

We want to determine if some function implemented on a quantum computer is satisfiable ie $f(x) = 1$ for some x from $\{0,1\}^n$. To do this, run quantum period finding on the function $f(x)$, if its period is greater than 1, then surely it is satisfiable. Period finding can be done in polynomial time $O(\text{poly}(n))$ quite fast. There are only two functions without output in $\{0,1\}$ with period one: $f(x) = 0$ and $f(x) = 1$. Once we know the period, we can know if our function is $f(x) = 0$ for all x , by simply testing a handful of values such as $f(0), f(1)$. The time to run period finding and do this small test is still polynomial in n . If there is worry about the period being too large, if say only 1 value of x gives $f(x) = 1$, then the period is 2^n , if this is an issue append an ancilla as the most significant bit to the input, but don't give it to the oracle for $f(x)$, just send the ancilla into oblivion. Then the period for such a contrived f will be half the total domain length. This seems to be exactly like the class NP, since quantum period finding is probabilistic but fast. Since SAT was solved in poly time we have that NP subset of BQP?