A primer for the DELTA System

10 September 2016

T.R. Partridge*, M.J. Dallwitz and L.Watson

* Landcare Research New Zealand Ltd, PO Box 69, Lincoln 8152, New Zealand

Preface

This document contains up-to-date information about entering and editing DELTA data via a text editor, and running the programs in a command window. The first two editions (see References) were written before the availability of the DELTA Editor, which is now the recommended method for entering and editing DELTA data.

The User's guide to the DELTA Editor also contains an introduction to using the DELTA programs, including an updated version of Section 12. Hints on organization and maintenance of data.

Contents

1. Introduction

DELTA (DEscription Language for TAxonomy) is a standardized format for coding taxonomic descriptions. DELTA-format data can be converted into natural-language descriptions, and into formats required by programs for key generation, interactive identification and information retrieval, phenetic analysis, and phylogenetic analysis.

This Primer is an introduction to preparing DELTA-format data, and processing it with a system of programs developed at the CSIRO Division of Entomology.

The Primer is intended to be read while experimenting with the programs. The programs are supplied with a working set of data, and we suggest that you first examine and experiment with these data, as explained in Sections 4 to 11. You should then read Section 12, before proceeding to gather and use your own data. More detailed information can be found in the User’s guide to the DELTA System.

The Primer assumes that you have a basic knowledge of your computer’s operating system and text editor. You can use a word processor to prepare and edit your files, but make sure you use it in a mode in which it does not insert extra, hidden symbols, relating to word-processing functions. Such symbols will cause error messages when you run the DELTA programs, and the source of the errors may be difficult to locate, because when you look at the file with the word processor, you will not be able to see anything wrong. In these circumstances, it may be helpful to display the file by some other means, for example, the editor NOTEPAD, which supplied with the Windows operating system.

2. Preparing to use the system

The programs and installation instructions are available at delta-intkey.com/www/programs.htm.

A complete list of the programs and options, and a short description of each, is available at delta-intkey.com/www/programs-list.htm. This document also contains instructions opening a command window, and for running programs in a command window.

For each set of data you intend to work on, you will need a separate subdirectory (folder), the name of which is best the name (or shortened name) of the group involved, for example, POA, COTULA. Subdirectories can be created with the MD command, for example:

```
MD POA
```

The command TO (or the operating-system command CD) is used to move to the required subdirectory, for example:

```
TO POA
```

You enter this command before starting work on a particular set of data.

3. The program CONFOR

The program CONFOR plays a central role in processing the data. In some cases, it produces the required output directly from the DELTA-format data, for example, natural-language descriptions. In other cases, it produces intermediate data formats, which are then processed by other programs. To tell CONFOR what you want it to do, you must provide it with a file of instructions or ‘directives’. For example, to produce natural-language descriptions, you would run CONFOR with a directives file including the directive *TRANSLATE INTO NATURAL LANGUAGE.

4. Sample directives and data files

Working examples of directives and data files are stored in the system. To call them up, enter

SAMPLE *filename*

For example, SAMPLE TONAT gets directives for translation of the descriptions into natural language; and SAMPLE CHARS gets an example of a character list.

If the file is already present in your current directory, a new file with the suffix (file ‘extension’ or ‘type’) .TMP will be formed (for example, TONAT.TMP).

Entering SAMPLE by itself gives a list of the sample files available.

The sample files are extracts from a much larger data set (Watson, Dallwitz, and Johnston 1986; Watson and Dallwitz 1992 onwards). The samples were chosen to illustrate the use of the programs, and are not intended to be a biologically meaningful subset of the full data.

Most of the examples in this Primer are taken from the sample data, and can be further examined there in context. Examples not taken from the sample data are distinguished by having character numbers of 100 or greater (there are 88 characters in the sample data).

5. Data files

All applications of the DELTA system are based on information contained in three files: the characters (CHARS), the items or taxa (ITEMS) and the specifications (SPECS). CHARS contains a numbered list of the characters and states which are to be used to describe the taxa; ITEMS contains descriptions of the taxa, coded in terms of the character and state numbers; and SPECS contains information about the other two files (for example, the total number of characters). Examples of these files are available via the SAMPLE command (see Section 4).

First, create a subdirectory called EXAMPLE, by entering

```
MD EXAMPLE
```

(see Section 2). This will be used to hold copies of the sample files, and to experiment with applying the programs to these files, as described later.

5.1. The characters file, CHARS

Go to the EXAMPLE directory, if you are not already there, by entering

```
TO EXAMPLE
```

and get a copy of the sample characters file by entering

```
SAMPLE CHARS
```

Now look at this file by entering

```
NOTEPAD CHARS
```

The first line of the CHARS file is

```
*SHOW Grass Genera - character list. 26-JUN-97.
```

The information in this directive is displayed on the screen when the file is used by CONFOR. The directive is not essential, but it provides a handy label for the file. The effect of this SHOW directive, and of the similar ones in other files, will become apparent when you run CONFOR (see Section 6). The displaying of the information allows you to verify that you ran the option that you intended to, and that you are using the latest revisions of the files.

The second line of CHARS is blank. Blank lines are generally ignored by CONFOR, and can be used freely to improve the readability of the files.

The third line is *CHARACTER LIST. This line is mandatory: it tells the program that the characters are following. The rest of the file contains the definitions of the characters.

Each character consists of a feature and a set of states. The individual characters are of five different kinds.

1. *unordered multistate* (UM): 2 or more states, with no relation of ordering between them. Examples:

```
#100. petals <colour>/
    1. white/
    2. yellow/
    3. pink/
    4. red/
#2. <longevity of plants>/
    1. annual <or biennial, without remains of old sheaths or culms>/
    2. perennial <with remains of old sheaths and/or culms>/
```

2. *ordered multistate* (OM): 2 or more states, with a relation of ordering between them. Examples:

```
#7. leaf blades <shape: data incomplete>/
    1. linear/
    2. linear-lanceolate/
    3. lanceolate/
    4. ovate-lanceolate/
    5. ovate/
    6. elliptic <oblong>/
    7. obovate/
```

```
#27. <female-fertile> spikelets <plane of compression>/
    1. compressed laterally <lying on the side when placed on a flat surface>/
    2. not noticeably compressed <terete>/
    3. compressed <dorsally, ventrally or> dorsiventrally <lying on front or back when placed on a flat
    surface>/
```

3. *integer numeric* (IN): a measurement which is always a whole number. Examples:

```
#62. stigmas <number>/
#54. <female-fertile> lemmas <number of nerves traversing mid-region>/
    nerved/
```

4. *real numeric* (RN): a measurement which may take fractional values. Example:

```
#3. <mature> culms <maximum height: data unreliable for large genera>/
    cm high/
```

5. *text* (TE): any text. Examples:

```
#101. references:/
#1. including <synonyms: ‘genera’ included in the current description>/
```

Each character is preceded by a cross hatch (#), a unique number, a period (.), and a space.

Each state is preceded by a number, a period, and a space.

The feature and states are terminated by a slash (/).

Units may be included with numeric characters. They have no state number, and are terminated by a slash.

Comments are contained within angle brackets (< >) and can appear anywhere in the text of the features and states. The left (opening) bracket must be preceded by a blank, and the right (closing) bracket must be followed by a blank or slash. These comments are used in the interactive identification program, INTKEY, but not in natural-language descriptions or keys.

Lower-case letters should be used, except for proper nouns, etc.

5.2. The items file, ITEMS

Get a copy of the sample items file by entering
```
SAMPLE ITEMS
```
and look at it with the text editor.

The first line of the file is an optional SHOW directive (see Section 5.1).

The third line is the mandatory directive *ITEM DESCRIPTIONS. This indicates that the descriptions of the items (taxa) follow.

An item is a taxon name and description in terms of the character set. Example:

```
# Cynodon <Rich.>/
1<[I]Capriola['I] Adans., [I]Dactilon['I] Vill., [I]Fibichia['I] Koel.> 2,2 3,4–60(–100) 4,2 6,2 7,1 10,1
11,2<very short>/3 12,2 13,2 14,– 15,2 16,2 18,1 19,1 20,– 26,1.7–3 27,1 28,1/2 30,1/2<[I]C.
incompletus['I]> 31,2 33,2 34,1–2 35,1 36,2 37,1 38,1/1–3 39,1<normally>/2 40<(when present)>,2
44,1 45,1 46,1/2 47,1 52,1 54,1–4 55,1 56,1 57,1 58,1 59,2 60,3 61,1 62,2 63,2 64,1 66,1 67,3 68,1
69,2<2 species> 70,1 73,1 74,1/2 76,2 78,4 83,4 85,10 86,1&2&3&4&5&6 87<this project>
88<cy01.gif>
```

The taxon name is preceded by a cross hatch and terminated by a slash.

Comments between angle brackets may be put with the taxon name. The opening bracket must be preceded by a space, and the closing one followed by a space or slash. The authority should normally be put in this comment. It is printed with the name in most contexts (but without the angle brackets). For example,

```
    # Eriochloa punctata <(L.) Desv. ex Hamilt.>/
```

becomes

  Eriochloa punctata (L.) Desv. ex Hamilt.

The description consists of character numbers and state values. Each character number is separated from its state value (or values) by a comma.

Each character with its corresponding state values (an ‘attribute’), is separated from the others by a space.

If there are alternative state values, they are separated by a slash meaning ‘or’. For example, 2/3 means state 2 or state 3.

Ranges of state values are expressed by separating the values by a dash, for example, 2–4. N.B. For unordered multistate characters (only), a range does not include values between the ends of the range. For example, for character 100 in Section 5.1, and 100,2–4 would mean ‘petals yellow to red’, that is, presumably including orange, but not pink.

If states exist in combination (as, for instance, two colours being present), then the values are separated by an ampersand (&). For example, 2&3 means state 2 and state 3. Note, however, that in most contexts the programs do not distinguish between ‘and’ (&) and ‘or’ (/) (the distinction is only made in generating natural-language descriptions). Therefore, if this distinction is essential for identification purposes, it is best to add an extra state to the character for each required ‘and’ combination, for example,

```
    #40. the incomplete <male or sterile> florets <position in spikelet>/
         1. <all> proximal to the female-fertile florets/
         2. <all> distal to the female-fertile florets/
         3. both distal and proximal to the female-fertile florets/
```

However, this method of coding may be less suitable for other purposes, such as phenetic or cladistic analysis.

Combinations, ranges, and alternatives may be combined. For example, for character 100 in Section 5.1, 100,4/4&1 would mean ‘petals red, or red and white’, and 100,3–4/1–2 would mean ‘petals pink to red, or white to yellow’.

For integer or real numeric characters, the outermost values of a range may be enclosed in parentheses, to denote rarely occurring extreme values, for example, (1.4–)1.9–2.5(–3.2) or 1(–3).

If the state of the character is unknown, then the character is omitted, or the state value coded as U.

If the character is inapplicable, the state may be coded ‘–’, for example, for petal colour on flowers without petals. Alternatively, the logical dependency between characters such as ‘presence of petals’ and ‘colour of petals’ may be specified in a *DEPENDENT CHARACTERS directive, and the character need not be coded when inapplicable (see Section 5.3).

Comments between angle brackets may be put after the character number or after state values in the description, *except* when the state value is followed by ‘–’ or ‘&’. There must be no space before the opening bracket or after the closing bracket (except for the space required between the parts of the description corresponding to different characters). For example,

```
    20,1<male>/2<female> 34,2<the upper slightly longer> 85<about>,200
```

These comments are used only in the generation of natural-language descriptions.

As it is fairly difficult to change the order of items in the file, you should put them in the required order from the start. Items added later should be inserted at the correct place. Alphabetical order is usually the best.

5.3. The specifications file, SPECS

The ‘specifications’ are directives which give CONFOR information about the characters and items.

Get a copy of the sample specifications file by entering

```
    SAMPLE SPECS
```

This file is reproduced below.

```
    *SHOW Grass Genera – specifications. 14 March 1995.
    *NUMBER OF CHARACTERS 88
    *MAXIMUM NUMBER OF STATES 15
    *MAXIMUM NUMBER OF ITEMS 14
    *DATA BUFFER SIZE 2500
    *CHARACTER TYPES 1,TE 3,RN 7,OM 8,RN 11,OM 25,TE 26,RN 27,OM 33,OM 38,IN 41,IN
    44,IN 47,OM 48,IN 51,OM 54,IN 56,OM 60,IN 62,IN 64,OM 85,IN 87–88,TE
    *NUMBERS OF STATES 7,7 11,4 12,3 13,5 18,3 23–24,4 27,3 28,4 33,3 40,3 47,3 49,3 51,3 56,3
    63–64,3 67,3 69,3 74,3 78,5 79,6 80,7 81,15 82,12 83,4 84,6 86,6
    *IMPLICIT VALUES 9,2 10,1 16,2 20,2 32,1 65,2 67,3 71–72,2 75,2 77,2
    *DEPENDENT CHARACTERS 10,2:11 16,2:17 20,2:21–24 32,2:33–38 39,1:40–43 47,1/2:48–51
    55,2:56 57,2:58–59 68,2:69 78,1:81–84 78,2:80:82–84 78,3:79–81:83–84 78,4:79–82:84 78,5:80–83
    *MANDATORY CHARACTERS 12 78–86
```

The ‘maximum number of states’ must be greater than or equal to the largest number of states in any of the characters.

The ‘maximum number of items’ must be at least as large as the actual number of items in the ITEMS file.

‘Character types’ are the codes described in Section 5.1. The default is UM.

The default for the ‘number of states’ is 2.

‘Implicit values’ are values which are assumed to apply to every item in which the character is not coded. This directive can be used to shorten natural-language descriptions by omitting common values. For example, for the character

```
    #77. <[I]Zea mays['I]>/
        1. fruiting inflorescence a massive, spatheate cob, the fruits in many rows/
        2. fruiting inflorescence not as in maize <<implicit>>/
```

state 2 has been made implicit by specifying 77,2 in the IMPLICIT VALUES directive. The comment ‘<<implicit>>’ has no effect — it is only there as a reminder to the compiler of the data. The double angle brackets make it an ‘inner’ comment, which can be omitted from generated output.

The ‘dependent characters’ directive specifies that certain character values imply that certain other characters are inapplicable. For example, consider the characters

```
    #57. lodicules <presence in female-fertile florets>/
        1. present/
        2. absent/
    #58. lodicules <of female-fertile florets, texture>/
        1. <distally> fleshy <‘cuneate’; panicoid type>/
        2. <distally> membranous <i.e. pooid type>/
    #59. lodicules <of female-fertile florets, whether hairy>/
        1. ciliate <or hairy>/
        2. glabrous/
```

If the lodicules are absent (character 57, state 2), then characters 58 and 59 are not applicable. This is expressed in the DEPENDENT CHARACTERS directive as 57,2:58–59. Character 57 is called the ‘controlling’ character, and characters 58 and 59 the ‘dependent’ characters. Because this dependency has been specified, characters 58 and 59 do not need to be coded inapplicable in any items in which character 57 has state 2. Besides this saving in coding, the dependency can be important in keys if the controlling character is variable.

The MANDATORY CHARACTERS directive merely provides a check that the specified characters are coded in every item.

6. Checking the data

The first thing you should do after preparing your data files is to check them for format errors and inconsistencies by running CONFOR. The appropriate directives file is called CHECK. This instructs CONFOR to check the files SPECS, CHARS, and ITEMS. (All CONFOR runs check the data, but CHECK is quicker, because it does nothing else.)

The sample data files are (or should be) correct, but it is worthwhile deliberately introducing some errors, to see what error messages result. This will make it easier to interpret error messages when you run your own data.

Obtain a sample directives file by entering

```
SAMPLE CHECK
```

and look at it with the text editor. Then run the program by entering

```
CONFOR CHECK
```

There should be no error messages (these are identified by a row of stars), but various other information will be displayed on the screen. Notice how some of this information originates from the SHOW directives in the various files. You will also be told that the program has terminated normally, and the name(s) of output files that have been produced. In the case of the CHECK run, the only output file is CHECK.LST (see below).

Using the text editor, introduce some errors into the files SPECS, CHARS, and ITEMS. Pay particular attention to the kinds of error that you feel *you* are likely to introduce by accident. Some possibilities are: misspelling or putting in lower case the words which identify the directives; leaving out the star in front of a directive; omitting the slash which terminates the features, states, and item names; omitting opening or closing angle brackets from comments in characters and items; omitting the space before or after angle brackets where required; omitting the cross hatch (#) before a character or taxon name; omitting the period and/or space after a character or state number; introducing discrepancies between the SPECS, CHARS, and ITEMS (for example, character types or numbers of states not in agreement); putting a space within an attribute (for example, after the comma); using invalid symbols in any contexts (for example, capital O instead of zero, lower case l instead of one, parenthesis instead of angle bracket in an attribute). (It is probably best not to try all these at once, at least for a start!)

Now run CONFOR CHECK again. The invalid lines will be displayed on the terminal, with an arrow pointing to the position of each error. Note that the program can only point to the place where it first became aware of an error. The actual cause of the error may sometimes be in a previous line. For example, a missing slash in the character list might not be detected until the next cross hatch is encountered. The program does not know where the slash should have been, but it does know that there must be a slash before the next cross hatch. Inconsistencies between files may also result in a misleading position for the error message. For example, if the SPECS file states (or implies) that a character is multistate, while the CHARS file contains a definition appropriate to a numeric character, the error message will point to the CHARS file, simply because this is read *after* the SPECS file. The program has no way of knowing which file is in error — only you know what was intended.

CONFOR will not continue to process the data after detecting errors which will seriously affect its ability to make sense of what follows. For example, if there is an error in SPECS, it will not go on to process CHARS. You will have to correct SPECS, and then run the check again.

As well as being displayed on the terminal, error messages are also stored in the file CHECK.LST. If there are numerous errors, you can view this file by entering NOTEPAD CHECK.LST. This file also contains a listing of the directives that were used in the run. The character list and items are not listed, but this can be done by inserting the directives *LIST CHARACTERS and *LIST ITEMS after the INPUT FILE SPECS directive in CHECK.

After you have finished experimenting with errors in the data, delete SPECS, CHARS, and ITEMS, and get fresh copies with SAMPLE, before going on to the next section.

7. Producing a conventional key

This is a two-step process. Firstly, CONFOR is run with a directives file TOKEY, to convert the data to the format required by the key-generation program. Secondly, the program KEY is run to produce the key.

Obtain a sample TOKEY file (see Section 4), and look at it with the text editor. It includes the following directives.

*HEADING. The heading will be printed at the tops of keys.

*EXCLUDE CHARACTERS. These characters will not be used in keys.

*USE NORMAL VALUES. For the characters specified in this directive, numeric values specified in the items as ‘extreme’ (by being enclosed in parentheses — see Section 5.2) will not be used in keys. See Section 12 for further discussion.

*CHARACTER RELIABILITIES. Characters with high reliabilities will be given preference, and tend to come out early in keys. Good characters are assessed as such by the compiler or user of the data. Reliabilities must be between 0 and 10, and the default is 5.

*KEY STATES. If they are to be used in keys, numeric characters *must* be broken up into pre-determined ranges to become multistate characters. The ranges are decided by the compiler or user of the data. For instance, a character whose values range from 4.6 to 11.9 might be divided into the three ranges 4.6–6.2/6.2–8.9/8.9–11.9. The ends of the first and last ranges can be made indefinite, thus: ~6.2/6.2–8.9/8.9~. Overlapping is allowed, for example, ~6.4/6.0–9.1/8.7~. This will allow a margin for error. This directive can also be used to combine states of multistate characters, for example, 5,1&2/3.

Once you have successfully run CONFOR, run KEY to produce the key. A directives file can be used with KEY, but is not mandatory. You should first try running it without one. You will see that the key output is stored on a file named KEY.PRT. To view it, enter NOTEPAD KEY.PRT. You will see first a tabular key with characters and states represented by numbers and letters, and second a conventional key as it would appear in a flora or fauna.

You can use a directives file, KEY, to modify the way the program generates the key. Use SAMPLE KEY to obtain a sample file. The most commonly used directives are as follows.

*RBASE varies the emphasis given to character reliabilities. The value may range from 1 (reliabilities ignored) to 5. The default is 1.4.

*VARYWT influences the number of times taxa appear in the key. The value may range from 0 to 1. If it is 0, then each taxon will appear only once; if it is 1, there is only a small penalty for variability. The default is 0.8.

*NUMBER OF CONFIRMATORY CHARACTERS specifies the maximum number of confirmatory characters used at each branch of the key. The maximum number is 4, and the default is 0.

*NO TABULAR KEY stops the production of the tabular key.

*NO BRACKETED KEY stops the production of the conventional key.

*CHARACTER RELIABILITIES works as in TOKEY (see above), except that the default values are the values previously set in TOKEY. (Many users prefer to omit the CHARACTER RELIABILITIES directive from TOKEY; in this case, the default reliability is 5.)

*EXCLUDE (or INCLUDE) CHARACTERS excludes (or includes) the specified characters. These directives override the effects of the same directives in TOKEY. Example:

```
    *EXCLUDE CHARACTERS 1 64–72 75–83
```

*EXCLUDE (or INCLUDE) ITEMS excludes (or includes) the specified items. These directives override the effects of the same directives in TOKEY. Example:

```
    *INCLUDE ITEMS 1 3–7 9 11–13
```

Try changing some of the directives in KEY, or adding directives. Then rerun KEY, and look at the effects on the resulting key. Note that you can rerun KEY without also rerunning CONFOR unless you have changed the CHARS or ITEMS files, or you want to specify different KEY STATES.

For details of the effects of various changes in the KEY directives, consult the User's guide to the DELTA System.

8. Interactive identification and information retrieval

Before you can run the interactive identification and information-retrieval program, INTKEY, you must run CONFOR to convert the data to the form required by INTKEY. This conversion requires a directives file TOINT.

Obtain sample TOINT, CNOTES, and CIMAGES files (see Section 4), and run CONFOR TOINT. Then obtain a sample INTKEY.INI file. This file contains INTKEY commands which are executed when the program is started. As this file is fairly complicated, it is best to postpone examining it until you have tried using INTKEY.

INTKEY is a Windows program. Run it from the DELTA program group in Windows 3.1, or from the Start > Programs > Delta menu in Windows 95/98/NT. The program has context-sensitive 'help', and the same information is also available in a MS-Word file INTKEY.DOC.

9. Producing a natural language description

Natural-language descriptions are produced by CONFOR, using directives files TONAT and LAYOUT.

Obtain sample TONAT and LAYOUT files (see Section 4), and look at them with the text editor. They include the following directives.

*OMIT CHARACTER NUMBERS. This causes character numbers to be omitted from the output.

*NEW PARAGRAPHS AT CHARACTERS. This breaks up the output by starting new paragraphs before the specified characters in each item description.

*LINK CHARACTERS. The directive specifies sets of characters which are to be 'linked' in the output. The parts of a description corresponding to a linked set of characters are separated by semicolons rather than by full stops, and repeated words from the 'features' (see Section 5.1) are omitted. For example, the output 'Spikelets 5 to 6 mm long. Spikelets compressed dorsiventrally. Spikelets falling with the glumes.' would become 'Spikelets 5 to 6 mm long; compressed dorsiventrally; falling with the glumes.'

*ITEM SUBHEADINGS. This places headings in front of the specified characters in each item description.

The output is on TONAT.PRT. To view it, enter NOTEPAD TONAT.PRT.

Try changing some of the TONAT directives, and observe the effects on the output.

10. Obtaining a summary of the data

A summary of the data can be obtained by running CONFOR with directives file SUMMARY. Numeric characters are expressed in terms of mean, standard deviation, maximum, and minimum, while multistate characters show the distribution of the items according to the individual states. If a KEY STATES directive is used, numeric characters are summarized in terms of the defined multistate characters.

Obtain the sample SUMMARY file, run CONFOR, and examine the output. If you have difficulty interpreting the output, compare it with the coded descriptions of the summarized items (i.e. those specified in the INCLUDE ITEMS directive).

A similar summary can also be obtained through INTKEY.

11. Adding to or changing data

Changes and additions to data files are made with an ordinary text editor. However, the program DELFOR has options which can assist you with this process by keeping your files tidy and in logical order.

Running DELFOR with directives file TIDY tidies the SPECS, CHARS, and ITEMS files. Lines in the files are made uniform in length (as far as possible), the character list is indented to improve readability, and attributes are placed in ascending order within items. The output files are new specification, characters, and items files: SPECS.NEW, CHARS.NEW, and ITEMS.NEW. To change the ‘new’ files into ‘current’ files, run MOVENEW. This also changes the ‘current’ files into ‘old’ files with extension .OLD.

If you add new characters to a list, you will want them to be in their logical places in the sequence. This as achieved by adding the characters to the *end* of the list, and then moving them to their correct places by using DELFOR with the directives file REORDER. Type in the new characters at the bottom of the file CHARS; make the corresponding adjustments to SPECS (number of characters, character types, numbers of states); and add the corresponding attributes to the ITEMS file. Then obtain the sample file REORDER, which contains a NEW CHARACTER ORDER directive. In this directive, the current character numbers are listed in what is to be their new order. For example, if character 89 has been added to the list, and is to be placed between characters 5 and 6, the required directive is *NEW CHARACTER ORDER 1–5 89 6–88. You can also reorder character states by including a NEW STATE ORDERS directive. The files which are to be reordered are specified in REFORMAT directives. It is essential to reorder all directives files which depend on the character order, such as TONAT, TOKEY, and INTKEY.INI. As with the TIDY option, the new files are given the file type .NEW. You should examine them to see whether the results seem to be what you wanted. If not, you can change the REORDER file and rerun DELFOR. When you are satisfied, rename the files with MOVENEW, and finally check them with a CONFOR TONAT run.

The programs currently have no facilities for placing the items in a new order. This can be done with the text editor, but is laborious. It is better to put new items in their required places when they are first added to the ITEMS file.

12. Hints on organization and maintenance of data

The DELTA system is not merely a device for facilitating the application of computer technology to the organization and manipulation of taxonomic data. Its use should have a constructive, critical influence on fundamental aspects of taxonomic practice. It encourages conscientious acquisition of properly comparative data (essential for rational classificatory argument), and it ensures that due attention is paid to questions of character and character-state definitions. Therefore, if you intend to prepare automated taxonomic descriptions, it is best where possible to take full advantage by using the DELTA system from the start, rather than trying to automate data already on hand.

The following hints on preparing character lists, defining character states, encoding data, and file management reflect wide experience with practical applications of DELTA and associated programs. They are directed towards minimizing encoding errors, maximizing the information content of files, saving labour, and achieving stylistically acceptable descriptions and keys.

12.1. Preparing character lists

Your character list is the first file you will need to prepare. Its quality (i.e., the taxonomic wisdom it embodies and the skill with which it is worded) will define the standard of the rest of the work.

Make sure that all the states of a character are logically related and mutually exclusive. The following are examples of kinds of ‘character’ definition to be avoided.

```
#102. leaves/
   1. green/
   2. yellow/
   3. wide/
   4. narrow/
#103. leaves/
   1. narrow/
   2. yellow/
   3. serrate/
   4. shiny/
#104. leaves/
   1. green/
   2. light green/
```

102 should be 2 characters (‘colour of leaves’ and ‘width of leaves’), and 103 should be 4 characters. The states of 104 should be ‘dark green’ and ‘light green’. A useful test is to ask yourself whether you would be happy to choose between the states as alternatives in a key. However, you must ask this question in relation to an arbitrary specimen: alternatives like those in 103 *can* work in a key when only a particular set of taxa is still to be separated, but they are still unsuitable for combining in a character definition.

For more information, see ‘Appendix 2. Why must character states be mutually exclusive?’ in the User’s guide to the DELTA Editor.

Take special care over the initial construction of the character list, with a view to minimizing the need for subsequent changes. In particular, include from the start all the likely characters you can think of, and try to order them from the beginning in a fashion which you consider logical, in the hope of avoiding subsequent major alterations in their relative positions. One needs to locate characters in the list easily, in order to find their numbers. It is exasperating having to make several passes through a long character list before encoding a titbit of information, and exasperation promotes errors.

One quickly comes to remember the code numbers of certain characters and character sequences, and it is well to foster this capability as an extra safeguard against coding errors. Extensive re-numbering is no more dangerous from this point of view than is frameshifting characters by one place. Therefore, instead of intercalating additional characters in their logical positions immediately, it is worth allowing them to accumulate at the end of the list, and conducting major re-ordering operations at relatively infrequent intervals. Remember that re-ordering of characters, and of states within characters, may require rewording of comments in the items.

Most of your character-state sequences will be a permanent feature of your character list and data. You will quickly learn to remember them, and it pays to give memory and intelligence every encouragement. To facilitate accurate coding, try to present states within characters in sequences that are logical, at least to your own mind; and be consistent, in this respect, from one character to another. For example, arrange things in ascending order of size, number or complexity, or in a sequence which has some significance for you. In expressing presence/absence, always place the positive state first: placing the negative state first can lead to (conventional) keys with odd-looking couplets.

It is sometimes necessary to allow the likely requirements of key-making to influence the form of the character list. However, any early temptation to overdo this, to the detriment of the quality of your descriptions, should be strongly resisted. In particular, try to avoid using inclusive character states, such as

```
1. leaves glabrous/
2. leaves hirsute, hispid, or hoary/
```

They may be acceptable in keys, but their presence in your character list will lead to embarrassing ‘lies’ in some of your descriptions (hirsute leaves, for example, being rendered as ‘hirsute, hispid, or hoary’). ‘2. leaves not glabrous <hirsute, hispid, or hoary>/’ would be preferable, but still unsatisfactory. If this sort of thing is unavoidable in the early stages through shortage of precise information, include a properly broken down version of the character as well, and record it as best you can from the beginning. The inclusive one may then be jettisoned later, as the data improve; or if it is useful in key-making, it can be retained for that purpose but excluded when making descriptions.

Perusal of the examples in the User’s guide to the DELTA System will indicate various ways in which imaginative use of the <comment> facility permits the development of a character list which is comprehensible in itself, and which at the same time gives presentable natural-language descriptions. Make generous use of comments and character notes (see the sample file CNOTES) throughout your character list, to qualify character and state definitions, to cover questions of applicability, to give references, etc. Consider constructing the character list and notes as a detailed glossary, with references to relevant articles, illustrations, etc.

Many users will be dissatisfied with the results of using the same character list for preparing both natural-language descriptions and conventional keys. The latter require that every character be set out in full, whereas this is unacceptably clumsy for descriptions, where the character sequence provides context. The only solution is to maintain a second version of the character list, appropriately modified for key making. For example, in the sample characters file (see also User’s guide to the DELTA System), a sequence of characters, starting from character 26, refers to the female-fertile spikelets. In most of these characters, the words ‘female-fertile’ are within angle brackets. These words are not necessary in the natural-language descriptions, because the characters appear under the subheading ‘Female-fertile spikelets’. However, in keys, the omission of these words is often misleading. A separate, key-making characters file is needed, in which such words are outside the angle brackets. The directives file TOKEY should reference this file instead of CHARS.

12.2. Formatting

If it is intended that descriptions and keys will ultimately be automatically typeset, the formatting information required for italics, etc. should be inserted in all character lists and data from the start (see examples in the User’s guide to the DELTA System).

12.3. Choosing items

INTKEY allows you to coalesce descriptions; for example, to prepare descriptions of genera from those of subgenera or species. Thus, if there is doubt or controversy about group delimitation, it is feasible to define some items at a lower taxonomic level than that generally being aimed at in the treatment. In addition to yielding descriptions serviceable for checking on the usefulness of alternative circumscriptions, this approach more readily accommodates future changes of mind, and may make your data more acceptable to a wider audience.

12.4. Encoding of data – technical aspects

Use text characters and comments to qualify and/or amplify the data. Apart from adding necessary detail and desirable polish to descriptions, these may eventually be converted into new characters, and their presence from the beginning will help minimize the inevitable but exasperating job of backtracking through specimens, slides and literature. If artfully framed, comments can help ‘humanize’ the printed descriptions. Bear in mind, though, that comments and text characters are inaccessible to many programs. Confine use of text characters to situations where data cannot be satisfactorily represented in other types of character (synonyms, references, lists of specimens seen, etc.); and to maximize their accessibility via INTKEY, try to standardize the style in which each is expressed in the descriptions. In deciding on the wording of a comment, perform the mental exercises necessary to determine how it will read in the finished product, not only *per se*, but in context with the surrounding data and comments.

DELTA is very versatile, and the same piece of information can often be expressed in several different ways. However, in the long run, some solutions will be preferable to others, depending on the uses to which the data will be put. In particular, if key-making has high priority, care should be taken to ensure that information of key-making value is not lost unnecessarily by inappropriate encoding.

For example, working at generic level with plants, one finds that as more information is gathered, more and more of the better known characters become ‘variable’ within genera. If these are habitually coded simply as variable, it becomes increasingly difficult to make acceptable keys: they become dependent upon obscure characters, which may well be ‘invariable’ merely through ignorance. So, instead of

    105,1/2

consider

    105<nearly always>,1<absent in *X. vulgaris*>

or

    105,1/2<rarely>

Try to be consistent in applying comments such as <rarely>: there will be the option later of deleting all such records for key-making purposes, should this seem desirable. When dealing with quantitative characters (IN or RN), values outside the commonly observed range should be enclosed in parentheses, for example, 41,(1–)2–10(–15). Then the extreme values can be omitted when making keys, by means of the CONFOR directive USE NORMAL VALUES.

Most users will choose, when making natural-language descriptions, to OMIT INAPPLICABLES. In this case, the wording will often be much improved by this kind of device:

    39,1/2 40<when present>,2

translating to

    Spikelets with female-fertile florets only, or with incomplete florets. The incomplete florets when present distal to the female-fertile florets.

Future programs for key making, interactive identification, and calculation of phenetic distances will be capable of using information on the relative frequencies of character states within a taxon. It is worthwhile anticipating this by incorporating this information into your comments wherever possible. Use a consistent format for such comments, to facilitate retrieval or conversion of the information.

Make sure that your DEPENDENT CHARACTERS directive is as comprehensive as possible. This will allow CONFOR to detect certain kinds of coding error, and allow KEY the greatest scope in using characters which are sometimes inapplicable.

12.5. Encoding of data – organizational aspects

Individual preferences and circumstances will largely decide the procedures by which data are acquired, transcribed, encoded and transferred to a computer file. Nevertheless, some generalisations are possible.

Any interruption resulting in a loss of concentration while transcribing, encoding or entering data into files is a major hazard. Refreshments breaks, for example, and especially ‘liquid lunches’, might best be postponed or foregone while such operations are in progress.

Obtain printouts of all newly entered or edited descriptions, and check them scrupulously against the original handwritten versions. For all but the most trivial of editing operations, this is a two-person job. The computer printout should be read aloud, to be checked against the original, handwritten version by somebody other than the person who typed it into the file. If all is going well, so that errors are few and far between, it may be necessary to test wakefulness by deliberately misreading from time to time! Some people (not necessarily from lack of motivation) seem unable to organize themselves and concentrate at the level necessary to achieve the essential standards of accuracy. They should be kept at safe distances from all aspects of data banking. When things are working smoothly, transcriptional and encoding errors should be sufficiently rare to permit conducting enquiries into individual cases, with a view to avoiding similar problems in future.

Use CONFOR to check all editing of files, and ‘TIDY’ the files regularly. This will detect format errors, and serves usefully as another form of insurance against coding mistakes. It will also ensure that your files remain in operational order. Users (i.e. taxonomists or biologists, as opposed to computer experts) find it extraordinarily depressing to have a job abort because a computer claims to have detected more than 100 errors in their data. An unnecessary rise in blood pressure may result from failure to realize that gross insults of this kind may be responses to ‘trivial’ errors in your specifications, rather than to genuine defects in your data.

Use INTKEY extensively — for example, to search for what you know to be unlikely or impossible combinations of characters. Obtain natural-language translations regularly, and make experimental conventional keys, in order to proof-read and search for nonsense. Testing keys is both entertaining, and effective in raising questions about the data and uncovering mistakes.

There is no need to become proficient in computer programming, nor even to be particularly knowledgeable about computers to become a successful user of the DELTA system; and computer-related chores should not usurp too much research time. Most taxonomists will need expert assistance in the initial stages of learning how to make files and use them, but once the system is running there should be relatively few demands on your consultant programmer. Most operations will require you to make quite simple modifications in the sample directives files which the system provides (see Section 4).

The User’s guide to the DELTA System is very comprehensive, and if used properly will be found to carry the answers to most questions which are likely to arise. However, most potential users are unfamiliar with this type of literature. They will find it intimidating at first sight, and some perseverance will be required. Note that many concepts which are hard to explain in words are easily grasped via practical examples, with which the Guide is liberally illustrated.

13. Preparing and testing your own data

The easiest way to prepare your own data and directives files is by editing the sample files. First, create a new subdirectory to hold your data (see Section 2). Then get sample SPECS, CHARS, and ITEMS files (see Section 4). All of these files will need extensive changes to be converted to your own data, but by doing this instead of starting from scratch, you reduce the risk of missing out essential information, or putting directives in the wrong order. (However, considerable flexibility is allowed in the order, and the correct place to add a new directive to an existing file can usually be guessed. Details of the required order can be found in the User’s guide to the DELTA System.)

The first file to work on is CHARS. All except the first few lines will have to be deleted, and replaced by your own character list.

Next, make the necessary changes to SPECS. Every directive will need to be changed to suit your own data. (Refer to Section 5.3 for information on the meanings of the directives.) At this stage, you can run CONFOR to check SPECS and CHARS. Obtain the sample directives file CHECKC (see Section 4), and enter CONFOR CHECKC. If errors are reported, edit the files and rerun the check. If you have difficulty interpreting the error messages, reread Section 6.

In the sample ITEMS file, all except the first few lines will have to be deleted, and replaced with your own items. Enter only one or two items before running the first check on the data (so that you will not waste too much effort if you have some misconceptions about how the data should be entered). Obtain the sample CHECK directives file, and enter CONFOR CHECK. Make any necessary corrections, and rerun the check until the data are free from errors. Add a few more items (say 5 or 10), and check again.

Now obtain a sample TOKEY file, and alter it to suit your data. (Refer to Section 7 for the meanings of the directives.) Run CONFOR TOKEY, and then KEY, as described in Section 7. Repeat this process for the other operations described in Sections 8–10. Examine the output critically, particularly the wording in the natural-language descriptions and keys, and make any necessary changes to the character list and directives files.

You should now have a sound basis for adding more items to the data, and, if necessary, more characters (see Section 11).

14. References

Dallwitz, M.J. 1974. A flexible computer program for generating identification keys. Syst. Zool. 23, 50–7.

Dallwitz, M.J. 1980. A general system for coding taxonomic descriptions. Taxon 29, 41–6. Also available at delta-intkey.com

Dallwitz, M.J., Paine, T. A., and Zurcher, E.J. 1993 onwards. User’s guide to the DELTA System: a general system for processing taxonomic descriptions. delta-intkey.com

Dallwitz, M.J., Paine, T.A., and Zurcher, E.J. 1995 onwards. User’s guide to Intkey: a program for interactive identification and information retrieval. delta-intkey.com

Dallwitz, M.J., Paine, T.A., and Zurcher, E.J. 1998. Interactive keys. In ‘Information Technology, Plant Pathology and Biodiversity’, pp. 201–212. (Eds P. Bridge, P. Jeffries, D. R. Morse, and P. R. Scott.) (CAB International: Wallingford.)

Dallwitz, M.J., Paine, T.A., and Zurcher, E.J. 1999 onwards. User’s guide to the DELTA Editor. delta-intkey.com

Partridge, T.R., Dallwitz, M.J., and Watson, L. 1986. A primer for the DELTA system on VMS, MS-DOS, and PRIMOS. CSIRO Aust. Div. Entomol. Rep. No. 38, 14 pp.

Partridge, T.R., Dallwitz, M.J., and Watson, L. 1988. A primer for the DELTA System on MS–DOS and VMS. 2nd edition. CSIRO Aust. Div. Entomol. Rep. No. 38, 17 pp.

Partridge, T.R., Dallwitz, M.J., and Watson, L. 1993 onwards. A primer for the DELTA System. delta-intkey.com

Watson, L., and Dallwitz, M.J. 1992 onwards. Grass genera of the world: descriptions, illustrations, identification, and information retrieval; including synonyms, morphology, anatomy, physiology, phytochemistry, cytology, classification, pathogens, world and local distribution, and references. delta-intkey.com

Watson, L., Dallwitz, M.J., and Johnston, C.R. 1986. Grass genera of the world: 728 detailed descriptions from an automated database. Aust. J. Bot. 34, 223–30.

Watson, L., Gibbs Russell, G.E., and Dallwitz, M.J. 1989. Grass genera of southern Africa: interactive identification and information retrieval from an automated data bank. S. Afr. J. Bot. 55, 452–63.