BYTE's Bits

Trees

Guy L Steele Jr, MIT Artificial Intelligence Lab,
Massachusetts Institute of Technology, Cambridge MA 02139

I think that I shall never see
A matrix lovely as a tree.
Trees are fifty times as fun
As structures a la PL/I
(Which Dijkstra claims are too baroque).
And SNOBOL's strings just can't compare
With all the leaves a tree may bear,
And COMIT strings are just a joke.
Vectors, tuples too, are nice,
But haven't the impressive flair
Of trees to which a LISP is heir.
A LISPer's life is paradise!

Many people think that JOSS
And others, too, are strictly boss;
And there are many BASIC fans
Who think their favorite language spans
All that would a user please.
Compared to LISP they're all a loss,
For none of them gives all the ease
With which a LISP builds moby trees.

RPG is just a nurd
(As you no doubt have often heard);
The record layouts are absurd,
And numbers packed in decimal form
Will never fit a base-two word
Without a veritable storm
Of gross conversions fro and to
With them arithmetic to do.
And one must allocate the field
Correct arithmetic to yield
And decimal places represent
Truncation loss to circumvent:
Thus RPG is second-rate.
In LISP one needn't allocate
(That boon alone is heaven-sent!)
The scheme is sheer simplicity:
A number's just another tree.
When numbers threaten overflow
LISP makes the number tree to grow,
Extending its significance
With classic tree-like elegance.
A LISP can generate reports,
Create a file, do chains and sorts;
But one thing you will never see
Is moby trees in RPG.

One thing the average language lacks
Is programmed use of push-down stacks.
But LISP provides this feature free:
A stack — you guessed it — is a tree.
An empty stack is simply NIL.
In order, then, the stack to fill
A CONS will push things on the top;
To empty it, a CDR will
Behave exactly like a pop.
A simple CAR will get you back
The last thing you pushed on the stack;
An empty stack's detectable
By testing with the function NULL.
Thus even should a LISPer lose
With PROGs and GOs, RETURNs and DOs,
He need his mind not overtax
To implement recursive hacks:
He'll utilize this clever ruse
Of using trees as moby stacks.
Some claim this method is too slow
Because it uses CONS so much
And thus requires the GC touch;
It has one big advantage, though;
You needn't fear for overflow.
Since LISP allows its trees to grow;
Stacks can to any limits go.

COBOL input is a shame:
The implementors play a game
That no two versions are the same.
And rocky is the FORTRAN road
One's alpha input to decode:
The FORMAT statement is to blame,
But on the user falls the load.
And FOCAL input's just a farce;
But all LISP input comes pre-parsed!
(The input reader gets its fame
By getting storage for each node
From lists of free words scattered sparse.
Its parses all the input strings
With aid of mystic mutterings;
From dots and strange parentheses,
From zeros, sevens, A's and Z's,
Constructs, with magic reckonings,
The pointers needed for its trees.
It builds the trees with complex code
With rubout processing bestowed;
When typing errors do forebode
The rubout makes recovery tame,
And losers then will oft exclaim
Their sanity to LISP is owed —
To help these losers is LISP's aim.)

The flow-control of APL
And OS data sets as well
Are best described as tortured hell.
For LISPers everything's a breeze;
They neatly output all their trees
With format-free parentheses
And see their program logic best
By how their lovely parens nest.
While others are by GOs possessed,
And WHILE-DO, CASE, and all the rest,
The LISPing hackers will prefer
With COND their programs to invest
And let their functions all recur
When searching trees in maddened quest.

Expanding records of fixed size
Will quickly programs paralyze.
Though ISAM claims to be so wise
In allocating overflow,
Its data handling is too slow
And finding it takes many tries.
But any fool can plainly see
Inherent flexibility
In data structured as a tree.

When all their efforts have gone sour
To swell fixed records, losers glower.
But list reclaimers hour by hour
By setting all the garbage free
Yield CONSequent capacity:
Thus trees indefinitely flower.
(And trees run on atomic power!)

To men of sensibility
The lesson here is plain to see:
Arrays are used by clods like me,
But only LISP can make a tree.

- The Great Quux (with apologies to Joyce Kilmer)

How this poem came to be printed
Notes by C Helmers

The above parody was found on the MIT Artificial Intelligence Laboratory's computer during a recent (July 3) visit made to Henry Baker at the University of Rochester Computer Science Department. Its content reflects the LISP orientation of our August 1979 issue, and in a humorous way summarizes the true artificial intelligence hacker's point of view about LISP as a tool.

Henry dug up an electronic view of the poem on the computer and communicated by that means my desire to make it more widely available. The poem's author, it turns out, is Guy Steele, who is presently connected with the MIT Artificial Intelligence Laboratory. I had in fact spent some time talking with Guy on a previous occasion, not knowing anything at all about his penchant for poetic parody. The poem was written in 1973.

One of Guy's major technical accomplishments to date is his recent student project at MIT: design and implementation of a LISP-machine chip in silicon.

In his letter accompanying the poem, Guy points out that probably the most obscure piece of jargon is the word "moby" used as an adjective. The etymology is a reference to Melville's whale, Moby-Dick. Thus a "moby tree" is a tree which is figuratively as large as a whale, or gigantic. Most of the other terminology referring to LISP is covered in recent BYTE issues; the references to other languages such as JOSS, RPG, FORTRAN, FOCAL, APL, the OS operating system of IBM, etc are best left undefined for the purposes of the poem.

The import of the communications network as a tool for individual computer users is signified by the practical example provided in this poem's arrangement for use in BYTE. The file containing "Trees" was publicly available to any person signing onto the MIT-AI computer. Henry Baker in particular was able to sign onto the computer from his usual location in Rochester, NY via the Arpanet, an electronic network connecting many research computers. Henry then left a "mail" message via the network for Guy at Stanford, California, where Guy was spending the summer. Guy then got in touch with me at my office by phone (also electronic). The arrangement was concluded with transmission of a physical copy to BYTE via the postal service.

Readers of BYTE who own personal computers with an RS-232 interface will soon be able to sign up for private services equivalent to the electronic mail functions used by Henry and Guy in arranging this over the Arpanet. At least 2 different companies now offer (see recent advertisements) private off-hours timesharing and networking services at relatively low rates. These are typically billed via Master Charge and VISA. One of these services, Telecommunications Corporation of America, *promises to offer a nationwide users' directory of identification numbers for its users, analogous to a phone directory. This arrival of individual-oriented digital communications-oriented networks will probably mark one of the great milestones of personal computing.* ■