Bernstein-Vazirani Algorithm

Episode 29. Deutsch-Jozsa Algorithm

Episode 30. Bernstein-Vazirani Algorithm

Episode 31. Simon's Algorithm

Statement of the Problem

Suppose that we are given a binary function $f : \{0, 1\}^n \to \{0, 1\}$, the value of which is determined by a secret string s of n bits

$$f : x \mapsto x \cdot s \pmod 2.$$

The Bernstein-Vazirani problem finds the secret bit string s by making queries to the oracle f.

Classically, one needs n queries to the function f to infer the secret string s.

Quantum Implementation

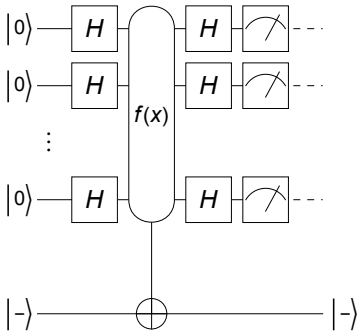


Figure 1. A quantum circuit for the Deutsch-Jozsa algorithm.

1. The first set of Hadamard gates:

$$\left|0\right\rangle \mapsto \frac{1}{2^{n/2}} \sum_{x=0}^{2^n-1} \left|x\right\rangle$$

2. The quantum oracle:

$$\frac{1}{2^{n/2}} \sum_{x=0}^{2^n-1} \left|x\right\rangle \mapsto \frac{1}{2^{n/2}} \sum_{x=0}^{2^n-1} \left|x\right\rangle \left(-1\right)^{f(x)} = \frac{1}{2^{n/2}} \sum_{x=0}^{2^n-1} \left|x\right\rangle \left(-1\right)^{x \cdot s},$$

where $x \cdot y$ denotes the bitwise inner product

$$x \cdot y := x_1\, y_1 + x_2\, y_2 + \cdots + x_n\, y_n \pmod 2,$$

3. The second set of Hadamard gates:

$$\frac{1}{2^{n/2}} \sum_{x=0}^{2^n-1} |x\rangle (-1)^{x\cdot s} \to \frac{1}{2^{n/2}} \sum_{y=0}^{2^n-1} |y\rangle \sum_{x=0}^{2^n-1} (-1)^{x\cdot s + x\cdot y} = \frac{1}{2^{n/2}} \sum_{y=0}^{2^n-1} |y\rangle \sum_{x=0}^{2^n-1} (-1)^{x\cdot (s\oplus y)} = |s\rangle$$

Example

Throughout this example, symbol **S** will be used to denote qubits and Pauli operators on them. Similarly, symbol c will be used to denote complex-valued coefficients.

In[•]:=
```
Let[Qubit, S]
Let[Complex, c]
```

Consider a secrete string of bits. The task is to find the secrete string.

In[•]:=
```
string = {0, 1};
```

In the Bernstein-Vazirani algorithm, the value of the classical oracle *f* is determined by the given secrete string.

In[•]:=
```
Clear[f];
f[x_List] := {Mod[x.string, 2]}
```

For example, `f[{1,1}]=0·1 + 1·1=1`.

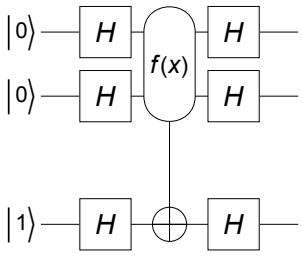
In[•]:=
```
f[{1, 1}]
```

Out[•]=
```
{1}
```

Here is a quantum circuit of the Bernstein-Vazirani algorithm. The final Hadamard gate on the third qubit is not necessary, and we put it here to make the output state more readable.

In[•]:=
```
cc = {1, 2};
tt = {3};
all = {1, 2, 3};
qc = QuantumCircuit[Ket[S[3] → 1, S@all],
  S[all, 6], Oracle[f, S@cc, S@tt], S[all, 6],
  "Invisible" → S@{2.5}]
```

Out[•]=



```
In[•]:= out = ExpressionFor[qc];
        ProductForm[out, {S@cc, S@tt}]
Out[•]=
        |01⟩ ⊗ |1⟩
```

Here is the secrete string successfully retrieved.

```
In[•]:= answer = out[S@cc]
Out[•]=
        {0, 1}
```

Summary

Keywords

- Oracle
- Decision making
- Bernstein-Vazirani problem

Functions

- `Oracle`

Related Links

- Section 4.2 of the Quantum Workbook (2022, 2023).
- Tutorial: Bernstein-Vazirani Algorithm