Chapter 17

Creating Inventories

Many research problems can be addressed by building an *inventory* — that is, identifying the number of occurrences of various types of data. Questions such as the following all pertain to the generation of inventories:

- Does Liszt use a greater variety of harmonies than Chopin?
- What is the most frequently used dynamic marking in Beethoven, and how does Beethoven's practice compare with that of Brahms?
- Are flats more common than sharps in Monteverdi?
- Did Bartók's preferred articulation marks change over his lifetime?
- Is there a tendency to use the subdominant pitch less often in pop melodies than in (say) French chanson?
- How frequent are light-related words such as "lumen" or "lumine" in the different monastic offices for Thomas of Canterbury?
- Is it true that 90 percent of the notes in a given work by Bach use just two durations (such as eighths and sixteenths, or eighths and quarters)?
- What is the most common instrumental combination for orchestral works by Musorgsky?

At the end of this chapter we will show the Humdrum commands needed to answer each of the above questions.

The above questions are all variations of one of the following forms:

> How many different types of _____ are there?
> What is the most/least common _____?
> What is the frequency of occurrence for various _____s?

In some cases, we're asked to compare two or more repertories when answering one of these basic questions.

For illustration purposes, consider the case of a Humdrum file named `alpha` containing the following simple input:

```
**alpha
A
B
A
A
C
B
*-
```

It doesn't matter what the data represent. The "A", "B", and "C" might signify different articulation marks, chords, harmonic intervals, or instrumental configurations. Whatever is represented, the process of generating an inventory is the same. Ultimately, we'd like to produce a simple distribution that indicates:

```
3 occurrences of "A"
2 occurrences of "B"
1 occurrence  of "C"
```

Filter, Sort, Count

Building an inventory is a three-step process. First we need to *filter* the input so only the data of interest is present. Second we need to *sort* like-with-like. And third we need to *count* the number of occurrences of each type of data token.

Let's begin by discussing the second process. In Chapter 3 we saw how the UNIX **sort** command will rearrange lines of data so that they are in alphabetical/numerical order. The command:

```
sort alpha > sorted.alpha
```

will sort the file `alpha` and place the results in a file named `sorted.alpha`. The file `sorted.alpha` will contain the following:

```
**alpha
*-
A
A
A
B
B
C
```

Notice that the asterisk is treated as alphabetically prior to the the letter 'A', so all the Humdrum interpretation records have been moved to the beginning of the output. Notice also that all of the lines beginning with the letter 'A' are now collected on successive lines in the output. Similarly, the 'B's have been rearranged on successive lines.

The third step in generating an inventory is to count the number of occurrences of each unique data token. The **uniq** command described in Chapter 3 will eliminate successive duplicate lines. For example, if we type:

```
uniq sorted.alpha
```

The output will be as follows:

```
**alpha
*-
A
B
C
```

Notice that repetitions of the data "A" and "B" have disappeared. The simple **uniq** command is useful for telling us *how many different things* there are in an input. For example, the above output identifies just five different records — and three different types of data records.

Recall that the **-c** option for **uniq** will cause a 'count' to be prepended to each output line. The command:

```
uniq -c sorted.alpha > unique.alpha
```

will produce the following output:

```
1   **alpha
1   *-
3   A
2   B
1   C
```

The prepended counts tell us that 'A' occurs three times, 'B' occurs twice, and all other records occur just once.

In the above output, `**alpha`, and `*-` are Humdrum interpretations rather than data, so we probably don't want them to appear in our inventory. If our file had contained comments, or null data records, then these would have also appeared in our output, although we are not likely to be interested in them. This leads us to what is normally the first step in generating an inventory — *filtering* the input in order to eliminate records that we'd prefer to omit from our final output.

Filtering Data with the *rid* Command

As we saw in Chapter 13, the **rid** command can be used to eliminate various classes of Humdrum records. For example, **rid -G** eliminates all global comments; **rid -D** eliminates all data records, etc. The option combination **-GLId** is very common with **rid** since only data records are retained in the output. That is, eliminating all global and local comments, omitting all interpretations, and deleting all null data records will result in an output consisting only of non-null data records.

Returning to our `**alpha` data, we can eliminate everything but data records as follows:

```
rid -GLId alpha > filtered.alpha
```

By way of summary, generating an inventory is a three-step process. First we *filter* the input so only the data of interest is present. Typically, this means using the **rid** command with one or more options to eliminate comments, interpretations, and perhaps null data records. Second we *sort* the

data using the **sort** command so that identical records are amalgamated as neighbors. Finally, we use the **uniq -c** to *count* the number of occurrences of each type of data token. All three steps can be amalgamated into a single pipeline:

```
rid -GLId alpha | sort | uniq -c > inventory.alpha
```

Notice that the inventory will pertain to whatever data was provided in the original input. We've been using the abstract data "A", "B", and "C". However, this data might represent any type of discrete data, such as Latin text, piano fingerings, or dance steps.

Inventories for Multi-spine Inputs

In the above example, we assumed that the input consists of a single Humdrum spine (i.e. a single column of data). However, Humdrum files can have any number of spines, and each spine might represent radically different types of data. For example, the following file (named `alphabet`) contains two spines, one with "alpha" data, and the second with "bet" data. These data types might represent melodic intervals and fingering information, or dynamic markings and stem-directions, or whatever.

```
**alpha   **bet
A         $50
B         $50
A         $50
A         $200
C         $50
B         $50
*-        *-
```

If we apply our above inventory-generating commands for the file "alphabet," the result will be as follows:

```
1   A   $200
2   A   $50
2   B   $50
1   C   $50
```

Notice that the inventory is based on *entire records* containing both "alpha" and "bet" data. This is the reason why the alpha-bet data-pair "`A  $50`" is considered different from alpha-bet data "`A  $200`". Depending on the user's goal, this may or may not be the most appropriate output.

A situation where this approach might be desired arises when we are counting the number of different spellings of chords (e.g., how many different sonorous arrangements are there?). If **alpha and **bet represent pitches in two concurrent voices, then it may be important to have both concurrent data tokens participating in the inventory.

In other circumstances, we may not want this. For example, if we are interested only in alpha-related data, we need to eliminate the irrelevant **bet data so it won't interfere. This can be done using the Humdrum **extract** command.

For example, we can create an inventory of just the `**bet` data:

```
extract -i '**bet' alphabet | rid -GLId | sort | uniq -c \
   > inventory.bet
```

The resulting `inventory.bet` file will contain:

```
1   $200
5   $50
```

— meaning 5 occurrences of the data "$50" and 1 occurrence of "$200".

Sometimes it is useful to create an aggregate inventory of the data in each separate spine. In such cases, we will need to use **extract** several times so that each spine is placed in a separate file:

```
extract -i '**alpha' alphabet > justalpha
extract -i '**bet' alphabet > justbet
```

The **cat** command can then be used to concatenate the files end-to-end so they form a single column of data. With each data token of interest is on its own line, we can generate the appropriate inventory:

```
cat justalpha justbet | rid -GLId | sort | uniq -c
```

Sorting By Frequency of Occurrence

When the output inventory list is short, it is easy to identify which records are the most common and which records are the least common. Frequently inventory lists will contain dozens or hundreds of items so it may be more difficult to scan through the output to find the most frequent or least frequent occurrences. For such long outputs, it might be more convenient to produce an output sorted according to frequency of occurrence. Notice that each output record from **uniq -c** begins with a number, and so the output is ideally suited for numerical sorting. We've already learned that the **sort** command rearranges input records in alphabetic/numeric order.

If we type

```
sort inventory.alpha
```

The output will be as follows:

```
1   C
2   B
3   A
```

Now the output is sorted so that the least frequent occurrences are at the beginning, and the most frequent occurrences are at the end of the output. Incidentally, **sort** has a **-r** option that causes the output to be sorted in reverse order. If we use **sort -r**, then the most common occurrences will be placed at the beginning of the output:

```
sort -r inventory.alpha
```

produces the following output:

```
3   A
2   B
1   C
```

Once again, we can amalgamate all of the required commands into a single pipeline. The following pipeline produces an inventory for any type of Humdrum input, sorted from the most common to the least common data:

```
rid -GLId alpha | sort | uniq -c | sort -r > inventory.alpha
```

Counting with the *wc* Command

In other circumstances, it may be helpful to determine the proportion or percentage values rather than the actual numerical count. This can be calculated by dividing each of the inventory count numbers by the total number of data records processed. A convenient way to count records is via the UNIX **wc** (word count) command. The **wc** command provides three options. With the **-c** option, **wc** counts the number of characters in an input. With the **-w** option, **wc** counts the number of words in an input. A "word" is defined as any sequence of characters delineated by white space, such as spaces, tabs or new lines. With the **-l** option, **wc** counts the nmber of lines or records in the input.

We can count the total number of non-null data records in a Humdrum input using the following pipeline:

```
rid -GLId alpha | wc -l
```

This will give us the total number of items in our inventory. Simple division will generate the percentages for each type of data record.

Suppose, for example, that the total number of data records was determined to be 874. Using the UNIX **awk** command will allow us to easily generate the percentages for each data type via the command:

```
awk '{print $1/847*100 "\t" $2}' inventory.alpha
```

This will create a two-column output. The first column will indicate the percentage of occurrence, and the second column will identify the corresponding type of data.

Excluding or Seeking Rare Events

Recall from Chapter 3 that the **uniq** command provides other options (besides the **-c** option). The **-d** option causes **uniq** to output *only* those records that are duplicated. In other words, records that occur only once are eliminated from the input. This option can be useful when there are a lot of single-occurrence data tokens and you are only interested in those data records that occur more frequently.

By contrast, the **uniq -u** option causes *only* those records that are unique (occur only once) to be

output. This option can be useful when looking for rare circumstances in our data.

```
rid -GLId alpha | sort | uniq -u    (output only the rare events)
rid -GLId alpha | sort | uniq -d    (eliminate all the rare events)
```

Transforming and Editing Inventory Data

Notice that two data records must be identical in order for them to be considered "the same" by **sort** and **uniq**. This means that records such as the following are considered entirely different:

```
ABC
abc
Abc
"ABC"
ABC.
CBA
```

Remember that step #1 in generating inventories requires that we filter the data so only the data of interest is passed to **sort** and **uniq**. This means we must be careful about the state of the input. Depending on your goal, we will either want to *translate* the input to some other more appropriate representation, or *edit* the existing representation in order to discard or transform otherwise confounding data.

Translating data involves changing from one type of information to another — that is, changing the exclusive interpretations. For example, if we want to produce an inventory of melodic intervals, then we might use the **mint** or **xdelta** commands to generate a suitable representation. Alternatively, we might want to generate an inventory of scale degrees using the **deg** or **solfa** commands.

Instead of translating our data, we might wish to edit the data using the **sed** or **humsed** stream editors. Suppose we had a file (named "notes") consisting of pitch information, and we wanted to create an inventory of the diatonic pitch-letter names. Our input might look like this:

```
**notes
A
B
B
D
F#
D#
E
*-
```

Without modification, our inventory would appear as follows:

```
1   A
2   B
1   D
1   D#
1   E
1   F#
```

But this inventory distinguishes D-sharp from D-natural — which is not what we want. The answer is to filter our input so that the sharps are removed.

Adding the appropriate **humsed** command to our pipe:

```
humsed 's/#//' notes | rid -GLId | sort | uniq -c
```

— will produce the following output:

```
1   A
2   B
2   D
1   E
1   F
```

Further Examples

Given your current background, you should now be able to generate inventories to answer a wide variety of questions. You should now understand how the commands given below can be used to solve the question posed:

Does Liszt use a greater variety of harmonies than Chopin?

```
extract -i '**harm' liszt* | rid -GLId | sort | uniq | wc -l
extract -i '**harm' chopin* | rid -GLId | sort | uniq | wc -l
```

What is the most frequently used dynamic marking in Beethoven, and how does Beethoven's practice compare with that of Brahms?

```
extract -i '**dynam' beeth* | rid -GLId | sort | uniq -c \
   | sort -r | head -1
extract -i '**dynam' brahm* | rid -GLId | sort | uniq -c \
   | sort -r | head -1
```

Are flats more common than sharps in Monteverdi? Let's presume that the input is monophonic `**kern` data.

```
humsed 's/[^#-]//g' montev* | rid -GLId | sort | uniq -c
```

Did Bartók's preferred articulation marks change over his lifetime? Assume that copies of early and late works have been concatenated to the files `early` and `late`. The **humsed** command here eliminates all data with the exception of `**kern` articulation marks. (See Chapter 6 for details on

**kern articulation marks.)

```
extract -i '**kern' early | humsed 's/[^"'`~^:I]//g' \
    | rid -GLId | sort | uniq -c
extract -i '**kern' late | humsed 's/[^"'`~^:I]//g' \
    | rid -GLId | sort | uniq -c
```

Is there a tendency to use the subdominant pitch less often in pop melodies than in (say) French chanson? Once again assume that the inputs are monophonic.

```
deg -t pop* | grep -c '4'
deg -t chanson* | grep -c '4'
```

How frequent are light-related words such as "lumen" or "lumine" in the different monastic offices for Thomas of Canterbury? Familiarity with regular expressions helps:

```
extract -i '**text' office* | egrep -ic 'lum.+n[e]*$'
```

Is it true that 90 percent of the notes in a given work by Bach use just two durations (such as eighths and sixteenths, or eighths and quarters)?

```
humsed 's/[^0-9.]//g' bach | rid -GLId | sort | uniq -c
```

(Repeat the above command for each work and inspect the results.)

What is the most common instrumental combination for sonorities by Mussorgsky?

This problem is addressed in Chapter 36.

Reprise

In this chapter we have discussed how to answer questions that involve the creation of inventories. Creating an inventory typically entails *filtering* some data so only the information of interest is preserved, *sorting* the data so that like data are amalgamated, and then *counting* each instance of each data type.

In later chapters we will see how classifying data, identifying musical contexts, and marking occurrences of patterns can be used to significantly enhance the inventory-building tools described in this chapter.