Exercise 5.6

Friday, 29 September 2023 16:45

(i) In the i-th step of the search, the 'current position' (which occurs after the i-th comparison) moves a distance of $\frac{n}{2^i}$ (ignoring any rounding)

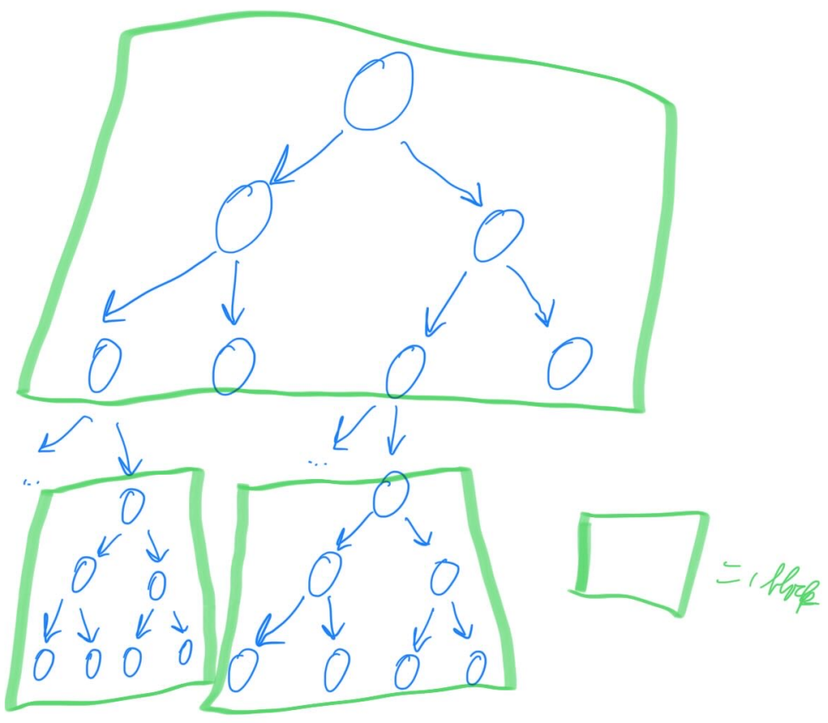
As long as $\frac{n}{2^i} > B$, all consecutive array elements (over which is being searched) are read from different blocks.

After $\frac{n}{2^i}$ becomes less than (or equal to) B, we have that the remaining elements over which the search takes place are from at most 2 blocks as $\frac{B}{2} + \frac{B}{4} + \frac{B}{8} + \ldots < B$. If the memory can store more than one block, then we would have that no more I/O's are needed after $\frac{n}{2^i} \leq B$; in other words, we would have that the number of I/O's can be bounded as $O\left(\log_2\left(\frac{n}{B}\right)\right)$.

$$\frac{n}{2^i} = B \rightarrow \frac{n}{B} = 2^i \rightarrow i = \log_2\left(\frac{n}{B}\right)$$

However, if we can only store one block in memory, then, in the worst case, we might need to read one block for each step in the search (if the boundary between the two blocks is continuously being crossed). In this case, the required number of I/O's would become $O(\log_2(n))$.

(ii)



Consider the following blocking strategy:

Let $2^s - 1 \leq B$, and choose s maximally (and integer) such that this inequality is satisfied. Then, all elements which could occur in s steps of binary search (namely: $2^s - 1$ elements) will fit in one block of size $2^s - 1$. One can put the $2^s - 1$ elements on the first s levels of the binary search tree corresponding to the array into the first block, and recursively handle the blocks in between in the same way.

Looking at the tree, we can easily see that this requires traversing $O(\log_2(n))$ levels, whereas one block needs to be read for each $\log_2(B)$ levels. All in all, this means that the total number of I/O's scales as $O\left(\frac{\log_2 n}{\log_2 B}\right) = O(\log_B n)$.

3. This solution improves spatial locality; whenever a block needs to be read (except for possibly the last one), log(B) choices can be made without reading another block. This means that data from the same block is used together more often.
   The solution does not improve temporal locality; both in the old and new strategy, each area element is used at most once. Hence, we do not need to take into account whether accesses to the same data element are clustered in time.