Haxcel: A Spreadsheet Interface to Haskell written in Java

Johan Malmström
Dept. of Computer Science and Engineering
Mälardalen University

jmm98002@idt.mdh.se http://www.mds.mdh.se/~dal98jmm
http://www.mrtc.mdh.se/projects/Haxcel/

2002-12-17

Johan Malmström 2002
Abstract

The spreadsheet paradigm offers a fast interactive loop, where the effects of updates to definitions and data are immediately visible. This makes the paradigm attractive for program development, where redefinitions can be immediately tested and the results displayed. We have designed a simple, compiler-independent spreadsheet interface to Haskell, where cells host Haskell definitions. Spreadsheets are also used for high-level array calculations. In order to meet that demand we present an extended array library for Haskell, which provides a number of typical array-language facilities. Together, the interface and the array library provide an interactive environment that can be used both for development of general Haskell code and for array-oriented spreadsheet calculations.
Introduction

Spreadsheets History and a introduction to common spreadsheets.
Introduction

**Spreadsheets** History and an introduction to common spreadsheets.

**Haskell** A brief introduction to the functional programming language Haskell
Introduction

**Spreadsheets**  History and a introduction to common spreadsheets.

**Haskell**  A brief introduction to the functional programming language Haskell

**Haxcel**  Our prototype.
**Introduction**

**Spreadsheets** History and a introduction to common spreadsheets.

**Haskell** A brief introduction to the functional programming language Haskell

**Haxcel** Our prototype.

**Xarray** A extended array library to Haskell which provides a number of typical array-language facilities.
Introduction

Spreadsheets  History and a introduction to common spreadsheets.

Haskell  A brief introduction to the functional programming language Haskell

Haxcel  Our prototype.

Xarray  A extended array library to Haskell which provides a number of typical array-language facilities.

Conclusions  and future work

Johan Malmström 2002
Spreadsheets

VisiCalc introduces spreadsheets for computers 1979 for Apple II
VisiCalc introduces spreadsheets for computers 1979 for Apple II

Common spreadsheet applications like Microsoft’s Excel and P & L Software’s Mesa share the same setup:

- 2-D area
- Cells, Rows, Columns
- Value, Formula, Format
**Spreadsheets**

**VisiCalc** introduces spreadsheets for computers 1979 for Apple II

**Common spreadsheet applications** like Microsoft’s *Excel* and P & L Software’s *Mesa* share the same setup:

- 2-D area
- Cells, Rows, Columns
- Value, Formula, Format

**Applications** without traditional programming skills
VisiCalc introduces spreadsheets for computers 1979 for Apple II

Common spreadsheet applications like Microsoft’s Excel and P & L Software’s Mesa share the same setup:

- 2-D area
- Cells, Rows, Columns
- Value, Formula, Format

Applications without traditional programming skills

Definition-Eval-Display loop gives immediate feedback
The Excel formula in the top-left corner of the sheet is:

\[ \frac{(D4 - D2)}{D4 \times 100} \]

The table below the formula contains the following columns:

- **A**: Series
- **B**: Start
- **C**: End
- **D**: Growth
- **E**: Decrease

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>G</th>
<th>H</th>
<th>I</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>0.0070E+</td>
<td>0.25</td>
<td>0.89</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>0.0070E+</td>
<td>0.22</td>
<td>0.86</td>
<td>-3.98</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>3</td>
<td>0.0070E+</td>
<td>0.17</td>
<td>0.79</td>
<td>-13.15</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>4</td>
<td>0.0070E+</td>
<td>0.0540E+</td>
<td>0.51</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>5</td>
<td>0.0070E+</td>
<td>0.0120E+</td>
<td>0.13</td>
<td>-563.3718E+</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>6</td>
<td>0.0070E+</td>
<td>0.0060E+</td>
<td>-0.0385E+</td>
<td>2419.5167E+</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>7</td>
<td>0.0070E+</td>
<td>0.0050E+</td>
<td>-0.0841E+</td>
<td>1162.6585E+</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Johan Malmström 2002
Problems in common spreadsheet applications

- Insertion of rows and columns
Problems in common spreadsheet applications

- Insertion of rows and columns
- Hard to do simple data manipulations
Problems in common spreadsheet applications

- Insertion of rows and columns
- Hard to do simple data manipulations
- Code reuse and function definitions
Problems in common spreadsheet applications

- Insertion of rows and columns
- Hard to do simple data manipulations
- Code reuse and function definitions
- No local variables
Problems in common spreadsheet applications

- Insertion of rows and columns
- Hard to do simple data manipulations
- Code reuse and function definitions
- No local variables
- Limited number of data types
Problems in common spreadsheet applications

- Insertion of rows and columns
- Hard to do simple data manipulations
- Code reuse and function definitions
- No local variables
- Limited number of data types
- No recursion in languages
Haskell

- Functional programming language
Haskell

- Functional programming language
- Haskell 98, Haskell Report
Haskell

- Functional programming language
- Haskell 98, Haskell Report
- Modules and declarations
Haskell

- Functional programming language
- Haskell 98, Haskell Report
- Modules and declarations
- hmake and hi works with nhc, ghc and hbc.
Haskell

- Functional programming language
- Haskell 98, Haskell Report
- Modules and declarations
- hmake and hi works with nhc, ghc and hbc.
- Hugs and runhugs fast interpreter.

Johan Malmström 2002
A very simple example of a Haskell module with one declaration.

module Max2 where

-- max2 returns the largest of two integers
max2 :: Int -> Int -> Int
max2 x y
= if x >= y then x else y
Haxcel

- Intro
  - Java application
  - Independent from Haskell compilers
  - Multi-window spreadsheet
Haxcel

- Intro
  - Java application
  - Independent from Haskell compilers
  - Multi-window spreadsheet

- Single Module
• Intro
  ★ Java application
  ★ Independent from Haskell compilers
  ★ Multi-window spreadsheet

• Single Module

• Create declarations
Haxcel

- Intro
  - Java application
  - Independent from Haskell compilers
  - Multi-window spreadsheet

- Single Module

- Create declarations

- Export Module to conventional Haskell file
Simple example
Xarray

- Array language
Xarray

- Array language

- Some of the basic features of Xarray:
  - Computing bounds of arrays with functions `join` and `meet`
  - Scalable arrays is provided through the type of the extended array.
  - Projection operations
  - Dimensions operations, makes matrices out of vectors and splits matrices into vectors
Budget example
Conclusions

- Compiler-independent spreadsheet interface for Haskell
Conclusions

- Compiler-independent spreadsheet interface for Haskell
- Overcomes some of the problems in conventional spreadsheets
Conclusions

- Compiler-independent spreadsheet interface for Haskell
- Overcomes some of the problems in conventional spreadsheets
- Haskell is good for spreadsheet programming
Conclusions

- Compiler-independent spreadsheet interface for Haskell
- Overcomes some of the problems in conventional spreadsheets
- Haskell is good for spreadsheet programming
- Fast with runhugs, annoyingly slow with hmake/nhc
Conclusions

- Compiler-independent spreadsheet interface for Haskell
- Overcomes some of the problems in conventional spreadsheets
- Haskell is good for spreadsheet programming
- Fast with runhugs, annoyingly slow with hmake/nhc
- Import of .hs file, literate scripts

Johan Malmström 2002
MRTC

http://www.mrtc.mdh.se/projects/Haxcel/