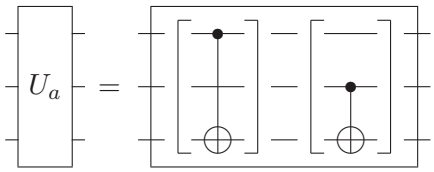
6. The Bernstein–Vazirani algorithm permits the efficient identification of members of a particular group of binary functions from n bits to 1 bit. These functions take the form $f(x) = a \cdot x$, where a is an n-bit integer identifying the particular function and the dot indicates a bitwise dot product, calculated by multiplying corresponding bits of a and x and adding the resulting bits modulo 2. For use with a reversible computer these functions can be mapped onto oracle propagators which perform the transformation

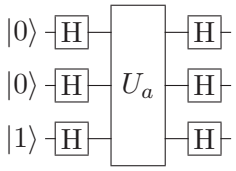
$$|x\rangle|y\rangle \longrightarrow |x\rangle|y \oplus (a \cdot x)\rangle$$

where y is an ancilla bit. For the case $n = 2$ the four possible propagators can be implemented using circuits of the general form



where the network can include either of the two bracketed sections, both of them, or neither. Give explicit circuits for each of the four possible values of a, and show how a classical reversible computer can be used to identify an unknown value of a with two oracle calls. How many oracle calls are required in the general n bit case? [7]

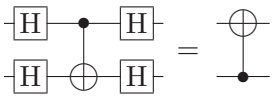
The Bernstein–Vazirani algorithm uses the quantum network



to identify a with a single oracle call. Use explicit state or matrix methods to evaluate the final state for each of the possible values of a and show how these can be identified. [10]

[You may wish to factor out the ancilla qubit to simplify calculations.]

A more efficient approach to analyse the algorithm is to use circuit identities to simplify the Bernstein–Vazirani network. Use standard network identities to show that



and hence simplify the Bernstein–Vazirani network for each value of a. Use this approach to explain why the ancilla qubit must start in $|1\rangle$, and outline why the Bernstein–Vazirani algorithm will work for any value of n. [8]