Bitwise expressions - solution

It can easily be shown that a greedy algorithm suffices. Having recognized this, there are probably several equivalent solutions, but I describe one. For each bit in the result, starting from the most significant, one finds out whether it may be set to one by any choice of variables. To do this, one simply considers this particular bit in each variable. If that bit has the same value in the minimum and maximum limit, then there is no choice. On the other hand, if it has different values, then one can choose to set it to zero or one. The 0-choice makes the *maximum* limit of the remaining bits 1111.... whereas the 1-choice makes the *minimum* limit of the remaining bits 0000.... To determine the choice, one has to consider the full expression. If the result bit cannot be set to one, then of course we should not do any 1-choices. On the other hand, if it can be set to one, then we should do that by as few 1-choices as possible. The trick is to see that if we have choices for several variables in the same subexpression, then it does not matter for which variable we do a 1-choice, because just having done a 0-choice for any variable, the subexpression is maximal (and thus uninteresting) for the remaining bits.

Apart from the trivial algorithm (testing all possible assignments) whose score can be discussed, one can make intermediate solutions that for example loops over all possible output values and for each such value determines bitwisely whether the expression can give this value or not (however, having got so far, I think most contestants would discover the greedy solution). Another intermediate solution is to follow the algorithm above, but make the choices recursively. The number of choices can be adjusted so that this solution gives a somewhat higher score than the trivial solution. There are also heuristic algorithms that should score some points.