Executable Formal Semantics for the POSIX Shell

Michael Greenberg and Austin J. Blatt

Pomona College Pomona ’18, now Puppet Labs
i’m interested in powerful programming languages
you want power?

you want the shell
i wrote a new POSIX shell

smoosh

the Symbolic, Mechanized, Observable, Operational SHell
system mode

core smoosh
expansion
evaluation

OS model
POSIX definitions

system calls
(OCaml: Sys, Unix, ExtUnix)

builtins

libdash parser

shell driver
your mode

core smoosh
evaluation

expansion

POSIX definitions
OS model

builtins
libdash parser

your analysis here?

your model here?
parsing

expansion

evaluation

basename `pwd`
basename `pwd`

basename /Users/mgree/talks/smoosh
basename `pwd`

basename /Users/mgreek/talks/smoosh

smoosh
basename `pwd`

basename /Users/mgree/talks/smoosh

smoosh
parsing

expansion

evaluation
parsing

expansion

evaluation

expansion

evaluation
the shteeper

http://shell.cs.pomona.edu/shteeper
\[ c ::= \texttt{v=w} \ldots \texttt{w} \ldots | \ c \ r \]

| \texttt{c}_1 | \texttt{c}_2 | \texttt{c}_3 | \ldots | \texttt{c}_n | \ c \ & \ | (\ c ) |
|----------------|----------------|----------------|----------------|
| \texttt{c}_1 \&\& \texttt{c}_2 | \texttt{c}_1 \ |\ | \texttt{c}_2 |
| \texttt{! c} | \texttt{c}_1 ; \texttt{c}_2 | \texttt{if c}_1 \ \texttt{c}_2 \ \texttt{c}_3 |
| \texttt{switch a} \ldots \{ \texttt{case w} \ldots \} \ c \} \ldots |
| \texttt{while c}_1 \ \texttt{c}_2 | \texttt{for x in w} \ldots \ c |
| \texttt{defun v} \ c |
\[ c ::= v=w \ldots w \ldots \mid c \; r \]

\[ \mid c_1|c_2|c_3|\ldots|c_n \mid c \; \& \mid (c) \]

\[ \mid c_1 \; \&\& c_2 \mid c_1 \; || c_2 \]

\[ \mid !c \mid c_1 \; ; c_2 \mid \text{if } c_1 \; c_2 \; c_3 \]

\[ \mid \text{switch } a \; \ldots \{ \text{case } w \ldots \} \; c \; \} \; \ldots \]

\[ \mid \text{while } c_1 \; c_2 \mid \text{for } x \; \text{in } w \; \ldots \; c \]

\[ \mid \text{defun } v \; c \]
c ::= v=w ... w ... | c r

| c_1 | c_2 | c_3 | ... | c_n | c & | ( c )

| c_1 && c_2 | c_1 || c_2

| ! c | c_1 ; c_2 | if c_1 c_2 c_3

| switch a ... { case w... } c } ...

| while c_1 c_2 | for x in w ... c

| defun v c
\[
c ::= \text{v}\text{=}w \ldots w \ldots \mid c \text{ r}
\]
\[
\mid c_1|c_2|c_3|\ldots|c_n \mid c \& \mid (c)
\]
\[
\mid c_1 \&\& c_2 \mid c_1 ||c_2
\]
\[
\mid !c \mid c_1 ; c_2 \mid \text{if } c_1 c_2 c_3
\]
\[
\mid \text{switch } a \ldots \{ \text{case } w \ldots \} c \} \ldots
\]
\[
\mid \text{while } c_1 c_2 \mid \text{for } x \text{ in } w \ldots c
\]
\[
\mid \text{defun } v \ c
\]
\[
\begin{align*}
c & ::= v = w \ldots w \ldots | c \ \& \ c \\
& \mid c_1 c_2 c_3 c_n | c \ \& \ (c) \\
& \mid c_1 \ \& \& c_2 | c_1 | | c_2 \\
& \mid \neg c | c_1 ; c_2 | \text{if } c_1 c_2 c_3 \\
& \mid \text{switch } a \ldots \text{case } w \ldots ) c \} ... \\
& \mid \text{while } c_1 c_2 | \text{for } x \text{ in } w \ldots c \\
& \mid \text{defun } v \ c
\end{align*}
\]
### User Input

1. **user input**

### Control Codes

2. **control codes**

### Splitting

3. **splitting**

### Globbing

4. **globbing**

### Quote Removal

5. **quote removal**

---

**Words**

\[ \text{words } w \]

**Control codes**

\[ \text{control codes } k \]

**Parameter formats**

\[ \text{parameter formats } \phi \]

**Strings**

\[ \text{strings } s \in \text{ASCII/locale} \]

**Expanded words**

\[ \text{expanded words } ew \]

**Intermediate fields**

\[ \text{intermediate fields } i \]

**Fields**

\[ \text{fields } f \]

---

**Informative, not in the spec!**

---

**Words**

\[ w := ( s | k | \omega )^* \]

**Control codes**

\[ k := \sim | \sim s | \$s|\phi\$ | \$(c) | \$(w) | "w" \]

**Parameter formats**

\[ \phi := \text{normal} | \text{length} | \text{default } w | \ldots \]

**Strings**

\[ s \in \text{ASCII/locale} \]

**Expanded words**

\[ ew := ( \text{usr } s | \text{exp } s | \omega | @ f | "s" )^* \]

**Intermediate fields**

\[ i := ( \text{ws } \omega | \omega | s | "s" )^* \]

**Fields**

\[ f := s_1 \ldots s_n \]
1. user input

2. control codes

3. splitting

4. globbing

5. quote removal

\[ w \rightarrow ^{*}_2 \text{ew} \rightarrow ^{*}_3 \text{i} \rightarrow ^{*}_{4,5} \text{f} \]

\[ \{x=w\} \rightarrow ^{2}_{f} \]
what’s unique about the shell’s semantics

• Modal: expansion and evaluation

• Lots of mutual recursion

• Language semantics uses system calls

• Runtime AST includes nested command loops
e.g., source a/k/a . and eval; top level and -i
who cares about semantics?

- **lingua franca** for PL folks
- **symbolic execution**
  - stepper
  - bug finding, static analysis
  - support DevOps tools like Rehearsal
- **correctness** baseline for compilers, other tools
- insights into the **shell** and other **interactive languages**
comparing smoosh

<table>
<thead>
<tr>
<th></th>
<th>Smoosh</th>
<th>bash*</th>
<th>dash</th>
<th>zsh†</th>
<th>mksh</th>
<th>ksh</th>
<th>yash*</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.1</td>
<td>4.4-12(1)</td>
<td>0.5.8-2.4</td>
<td>5.3.1-4+b2</td>
<td>54-2+b4</td>
<td>93u+</td>
<td>2.43-1</td>
</tr>
</tbody>
</table>

**POSIX test suite (418 tests)**

<table>
<thead>
<tr>
<th></th>
<th>Failing tests</th>
<th>Time to run</th>
</tr>
</thead>
<tbody>
<tr>
<td>Smoosh</td>
<td>0</td>
<td>12m41s</td>
</tr>
<tr>
<td>bash*</td>
<td>4 (8)</td>
<td>2m43s</td>
</tr>
<tr>
<td>dash</td>
<td>20</td>
<td>2m43s</td>
</tr>
<tr>
<td>zsh†</td>
<td>×</td>
<td>2m52s</td>
</tr>
<tr>
<td>mksh</td>
<td>35</td>
<td>3m24s</td>
</tr>
<tr>
<td>ksh</td>
<td>23</td>
<td>2m45</td>
</tr>
<tr>
<td>yash*</td>
<td>22 (23)</td>
<td>×</td>
</tr>
</tbody>
</table>

**Modernish’s shell diagnosis (91 potential bugs, 22 potential quirks) and test suite (312 tests)**

<table>
<thead>
<tr>
<th></th>
<th>Bugs</th>
<th>Quirks</th>
<th>Failing tests</th>
<th>Time to run</th>
</tr>
</thead>
<tbody>
<tr>
<td>Smoosh</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>5.5s</td>
</tr>
<tr>
<td>bash*</td>
<td>16</td>
<td>4</td>
<td>20</td>
<td>4.8s</td>
</tr>
<tr>
<td>dash</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>1.4s</td>
</tr>
<tr>
<td>zsh†</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>1.2s†</td>
</tr>
<tr>
<td>mksh</td>
<td>3</td>
<td>2</td>
<td>3</td>
<td>3.2s</td>
</tr>
<tr>
<td>ksh</td>
<td>14</td>
<td>3</td>
<td>17</td>
<td>2.2s</td>
</tr>
<tr>
<td>yash*</td>
<td>1</td>
<td>8</td>
<td>1</td>
<td>2.4s</td>
</tr>
</tbody>
</table>

**Smoosh’s test suite (161 tests)**

<table>
<thead>
<tr>
<th></th>
<th>Failing tests</th>
<th>Time to run</th>
</tr>
</thead>
<tbody>
<tr>
<td>Smoosh</td>
<td>0</td>
<td>23s</td>
</tr>
<tr>
<td>bash*</td>
<td>30 (35)</td>
<td>28s</td>
</tr>
<tr>
<td>dash</td>
<td>42</td>
<td>1m13s</td>
</tr>
<tr>
<td>zsh†</td>
<td>52</td>
<td>42s</td>
</tr>
<tr>
<td>mksh</td>
<td>34</td>
<td>21s</td>
</tr>
<tr>
<td>ksh</td>
<td>41</td>
<td>28s</td>
</tr>
<tr>
<td>yash*</td>
<td>43 (39)</td>
<td>29s</td>
</tr>
</tbody>
</table>
## Comparing Smoosh

<table>
<thead>
<tr>
<th></th>
<th>Smoosh</th>
<th>bash*</th>
<th>dash</th>
<th>zsh†</th>
<th>mksh</th>
<th>ksh</th>
<th>yash*</th>
</tr>
</thead>
<tbody>
<tr>
<td>Version</td>
<td>0.1</td>
<td>4.4-12(1)</td>
<td>0.5.8-2.4</td>
<td>5.3.1-4+b2</td>
<td>54-2+b4</td>
<td>93u+</td>
<td>2.43-1</td>
</tr>
</tbody>
</table>

### POSIX Test Suite (418 Tests)

<p>| | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Failing tests</td>
<td>0</td>
<td>4 (8)</td>
<td>20</td>
<td>×</td>
<td>35</td>
<td>23</td>
</tr>
<tr>
<td>Time to run</td>
<td>12m41s</td>
<td>2m43s</td>
<td>2m43s</td>
<td>×</td>
<td>2m52s</td>
<td>3m24s</td>
</tr>
</tbody>
</table>

### Modernish's Shell Diagnosis (91 Potential Bugs, 22 Potential Quirks) and Test Suite (312 Tests)

<p>| | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Bugs</td>
<td>1</td>
<td>16</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>14</td>
<td>1</td>
</tr>
<tr>
<td>Quirks</td>
<td>0</td>
<td>4</td>
<td>2</td>
<td>5</td>
<td>2</td>
<td>3</td>
<td>8</td>
</tr>
<tr>
<td>Failing tests</td>
<td>1</td>
<td>20</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>17</td>
<td>1</td>
</tr>
<tr>
<td>Time to run</td>
<td>5.5s</td>
<td>4.8s</td>
<td>1.4s</td>
<td>1.2s†</td>
<td>3.2s</td>
<td>2.2s</td>
<td>2.4s</td>
</tr>
</tbody>
</table>

### Smoosh's Test Suite (161 Tests)

<p>| | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Failing tests</td>
<td>0</td>
<td>30 (35)</td>
<td>42</td>
<td>52</td>
<td>34</td>
<td>41</td>
<td>43 (39)</td>
</tr>
<tr>
<td>Time to run</td>
<td>23s</td>
<td>28s</td>
<td>1m13s</td>
<td>42s</td>
<td>21s</td>
<td>28s</td>
<td>29s</td>
</tr>
</tbody>
</table>
## Comparing Smoosh

<table>
<thead>
<tr>
<th></th>
<th>Smoosh</th>
<th>bash*</th>
<th>dash</th>
<th>zsh‡</th>
<th>mksh</th>
<th>ksh</th>
<th>yash*</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time to run</td>
<td>0.1</td>
<td>4.4-12(1)</td>
<td>0.5.8-2.4</td>
<td>5.3.1-4+b2</td>
<td>54-2+b4</td>
<td>93u+</td>
<td>2.43-1</td>
</tr>
<tr>
<td>Failing tests</td>
<td>0</td>
<td>4 (8)</td>
<td>20</td>
<td>×</td>
<td>35</td>
<td>23</td>
<td>22 (23)</td>
</tr>
<tr>
<td>Modernish’s shell diagnosis (91 potential bugs, 22 potential quirks) and test suite (312 tests)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bugs</td>
<td>1</td>
<td>16</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>14</td>
<td>1</td>
</tr>
<tr>
<td>Quirks</td>
<td>0</td>
<td>4</td>
<td>2</td>
<td>5</td>
<td>2</td>
<td>3</td>
<td>8</td>
</tr>
<tr>
<td>Failing tests</td>
<td>1</td>
<td>20</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>17</td>
<td>1</td>
</tr>
<tr>
<td>Time to run</td>
<td>5.5s</td>
<td>4.8s</td>
<td>1.4s</td>
<td>1.2s‡</td>
<td>3.2s</td>
<td>2.2s</td>
<td>2.4s</td>
</tr>
<tr>
<td>Smoosh’s test suite (161 tests)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Failing tests</td>
<td>0</td>
<td>30 (35)</td>
<td>42</td>
<td>52</td>
<td>34</td>
<td>41</td>
<td>43 (39)</td>
</tr>
<tr>
<td>Time to run</td>
<td>23s</td>
<td>28s</td>
<td>1m13s</td>
<td>42s</td>
<td>21s</td>
<td>28s</td>
<td>29s</td>
</tr>
</tbody>
</table>

POSIX test suite (418 tests)

Failing tests 0 4 (8) 20 × 35 23 22 (23)
Time to run 12m41s 2m43s 2m43s × 2m52s 3m24s 2m45

Modernish’s shell diagnosis (91 potential bugs, 22 potential quirks) and test suite (312 tests)

Failing tests 1 16 2 3 3 14 1
Time to run 12m41s 2m43s 2m43s × 2m52s 3m24s 2m45

POSIX test suite (418 tests)

Failing tests 0 4 (8) 20 × 35 23 22 (23)
Time to run 12m41s 2m43s 2m43s × 2m52s 3m24s 2m45

Modernish’s shell diagnosis (91 potential bugs, 22 potential quirks) and test suite (312 tests)

Failing tests 1 16 2 3 3 14 1
Time to run 12m41s 2m43s 2m43s × 2m52s 3m24s 2m45

POSIX test suite (418 tests)

Failing tests 0 4 (8) 20 × 35 23 22 (23)
Time to run 12m41s 2m43s 2m43s × 2m52s 3m24s 2m45

Modernish’s shell diagnosis (91 potential bugs, 22 potential quirks) and test suite (312 tests)

Failing tests 1 16 2 3 3 14 1
Time to run 12m41s 2m43s 2m43s × 2m52s 3m24s 2m45

POSIX test suite (418 tests)

Failing tests 0 4 (8) 20 × 35 23 22 (23)
Time to run 12m41s 2m43s 2m43s × 2m52s 3m24s 2m45

Modernish’s shell diagnosis (91 potential bugs, 22 potential quirks) and test suite (312 tests)

Failing tests 1 16 2 3 3 14 1
Time to run 12m41s 2m43s 2m43s × 2m52s 3m24s 2m45

POSIX test suite (418 tests)

Failing tests 0 4 (8) 20 × 35 23 22 (23)
Time to run 12m41s 2m43s 2m43s × 2m52s 3m24s 2m45

Modernish’s shell diagnosis (91 potential bugs, 22 potential quirks) and test suite (312 tests)

Failing tests 1 16 2 3 3 14 1
Time to run 12m41s 2m43s 2m43s × 2m52s 3m24s 2m45

POSIX test suite (418 tests)

Failing tests 0 4 (8) 20 × 35 23 22 (23)
Time to run 12m41s 2m43s 2m43s × 2m52s 3m24s 2m45

Modernish’s shell diagnosis (91 potential bugs, 22 potential quirks) and test suite (312 tests)

Failing tests 1 16 2 3 3 14 1
Time to run 12m41s 2m43s 2m43s × 2m52s 3m24s 2m45

POSIX test suite (418 tests)

Failing tests 0 4 (8) 20 × 35 23 22 (23)
Time to run 12m41s 2m43s 2m43s × 2m52s 3m24s 2m45

Modernish’s shell diagnosis (91 potential bugs, 22 potential quirks) and test suite (312 tests)

Failing tests 1 16 2 3 3 14 1
Time to run 12m41s 2m43s 2m43s × 2m52s 3m24s 2m45

POSIX test suite (418 tests)

Failing tests 0 4 (8) 20 × 35 23 22 (23)
Time to run 12m41s 2m43s 2m43s × 2m52s 3m24s 2m45

Modernish’s shell diagnosis (91 potential bugs, 22 potential quirks) and test suite (312 tests)

Failing tests 1 16 2 3 3 14 1
Time to run 12m41s 2m43s 2m43s × 2m52s 3m24s 2m45

POSIX test suite (418 tests)

Failing tests 0 4 (8) 20 × 35 23 22 (23)
Time to run 12m41s 2m43s 2m43s × 2m52s 3m24s 2m45

Modernish’s shell diagnosis (91 potential bugs, 22 potential quirks) and test suite (312 tests)

Failing tests 1 16 2 3 3 14 1
Time to run 12m41s 2m43s 2m43s × 2m52s 3m24s 2m45

Modernish’s shell diagnosis (91 potential bugs, 22 potential quirks) and test suite (312 tests)

Failing tests 1 16 2 3 3 14 1
Time to run 12m41s 2m43s 2m43s × 2m52s 3m24s 2m45

POSIX test suite (418 tests)

Failing tests 0 4 (8) 20 × 35 23 22 (23)
Time to run 12m41s 2m43s 2m43s × 2m52s 3m24s 2m45

Modernish’s shell diagnosis (91 potential bugs, 22 potential quirks) and test suite (312 tests)

Failing tests 1 16 2 3 3 14 1
Time to run 12m41s 2m43s 2m43s × 2m52s 3m24s 2m45

POSIX test suite (418 tests)

Failing tests 0 4 (8) 20 × 35 23 22 (23)
Time to run 12m41s 2m43s 2m43s × 2m52s 3m24s 2m45

Modernish’s shell diagnosis (91 potential bugs, 22 potential quirks) and test suite (312 tests)

Failing tests 1 16 2 3 3 14 1
Time to run 12m41s 2m43s 2m43s × 2m52s 3m24s 2m45

MODERNISH’S SHELL DIAGNOSIS (91 POTENTIAL BUGS, 22 POTENTIAL QUIRKS) AND TEST SUITE (312 TESTS)

<table>
<thead>
<tr>
<th>Bugs</th>
<th>Quirks</th>
<th>Failing tests</th>
<th>Time to run</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>5.5s</td>
</tr>
<tr>
<td>16</td>
<td>4</td>
<td>20</td>
<td>4.8s</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>3</td>
<td>1.4s</td>
</tr>
<tr>
<td>3</td>
<td>5</td>
<td>3</td>
<td>3.2s</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>3</td>
<td>1.2s‡</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>17</td>
<td>2.2s</td>
</tr>
<tr>
<td>14</td>
<td>8</td>
<td>1</td>
<td>2.4s</td>
</tr>
</tbody>
</table>

Modernish’s test suite (161 tests)

<table>
<thead>
<tr>
<th>Bugs</th>
<th>Quirks</th>
<th>Failing tests</th>
<th>Time to run</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>23s</td>
</tr>
<tr>
<td>30</td>
<td>4</td>
<td>30</td>
<td>28s</td>
</tr>
<tr>
<td>42</td>
<td>2</td>
<td>42</td>
<td>1m13s</td>
</tr>
<tr>
<td>52</td>
<td>5</td>
<td>52</td>
<td>42s</td>
</tr>
<tr>
<td>34</td>
<td>2</td>
<td>34</td>
<td>21s</td>
</tr>
<tr>
<td>41</td>
<td>3</td>
<td>41</td>
<td>28s</td>
</tr>
<tr>
<td>43</td>
<td>8</td>
<td>43</td>
<td>29s</td>
</tr>
</tbody>
</table>
who cares about semantics?

- 10 bugs in the POSIX test suite
- 3 patches for dash
- dozens of other, uninvestigated bugs
POSIX test suite bug: impatient pipes

(trap "" PIPE

    sleep 2

    echo "it's alive" >out

) | true

[ $(cat out) = "it's alive" ]
POSIX test suite bug: impatient pipes

(trap "" PIPE
  sleep 2
  echo "it's alive" >out
) | true

[ $(cat out) = "it's alive" ]

If the pipeline is not in the background (see Asynchronous Lists), the shell shall wait for the last command specified in the pipeline to complete, and may also wait for all commands to complete.

IEEE Std 1003.1-2017 §2.9.2
dash bug: unset arithmetic

unset x

\[\$((x + 2)) -eq 2\]
dash bug: unset arithmetic

unset x

[ $((x + 2)) -eq 2 ]

dash: 1: Illegal number:
my motivation

“So, what do you do?”
my motivation

What happened?
smoosh

- ~11k loc  **shell semantics**  Lem, OCaml
- ~1k loc  **testing**  OCaml
- ~0.5 loc  **dash bindings**  C, OCaml
- ~5k loc  AST, OS model
- ~3k loc  expansion and evaluation
- ~2.5k loc  builtin commands

- **small-step** operational semantics
- **expansion** and **evaluation** are mutually recursive
- semantics uses an **abstract OS interface**
- **system** (real, working shell) or **symbolic** (fake OS)
- **1.5 person years** of effort

---