Quantum Oracle: Definition

Classical Oracle: Review

Classical Oracle

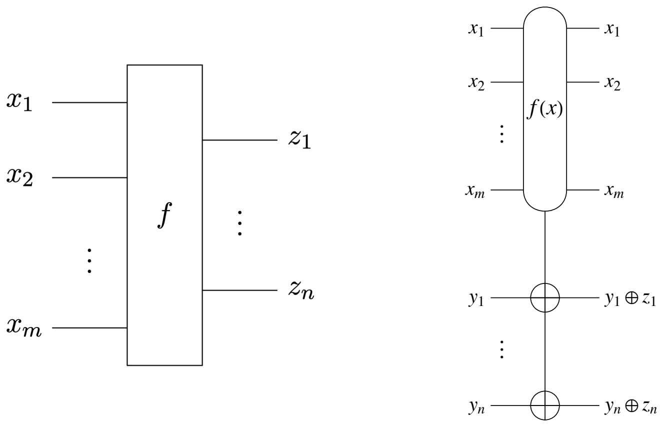


Figure 1. **(Left)** A circuit diagram of classical oracle $f:\{0,1\}^m \to \{0,1\}^n$. x is an m-bit string, $x \in \{0,1\}^m$, and z denotes the image of f at x, $z = f(x) \in \{0,1\}^n$. **(Right)** A reversible version of classical oracle $f:\{0,1\}^m \to \{0,1\}^n$. $x \in \{0,1\}^m$ and $y \in \{0,1\}^n$ are m-bit and n-bit strings, respectively, and $z = f(x) \in \{0,1\}^n$ denotes the image of f at x.

In[ ]:=
```
$m = 3;
$n = 2;
```

In[ ]:=
```
f[1] = f[2] = 3;
f[7] = 2;
f[_Integer] = 0;
```

In[ ]:=
```
ff = Oracle[f, $m, $n]
```

Out[ ]=
```
Oracle[f, 3, 2]
```

In[ ]:=
```
xx = Tuples[{0, 1}, $m]
```

Out[ ]=
```
{{0, 0, 0}, {0, 0, 1}, {0, 1, 0}, {0, 1, 1}, {1, 0, 0}, {1, 0, 1}, {1, 1, 0}, {1, 1, 1}}
```

In[∘]:=
```
zz = ff /@ xx
```
Out[∘]=
```
{{0, 0}, {1, 1}, {1, 1}, {0, 0}, {0, 0}, {0, 0}, {0, 0}, {1, 0}}
```

In[∘]:=
```
Thread[xx → zz] // TableForm
```
Out[∘]//TableForm=
```
{0, 0, 0} → {0, 0}
{0, 0, 1} → {1, 1}
{0, 1, 0} → {1, 1}
{0, 1, 1} → {0, 0}
{1, 0, 0} → {0, 0}
{1, 0, 1} → {0, 0}
{1, 1, 0} → {0, 0}
{1, 1, 1} → {1, 0}
```

Quantum Oracle

Definition

The *quantum oracle* corresponding to the classical oracle f is simply an implementation of the above extended mapping for reversible computation on quantum registers. It is a quantum gate operation defined by the association

$$U_f : \left|x\right\rangle \otimes \left|y\right\rangle \mapsto \left|x\right\rangle \otimes \left|f(x) \oplus y\right\rangle,$$

where $\left|x\right\rangle$ and $\left|y\right\rangle$ are the computational basis states belonging to the native and auxiliary register of m and n qubits, respectively.

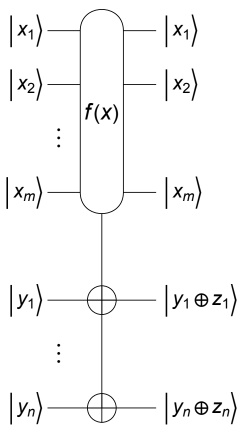


Figure 3. A quantum circuit for the quantum oracle corresponding to classical oracle $f : \{0, 1\}^m \to \{0, 1\}^n$., which is a direct analogue of the classical reversible oracle in Figure 2. $x \in \{0, 1\}^m$ and $y \in \{0, 1\}^n$ is m-bit and n-bit strings, respectively, and $z = f(x) \in \{0, 1\}^n$ denotes the image of f at x.

- Since the extended mapping is one-to-one and the computational basis states are orthonormal, the operator U_f is unitary.
- Recall that U_f is a linear operator and can act on any arbitrary superposition states.

Example

In[ ]:=
```
Let[Qubit, S, T]
```

In[ ]:=
```
SS = S[Range[$m], $];
TT = T[Range[$n], $];
```

In[ ]:=
```
op = Oracle[f, SS, TT]
```

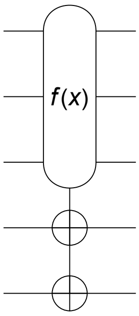
Out[ ]=

Oracle[f, {S_1, S_2, S_3}, {T_1, T_2}]

Note: Oracle[f, {c_1, c_2, …}, {t_1, t_2, …}] represents quantum oracle while Oracle[f, m, n] refers to the classical oracle.

In[ ]:=
```
qc = QuantumCircuit[op]
```

Out[ ]=



In[ ]:=
```
in = Basis[SS, TT];
ProductForm[in, {SS, TT}]
```

Out[ ]=

{|000⟩ ⊗ |00⟩, |000⟩ ⊗ |01⟩, |000⟩ ⊗ |10⟩, |000⟩ ⊗ |11⟩, |001⟩ ⊗ |00⟩, |001⟩ ⊗ |01⟩, |001⟩ ⊗ |10⟩, |001⟩ ⊗ |11⟩, |010⟩ ⊗ |00⟩, |010⟩ ⊗ |01⟩, |010⟩ ⊗ |10⟩, |010⟩ ⊗ |11⟩, |011⟩ ⊗ |00⟩, |011⟩ ⊗ |01⟩, |011⟩ ⊗ |10⟩, |011⟩ ⊗ |11⟩, |100⟩ ⊗ |00⟩, |100⟩ ⊗ |01⟩, |100⟩ ⊗ |10⟩, |100⟩ ⊗ |11⟩, |101⟩ ⊗ |00⟩, |101⟩ ⊗ |01⟩, |101⟩ ⊗ |10⟩, |101⟩ ⊗ |11⟩, |110⟩ ⊗ |00⟩, |110⟩ ⊗ |01⟩, |110⟩ ⊗ |10⟩, |110⟩ ⊗ |11⟩, |111⟩ ⊗ |00⟩, |111⟩ ⊗ |01⟩, |111⟩ ⊗ |10⟩, |111⟩ ⊗ |11⟩}

```
In[ ]:= out = op ** in;
        ProductForm[out, {SS, TT}]
Out[ ]=
{|000⟩ ⊗ |00⟩, |000⟩ ⊗ |01⟩, |000⟩ ⊗ |10⟩, |000⟩ ⊗ |11⟩, |001⟩ ⊗ |11⟩, |001⟩ ⊗ |10⟩,
 |001⟩ ⊗ |01⟩, |001⟩ ⊗ |00⟩, |010⟩ ⊗ |11⟩, |010⟩ ⊗ |10⟩, |010⟩ ⊗ |01⟩, |010⟩ ⊗ |00⟩,
 |011⟩ ⊗ |00⟩, |011⟩ ⊗ |01⟩, |011⟩ ⊗ |10⟩, |011⟩ ⊗ |11⟩, |100⟩ ⊗ |00⟩,
 |100⟩ ⊗ |01⟩, |100⟩ ⊗ |10⟩, |100⟩ ⊗ |11⟩, |101⟩ ⊗ |00⟩, |101⟩ ⊗ |01⟩,
 |101⟩ ⊗ |10⟩, |101⟩ ⊗ |11⟩, |110⟩ ⊗ |00⟩, |110⟩ ⊗ |01⟩, |110⟩ ⊗ |10⟩,
 |110⟩ ⊗ |11⟩, |111⟩ ⊗ |10⟩, |111⟩ ⊗ |11⟩, |111⟩ ⊗ |00⟩, |111⟩ ⊗ |01⟩}
```

In[ ]:=
```
ProductForm[Thread[in → out], {SS, TT}] // TableForm
```

Out[ ]//TableForm=

$|000\rangle \otimes |00\rangle \to |000\rangle \otimes |00\rangle$
$|000\rangle \otimes |01\rangle \to |000\rangle \otimes |01\rangle$
$|000\rangle \otimes |10\rangle \to |000\rangle \otimes |10\rangle$
$|000\rangle \otimes |11\rangle \to |000\rangle \otimes |11\rangle$
$|001\rangle \otimes |00\rangle \to |001\rangle \otimes |11\rangle$
$|001\rangle \otimes |01\rangle \to |001\rangle \otimes |10\rangle$
$|001\rangle \otimes |10\rangle \to |001\rangle \otimes |01\rangle$
$|001\rangle \otimes |11\rangle \to |001\rangle \otimes |00\rangle$
$|010\rangle \otimes |00\rangle \to |010\rangle \otimes |11\rangle$
$|010\rangle \otimes |01\rangle \to |010\rangle \otimes |10\rangle$
$|010\rangle \otimes |10\rangle \to |010\rangle \otimes |01\rangle$
$|010\rangle \otimes |11\rangle \to |010\rangle \otimes |00\rangle$
$|011\rangle \otimes |00\rangle \to |011\rangle \otimes |00\rangle$
$|011\rangle \otimes |01\rangle \to |011\rangle \otimes |01\rangle$
$|011\rangle \otimes |10\rangle \to |011\rangle \otimes |10\rangle$
$|011\rangle \otimes |11\rangle \to |011\rangle \otimes |11\rangle$
$|100\rangle \otimes |00\rangle \to |100\rangle \otimes |00\rangle$
$|100\rangle \otimes |01\rangle \to |100\rangle \otimes |01\rangle$
$|100\rangle \otimes |10\rangle \to |100\rangle \otimes |10\rangle$
$|100\rangle \otimes |11\rangle \to |100\rangle \otimes |11\rangle$
$|101\rangle \otimes |00\rangle \to |101\rangle \otimes |00\rangle$
$|101\rangle \otimes |01\rangle \to |101\rangle \otimes |01\rangle$
$|101\rangle \otimes |10\rangle \to |101\rangle \otimes |10\rangle$
$|101\rangle \otimes |11\rangle \to |101\rangle \otimes |11\rangle$
$|110\rangle \otimes |00\rangle \to |110\rangle \otimes |00\rangle$
$|110\rangle \otimes |01\rangle \to |110\rangle \otimes |01\rangle$
$|110\rangle \otimes |10\rangle \to |110\rangle \otimes |10\rangle$
$|110\rangle \otimes |11\rangle \to |110\rangle \otimes |11\rangle$
$|111\rangle \otimes |00\rangle \to |111\rangle \otimes |10\rangle$
$|111\rangle \otimes |01\rangle \to |111\rangle \otimes |11\rangle$
$|111\rangle \otimes |10\rangle \to |111\rangle \otimes |00\rangle$
$|111\rangle \otimes |11\rangle \to |111\rangle \otimes |01\rangle$

In[ ]:=
```
Thread[xx → zz] // TableForm
```

Out[ ]//TableForm=
```
{0, 0, 0} → {0, 0}
{0, 0, 1} → {1, 1}
{0, 1, 0} → {1, 1}
{0, 1, 1} → {0, 0}
{1, 0, 0} → {0, 0}
{1, 0, 1} → {0, 0}
{1, 1, 0} → {0, 0}
{1, 1, 1} → {1, 0}
```

Summary

Keywords

- Oracle
- Quantum oracle
- Quantum decision making

Functions

- `Oracle`
- `ControlledExp`

Related Links

- Section 4.2 of the Quantum Workbook (2022, 2023).
- Tutorial: Quantum Oracle
- Tutorial: Quantum Decision Algorithms