Time to think about the future directions of APL

by Graham Parkhouse

Not long after returning from APL86 in Manchester this July, I received an invitation to a one-day seminar called *APL3 – a seminar on the future directions of APL*. Now I am a comparative new-boy to this APL terminology; APL86 was the first annual APL conference I had been to. From the conference I had heard so much about APL2 and second generation APLs that I quickly discerned the implication of the APL3 in the title to the seminar. Being an engineer, I am a practical man who might be more attracted to APL3 were he persuaded that the number attached to the APL referred to the language's horsepower. But, being theoretically inclined also, I am fascinated by APL as a phenomenon and needed no encouragement to join a discussion on its merits and its future directions.

I want to tell you a little about the seminar and to share some of my thoughts about APL development, but before doing so let us consider some practical issues surrounding second generation APLs. By "second generation APL" we mean an APL implementation that includes nested arrays, i.e. arrays of arrays. With the nested arrays come many new primitive functions and operators. Additional features of most second generation APLs are the facility for using primitive operators with user-defined functions and the facility for user-defined operators. There is no generally accepted standard for second generation APLs, and all those currently being used do differ significantly from each other. This divergence in the development of APL is weighted by IBM's presence; due to their influence, APL2, which is their particular dialect, is likely to become the industry standard. What is more, "APL2" is already in popular use as the generic name for second generation APLs, just as "APL1" is being used as the generic name for first generation APLs. So the name "APL2" is likely to continue to have dual meaning in the same way as well-known brand names like "Biro" and "Hoover".

I suspect that the title of this seminar I am going to tell you about was designed to get our attention OFF APL2, being organised as it was by I.P.Sharp Associates, a company who have done so much to promote advanced computer systems and who, in the process, have promoted APL. Their own Sharp APL was the first commercially available second generation APL. But the seminar was not about Sharp APL; Graeme Robertson, of I.P.Sharp's education section, presented a survey of APL principles which embraced more aspects of APL than I had ever thought of. Graeme obviously appreciates structure, because he had cleverly organised the whole day's contents into a special ternary tree structure, subdivided to five levels, each subdivision containing three items. This resulted in 3*5 pieces of information to be delivered in three hours at an average pace of one piece every 45 seconds. You will be glad to know that he was very flexible in his delivery, taking questions and diversions in his stride, but he had to triple his rate after lunch. (Note the factor of three.) But he managed it so well that we were able to enjoy the virtuosity of his delivery as well as the quality of the content.

What did he say? He talked about parts of speech, cells and frames, parsing rules; he introduced me to the concepts of potency and weights of operators; we discussed direct definition of functions and covered some of the new primitive functions described in Iverson's *Dictionary of APL*. Graham went on to demonstrate how APL has been used to express all the well known scientific theories most concisely, and tried to demonstrate that APL was relevant to many of the popular mathematical techniques such as predicate calculus.

It was at this stage that I realised that we were not going to be introduced to the nub of APL3 (in the same way as nested arrays are the nub of APL2). Instead Graham gave us an illustrated glimpse into topics of current research interest: data representations, tensor analysis and functional analysis, finishing with an introduction to fractals, a new branch of mathematics concerned with hierarchical patterns. He emphasised the importance of computer graphics, and suggested that future usage of APL would be more graphically orientated. I was challenged by the need for mathematical tools that will help us understand and manipulate the data supporting these concepts, tools as elegant as the concepts themselves. Can APL meet this need?

Who can be sure? What is certain is that practitioners of APL are breaking new ground. APL is unique in being the first, the most highly developed, and the most used executable analytical notation. Being a symbolic analytical notation it matches our thought processes which are themselves symbolic, and being executable it gives us the benefit of experimentation; in other words a mighty tool for intellectual explorers.

I cannot forget Phil Smith's presentation to APL86 called *A Programming Language for Thoughts and Dreams*. Those who possess the Tutorial Volume of the Proceedings can enjoy experimenting with the random dot pattern illustrated in his paper. The illustration is a collection of black, red and blue dots randomly distributed over a rectangular area. Looked at normally there is no evidence of any pattern whatsoever within the picture; the rectangular boundary is the only shape, and all dots seem evenly distributed. But looked at through the coloured glasses provided, rows of steps are apparent, with the lowest steps along the top and the bottom row, rising to higher steps towards the middle row, which itself is the highest step. Without the glasses there is no evidence of these steps. But I have not mentioned Phil's purpose behind the demonstration which has so impressed me. This was to demonstrate the working of the right-hand side of our brains. When you first look at the picture through the glasses, nothing happens! You have to study it for several minutes before the steps start to take off into three dimensions. During these several minutes your brain is matching the dots seen by one eye with those seen by the other (because with the glasses each eye is seeing a different pattern) and decides that the patterns would be identical if they were not on a flat piece of paper but on rows of steps. Do not worry if you do not understand my explanation; the point is that our brains perform a very sophisticated calculation without us being conscious of what it is. We do not know what our brains are doing, but they do it, and it takes them time.

How often have you stopped grappling with a problem on your mind and turned to other things, when suddenly you present yourself with a solution to your original problem? Inspiration? Yes, but probably only after the right side of your brain had done a lot more work on the original problem while you were consciously thinking about the next one. It does seem that the right side of the brain is a parallel subconscious processor with the left, more conscious side. I am not suggesting that we all have split personalities; both sides have been designed to work harmoniously together as a team, probably with the left side dominant. Following Phil's recommendation I got hold of Betty Edwards' amazing book *Drawing on the Right Side of the Brain* (see reference) – a book that has not only taught me about how to draw, but which has demonstrated to me the potential struggle between the two halves of the brain. She explains why most of us are so bad at drawing; it is because of our impatience. In our impatience the left sides of our brains keep guiding our hands to draw stereotyped symbols for each familiar object, such as the sun with its childish rays; when we inhibit the left side, and she explains how to do this, then the right side is given a chance to display its natural artistic ability. With practice we can encourage the right side to do what it is designed to do, and stop the left side from being over-dominant.

What has this got to do with APL? A great deal if, as I suspect, our understanding of mathematics is shared between the two sides of our brains. Then we should expect to need time to assimilate ideas, time to gain sufficiently deep understanding to be able to recognise the future directions of APL. Through their pioneering, Ken Iverson and his collaborators have given us a most valuable means of expression with which we may experiment and learn. APL1 has modified the way many of us think. Developments in APL1 will lead us on further, but I believe that what we need most is to use what we have, at the same time gaining deeper insight into the problems facing us. APL development is a time-dependent process; dependent on the time it takes our subconscious minds to assimilate new ideas, which should not be hurried.

Reference:

Edwards, B. *Drawing on the right side of the Brain* J.P. Tarcher Inc., Los Angeles.