CMPT 476 Lecture 15

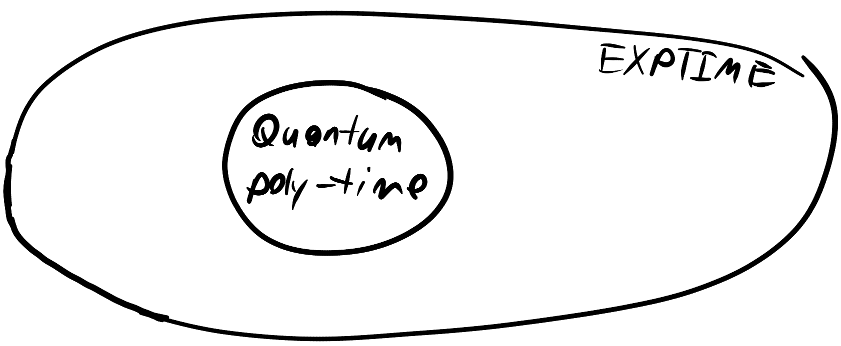
Reversible computation

Now that we have a well-defined model of what a quantum computation should be, we can ask whether we can do some things faster on a quantum computer. But first we need to narrow down the relationship of quantum and classical computing.

First question:

Can a classical (probabilistic) computer do everything a quantum one can?

Yes! Given a quantum circuit with n qubits and m gates we can simulate it classically in time $O(2^n m)$ by explicit matrix-vector multiplication and probabilistic choices for measurements (or density matrices).



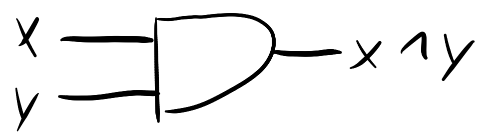
Second question:

Can a quantum computer do everything a classical computer can?

Yes! But it's a bit complicated...

(Invertibility)

Recall the classical AND gate



The AND gate has no inverse since

$$00, 01, 10 \longmapsto 0$$

Since quantum gates U are invertible ($U^{-1} = U^{+}$) and hence quantum circuits (without measurement) are invertible, we can not implement AND directly!

Second question, redux

Can we do classical computation invertibly?

(Reversible computation)

Researchers looked at this question independently of quantum computing, motivated by reducing energy use of computers. In the 60's, Landauer showed that erasing one bit of information dissipates

$$K_B T \ln 2 \quad \text{Joules} \qquad \text{(Landauer's Principle)}$$

of energy. This led to a model of reversible computing where information, and hence energy, is conserved in a computation.

How could we compute conservatively?

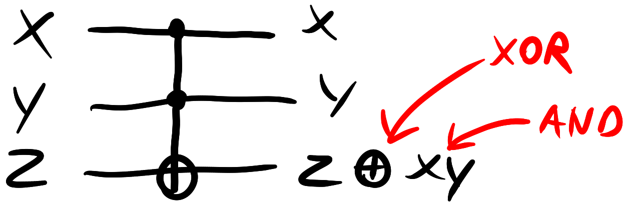
The obvious answer is whenever you compute a gate, save the inputs:

This is invertible in the sense that we can retrieve the values of x & y from the output, but it isn't conservative — we add energy to the system by adding a bit! Conversely, if we "invert" the computation, we erase the bit of information $x \wedge y$.

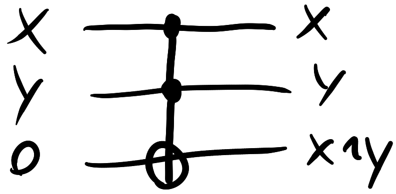
For energy to be truly conserved (and computation truly invertible/quantum) we need to account for the initial state of the bit which will store the result $x \wedge y$.

(The Toffoli gate)

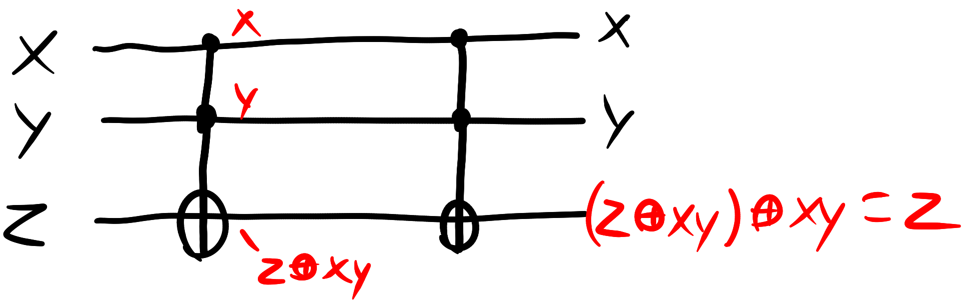
The **Toffoli gate** is a 3-bit classical gate proposed by Toffoli in 1980 as a **reversible AND**.



If $z=0$, then $z \oplus xy = xy = x \wedge y$, so we can reversibly AND two bits **into** an ancilla which we previously initialized in the 0 state



As a function of 3 bits, the Toffoli gate is also invertible (and is in fact **self-inverse**)



As a matrix,

$$
\text{Toffoli} = \begin{array}{c|c}
 & \begin{matrix} 000 & 001 & 010 & 011 & 100 & 101 & 110 & 111 \end{matrix} \\
\hline
\begin{matrix} 000 \\ 001 \\ 010 \\ 011 \\ 100 \\ 101 \\ 110 \\ 111 \end{matrix} &
\begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0
\end{bmatrix}
\end{array}
$$

Note that the Toffoli gate is a **controlled-CNOT**:

$$
\begin{aligned}
\text{Toffoli}: |0\rangle|y\rangle|z\rangle &\longmapsto |0\rangle|y\rangle|z\rangle \\
|1\rangle|y\rangle|z\rangle &\longmapsto |1\rangle|y\rangle|z\oplus y\rangle = |1\rangle(\text{CNOT}|y\rangle|z\rangle)
\end{aligned}
$$

(Universality of Toffoli (and reversible computing))

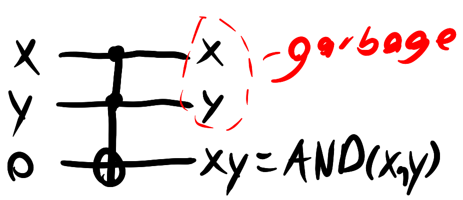
Any function $f: \{0,1\}^n \longrightarrow \{0,1\}^m$ can be implemented as a circuit over {Toffoli} with sufficiently many 0/1-initialized ancillas and some leftover garbage.

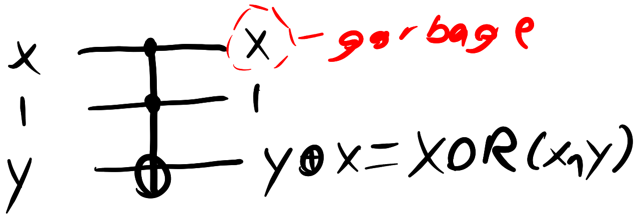
0 suffices if you add NOT which is reversible... NOT(0)=1

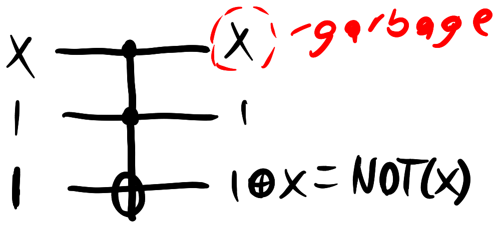
Proof Sketch

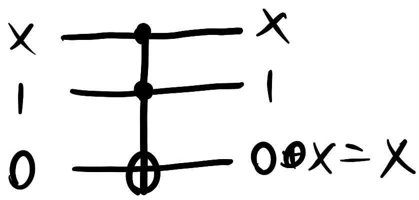
Recall that {AND, XOR, NOT, FANOUT} are univ.

We can simulate each gate with ancillas & garbage:









Hence, write f as a circuit over {AND, XOR, NOT, FANOUT} and replace each gate with the circuit above.

How much space does this use?

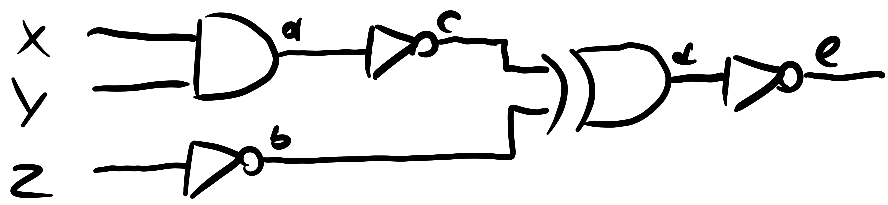
Since every circuit above uses 1-2 ancillas and leaves some garbage which can't be re-used, we simulate a circuit having T gates and S space with T gates but $O(T+S)$ space!

Moreover, all of this garbage can't be used again until the computer is "reset" somehow.

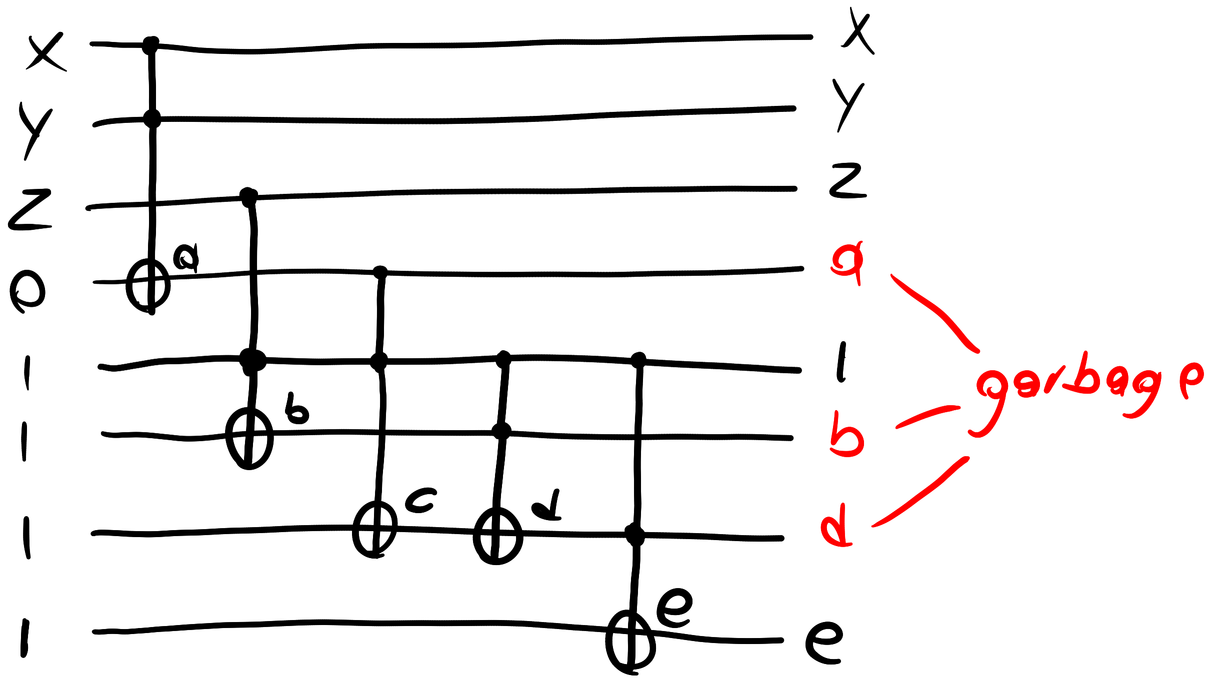
(In a quantum context, the garbage is also entangled with the state, so leaving it laying around can have unintended consequences)

Ex.

To implement the classical circuit

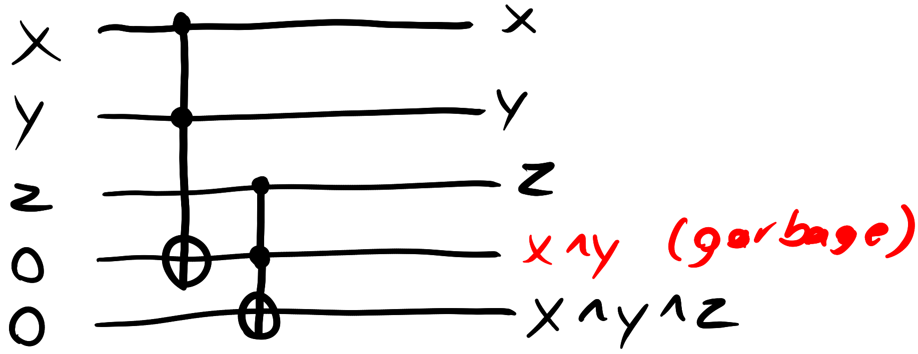


We use a Toffoli to compute each intermediate or temporary value (a, b, c, d, e)

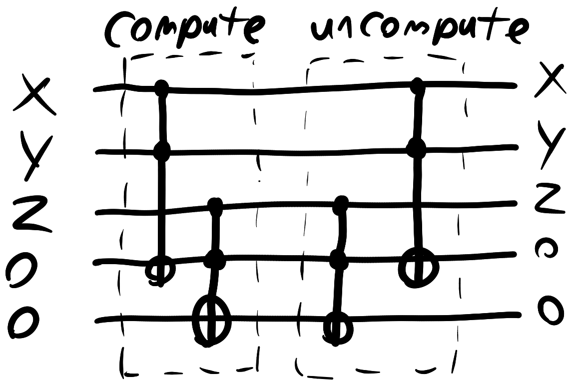


(Uncomputation)

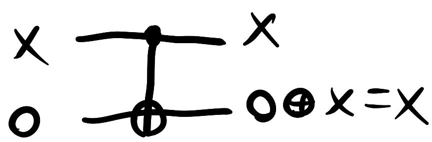
We can reversibly re-claim space by uncomputing garbage. E.g. to compute $x \wedge y \wedge z$



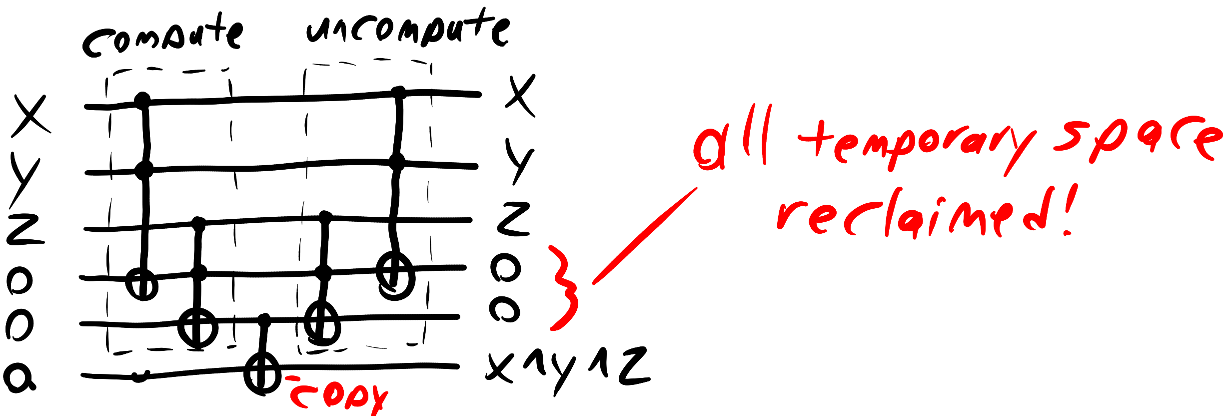
Running the circuit in reverse would return the ancilla storing $x \wedge y$ to 0 (called uncomputing) but it would also uncompute $x \wedge y \wedge z$.



To uncompute temporary values (i.e. garbage) we can copy any values we wish to save and then run in reverse. Recall that the CNOT gate is reversible and can copy a bit.

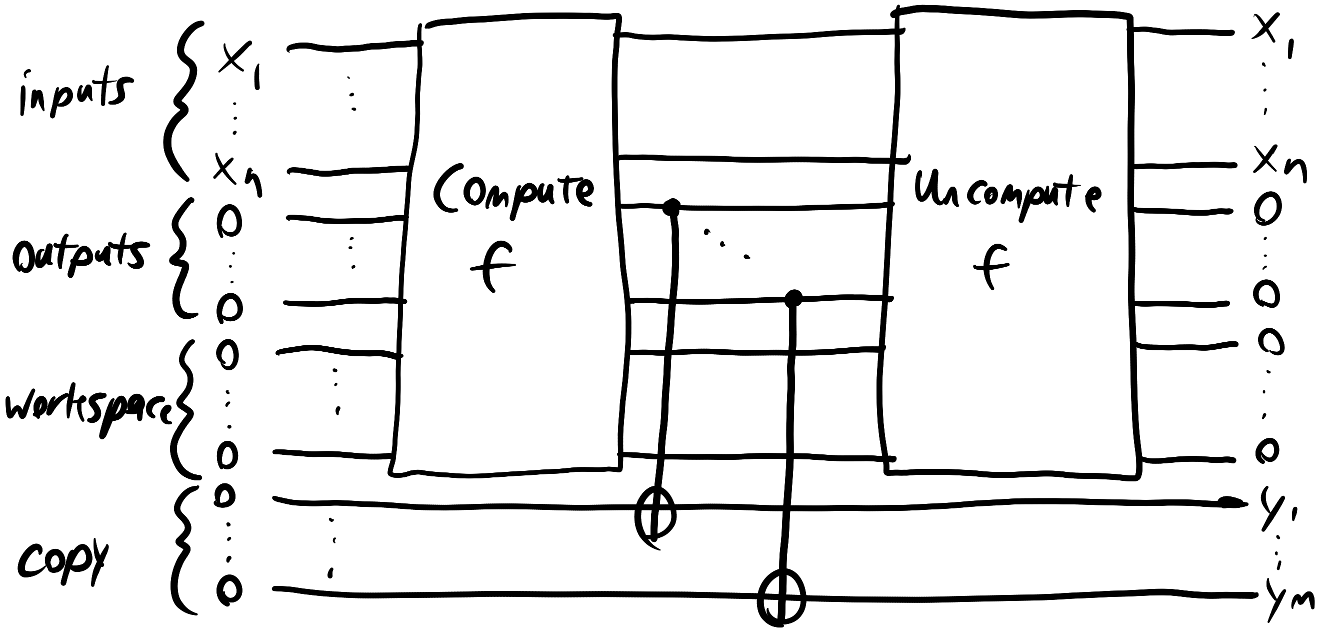


So



(Bennett's trick)

To reversibly compute a function $f:\{0,1\}^n \longrightarrow \{0,1\}^m$ with no garbage, compute f, copy, then uncompute:



If the copy register itself has initial state $|a\rangle = |a_1 a_2 \cdots a_m\rangle$, then the above circuit implements

$$|x\rangle|00\cdots0\rangle|a\rangle \longmapsto |x\rangle|00\cdots0\rangle|a\oplus f(x)\rangle$$

(Reversibility)

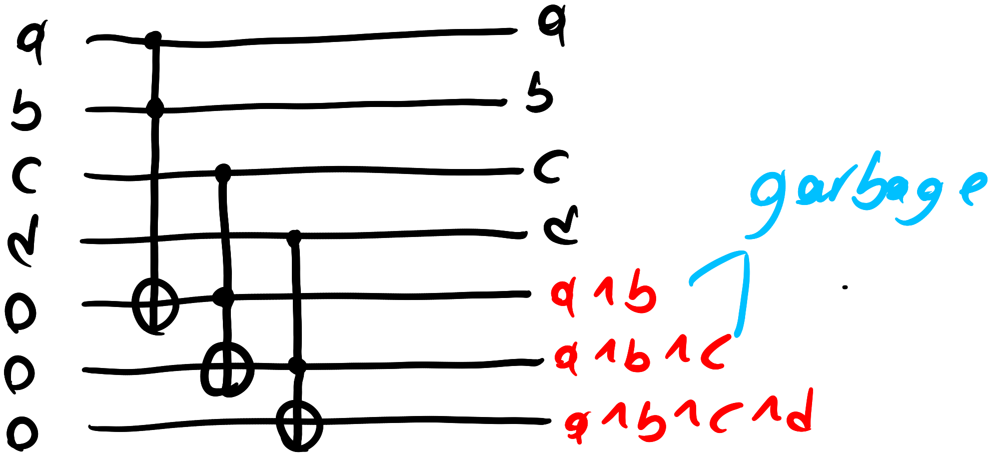
For any classical function $f:\{0,1\}^n \longrightarrow \{0,1\}^m$,

$$|x\rangle|a\rangle \longmapsto |x\rangle|a\oplus f(x)\rangle$$

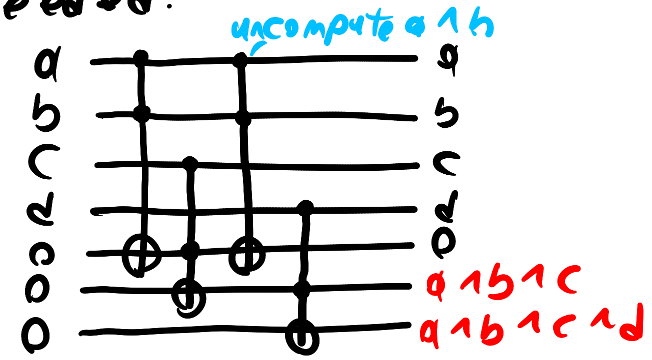
is an invertible (in fact, self inverse) transformation.

Ex.

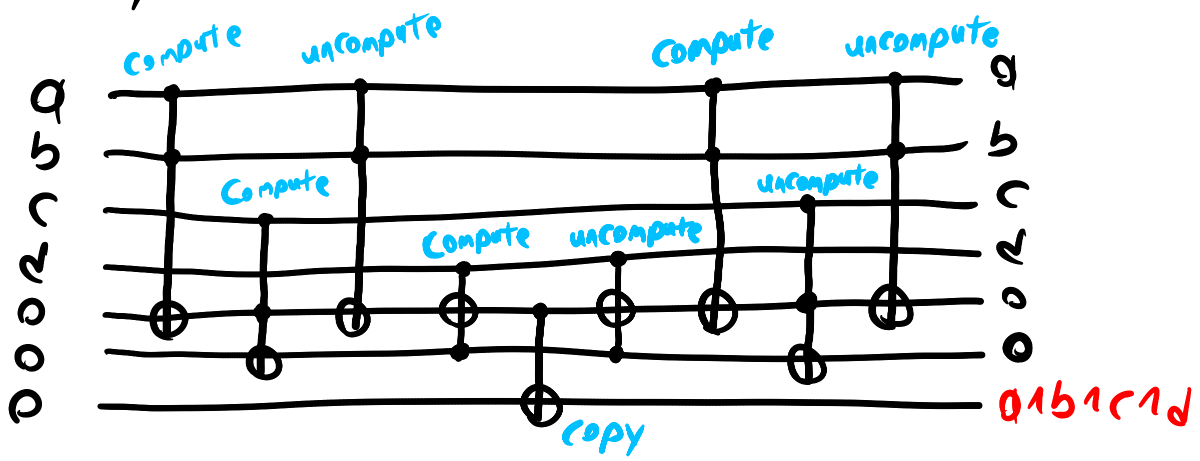
Let $f(a,b,c,d) = a \wedge b \wedge c \wedge d$. A circuit for f is



We could reclaim all space at the end of computation, in which case we would use 8 bits total and 7 gates. We could instead add an intermediate cleanup state to reclaim space once $a \wedge b$ is no longer needed:



In this case we can reuse the first ancilla to compute the result. However, to uncompute $a \wedge b \wedge c$ we actually need to recompute $a \wedge b$!



The result is only 7 bits but 9 gates.

(Bennett 1989)

An irreversible circuit with T gates and space S can be simulated by a reversible one with

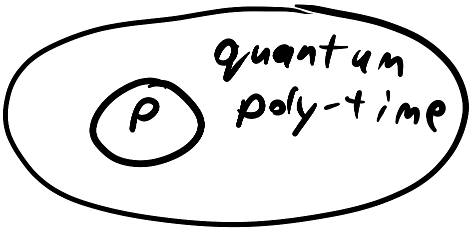
- Time $O(T^{1+\varepsilon})$ & space $O(S \log T)$
- or Time $O(T)$ & space $O(S T^{\varepsilon})$

for any $\varepsilon > 0$.

These result from different choices of when to uncompute temporary values and are called pebble games or pebbling strategies.

(Reversibility & quantum computing)

Since reversible computations are a subset of quantum computations, this tells us that quantum computers can simulate classical ones in polynomial time and space.



In general, the power of quantum computing is believed to lie somewhere between P & NP