1 Various Questions

1.1 Assume that one qubit technology (A) has four hundred thousand (400000) times shorter coherence time than qubit technology (B). Technology A also has eighty (80) times shorter average gate time than technology (B). Derive an expression for the relative circuit depth between technology A and B. Present all your calculations. (3 p)

1.2 Discuss the role of High Performance Computers in the compilation tool chain for quantum computers. For what optimization passes would it make sense to use high performance resources? (2 p)

1.3 What impact can the quantum control hardware have on gate fidelity? (4 p)

1.4 Explain how a compiler could improve the measurement noise of a quantum circuit. (4 p)

1.5 Explain why input state preparation, where many qubits are entangled, is difficult in the NISQ era. (3 p)

2 Compilers

2.1 Consider the simple grammar:

$\langle id \rangle$ ::= [a-zA-Z][a-zA-Z0-9]*

$\langle int \rangle$ ::= [0-9]+

$\langle function \rangle$ ::= $\langle id \rangle$ | $\langle function \rangle$ `o` $\langle function \rangle$

$\langle exp \rangle$ ::= $\langle int \rangle$ | $\langle id \rangle$ | $\langle exp \rangle$ `*` $\langle exp \rangle$ | $\langle exp \rangle$ `+` $\langle exp \rangle$ | $\langle function \rangle$ $\langle exp \rangle$ | `(` $\langle exp \rangle$ `)`

Rewrite the grammar to resolve ambiguity in the following way:

- All binary operators and function application should be left-associative.
- Function composition '`o`' should be right-associative.
- The precedence order from tightest to weakest binding should be:
  `o` > function application > `*` > `+`.

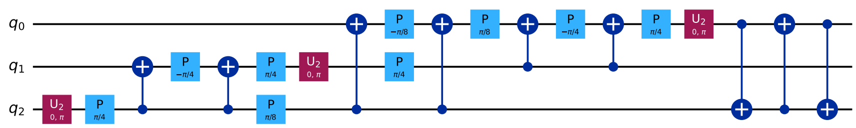
Explain how the revisions resolve the ambiguity in the original grammar in the desired way. (3 p)

2.2 Derive the syntax tree for the expression "`f o g(a * b * c + g 3*2)`" using your disambiguated grammar. Show the intermediate steps. (4 p)

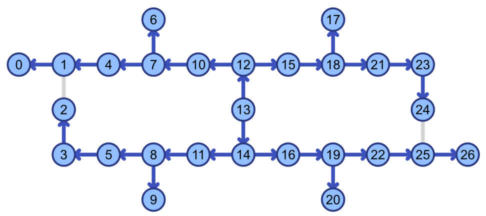
2.3 Which parts of the syntax tree would be ambiguous in the original grammar? Show that the syntax tree obeys the desired associativity and precedence rules. (2 p)

3 Quantum Transformations

Consider the following quantum circuit, which implements a 3-qubit quantum Fourier transform (QFT_3):



Assume the program has been allocated Qubits 0, 1, and 4 on the IBM Falcon R4[1] architecture shown below:



You are now to apply program transformations needed to execute the QFT_3 program on the allocated physical sub-architecture. Notice that the topology is directional, i.e., a CNOT with control p_i and target p_j is allowed only if there is a directed edge from p_i to p_j.

3.1 Make an initial assignment of logical qubits q_1, q_2, q_3 to physical qubits p_0, p_1, p_4 that allows the first two CNOT operations to be performed directly. (2 p)

3.2 Insert appropriate SWAP operations to allow the full program to execute, and draw the resulting circuit. Is the number of SWAP operations minimal? (4 p)

3.3 Now implement the SWAP operations with CNOTs that comply with the directionality of the architecture by using Hadamard gates. Draw the resulting circuit. (3 p)

[1]Image from: https://www.ibm.com/quantum/blog/whole-device-entanglement