CS 238 Homework

Jens Palsberg

January 2022

Catalog of matrices

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \qquad Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} \qquad Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

$$I = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \qquad H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

$$R_x(\theta) = \begin{pmatrix} \cos(\theta/2) & -i\sin(\theta/2) \\ -i\sin(\theta/2) & \cos(\theta/2) \end{pmatrix} \qquad R_y(\theta) = \begin{pmatrix} \cos(\theta/2) & -\sin(\theta/2) \\ \sin(\theta/2) & \cos(\theta/2) \end{pmatrix}$$

$$R_z(\theta) = \begin{pmatrix} e^{-i\theta/2} & 0 \\ 0 & e^{i\theta/2} \end{pmatrix}$$

$$R_\varphi = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\varphi} \end{pmatrix} \qquad S = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix} \qquad T = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{pmatrix}$$

where $\theta, \varphi \in \mathbb{R}$.

Unitary matrices

Question 1. Prov ces in the catalog above are unitary.

Question 2. Show , then $U^\dagger$ is unitary.

Question 3. Show f two unitary matrices is unitary.

Question 4. For any complex $N \times N$ matrix U, we can uniquely write $U = R + iQ$, where R and Q have real entries. Show that if U is unitary, then so is the $2N \times 2N$ matrix U' given in block form by

$$U' = \begin{pmatrix} R & Q \\ -Q & R \end{pmatrix}$$

Thus, by doubling the dimension, we can remove the need for complex-number entries. Show the result of applying this construction to the Pauli matrix Y.

Question 5. Show that the four Pauli matrices X, Y, Z, I form an orthonormal basis for the space of 2×2 matrices. Thus, we can regard the space of 2×2 matrices as a 4-dimensional complex Hilbert space.

Simulate quantum circuits

Try the quantum c[illegible]led Quirk, linked here: `https://algassert.com/quirk`. Submit thre[illegible]

Question 1. Subm[illegible] a circuit that involves at least one single-qubit gate and at least one 2-q[illegible]

Question 2. Submit a screenshot of an extension of the circuit in (1), where all the new gates are to the right of (1), and where the end result is that the entire circuit is the identity function on the input.

Question 3. Submit an explanation of how you reversed the computation performed by (1), in the sense that first (1) took the qubits from input to output, and then the second half of (2) took that output back to the input.

Hilbert spaces

Question 1. Defi

$$|0\rangle + \beta|1\rangle \mid \alpha, \beta \in \mathbb{C} \}$$

The inner product i

$$|0\rangle + \beta_2|1\rangle \,\rangle \;=\; \alpha_1^\dagger \alpha_2 + \beta_1^\dagger \beta_2$$

for all $\alpha_1, \alpha_2, \beta_1, \beta_2 \in \mathbb{C}$. Show that the inner product satisfies the following four properties:

1. $\langle \varphi \mid \varphi \rangle \geq 0$
2. $\langle \varphi \mid \varphi \rangle = 0$ if and only if $|\varphi\rangle = 0$.
3. $\langle \varphi \mid \psi \rangle = \langle \psi \mid \varphi \rangle^\dagger$.
4. $\langle \varphi \mid \lambda_1 \psi_1 + \lambda_2 \psi_2 \rangle = \lambda_1 \langle \varphi \mid \psi_1 \rangle + \lambda_2 \langle \varphi \mid \psi_2 \rangle$

for any $|\varphi\rangle, |\psi\rangle, |\psi_1\rangle, |\psi_2\rangle \in H_2$ and for any $\lambda_1, \lambda_2 \in \mathbb{C}$.

Question 2. Suppose f, g are Boolean functions on n inputs. Define $h(x) = f(x) \oplus g(x)$, where $\oplus$ denotes "exclusive or". Prove that h is always false (also written 0) if and only if f and g are the same function.

Question 3. For a Boolean string $x = x_1 \ldots x_n$, define

$$\begin{aligned} (-1)^x &= (-1)^{(x_1 + \ldots + x_n)} \\ \mathsf{XOR}(x) &= x_1 \oplus \ldots \oplus x_n \end{aligned}$$

Show that $(-1)^x = 1$ if and only if $\mathsf{XOR}(x) = 0$.

Deutsch-Jozsa and Bernstein-Vazirani, classically

The Deutsch-Jozsa
Input: a function
Assumption: f is ... lanced.
Output: 0 if f is c ... lanced.
Notation: $\{0,1\}$ is ... $\{0,1\}^n$ is the set of bit strings of length n.
Terminology:
Constant: f is constant if either f always outputs 0 or f always outputs 1.
Balanced: f is balanced if f outputs 0 on exactly half of the inputs.

The Bernstein-Vazirani problem:
Input: a function $f : \{0,1\}^n \to \{0,1\}$.
Assumption: $f(x) = a \times x + b$.
Output: a, b.
Notation: $\{0,1\}^n$ is the set of bit strings of length n, a is an unknown bit string of length n, $\times$ is inner product mod 2, $+$ is addition mod 2, and b is an unknown single bit.

The assignment: On a classical computer, in a classical language of your choice (such as C, Java, Python, etc), program solutions to the Deutsch-Jozsa problem and to the Bernstein-Vazirani problem. In each case, treat the input function f as black box that you can call but cannot inspect in any way at all. Each solution will be code that includes one or more calls of f.

Submit two files, one for each problem. Write detailed comments in the code about why it works.

Circuit identities

Let U be a 2×2 uni[illegible]ntrolled-U is a two-qubit gate, written $C(U)$, which when applied to qu[illegible] defined by:

$C(U)[q_1, q_2]\ |$ [illegible] $\ldots |k_n\rangle \;=\; |k_1\rangle \ldots |k_{q_1}\rangle \ldots (U^{k_{q_1}} |k_{q_2}\rangle) \ldots |k_n\rangle$

where q_1 is the cont[illegible] the target qubit, and where every $k_i \in \{0, 1\}$. The matrix representatio[illegible]cation to two qubits is:

$$C(U) \;=\; \begin{pmatrix} I & \mathbf{0} \\ \mathbf{0} & U \end{pmatrix}$$

where I is the 2×2 identity matrix and $\mathbf{0}$ is the 2×2 matrix in which every entry is 0. Notice that CNOT $= C(X)$, where X is one of the Pauli matrices.

Define SWAP to be the two-qubit gate that swaps the states of two qubit registers:

$$\mathrm{SWAP}[q_1, q_2]\ |k_1 \ldots k_{q_1} \ldots k_{q_2} \ldots k_n\rangle \;=\; |k_1 \ldots k_{q_2} \ldots k_{q_1} \ldots k_n\rangle$$

where every $k_i \in \{0, 1\}$.

The assignment: Prove the following properties of controlled gates:

1. $\mathrm{SWAP}[q_1, q_2] = C(X)[q_1, q_2]\ C(X)[q_2, q_1]\ C(X)[q_1, q_2]$.
2. $C(X)[p, q] = H[q]\ C(Z)[p, q]\ H[q]$.
3. $C(Z)[p, q] = C(Z)[q, p]$.
4. $H[p]\ H[q]\ C(X)[p, q]\ H[p]\ H[q] = C(X)[q, p]$.
5. $C(X)[p, q]\ X[p]\ C(X)[p, q] = X[p]\ X[q]$.
6. $C(X)[p, q]\ Y[p]\ C(X)[p, q] = Y[p]\ X[q]$.
7. $C(X)[p, q]\ Z[p]\ C(X)[p, q] = Z[p]$.
8. $C(X)[p, q]\ X[q]\ C(X)[p, q] = X[q]$.
9. $C(X)[p, q]\ Y[q]\ C(X)[p, q] = Z[p]\ X[q]$.
10. $C(X)[p, q]\ Z[q]\ C(X)[p, q] = Z[p]\ Z[q]$.

Simon and Grover, classically

Simon's problem:
Input: a function [illegible].
Assumption: there [illegible] uch that
[illegible] $f(y)]$ iff $[(x + y) \in \{0^n, s\}]$.

Output: s.
Notation: $\{0,1\}^n$ i[illegible]gs of length n, s is an unknown bit string of length n, $=$ is comparison of bit strings of length n, $+$ is pointwise addition mod 2 of bit strings of length n, and 0^n is a bit string of length n with all 0.

Grover's problem:
Input: a function $f : \{0,1\}^n \rightarrow \{0,1\}$.
Output: 1 if there exists $x \in \{0,1\}^n$ such that $f(x) = 1$, and 0 otherwise.
Notation: $\{0,1\}^n$ is the set of bit strings of length n.

The assignment: On a classical computer, in a classical language of your choice (such as C, Java, Python, etc), program a solution to Simon's problem and to Grover's problem. Treat the input function f as black box that you can call but cannot inspect in any way at all. Each solution will be code that includes one or more calls of f.

Submit two files, one for each problem. Write detailed comments in the code about why it works.

Quantum states

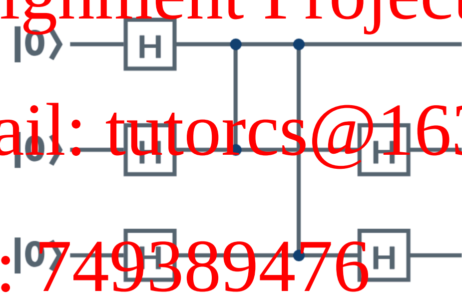
For a ket $|k_1 \dots k_n\rangle$, … s from left to right, starting with index 1.

Question 1. For … suppose we measure the qubit with index 2 in the standard basis and … ulting state. Justify your answer.

$$\ldots |01\rangle + \frac{1}{4}|10\rangle - \frac{\sqrt{7}}{4}|11\rangle$$

Question 2. Suppose we apply $H^{\otimes 3}$ to the state $|101\rangle$, after which we measure the two qubits with indexes 1,2 in the standard basis. What is the probability that we get 11 ?

Question 3. Consider the following circuit with three qubits.



Here H is the Hadamard gate, while each 2-qubit connection is CZ $= C(Z)$.

Suppose that at the end we measure all three qubits in the standard basis. What is the probability that we will get 000 ? Justify your answer.

Question 4. Consider the following state.

$$\frac{1}{2}|01\rangle - \frac{1}{2}|10\rangle + \frac{1}{\sqrt{2}}|11\rangle$$

Suppose we measure the qubit with index 1 in the standard basis. What is the probability of getting 0, and if that happens, what is the state of the other qubit? Also, suppose we measure the qubit with index 2 in the standard basis. What is the probability of getting 1, and if that happens, what is the state of the other qubit?

Deutsch-Jozsa in a simulator

The homework calle[illegible] *Bernstein-Vazirani, classically* defined the Deutsch-Jozsa problem.

Implement four [illegible]Jozsa algorithm and run the implementations on the quantum circuit sim[illegible] linked here: `https://algassert.com/quirk`. The first two implementa[illegible]$n = 1$, with one implementation using an oracle for a constant function an[illegible]ntation using an oracle for a balanced function. The next two implementations should be for $n = 2$, with one implementation using an oracle for a constant function and the other implementation using an oracle for a balanced function.

Submit screenshots that show your implementations and illustrate that they work, and submit a file with an explanation of how it works.

Quantum algorithms

Question 1. Show [illegible] the Deutsch-Jozsa algorithm works for the case of f, where $f(0) = f(1) =$ [illegible]

Question 2. For t[illegible]d a function f where

$$\begin{array}{lllll} f(000) & [illegible] & & & [illegible]0 \\ f(001) & = & f(011) & = & [illegible]01 \end{array} \qquad \begin{array}{lllll} f(100) & = & f(110) & = & 011 \\ f(111) & = & f(101) & = & 111 \end{array}$$

give two different examples of equations that the first step of Simon's algorithm may produce. Explain what those equations mean.

Question 3. Show, step-by-step, that Grover's algorithm works for the case of 2 qubits and a function f where $f(01) = 1$ and $f(00) = f(10) = f(11) = 0$.

Grover in a simulator

The homework calle[illegible]r, *classically* defined Grover's problem.

Implement three [illegible]orithm and run the implementations on the quantum circuit simulator cal[illegible]re: `https://algassert.com/quirk`. The first two implementations sh[illegible]th one implementations using an oracle for the case where $f(00) = 1$, an[illegible]ntation using an oracle for the case where $f(11) = 1$. The third implemen[illegible] $n = 3$, using an oracle for the case where $f(101) = 1$.

Submit screenshots that show your implementations and illustrate that they work, and submit a file with an explanation of how it works.

Four algorithms in Cirq

In Cirq, implement [illegible]lgorithm, the Bernstein-Vazirani algorithm, Simon's algorithm, and Grov[illegible]e detailed comments in the code about why it works. Run the programs o[illegible]ite a report that covers the following three points.

1. Design and eva[illegible]

- Present the d[illegible]plemented the black-box function U_f. Assess how easy to read it is.
- Present the design of how you parameterized the solution in n.
- Discuss the number of lines and percentage of code that your four programs share. Assess how well you succeeded in reusing code from one program to the next.
- Discuss your effort to test the four programs and present results from the testing. Report on the execution times for different choices of U_f and discuss what you find.
- What is your experience with scalability as n grows? Present a diagram that maps n to execution time.

2. Instructions

- Present a README file that describes how to input the function f, how to run the program, and how to understand the output.

3. Cirq

- List three aspects of quantum programming in Cirq that turned out to be easy to learn and list three aspects that were difficult to learn.
- List three positives and three negatives of the documentation of Cirq.

Submit five files, one for each program and one with the report.

QAOA and Shor in Cirq

In Cirq, implement [illegible]gorithm. Write detailed comments in the code about why it works. Run t[illegible] simulator. Write a report that covers the following two points.

1. Design and eva[illegible]

- Present the de[illegible]arameterized the solution in n.
- Discuss your effort to test the two programs and present results from the testing.
- What is your experience with scalability as n grows? Present a diagram that maps n to execution time.

2. Instructions

- Present a README file that describes how to provide input, how to run the program, and how to understand the output.

Submit three files, one for each program and one with the report.

Implement a quantum circuit simulator

On a classical comp[illegible] language of your choice (such as C, Java, Python, etc), implement a q[illegible]lator for a subset of QASM programs. The subset of QASM is defined [illegible]ven on the course webpage, under *Modules*, in the section *Quantum A*[illegible]*QASM)*.

Write a report t[illegible]ing two points.

1. Design and evaluation

- Present the design of how you parameterized the solution in n.
- Discuss your effort to test your simulator and present results from the testing.
- What is your experience with scalability as n grows? Present a diagram that maps n to execution time.

2. Instructions

- Present a README file that describes how to provide input, how to run the program, and how to understand the output.

Submit two files, one with your program and one with the report.

Run on a quantum computer

Port your Cirq imple[illegible]ch-Jozsa, Bernstein-Vazirani, Simon, Grover, QAOA and Shor's algorithn[illegible]r implementations on the IBM quantum computer. Modify the program[illegible] the restrictions of the quantum computer.

Write a report t[illegible]ing three points.

1. **Evaluation**

   - Discuss your effort to test the programs and present results from the testing. Run each program multiple times and present statistics of the results.
   - What is your experience with scalability as n grows? Present a diagram that maps n to execution time.
   - Compare your results from running Cirq programs on the IBM quantum computer with your results from running those same Cirq programs on the simulator.

2. **Instructions**

   - Present a README file that describes how to provide input, how to run the program, and how to understand the output.

3. **Experience**

   - Which modifications of the programs did you have to make when moving from the simulator to the quantum computer?

Submit seven files, one for each program and one with the report.