NAME

simil — measure the non-numeric similarity between two single-spine inputs

SYNOPSIS

simil [**-nr**] [**-x** *length*] *sourcefile* [*templatefile*] [> *outputfile.sim*]

DESCRIPTION

The **simil** command measures the degree of similarity between two single-spine inputs. Similarity measures are calculated by determining the minimum *edit distance* — that is, the least amount of editorial manipulation required to transform the two inputs so that they are identical. In contrast to the **correl** command, **simil** can act on both numeric and non-numeric data, and so can be used to characterize similarity for any form of Humdrum input.

Two inputs are required by **simil** — the *source* and *template* inputs. Both inputs must contain single columns of data; multi-column inputs are forbidden. The *source* input must conform to the Humdrum syntax, however the *template* should contain only data records. Comments and interpretations should not appear in the *template*. In addition, following the initial exclusive interpretation, all comments and interpretations in the *source* input should be removed.

Depending on the mode of operation, **simil** outputs either one or two spines of continuous information regarding the similarity of the two inputs. The length of **simil**'s output matches that of the *source* file.

Two modes of operation are provided by **simil**: the *fixed template mode* and the *variable template mode*. The fixed template mode is useful for scanning a source input for patterns similar to a given template. This mode of operation is useful when the user knows in advance what feature or pattern is being sought. The variable template mode, by contrast, provides an automated means for discovering common patterns shared by the template and source inputs.

In the *fixed template mode*, a single output spine is generated, dubbed `**simil`. Output similarity values are numbers ranging between zero and one. These numbers indicate the relative similarity between the source and template inputs at the current location in the *source* file. Values near zero indicate great dissimilarity whereas a value of precisely 1.00 indicates that the template and source match exactly at the current position.

In *fixed template mode*, the entire *template* input is treated as a single pattern and compared with the *source* input. In this mode, the template may not be longer than the source input. If the length of the template is the same as the length of the source input, then only a single output value is generated — representing the edit-distance similarity of the two files. The remaining output is padded by null tokens. If the template is shorter than the source input,

then the source input is scanned using the template. For each data record in the source input the edit-distance similarity is measured between the entire template and a string of corresponding length beginning at the current point in the source input. Each successive output value indicates the degree of similarity between the template and source inputs as the template is shifted along the source input.

In the *variable template mode,* the template input provides a "reservoire" from which multiple templates are derived. Specifically, the template input is broken-up into a series of (shorter) overlapping "subordinate" templates. The length of each of these subordinate templates is determined by the **-x** option parameter. For example, consider a template input consisting of the values: 1, 2, 3, 4 — each number appearing on successive lines. With **-x 3**, two subordinate templates will be generated, each consisting of three data records: 1, 2, 3 and 2, 3, 4. With **-x 2**, three subordinate templates will be generated, each consisting of two data records: 1, 2; 2, 3; and 3, 4.

Two output spines are generated in the *variable template mode:* `**simil` and `**simxrf`. The `**simxrf` spine provides cross-reference information identifying the place in the template input where a subordinate pattern is highly similar to the current position in the source file. As each record is encountered in the source input, **simil** scans the list of all possible subordinate templates and identifies the template with the highest similarity value. This value is output (in the `**simil` spine) along with the *line number* in the original template input where the subordinate template begins. If more than one subordinate template shows the same similarity value, then the line numbers for each high-similarity template appear in the `**simxrf` spine, separated by commas. Common subordinate patterns will appear frequently in the `**simxrf` output.

It is recommended that output files produced using the **simil** command should be given names with the distinguishing '.sim' extension.

FURTHER DETAILS

The **simil** program implements the Damerau-Levenshtein metric for edit distance (see REFERENCES). Permissible edit operations include substitutions and deletions. Each edit action incurs a penalty, and the cummulative edit-distance penalty determines the similarity.

In the default operation, **simil** assigns equivalent edit penalties (1) for deletions and substitutions. However, the user can explicitly define these penalties.

The **simil** command allows the user to define the cost of each edit operation via an initialization file. The initialization file must be named `simil.rc` and be located in the current directory or the user's home directory. Arbitrary costs may be assigned to any of eight edit operations shown in the following table:

Name Tag	Edit Operation
D1	Delete a nonrepeated token in String 1
D2	Delete a nonrepeated token in String 2
R1	Delete a repeated token in String 1
R2	Delete a repeated token in String 2
S0	Substitute a token that is repeated in neither String 1 nor String 2
S1	Substitute a token that is repeated in String 1 only
S2	Substitute a token that is repeated in String 2 only
S3	Substitute a token that is repeated in String 1 and String 2

Edit operations used by **simil**.

In describing the edit operations, String 1 is the source string and String 2 is the template string. Notice that there is no overt edit operation for insertion: an insertion in String 1 is deemed equivalent to a deletion in String 2. However, different edit penalties may be defined for deletions from String 1 (D1) compared with deletions from String 2 (D2). In musical applications defining such asymmetrical penalties may be important. For example, two inputs may represent a basic melody and an embellished variant of the melody. Using asymmetrical penalties allows the user to specify that the deletion of tones from the embellished version is less costly than deletion of tones from the basic melody.

Since repetition is a common form of musical variation, **simil** allows the user to distinguish between repeated and non-repeated tokens. A repeated token is defined as one that is immediately preceded by an identical token. Thus, in deleting a sequence of identical symbols in String 1, say, all deletions except the first occurrence are R1 operations, whereas the deletion of the first occurrence is a D1 operation.

Note that the minimum theoretical edit-distance for any set of penalty weightings can be determined empirically by providing the **simil** program with source and template strings that share no symbols in common. For example, the source input may consist entirely of numbers, whereas the template input consists entirely of alphabetic characters. In the case where all edit operations are assigned a penalty of +1, the minimum quantitative similarity between two strings is 0.37.

Some user-defined weightings may give rise to peculiar results — such as negative costs — but **simil** does not forbid this. **Simil** generates warning messages if the weighting seem illogical; for example, if the cost of R1 is more than that of D1. In addition, **simil** will abort operation if the defined edit penalties transgress the triangular inequality (see REFERENCES). The default weighting for all operations is +1.

Below is a sample initialization file that defines the R1 substitution has having an edit penalty of 0.7, whereas the R2 substitution is given a penalty of 0.9. Edit penalties are defined by specifying the operation, followed by some spaces or tabs, followed by some real number. Since no other penalties are defined in this file, the remaining edit operations use the default edit penalty of 1.0. If any operation is assigned more than one weight, the latest assignment is used. The user may effectively eliminate a given edit operation by defining an arbitrarily high edit penalty.

```
# This is a comment.
R1      0.7
R2      0.9
```

OPTIONS

The **simil** command provides the following options.

-n do not scale similarity measures according to template length

-r reverse the order of *source* and *template* inputs on the command line; permits the source file to be entered using the standard input.

-x *length* invoke *variable template mode*; break-up template file input into subordinate patterns of length *length*

Options are specified in the command line.

Raw edit-distance scores are normally unreliable estimates of similarity, unless the length of the template is considered. For example, 3 editing operations constitutes a rather modest change for a template consisting of 20 elements. However, 3 edit operations is significant for a template consisting of only 5 elements. In the default operation, **simil** scales the edit-distance scores according to the length of the comparison template. This ensures that all similarity values remain between 0 and 1. The **-n** option defeats this scaling procedure, and outputs the raw similarity scores.

The **-r** option reverses the order of the *source* and *template* input specifications on the command line. If both inputs are files, this option is of little use. Where one input is to be typed manually via the standard input, this option allows the user to specify a template file as input, and to type the source document manually.

The **-x** option invokes the *variable template mode* discussed above. The numerical argument given to the **-x** option determines the length of the subordinate templates drawn from the template file.

EXAMPLES

The following examples illustrate the operation of **simil.** Consider first, the *fixed template mode*. In the following example, the source input consists of the left-most spine (labelled `**foo`) and is held in a file named `source`; the middle column (not Humdrum) consists of the letters A, B and C, and is held in the file named `template`. The following command:

```
simil source template
```

generates the third column (labelled `**simil`):

```
(source     (template    (simil
 input)      input)      output)
**foo       A            **simil
X           B            0.51
A           C            1.00
B                        0.51
C                        0.37
D                        0.51
A                        0.72
B                        0.72
B                        0.51
C                        0.51
B                        .
A                        .
*-                       *-
```

Each successive value in the output spine is matched with a data token in the source input file. For example, the second value (1.00) in the `**simil` spine arises from an exact match of the (A, B, C) pattern *beginning* with the second data token in the source input. The second highest value (0.72) occurs in both the sixth and seventh `**simil` data records, indicating that fairly similar sequences occur beginning with the sixth and seventh data records in the source input. Specifically, **simil** has recognized that the sequence (A, B, B, C) is only one edit-operation (a deletion) different from the template (A, B, C). In the ensuing record, **simil** has recognized that the sequence (B, B, C) is only one edit-operation (substitution A/B) different from (A, B, C). Notice that the final value (0.51) indicates that the edit distance for (C, B, A) is less like the template. Also notice that the lowest value (0.37) corresponds to an input pattern (beginning D, D, A) that bears little resemblance to the template.

If the above input were pitches, it might be argued that *changing* a pitch is more dissimilar than *repeating* a pitch. In the following `simil.rc` file, an increased penalty has been assigned for dissimilar substitution, and decreased penalties have been assigned for repetition.

```
S0   1.6
S1   0.7
S3   0.7
```

Repeating the above command with this new `simil.rc` file produces the following results:

```
(source     (template     (simil
 input)       input)     output)
**foo        A           **simil
X            B           0.51
A            C           1.00
B                        0.51
C                        0.26
D                        0.51
A                        0.79
B                        0.59
B                        0.51
C                        0.51
B                        .
A                        .
*-                       *-
```

Notice that the similarity measure for the pattern (A, B, B, C) has increased from 0.72 to 0.79, whereas the similarity measure for (B, B, C) has decreased from 0.72 to 0.59.

Consider now the use of the *variable template mode.* Once again, we will use the same source and template files. Given the short length of the template, there is little choice regarding the length of the subordinate templates. In the following command, a template length of two elements is specified:

```
simil -x 2 source template
```

This command produces the following output:

```
*simil    *simxrf
0.37      1,2
1.00      1
1.00      2
0.37      1,2
0.37      1,2
1.00      1
0.61      1,2
1.00      2
0.61      1
0.61      2
0.61      1
.         .
```

Only two two-element subordinate patterns are possible given out template — A, B and B, C. The first subordinate template begins on line 1 of the template file, while the second subordinate template begins on line 2. The `**simxrf` spine identifies which of the subordinate patterns is most similar to the source file at the given input record. The `**simil` spine identifies the corresponding similarity measure for the most similar pattern. For example, the second and third `**simil` records both report similarity values of 1.00. However, the first instance is associated with the pattern beginning on template record 1 (A,

B), whereas the second instance is associated with the pattern (B, C) beginning on template record 2.

PORTABILITY

DOS 2.0 and up. OS/2. All UNIX systems.

SEE ALSO

context (4), ****correl** (2), **correl** (4), **patt** (4), **pattern** (4)

WARNINGS

In *variable template mode*, execution times may be quite lengthy.

REFERENCES

The Damerau-Levenshtein metric is described in P. Hall & G. Dowling "Approximate string matching," *ACM Computing Surveys*, Vol. 12 (1980) pp. 381-402.

The theory and operation of *simil* is explained in Keith Orpen and David Huron, "Measurement of similarity in music: A quantitative approach for non-parametric representations," *Computers in Music Research*, Vol. 4 (1992), pp. 1-44.

AUTHOR

Written by Keith S. Orpen, University of Waterloo.