SIM, An S-Expression Pattern-Matching Function

ABSTRACT

This document describes SIM, a LISP 2 pattern-matching function that is used for analyzing S-expressions. A basic knowledge of LISP 2 Intermediate Language is assumed.

1. PURPOSE
SIM is a LISP 2 pattern-matching function\(^1\) that is used for analyzing S-expressions.

2. DESCRIPTION
SIM has two arguments: a pattern written in a Backus Normal Form-like language for S-expressions and a sample. The sample is matched to the pattern using the rules given below. The value of SIM is Boolean. If the sample and the pattern match, the value is true; otherwise the value is false. No reconstruction capability is included as with other similar pattern matchers (e.g., CONVERT, FLIP, etc.).

3. USAGE
SIM was originally designed as an aid for implementing the LISP 2 compiler; however, uses have been found in several other areas of LISP programming. Therefore, SIM is included as a primitive in the LISP 2 system.

3.1 FUNCTION FORMAT
The SIM function declaration has the form\(^2\):

\[
\text{(FUNCTION((SIM . LISP) BOOLEAN)((PATTERN SYMBOL) (SAMPLE SYMBOL)))}
\]

where PATTERN is any legal pat (see Section 3.2) and SAMPLE is any S-expression.

---

\(^1\)Nonprimitive syntactic entities are italicized. If compound words are used to denote the entities, the words are, in most cases, joined by italicized colons.

\(^2\)This and all succeeding examples are written in LISP 2 Intermediate Language.
3.2 LEGAL PATTERNS

A legal pattern is either a simple pattern or is defined in terms of simple patterns. In the equation below, a legal pattern, \( \text{pat} \), is defined in terms of simple patterns, \( \text{sp} \), and other legal patterns. The following BNF-like equation defines the legal construction rules for patterns:

\[
\text{pat} = \text{sp} | \text{CONS}(\text{pat},\text{pat}) | \text{APPEND}(\text{pat},\text{pat})
\]

The following are legal simple patterns (\( \text{sp} \)):

- **A.** This \( \text{sp} \) matches any atom.
- **N.** This \( \text{sp} \) matches any number.
- **S.** This \( \text{sp} \) matches any \( S \)-expression.
- **L.** This \( \text{sp} \) matches any nonatomic \( S \)-expression.
- **ID.** This \( \text{sp} \) matches any identifier.
- **V.** This \( \text{sp} \) matches any variable, i.e., an \( ID \), or \( (ID \ldots ID) \)
- **NIL** This \( \text{sp} \) matches any NIL.

Other atoms Atoms not in the above list are patterns that match samples with which they are \text{EQUALN}. (See descriptions of \text{METALST} in Section 3.2.1 for exceptions.)

\[
(\text{OR.} \text{pat}_1 \ldots \text{pat}_n) \quad \text{See text below for description.}
\]

\[
((\text{C.} i j \text{pat})) \quad \text{See text below for description.}
\]

\[
((\text{ANY.} \text{pat.} \ldots \text{pat}_n)) \quad \text{See text below for description.}
\]

The simple pattern,

\[
(\text{OR.} \text{pat}_1 \ldots \text{pat}_n)
\]

matches any sample which matches at least one \( \text{pat}_j \) where \( 1 \leq j \leq n \). For example, the value of

\[
(\text{SIM} (\text{QUOTE}(\text{OR. A B N.}))X)
\]

is true if \( X \) is \( A, B, \) or any number and false otherwise.

The simple pattern \((\text{ANY. pat}_1 \ldots \text{pat}_n))\) matches any sample that matches at least one \( (\text{pat}_j) \) where \( 1 \leq j \leq n \). Further, if no match is found, the pattern behaves as if the \text{any:clause} had been deleted.

For example, the value of

\[
(\text{SIM} (\text{QUOTE}((\text{ANY. A B})))X)
\]

is true if \( X \) is \((A), (B), \) or \text{NIL} and false otherwise. (See example 4.6 for a restriction of the use of this option.)
The simple pattern

$$((C \cdot i \ j \ \text{pat}))$$

matches anything of the form

$$(a_1 \ldots a_i), (a_1 \ldots a_i a_{i+1}) \ldots (a_1 \ldots a_j)$$

where $$0 \leq i \leq j$$, and each $$a_x$$ matches pat where $$1 \leq x \leq j$$. If $$i$$ is 0 and no match is found, SIM behaves as if $$(C \cdot i \ j \ \text{pat})$$ were deleted from the pattern.

For example, the value of

$$\text{SIM} \ (\text{QUOTE}((C \cdot 2 \ 3 \ A)))X)$$

is true if $$X$$ is $$(A \ A)$$ or $$(A \ A \ A)$$ and false otherwise.

The value of

$$\text{SIM} \ (\text{QUOTE}((C \cdot 0 \ 2 \ R)))X)$$

is true if $$X$$ is NIL, $$(R)$$, or $$(R \ R)$$ and false otherwise.

The value of

$$\text{SIM} \ (\text{QUOTE}((C \cdot 1 \ 2 \ (\text{OR} \cdot A \ B))))X)$$

is true if $$X$$ is $$(A)$$, $$(B)$$, $$(A \ A)$$, $$(B \ A)$$, $$(A \ B)$$, or $$(B \ B)$$ and false otherwise. In this case $$(\text{OR} \cdot A \ B)$$ is pat in $$((C \cdot i \ j \ \text{pat}))$$.

(See Example 7 in Section 4 for restrictions of the use of $$((C \cdot i \ j \ \text{pat}))$$ when concatenated with other patterns.)

3.2.1 Atomic Pattern Extensions

All nonidentity matching atomic patterns such as $$A$$, $$V$$, etc., are referenced through $$(\text{METALST} \cdot \text{MANIP})$$, a POTENTIAL FLUID SYMBOL variable. METALST is constructed in the following format:

$$\text{METALST} = ((p_1 \cdot f_1) \ldots (p_n \cdot f_n))$$

where the $$p_i$$'s are atoms, the $$f_i$$'s are (FORMAL BOOLEAN SYMBOL), $$1 \leq i \leq n$$.

When an atomic pattern is encountered, it is searched for as a $$p_i$$, and, if it is found, the corresponding $$f_i$$ is applied to the sample to obtain the value of SIM. Therefore, by redefining METALST, new options may be easily added to the pattern matches.
For example, consider the following sequence:

(SECTION (USER MANIP) SYMBOL)
(SET METALST
 (CONS (QUOTE ARRAY.) (FUNARG BOOLEAN (A) (ARRAYP A))) METALST)

Then the value of (SIM (QUOTE ARRAY.) X) is true if X is an array and false otherwise.

4. EXAMPLES

If X and Y are pats then Z = (CONS X Y) is also a pat. Further, if (SIM X A) is true and (SIM Y B) is true, then (SIM (CONS X Y) (CONS A B)) is true.

Example 1:

Let X = N.
  Y = L.
  A = 1.7
  B = (A B)

then (SIM X A) is true
  (SIM Y B) is true

therefore (SIM (CONS X Y) (CONS A B)) = (SIM (QUOTE (N. L.))) (QUOTE (1.7 (A B))) is true.

Example 2:

Let X = Z
  Y = (OR. A B)

then (SIM (CONS X Y) S) = (SIM (QUOTE (Z OR. A B)) S) has value true if S is (Z . A) or (Z . B) and false otherwise.

Example 3:

Let P = (X Y Z)

then P is a legal pat because X, Y, Z and NIL are all legal pats and
P = (X Y Z) = (CONS X (CONS Y (CONS Z NIL))).

(SIM (QUOTE (X (OR. A B) Z)) S) is true if S is (X A Z) or (X B Z) and false otherwise.
Example 4:

(SIM (QUOTE(OR. A (C. 1 3 B)))S) is true if S is A or (C. 1 3 B) and false otherwise.

Note: (((C. i j pat )) is a sp but (C. i j pat) is not.

(SIM (QUOTE(OR. A((C. 1 3 B))))S) is true if S is A, (B), (B B), or (B B B) and false otherwise.

Caution is necessary when using various arrangements of or:phrases, a:phrases and any:phrases. In the example above, pat_2 in (OR. pat_1 -- pat_n) is specified first as (C. 1 3 B) and second as ((C. 1 3 B). These are two quite different things.

Example 5:

(A (ANY. X Y)B) is a legal pat constructed by (CONS A(APPEND((ANY. X Y)) (B))) and A, ((ANY. X Y)) and B are all legal pats. (Recall if X and Y are pats, then (APPEND X Y) is a pat.)

(SIM (QUOTE(A(ANY. X Y)B))S) is true if S is (A X B) or (A Y B). It is also true if S is (A B) because if no matches are found for an any:clause the match looked for is with NIL and recall (APPEND NIL S) = S.

Example 6:

(SIM (QUOTE((ANY. A.)B))S) has the same value as (SIM (QUOTE(A. B))S). The reason that (SIM (QUOTE((ANY. A.)B))(QUOTE (B))) is false is that B is an atom and the any:phrase matches it leaving nothing in the sample to match B in the pat. No attempt is made by SIM to "slide" the pat over the sample.

Example 7:

(SIM (QUOTE((C. 0 2 A.)B))S) has value true if S is (atom atom B) and is false otherwise. The reason that (SIM (QUOTE((C. 0 2 A.)B))(QUOTE(B))) and (SIM (QUOTE((C. 0 2 A.)B))(QUOTE(X B))) have value false is the a:phrase accepts B in the sample as a match for A, and leaves nothing in the sample to match B in the pat. Caution is necessary when using a:phrases because SIM makes no attempt to "slide" the pat over the sample.

Example 8:

The example below demonstrates an example using an extended METALST. The new simple pat ARITHP. matches any prefix conglomeration of the operators + - * /.
(SECTION (USER MANIP) SYMBOL)

(SET METALST (CONS (CONS (QUOTE ARITHP.) ARITHP) METALST))

(FUNCTION (ARITHP BOOLEAN)(X)
  (SIM (QUOTE
    (OR. V. ((OR. PLUS TIMES)(C. φ 1000000 ARITHP.))
     ((OR. DIFFERENCE QUOTIENT IQUOTIENT)ARITHP. ARITHP.)
     (MINUS ARITHP.)))X))

The value 1000000 is an arbitrary limit. Note that ARITHP is involved with
the recursion of SIM; that is, ARITHP and SIM use each other for the evaluation.
<table>
<thead>
<tr>
<th>Name</th>
<th>Phone</th>
</tr>
</thead>
<tbody>
<tr>
<td>B. Barancik</td>
<td>2105</td>
</tr>
<tr>
<td>J. Barnett</td>
<td>2025</td>
</tr>
<tr>
<td>E. Book</td>
<td>2332</td>
</tr>
<tr>
<td>R. Bosak</td>
<td>2013</td>
</tr>
<tr>
<td>J. Burger</td>
<td>9919</td>
</tr>
<tr>
<td>D. Drukey</td>
<td>2105</td>
</tr>
<tr>
<td>S. Feingold</td>
<td>9525</td>
</tr>
<tr>
<td>Donna Firth</td>
<td>2310</td>
</tr>
<tr>
<td>E. Jacobs</td>
<td>2344</td>
</tr>
<tr>
<td>S. Kameny (50)</td>
<td>2009</td>
</tr>
<tr>
<td>E. Myer</td>
<td>2227</td>
</tr>
<tr>
<td>M. Perstein</td>
<td>2332</td>
</tr>
<tr>
<td>V. Schorre</td>
<td>2330</td>
</tr>
<tr>
<td>J. Schwartz</td>
<td>2123</td>
</tr>
<tr>
<td>R. Simmons</td>
<td>9439</td>
</tr>
<tr>
<td>E. Stefferud</td>
<td>9734</td>
</tr>
<tr>
<td>A. Vorhaus</td>
<td>2213</td>
</tr>
<tr>
<td>C. Weissman (10)</td>
<td>2214</td>
</tr>
<tr>
<td>A. Irvine</td>
<td>9627</td>
</tr>
<tr>
<td>R. Long</td>
<td>9913</td>
</tr>
<tr>
<td>H. Howell</td>
<td>9912</td>
</tr>
<tr>
<td>M. Howard</td>
<td>2042</td>
</tr>
<tr>
<td>D. Perry</td>
<td>2042</td>
</tr>
<tr>
<td>R. Berman</td>
<td>4317</td>
</tr>
<tr>
<td>R. Wolfson</td>
<td>2368</td>
</tr>
</tbody>
</table>