WHICH WAY TO THE SQUARE?

RICHARD SABEY
Swindon, Wiltshire, England

Which is the best way for a computer program to search for regular word squares, top-down or bottom-up? This article records some statistics produced during computer searches for regular 8-squares in the Official Scrabble Players Dictionary. I look forward to similar statistics for 9-squares from those square-hunters whose 9-letter word stocks are more fully developed than my own.

My square-searching program uses two pruning techniques which it is well to bear in mind, for a program running on the same word list as mine would only produce the same statistics if it used both these techniques. My program prunes in the way Leonard Gordon described in "Bottoms Up!" in the February 1993 Word Ways. That is, a word is accepted in a row of a partially-built square only if each of the incomplete rows is fillable. I record statistics in two arrays, "accept" and "try". Every time I have accepted words for k-1 rows of the square, every word which begins (or ends, for the bottom-up run) with the appropriate k-1 letters is a candidate for the kth row. Every time I try a candidate word for the kth place, I increment try[k]. If this candidate is such that the remaining rows are all fillable, I increment accept [k]. In addition, I use a short cut pointed out by Leonard Gordon in "Significantly-Different Word Squares" in the November 1993 Word Ways.

Leonard Gordon suggested in the May 1993 Colloquy doing top-down and bottom-up searches for 8-squares. Accordingly, here are the statistics resulting from such searches on OSPD. This word list is conveniently small (26,444) so that the runs do not take long, yet large enough that two completed squares result.

	Top down	Bottom up
accept[1]	23,947	23,653
try[2]	25,218,612	45,742,681
accept[2]	2,576,610	5,029,871
try[3]	193,322,738	492,958,372
accept[3]	15,082,968	7,931,767
try[4]	109,915,147	57,065,107
accept[4]	317,915	122,210
try[5]	1,080,632	769,009
accept[5]	1,728	1,939
try[6]	2,780	3,965
accept[6]	14	34
try[7]	17	41
accept[7]	2	2

There was a clear difference in the time taken for the two runs: the bottom-up run took 61 per cent longer than the top-down run.

Bottom-up takes longer because bigram pruning is less efficient bottom-up than top-down. This means more time trying candidates for the third word in the bottom-up run than in the top-down run.

In contrast, once a set of three words has been accepted, bottom-up is more efficient. In the bottom-up run only 1.6 per cent candidate third words are accepted, whereas in the top-down run 7.8 per cent are. The bottom-up run had only about half as many acceptable sets of three words as the top-down run. It thus took less time trying to extend sets of three words to complete squares. But this does not compensate for the extra time which the bottom-up run took in finding those sets of three words in the first place. (Strangely, from the try [5] point onwards, top-down becomes more efficient again. But this phenomenon has still less effect on the total run time.)

So: top-down wins. However, testing to see whether three words can be extended to a square, given that the remaining rows are all fillable, is quicker if they are the bottom three than if they are the top three.

In "Bottoms Up!", Eric Albert says "I would not argue with the claim that it is possible to write some program that constructs word squares quicker from the top down, but I believe that any well-written sophisticated program and database package will, in general, work much more quickly from the bottom up". I would be delighted to learn of any techniques which can speed up bottom-up runs, but I am not yet convinced that, if these techniques are used, a bottom-up run would be quicker than a top-down run.