東京大学
量子ソフトウェア寄付講座

第7回 量子ソフトウェアハンズオン

2025年 1月 29日

本日の演習の概要

- Python上で使えるライブラリ「TensorNetwork」を用いて量子コンピュータのテンソルネットワークシミュレーションを実際に動かしてもらいます。
- 演習を通じて、以下の目標を達成することを目指します。

本日の目標

1. Python上でのテンソルネットワークによる量子回路シミュレーションの実装方法を学ぶ
2. 実際にいくつかのアルゴリズムを、テンソルネットワークを用いてシミュレーションすることで、テンソルネットワークシミュレーションの強み/弱みについて考察する
3. 量子コンピュータが役立ちそうな場所はどのような場所か（古典コンピュータでは解けない問題は何か）を考え、ディスカッションする

演習の内容

- 前述の目標を達成するため、本日は4つの演習に取り組んでいただきます。
- 分からない点などがあれば、積極的に周囲の人と相談しながら取り組んでください。
  - 講師側でも適宜サポートに回ります。

#	演習テーマ	概要
1	TensorNetworkの使い方 - 特定の状態の振幅の計算	• 最初に、簡単な回路を題材に、ライブラリの使い方の解説を行いながら、特定の状態の振幅を計算する手法について、学びます。 • その後、実際に自分の手でテンソルネットワークを組み、量子回路のシミュレーションを行う演習課題に取り組んでもらいます。
2	特定の量子ビットに注目した測定	• 量子コンパレータというアルゴリズムを題材に、特定の量子ビットに着目して、確率を計算する方法について学びます。
3	量子回路からのサンプリング	• 簡単な回路を題材に、テンソルネットワークを使い、量子回路からサンプリングを行う方法を学びます。
4	量子アルゴリズムのシミュレーション - Groverのアルゴリズム	• 今まで学んだ結果を活用して、Groverのアルゴリズムに対して、特定の状態の振幅の計算や、サンプリングを行ってみます。

演習環境の準備

- 今回の演習はGoogle Colaboratoryを使って、演習・解説を行います。
- 各自のPCのローカル環境上で行っていただいても構いませんが、ご自身の環境に依存するエラー対応については、自己責任でお願いいたします。

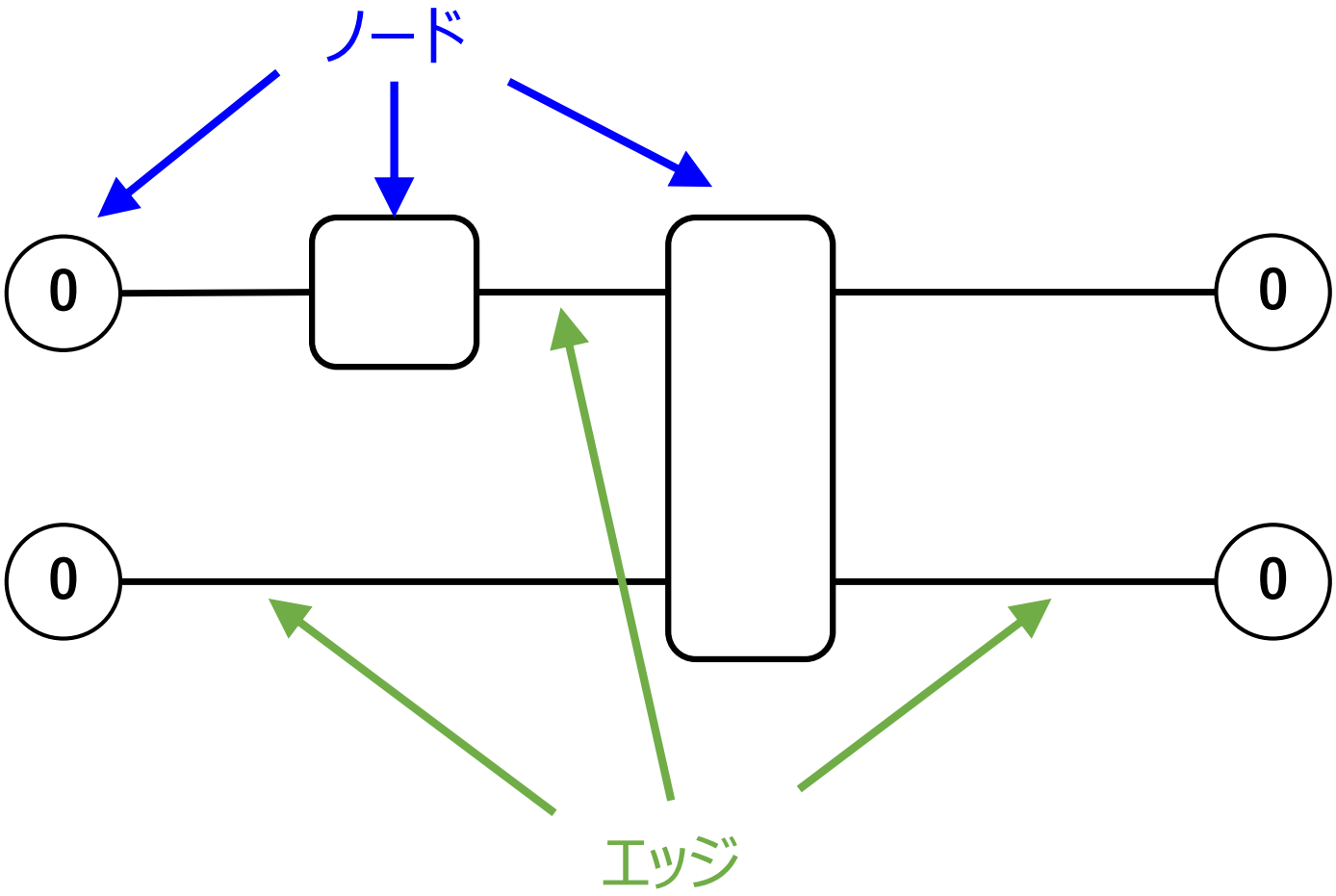
今回の講義のgithubページに以下の4つのノートブックを掲載してあります。

- 「01_tensornetworkの使い方.ipynb」
- 「02_comparatorのシミュレーション.ipynb」
- 「03_量子回路からのサンプリング.ipynb」
- 「04_groverのシミュレーション.ipynb」

ご自身のドライブにコピーしていただき、Google Colab環境でノートブックを開いてください。

演習1．TensorNetworkライブラリについて

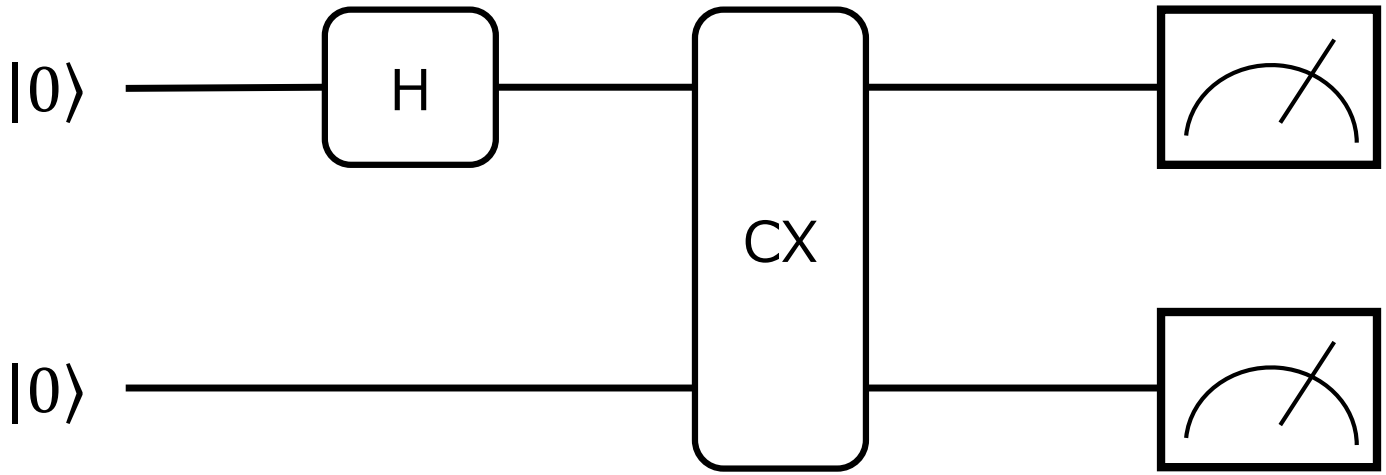
- 「TensorNetwork」はGoogleの作成したテンソルネットワークの計算を行うライブラリです。
- ノードとエッジを定義することで、ネットワークの形状を指定することができます。
- その後、用意されている関数を利用するだけで、自動で縮約計算を行ってくれます。



演習1．簡単な回路の実装

- ベル状態を作成する回路（下図参照）を題材に、実際に実装例を確認します。
- この回路を実行すると、|00⟩と|11⟩が等確率で出現します。
- まずは特定の状態の振幅を計算する手法を用いて|00⟩が出現する確率が50%であることを確認します。

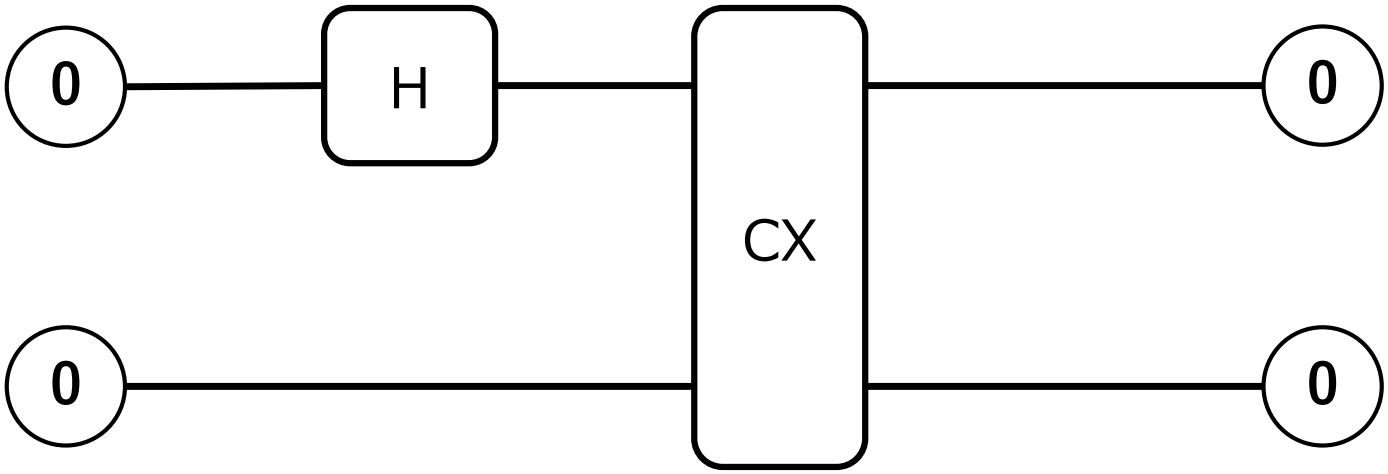
＜ベル状態を作成する回路＞



演習1. 簡単な回路の実装

- テンソルネットワークを用いて、|00⟩の振幅を計算する際には、以下のようなテンソルネットワークを構築します。
- このテンソルネットワークの縮約を計算することで、振幅が得られます。
- 実際にJupyter Notebook上でこのテンソルネットワークを作成し、動作を確認してみましょう。

＜ベル状態を作成する回路に対応するテンソルネットワーク＞



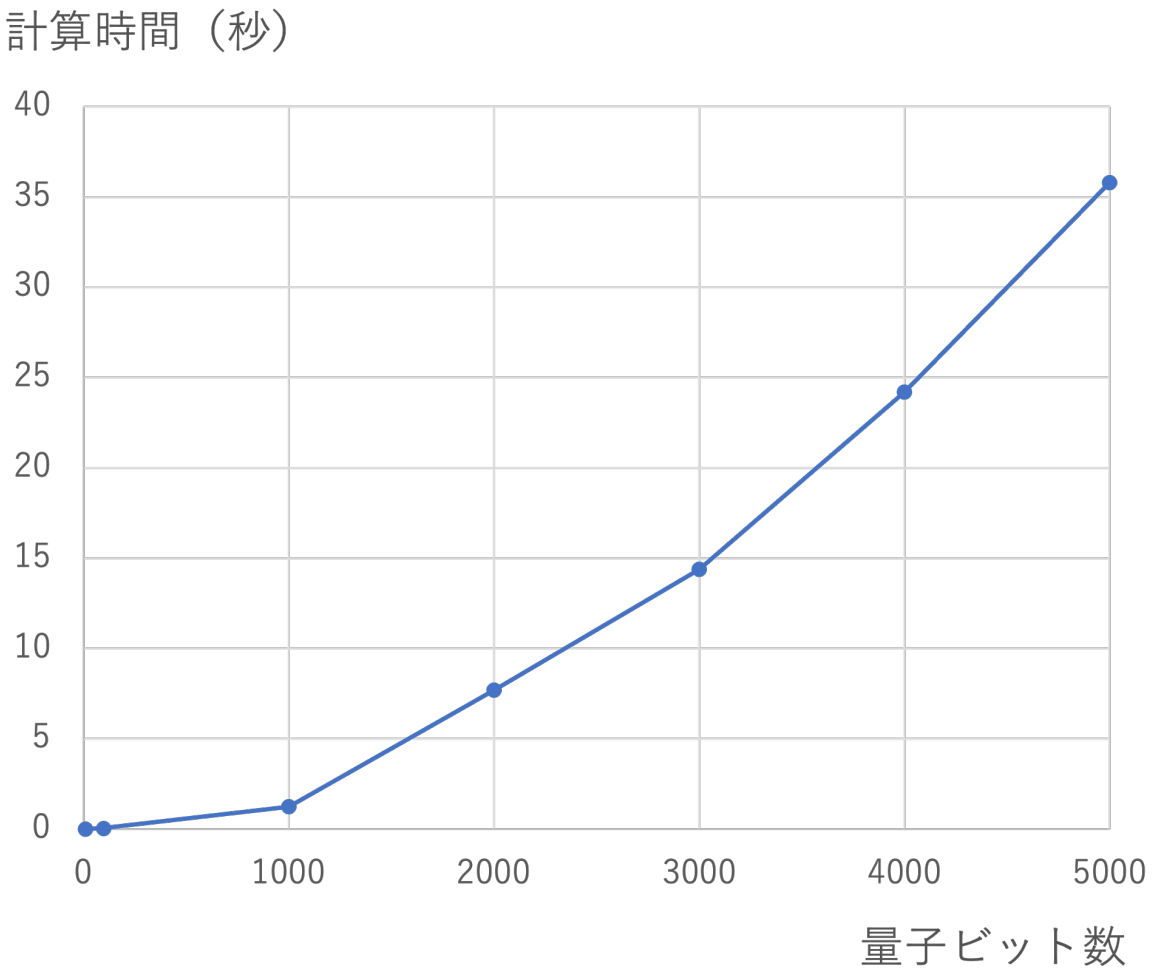
演習1. TensorNetworkの実装演習

- 前述の実装例を参考に、以下の3つの課題に順次取り組みます。
- 実装の方針に悩むことがあれば、遠慮なく質問してください。

#	課題	概要
1	計算する状態の変更	• $\|00\rangle$の出現確率を求める回路を元に、$\|11\rangle$や$\|01\rangle$の出現確率を求めてみましょう。
2	GHZ状態のシミュレーション	• ベル状態を作成する回路のテンソルネットワークを参考に、GHZ状態を作成する回路のテンソルネットワークシミュレーションをやってみましょう。 • GHZ状態：3量子ビットのもつれ状態
3	より多くの量子ビットを用いたもつれ状態作成のシミュレーション	• より一般に、n量子ビットに対して、もつれ状態を作成できるような回路を作成し、テンソルネットワークシミュレーションをやってみましょう。 • 量子ビット数が非常に多くなっても計算できることを確認しましょう。

演習1. TensorNetworkの実装演習

- 量子ビット数を変更しながら、計算時間を計測した結果を以下に記載。
- 通常の状態ベクトルシミュレーションでは動かすことのできないビット数でも高速に計算ができます。



演習2. 特定の量子ビットに注目した測定

- QPEなど、計算に使う量子ビットは多くとも、測定が必要な量子ビットはその一部であるアルゴリズムは多いです。
- 今回は、Comparatorというアルゴリズムを題材に、特定の量子ビットのみに着目した測定を実装してみます。
  このアルゴリズムは2量子ビットのみの測定で十分です。

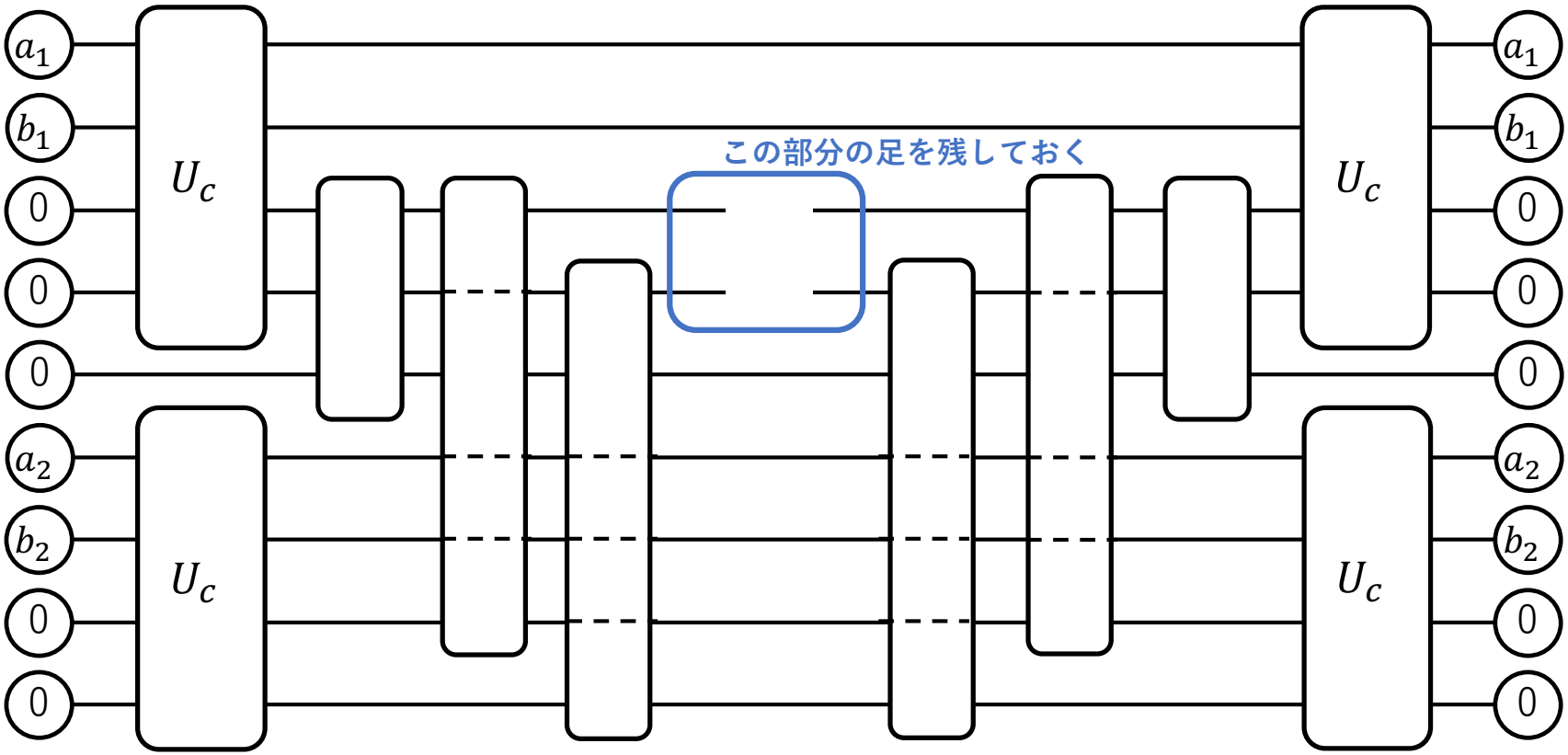
Quantum Bit String Comparator [1]

- 与えられた二つのビット列A,Bの大小関係を判定するアルゴリズム。
- 重ね合わせ状態にも対応しているため、確率的に大小関係がどうなるかが判別できる。
- 結果は以下の3パターン。
  00:等しい　10:Aのほうが大きい　01:Bのほうが大きい　（11は出現しない。）

[1] Oliveira, David & Ramos, Rubens. (2007). Quantum bit string comparator: Circuits and applications. Quantum Computers and Computing. 7.

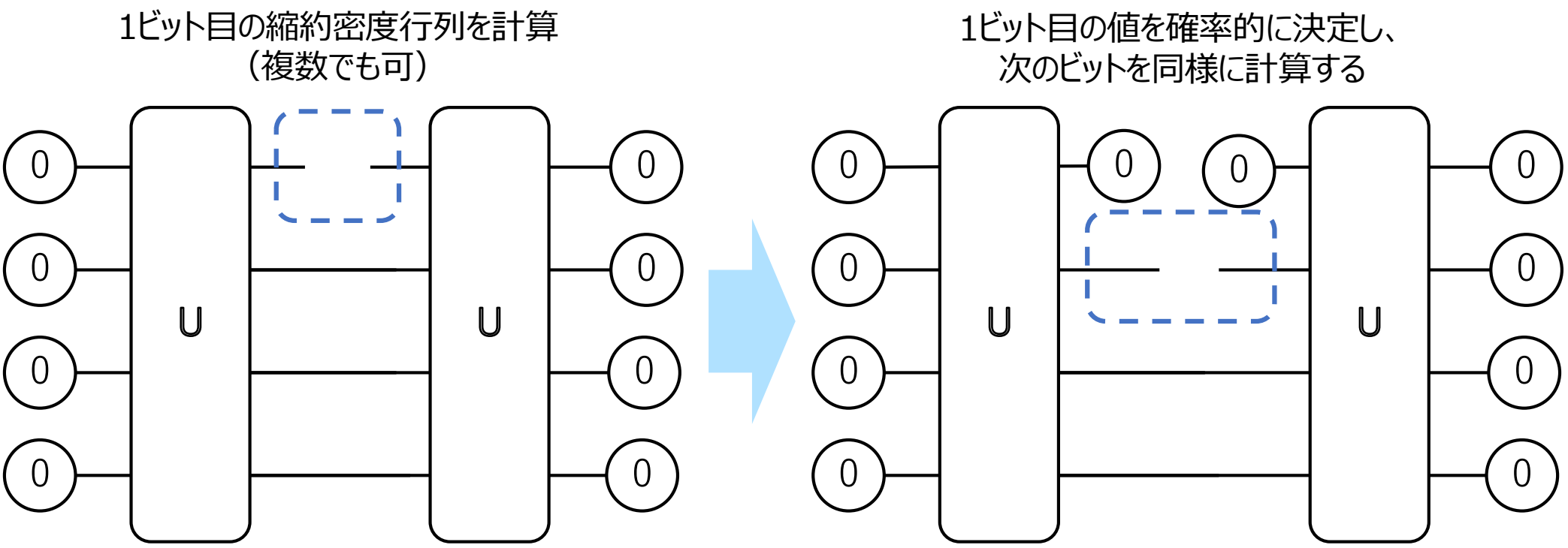
演習2. 特定の量子ビットに注目した測定

- Comparatorのテンソルネットワーク実装の概略図は以下のような形になります。
- 量子回路を反転させて、くっつけたような形状をしています。
- 測定を行いたい量子ビットの部分のみ足を残しておき、縮約を計算します。



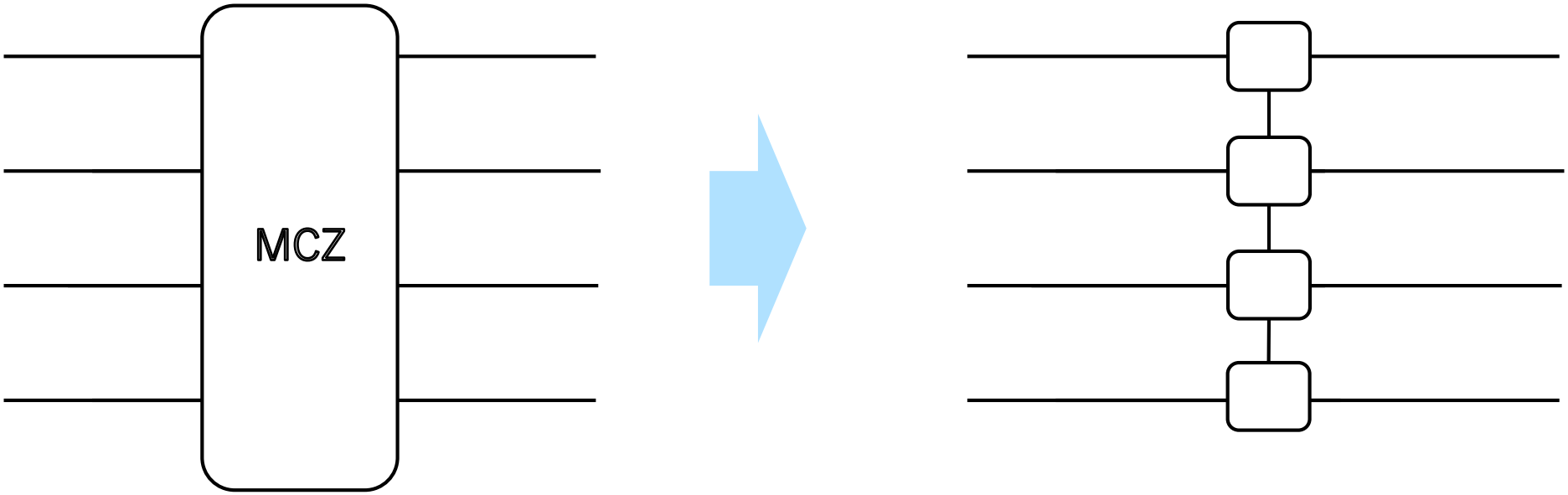
演習3. 量子回路からのサンプリング

- 次に、GHZ状態の回路に戻り、この回路からサンプリングを行ってみます。
- サンプリングを行う際には、演習2で学んだ考え方を利用し、一つずつ順番に量子ビットの確率分布を確認していき、順番に値を確定させていきます。
  - 一つずつである必要はなく、複数単位でもよいですが、今回の演習では一つずつにします。



演習4. Groverのシミュレーション

- 今まで学んだ要素を使って、より複雑なアルゴリズムであるGroverのアルゴリズムのシミュレーションを行ってみましょう。
- 今回は、すべてのビットが1になっている状態$|11\ldots111\rangle$を探索することにします。
- マルチコントロールゲートは、そのまま実装すると巨大な行列になってしまうため、事前に複数のテンソルの積に分解しておきます。



ディスカッションについて

- 今回の演習の内容を振り返り、感想、学び、疑問点などをグループ内で共有してください。
- 量子コンピュータやテンソルネットワークに対する期待、疑問点などもディスカッションしてみてください。
- 最後にグループの代表者から全体に共有していただきます。