Automatically Computing Functional Instantiations

J Strother Moore
Department of Computer Sciences
University of Texas at Austin
(defstub g (x y) t)

(defun mapg (x ans)
  (if (endp x)
      ans
      (mapg (cdr x)
            (g (car x) ans)))))

(defthm mapg-append
  (equal (mapg (append u v) ans)
         (mapg v (mapg u ans))))
(DEFUN BIG-INTS (X MIN A)
  (COND
    ((CONSP X)
     (COND ((AND (INTEGERP (CAR X))
               (>= (CAR X) MIN))
            (BIG-INTS (CDR X)
                     MIN
                     (CONS (CAR X) A))))
     (T (BIG-INTS (CDR X) MIN A))))
  (T A)))
(defthm main-old
  (equal (big-ints (append b c) min a)
         (big-ints c min
                 (big-ints b min a)))
  :hints ("Goal" :use
            (:instance
              (:functional-instance mapg-append
                (mapg (lambda (x ans)
                       (big-ints x min ans)))
                    (g (lambda (x y)
                    (if (integerp x)
                        (if (< x min) y (cons x y))
                        y)))
                    (ans a) (u b) (v c) (min min)))))
(defthm main-new
   (EQUAL (BIG-INTS (APPEND B C) MIN A)
          (BIG-INTS C MIN
                (BIG-INTS B MIN A)))
   :hints ("Goal" :consider mapg-append))
Implementation

See:

- `books/huet-lang-algorithm.lisp`,
- `books/consider-hint.lisp`, and
- `books/consider-hint-tests.lisp`. 
Issues

• second-order matching
  
  \((g \ (\text{CAR} \ x) \ \text{ans}) \ \text{versus} \ \)

  \((\text{CONS} \ (\text{CAR} \ B) \ A)\)
Issues

• second-order matching
  \[(g \text{ (CAR } x\text{) ans}) \text{ versus } (\text{CONS} \text{ (CAR } B\text{) A})\]

• rearranging terms
  \[(\text{IF } \alpha \text{ (h } \beta\text{) (h } \gamma\text{)) versus } (h \text{ (IF } \alpha \text{ } \beta\text{ } \gamma\text{))}\]
Issues

• diving into definitions
  the body of mapg versus
  the body of BIG-INTS
Issues

• diving into definitions
  the body of mapg *versus*
  the body of BIG-INTS

• selecting among myriad choices
Issues

• selecting among myriad choices

\[(h \ x)/\sigma \text{ is } (\text{CAR } \text{ (CDR } A)) \text{ when} \]

\[\sigma \Rightarrow \{h \leftarrow (\lambda (Z) \text{ (CAR } \text{ (CDR } A)))\}\]
Issues

• selecting among myriad choices

\[(h \, x)/\sigma \text{ is } (\text{CAR} \, (\text{CDR} \, A)) \text{ when} \]

\[\sigma = \{h \leftarrow (\lambda \, (Z) \, (\text{CAR} \, (\text{CDR} \, A)))\}\]

\[\sigma = \{h \leftarrow (\lambda \, (Z) \, (\text{CAR} \, (\text{CDR} \, Z))), \, x \leftarrow A\}\]
Issues

- selecting among myriad choices

\[(h \ x)/\sigma \text{ is } (\text{CAR } (\text{CDR } A)) \text{ when} \]

\[\sigma = \{h \leftarrow (\lambda (Z) (\text{CAR } (\text{CDR } A)))\}\]

\[\sigma = \{h \leftarrow (\lambda (Z) (\text{CAR } (\text{CDR } Z))), x \leftarrow A\}\]

\[\sigma = \{h \leftarrow (\lambda (Z) (\text{CAR } Z)), x \leftarrow (\text{CDR } A)\}\]
Issues

• selecting among myriad choices

\((h \ x)/\sigma \text{ is } (\text{CAR } (\text{CDR } A)) \text{ when }\)

\(\sigma =\{h \leftarrow (\lambda \ (Z) \ (\text{CAR } (\text{CDR } A)))\}\)

\(\sigma =\{h \leftarrow (\lambda \ (Z) \ (\text{CAR } (\text{CDR } Z))), \ x \leftarrow A\}\)

\(\sigma =\{h \leftarrow (\lambda \ (Z) \ (\text{CAR } Z)), \ x \leftarrow (\text{CDR } A)\}\)

\(\sigma =\{h \leftarrow (\lambda \ (Z) \ Z), \ x \leftarrow (\text{CAR } (\text{CDR } A))\}\)
Summary

• Use Huet-Lang matching, starting from a (possibly empty) seed substitution to limit possible candidates and rewriting “in all possible ways” to deal with minor variants,

• extend each plausible substitution by diving into corresponding definitions,
• rank the plausible substitutions, and
• generate an OR hint that considers each of the highest ranking substitutions.
The Huet-Lang Theorem

Let $t$ be a second-order term and $s$ be a term. A (second-order) substitution $\sigma$ such that $t/\sigma = s$ exists iff $\{< t, s >\} \Rightarrow^* \emptyset$, where each “$\Rightarrow$” step is one of the following five possibilities:
Identity

\[ \{<s, s>\} \cup E \Rightarrow E \]
Binding

\( \{ < v, s > \} \cup E \Rightarrow E/\{ v := s \} \),

where \( v \) is an individual (first-order) variable symbol
Simplification

\[ \{< (F \ t_1 \ldots \ t_n), \ (F \ s_1 \ldots \ s_n) >\} \cup E \]
\[ \Rightarrow \{< t_1, s_1 >, \ldots, < t_n, s_n >\} \cup E \]
Projection

\[ E \Rightarrow E / \{ f := (\lambda (v_1 \ldots v_n) \, v_i) \} \]

if one of the elements of \( E \) is

\[ < (f \, t_1 \ldots t_n), s > \]

where \( f \) is a constrained function symbol
Imitation

\[ E \Rightarrow E/\{ f := (\lambda (v_1 \ldots v_n) \\
(F (h_1 v_1 \ldots v_n) \\
\ldots \\
(h_m v_1 \ldots v_n)) )) \}, \]

if \( E \) contains

\[ < (f \ t_1 \ldots t_n), (F \ s_1 \ldots s_m) > \]

where \( f \) is a constrained function symbol
and the \( h_i \) are new constrained function symbols
The Algorithm

To match $t$ to $s$, try all possible combinations of \( \Rightarrow \) steps to reduce \( \{< t, s > \} \) to the empty set, collecting as you go every substitution pair \( \alpha := \beta \).
**Example**

\[(g \ x \ y) \ vsus (\text{CAR} \ (\text{CDR} \ A))\]

- **Projection:** \(g \leftarrow (\lambda \ (u \ v) \ u)\)
  \(\Rightarrow\)
  match \(x\) with \((\text{CAR} \ (\text{CDR} \ A))\)

- **Projection:** \(g \leftarrow (\lambda \ (u \ v) \ v)\)
  \(\Rightarrow\)
  match \(y\) with \((\text{CAR} \ (\text{CDR} \ A))\)
Example

\((g \ x \ y) \text{ versus } (\text{CAR} \ (\text{CDR} \ A))\)

- **Imitation**: \(g \leftarrow (\lambda \ (u \ v)\)
  
  \((\text{CAR} \ (h_1 \ u \ v)))\)

\(\Rightarrow\)

match \((\text{CAR} \ (h_1 \ u \ v))\) with

\((\text{CAR} \ (\text{CDR} \ A))\)

\(\Rightarrow\)

**Simplification**

match \((h_1 \ u \ v)\) with \((\text{CDR} \ A)\)
Example

\[(g \ x \ y) \textit{ versus } (\text{CAR} \ (\text{CDR} \ A))\]

\[
\{ \quad x \leftarrow (\text{CDR} \ A), \quad g \leftarrow (\lambda \ U \ V \ (\text{CAR} \ U)) \} \\
\{ \quad x \leftarrow A, \quad g \leftarrow (\lambda \ U \ V \ (\text{CAR} \ (\text{CDR} \ U))) \} \\
\{ \quad y \leftarrow A, \quad g \leftarrow (\lambda \ U \ V \ (\text{CAR} \ (\text{CDR} \ V))) \} \\
\{ \quad y \leftarrow (\text{CDR} \ A), \quad g \leftarrow (\lambda \ U \ V \ (\text{CAR} \ V)) \} \\
\{ \quad x \leftarrow (\text{CAR} \ (\text{CDR} \ A)), \quad g \leftarrow (\lambda \ U \ V \ U) \} \\
\{ \quad y \leftarrow (\text{CAR} \ (\text{CDR} \ A)), \quad g \leftarrow (\lambda \ U \ V \ V) \} \\
\{ \quad g \leftarrow (\lambda \ U \ V \ (\text{CAR} \ (\text{CDR} \ A))) \} \]
Rewriting

Rewrite the ground term in all possible ways using a small set of rules and then use conventional Huet-Lang with each variant. The rewriting is done incrementally on the fly and we quit as soon as we discover a variant that matches.
Sample Rules

(IF x (F y v) (F z v))
=  ; rewrites to
(F (IF x y z) v)

(IF x (F v1 v2) (F w1 w2))
=  
(F (IF x v1 w1) (IF x v2 w2))

...
(ENDP x)
=
(NOT (CONSP x))

(:META FOLD-TO-ISOLATE)
Fold to Isolate

\[(\text{IF } \tau (\phi (F \alpha_1 \alpha_2)) (\psi (F \beta_1 \beta_2)))\]

= \\

\[((\lambda (Z) \\
(\text{IF } \tau (\phi Z)(\psi Z))) \\
(F (\text{IF } \tau \alpha_1 \beta_1)) \\
(\text{IF } \tau \alpha_2 \beta_2)))\]
Diving Through Defuns

(mapg x ans) versus
(BIG-INTS X MIN A):

mapg ← (λ (X ANS)
    (BIG-INTS X MIN ANS))
Diving Through Defuns

(mapg x ans) *versus*
(BIG-INTS B MIN A):

mapg ← (λ (X ANS)
        (BIG-INTS X MIN ANS))

(defun mapg (x ans)
  (if (endp x)
      ans
      ans
      (mapg (cdr x)
            (mapg (car x) ans))))
Diving Through Defuns

(mapg x ans) versus (BIG-INTS B MIN A):

mapg ← (λ (X ANS)
           (BIG-INTS X MIN ANS))

g ← ???

(defun mapg (x ans)
  (if (endp x)
      ans
      (mapg (cdr x)
            (mapg (car x) ans)))))
(IF (ENDP x)
   ans
   (mapg (CDR x)
      (g (CAR x) ans))))

versus

(IF (CONSP X)
   (IF (AND (INTEGERP (CAR X))
          (>= (CAR X) MIN))
      (BIG-INTS (CDR X)
          MIN
          (CONS (CAR X) A))
      (BIG-INTS (CDR X) MIN A))
   A)
(IF (ENDP x)
  ans
  (mapg (CDR x)
    (g (CAR x) ans)))

versus

(IF (ENDP X)
  A
  (IF (AND (INTEGERP (CAR X))
    (>= (CAR X) MIN))
    (BIG-INTS (CDR X)
      MIN
      (CONS (CAR X) A))
    (BIG-INTS (CDR X) MIN A)))
(IF (ENDP x)
    ans
    (mapg (CDR x)
      (g (CAR x) ans)))))

versus

(IF (ENDP X)
    A
    (BIG-INTS (CDR X)
      MIN
      (IF (AND (INTEGERP (CAR X))
        (>= (CAR X) MIN))
        (CONS (CAR X) A)
        A))))
mapg ← (\( \lambda \) (X ANS) 
     (BIG-INTS X MIN ANS))

g ← (\( \lambda \) (X Y) 
     (IF (INTEGERP X) 
       (IF (< X MIN) Y (CONS X Y)) 
       Y))
Selecting Among Myriad Choices

Each workable substitution is heuristically scored.

\((g \ x \ y) \text{ versus } (\text{CAR}\ (\text{CDR}\ A))\)
(g x y) versus (CAR (CDR A))

((19/6 (X . (CDR A))
  (G . (LAMBDA (X Y) (CAR X))))
(19/6 (X . A)
  (G . (LAMBDA (X Y) (CAR (CDR X))))))
(19/6 (Y . A)
  (G . (LAMBDA (X Y) (CAR (CDR Y))))))
(19/6 (Y . (CDR A))
  (G . (LAMBDA (X Y) (CAR Y))))
\[
\begin{align*}
(8/3 & (X . \ (\text{CAR} \ (\text{CDR} \ A)))) \\
& (G . \ (\text{LAMBDA} \ (X \ Y) \ X))) \\
(8/3 & (Y . \ (\text{CAR} \ (\text{CDR} \ A)))) \\
& (G . \ (\text{LAMBDA} \ (X \ Y) \ Y))) \\
(7/3 & (G . \ (\text{LAMBDA} \ (X \ Y) \ (\text{CAR} \ (\text{CDR} \ A))))))
\end{align*}
\]
Conclusion

This is a good topic for a student project, possibly including a dissertation.

• Clean up the code in consider-hint and huet-lang-algorithm.

• Explore less explosive ways to consider “all possible rewrites,” including modern
matching algorithms that consider equational theories.

- Investigate other ways to produce fewer plausible substitutions.

- Investigate ways to take into account the actual constraints on the function symbols being instantiated.