Quantum Computation
Hilary Term 2023
Trinity College Dublin

Felix Binder &
Marco Radaelli

Problem Set 1

due: 17 February, via Blackboard

Problem 1.1: Turing Machine (4 points)

Two integer numbers n_1, n_2 are represented in *unary form*, i.e. each number is represented by a single character repeated, respectively, n_1 and n_2 times. Let `A` be that character. Let the alphabet be composed by `A` itself, the blank symbol `X`, and a special character `+`. The numbers n_1 and n_2, in unary form, are written in sequence on the tape, separated by the symbol `+`. An infinite sequence of `X` blank symbols follows after the end of n_2. For example, for $n_1 = 3$ and $n_2 = 4$ we have the input tape:

```
AAA+AAAAXXXXXXXXX...
```

1. Write explicitly a Turing Machine that retrieves the sum, in unary form, of the two numbers. In our example, the tape at the end of the calculation should give:

   ```
   AAAAAAA+XXXXXXXXXX...
   ```

   Remember to check the limiting cases $n_1 = 0$ and $n_2 = 0$.
   Optionally, you can also try to run your Turing Machine using a small script, and verify that it works correctly (no extra points for this).

2. Determine the scaling of the maximal number $T(n)$ of execution steps for an input of length n, as a function of n: is it $\mathcal{O}(n)$, or $\mathcal{O}(n^2)$, or maybe exponential? Explain your answer.

Problem 1.2: Bi-infinite Turing Machine (4 points)

Our model of the Turing Machine considered a tape that infinitely extends to the right, but not to the left. It is possible to relax this requirement, and to consider a bi-infinite tape Turing Machine instead.

1. Prove that a bi-infinite tape Turing Machine can be simulated on a right-infinite tape Turing Machine (i.e., an ordinary one). Explicitly describe the required tape structure, the internal states and the transition function.

2. Prove that the correspondence in the number of steps is linear; in other words, if an algorithm requires $\mathcal{O}(p(n))$ steps on the bi-infinite TM, with $p(n)$ polynomial, the same algorithm can run in $\mathcal{O}(p(n))$ steps on a right-infinite TM.

Problem 1.3: A satisfiability problem/1 (4 points)

Consider a logical (Boolean) expression ϕ of the form:

$$\phi(x_1, x_2, \dots, x_n) = \bigwedge_{\{i,j=1,\dots,n\}} (\tilde{x}_i \vee \tilde{x}_j) \tag{1}$$

where $\tilde{x}_i$ indicates either x_i or $\neg x_i$, and the curly brackets denote any possible values of i and j (not all possible terms need to appear). Being Boolean, the expression takes value 1 or 0 (or `TRUE` and `FALSE`). In other words, ϕ belongs to a restricted class of 3-CNF expressions, in which only two variables are allowed to appear in each clause in Conjunctive Normal Form. Now, consider in particular the following logical expression:

$$\phi(x_1, x_2, x_3, x_4) = (\neg x_1 \vee x_2) \wedge (x_2 \vee \neg x_3) \wedge (x_4 \vee \neg x_1) \wedge (x_2 \vee x_3). \tag{2}$$

Any expression of this kind can be represented with a directed graph $G(\phi)$, built according to the following rules:

- the vertices of $G(\phi)$ correspond to the variables x_i and their negations $\neg x_i$ in ϕ.
- there is a directed edge (α, β), connecting the vertices α and β, if and only if the clause $(\neg\alpha \vee \beta)$ or the clause $(\beta \vee \neg\alpha)$ is present in ϕ.

1. Draw the graph $G(\phi)$ for the logical expression ϕ defined in Eq. (2).

Problem 1.4 (Optional): A satisfiability problem/2 (4 points)

2. Now consider the satisfiability problem on a generic ϕ of form (1): is there a set of variables $x_1, \dots, x_n$ such that the expression is `TRUE`? Show that ϕ is not satisfiable if and only if there exists a variable x such that there are paths from x to $\neg x$ and from $\neg x$ to x in $G(\phi)$.
3. For a generic directed graph G with m vertices, show that it is possible to determine whether two vertices v_1 and v_2 are connected (i.e., there is a path going from one to the other) in polynomial time.
4. Using the results of the previous steps, find an algorithm, working in polynomial time w.r.t. the number of variables, to solve the satisfiability problem for expressions of the form (1).

Problem 1.5: Boolean circuits (5 points)

1. Show that the `NAND` gate is universal (`FANOUT` (a.k.a. `COPY`) and `CROSSOVER` (a.k.a. `SWAP`) are also allowed).
2. It can be shown that two-bit reversible gates are not sufficient for universal reversible computation. We have to introduce another operator acting on three input bits. Consider the `TOFFOLI` gate on three input bits (see Fig. 1), defined via the following truth table:

in_1	in_2	in_3	out_1	out_2	out_3
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	0
0	1	1	0	1	1
1	0	0	1	0	0
1	0	1	1	0	1
1	1	0	1	1	1
1	1	1	1	1	0

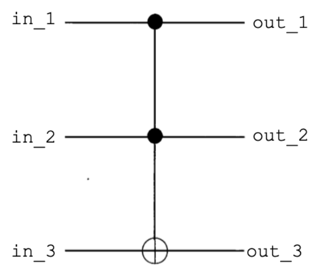


Figure 1: Symbol for the Toffoli gate.

Show that the `TOFFOLI` gate defined in this way is:

(a) reversible,

(b) universal (fixed assignments of values to bits are allowed).

Problem 1.6: A classical adder (4 points)

The full-adder circuit (see Fig. 2) receives three bits x, y, and c as input, and returns outputs c' and r.

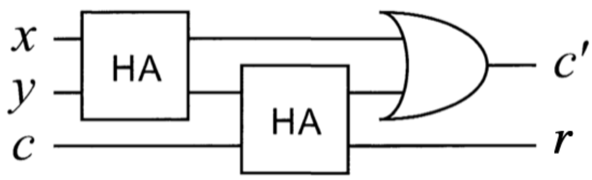


Figure 2: The full-adder (FA) circuit makes use of half-adders (HA) as subroutines.

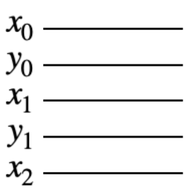
1. What does r represent in the full-adder circuit of Figure 2?

Consider two numbers x and y, with binary representation $x = x_2x_1x_0$, and $y = y_1y_0$, written using the convention:

$$x = \sum_i x_i 2^i. \tag{3}$$

2. What is the smallest number of bits required to represent x? And y? And their sum $x+y$?

3. Using half-adders, full-adders and additional logic gates, draw a circuit that computes the sum $x+y$ (in the same binary notational convention as for x and y), given as input the binary representation of x and y. Complete the scheme below, without adding ancillary bits.



Problem 1.7: Reversible computing (4 points)

Consider the following truth table for a function with two input bits and one output bit.

in_1	in_2	out
0	0	0
0	1	1
1	0	1
1	1	1

1. What gate does this function represent?
2. Show that the function is not reversible.
3. Following the same procedure considered during the lectures, build a reversible version of the gate (using additional bits as needed), and write down its truth table. Is it unique?
4. Let U_f be the reversible implementation of the gate and let k be the number of input and output bits for U_f. Write U_f as a suitable $2^k \times 2^k$ matrix, and show that it is unitary.