

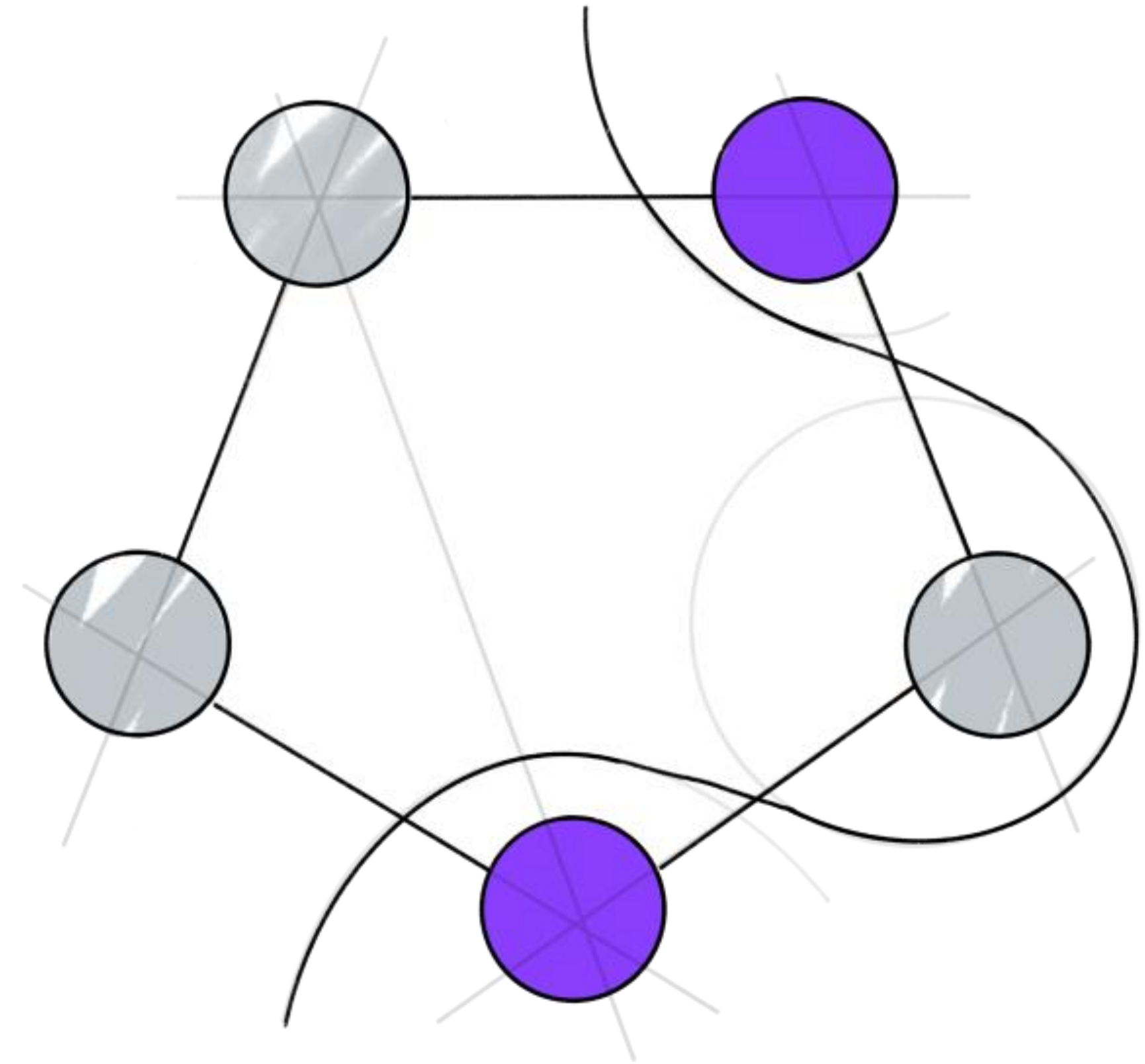
最適化問題

Sep 30, 2025

小林 有里

Yuri Kobayashi

IBM Quantum



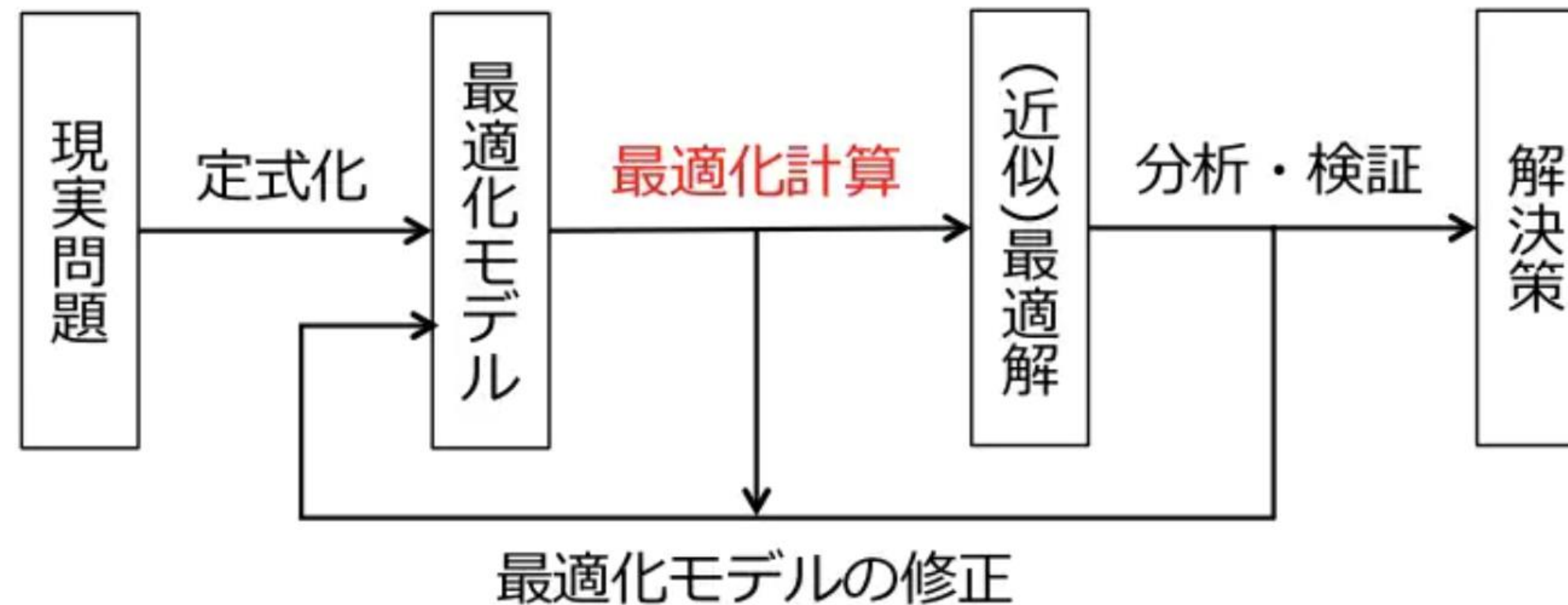
IBM Quantum

今回のトピック

- 最適化入門
- 期待値とは？
- 変分原理とは？
- 変分量子アルゴリズム(VQA)について
- 量子近似最適化アルゴリズム(QAOA)について
- 最適化問題とは？
- QAOAでMax-cut問題を解く

最適化問題の流れ

- 最適化は意思決定・問題解決のための一つの手段
- 最適化問題の流れ：定式化 + 最適化計算 + (近似) 最適解の分析・検証 + 最適化モデルの修正



- 最適化問題
制約条件を満たす解の中で目的関数を最小（最大）にする解を求める問題です。

定式化の際は一般にこう記述をします

minimize $f(x)$
subject to $x \in F$

こちらの記述もよく使われます

$\min (\max)$ $f(x)$
s.t. $x \in F$

定式化とは？

解きたい問題を数式であらわすこと

目的関数・・・最小化または最大化したいもの

- 最小化：コスト、距離、重量、処理時間
 - 例） AからBへ最短時間で行く経路を求める
- 最大化：利益、価値、生産高、効率、満足度
 - 例） 300円の予算で最も満足度の高い遠足に
持っていくお菓子を決める



定式化のお作法

目的関数：最小化または最大化したいもの

（最小化：園内の移動距離、最大化：満足度、報酬）

（最小化） **Minimize** ～ （最大化） **Maximize** ～

決定変数：(binary, integer)

制約条件：探すべき変数がある集合に含まれることを指す

（閉園時間まで $t=4$ 時間、アトラクション数、2回連続で
同じアトラクションには乗車しないルール, etc.)

Subject to ～ 具体的な制約条件の記述

定式化の具体例

ワインの生産計画問題

あるシャトーでは3種類のブドウ（カベルネ、メルロ、セミヨン）を原料として赤ワイン、白ワイン、ロゼワインを製造しています。収益が最大となる各ワインの一日当たりの製造量を求めてください。

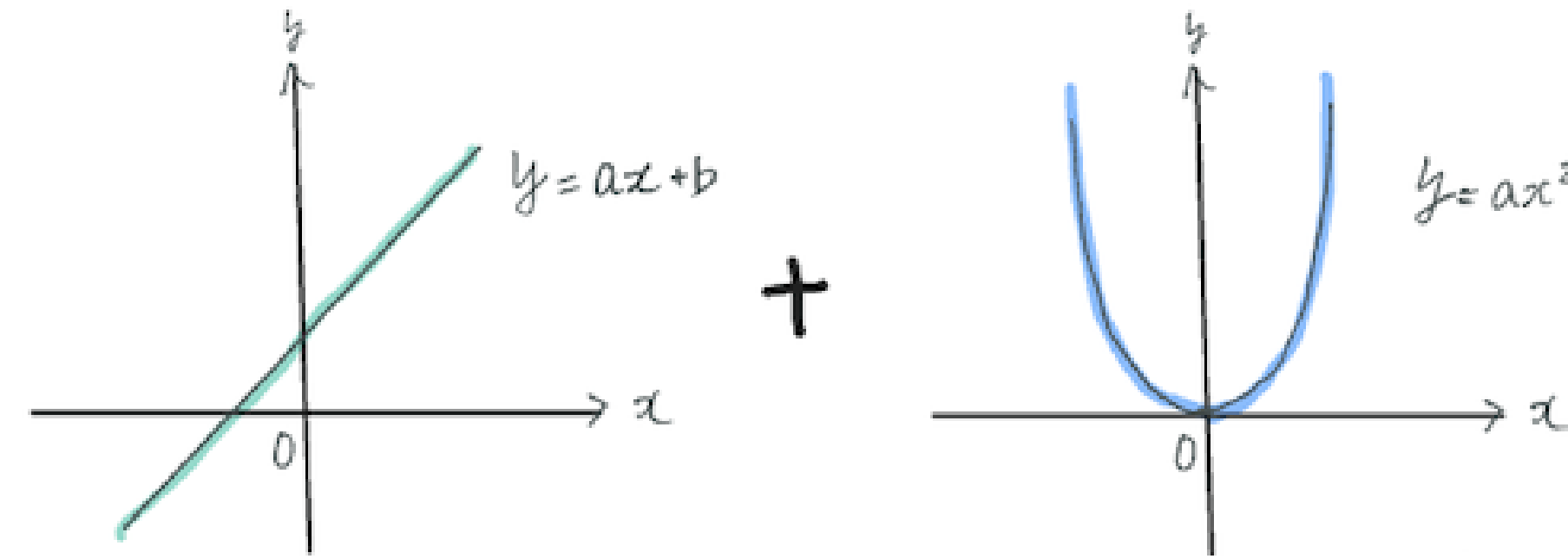
| | x_1 | x_2 | x_3 | |
|------|-------|-------|-------|-----------|
| 品種 | 赤ワイン | 白ワイン | ロゼワイン | 最大供給量 |
| カベルネ | 2 | 0 | 0 | 4(t/day) |
| メルロ | 1 | 0 | 2 | 8(t/day) |
| セミヨン | 0 | 3 | 1 | 6(t/day) |
| 収益 | 3 | 4 | 2 | (1万€/day) |



- Maximize $3x_1 + 4x_2 + 2x_3 \Rightarrow$ 収益を最大化する目的関数
- Subject to $2x_1 \leq 4 \Rightarrow$ カベルネの供給量は4t/day 以内
- $x_1 + 2x_3 \leq 8, \Rightarrow$ メルロの供給量は8t/day 以内
- $3x_2 + x_3 \leq 6, \Rightarrow$ セミヨンの供給量は6t/day 以内
- $x_1, x_2, x_3 \geq 0. \Rightarrow$ 各ワインの製造量は非負

目的関数

目的関数： **1 次式** + **2 次式**
(線形式)



例)

変数が1つの場合： **2x** + **4** + **x²**

変数が2つの場合： **2x₁** + **x₂** + **4 x₁²** + **x₁x₂**

変数が3つの場合： **2x₁** + **1x₂** + **4x₃** + **2x₁x₂** + **4x₁x₃** + **1x₂x₃**

変数の前についている**係数**は、各変数の**特徴量**として、過去データから予め提供されることが多い

ある制約条件のもとで 2 次式の最小値または最大値を求める問題を

「**2 次計画問題** (quadratic program)」と呼びます。

代表的な最適化問題

線形計画問題(Linear Programming; LP)

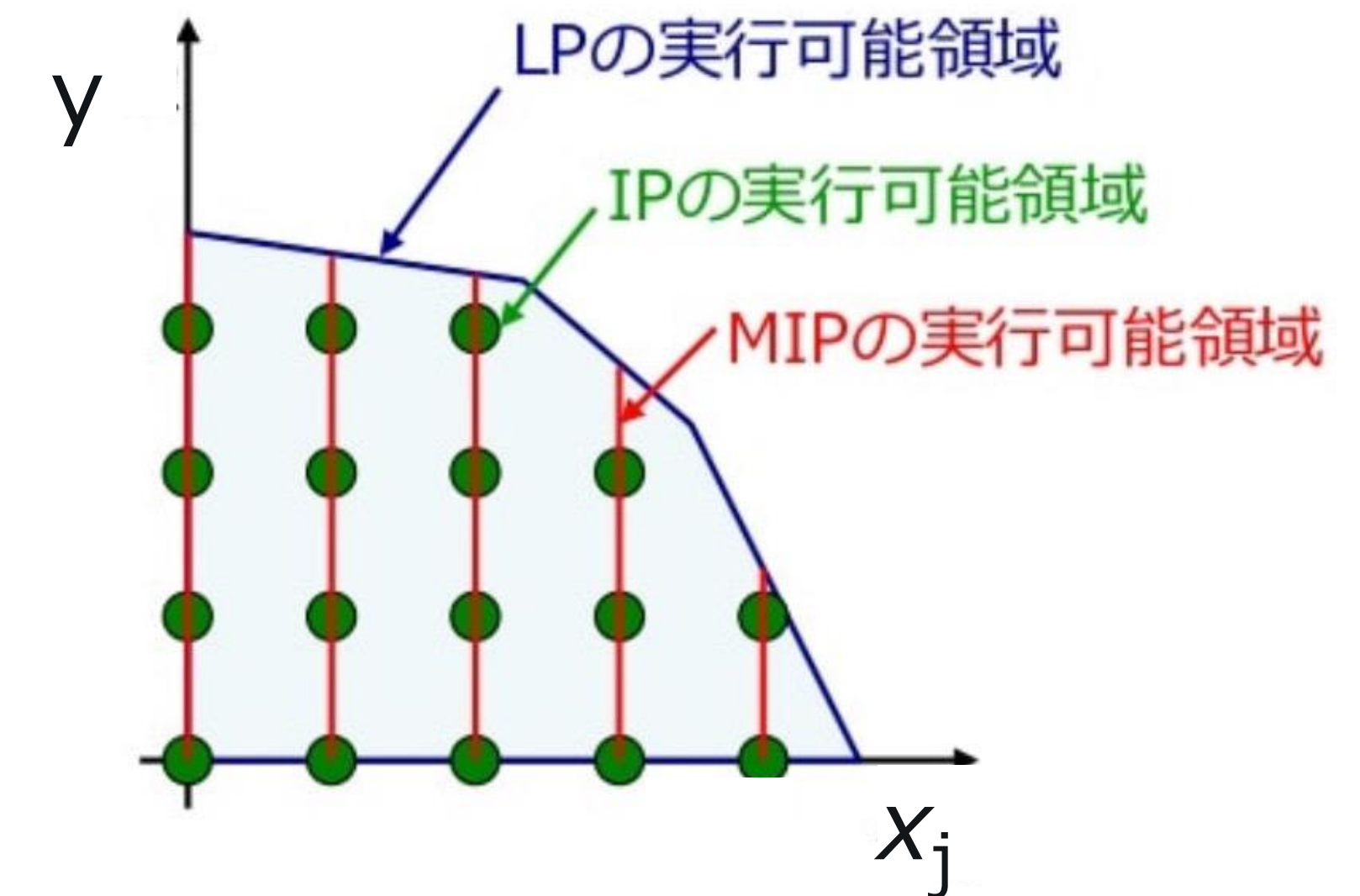
線形制約の下で線形関数を最小化（最大化）

整数計画問題(Integer Programming; IP)

- ・広義：変数に整数条件の付いた最適化問題
- ・狭義：線形計画問題 + 変数の整数条件 (**ILP**)

混合整数計画問題 (Mixed Integer Programming; MIP)

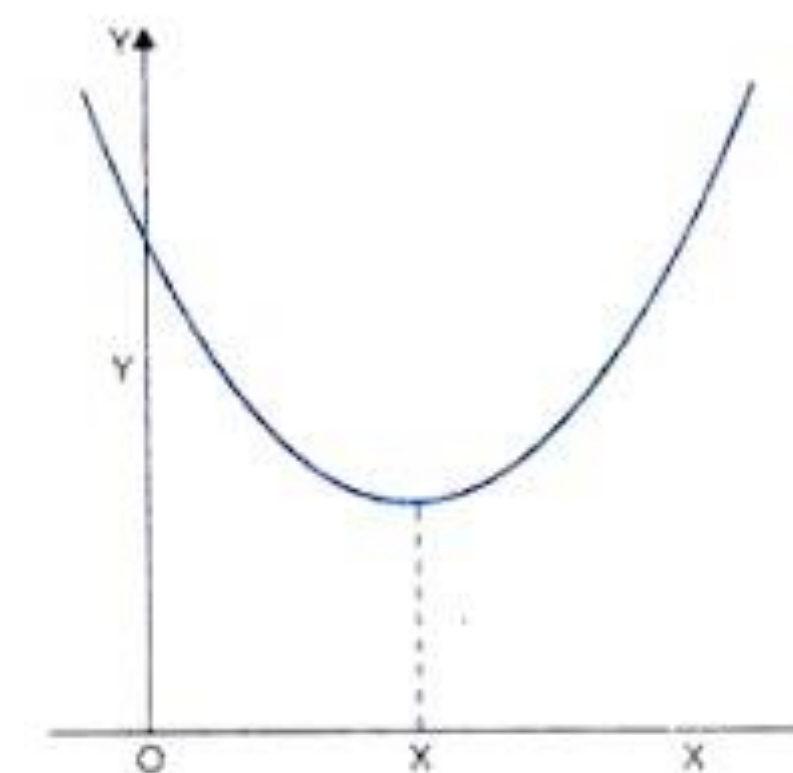
連続変数と離散変数が混在する問題



二次計画問題

非線形計画法の**代表例の一つ**であり、いくつかの変数からなる二次関数を線形制約の下で最適化(min/max) 巡回セールスマン、Max-cut、頂点被覆、ナップザック問題などさまざまな最適化問題の定式化で用いられています。

$$\begin{aligned}
 &\text{Minimize} && x^T A x + B x \\
 &\text{Subject to} && C_i^T x \leq d_i \\
 & && x = (x^1, \dots, x_n) \\
 & && l_i \leq x_i \leq u_i \\
 & && x_i: \text{binary / integer}
 \end{aligned}$$



組合せ最適化問題

組合せ最適化問題

解が順序や割当のように'組合せ的な構造'をもつ最適化問題のことを言います。

組合せ最適化問題の例

- ・配送計画（宅配便、コンビニへの商品配達、ゴミ収集など）
- ・スケジューリング問題（工場での人員配置計画、乗務員・看護師などの勤務表、スポーツの対戦表・日程表など）
- ・ネットワーク設計問題（無線通信網、交通網、石油・ガスのパイプライン網の設計など）
- ・施設配置問題（店舗出店計画、公共施設の計画など）
- ・最短経路問題（カーナビのルート検索、乗り換え案内など）

巡回セールスマン問題（Traveling Salesman Problem）

都市の集合と各2都市間の移動コスト（例えば距離）が与えられた時全ての都市を一度ずつ訪問し、出発地に戻る巡回路の中で総移動コストが最小のものを求める問題。 **NP困難**な問題として知られています。



Source: <https://physics.aps.org/articles/v10/s32>

巡回セールスマン問題(TSP)はなぜNP困難？

n 都市に対して何通りの巡回路を探索する必要があるでしょうか？

$(n - 1)!/2$ 通りの経路を調べればよい。 $n \rightarrow \infty$ おおよそ $O(n!)$

| Cities | Total routes | Computation Time (sec.) |
|--------|-----------------------|-------------------------|
| 6 | 60 | 4.32×10^{-10} |
| 8 | 2520 | 3.23×10^{-8} |
| 10 | 1.81×10^5 | 3.63×10^{-6} |
| 15 | 4.36×10^{10} | 1.96 |
| 20 | 6.08×10^{16} | 4.87×10^6 |
| 25 | 3.10×10^{23} | 3.88×10^{13} |
| 30 | 4.42×10^{30} | 7.96×10^{20} |

巡回セールスマン問題は長年研究されており、TSPに特化した高性能なアルゴリズムが開発されています。この専用のTSPソルバー(Concorde と呼ばれる) を使えば現在のノートブックPCで数百都市のTSPが1秒以内に計算可能です。

<https://www.math.uwaterloo.ca/tsp/concorde/index.html>

~56 days

~1.22 million years (122万年)

~2.5 trillion years (2.5兆年!)

100T Flops(1秒間に 10^{13} 回四則演算できる) のコンピュータの場合

| | | | | | | | | | | |
|------|------|-------|-------|-------|--------|--------|---------|---------|---------|---------|
| 1954 | 1962 | 1977 | 1987 | 1987 | 1987 | 1994 | 1998 | 2001 | 2004 | 2006 |
| n=49 | n=33 | n=120 | n=532 | n=666 | n=2392 | n=7397 | n=13509 | n=15112 | n=24978 | n=85900 |

最適化問題を解くアプローチ

厳密解法 (Exact Algorithm)

最適な解であることが保証された厳密解を求めるアルゴリズム。比較的小さな問題サイズに使えます。

例. 全数探索 (Exhaustive search), 動的計画法 (Dynamic Programming), 分枝限定法 (Branch and bound)

近似アルゴリズム (Approximation Algorithm)

最適性の保証はないが、多項式実行時間内である一定の品質が保証 (生成された解の重みが最適な解の重みに対しある一定の距離以内であることが保証) された解を見つけることができるアルゴリズム

発見的手法・ヒューリスティック (Heuristic)

最適性の保証はないが、特定の計算問題においてある程度のレベルで正解に近い解を得ることができる手法。答えの精度が保証されない代わりに、解答に至るまでの実行時間が短いという特徴がある。

例. 貪欲法 (Greedy search), 局所探索法 (Local search)



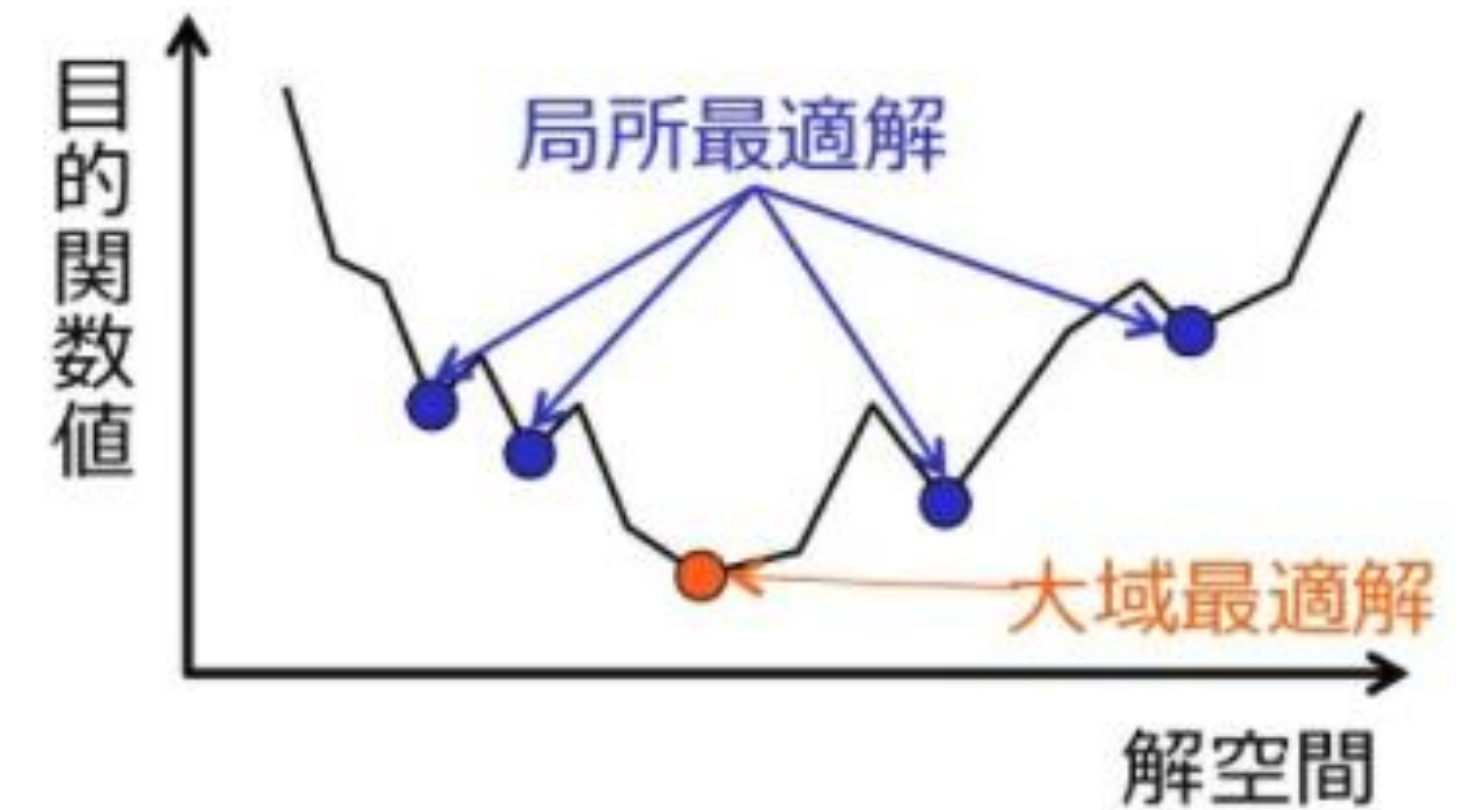
計算困難な問題に対するアプローチ

- ・以下の3つの要素で構成される最適化問題のうち

- ・決定変数(離散or整数)

- ・目的関数(最小化or最大化する関数)

- ・制約(なしor不等式or等式)



- ・目的関数の複雑さ、制約条件の数によって解の候補が膨大にあり(組み合わせ爆発)、厳密解・近似解を求めることが難しいケースに適用

- ・前述の**ヒューリスティックな(発見的な)アルゴリズム**を使用して筋の良い解が求められている

→今回は、QAOAで最適化問題を解いてみます

期待値とは？

確率統計における確率 P

$$1 = \sum_i P_i$$

確率統計における期待値 E (例：さいころ)

$$E(x_i) = \sum_i x_i P(X = x_i)$$

規格化された状態ベクトル $|\psi\rangle$

$$1 = \langle\psi|\psi\rangle$$

量子力学における物理量 A (行列も取りうる)の期待値 $\langle A \rangle$

$$\langle A \rangle = \langle\psi|A|\psi\rangle \neq A\langle\psi|\psi\rangle$$

物理量と期待値の例

量子力学における物理量 A の期待値 $\langle A \rangle$

$$\langle A \rangle = \langle \psi | A | \psi \rangle.$$

例：ハミルトニアン H （量子コンピューターをつかって期待値を求めたい物理量代表）

物理量：ハミルトニアン

$$H\varphi(x) = E\varphi(x)$$

物理系の全エネルギーを表す演算子（エネルギー関数）

$$-\frac{\hbar^2}{2m} \frac{d^2\varphi(x)}{dx^2} + V(x)\varphi(x) = E\varphi(x)$$

運動エネルギー

位置エネルギー

期待値：粒子のエネルギー

期待値の例

例1：ハミルトニアン H が $H|\psi_0\rangle = \varepsilon_0|\psi_0\rangle$ を満たすとき、 ε_0 はスカラーなので

$$\langle H \rangle = \langle \psi_0 | H | \psi_0 \rangle = \langle \psi_0 | \varepsilon_0 | \psi_0 \rangle = \varepsilon_0 \langle \psi_0 | \psi_0 \rangle = \varepsilon_0.$$

※行列形式で考えると、固有値を求める操作に対応。

ハミルトニアン H の期待値（エネルギー）を求めるには、
ハミルトニアン H の最小固有状態で、最小固有値 ε_0 （基底エネルギー）もとめればよい

例2： $|\psi\rangle = \begin{bmatrix} \alpha \\ \beta \end{bmatrix}$ における、演算子 $Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$ の期待値は、 α, β が複素数なので

$$\langle Z \rangle = \langle \psi | Z | \psi \rangle = [\alpha^* \quad \beta^*] \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = |\alpha|^2 - |\beta|^2.$$

※ $|0\rangle$ だと1になり、 $|1\rangle$ だと-1になる。

どうやって最小固有値を求めるのか-変分原理について

ハミルトニアン H の基底状態における最小固有値が ε_0 、対応する状態ベクトルが $|\psi_0\rangle$

$$E |\psi_0\rangle = \langle \psi_0 | H | \psi_0 \rangle = \varepsilon_0,$$

とエネルギー期待値 $E |\psi_0\rangle$ と書けるととき、試行ベクトル $|\psi\rangle$ における $E |\psi\rangle$ は

$$E |\psi\rangle = \langle \psi | H | \psi \rangle = \varepsilon \geq \varepsilon_0$$

となる。これを変分原理と呼ぶ(証明略)。

- ・ハミルトニアン H に対する ε_0 を求めることは一般に難しい。

※ n 量子系では、ハミルトニアン H は $2^n \times 2^n$ の行列であり計算量が大きくなる

→変分原理を用いてより低い期待値 ε を探索することで、最小固有値の近似値を求めることができる

最適化のための変分量子アルゴリズム

変分原理を活用したアルゴリズム(VQA)

$$E[\psi] = \langle \psi | H | \psi \rangle = \varepsilon \geq \varepsilon_0$$

- ・ 変分原理を用いて量子変分アルゴリズム(VQA)が開発されている

例：量子変分固有値ソルバー(VQE)

→ ε を出来るだけ小さくするために、変分原理を活用するソルバー

→物性物理や応用化学で応用される

量子近似最適化アルゴリズム(QAOA)

→コスト関数の近似解を求めるために、変分原理を活用するアルゴリズム

→最適化問題で応用される

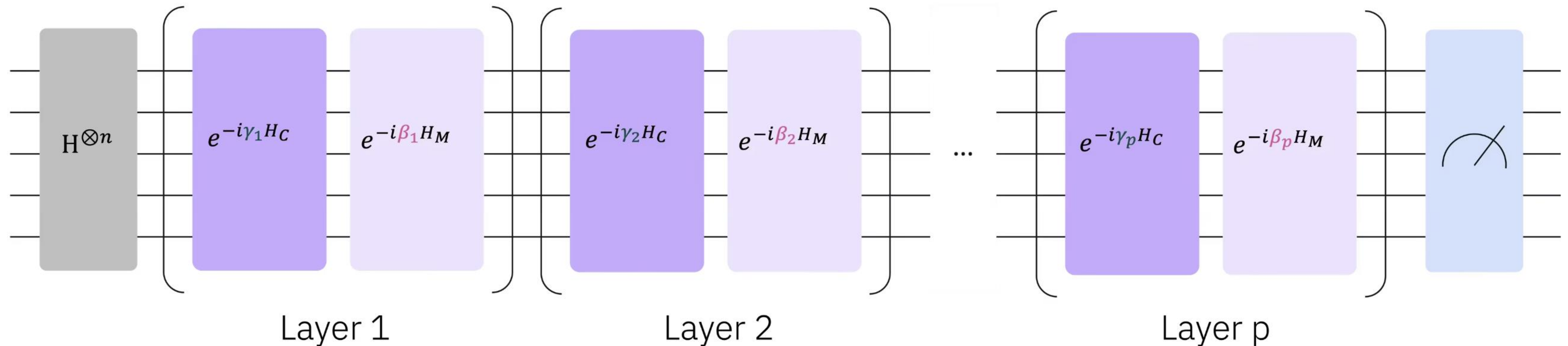
→今日のハンズオンではQAOAをつかった演習を行います

QAOAとは？

- ・最適化問題に対する近似値を得ることを目標に、変分原理を活用するアルゴリズム
- ・初期状態を既知な行列(例えば $H^{\otimes n}|0\rangle$)とユニタリ一行列 $U(\theta)$ を用いて設定

$$|\psi(\theta)\rangle \equiv U(\theta)H^{\otimes n}|0\rangle$$

→ $U(\theta)$ を最適化することで、筋の良い $|\psi\rangle$ を求める。(下図は p 組の β と γ で $U(\theta)$ を最適化)

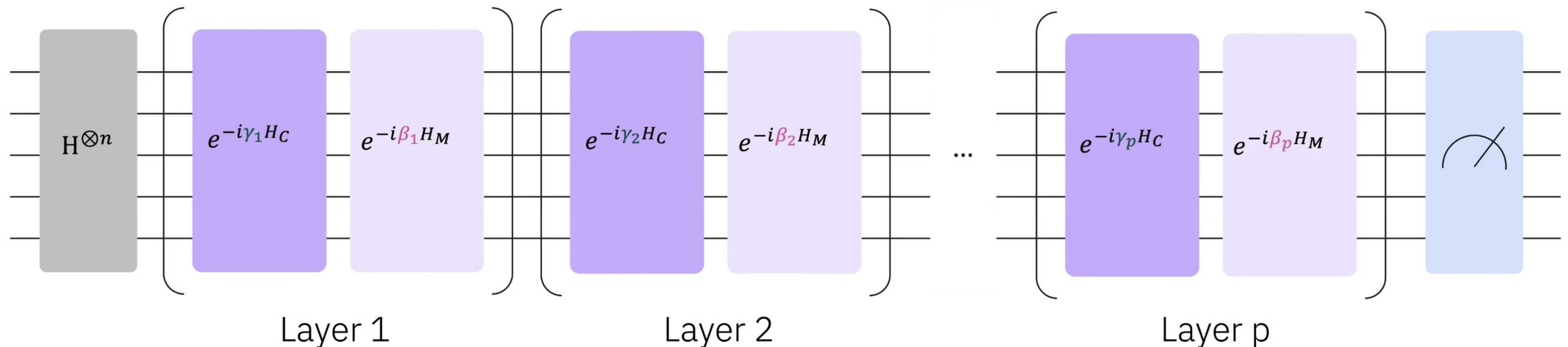


※ H_C は解きたい問題を、 H_M は初期状態を表すハミルトニアン(量子断熱計算を基に設計)

※ QiskitではQAOAAnsatzで実行可能(H_C , H_M も作成してくれる)

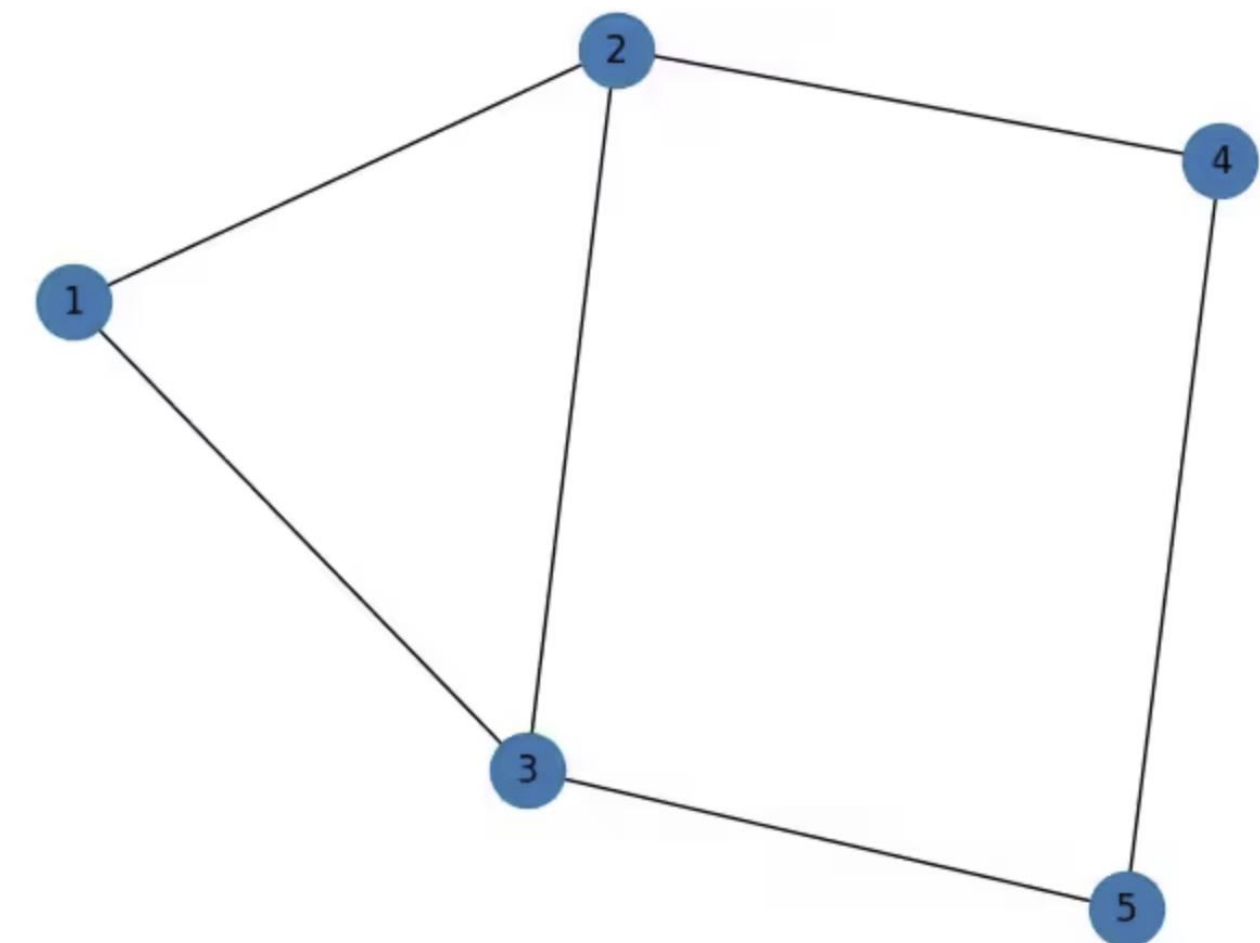
QAOAの特徴

- ・ $2^n \times 2^n$ の行列となるハミルトニアン H に対し、 $2p$ 個のパラメータを最適化することで固有値 ε を最小化する
- ・ 「量子回路を実行→コスト関数を計算しパラメータ(下図の場合は β, γ)を更新」を繰り返す
→量子アルゴリズムと古典アルゴリズムのハイブリッド計算



QAOAの最適化問題への適用

- 以下の手続きで最適化問題を解く
 - 問題をQUBO形式へ定式化
 - 定式化した問題をハミルトニアンへ変換
 - ハミルトニアンを量子回路に変換
 - QAOAを使ってハミルトニアンに対する固有値を最適化
 - 最適化された固有値が、問題に対する筋の良い解を与える(はずである)
- 最適化問題をグラフ問題として解いてみる
→今回はMax-Cut問題を解く



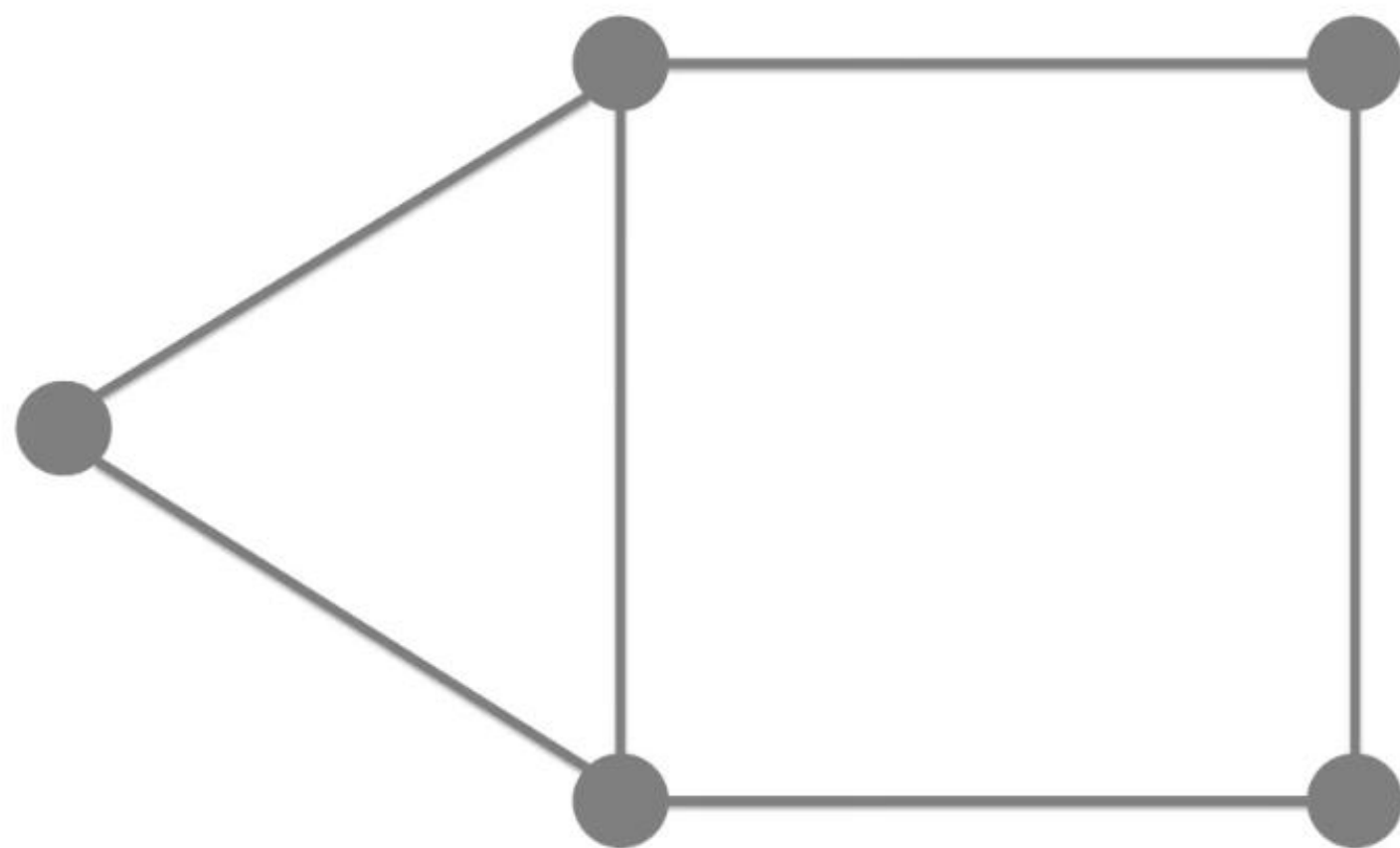
Max-Cut問題とは？

- ・グラフに対して、ノードを2分割する際に、エッジそれぞれの重みの和を最大化する方法を探索する問題

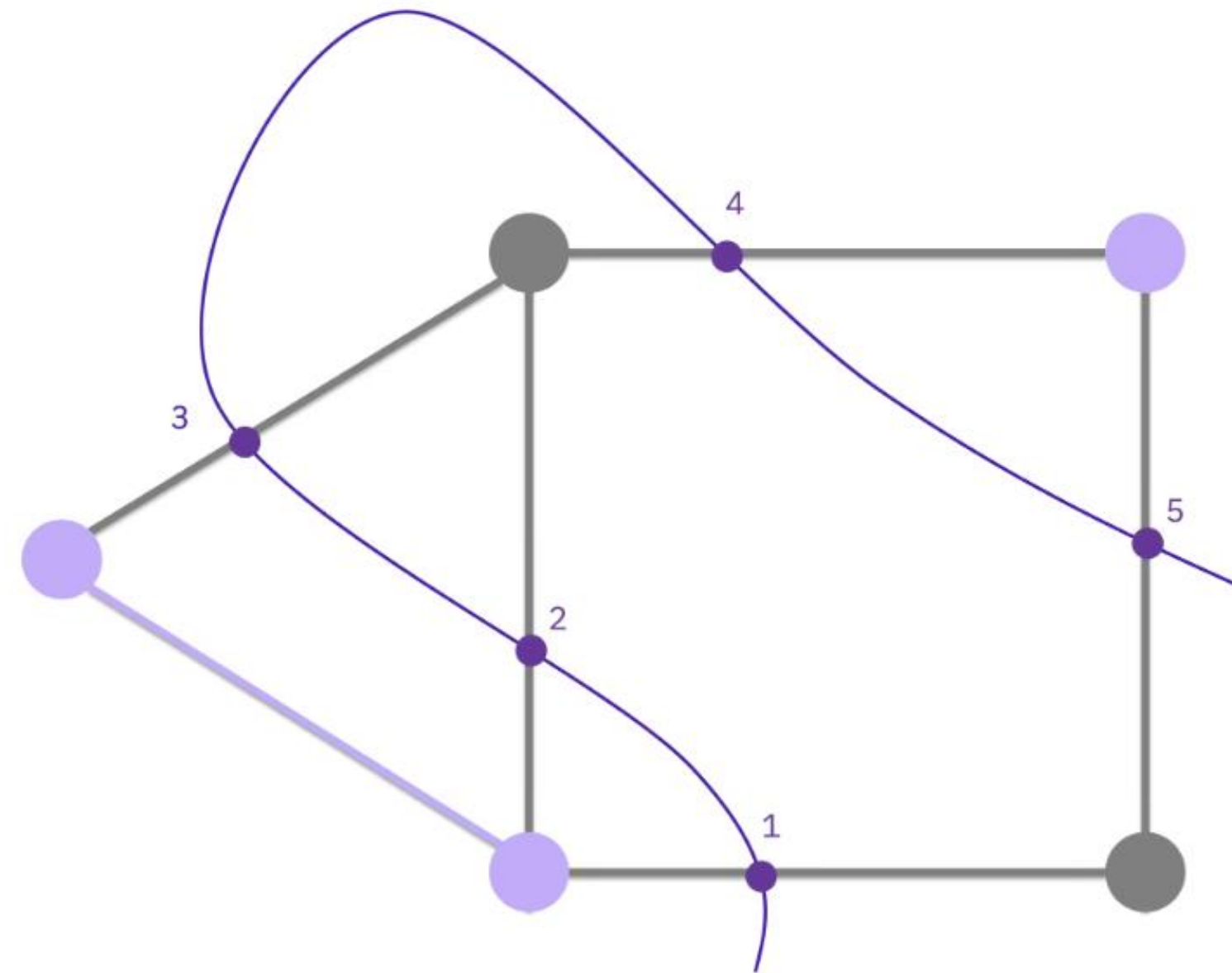
→基本的には厳密解を求めることが難しい(NP-hard)

※画像処理やクラスタリング、マーケティング等に応用

- ・今回は各重みが1(辺の数が最大になる問題)として問題を解く



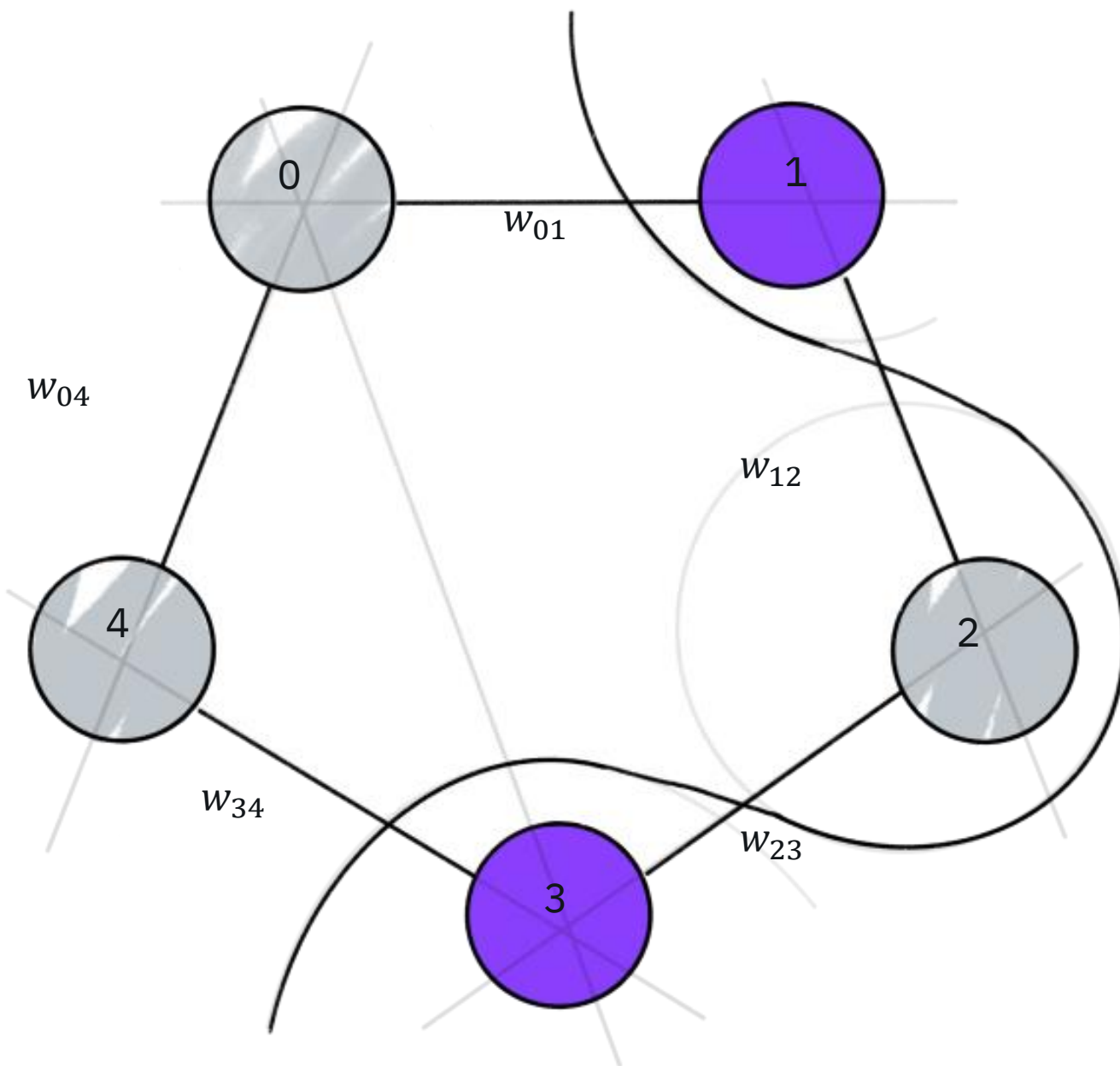
今回取り扱うグラフ



－ 2分割した例(この場合は5か所)

Max-cut応用例：マーケティング戦略

マーケティング戦略： 個々人の影響の度合いが分かっている場合に
無料の試供品をどの人に配れば、利益が最大になるか？



多くの人々が相互に作用し影響を及ぼしあうシステム

- グラフの頂点：個人
- 頂点間の結合：個人がお互いの購買決定に影響を与えるか
- 重み w_{ij} ： i が試供品を手に入れた後に、 j が製品を購入する確率

頂点を二つのグループ（試供品をあげる、あげない）に分けるとき、
どの頂点の間の結合をカットすれば、カットされた重みの合計が最大になるか？

ハミルトニアン

$$H = -\frac{1}{2} \sum_{i,j} w_{i,j} (1 - Z_i Z_j) \quad (\text{ここで、} Z_i = \{1, -1\}, w_{i,j} \text{ は重み}) \quad : \text{最小値とそのときの} Z_i \text{の値を求める}$$

Max-Cut問題の定式化

- ・各ノードは最終的に二分されるので、それぞれのノードに0 or 1を割り振る

→ $x_i \in \{0,1\}, i \in \{1,2,3,4,5\}$

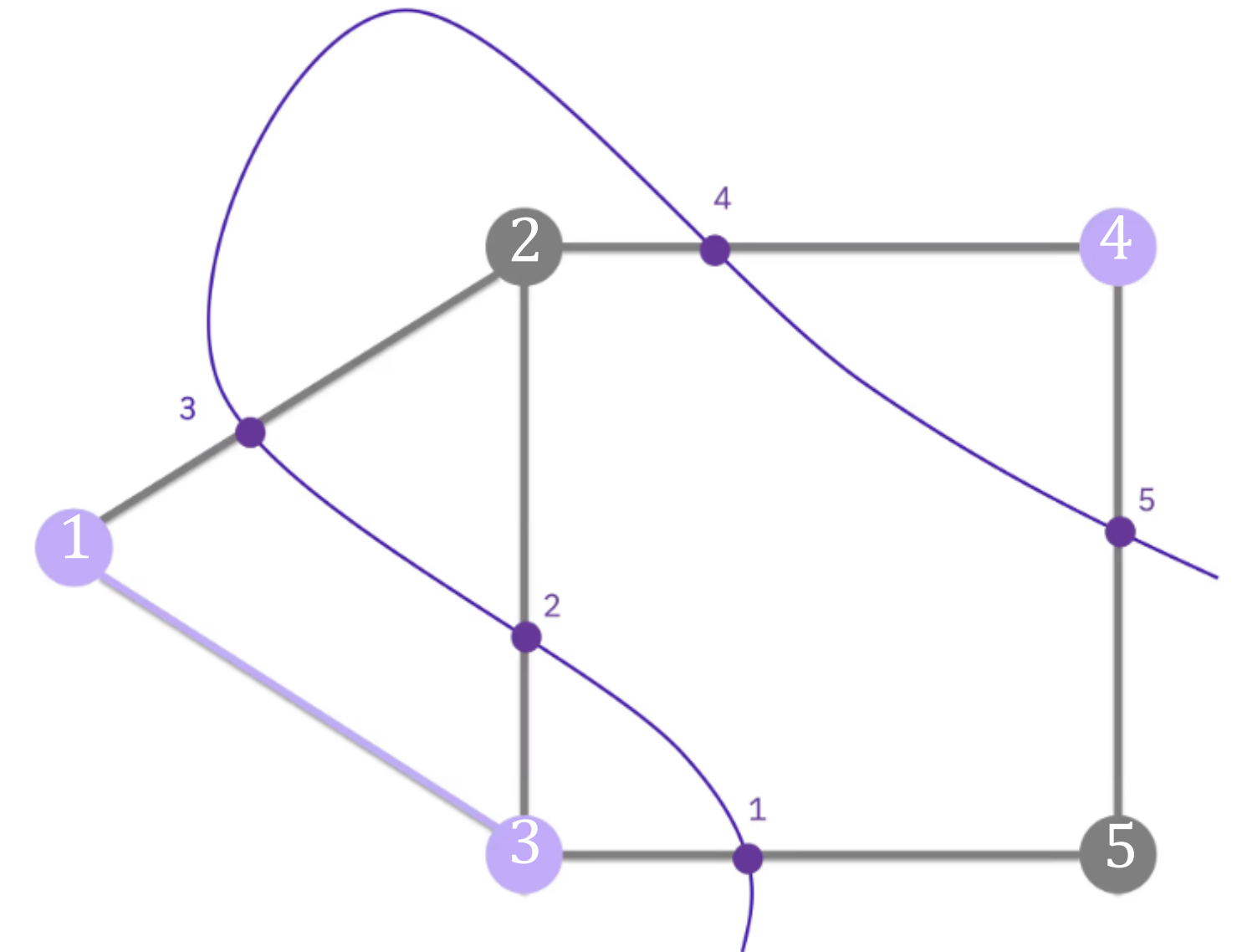
- ・エッジでつながっているノード間で値が異なる場合、そこで分割が行われているため、それを1として数え上げる。

→ 最大化する目的関数： $\sum_{(i,j)} [x_i(1-x_j) + x_j(1-x_i)]$,
 $(i,j) \in [(1,2), (1,3), (2,3), (2,4), (3,5), (4,5)]$

※ $(x_i, x_j) = (0,1), (1,0)$ のとき1、 $(x_i, x_j) = (0,0), (1,1)$ のとき0

※ QUBO形式とする必要がある(2次まで、制約条件なし、変数が二値)

QUBO : Quadratic Unconstrained Binary Optimization



QUBOからハミルトニアンへ変換

目的関数： $\sum_{(i,j)} [x_i(1 - x_j) + x_j(1 - x_i)]$ where $(i,j) \in [(1,2), (1,3), (2,3), (2,4), (3,5), (4,5)]$.

変数： $x_i \in \{0,1\}, i \in \{1,2,3,4,5\}$.

- ・ 変数 x_i を変数 z_i (1,か-1をとる) に変換した後、パウリの演算子 Z へ変換する。

$$x_i = \frac{1 - z_i}{2} \text{ where } z_i \in \{-1,1\}, z_i \rightarrow Z_i (= I \otimes \cdots Z \cdots \otimes I)$$

※ $Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$ の固有値は $\langle 0|Z|0 \rangle = 1, \langle 1|Z|1 \rangle = -1$, であるため。 i 番目の qubit に変換

- ・ 加えて、変分原理の性質上、 **-1** をかけて最小化問題へ

目的関数, 変数： $\sum_{(i,j)} \left[-\frac{1}{2} (1 - z_i z_j) \right], z_i \in \{-1,1\}$

ハミルトニアン H ： $\sum_{(i,j)} \frac{1}{2} Z_i Z_j - \sum_{(i,j)} \frac{1}{2} I^{\otimes n}$ where $Z_i = I^{n-i-1} \otimes Z \otimes I^{\otimes i-1}$

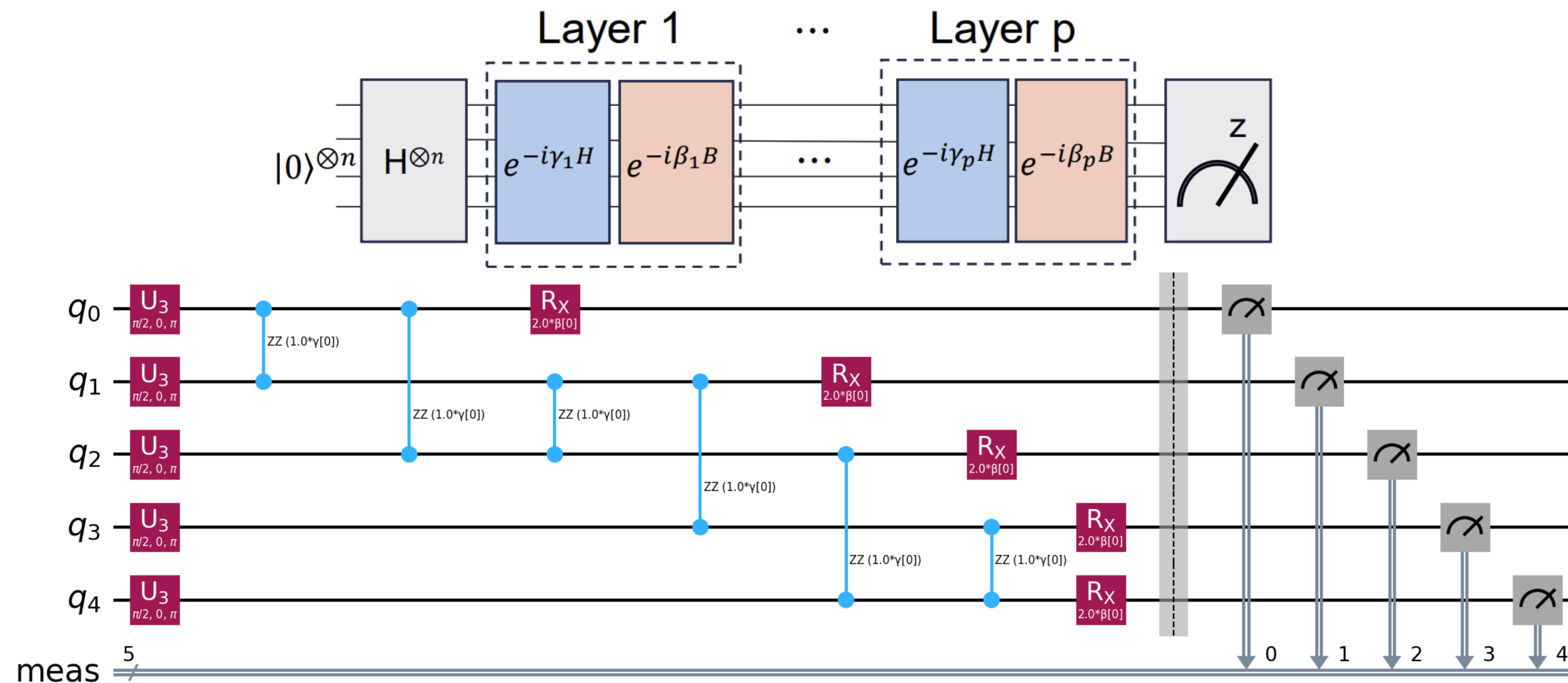
QAOAを使って解を得る

ハミルトニアン H : $\sum_{(i,j)} \frac{1}{2} Z_i Z_j$ where $(i,j) \in [(1,2), (1,3), (2,3), (2,4), (3,5), (4,5)]$

※定数項 $-\sum_{(i,j)} \frac{1}{2} I^{\otimes n}$ は省略

$$H = \frac{1}{2} (Z_1 Z_2 + Z_1 Z_3 + Z_2 Z_3 + Z_2 Z_4 + Z_3 Z_5 + Z_4 Z_5)$$

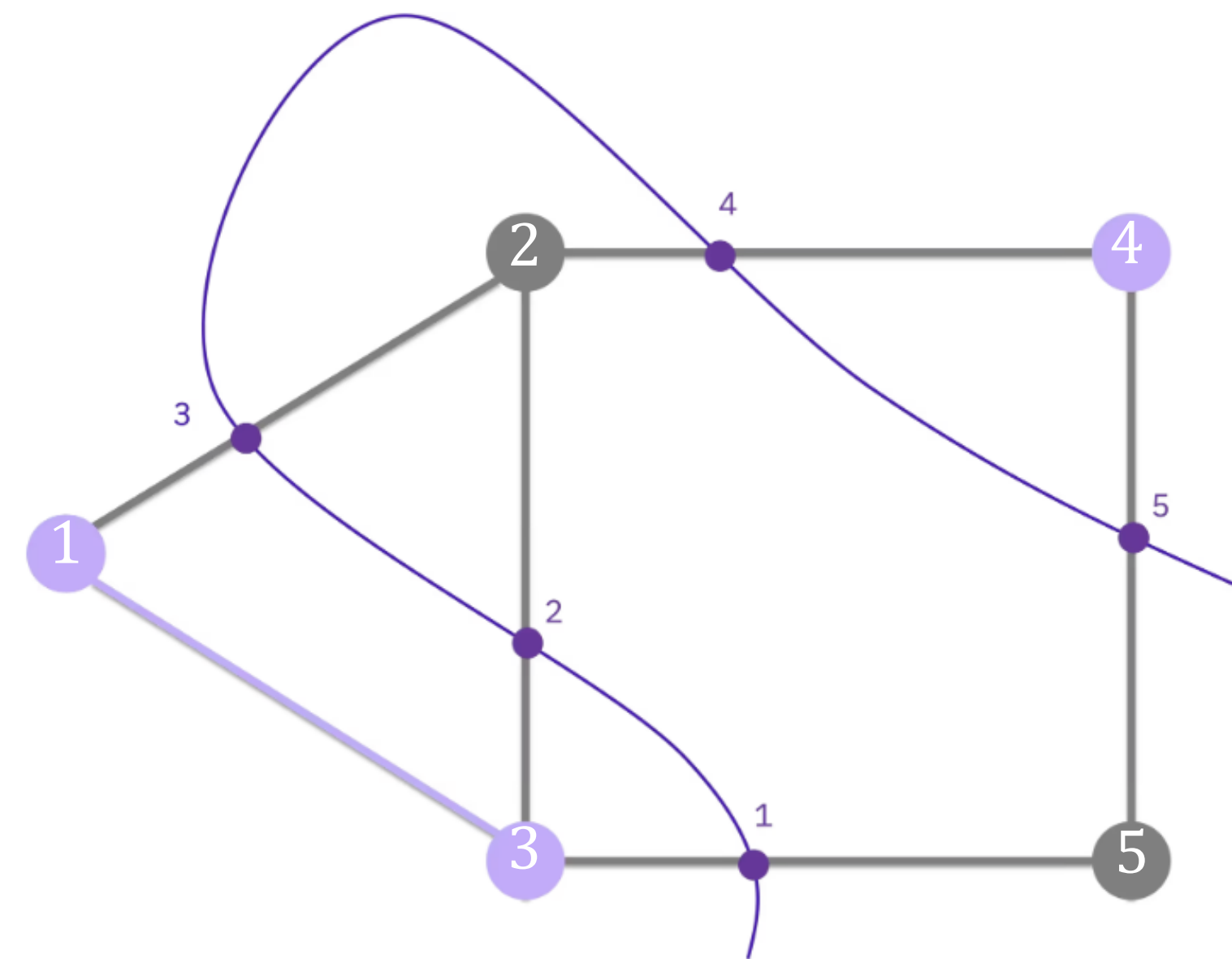
- QAOAを使用して、 $2p$ 個の β と γ でコスト関数を最適化(回路は $p = 1$ の場合)



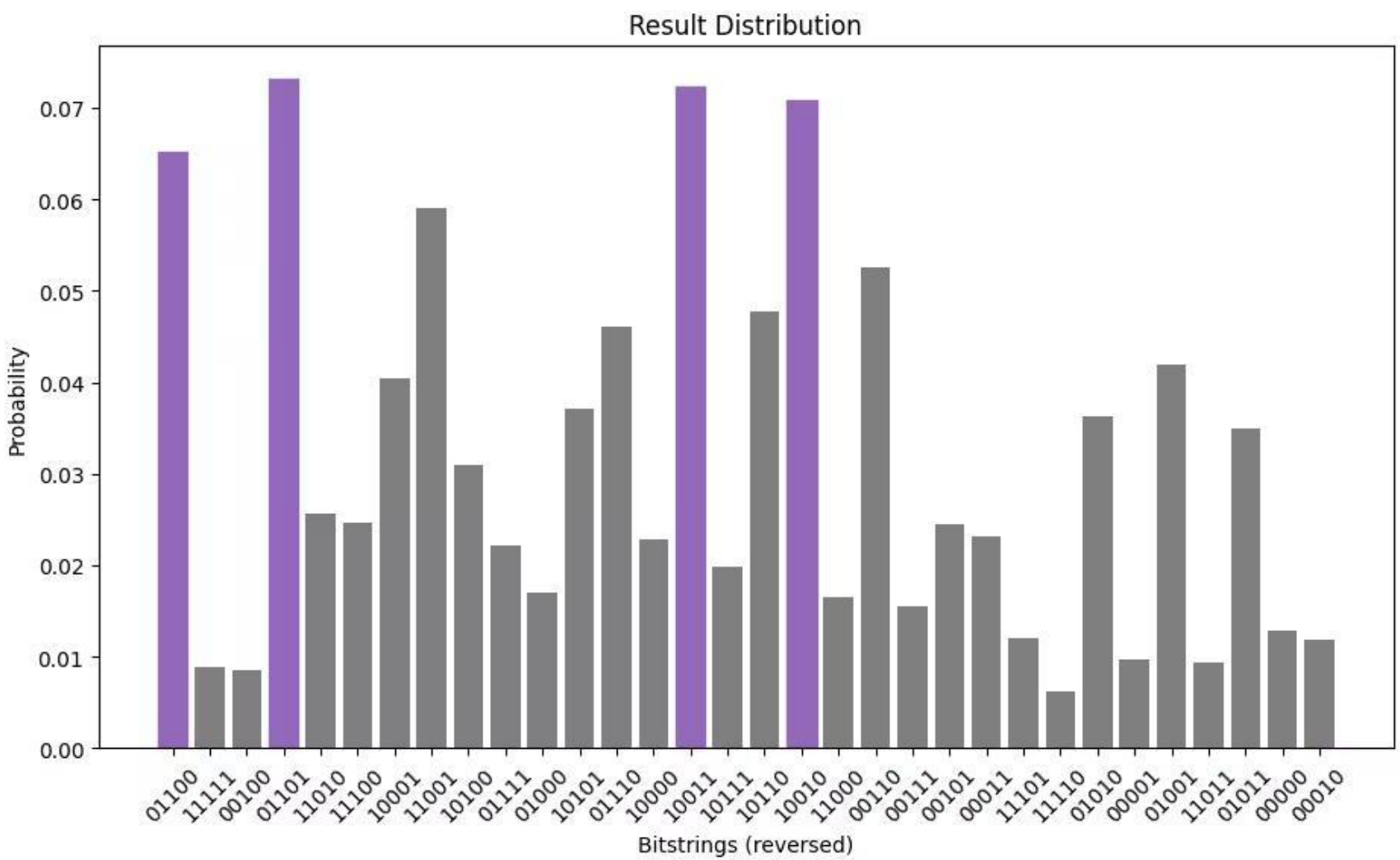
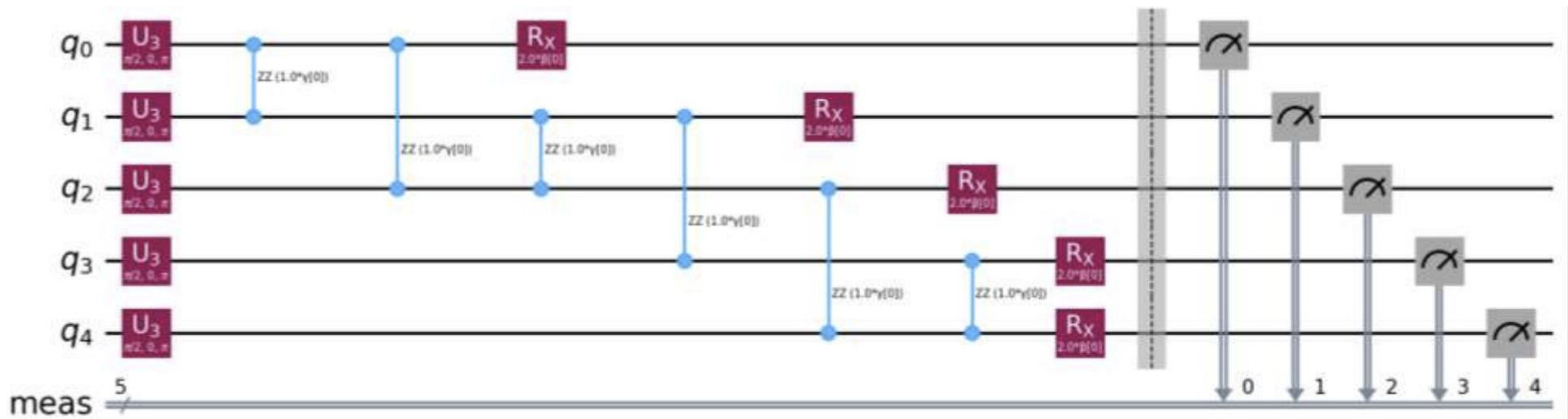
実機による計算結果

$$H = \frac{1}{2} (Z_1 Z_2 + Z_1 Z_3 + Z_2 Z_3 + Z_2 Z_4 + Z_3 Z_5 + Z_4 Z_5)$$

※最小の固有値： -5 , 対応する $|\psi\rangle = |00110\rangle, |11001\rangle, |10110\rangle, |01001\rangle$



$|\psi\rangle = |10110\rangle, |01001\rangle$ の場合

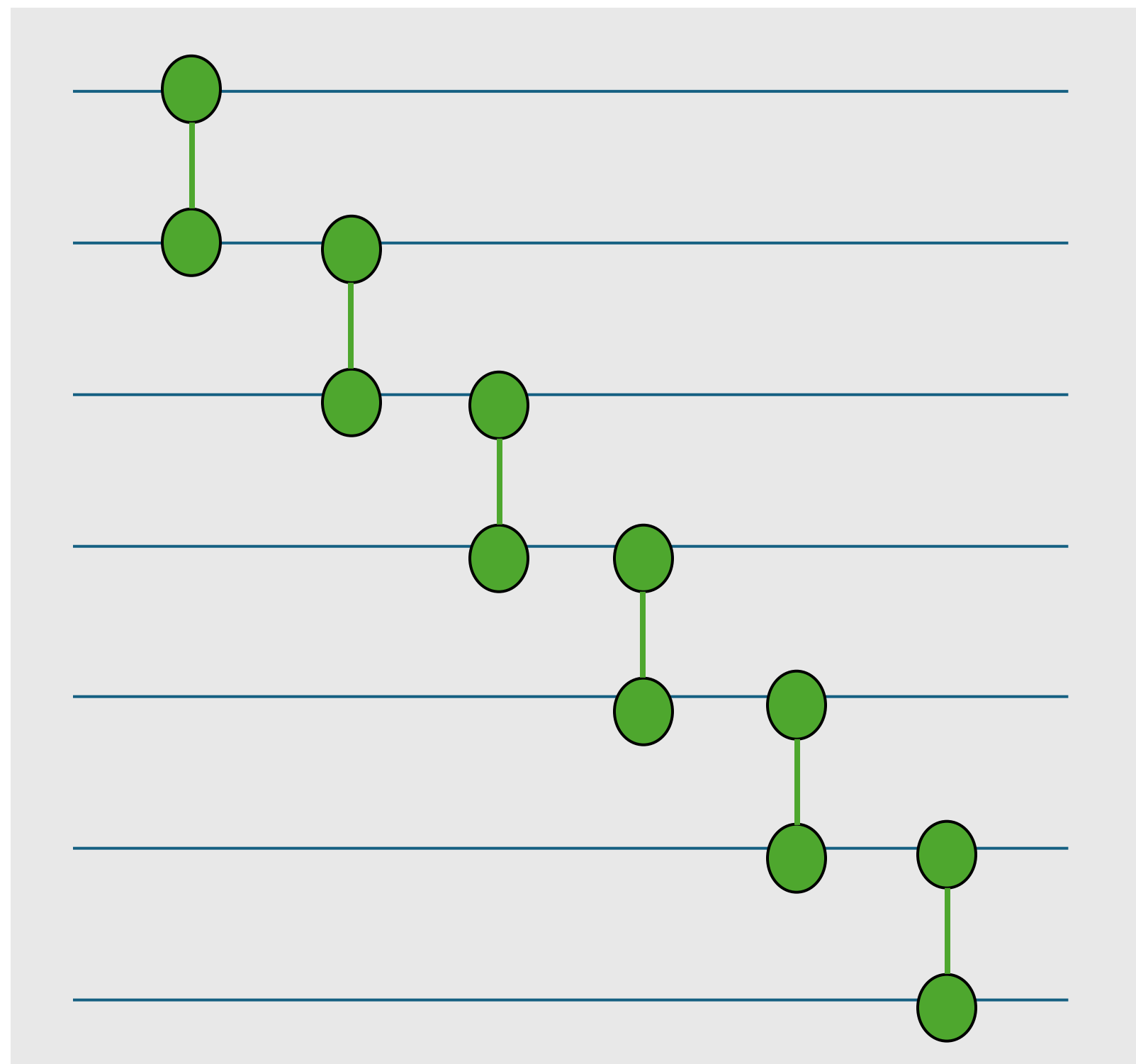


→最小固有値に対応する $|\psi\rangle$ の確率が高い

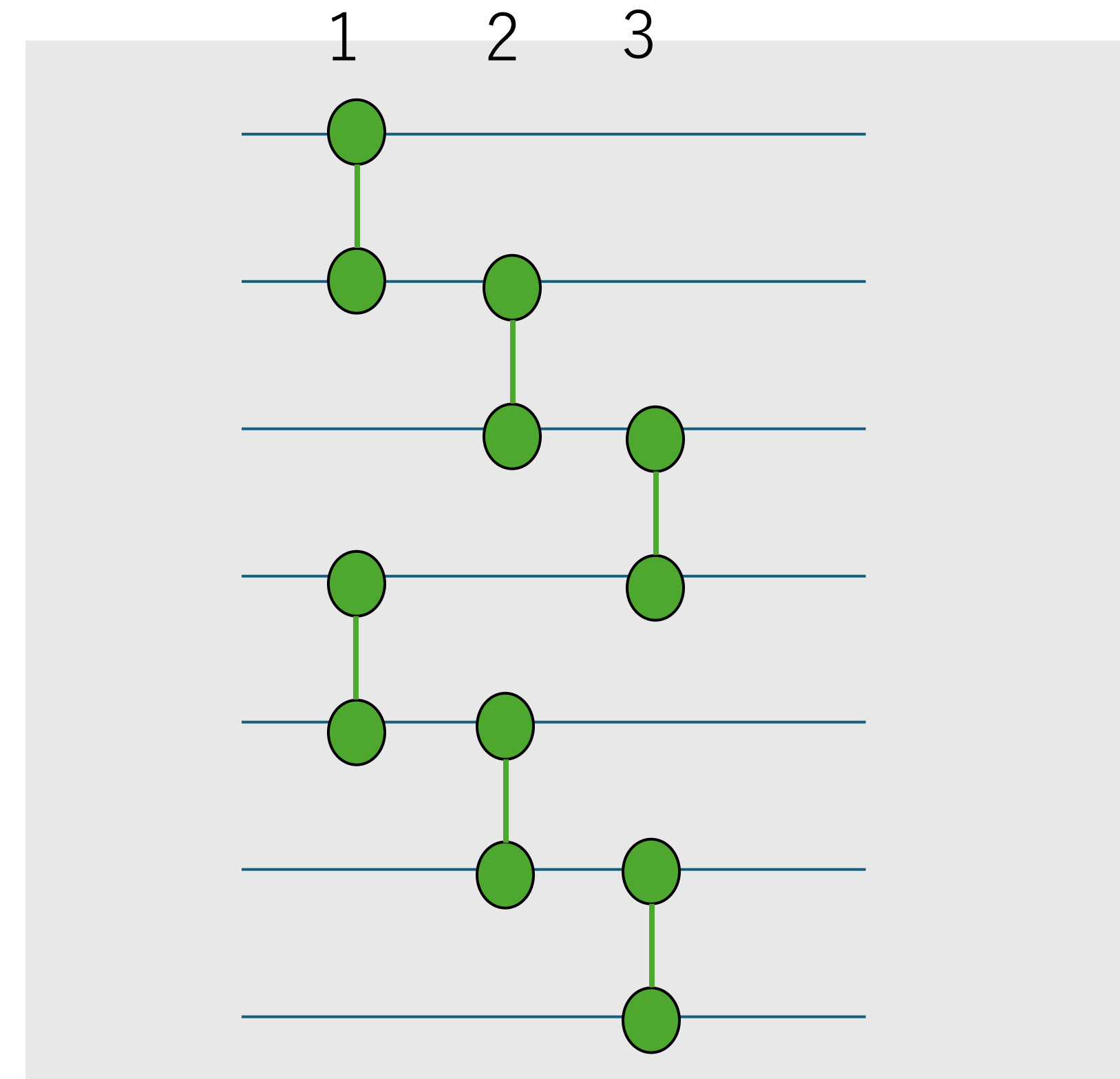
量子回路最適化の例

入れ替え可能な量子ゲートの入れ替えをして 並列実行可能なものを前にもってくる

- 量子ゲートの中には順番を入れ替えても結果が変わらないものがあり、
- QAOAで使われる左図のRZZゲートもその内の一つ



最適化なしの場合、階段状に続く



最適化ありの場合、量子回路の深さを浅くできる

演習：
QiskitをつかってMax-Cut問題を解いてみましょう

20250930_7_maxcut.ipynbを使います。

ご自分のパソコンにダウンロードしてください。

高度なアルゴリズムに関するリンク集

- チュートリアル

- Quantum approximate optimization algorithm
- Advanced techniques for QAOA
- Pauli Correlation Encoding to reduce Maxcut requirements
 - <https://quantum.cloud.ibm.com/docs/en/tutorials/quantum-approximate-optimization-algorithm>

- ベストプラクティス

- Transpile, Swap strategy, fractional gates, etc.
- <https://github.com/qiskit-community/qopt-best-practices>

- Qiskit Functions

- Higher-order binary optimization with Q-CTRL's Optimization Solver
 - <https://quantum.cloud.ibm.com/docs/en/tutorials/solve-higher-order-binary-optimization-problems-with-q-ctrls-optimization-solver>
- Iskay Quantum Optimizer (Kipu Quantum)
 - <https://quantum.cloud.ibm.com/docs/en/guides/kipu-optimization>

Max-cut 問題の応用

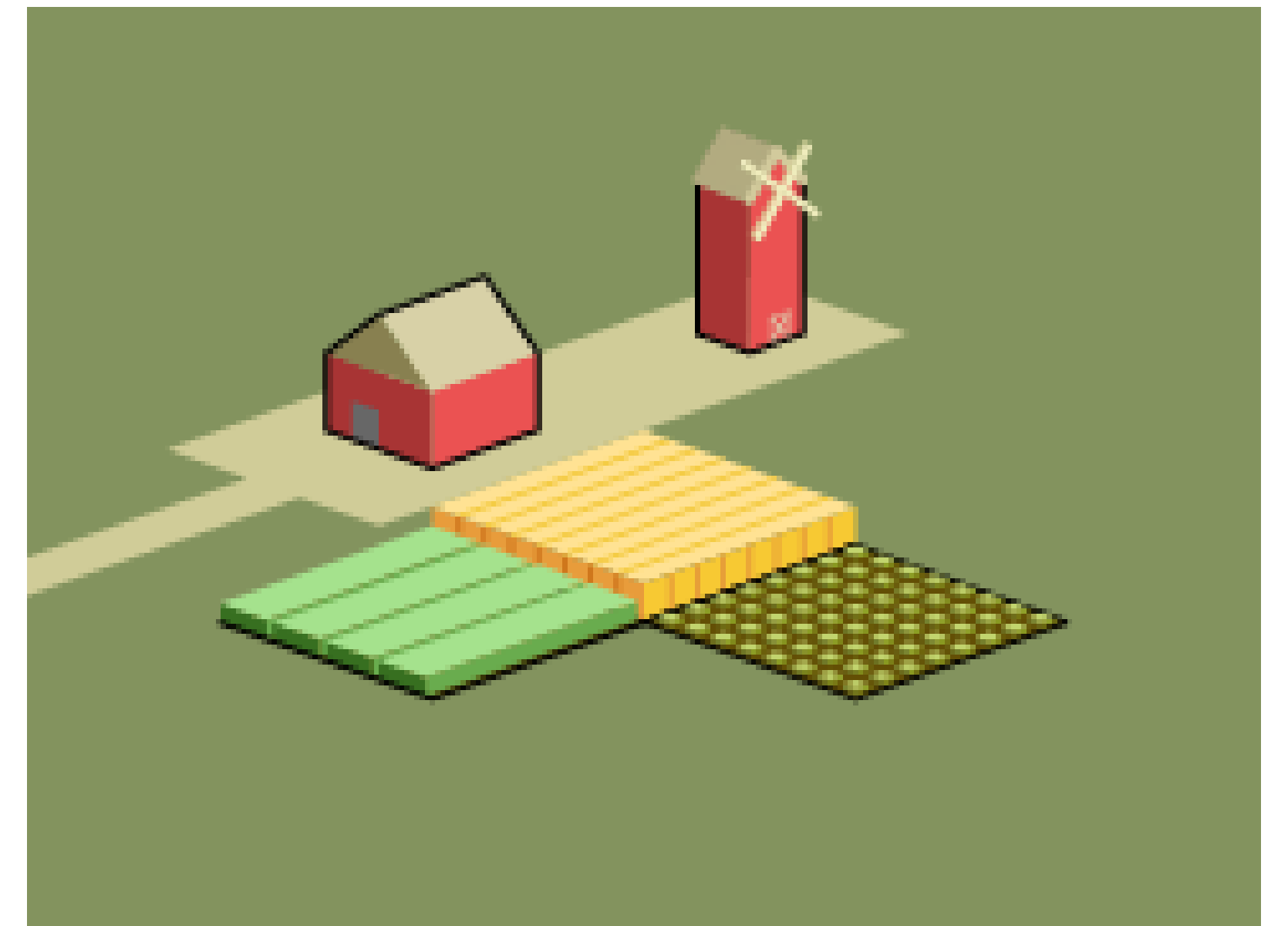


Max-Cut問題は、その理論的重要性に加え、**ネットワーク設計、統計物理学、VLSI設計、回路レイアウト設計、データクラスタリング**など様々な分野で応用されています

QAOAアルゴリズムを利用したMax-cut問題の詳しい解説が以下のチュートリアルでも紹介されています。

<https://quantum.cloud.ibm.com/docs/ja/tutorials/quantum-approximate-optimization-algorithm>

今日の演習課題： 「農地の収穫量最適化問題」



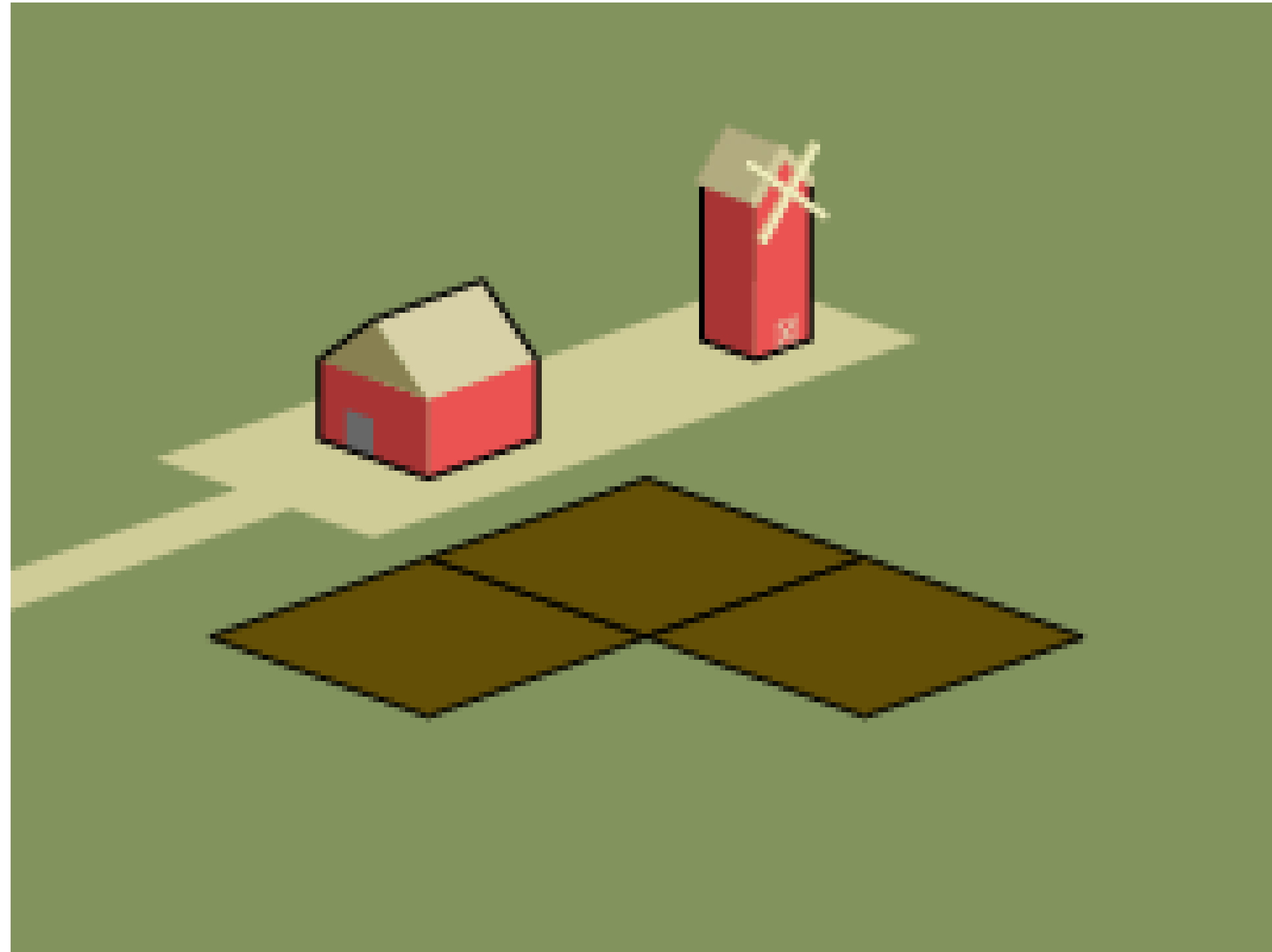
農作物の収穫量問題について

2021年の IBM Quantum Challengeでは、アフリカのQiskit Communityメンバーが、現在のアフリカが抱える課題をテーマに問題設定を行いました。そのうちの 하나가「**農業の生産性**」というテーマでした。アフリカの農業は生産性の観点から、人口増加による食料需要に供給が追いつかないなどの課題があることが指摘されています。

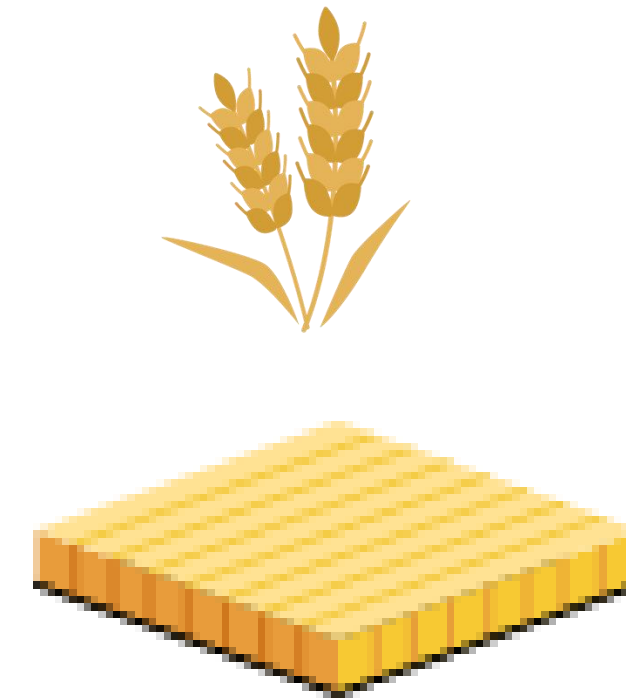


画像出典: The Conversation Com

農地の収穫量を最大化するには？

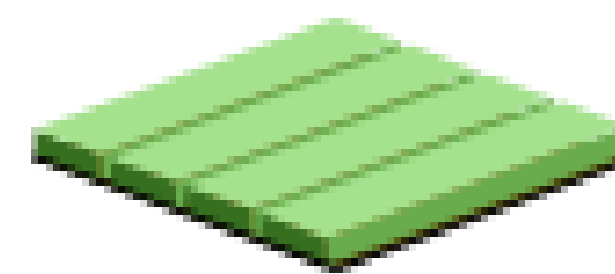


3 ヘクタール



Wheat

麦



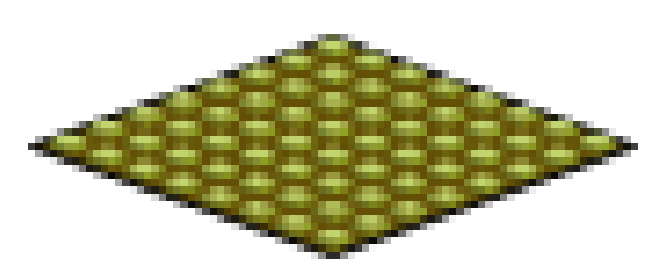
Soybeans

大豆



Maize

トウモロコシ

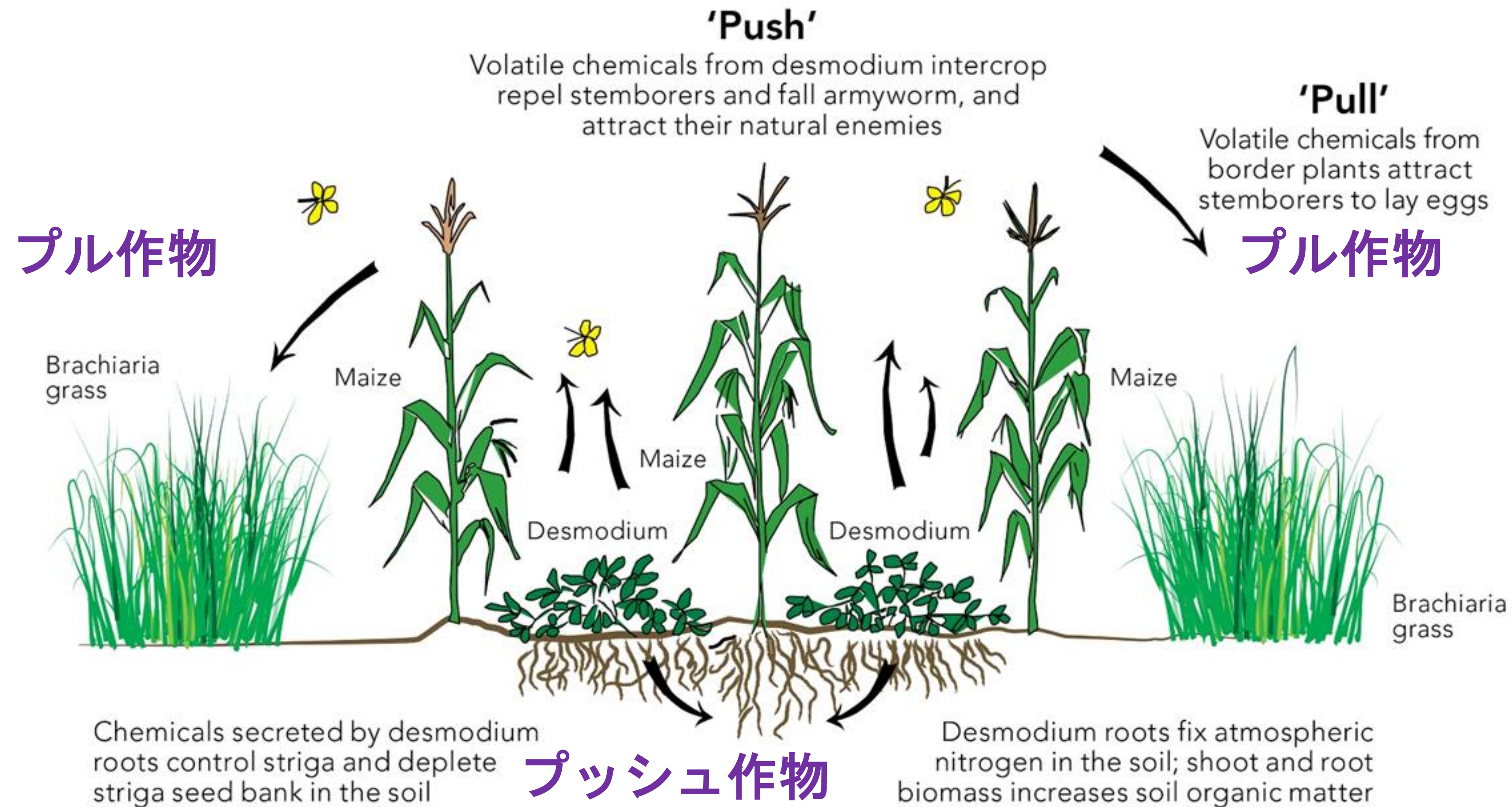


PushPull

プッシュプル

プッシュプル農法

プッシュプル農法とは、害虫を寄せ付けないプッシュ作物と、害虫を引き寄せるプル作物をペアで栽培することです。化学薬品をつかわずに害虫被害を抑制する農法。



演習：
Qiskitをつかって農地収穫量問題を解いてみましょう







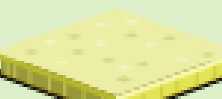

20250930_7_cropyield.ipynbを使います。

ご自分のパソコンにダウンロードしてください。

農法別の収穫量係数

農法には「**単作**」「**間作**」「**プッシュプル農法**」の3種類があります。単作とは、一つの作物だけを栽培する方法です。単作では、病気や害虫の影響を受けやすく、収穫量全体に影響を及ぼします。間作とは、収穫量を 増やすために2つの異なる作物を選択することです。組合せによって、両方の収穫量が増えたり、逆に収穫量が減ったりする場合があります。プッシュプル農法とは、害虫を寄せ付けないプッシュ作物と、害虫を引き寄せるプル作物をペアで栽培することです。

注) 以下は**大豆を単作したときの収穫量を「1」としたときの比較係数**であり、**単位はヘクタールではありません**。

| |  小麦 |  大豆 |  トウモロコシ |  プッシュプル |
|--|--|--|--|--|
| 小麦  | 2 | 2.4 | 4 | 4 |
| 大豆  | 2.4 | 1 | 2 | 1 |
| トウモロコシ  | 4 | 2 | 4 | 5 |
| プッシュプル  | 4 | 1 | 5 | -- |

単作
間作
プッシュプル

定式化（農地の収穫量）

目的関数（最大化）

maximize

$$2 \times \text{orange} + \text{green} + 4 \times \text{yellow}$$

$$+ 2.4 \times (\text{orange} \times \text{green})$$

$$+ 4 \times (\text{orange} \times \text{yellow})$$

$$+ 4 \times (\text{orange} \times \text{dark green})$$

$$+ 2 \times (\text{green} \times \text{yellow})$$

$$+ 1 \times (\text{green} \times \text{dark green})$$

$$+ 5 \times (\text{yellow} \times \text{dark green})$$

1 次式のパート：
単作の係数をあてはめていく

2 次式のパート：
間作とプッシュアップの係数をあてはめていく

制約条件

subject to

$$\text{orange} + \text{green} + \text{yellow} + \text{dark green} \leq 3$$

$$0 \leq \text{orange} \leq 1$$

$$0 \leq \text{green} \leq 1$$

$$0 \leq \text{yellow} \leq 1$$

$$0 \leq \text{dark green} \leq 1$$

(1) 選べる作物の品種は3種類以下

(2) 各作物は0 haまたは1 ha作付け可能

まとめ

最適化問題は意志決定の手段として身近な問題から大きな問題まで、あらゆるところで見受けられる問題であり、**量子コンピュータの応用分野として注目**されています

多くの組合せの中から最適な解を見つける問題は、データ量が爆発的に膨れ上がり（**組み合わせ爆発**）、ふつうのコンピュータでは**計算が困難**です

計算するためには、**現実問題を定式化**することが必要で、与えられた制約条件の下、ある**コスト関数**（目的関数）を**最大**または**最小**にする答えを見つけることがゴールです

量子コンピュータで最適化問題を解くにはQUBO/イジングモデルへの変換が必要です。**Qiskit**にはこれらの変換を行ってくれる便利な変換ツールが備わっており、最適化問題を解きやすくしてくれています。

皆さんの身近にある最適化問題についてぜひ色々と考えてみてください。小さな問題からで良いので、量子コンピュータで解いてみることにチャレンジしてみてください。

Thank you