Quoting side effects

Chung-chieh Shan
Rutgers University
2007-10-13

Hi, this is your son’s school. We’re having some computer trouble.

Oh, dear – did he break something? In a way –

Did you really name your son Robert?; drop table Students;--?

Oh, yes. Little Bobby Tables, we call him.

Well, we’ve lost this year’s student records. I hope you’re happy.

And I hope you’ve learned to sanitize your database inputs.
Computational Linguistics
Computational Linguistics
Computational Linguistics
Outline

- Natural vs programming languages

  Side effects
  - State in programming languages
  - Control in programming languages
  - State in natural languages
  - Control in natural languages

  Quotation
  - Code generation
  - Mixed quotation

Together
How do natural languages work?

How should programming languages work?
How do natural languages work?
How do people learn to speak?

How should programming languages work?
How should computers be designed?
How do natural languages work?
How do people learn to speak?
How do people understand utterances?

How should programming languages work?
How should computers be designed?
How should computers run programs?
How do natural languages work?
How do people learn to speak?
How do people understand utterances?

Expression
‘Who is here?’

Interpret
Behavior in context
(look) (respond)

Compile
Meaning representation
\( \exists y. \text{here}(y) \land \text{at}(x, y) \)

How should programming languages work?
How should computers be designed?
How should computers run programs?

Expression
\( \text{print}(++x) \)

Interpret
Behavior in context
(increment) (print)

Compile
Meaning representation
\( \text{add } $1,%eax; \text{call } \ldots \)
Fact checking

Concrete syntax $\rightarrow$ Abstract syntax $\rightarrow$ Semantics $\rightarrow$ run-time support

Challenges: language and the world are ambiguous and complex.

Never mind the query language—what is the database schema?
Fact checking

Concrete syntax $\xrightarrow{\text{parse/ type-check}}$ Abstract syntax $\xrightarrow{\text{interpret/ compile}}$ Semantics $\xrightarrow{\text{run-time support}}$ $\cdots$

‘The politician used an epithet.’

$\exists e. \text{epithet}(e) \land \text{used}(p, e)$
Fact checking

Concrete syntax → Abstract syntax → Semantics → …

‘The politician used an epithet.’

S

NP

the politician

VP

TV

used

NP

an epithet

∃e. epithet(e) ∧ used(p, e)

Challenges: language and the world are ambiguous and complex.

Never mind the query language—what is the database schema?

‘The journalist knows which politician used an epithet.’
Fact checking

Concrete syntax → Abstract syntax → Semantics → run-time support

Challenges: language and the world are ambiguous and complex.

Never mind the query language—what is the database schema?

‘The journalist knows which politician used an epithet.’
Outline

Natural vs programming languages

➤ Side effects
   - State in programming languages
   - Control in programming languages
   - State in natural languages
   - Control in natural languages

Quotation
   - Code generation
   - Mixed quotation

Together
Unsound reasoning

A is better than B. B is better than C.

Therefore, A is better than C.
Unsound reasoning

The devil is better than nobody. Nobody is better than God.

Therefore, the devil is better than God.
Unsound reasoning

The devil is better than nobody. Nobody is better than God.

Therefore, the devil is better than God.

A is shorter than B. B is shorter than C.

Therefore, A is shorter than C.
Unsound reasoning

The devil is better than nobody. Nobody is better than God.

Therefore, the devil is better than God.

Alicia is shorter than her mom. Her mom is shorter than Alicia.

Therefore, Alicia is shorter than Alicia.
Unsound reasoning

The devil is better than nobody.  
Nobody is better than God.  

Therefore, the devil is better than God.

Alicia is shorter than her mom.  
Her mom is shorter than Alicia.  

Therefore, Alicia is shorter than Alicia.

\[
\begin{align*}
A &< B \\
B &< C \\
\hline
A &< C
\end{align*}
\]
Unsound reasoning

The devil is better than nobody. Nobody is better than God.

Therefore, the devil is better than God.

Alicia is shorter than her mom. Her mom is shorter than Alicia.

Therefore, Alicia is shorter than Alicia.

\[ x < -y \quad -y < x \]

\[ x < x \]
Unsound reasoning

The devil is better than nobody. Nobody is better than God.

Therefore, the devil is better than God.

Alicia is shorter than her mom. Her mom is shorter than Alicia.

Therefore, Alicia is shorter than Alicia.

\[
x < \quad --y \quad --y < x \quad \frac{x < x}{x < x}
\]

Side effects make substitution unsound.

In natural language:

nobody  her  who  anyone  know  the king of France  ...

In programming languages:

-- throw print open amb  ...
Operational semantics

\[ x < --y \land --y < x \]
Operational semantics

\[ x < --y \land --y < x \]

\[
\begin{array}{c|c|c}
\text{x} & \text{y} \\
1 & 2.5 \\
\end{array}
\]
Operational semantics

\[
\begin{array}{rcl}
x < --y && --y < x & \hspace{1cm} & x \\
1 < --y && --y < x & \hspace{1cm} & y
\end{array}
\]

\[
\begin{array}{c|c}
x & 1 \\
y & 2.5
\end{array}
\]

Why is state useful?
Operational semantics

\[
\begin{align*}
\text{x < \text{--y} \&\& \text{--y < x}} & \quad \begin{array}{|c|c|}
\hline
x & y \\
\hline
1 & 2.5 \\
\hline
\end{array} \\
1 < \text{--y} \&\& \text{--y < x} & \begin{array}{|c|c|}
\hline
x & y \\
\hline
1 & 2.5 \\
\hline
\end{array} \\
1 < 1.5 \&\& \text{--y < x} & \begin{array}{|c|c|}
\hline
x & y \\
\hline
1 & 1.5 \\
\hline
\end{array}
\end{align*}
\]
### Operational semantics

<table>
<thead>
<tr>
<th>Condition</th>
<th>x</th>
<th>y</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x &lt; --y$ &amp;&amp; $--y &lt; x$</td>
<td>1</td>
<td>2.5</td>
</tr>
<tr>
<td>$1 &lt; --y$ &amp;&amp; $--y &lt; x$</td>
<td>1</td>
<td>2.5</td>
</tr>
<tr>
<td>$1 &lt; 1.5$ &amp;&amp; $--y &lt; x$</td>
<td>1</td>
<td>1.5</td>
</tr>
<tr>
<td>$--y &lt; x$</td>
<td>1</td>
<td>1.5</td>
</tr>
</tbody>
</table>

Why is state useful?
### Operational semantics

<table>
<thead>
<tr>
<th>Condition</th>
<th>x</th>
<th>y</th>
</tr>
</thead>
<tbody>
<tr>
<td>x &lt; --y &amp;&amp; --y &lt; x</td>
<td>1</td>
<td>2.5</td>
</tr>
<tr>
<td>1 &lt; --y &amp;&amp; --y &lt; x</td>
<td>1</td>
<td>2.5</td>
</tr>
<tr>
<td>1 &lt; 1.5 &amp;&amp; --y &lt; x</td>
<td>1</td>
<td>1.5</td>
</tr>
<tr>
<td>--y &lt; x</td>
<td>1</td>
<td>1.5</td>
</tr>
<tr>
<td>0.5 &lt; x</td>
<td>1</td>
<td>0.5</td>
</tr>
</tbody>
</table>

Why is state useful?
Operational semantics

<table>
<thead>
<tr>
<th>Condition</th>
<th>x</th>
<th>y</th>
</tr>
</thead>
<tbody>
<tr>
<td>( x &lt; \neg\neg y \land \neg\neg y &lt; x )</td>
<td>1</td>
<td>2.5</td>
</tr>
<tr>
<td>( 1 &lt; \neg\neg y \land \neg\neg y &lt; x )</td>
<td>1</td>
<td>2.5</td>
</tr>
<tr>
<td>( 1 &lt; 1.5 \land \neg\neg y &lt; x )</td>
<td>1</td>
<td>1.5</td>
</tr>
<tr>
<td>( \neg\neg y &lt; x )</td>
<td>1</td>
<td>1.5</td>
</tr>
<tr>
<td>( 0.5 &lt; x )</td>
<td>1</td>
<td>0.5</td>
</tr>
<tr>
<td>( 0.5 &lt; 1 )</td>
<td>1</td>
<td>0.5</td>
</tr>
</tbody>
</table>

Why is state useful?
Operational semantics

\[
\begin{array}{c|c|c}
\hspace{1cm} x < --y \land --y < x \hspace{1cm} & \hspace{1cm} 1 \hspace{1cm} & \hspace{1cm} 2.5 \\
1 < --y \land --y < x & 1 & 2.5 \\
1 < 1.5 \land --y < x & 1 & 1.5 \\
--y < x & 1 & 1.5 \\
0.5 < x & 1 & 0.5 \\
0.5 < 1 & 1 & 0.5 \\
\text{true} & 1 & 0.5 \\
\end{array}
\]

Why is state useful?
Manipulating the context

\[
every(4,6,9) \% \text{some}(2,3) = 0
\]
Manipulating the context

\[\text{every}(4, 6, 9) \mod \text{some}(2, 3) = 0\]

\[\left[4 \mod \text{some}(2, 3) = 0\right] \& \&
\left[6 \mod \text{some}(2, 3) = 0\right] \& \&
\left[9 \mod \text{some}(2, 3) = 0\right]\]
Manipulating the context

every(4,6,9) % some(2,3) == 0

[4 % some(2,3) == 0] &&
[6 % some(2,3) == 0] &&
[9 % some(2,3) == 0]

[4 % 2 == 0 || 4 % 3 == 0] &&
[6 % some(2,3) == 0] &&
[9 % some(2,3) == 0]
Manipulating the context

\[
every(4, 6, 9) \% \text{some}(2, 3) = 0
\]

\[
[4 \% \text{some}(2, 3) = 0] \&\&
\]

\[
[6 \% \text{some}(2, 3) = 0] \&\&
\]

\[
[9 \% \text{some}(2, 3) = 0]
\]

\[
[4 \% 2 = 0 |\| 4 \% 3 = 0] \&\&
\]

\[
[6 \% \text{some}(2, 3) = 0] \&\&
\]

\[
[9 \% \text{some}(2, 3) = 0]
\]

\[
[6 \% \text{some}(2, 3) = 0] \&\&
\]

\[
[9 \% \text{some}(2, 3) = 0]
\]
Manipulating the context

every(4,6,9) % some(2,3) == 0

[4 % some(2,3) == 0] &&
[6 % some(2,3) == 0] &&
[9 % some(2,3) == 0]

[4 % 2 == 0 || 4 % 3 == 0] &&
[6 % some(2,3) == 0] &&
[9 % some(2,3) == 0]

[6 % some(2,3) == 0] &&
[9 % some(2,3) == 0]

[6 % 2 == 0 || 6 % 3 == 0] &&
[9 % some(2,3) == 0]
Manipulating the context

```latex
\text{every}(4,6,9) \% \text{some}(2,3) == 0

[4 \% \text{some}(2,3) == 0] &&
[6 \% \text{some}(2,3) == 0] &&
[9 \% \text{some}(2,3) == 0]

[4 \% 2 == 0 || 4 \% 3 == 0] &&
[6 \% \text{some}(2,3) == 0] &&
[9 \% \text{some}(2,3) == 0]

[6 \% \text{some}(2,3) == 0] &&
[9 \% \text{some}(2,3) == 0]

[6 \% 2 == 0 || 6 \% 3 == 0] &&
[9 \% \text{some}(2,3) == 0]

...
Anaphora

Alicia is shorter than her mom
and her mom is shorter than Alicia
Anaphora

Alicia is shorter than her mom
   and her mom is shorter than Alicia
Alicia is shorter than her mom
   and her mom is shorter than Alicia
   Alicia

More sophisticated theory of discourse referents
Anaphora

Alicia is shorter than her mom and her mom is shorter than Alicia

Alicia is shorter than her mom and her mom is shorter than Alicia

Alicia is shorter than Alicia’s mom and her mom is shorter than Alicia
Anaphora

Alicia is shorter than her mom
    and her mom is shorter than Alicia

Alicia is shorter than her mom
    and her mom is shorter than Alicia Alicia

Alicia is shorter than Alicia’s mom
    and her mom is shorter than Alicia Alicia

Alicia is shorter than Beatrice
    and her mom is shorter than Alicia Beatrice

Clara is shorter than Alicia
    Clara

More sophisticated theory of discourse referents
Anaphora

Alicia is shorter than her mom and her mom is shorter than Alicia
Alicia is shorter than her mom and her mom is shorter than Alicia
Alicia is shorter than Alicia’s mom and her mom is shorter than Alicia
Alicia is shorter than Beatrice and her mom is shorter than Alicia
her mom is shorter than Alicia
Anaphora

Alicia is shorter than her mom
    and her mom is shorter than Alicia
Alicia is shorter than her mom
    and her mom is shorter than Alicia Alicia
Alicia is shorter than Alicia’s mom
    and her mom is shorter than Alicia Alicia
Alicia is shorter than Beatrice
    and her mom is shorter than Alicia Beatrice
her mom is shorter than Alicia
Beatrice’s mom is shorter than Alicia

Beatrice Alicia

Clara is shorter than Alicia Clara
Beatrice Alicia
Anaphora

Alicia is shorter than her mom
   and her mom is shorter than Alicia

Alicia is shorter than her mom
   and her mom is shorter than Alicia Alicia

Alicia is shorter than Alicia’s mom
   and her mom is shorter than Alicia Alicia

Alicia is shorter than Beatrice
   and her mom is shorter than Alicia Beatrice
   Alicia

her mom is shorter than Alicia Beatrice Alicia

Beatrice’s mom is shorter than Alicia Beatrice Alicia

Clara is shorter than Alicia Clara Beatrice Alicia
Anaphora

Alicia is shorter than her mom
and her mom is shorter than Alicia

Alicia is shorter than her mom
and her mom is shorter than Alicia

Alicia is shorter than Alicia’s mom
and her mom is shorter than Alicia

Alicia is shorter than Beatrice
and her mom is shorter than Alicia
her mom is shorter than Alicia
Beatrice’s mom is shorter than Alicia

Clara is shorter than Alicia
▶ true

More sophisticated theory of discourse referents
the devil is better than nobody
Quantification

the devil is better than nobody

the devil is not better than Alicia
and the devil is not better than Beatrice
and the devil is not better than Clara
Quantification

the devil is better than nobody

the devil is not better than Alicia
and the devil is not better than Beatrice
and the devil is not better than Clara

the devil is not better than Beatrice
and the devil is not better than Clara
the devil is better than nobody

the devil is not better than Alicia
    and the devil is not better than Beatrice
    and the devil is not better than Clara

the devil is not better than Beatrice
    and the devil is not better than Clara

the devil is not better than Clara
Quantification

the devil is better than nobody
the devil is not better than Alicia
  and the devil is not better than Beatrice
  and the devil is not better than Clara
the devil is not better than Beatrice
  and the devil is not better than Clara
the devil is not better than Clara
▶ true

In-situ quantifiers
Outline

Natural vs programming languages

Side effects
  State in programming languages
  Control in programming languages
  State in natural languages
  Control in natural languages

▶ Quotation
  Code generation
  Mixed quotation

Together
A recurring situation

render (scene, lighting)
A recurring situation

<table>
<thead>
<tr>
<th>Program</th>
<th>Static input</th>
<th>Dynamic input</th>
</tr>
</thead>
<tbody>
<tr>
<td>render</td>
<td>(scene, lighting)</td>
<td></td>
</tr>
<tr>
<td>power</td>
<td>(exponent, base)</td>
<td></td>
</tr>
<tr>
<td>parse</td>
<td>(grammar, string)</td>
<td></td>
</tr>
<tr>
<td>invert</td>
<td>(size, matrix)</td>
<td></td>
</tr>
<tr>
<td>compile</td>
<td>(headers, source)</td>
<td></td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
A recurring situation

<table>
<thead>
<tr>
<th>Specialized program</th>
<th>Dynamic input</th>
</tr>
</thead>
<tbody>
<tr>
<td>render_robot</td>
<td>(lighting)</td>
</tr>
<tr>
<td>power_12</td>
<td>(base)</td>
</tr>
<tr>
<td>parse_java</td>
<td>(string)</td>
</tr>
<tr>
<td>invert_16</td>
<td>(matrix)</td>
</tr>
<tr>
<td>compile_gui</td>
<td>(source)</td>
</tr>
</tbody>
</table>

...
A recurring situation

<table>
<thead>
<tr>
<th>Cogen</th>
<th>Static input</th>
<th>Dynamic input</th>
</tr>
</thead>
<tbody>
<tr>
<td>renderGen</td>
<td>(scene)</td>
<td>(lighting)</td>
</tr>
<tr>
<td>powerGen</td>
<td>(exponent)</td>
<td>(base)</td>
</tr>
<tr>
<td>parseGen</td>
<td>(grammar)</td>
<td>(string)</td>
</tr>
<tr>
<td>invertGen</td>
<td>(size)</td>
<td>(matrix)</td>
</tr>
<tr>
<td>compileGen</td>
<td>(headers)</td>
<td>(source)</td>
</tr>
</tbody>
</table>

...
Quotation helps write cogens

\[
power (0, x) = 1 \\
power (n, x) = x \times power (n-1, x) \\
\]

\[
power 12 2 \quad \Rightarrow 4096
\]
Quotation helps write cogens

\[
\begin{align*}
\text{power} (0, x) &= 1 \\
\text{power} (n, x) &= x \times \text{power} (n-1, x) \\
\end{align*}
\]

\[
\begin{align*}
\text{power} 12 \ 2 &\quad \Rightarrow 4096 \\
\end{align*}
\]

\[
\begin{align*}
\text{powerGen} (0, x) &= \langle 1 \rangle \\
\text{powerGen} (n, x) &= \langle \neg x \times \neg (\text{powerGen} (n-1, x)) \rangle \\
\end{align*}
\]
Quotation helps write cogens

\[
\begin{align*}
\text{power} \ (0, \ x) &= 1 \\
\text{power} \ (n, \ x) &= x \ast \text{power} \ (n-1, \ x) \\
\text{power} \ 12 \ 2 & \Rightarrow 4096
\end{align*}
\]

\[
\begin{align*}
\text{powerGen} \ (0, \ x) &= \langle 1 \rangle \\
\text{powerGen} \ (n, \ x) &= \langle \neg x \ast \neg (\text{powerGen} \ (n-1, \ x)) \rangle \\
\langle \text{fun} \ x \rightarrow \neg (\text{powerGen} \ (12, \ \langle x \rangle)) \rangle & \Rightarrow \langle \text{fun} \ x \rightarrow x \ast x \ast x \ast x \ast x \ast x \ast x \ast x \ast x \ast x \ast x \ast 1 \rangle
\end{align*}
\]
I am sorry to have used an ‘epithet’.
I am sorry to have used an ‘epithet’.

Quine said that quotation ‘has a certain anomalous feature’.  
(Davidson 1979)

Bush also said his administration would ‘achieve our objectives’ 
in Iraq.  
Outline

Natural vs programming languages

Side effects
- State in programming languages
- Control in programming languages
- State in natural languages
- Control in natural languages

Quotation
- Code generation
- Mixed quotation

► Together
State for counting operations

\[ \text{count} = 0 \]

\[
\text{powerGen} (0, x) = \langle 1 \rangle\\
\text{powerGen} (n, x) = \text{++count}; \langle \neg x * \neg (\text{powerGen} (n-1, x)) \rangle
\]
Code generation with side effects

State for counting operations

count = 0

count

t = 0

t

taxGen (0, x) = 〈1〉
taxGen (n, x) = ++count; 〈~x * ~(taxGen (n-1, x))〉

Control for generating variable bindings

〈fun x -> ~(taxGen (4, taxGen (3, 〈x〉)))〉
▶ 〈fun x -> (x*x*x*1)*(x*x*x*1)*(x*x*x*1)*(x*x*x*1)*1〉
State for counting operations

count = 0

\[
\text{powerGen (0, x)} = \langle 1 \rangle \\
\text{powerGen (n, x)} = \text{++count}; \langle \sim x \ast \sim (\text{powerGen (n-1, x)}) \rangle
\]

Control for generating variable bindings

\[
\langle \text{fun x ->} \sim (\text{powerGen (4, powerGen (3, \langle x \rangle)})) \rangle \\
\langle \text{fun x -> let y = x\ast x\ast x\ast 1 in} \sim (\text{powerGen (4, \langle y \rangle)}) \rangle
\]
Code generation with side effects

State for counting operations

```plaintext
count = 0

powerGen (0, x) = [1]
powerGen (n, x) = ++count; [~x * ~(powerGen (n-1, x))]
```

Control for generating variable bindings

```plaintext
<fun x -> ~(powerGen (4, powerGen (3, [x])))>
<fun x -> let y = x*x*x*1 in ~(powerGen (4, [y]))>
<fun x -> let y = x*x*x*1 in let z = y*y*y*y*1 in ~(z)>
```

Also, generating code with side effects
State for counting operations

\[
\text{count} = 0
\]

\[
\text{powerGen} (0, x) = \langle 1 \rangle
\]

\[
\text{powerGen} (n, x) = ++\text{count}; \langle \neg x \ast \neg (\text{powerGen} (n-1, x)) \rangle
\]

Control for generating variable bindings

\[
\langle \text{fun x} \rightarrow \neg (\text{powerGen} (4, \text{powerGen} (3, \langle x \rangle))) \rangle
\]

\[
\langle \text{fun x} \rightarrow \text{let } y = x\ast x\ast x\ast 1 \text{ in } \neg (\text{powerGen} (4, \langle y \rangle)) \rangle
\]

\[
\langle \text{fun x} \rightarrow \text{let } y = x\ast x\ast x\ast 1 \text{ in let } z = y\ast y\ast y\ast y\ast 1 \text{ in } \neg \langle z \rangle \rangle
\]

▶ \[
\langle \text{fun x} \rightarrow \text{let } y = x\ast x\ast x\ast 1 \text{ in let } z = y\ast y\ast y\ast y\ast 1 \text{ in } z \rangle
\]
Code generation with side effects

State for counting operations

```plaintext
count = 0

powerGen (0, x) = 〈1〉
powerGen (n, x) = ++count; 〈~x * ~(powerGen (n-1, x))〉
```

Control for generating variable bindings

```plaintext
〈fun x -> ~(powerGen (4, powerGen (3, 〈x〉))))〉
〈fun x -> let y = x*x*x*1 in ~(powerGen (4, 〈y〉)))〉
〈fun x -> let y = x*x*x*1 in let z = y*y*y*y*1 in ~(z)〉
▷ 〈fun x -> let y = x*x*x*1 in let z = y*y*y*y*1 in z〉
```

Also, generating code with side effects
Mixed quotation with side effects

Anaphora and mixed quotation

Bush also said his administration would ‘achieve our objectives’ in Iraq, but what are they precisely?
Mixed quotation with side effects

Anaphora and mixed quotation

Bush also said his administration would ‘achieve our objectives’ in Iraq, but what are they precisely?

Quantification and mixed quotation

Someone is shorter than everyone.  (scope ambiguity)
Anaphora and mixed quotation

Bush also said his administration would ‘achieve our objectives’ in Iraq, but what are they precisely?

Quantification and mixed quotation

Someone is shorter than everyone. (scope ambiguity)
‘Someone is shorter than [everyone].’ (code generation)
Mixed quotation with side effects

Anaphora and mixed quotation

Bush also said his administration would ‘achieve our objectives’ in Iraq, but what are they precisely?

Quantification and mixed quotation

Someone is shorter than everyone. (scope ambiguity)
‘Someone is shorter than [everyone].’ (code generation)
Nobody is shorter than anybody. (polarity sensitivity)
Mixed quotation with side effects

Anaphora and mixed quotation

Bush also said his administration would ‘achieve our objectives’ in Iraq, but what are they precisely?

Quantification and mixed quotation

Someone is shorter than everyone. (scope ambiguity)
‘Someone is shorter than [everyone].’ (code generation)

Nobody is shorter than anybody. (polarity sensitivity)

× Anybody is shorter than nobody. (no inverse scope)
Mixed quotation with side effects

Anaphora and mixed quotation

Bush also said his administration would ‘achieve our objectives’ in Iraq, but what are they precisely?

Quantification and mixed quotation

Someone is shorter than everyone.  
‘Someone is shorter than [everyone].’  
Nobody is shorter than anybody.  
× Anybody is shorter than nobody.  
× ‘Anybody is shorter than [nobody].’
Conclusion

Natural languages

Programming languages

Side effects

• State

• Control

Quotation

her

nobody

use an ‘epithet’

—y

some(2,3)

〈

fun x -> x*x*x*1

〉
## Conclusion

<table>
<thead>
<tr>
<th></th>
<th>Natural languages</th>
<th>Programming languages</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Side effects</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>- State</td>
<td></td>
<td></td>
</tr>
<tr>
<td>- Control</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Quotation</strong></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Conclusion

<table>
<thead>
<tr>
<th>Side effects</th>
<th>Natural languages</th>
<th>Programming languages</th>
</tr>
</thead>
<tbody>
<tr>
<td>State</td>
<td>her</td>
<td>--y</td>
</tr>
<tr>
<td>Control</td>
<td>nobody</td>
<td>some(2,3)</td>
</tr>
</tbody>
</table>

| Quotation | use an ‘epithet’ | (fun x -> x*x*x*1)   |
## Conclusion

<table>
<thead>
<tr>
<th>Side effects</th>
<th>Natural languages</th>
<th>Programming languages</th>
</tr>
</thead>
<tbody>
<tr>
<td>State</td>
<td>her</td>
<td>--y</td>
</tr>
<tr>
<td>Control</td>
<td>nobody</td>
<td>some(2, 3)</td>
</tr>
</tbody>
</table>

| Quotation     | use an ‘epithet’  | \(<\text{fun } x \rightarrow x*x*x*1>\) |
## Conclusion

<table>
<thead>
<tr>
<th>Side effects</th>
<th>Natural languages</th>
<th>Programming languages</th>
</tr>
</thead>
<tbody>
<tr>
<td>• State</td>
<td>her</td>
<td>--y</td>
</tr>
<tr>
<td>• Control</td>
<td>nobody</td>
<td>some(2,3)</td>
</tr>
</tbody>
</table>

Quotation

use an ‘epithet’ ←→ \( \langle \text{fun } x \rightarrow x^3*1 \rangle \)