Present

Let's calculate each bit in the answer separately. Suppose we want to know the value of k-th (in 0-indexation) bit in the answer. Then we can notice that we are only interested in bits from 0-th to k-th, which means that we can take all numbers modulo 2^{k+1}. After that, the sum of the two numbers can't exceed $2^{k+2} - 2$. k-th bit is 1 if and only if sum belongs to $[2^k; 2^{k+1})$ or $[2^{k+1} + 2^k; 2^{k+2} - 2]$.

So, we have to count the number of pairs of numbers that give a sum that belongs to these segments. Let's sort all numbers (taken by modulo) and make a pass with two pointers or do binary searches for each number.

Total complexity: $O(n \log n \log C)$

Bonus: can you do it in $O(n \log C)$?