ДЗ №3: Машины Тьюринга и квантовые вычисления

Новичихин И.В.

31 мая 2022 г.

1 Машины Тьюринга

Работу требуется выполнять в системе turingmachine.io.

Для сдачи заданий 1-2 требуется прикрепить файлы YAML с исходным кодом проекта. Каждый файлы должен иметь наименование задание_пункт.yml, к примеру 1_1.yml для первой задачи первого задания.

1.1 Операции с числами

Реализуйте машины Тьюринга, которые позволяют выполнять следующие операции:

1.1.1 Сложение двух унарных чисел (1 балл)

Решение представлено в файле 1_1.yaml.

Алгортим: заменяем "+"на 1, и удаляем последнюю единицу

1.1.2 Умножение унарных чисел (1 балл)

Решение представлено в файле 1_2.yaml.

Алгоритм: пишем справа знак равно и переносим единицы по одной. Уже перенесенную единицу отмечаем симолом x, после того как всё перенесли стираем то что слева от равно и само равно

1.2 Операции с языками и символами

Реализуйте машины Тьюринга, которые позволяют выполнять следующие операции:

1.2.1 Принадлежность к языку $L = \{0^n1^n2^n\}, n \geq 0$ (0.5 балла)

Решение представлено в файле 2_1.yaml.

На вход поступает строка из 0, 1, 2. Далее по очереди заменяем 0, 1, 2 на x. Если в результате всё заменилось на x, то результат работы y (слово принадлежит языку L). Если нарушится порядок символов или их будет не одинаковое количество, то результат работы n (слово не принадлежит языку L)

1.2.2 Проверка соблюдения правильности скобок в строке (минимум 3 вида скобок) (0.5 балла)

Решение представлено в файле 2_2.yaml.

На вход поступает строка их скобок (), [], {}, Будем идти по слову и искать первую закрывающую скобку. Как только она найдена, заменяем скобку на x и идем направо искать открывающую такого же вида (идем направо пока есть символ x), иначе слово не принадлежит языку. Если в итоге в строке остались только символы x, то стираем все символы записываем y, это будет результат программы (слово принадлежит языку). Если не принадлежит языку результат программы символ n

1.2.3 Поиск минимального по длине слова в строке (слова состоят из символов 1 и 0 и разделены пробелом) (1 балл)

Решение представлено в файле 2_3.yaml.

На вход поступает строка из слов из 0 и 1, которые разделены пробелом. Обрабатываем по два слова: 0 заменяем на x, 1 заменяем на w, и затем сравниваем слова по длине. Если первое слово больше, то стираем его, а во втором слове обратно заменяем x на 0, w на 1. Если второе слово больше, то перемещаем первое слово на место второго и удаляем лишние символы.

2 Квантовые вычисления

Для выполнения заданий по квантовым вычислениям требуется QDK. Его можно скачать здесь: https://docs.microsoft.com/en-us/azure/quantum/install-overview-qdk.

Но можно использовать любой пакет, типа https://qiskit.org/.

В качестве решения задачи надо предоставить схему алгоритма для частного случая при фиксированном количестве кубитов и фиксированных состояниях.

2.1 Генерация суперпозиций 1 (1 балл)

Дано N кубитов ($1 \leq N \leq 8$) в нулевом состоянии $|0\dots0\rangle$. Также дана некоторая последовательность битов, которое задаёт ненулевое базисное состояние размера N. Задача получить суперпозицию нулевого состояния и заданного.

$$|S\rangle = \frac{1}{\sqrt{2}}(|0\dots0\rangle + |\psi\rangle)$$

То есть требуется реализовать операцию, которая принимает на вход:

1. Массив кубитов q_s

2. Массив битов $bits$ описывающих некоторое состояние $|\psi\rangle$. Это массив имеет тот же самый размер, что и qs. Первый элемент этого массива равен 1.

Так как первые кубиты векторов различны, то применяем оператор Адамара к первому кубиту

Далее если bits[i] равно 1, то запутываем i-ый кубит (оператор CNOT)

```
namespace Solution {
    open Microsoft.Quantum.Primitive;
    open Microsoft.Quantum.Canon;
    operation Solve (qs : Qubit[], bits : Bool[]) : ()
    {
        body
        {
            H(qs[0]);
            for (i in 1..Length(qs) - 1)
                if (bits[i])
                    CNOT(qs[0], qs[i]);
        }
    }
}
```

2.2 Различение состояний 1 (1 балл)

Дано N кубитов ($1 \leq N \leq 8$), которые могут быть в одном из двух состояний:

$$|GHZ\rangle = \frac{1}{\sqrt{2}}(|0\ldots0\rangle + |1\ldots1\rangle)$$

$$|W\rangle = \frac{1}{\sqrt{N}}(|10\ldots00\rangle + |01\ldots00\rangle + \cdots + |00\ldots01\rangle)$$

Требуется выполнить необходимые преобразования, чтобы точно различить эти два состояния. Возвращать 0, если первое состояние и 1, если второе.

```
namespace Solution {
    open Microsoft.Quantum.Primitive;
    open Microsoft.Quantum.Canon;
    operation Solve (qs : Qubit[]) : Int
    {
        body
        {
            mutable countOnes = 0;
            for (i in 0..Length(qs) - 1) {
                if (M(qs[i]) == One) {
                    set countOnes = countOnes + 1;
                }
            }
```

```
            if (countOnes == Length(qs) or countOnes == 0)
            {
                return 0;
            }
            return 1;
        }
    }
}
```

2.3 Различение состояний 2 (2 балла)

Дано 2 кубита, которые могут быть в одном из двух состояний:

$$|S_0\rangle = \frac{1}{2}(|00\rangle + |01\rangle + |10\rangle + |11\rangle)$$
$$|S_1\rangle = \frac{1}{2}(|00\rangle - |01\rangle + |10\rangle - |11\rangle)$$
$$|S_2\rangle = \frac{1}{2}(|00\rangle + |01\rangle - |10\rangle - |11\rangle)$$
$$|S_3\rangle = \frac{1}{2}(|00\rangle - |01\rangle - |10\rangle + |11\rangle)$$

Требуется выполнить необходимые преобразования, чтобы точно различить эти четыре состояния. Возвращать требуется индекс состояния (от 0 до 3).

Заготовка для кода:

```
namespace Solution {
        open Microsoft.Quantum.Primitive;
        open Microsoft.Quantum.Canon;
        operation Solve (qs : Qubit[]) : Int
        {
            body
            {

                return
            }
        }
}
```

2.4 Написание оракула 1 (2 балла)

Требуется реализовать квантовый оракул на N кубитах ($1 \leq N \leq 8$), который реализует следующую функцию: $f(\boldsymbol{x}) = (\boldsymbol{bx}) \mod 2$, где $\boldsymbol{b} \in \{0,1\}^N$ вектор битов и $\boldsymbol{x}$ вектор кубитов. Выход функции записать в кубит $\boldsymbol{y}$. Количество кубитов N ($1 \leq N \leq 8$).

Заготовка для кода:

```
namespace Solution {
        open Microsoft.Quantum.Primitive;
        open Microsoft.Quantum.Canon;
        operation Solve (x : Qubit[], y : Qubit, b : Int[]) : ()
        {
            body
            {

            }
        }
}
```