Abstract: Lossless compression algorithms of the Lempel-Ziv (LZ) family are widely used nowadays. Regarding time and memory requirements, LZ encoding is much more demanding than decoding. In order to speed up the encoding process, efficient data structures, like suffix trees, have been used. In this paper, we explore the use of suffix arrays to hold the dictionary of the LZ encoder, and propose an algorithm to search over it. We show that the resulting encoder attains roughly the same compression ratios as those based on suffix trees. However, the amount of memory required by the suffix array is fixed, and much lower than the variable amount of memory used by encoders based on suffix trees (which depends on the text to encode). We conclude that suffix arrays, when compared to suffix trees in terms of the trade-off among time, memory, and compression ratio, may be preferable in scenarios (e.g., embedded systems) where memory is at a premium and high speed is not critical.