

Stratego Shell



Martin Bravenboer

`martin@cs.uu.nl`

Institute of Information and Computing Sciences, University Utrecht, The Netherlands

Features

- reusable *Stratego interpreter*
 - major improvement of `stri`
- provides *various interfaces*
 - command line
 - script
 - program
- *education* and *debugging* features
 - examine the environment
 - breakpoints
 - insert in pipe
- supports *concrete object syntax*

Techniques: Command Line

- *user-friendly*
 - history
 - backspace, del, home, end works as it should
 - tab completion of strategy names
- *reusable*
 - primitives for GNU Readline
 - register strategy for tab completion
 - implementing your own shell is *trivial*
- command parser: *sgr*
 - performance is no issue

Techniques: Interpreter and Shell

- *interpreter*
 - implemented in Stratego
 - evaluates 'core' Stratego
 - reusable in different applications
 - allows extension of Stratego
- *normalization*
 - reuses *strc* components
 - no defined checks
- *concise implementation*
 - interpreter: 720 (Stratego)
 - shell: 667 (Stratego) + 116 (C) + 115 (SDF)

Various Interfaces

- *interactive* Stratego commands
 - if not script or program
 - `stratego-shell`
- Stratego *script*
 - sequence of commands terminated by `;;`
 - preferred extension: `.strs`
 - `stratego-shell --script file`
- Stratego *program*
 - invoke a main strategy in a module
 - desugared to a script (no options)
 - `stratego-shell --prg main@file`

Input and Output

- *atarm to atarm*
 - shell is *specialized* to a transformation tool

- all interfaces can be used in pipelines

```
... | stratego-shell --script ... | ...
```

```
... | stratego-shell --prg ... | ...
```

```
... | stratego-shell | ...
```

- input: `-i` or `stdin`
 - `stdin` only used if not a terminal
- output: `-o` and `stdout`

Basic Concepts

- *current term*
 - `:show`
 - by default shown after each command
 - `:autoshow on/off`
- *rewrite* current term
- *failure* preserves current term
- *environment* of term bindings
 - `:binding x`
- commands are interpreted in the same *scope*

Strategy Definition

- *environment* of strategy definitions
 - `:showdef x`
 - `:showast x`
- `import lib`
 - supports overloaded definitions
 - supports concrete object syntax
- definition at command line
 - `foo = s`
 - `Foo : p1 -> p2`
- overloaded definitions are composed with choice

Breakpoints

- interrupt evaluation to inspect the environment
- `\#foo` strategy opens a *break shell*
- inspect
 - `:binding x`
 - `:showdef x`
 - including local ones and arguments
- `:continue` resumes evaluation
- *promising* debugging technique

Concrete Object Syntax

- `:syntax Stratego-Box`
- Stratego Shell's command language is added automatically
- no escape from bad syntax definition
- useful tool for experimenting with embeddings
- also supported in scripts
- demos in the next presentations

Future Work

- commands are not isolated from the shell
- multiple extensions of Stratego syntax
- use signatures
 - constructor checks
 - tab completion of constructors
- more educational and debugging features
 - step, trace, dynamic rule visualization (Arthur van Dam)
- pretty-printing of intermediate results
- support exotic Stratego features
`Thread, Abs, SVar (Mod("Cons" , "T"))`
- combine with xtc shell (wednesday)