Optimization Problems on Quantum Computers
CEMRACS Summer School 2025
Instructor: Yassine Hamoudi
Course page: https://yassine-hamoudi.github.io/cemracs2025/

Problem Session

Solving MAX-3SAT with Grover's search in Qiskit

MAX-3SAT is a Boolean optimization problem that asks for the maximum number of clauses that can be simultaneously satisfied in a given 3-CNF formula. For instance, in the formula:

$$(\bar{x}_1 \vee x_2 \vee x_4) \wedge (\bar{x}_2 \vee x_3 \vee x_4) \wedge (x_1 \vee \bar{x}_3 \vee x_4) \wedge (x_1 \vee \bar{x}_2 \vee \bar{x}_4) \wedge$$
$$(x_2 \vee \bar{x}_3 \vee \bar{x}_4) \wedge (\bar{x}_1 \vee x_3 \vee \bar{x}_4) \wedge (x_1 \vee x_2 \vee x_3) \wedge (\bar{x}_1 \vee \bar{x}_2 \vee \bar{x}_3)$$

one can check that at most 7 out of the 8 clauses can be satisfied simultaneously. An example of a maximizing assignment is $x = x_1x_2x_3x_4 = 1101$.

Input encoding. We let n denote the number of variables and m the number of clauses. In the example above, $n = 4$ and $m = 8$. A 3-CNF formula will be represented in Python as a list F of size m, where each entry encodes a clause as a 6-tuple (i, j, k, a, b, c), defined as follows: i, j, k are the indices of the three variables in the clause, and a, b, c are Boolean values indicating whether the corresponding variables are negated or not. For instance, the clause $\bar{x}_1 \vee x_2 \vee x_4$ is represented by the tuple `(1,2,4,False,True,True)` and the formula above is encoded as:

```
F = [(1,2,4,False,True,True), (2,3,4,False,True,True), (1,3,4,True,False,True), (1,2,4,True,False,False),
  (2,3,4,True,False,False), (1,3,4,False,True,False), (1,2,3,True,True,True), (1,2,3,False,False,False)]
```

Algorithm. We aim to solve the problem using Grover's algorithm, as presented[1] in Lecture 2. Let $W_F(x) \in \{0, \dots, m\}$ be the number of clauses satisfied by $x \in \{0,1\}^n$. The algorithm is:

1. Set $x = 0 \dots 0 \in \{0,1\}^n$ and $w = W_F(x)$.
2. Repeat until no further progress is made:
   (a) Use Grover's algorithm to search for a string $x' \in \{0,1\}^n$ such that $W_F(x') > w$.
   (b) If such an x' is found, update $x = x'$ and $w = W_F(x')$.

Qiskit. We recommend using the modules `qiskit.circuit`[2] and `qiskit.circuit.library`[3], in particular the class `QuantumCircuit`[4]. For simulating quantum circuits and collecting statistics about measurement outcomes, we recommend using the Qiskit Aer simulator[5] (install the module `qiskit-aer`) and the `visualization` module[6].

[1] https://yassine-hamoudi.github.io/files/cemracs2025/Lecture2.pdf
[2] https://quantum.cloud.ibm.com/docs/en/api/qiskit/circuit
[3] https://quantum.cloud.ibm.com/docs/en/api/qiskit/circuit_library
[4] https://quantum.cloud.ibm.com/docs/en/api/qiskit/qiskit.circuit.QuantumCircuit
[5] https://qiskit.github.io/qiskit-aer/tutorials/1_aersimulator.html
[6] https://quantum.cloud.ibm.com/docs/en/api/qiskit/visualization

Grover's search

Before addressing the MAX-3SAT problem, we will familiarize ourselves with Grover's algorithm by implementing it on a simpler problem. The goal of this section is to search, among all $x \in \{0,1\}^n$, for an assignment that satisfies the clause $x_1 \vee \bar{x}_3 \vee \bar{x}_4$. We define $C : \{0,1\}^n \to \{0,1\}$ as the Boolean function that evaluates this clause on input x (i.e., $C(x) = 1 \Leftrightarrow x_1 \vee \bar{x}_3 \vee \bar{x}_4 = \texttt{True}$).

Question 1. Implement the function `oracleClause(n)` that returns a quantum circuit over $n+1$ qubits simulating the oracle

$$U_C : |x\rangle|b\rangle \mapsto |x\rangle|b \oplus C(x)\rangle$$

for all $x \in \{0,1\}^n$ and $b \in \{0,1\}$, i.e., $U_C|x\rangle|0\rangle = |x\rangle|C(x)\rangle$ and $U_C|x\rangle|1\rangle = |x\rangle|1 - C(x)\rangle$ (the second caseensures that U_C is a unitary operator). You may use the X gate controlled on 3 qubits[7]. Run the circuit on the Aer simulator and check that it returns the correct outcomes.

Grover's algorithm requires a different kind of oracle, known as a *phase-flip oracle* P_C. Instead of writing the value of C in an extra register, this oracle flips the phase of the basis state $|x\rangle$ whenever $C(x) = 1$, i.e.,

$$P_C : |x\rangle \mapsto (-1)^{C(x)}|x\rangle$$

Question 2. Show that, for any Boolean function $C : \{0,1\}^n \to \{0,1\}$, the phase-flip oracle P_C can be efficiently computed using one call to U_C, two additional single-qubit gates, and one ancilla qubit. That is, it computes $|x\rangle|0\rangle \mapsto (-1)^{C(x)}|x\rangle|0\rangle$ where the second register holds an ancilla qubit that may be used during the computation but must be restored to $|0\rangle$ at the end.

Question 3. Implement the function `phaseOracleClause(n)` that returns a quantum circuit over $n+1$ qubits simulating the phase-flip oracle $|x\rangle|0\rangle \mapsto (-1)^{C(x)}|x\rangle|0\rangle$ corresponding to the clause $C(x) = x_1 \vee \bar{x}_3 \vee \bar{x}_4$.

Grover's algorithm works by repeated application of the following operator $\mathcal{Q}$ (known as the *Grover operator*), which acts on the Hilbert space spanned by $\{|x\rangle : x \in \{0,1\}^n\}$:

$$\mathcal{Q} = H^{\otimes n}\mathcal{R}_0 H^{\otimes n} P_C.$$

This operator is composed of two layers of Hadamard gates, a reflection $\mathcal{R}_0 = 2|0\dots0\rangle\langle 0\dots0| - \mathrm{Id}$ about the all-zero state and the phase flip oracle P_C. When applied for the correct number T of iterations to the initial state $H^{\otimes n}|0\dots0\rangle$, this operators prepares the uniform superposition over all satisfying assignments:

$$\mathcal{Q}^T H^{\otimes n}|0\dots0\rangle \approx \frac{1}{\sqrt{|\{x : C(x) = 1\}|}} \sum_{x : C(x)=1} |x\rangle.$$

The correct number of iterations is on the order of $T \approx \sqrt{2^n/|\{x : C(x) = 1\}|}$. If the number of satisfying assignments is unknown, one can try increasing values $T = 1, 2, 4, 8, \dots$, and measure the state $\mathcal{Q}^T H^{\otimes n}|0\dots0\rangle$ at each step, until the measurement yields a satisfying assignment.

[7] `https://quantum.cloud.ibm.com/docs/en/api/qiskit/qiskit.circuit.library.C3XGate`

Question 4. Implement the function `groverOperatorClause(n)` that takes as input an integer $n \geq 4$ and returns a quantum circuit simulating the Grover operator for the function $C(x) = x_1 \vee \bar{x}_3 \vee \bar{x}_4$. It is recommended to use the function from Question 3 and the multi-controlled X gate[8]. You may use ancilla qubits during the computation (as in Question 2), provided they are restored to $|0\rangle$ at the end.

Question 5. Implement the function `groverClause(n)` that takes as input an integer $n \geq 4$ and returns a list $x = [x_1, \ldots, x_n]$ of binary values representing an assignment that satisfies the clause $x_1 \vee \bar{x}_3 \vee \bar{x}_4$. You must use the function from Question 4 and Grover's algorithm to search for a solution (i.e., do not simply return a valid hardcoded assignment such as $[1, 0, \ldots, 0]$).

Oracle for MAX-3SAT

We now move on to the MAX-3SAT problem. Given a 3-CNF formula represented by a list F and integer w, we define $F_w : \{0,1\}^n \to \{0,1\}$ as the Boolean function that evaluates to 1 if and only if x satisfies more than w clauses in F (i.e., $F_w(x) = 1 \Leftrightarrow W_F(x) > w$). Our goal is to implement the corresponding phase-flip oracle $|x\rangle \mapsto (-1)^{F_w(x)}|x\rangle$. If necessary, additional ancilla qubits initial in the all-zero state may be used in the circuits implemented below, provided they are guaranteed to be restored to zero at the end of the computation (this is necessary to be able to correctly combine quantum circuits together).

Question 6. Modify the code from Question 1 to write a function `oracleClause(n,C)` that takes as input an integer n and a 6-tuple C representing a clause, and returns a quantum circuit simulating the operation

$$U_C : |x\rangle|b\rangle \mapsto |x\rangle|b \oplus C(x)\rangle$$

for all $x \in \{0,1\}^n$ and $b \in \{0,1\}$, where $C(x) = 1$ if and only if the clause is satisfied by x.

Question 7. Implement the function `countClauses(n,F)` that takes as input an integer n and a 3-CNF formula represented as a list F, and returns a quantum circuit simulating the operation

$$|x\rangle|0\ldots0\rangle \mapsto |x\rangle|W_F(x)\rangle$$

where $W_F(x)$ denotes the number of clauses satisfied by x in F. You may use the `ModularAdderGate`[9].

Question 8. Implement the function `MAX3SATOracle(n,w,F)` that takes as input an integer n, an integer w and a 3-CNF formula represented as a list F, and that returns a quantum circuit simulating the phase-flip oracle

$$|x\rangle \mapsto (-1)^{F_w(x)}|x\rangle.$$

You may use the `IntegerComparatorGate`[10].

[8] https://quantum.cloud.ibm.com/docs/en/api/qiskit/qiskit.circuit.library.MCXGate
[9] https://quantum.cloud.ibm.com/docs/en/api/qiskit/qiskit.circuit.library.ModularAdderGate
[10] https://quantum.cloud.ibm.com/docs/en/api/qiskit/qiskit.circuit.library.IntegerComparatorGate

Final algorithm

Question 9. Implement the function `decisionMAX3SAT(n,w,F)` that takes as input a list F representing a 3-CNF formula over n variables and an integer w, and returns a list $x = [x_1, \ldots, x_n]$ such that x is an assignment satisfying more than w clauses of F, if such an assignment exists. Otherwise, the function should return $x = [-1, \ldots, -1]$. You must use Grover's algorithm together with the phase-flip oracle implemented in Question 8.

Question 10. By using `decisionMAX3SAT` and the Quantum Minimum Finding algorithm (as presented in Lecture 2), implement the function `MAX3SAT(n,F)` that takes as input a list F representing a 3-CNF formula over n variables, and returns a list $x = [x_1, \ldots, x_n]$ representing an assignment that solves the MAX-3SAT problem on F.

(Bonus) K-Maximum Finding

We now aim to extend the above algorithm to find the top-K assignments $x^{(1)}, \ldots, x^{(K)}$ that satisfy the largest number of clauses in a given 3-CNF formula.

Question 11. Given a quantum oracle access to an arbitrary function $W : \{0, \ldots, 2^n - 1\} \to \{0, \ldots, m\}$, describe a quantum algorithm for finding the K largest elements under W, using $O(\sqrt{K 2^n})$ quantum queries to an oracle for W.

Question 12. Using this algorithm, implement the function `topMAX3SAT(n,k,F)` that returns the top-k assignments satisfying the most clauses in a 3-CNF formula F.