Chapter 26

Moving Signifiers Between Spines

In many types of tasks is it useful to be able to transfer signifiers from one spine to another. In this chapter we will discuss two commands. The **rend** command makes it possible to split characters in a single spine and distribute them across two or more spines. The **cleave** command does the reverse: it allows characters that are distributed across two or more spines to be gathered into a single new spine. These two commands provide opportunities to create Humdrum spines that contain precisely the information of interest to the user.

The *rend* Command

The **rend** command allows a Humdrum spine to be broken apart into two or more spines. Different pieces of information can be distributed to the individual output spines. Consider for example the following spine containing `**pitch` data:

```
**pitch
Ab3
F#4
C5
*-
```

The **rend** command might be used to structure this as three independent spines:

```
**octave        **note        **accidental
3               Ab            b
4               F#            #
5               C             .
*-              *-            *-
```

The operation of **rend** requires a reassignment file where each line contains an output exclusive interpretation, followed by a tab, followed by a regular expression. In the above case, the reassignment file (`reassign`) contained the following:

```
**octave        [0-9]
**note          [A-Gb#x]
**accidental    [b#x]
```

The first line tells **rend** that any signifiers matching the character-class 0-9 should be output in a spine labelled `**octave`. The second line causes signifiers matching the upper-case letters A to G and the `b`, `#` and `x` signifiers to be output in a spine labelled `**note`. The third line causes signifiers matching just `b`, `#` and `x` to be output in a spine labelled `**accidental`.

The above output was generated by invoking the following command:

```
rend -i '**pitch' -f reassign inputfile
```

Note that the **-i** and **-f** options are mandatory. The **-i** option tells **rend** which input spines to process and the **-f** option tells **rend** the name of the file containing the spine-reassignments.

The **rend** command is typically paired with a subsequent **cleave** command.

The *cleave* Command

The **cleave** command amalgamates concurrent data tokens in two or more spines into a single data spine. In effect, **cleave** does the opposite of **rend.** By way of example, **cleave** can be used to transform the following:

```
**A        **B        **C
a          b          c
A          B          C
*-         *-         *-
```

into:

```
**new
abc
ABC
*-
```

Specifically, the above "cleaving" would be done using the following command:

```
cleave -i '**A,**B,**C' -o '**new' inputfile
```

Both the **-i** and **-o** options are mandatory. The **-i** option tells **cleave** which exclusive interpretations should be cleaved together. In the above case, we have provided a list of three types of data. The **-o** option tells **cleave** what to call the resulting cleaved spine. In this case, we've simply called the result `**new`.

Suppose that we would like to automatically add key-velocities to some `**MIDI` data that reflect the normal accents arising from the meter. For example, in 4/4 meter, we would like the first note in each measure to be strongest, the third beat to be next most strongest and so on. Recall that `**MIDI` data tokens consist of three elements: (1) the duration in MIDI clock ticks, (2) the key number (also on/off indication), and (3) the key velocity. In order to add accents, we need to change the key velocity values. The maximum MIDI key velocity value is 127; the minimum value is 0; and the default value is 64.

Consider the following hypothetical input file:

```
**kern
*M4/4
=1-
4c
8d
8e
8f
8g
8a
8b
=2
2cc
*-
```

The **metpos** command can be used to identify the metric position of various note onsets. Before using **metpos** however, we must use the **timebase** command to create an isorhythmic record structure. Since the shortest note is an eighth-note, the appropriate command is as follows.

```
timebase -t 8 scale > scale.tb
```

Using the timebased output, we can then invoke the **metpos** command:

```
metpos scale.tb > scale.met
```

The resulting output contains both the original input (on the left) and the metric position spine (on the right):

```
**kern   **metpos
*M4/4    *M4/4
*tb8     *tb8
=1-      =1-
4c       1
.        4
8d       3
8e       4
8f       2
8g       4
8a       3
8b       4
=2       =2
2cc      1
.        4
.        3
.        4
*-       *-
```

Let's now eliminate the null data tokens introduced by **timebase**. Using **humsed**, we delete each data record beginning with a period character:

```
humsed '/^\./d' scale.met > scale.tmp
```

Next, we can use the **recode** command to change the metric position values to appropriate MIDI key velocities. We might use the following reassignment file (named `accent`):

```
==1     100
==2     80
==3     60
==4     40
else    error
```

In applying **recode** we will take care to avoid processing measure numbers using the **-s** (skip) option:

```
recode -f accent -s ^= -i '**metpos' scale.tmp > scale.acc
```

The output will now appear as follows:

```
**kern   **metpos
*M4/4    *M4/4
*tb8     *tb8
=1-      =1-
4c       100
8d       60
8e       40
8f       80
8g       40
8a       60
8b       40
=2       =2
2cc      100
*-       *-
```

Now we can use the **midi** command to generate `**MIDI` data:

```
midi scale.acc > scale.mid
```

The result is given below:

```
**MIDI                  **metpos
*Ch1                    *
*M4/4                   *M4/4
*tb8                    *tb8
=1-                     =1-
72/60/64                100
72/-60/64 72/62/64      60
36/-62/64 36/64/64      40
36/-64/64 36/65/64      80
36/-65/64 36/67/64      40
36/-67/64 36/69/64      60
36/-69/64 36/71/64      40
```

```
=2                    =2
36/-71/64 36/72/64    100
144/-72/64            .
*-                    *-
```

Before using **cleave** to join the new key velocity values to the `**MIDI` data we need to delete the current key-down velocities. These are the values '64' preceding the tab character. The **humsed** command can be used as follows:

```
humsed 's/64<tab>/<tab>/' scale.mid > scale.tmp
```

The modified output will now be:

```
**MIDI                **metpos
*Ch1                  *
*M4/4                 *M4/4
*tb8                  *tb8
=1-                   =1-
72/60/                100
72/-60/64 72/62/      60
36/-62/64 36/64/      40
36/-64/64 36/65/      80
36/-65/64 36/67/      40
36/-67/64 36/69/      60
36/-69/64 36/71/      40
=2                    =2
36/-71/64 36/72/      100
144/-72/              .
*-                    *-
```

Finally, we use **cleave** to add the new key-down velocities.

```
cleave -i '**MIDI,**metpos' -o '**MIDI' scale.tmp > scale.mid
```

The final output is:

```
**MIDI
*
*M4/4
*tb8
=1-=1-
72/60/100
72/-60/64 72/62/60
36/-62/64 36/64/40
36/-64/64 36/65/80
36/-65/64 36/67/40
36/-67/64 36/69/60
36/-69/64 36/71/40
=2=2
36/-71/64 36/72/100
```

```
144/-72/
*-
```

Creating Mixed Representations

For some analytic tasks it is often useful to generate a special representation that combines all of the elements or types of data of interest to the researcher. For example, suppose we were working on a model of melodic organization that reduced melodies to three types of information: relative-duration context, gross pitch height, and scale step. Sample data tokens for our representation and their meanings are given in the following table. Notice that the order of signifiers is important:

token	**meaning**
`LSLHto`	long-short-long rhythm, high pitch, tonic
`LLSLsd`	long-long-short rhythm, low pitch, subdominant
`MLSMlt`	medium-long-short rhythm, medium pitch, leading tone
`r`	rest

In Chapter 22 we learned how to use **recode** to classify various numerical ranges and **humsed** to classify non-numeric data. We already know how to create the elements of our new representation.

The scale degree information can be created by using **deg** and **humsed** can be used to transform the signifiers as in the following `degree` file:

```
s/1.*/to/
s/2.*/st/
s/3.*/me/
s/4.*/sd/
s/5.*/do/
s/6.*/sm/
s/7.*/lt/
```

We can classify the pitch ranges into high, medium, and low using the **semits** command, followed by **recode**. For example, we could transform the `**semits` data using the following reassignment file:

```
<0      L
>16     H
>=0     M
else    r
```

Durations can be similarly classified into long (L), medium (M), and short (S) using the **dur** command, followed by recode.

```
>1.0      L
>0.5      M
>0        S
```

Using **context -n 3** we could then create contextual 'triples' so that data records contain three durations. Suppose also that we have used **sed** to change the names of the exclusive interpretations so they are more appropriate. As a result we have three spines that, when assembled together are organized as in Example 26.1

Example 26.1

```
**rhythm      **range    **scale-step
L S L         H          to
L L S         L          sd
M L S         M          lt
r             r          r
*-            *-         *-
```

We need to process the first spine with **humsed** again to eliminate the spaces in the multiple stops. The rhythm spine would be processed as follows:

```
humsed 's/ //g' rhythm > rhythm.new
```

We could assemble these spines using the **assemble** command:

```
assemble rhythm.new range scale.step > newfile
```

Finally we can use **cleave** to amalgamate all of the data into a single final spine.

```
cleave -i '**rhythm,**range,**scale-step' -o '**complex' \
   newfile > output
```

Having created our new representation, we can continue to process this new data with the various Humdrum tools. For example, we could generate inventories that answer questions such as "How often does a high subdominant note in a long-short-long rhythmic follow a low submediant in a long-long-short context?

A similar approach can be used to address other questions, such as do large leaps involving chromatically-altered tones tend to have a longer duration on the altered tone? Etc.

Reprise

In this chapter we have seen how **rend** and **cleave** can be used to take bits and pieces of signifiers from potentially many spines, and assemble a composite Humdrum spine that contains precisely the information of interest. Before amalgamating spines, you can use the **humsed** command to translate the characters/signifiers so that you use your preferred way of representing something.