2024/09/17

Ryosuke Nagai (M1)
Department of Intelligence
Science and Technology

KYOTO UNIVERSITY

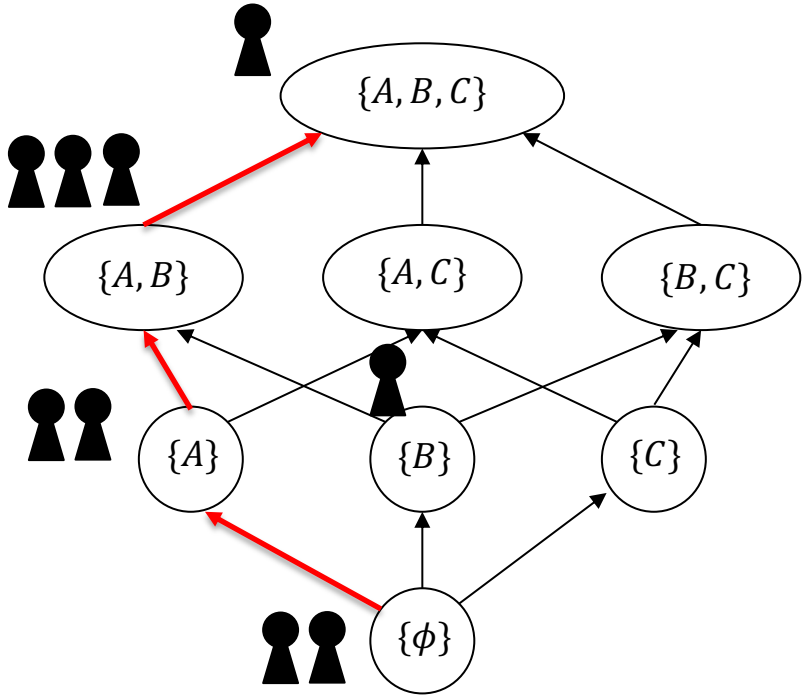
集団テストからの時系列復元:

- 最終的にやりたいこと

問題A,B,Cがあった時に、その人が次に解くべき（解きやすい）問題がどれか当てる

各問題には、その背景に習得すべき知識（依存関係）があり、
その順番が分かれば嬉しい

A→B→Cで学ぶといい



全ての問題を解けるようになるまでの遷移の順番を知りたい

Step1：遷移経路が分かってる人工データ

Step1：遷移経路が分かってる人工データ
目的

- 問題ごとの依存関係を前提とした人工データを作成
- 遷移経路は分かる

```
# 問題の依存関係の行列 A
A = np.array([
    [0, 0, 0, 0, 0],   # 初期状態
    [1, 0, 0, 0, 0],   # 問題1は初期状態のみに依存
    [0, 1, 0, 0, 0],   # 問題2は問題1に依存
    [0, 0, 1, 0, 0],   # 問題3は問題2に依存
    [0, 0, 1, 1, 0]    # 問題4は問題2、問題3に依存
])
```

問題の依存関係：
この場合だと４問出題されたと想定

例えば、問題2は問題1が解けた後に解けると想定している

遷移経路のデータから依存関係を復元したい

Step1：遷移経路が分かってる人工データ作り方（適切か分かってない）

```python
# 問題の依存関係の行列 A
A = np.array([
    [0, 0, 0, 0, 0],
    [1, 0, 0, 0, 0],
    [0, 1, 0, 0, 0],
    [0, 0, 1, 0, 0],
    [0, 0, 1, 1, 0]
])
```

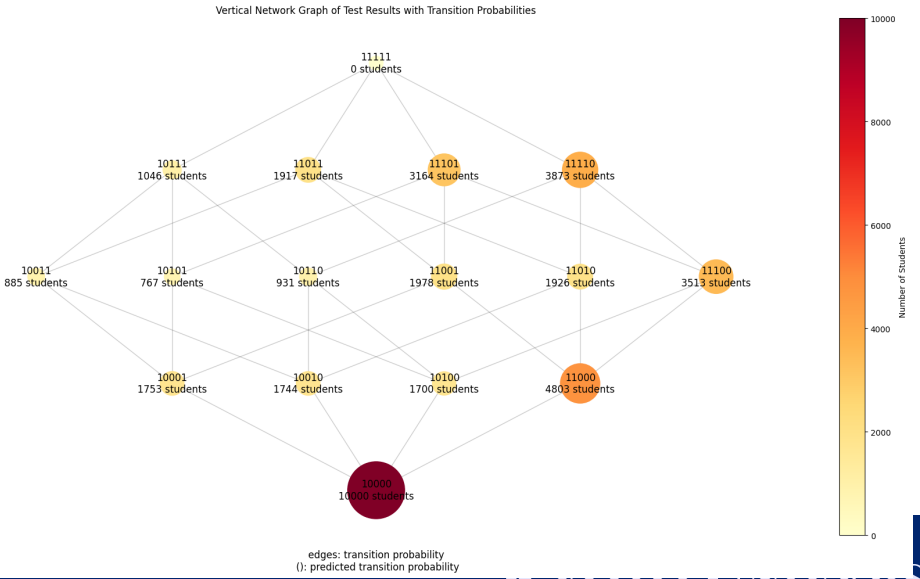
- まずは依存関係行列Aからデータ生成

初期状態$\boldsymbol{X}_0 = [0,0,0,0,0]$からスタートし、
まだ解けてない問題(k)が次に解ける確率（遷移確率）を求める

$$\mathrm{p_j} = \frac{\exp(\boldsymbol{X}_i \boldsymbol{A}_j / sum(\boldsymbol{A}_j))}{\sum_{k \in \{l | X_i[l]=0\}} \exp(\boldsymbol{X}_k \boldsymbol{A}_k / sum(\boldsymbol{A}_k))}$$

その遷移確率を元に、
次の状態を生成する。

繰り返してデータセットを
作る。



Step1：遷移経路が分かってる人工データモデル

- ５次元→５次元のFC層とSoftmax層を用いたシンプルなモデルを実装し、遷移確率を比較する。

```python
class Model(nn.Module):
    def __init__(self, num_questions):
        super(Model, self).__init__()
        self.fc = nn.Linear(num_questions, num_questions)  # 全結合層

    def forward(self, x):
        x = self.fc(x)  # 全結合層の適用
        x = F.softmax(x, dim=1)  # ソフトマックスを適用
        return x
```

- クロスエントロピー損失、L2正則化weight_decay=0.001
- 学習データ1000人分、遷移ステップ5回遷移なので、5000個のデータセット

Step1：遷移経路が分かってる人工データ
結果

- データ生成元の依存関係行列とモデルのFC層の出力の比較

これを元にsoftmaxして確率求めているので1の場所が大きいほど良い

```
# 問題の依存関係の行列 A
A = np.array([
    [0, 0, 0, 0, 0],
    [1, 0, 0, 0, 0],
    [0, 1, 0, 0, 0],
    [0, 0, 1, 0, 0],
    [0, 0, 1, 1, 0]
])
```

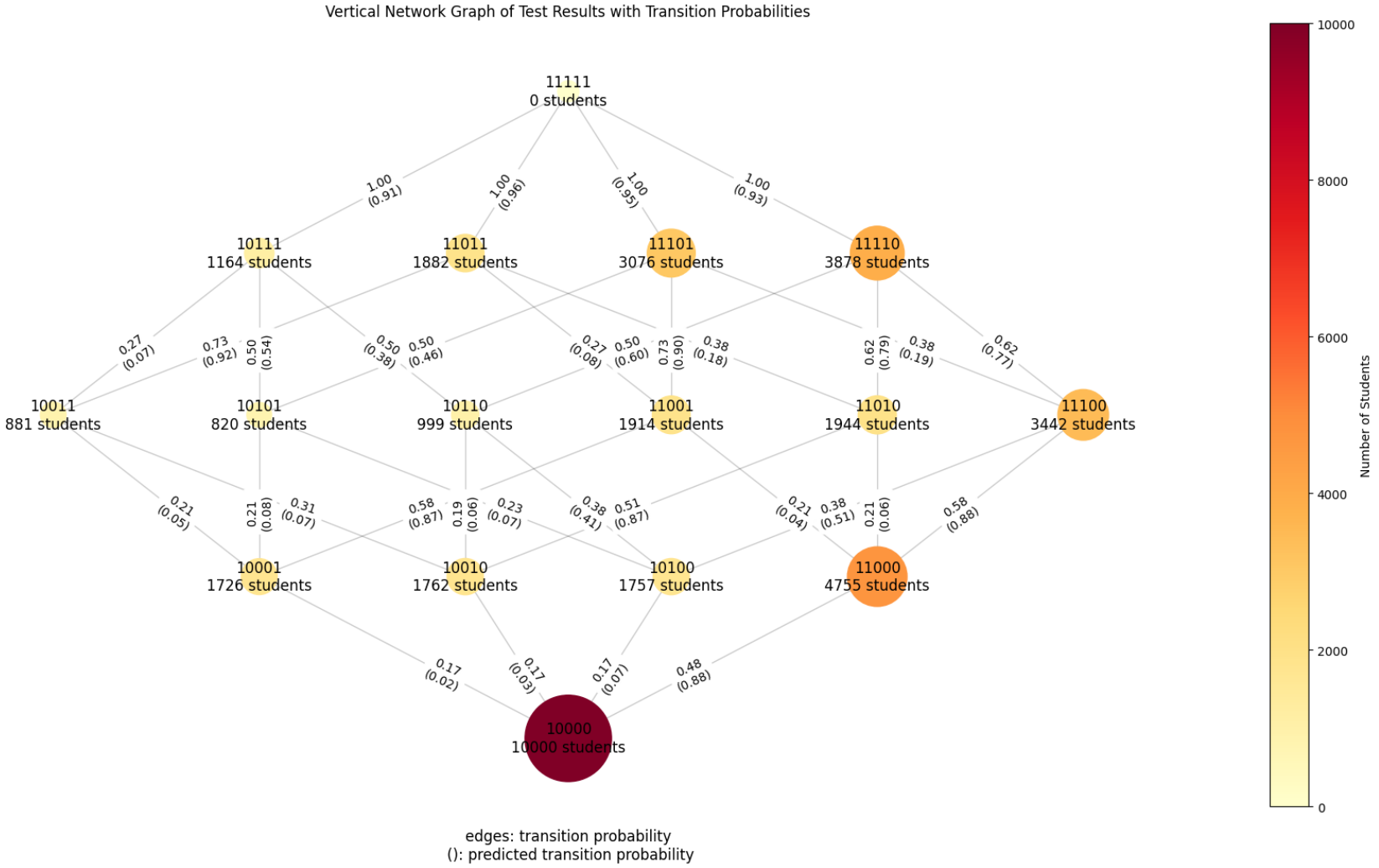
実際

```
[[-0.9905, -0.4473, -0.3180, -0.2291, -0.1969],
 [ 1.4869, -3.7660, -0.4600,  0.5670,  0.5640],
 [ 0.2104,  2.5845, -3.5142,  0.4133,  0.6997],
 [-0.1668,  0.6523,  2.6231, -2.8007,  0.9477],
 [-0.5400,  0.9765,  1.6689,  2.0496, -2.0145]],
```

学習結果

Step1：遷移経路が分かってる人工データ結果

- 遷移確率（ノードの値は実際の遷移確率。()内は予測値）



Step2：遷移経路が分からない人工データ

集団テストからの時系列復元:
Step2：遷移経路が分からない人工データ

- 問題ごとの依存関係を前提とした人工データを作成
- ただし、遷移経路は分からない
- 各ノードの分布のみから依存関係を復元したい

- 学習データはどうすればいい？（考え中）

Step3：遷移も分かってる人工データ

集団テストからの時系列復元:
Step3：遷移も分かってる人工データ

- 実データを学習データ・テストデータに分けて、学習データで作ったグラフ構造の分布通りにテストデータが分布しているか検証？