ARCHAEOLOGICAL CONTEXT SORTING BY COMPUTE

J.D. Wilcock.

Department of Computing
North Staffordshire Polytechnic

Abstract

The paper describes the production by computer of phase diagrams giving the relationships between archaeological contexts (trenches, walls, layers, etc.) in the sections of a large complex site excavated by several teams. Particular attention is paid to the detection and deletion of inconsistent data (it is arguable that a well-conducted archaeological site should produce consistent data, but it seems human errors do creep in unless a systematic recording method such as that of Harris is used). If all inconsistencies can be removed while the site is being excavated, then a much shorter and simpler computer program may be used.

The Problem

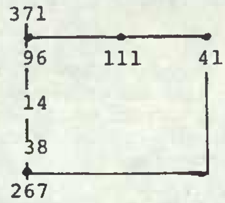
The need for machine assistance in the ordering of archaeological contexts from a large site was first indicated to me in April 1974 by York Archaeological Trust. When a site is being dug each feature is given an arbitrary "context number" and its relationship to contexts above and below it is recorded. A context may be a trench or wall, or, more frequently, just a soil layer. The observations thus yield a series of pairs of context numbers, each pair indicating the relationship between an upper and a lower context. From this information it is possible to draw a phase diagram for the site (see Figure 1 for an illustration of this process) and this is the basic tool used in the interpretation of the site. Since a large site has many hundreds of contexts, York Archaeological Trust discovered that the drawing of phase diagrams for such a site entails months of work for someone (usually the site supervisor) and is probably the greatest bottleneck in the work of the Trust. This is a job well-suited to machine methods, and the paper describes one attempt to solve the problem by means of an ALGOL 60 program. Other workers (Graham,Harris) have also tackled the same problem.

Figure 1

An illustration of a phase diagram

Initial data:

Context	371	overlies	context	96
	371	"	"	111
	371	"	"	41
	14	"	"	38
	96	"	"	14
	41	"	"	267
	38	"	"	267



Data preparation

Context relationships were recorded on forms designed by the Trust and illustrated in Figure 2. A box is provided for each numbered context, containing columns for recording the numbers of contexts lying above and below and space for a word description of the context. Additional relationships may be indicated by the "equals sign" (e.g. 47 = 131 means context 47 is the same age as context 131) and the equivalence sign (e.g. 46 ≡ 107 means the two numbers 46 and 107 have been allocated to the same layer in different parts of the site). The forms completed on site can be used to produce computer input, consisting of an identity list (pairs of context numbers allocated to the same context) and a list of context pairs each giving an upper and lower context.

Stages in the development of the methodology of context-sorting

1. Basic context-sorting algorithm

The basic context-sorting algorithm for the vertical and horizontal positioning of contexts in the phase diagram reorders context pairs into the most logical sequence and it concludes with the allocation of a unique position in the phase diagram for each context. The flowchart for this algorithm cannot be included here for reasons of space.

2. Identity between contexts

In order to give a context a unique number (in the case where several numbers have been allocated to the same context in different parts of the site), an identity list is read before input of the context pairs, and context numbers are modified as necessary.

3. Duplicate identity specifications

It is possible for the identity list to include duplicate pairs, both in the same and reverse orders. To increase efficiency, all such duplicates are erased from the identity list before context numbers are modified.

4. Erasures from the context list

It is also possible for the context list to include duplicate pairs. These are erased, since it is necessary for the relationship between a pair of contexts to be recorded once only; duplicate pairs are very common in practice, for several different site recorders may note the relationship between a given pair of contexts at different times.

It is possible to detect at this stage one form of illogicality in the context list, viz: where context pairs for both context A above context B, and context B above context A occur. Since only one of these can be correct the order which occurs second in the list is erased. These erased pairs are designated "illogical one-step links".

Finally, the operation of the identity list may give rise to context pairs where the same context occurs in both upper and lower positions (this can only occur where two layers recognised as separate in one part of the site are not so recognised in another). These "same context" links are erased.

5. Detection of other illogical sequences

It is possible for context "loops" of the form $c_1 - c_2, c_2 - c_3 \ldots\ldots c_{n-1} - c_n, c_n - c_1$ to occur for any numbers of contexts n. The "illogical one-step link" is a special case of this for n=2. A tree search was tried first to detect such loops, where all possible ramifications from a given context were set up; it was later abandoned because the run time was too long and another method of detecting illogical links was tried.

6. The effect of batch length on run time

A large site typically yields about one thousand context pairs and it was decided to investigate how long it would take to process such a large batch of data on the computer. The run time of the ALGOL 60 program was found to be proportional to the square of the number of links in a batch of context pairs. For 1000 context pairs processed in one batch the run time was estimated to be two hours, obviously impracticable.

To reduce the run time, the effect of splitting the run into a number of equal batches was investigated.

Thus:

Run time = c(number of batches)(run time for one batch in ms)

$$= c(^n/_x)(x^2) = cnx$$

where n is the number of context pairs in a run, x is the batch length and c is a constant. By experiment, c was found to have the value 7.2, hence the run time was as follows for various batch lengths:

For n = 1000 context pairs

Batch length (x)	Number of batches	Run time per batch	Total run time
1000	1	2 hrs	2 hrs
500	2	30 mins	1 hr
200	5	288 s	24 min
100	10	72 s	12 min
50	20	18 s	6 min

Batch working, while reucing the total run time, gives rise to the disadvantage that no links are possible between batches, hence some illogical links may go undetected.

7. An attempt to detect all illogical links irrespective of batch length

To detect all illogical links the program was modified to introduce all relevant links from previous batches into a batch currently being run. This, however, was found to increase the effective batch length to such an extent that the total run time became too long. This attempt was therefore abandoned.

Typical Results

A set of typical results is given in Figure 3. The actual computer output is given on the right and the resulting phase diagram at the bottom of the left hand column. It can be seen that there are a number of links which cross in the diagram, so an improvement would be to remove crossing links as far as possible by amending the layout of the diagram.

Plans for future improvements in the context sorting software

The first improvement planned is to merge batch outputs on disc and hence detect illogical links previously undetected. The phase diagrams of the individual batches will be collated in this way.

The second improvement planned is to alter the layout of phase diagrams so that as many crossing links as possible are removed.

Acknowledgements

I wish to thank Mr. J. Harradine and Mr. P. Addyman of York Archaeological Trust for first making known to me the context sorting problem, and for giving frequent advice during the development of the software. The program package has now been successfully transferred to the Computer Centre at the University of York, where it is available for day-to-day use by the Trust.

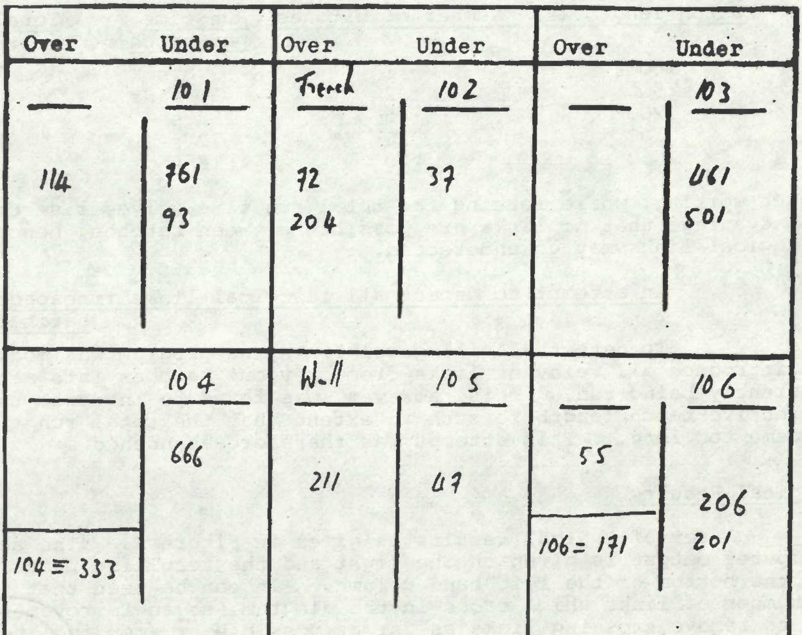
References

Graham, I. 1974 — Private communication

Harris, E.C. 1975 — 'Stratigraphic Analysis and the computer' ibid P.

Harris, E.C. 1975 — 'The stratigraphic sequence: a question of time' WORLD ARCHAEOLOGY (in press).

Figure 2. Input Form designed by York Archaeological Trust.

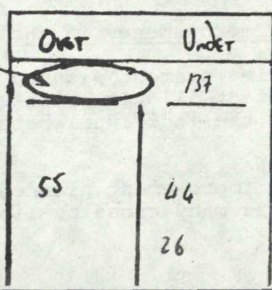


Notes.

Context 137.

Nature of Context: Trench, wall etc. written in this space. If space is blank, soil layer is assumed.

137 is over 55 and under 44 and 26.



106 = 171

Means 'same age as'

104 ≡ 333

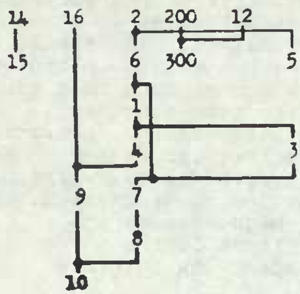
Means the two numbers refer to the same context.

Identity List

100	13	
200	100	
100	13	DUPLICATE LAYER IDENTITY
13	100	DUPLICATE LAYER IDENTITY
300	11	

Context Pairs

7	8	
9	10	
14	15	
2	5	
4	9	
3	7	
16	9	
1	4	
7	8	DUPLICATE LINK
13	11	
6	6	SAME CONTEXT
6	7	
5	8	ILLOGICAL LINK (8-1-2-5-8)
10	9	ILLOGICAL ONE-STEP LINK
1	2	ILLOGICAL LINK (2-6-1-2)
12	11	
2	1	ILLOGICAL ONE-STEP LINK
7	1	ILLOGICAL LINK (1-3-7-1)
10	1	ILLOGICAL LINK (1-4-9-10-1)
8	1	ILLOGICAL LINK (1-3-7-8-1)
6	1	
8	10	
14	15	DUPLICATE LINK
1	3	
5	5	SAME CONTEXT
2	6	
9	4	ILLOGICAL ONE-STEP LINK



DUPLICATE LAYER IDENTITY 100 13

DUPLICATE LAYER IDENTITY 13 100

13 BECOMES 100

100 BECOMES 200

11 BECOMES 300

11 BECOMES 300

DUPLICATE LINK 7 8

SAME CONTEXT 6

SAME CONTEXT 5

ILLOGICAL ONE-STEP LINK 10 9

DUPLICATE LINK 14 15

ILLOGICAL ONE-STEP LINK 9 4

ILLOGICAL ONE-STEP LINK 2 1

LOOP MEMBERS
1
2
5

ILLOGICAL LINK 5 8

LOOP MEMBERS
4
9
10

ILLOGICAL LINK 10 1

LOOP MEMBERS
3
7

ILLOGICAL LINK 7 1

LOOP MEMBERS
3
7
8

ILLOGICAL LINK 8 1

LOOP MEMBERS
6
1

ILLOGICAL LINK 1 2

Figure 3 Typical computer results, and the corresponding phase diagram.