MAXimal

home
algo
bookz
forum
about

added: 16 Jan 2009 0:58
Edited: 25 Apr 2012 3:11

Enumeration of all subpatterns this mask

Contents [hide]

Bust subpatterns fixed mask

Dana bitmask m. Required to effectively sort out all its subpatterns, ie such masks s, which can be included only those bits that were included in the mask m.

Immediately look at the implementation of this algorithm, based on tricks with Boolean operations:

```
int s = m;
while (s > 0) {
        ... можно использовать s ...
        s = (s-1) & m;
}
```

or by using a more compact operator for:

```
for (int s=m; s; s=(s-1)&m)
        ... можно использовать s ...
```

The only exception for the two versions of the code - subpattern is equal to zero, will not be processed. Her treatment will have to take out of the loop, or use a less elegant design, for example:

```
for (int s=m; ; s=(s-1)&m) {
        ... можно использовать s ...
        if (s==0)  break;
}
```

Let us examine why the code above is really all subpatterns this mask, with no repetitions in O (number), and in descending order.

Suppose we have a current subpattern is s, and we want to move to the next subpattern. Subtract from the mask $s$unit, thus we remove the rightmost single bit, and all the bits to the right of him to put in 1. Next, remove all the "extra" one bits that are not included in the mask m, and therefore can not be included in the subpattern. Removal operation is performed bit $\&m$. As a result, we "cut off the" mask $s-1$before the greatest importance that it may take, ie until the next subpattern following $s$in descending order.

Thus, the algorithm generates all subpatterns this mask in order strictly decreasing, spending on each transition on two elementary operations.

Particularly consider when $s = 0$. After performing $s - 1$we get the mask in which all bits are turned on (the bit representation of the number -1), and after removing the extra bit operation $(s-1) \& m$will not nothing but a mask m. Therefore, the mask $s = 0$should be careful - if time does not stop at zero mask, the algorithm may enter an infinite loop.

Through all the masks with their subpatterns. Qualification3^n

In many problems, especially in the dynamic programming masks are required to sort out all the masks, and masks for each - all subpatterns:

```
for (int m=0; m<(1<<n); ++m)
        for (int s=m; s; s=(s-1)&m)
                ... использование s и m ...
```

We prove that the inner loop will execute a total $O(3^n)$iterations.

Proof: 1 way . Consider ith bit. For him, generally speaking, there are exactly three ways: it is not included in the mask m(and therefore in the subpattern s); it is included in m, but is not included s; it enters $m$in s. Total bits n, so all the different combinations will 3^n, as required.

Proof: 2 way . Note that if the mask $m$has $k$included bits, it will have 2^k subpatterns. Since the length of the mask $n$with the $k$bits is enabled C_n^k(see. "binomial coefficients"), then all combinations will be:

$$\sum_{k=0}^{n} C_n^k 2^k.$$

Calculate this amount. To do this, we note that it is nothing like the binomial theorem expansion in the expression $(1+2)^n$, ie 3^n, as required.