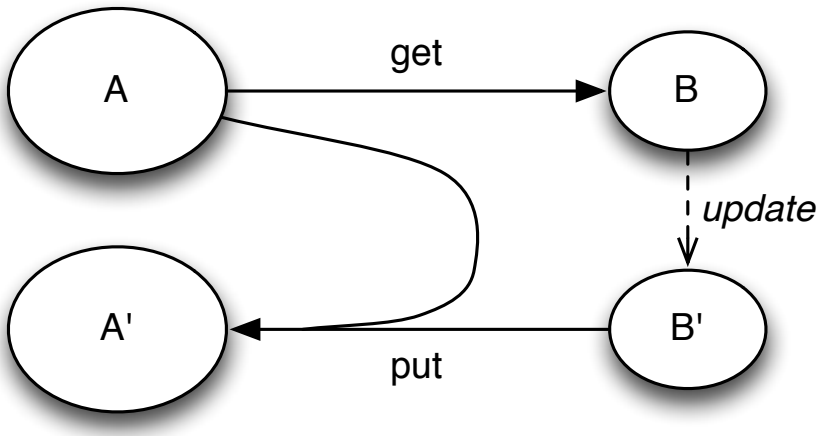
$$\text{get}\ (\text{put}\ (v, s)) = v \qquad \text{(Acceptable)}$$

$$\text{put}\ (\text{get}\ s, s) = s \qquad \text{(Stable)}$$

- the transformation get entails the putback put.

Lenses

- *lenses* are one of the most popular approaches;
- designed for asymmetrical scenarios;



$$\text{get}\ (\text{put}\ (v, s)) = v \qquad \text{(Acceptable)}$$

$$\text{put}\ (\text{get}\ s, s) = s \qquad \text{(Stable)}$$

- the transformation get entails the putback put.

Online Setting

- source cell updates *automatically* trigger target updates;
- target updates should also trigger updates on source cells;
- no cyclic dependencies;
- system converges to consistent state:
  - ACCEPTABLE: after update propagation no forward computation is required;
  - STABLE: no update means no forward computation.

Conservative Updating

- user controls which cells are susceptible to updates;
- #-marks on source cells: C = #A + B;
- example: changes on profit shall not alter production costs;
- *conservative*: no # marks result in standard behavior.

Scheme

- spreadsheet formula $f : A_1 \times ... \times A_n \rightarrow B$;
- for each #-marked cell, a putback is generated;
- behavior depends on the set of marked cells.

Example: Addition

- $B = \#A_1 + A_2$

  $\text{put}_{\blacksquare + A_2} : B \times (A_1 \times A_2) \to A_1$
  $\text{put}_{\blacksquare + A_2}\ (b, (a_1, a_2)) = b - a_2$

- $B = \#A_1 + \#A_2$

  $\text{put}_{\blacksquare + \square} : B \times (A_1 \times A_2) \to A_1$
  $\text{put}_{\blacksquare + \square}\ (b, (a_1, a_2)) = b \,/\, 2$

  $\text{put}_{\square + \blacksquare} : B \times (A_1 \times A_2) \to A_2$
  $\text{put}_{\square + \blacksquare}\ (b, (a_1, a_2)) = b \,/\, 2$

Example: Length

- $B = LEN(\#A)$

$$\text{put}_{\text{LEN}\ (\blacksquare)} : B \times A \to A$$
$$\text{put}_{\text{LEN}\ (\blacksquare)}\ (b, a) = \textbf{if}\ b \leqslant \text{LEN}\ (a)\ \textbf{then}\ \text{LEFT}\ (a, b)$$
$$\textbf{else}\ a\ \&\ \text{REPEAT}\ (\texttt{"A"}, b - \text{LEN}\ (a))$$

Formula Chaining

- nested formulas ($B = g\ (f\ (A))$) are assumed to be decomposed into chains ($B = g\ (X), X = f\ (A)$);
- composition relies on spreadsheets' reactive nature;

A	**B**	**C = LEN(#B)**	**D = A + #C**
10	"hello"	5	15

- some restrictions needed to be sound: dependencies must form a tree whose nodes are value cells.

Formula Chaining

- nested formulas ($B = g\ (f\ (A))$) are assumed to be decomposed into chains ($B = g\ (X), X = f\ (A)$);
- composition relies on spreadsheets' reactive nature;

A	**B**	**C = LEN(#B)**	**D = A + #C**
10	"hello"	5	12

- some restrictions needed to be sound: dependencies must form a tree whose nodes are value cells.

Formula Chaining

- nested formulas ($B = g\ (f\ (A))$) are assumed to be decomposed into chains ($B = g\ (X), X = f\ (A)$);
- composition relies on spreadsheets' reactive nature;

A	**B**	**C = LEN(#B)**	**D = A + #C**
10	"hello"	2	12

- some restrictions needed to be sound: dependencies must form a tree whose nodes are value cells.

Formula Chaining

- nested formulas ($B = g\ (f\ (A))$) are assumed to be decomposed into chains ($B = g\ (X), X = f\ (A)$);
- composition relies on spreadsheets' reactive nature;

A	**B**	**C = LEN(#B)**	**D = A + #C**
10	"he"	2	12

- some restrictions needed to be sound: dependencies must form a tree whose nodes are value cells.

Invariants: Range

- formulas are not *surjective*;
- updates to values outside the range render the system *inconsistent*;

A	**B**	**C = LEN(#B)**	**D = A + #C**
10	"hello"	-5	5

- unlike unidirectional scenario, the user may not be aware of the range of the formulas;
- aggravated by chains of formulas.

Invariants: User

- the user may wish to further control update propagations;
- may introduce *constraints* into (source) cells;
- must be propagated to target cells in order to restrict updates.

A $\geqslant 0$	**B**	**C = LEN(#B)** $\geqslant 0$	**D = #A + #C** $\geqslant 0$
0	""	0	0

Invariant Language

- tradeoff between expressiveness and effective manipulation;
- should be propagated through the formula chains;
- each function calculates range through symbolic execution:

$$\{[0..10]\} + \{[20..30]\} = \{[20..40]\}$$
$$\mathsf{LEN}\,\{\texttt{"abc"}, \texttt{"xyz"}, \mathsf{len}_{[4..10]}\} = \{[3..10]\}$$

Example Revisited

=RIGHT(#B2;4) =D2+E2 =IF(H2>0;#H2;"Loss")

	A	B	C	D	E	F	G	H	I
1	id	Name	Ref.	Cost	Taxes	Profit %	T. Cost	Profit €	Print
2	1	TV LCD Ref. 5555	5555	50,00 €	3,00 €	1,4	53,00 €	21,20 €	21,20 €
3	2	Blu-ray Player Ref. 1231	1231	20,00 €	2,00 €	1,5	22,00 €	11,00 €	11,00 €
4	3	Digital Camera Ref. 4235	4235	5,00 €	1,00 €	0,5	6,00 €	- 3,00 €	Loss
5	4	GPS Navigator Ref. 3468	3468	24,00 €	5,00 €	2	29,00 €	29,00 €	29,00 €

=#F2*G2-G2

- Profit % $\leqslant 2$;
- Profit € $\leqslant$ T. Cost;
- $0 \leqslant$ Print $\leqslant$ T. Cost $\vee$ Print $=$ `Loss`.

Formula Synthesis

- putbacks must *adapt* themselves to the cell's invariants;
- *white-box*: backward transformations are written as spreadsheet formulas;
- for each function, a synthesis procedure that generates putback for current invariants;
- prevailing ambiguities can be controlled by the user.

Synthesis: Example

- LEN (A) over constraint $\{\text{"abc"}, \text{"xyz"}, \text{len}_{[4..10]}\}$;
- range $\{[3..10]\}$;
- synthesized putback:

$\text{put}_{\text{LEN}\ (\blacksquare)}\ (b, a) =$
  if $b : \{3\}$ **then**
    if $b \leqslant \text{LEN}\ (a)$ **then**
      if $\text{LEFT}\ (b, a) = \text{"abc"}$ **then** "abc"
      else if $\text{LEFT}\ (b, a) = \text{"xyz"}$ **then** "xyz"
      else $\text{sel}\ (\{\text{"abc"}, \text{"xyz"}\}, \text{LEFT}\ (b, a))$
    else
      if $\text{LEFT}\ (b, \text{"abc"}) = a$ **then** "abc"
      else if $\text{LEFT}\ (b, \text{"xyz"}) = a$ **then** "xyz"
      else $\text{sel}\ (\{\text{"abc"}, \text{"xyz"}\}, a)$
  if $b : \{[4..10]\}$ **then**
    if $b \leqslant \text{LEN}\ (a)$ **then** $\text{LEFT}\ (b, a)$
    else $a\ \&\ \text{sel}\ (\{\text{len}_{b-\text{LEN}\ a}\}, a)$

Conclusions

- we have proposed a technique for the bidirectionalization of spreadsheet formulas;
- prototype implemented as an *Excel* plugin;
- overhead: every supported function must be have a synthesis procedure defined;
- intuitive, transparent and conservative? would require an empirical study.