Atti Accademia Nazionale dei Lincei

Classe Scienze Fisiche Matematiche Naturali

Rendiconti

Aldo Domenicano, Riccardo Spagna, Alessandro Vaciago

A system of crystallographic programmes for the electronic computer UNIVAC 1108. Nota I

Atti della Accademia Nazionale dei Lincei. Classe di Scienze Fisiche, Matematiche e Naturali. Rendiconti, Serie 8, Vol. **47** (1969), n.5, p. 331–336.

Accademia Nazionale dei Lincei

<http://www.bdim.eu/item?id=RLINA_1969_8_47_5_331_0>

Atti della Accademia Nazionale dei Lincei. Classe di Scienze Fisiche, Matematiche e Naturali. Rendiconti, Accademia Nazionale dei Lincei, 1969.

Strutturistica chimica. — *A system of crystallographic programmes for the electronic computer UNIVAC 1108* (*). Nota I di ALDO DOMENICANO, RICCARDO SPAGNA e ALESSANDRO VACIAGO, presentata (**) dal Socio V. CAGLIOTI.

RIASSUNTO. — Si descrive nelle sue linee generali un sistema di programmi cristallografici in Fortran V per l'elaboratore elettronico UNIVAC 1108. Si discute il problema dell'utilizzo delle memorie di massa come supporto di dati cristallografici, e se ne propone una soluzione che consente di mettere a comune tra più cristallografi un'area di tamburo magnetico a media velocità (Fastrand).

INTRODUCTION.

A system of computer programmes for routine crystal-structure analysis calculations has been developed in this laboratory, as a result of the recent installation of a UNIVAC 1108 electronic computer at the Rome University Computing Centre.

UNIVAC 1108 is a high-speed electronic computer, with access times of 750 ns for the 36-bit words of core storage and of 125 ns for the 36-bit control registers. The installation at the Rome University Computing Centre is equipped with 131 K words of core storage, 6 magnetic tapes (Uniservo VI-C), 4 high-speed drum storages (FH 432: 1,048 K words of overall capacity) and one medium-speed drum storage (Fastrand: 22,020 K words of capacity). The operating system presently used is a locally modified version of EXEC 2.

The system of programmes which will be described in the Notes of this series has been designed with careful consideration of the hardware and software peculiarities of the computer. It is particularly oriented to an extensive use of Fastrand as a data storage. The present paper is mainly concerned with a general description of the system and of its use of mass storages (1).

DEFINITIONS.

Let us define some terms extensively used throughout this paper, and which may be obscure for readers with no programming experience. Some of the definitions are from Stewart [1] (with modifications).

(*) Lavoro eseguito presso il Laboratorio di Strutturistica Chimica del CNR e l'Istituto di Chimica Farmaceutica e Tossicologica dell'Università, Città Universitaria, 00100 Roma.

(**) Nella seduta del 15 novembre 1969.

(1) A brief account on this subject has been given at the 3rd Congress of the Italian Crystallographic Association (Torino, 24-26 September, 1969).

The *core storage* is the immediate access storage of the computer. In the actual case it consists of 131,072 36–bit words, each of which can be directly addressed.

The *mass storages* are the non-immediate access storages, such as disks, drums or magnetic tapes, whose content cannot be directly addressed. The access is *sequential* for tapes, *random* for disks and drums. Mass storages may be used for storing extensive sets of data as well as sets of programmes.

A *mass storage area* is a portion of mass storage, which has been associated with a particular input-output unit in a Fortran programme.

A *record* (or *block*) is a collection of data items to be read or written on a mass storage area through a single input-output operation.

A *file* is a complete set of data for a given problem, usually composed of more than one record. Data in a mass storage area are normally arranged in files.

A *buffer* is a portion of core storage used by a programme as a temporary support of data during input-output operations. Its length is usually that of the largest record within the files to be read or written by that programme.

Structure of the system.

The system is presently composed of the following programmes, all of which have been successfully used for several months in this and other laboratories in Rome:

/CORR/ – Intensity correction programme for equi-inclination Weissenberg data;

/SCAL/ – Inter-layer scaling and Wilson plot;

/FAST/ – Structure factor calculation;

/STAM/ – Tabulation of the structure factor list in a way suitable for direct publication;

/KSTR/ – Calculation of the scale factors as $\Sigma |F_c| / \Sigma |F_o|$ for intersecting reciprocal lattice layers;

/FOUR/ – Three-dimensional Fourier synthesis including peak search and bond distance analysis;

/MNQD/ – Block-diagonal least-squares refinement of the atomic parameters;

/DIST/ – Calculation of bond distances and angles;

/CFIL/ – Control, transmission and deletion of data files on a mass storage area.

All programmes have been written entirely in Fortran V in order to allow easy modification by the user. The input-output routines involving mass storage devices make extensive use of the NTRAN subroutine, which is part of the EXEC 2 and EXEC 8 operating systems. The required programming effort has been, therefore, greatly reduced.

Except for the /CFIL/ programme, which is described later in the Appendix, all programmes in the system will be described in the following Notes of this series as soon as they are considered available for distribution outside Rome.

A number of other programmes are also being prepared, including (i) a Σ_2 programme for centrosymmetric structures, (ii) a full-matrix least-squares refinement programme, and (iii) a data reduction programme for diffractometric data. Programmes from other authors, covering less frequently used crystallographic calculations, will probably be adapted to the system in order to increase its capabilities and performances.

STRUCTURE AND USE OF THE DATA FILES.

As for other computers of similar size, the internal computing speed of UNIVAC 1108 is much faster than the speed of data transmission from or to a mass storage device. The transmission times per word for the mass storage devices presently available at the Rome University Computing Centre are of 4.2 μs for FH 432, 39.1 μs for Fastrand and 175 μs for Uniservo VI–C magnetic tapes. As much as 5.6, 52 and 233 instructions, respectively, can be executed during these times, assuming an average execution time of 0.75 μs per instruction.

In addition to transmission times, access times must be considered for records transmitted from or to mass storage devices with random access. These are, on average, 4.3 ms per record for FH 432 and 92 ms per record for Fastrand, which correspond to the execution of 5,700 and 122,000 instructions, respectively.

The problem of properly constructing and handling the input-output data files must be, therefore, carefully considered when planning a system of programmes for routine calculations. Undue waste of computing time is likely to occur if some precautions are not taken. This is particularly true if the programmes are to be used in a non-time-shared environment, such as that which exists under the control of EXEC 2, because of the lack of possibility of transmitting control from a programme to another every time access is requested to a mass storage device.

The following strategies have been adopted in the system of crystallographic programmes described here:

(i) The number of records per file has been kept to a minimum. The basic structure factor file, for instance, consists of two introductory records plus one record for each group of reflections having a common value of a Miller index.

(ii) The input-output operations have been extensively overlapped with calculation, normally through the use of the double-buffer technique. This enables a programme to process the content of a record whilst the following record is read in a different buffer area.

(iii) The number of words within each file has been reduced as much as possible. This has been achieved, firstly, by storing only the minimum amount of independent quantities [2], and, secondly, by using fixed-point variables and saving only the figures which are effectively significant. By using this policy the need of mass storage has been reduced, on average, to a little more than two 36–bit words per reflection.

(iv) Most programmes within the system save the entire structure factor set in the core storage, so that the input file is to be read only once. All calculations are carried out, therefore, directly in the core, with a considerable saving of computing time. The number of reflections which can be processed by a programme is of course limited, but we succeeded in keeping this limit at 6,000 using always no more, and often much less, than 50 K words of core storage. This should give no problems in a time-shared environment.

Organization of the data file storage.

The data files may be stored both on magnetic tapes and on Fastrand. The FH 432 may be used only as a temporary storage, since its content is lost after each run.

We have found that Fastrand is a very convenient medium for storing crystallographic data files, since (i) it avoids the costly operations of mounting and demounting tape reels, and (ii) it produces execution times which, owing to random access, are not dependent from the position of the input-output files in the available area.

The use of Fastrand is uneconomical if large data files are to be handled in sequential way. Crystallographic data files, however, need not fall in this category, particularly if they are written in the compact way described above. Our experience shows that a limited Fastrand area, say 100 K words, can accommodate a number of crystallographic data files, so that it can be used for storing original data as well as intermediate results for several structures.

The most efficient way of using such a Fastrand area is that of sharing it among several crystallographers. The problem of identifying the different files is no longer a trivial one in the case of several users, since each user must be able to read, write or delete its own files without interacting with other users. The experience within this laboratory is that many files are often created or destroyed in a very short period of time, so that it would be extremely unpractical to identify a particular file through its sequential number in the area.

(2) Typically, only h, k, l, F_o, $\sigma(F_o)$ and the observed/unobserved code are stored for each reflection in the basic structure factor file. Quantities such as $\sin\vartheta/\lambda$ are not stored, since they can be easily recalculated whenever necessary.

This problem has been solved by associating a six-character alphameric code to each file in the Fastrand area used as a data file storage (hereafter this area will be shortly referred to as the *data file storage*). The code table is reported in a second Fastrand area, whose length may be kept to a minimum (only 203 words are required for controlling up to 100 files). The content of this table is read by any programme in the system before searching for the input file, and is in turn modified and rewritten when a new file is created. This makes it possible to have access to the data file storage without knowing the positions of the requested files, since these are completely identified by the associated codes. No problem arises, therefore, if several crystallographers share the data file storage.

In order to avoid possible damage to the content of the data file storage the new files are not inserted, but are added immediately after the last file indicated by the code table. Meaningless computations are prevented by interrupting any programme in the system when one of the following situations occurs:

(i) the content of the code table is damaged;
(ii) the code of an input file is not found in the code table;
(iii) the code of the output file is already present in the code table;
(iv) the code table is complete and the output file cannot be produced.

The system includes a house-keeping programme, /CFIL/, which allows the files in the data file storage to be easily deleted or moved to or from other mass storage areas. A description of this programme is given in the Appendix.

After several months of routine work in this laboratory, the method we have just described has proved to be a very efficient way of using Fastrand as a data storage. Nevertheless, the option of identifying a file through its sequential number in a mass storage area has been maintained in all the programmes of the system.

APPENDIX (by A. Domenicano).

/CFIL/ – *A programme for controlling, transmitting and deleting data files on a mass storage area.*

This programme gives the user the possibility of performing a number of operations on the data file storage, provided this is a mass storage area (usually a Fastrand area) controlled by a code table. Namely, it is possible:

(i) to read the code table from the mass storage area where it is contained;
(ii) to clear the code table;
(iii) to modify items in the code table;
(iv) to print out the content of the code table;
(v) to copy one or more files from the data file storage to a mass storage area;

(vi) to copy one or more files from a mass storage area to the data file storage, inserting their alphameric codes in the code table;
(vii) to delete one or more files in the data file storage, closing up any gap which may remain after deletion, and erasing the codes from the code table;
(viii) to write back the code table into the mass storage area reserved for it.

All the interactions between the user and the programme take place through a very simple symbolic language, composed of eight instructions. By arranging several instructions in any logical sequence a high number of operations can be performed within a single run of the programme.

Structurally, the programme consists of two separate parts. During the first part all the instruction cards are read from the system input unit and checked for formal correctness. A list of the instruction codes with diagnostic messages, if any, is also printed out.

Control is transfered to the second part of the programme only if no formal errors have been discovered in the given instruction set. Within this part the instructions are interpreted and executed sequentially, each independently of the others. If a non-formal error is discovered, such as the request to delete a file whose alphameric code is not found in the code table, execution is interrupted and control transfered to the END instruction of the set.

The author aknowledges the use of the INOUT subroutine from the UNIVAC library. This is a Fortran subroutine which copies a file from one Fortran unit to another.

Copies of the Fortran listing of the programme, completed with a detailed operation manual, are available on request.

References.

[1] J. M. Stewart, *Novel computing techniques*, paper read at the I. U. Cr. International Summer School on Crystallographic Computing, Ottawa, 4–12 August 1969.