that they could be further developed by users. It was felt that basic training in EDP should be reshaped with more emphasis on information handling and less on flow-charting and the processing of scalars. The group also decided to circulate material through the chairman or secretary who would distribute it directly rather than start a European Quote-Quad.

Adin Falkoff of the IBM Scientific Centre, Philadelphia gave a talk on the use of APL in teaching and system design. He emphasized the inspiring effect APL produces in giving students an easy way of working with real problems. APL could be used as a substitute for Course-Writer – the advantage being that a student could be given control. He can then experiment with some functions analogously to performing a laboratory experiment.

MINIPERT designed in Assembler took 6 man years, in FORTRAN – 2 man years, in APL – 6 man months. APL tends to keep thinking simple in system design. Generality is important – APL is not committed to specific hardware or software, to a special logical system or specific data-types.

Mr. Falkoff stated that only a small part of OS had been modelled in APL. The experimental execute function implemented at the Philadelphia Scientific Centre does not execute system commands. Mr. Falkoff could not tell if or when IBM would implement an execute function with or without an ability to execute system commands.

Appendices to the report are:

I List of Participants.
II Article – "Selection of a Medium for Program Exchange". This article is reproduced below.
III APL notes for Principles of Transistors course from M.I.T.
IV An algorithm using the backspace character and the number 1167114 typeball for plotting fine curves. This will be reproduced in the algorithms section.
V A listing of the part numbers of the APL typeballs and the language BCD typeballs used with APL in various countries.

Anyone wishing for more information on this meeting should contact:

Niels Gellert, NEUCC,
Tech. University of Denmark, Building 305,
2800 Lyngby, Denmark.

Selection of a medium for program exchange (Appendix II to SEAS APL Proceedings)

Ed. Hendricks

During the SEAS–APL Working Committee meeting at NEUCC in October 1970, I was given the task of investigating the problem of "selection of a medium for program exchange". Although I am not certain what might be involved in a proper investigation, I have given the topic some thought and I now offer a rather obvious suggestion and some comments.

I propose that all APL systems provide a mechanism for reading and writing source program in ordinary EBCDIC card image records. Of course, the actual physical medium could be cards, tape, disks, or most anything else. The actual procedures for the processing of the symbolic program records would involve some new system commands, no doubt, and details would necessarily vary between unlike processors. (I presume that a variety of APL language processors will begin to appear shortly). The advantages of card image source as a medium are obvious – program exchange between unlike APL systems would be most greatly facilitated, and APL source programs could be processed (key-punched, edited, updated, copied, etc.) by existing non-APL facilities. This is clearly not a very revolutionary proposal, as practically every other language processor I can bring to mind offers some similar facility.

APL systems could also provide facilties for more efficient non-symbolic dumping and restoring of user programs (and data, perhaps) at a single installation. I think there are some important drawbacks, however, using a dump-restore facility as a medium for program exchange. The physical format of this sort of dump is usually chosen in an attempt to optimize efficiency under conditions existing in a particular system. It is probably inevitable that these conditions will vary and diversity rapidy when more than one system is involved. Adopting a standard non-symbolic dump format for a wide range of unlike systems would result in less optimum efficiency and occasional incompatibility, and so I would expect any such standard to be generally ignored. Furthermore, the potential for confusion inherent in program exchange is considerable, and so choosing an exchange medium to help reduce that confusion is most desireable. Any format which involves its own unique syntax rules can only serve to compound the confusion.

Card-image format is a simple and general medium for practically any program exchange, and it offers the best hope for easy analysis of difficulties when they arise. In any case, I don't think we can realistically expect any standard other than symbolic 80-column card format to be widely adopted.

Modifications to the APL 1130/System to Provide More Convenient Operating on a Fortran User's Machine

Part I

J. F. Clementi
IBM (Australia),
95 North Terrace,
Adelaide. 5000
South Australia

R. P. Fletcher,
School of Mathematical Sciences,
The Flinders University of South Australia,
Bedford Park. 5042
South Australia

This year APL/1130 is being introduced at Flinders University for use as a second language to FORTRAN. It is being taught to a small group of second year under-graduates, who will use it, rather than FORTRAN, for their programming in an intro-ductory Numerical Analysis course, and to other persons wishing to use APL/1130 as a second (or first) language.

Since FORTRAN is the University's main language and since the machine (an 8k, 3.6 microsecond, machine, with single disk drive, 1132 printer and 1442 card reader/punch) carries a heavy load, it is imperative that day-to-day APL computing be done via cards. This led us to look at possibilities of:

- i Improving throughput
- ii Making the system more convenient for the operators
- iii Making the system more convenient for the users