2019/05/05

『アルゴリズムとデータ構造』

19章 ヒューリスティック探索

B4 yuui

ヒューリスティック探索

知識に基づかないアルゴリズムでは解決できない
さまざまな難しい問題を解決

- **問題に関する情報を活用して解をより効率的に検出する状態空間探索の１つ**
- **パズルやゲームにみられる難問に対して，筋道を立てて問題を解くアルゴリズムを見出すことができない場合に使用**

d-hacks
JIN NAKAZAWA LAB

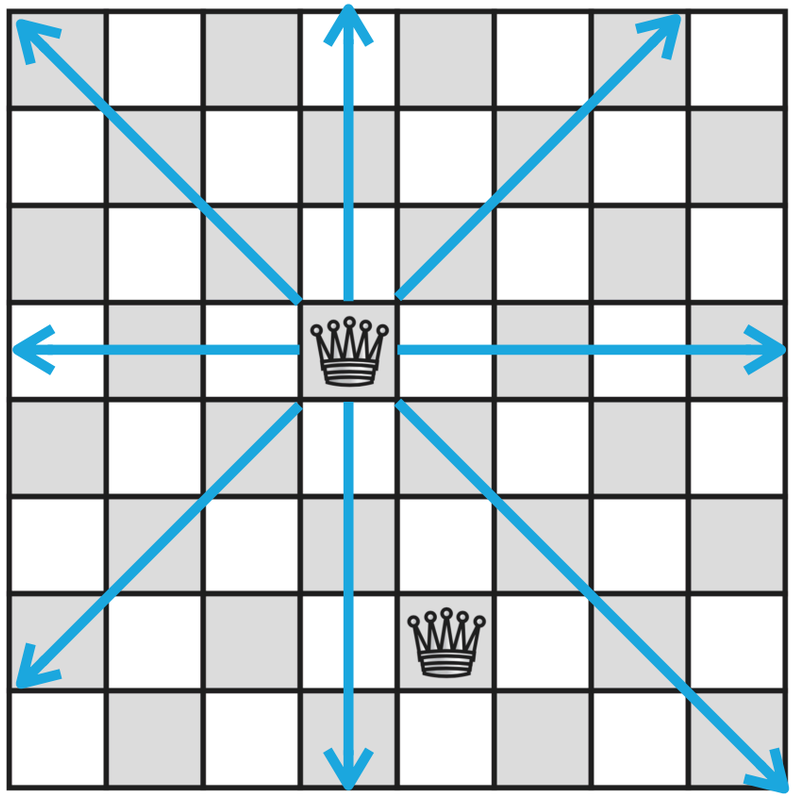
ヒューリスティック探索

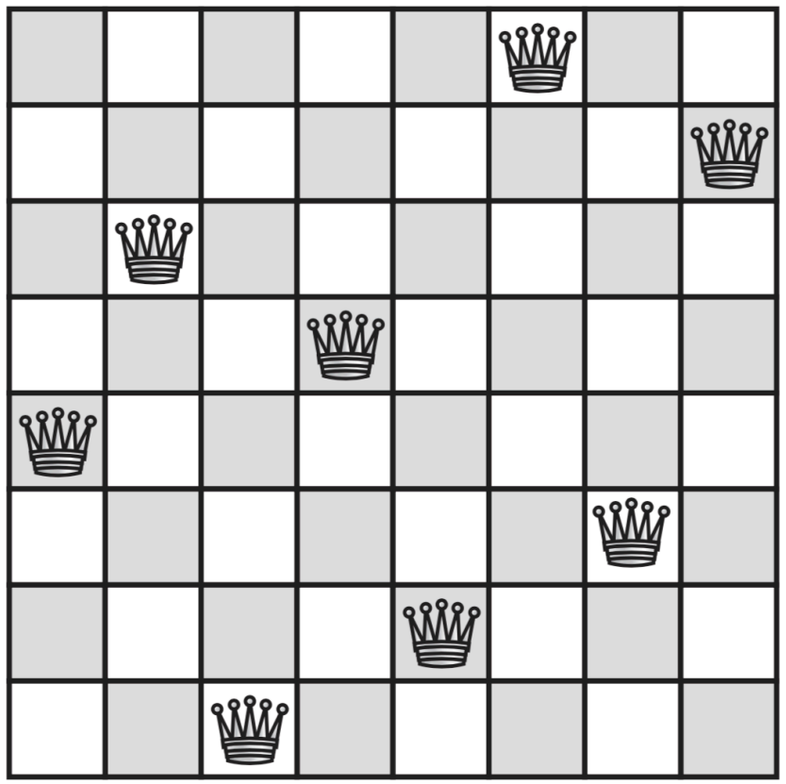
扱う問題

- 8 Queens Problem
- 8 Puzzle
- 15 Puzzle

8 Queens Problem

- チェス盤　８×８
- ８つのQueenをチェス盤に配置
- ８つのQueenがそれぞれぶつからないように配置
- 数学で解析的に説く方法がない





d-hacks
JIN NAKAZAWA LAB

8 Queens Problem（入力・出力）

入力

Queenの数（すでに配置）

Queenの位置

Queenの位置

2

2 2

5 3

出力

```
.....Q..
Q.......
..Q.....
.......Q
.....Q..
...Q....
.Q......
....Q...
```

Queenが配置された場所：’Q’　それ以外：’.’

d-hacks
JIN NAKAZAWA LAB

8 Queens Problem

解説　総当たりで考えた場合

- 8×8=64
- 任意の位置にQueenを1個配置
- 残りは63個

64×63×62×61×60×59×58×57 = 178,462,987,637,760パターン

d-hacks
JIN NAKAZAWA LAB

8 Queens Problem

解説　総当たりで考えた場合

- 8×8=64
- 任意の位置にQueenを1個配置
- 残りは63個

64×63×62×61×60×59×58×57 = 178,462,987,637,760パターン

総当たりは無理

d-hacks
JIN NAKAZAWA LAB

8 Queens Problem

解説　Queenの動きを考慮

・８方向に移動

・各行に１つ

・各列に１つ

64×63×62×61×60×59×58×57 = 178,462,987,637,760 パターン

↓

8! = 8×7×6×5×4×3×2×1=40,320 パターン

d-hacks
JIN NAKAZAWA LAB

8 Queens Problem

解説　Queenの動きを考慮

・８方向に移動

・各行に１つ

・各列に１つ

64×63×62×61×60×59×58×57 = 178,462,987,637,760 パターン

↓

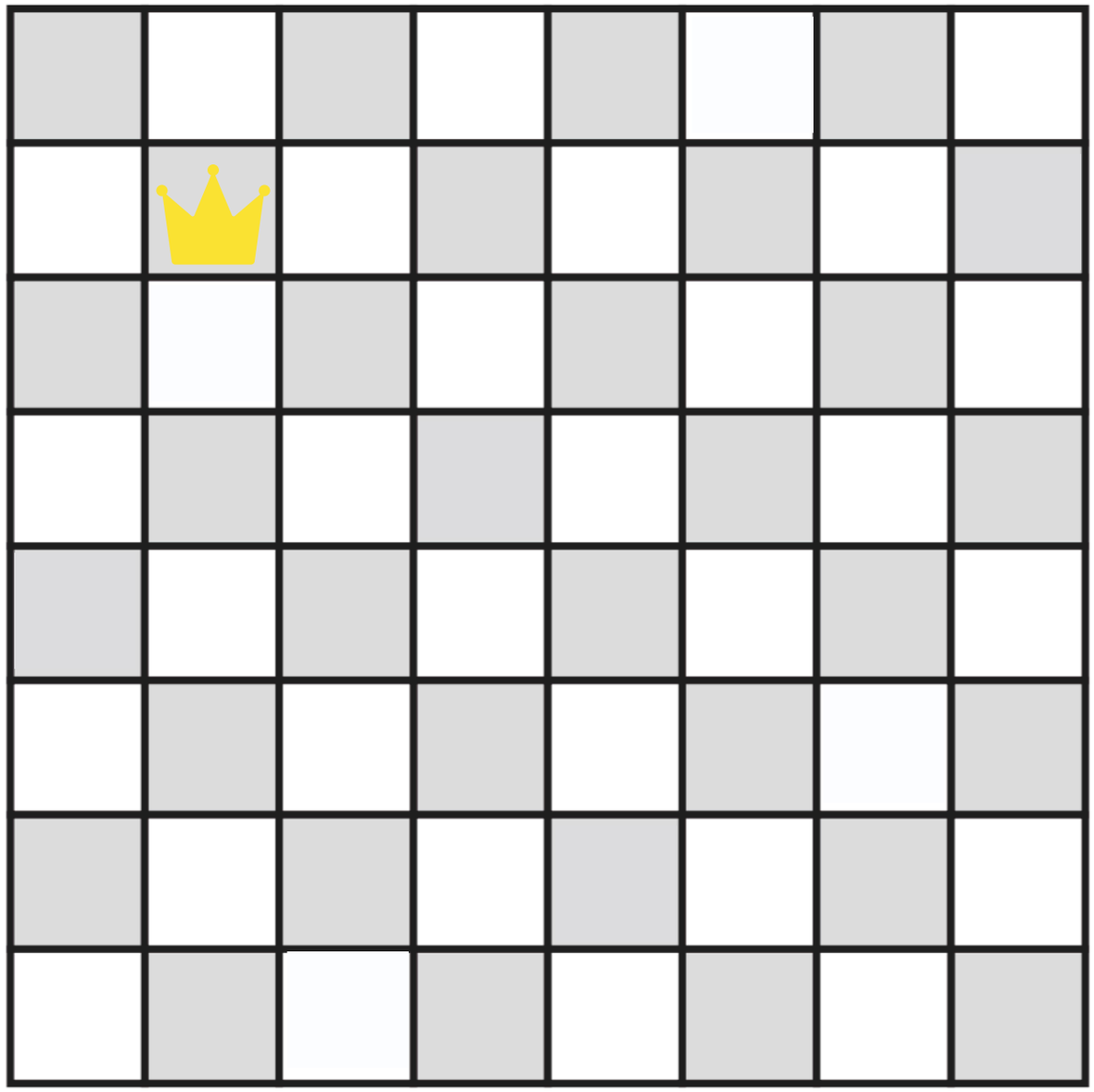
8! = 8×7×6×5×4×3×2×1=40,320 パターン

バックトラックを適用し効率的に問題を解く

d-hacks
JIN NAKAZAWA LAB

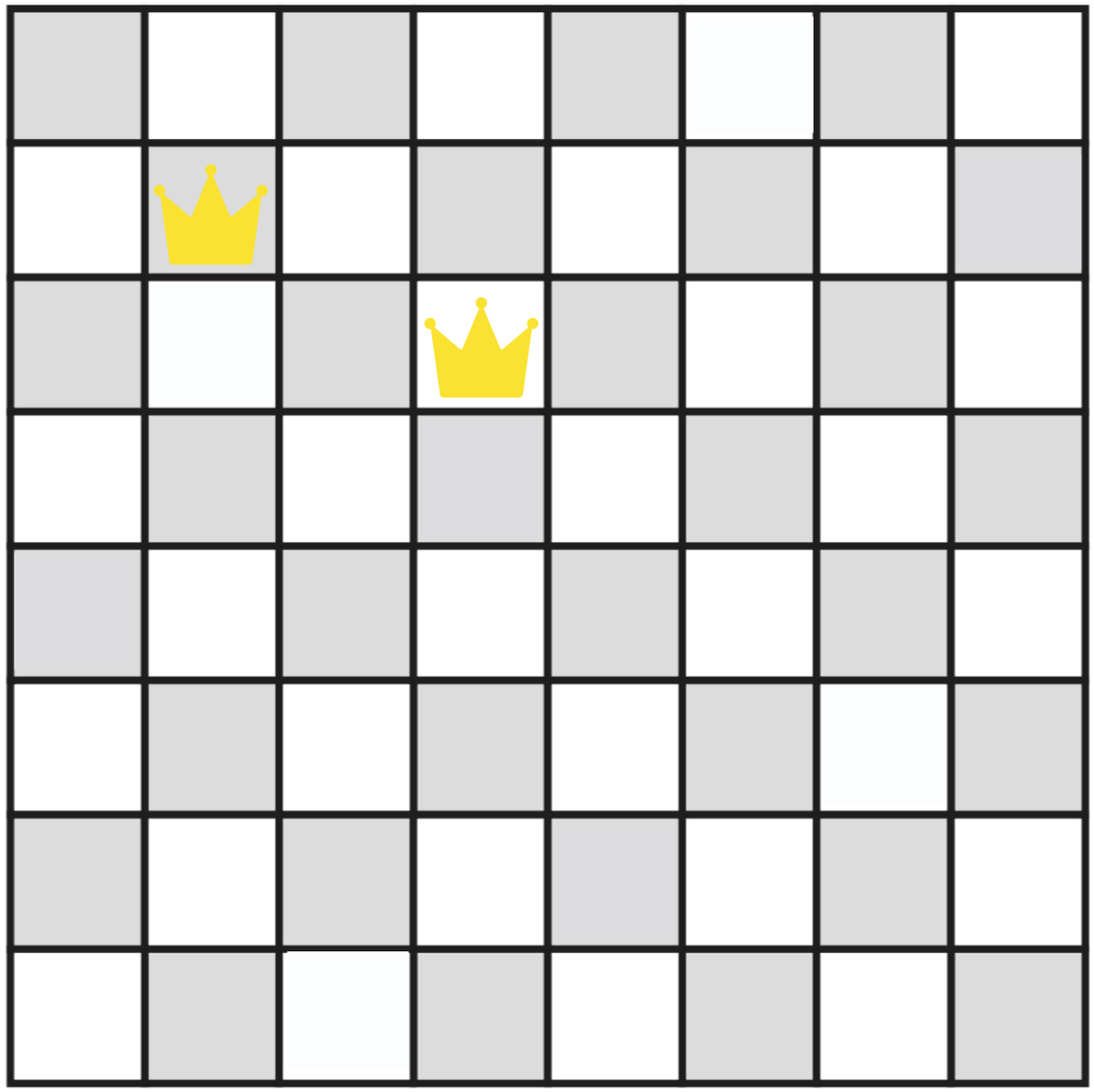
8 Queens Problem

バックトラックとは



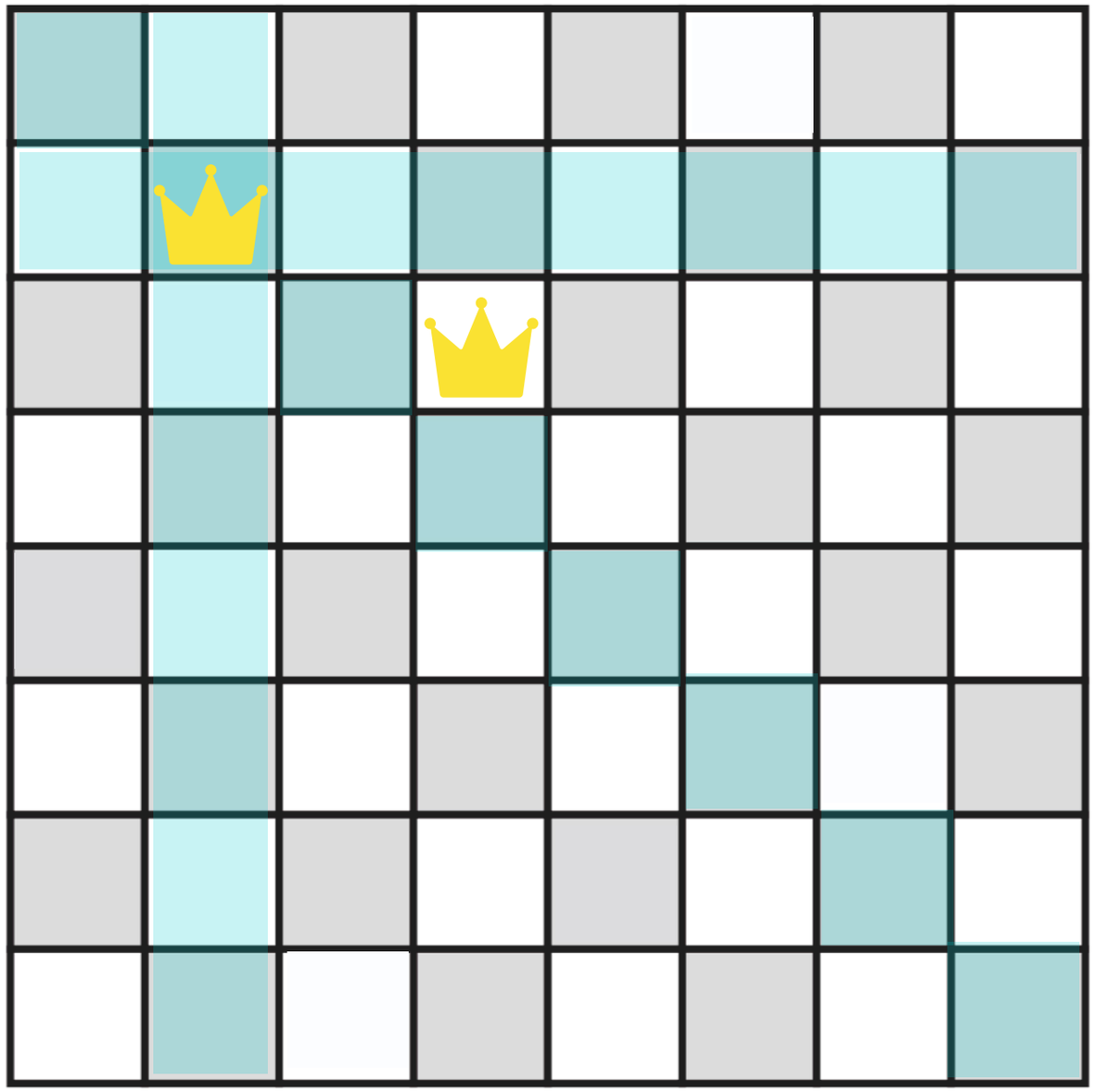
8 Queens Problem

バックトラックとは



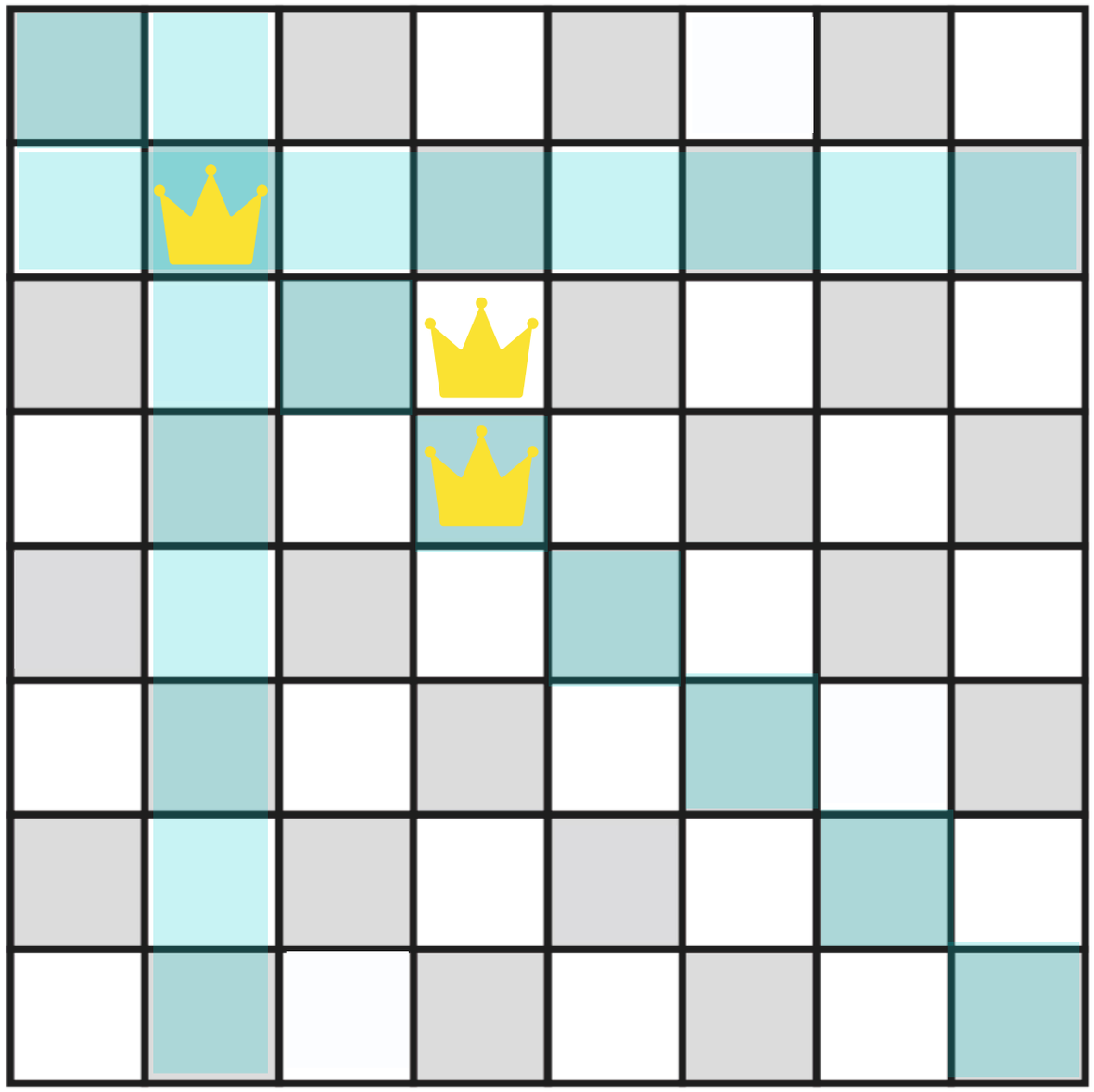
8 Queens Problem

バックトラックとは



8 Queens Problem

バックトラックとは



8 Queens Problem

バックトラックとは

・可能性のある状態を体系的に試していく

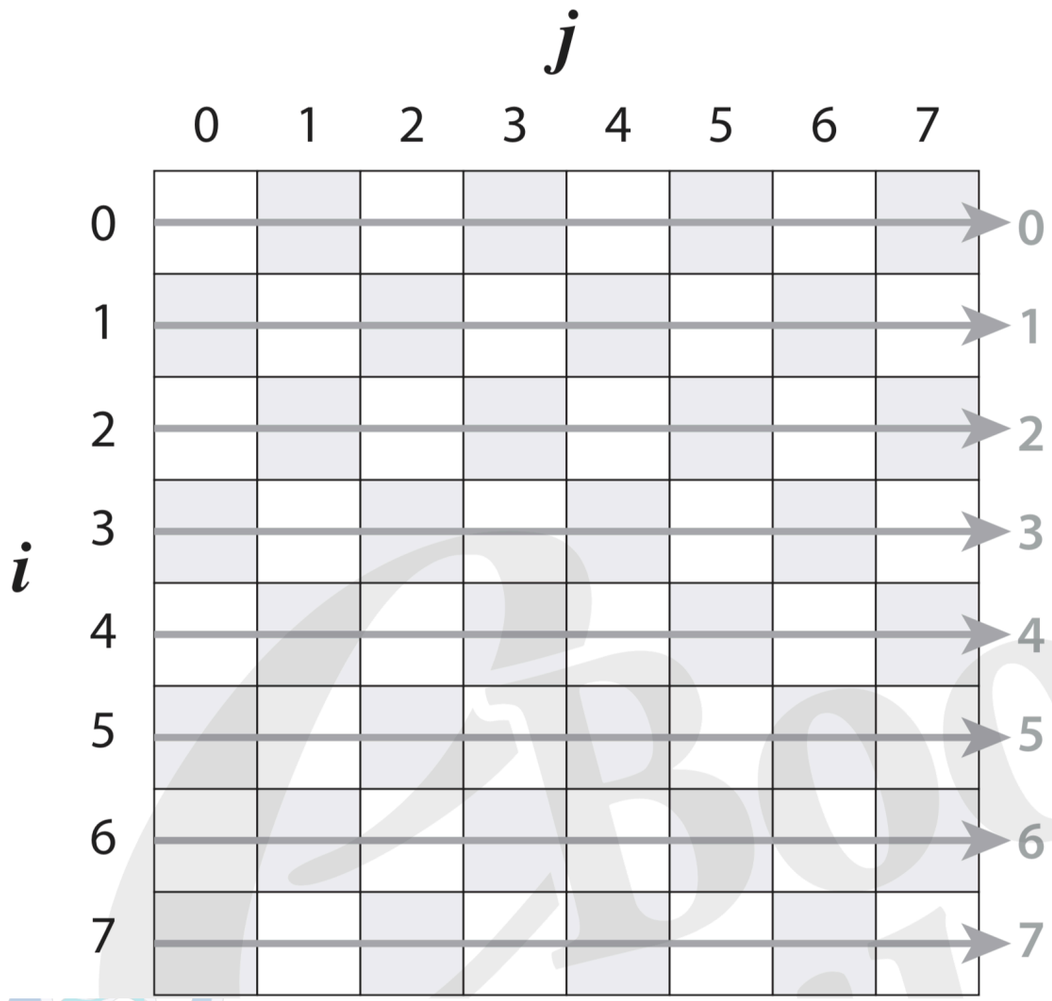
・現在の状態から解は得られない場合，探索を打ち切る

・一つ前の状態に戻って(途中から)探索を再開

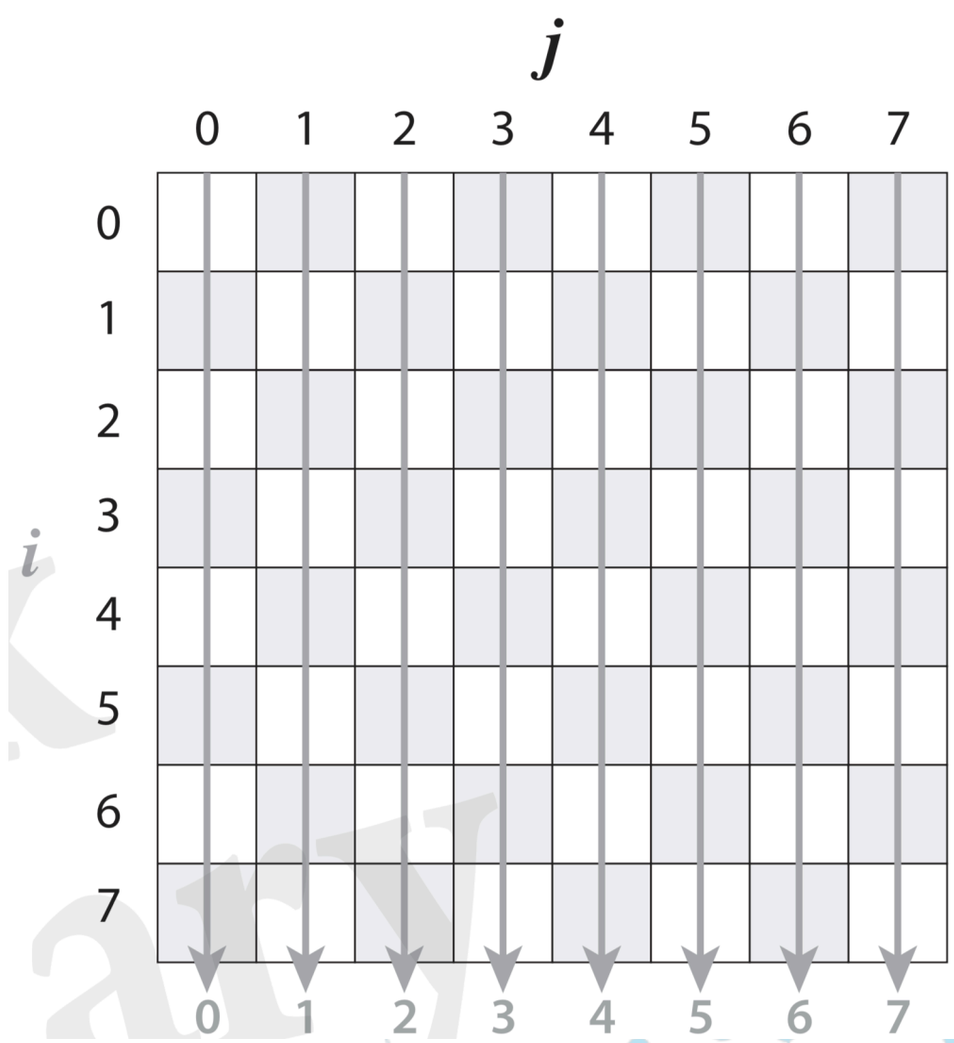
8 Queens Problem

実装

・他のクイーンによって襲撃されているかされていないかを判定



x = i

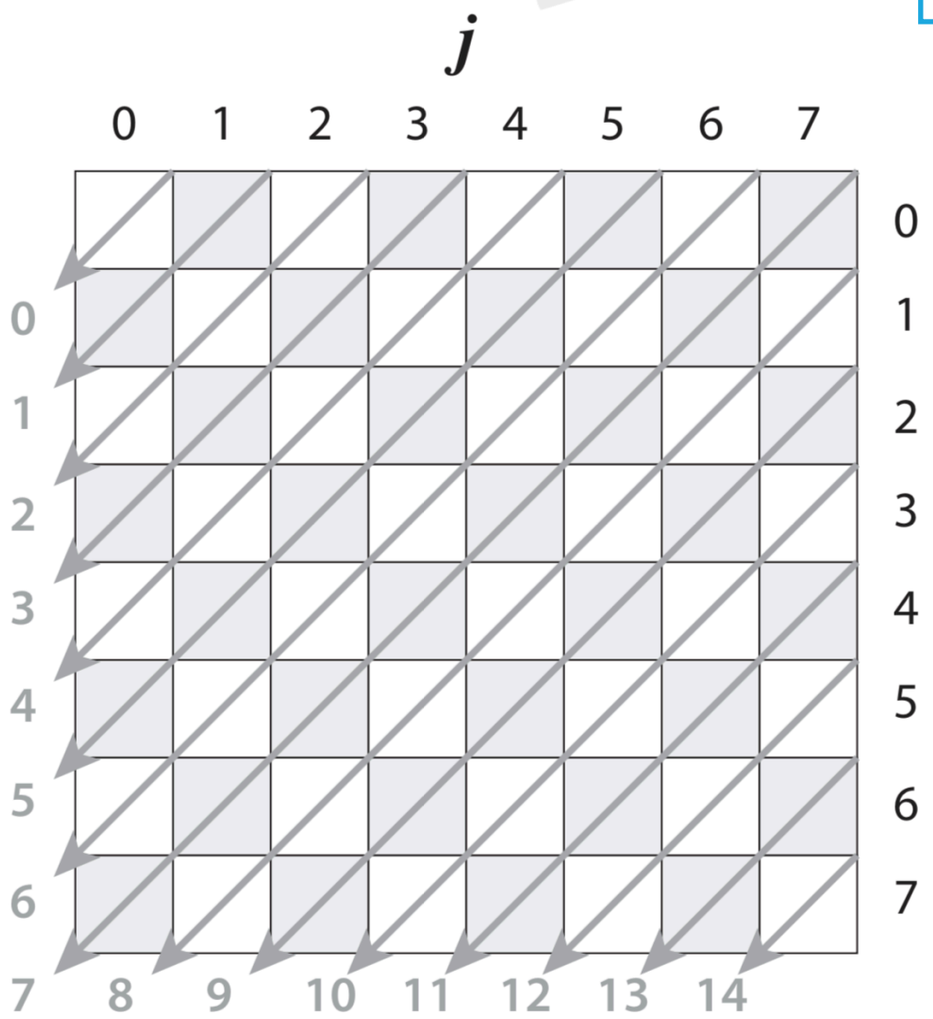


x = j

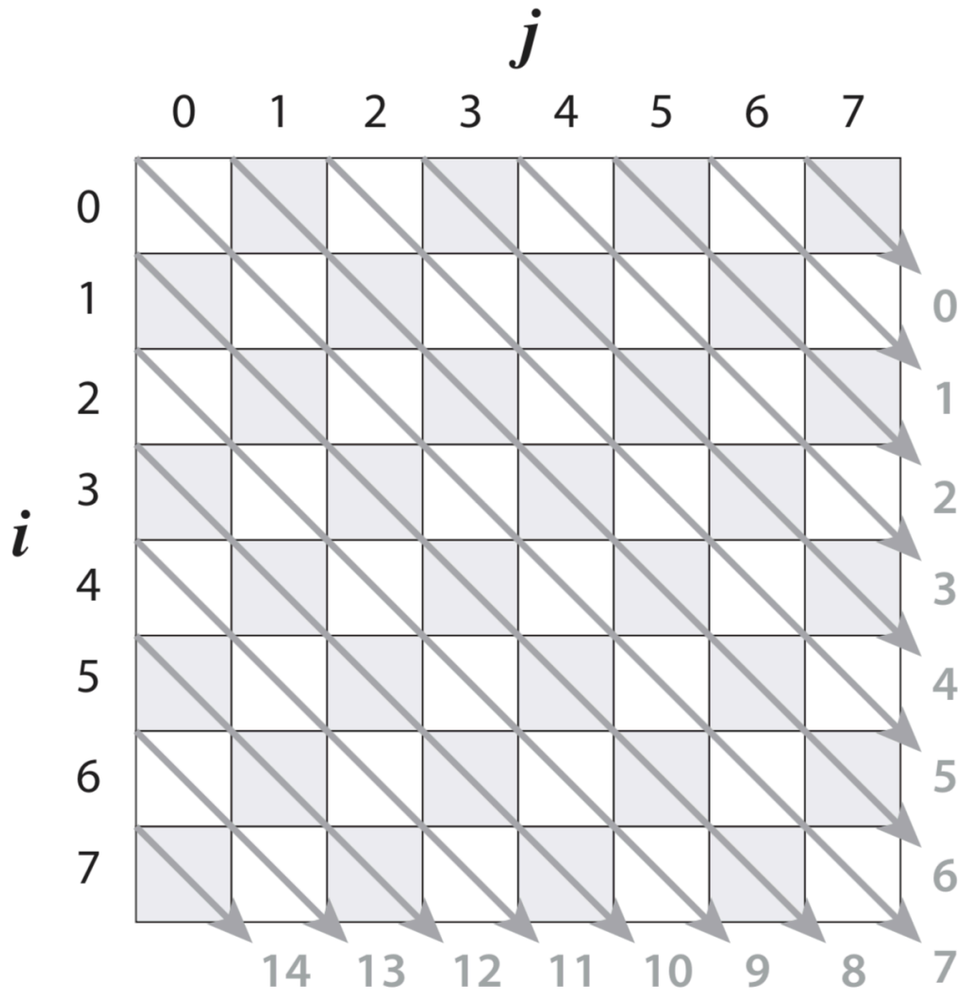
d-hacks
JIN NAKAZAWA LAB

8 Queens Problem

実装




```
x = i + j
```

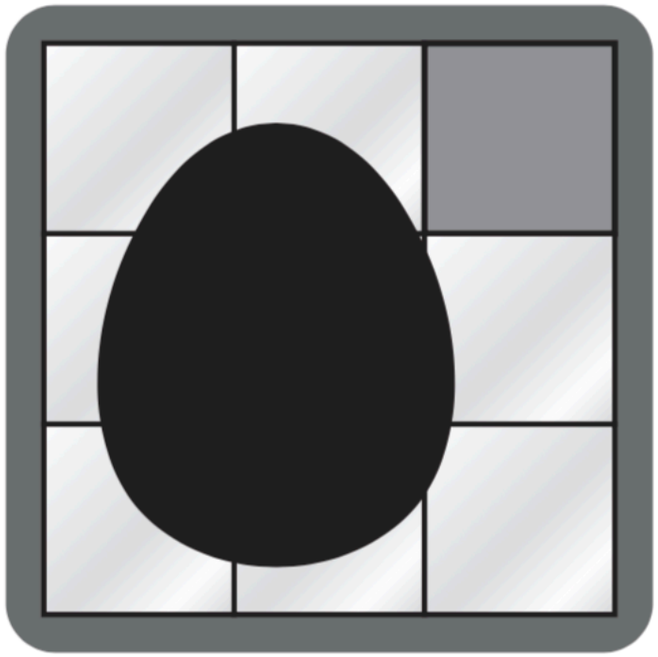



```
x = i - j + (N - 1)
```

d-hacks
JIN NAKAZAWA LAB

8 Puzzle

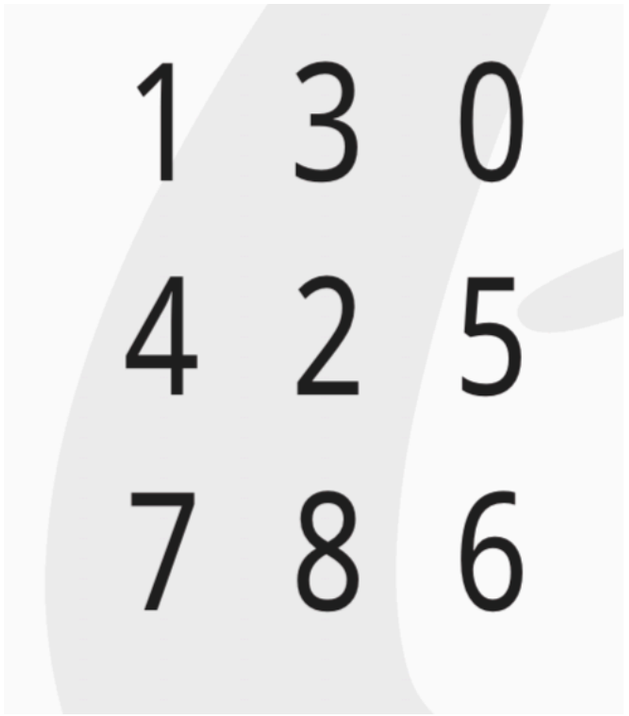
- １つの隙間を利用してパネルを上下左右に移動させ絵柄を揃える



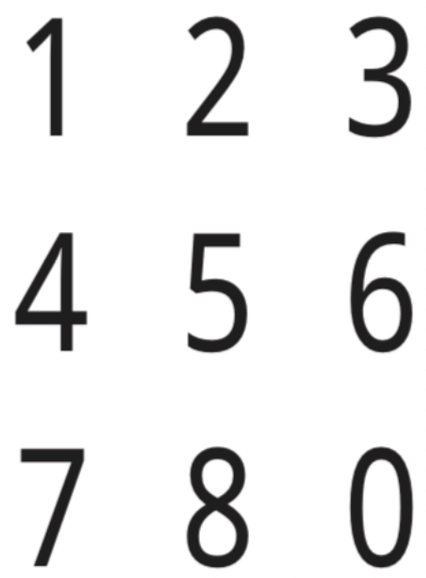
d-hacks
JIN NAKAZAWA LAB

8 Puzzle

パズルを　「空白：０」「各パネル：１～８」で表す



→



d-hacks
JIN NAKAZAWA LAB

8 Puzzle

パズルを　「空白：０」「各パネル：１～８」で表す

入力：３×３の整数

```
1 3 0
4 2 5
7 8 6
```

出力：最短手数

```
4
```

8 Puzzle

パズルを　「空白：０」「各パネル：１～８」で表す

入力：３×３の整数

```
1 3 0
4 2 5
7 8 6
```

出力：最短手数

```
4
```

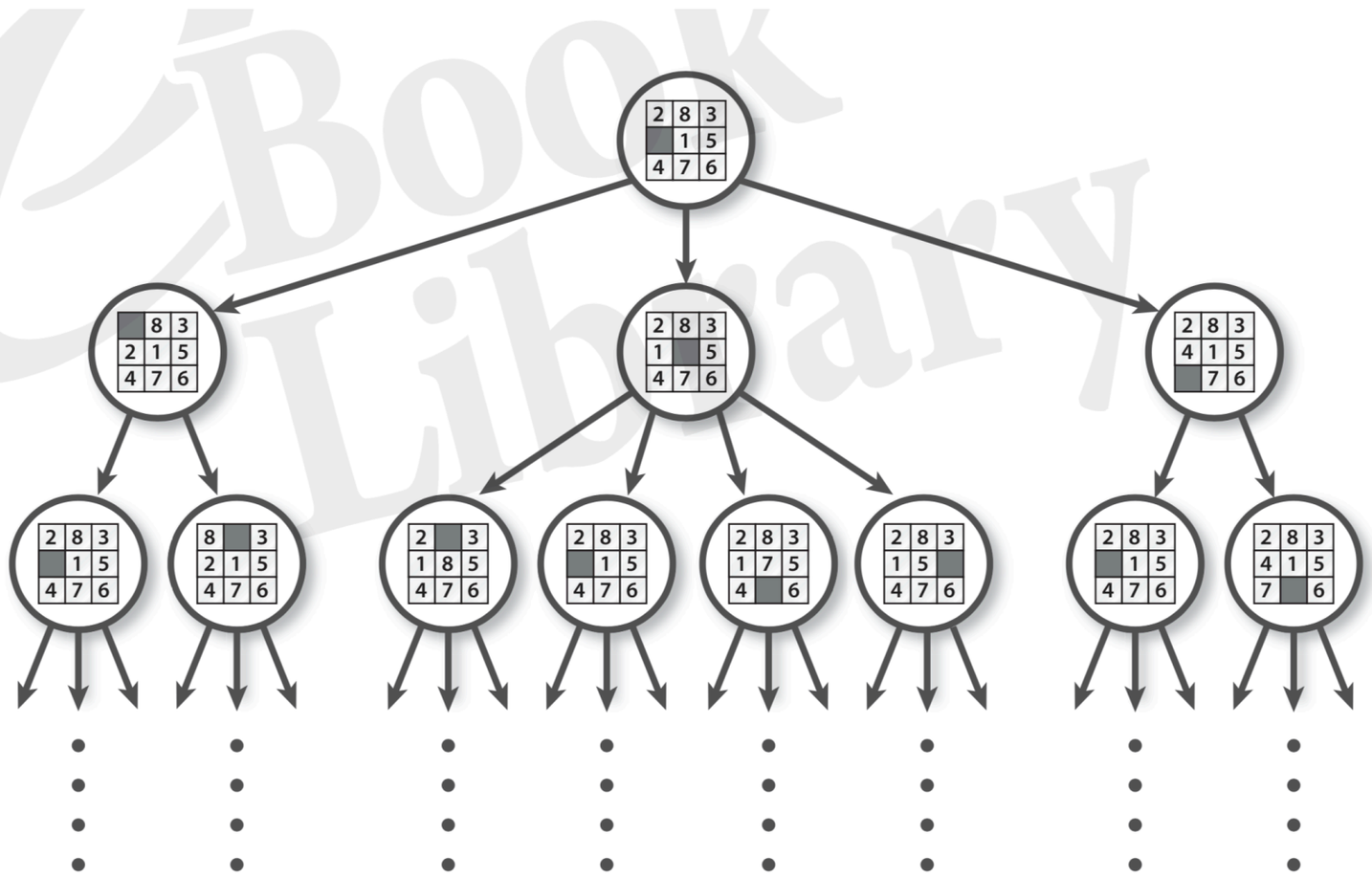
8 Puzzle

- 探索アルゴリズムを使用
- 初期状態から最終状態までの状態変化の列を生成
- 列の中で最も短いものを探す
- 考えられる状態の総数が多くないい場合には

深さ優先探索 / 幅優先探索で解を求める

8 Puzzle

- 一度生成した状態を再度生成しないために,
  探索の空間は木構造
- ノード：状態　エッジ：状態遷移



d-hacks
JIN NAKAZAWA LAB

8 Puzzle

深さ優先探索

・初期状態から最終状態まで可能な限り状態繊維を繰り返す

・以下の場合，探索を断ち切り前の状態に戻る（バックトラック）

▷現在の状態からそれ以上状態遷移が不可能な場合

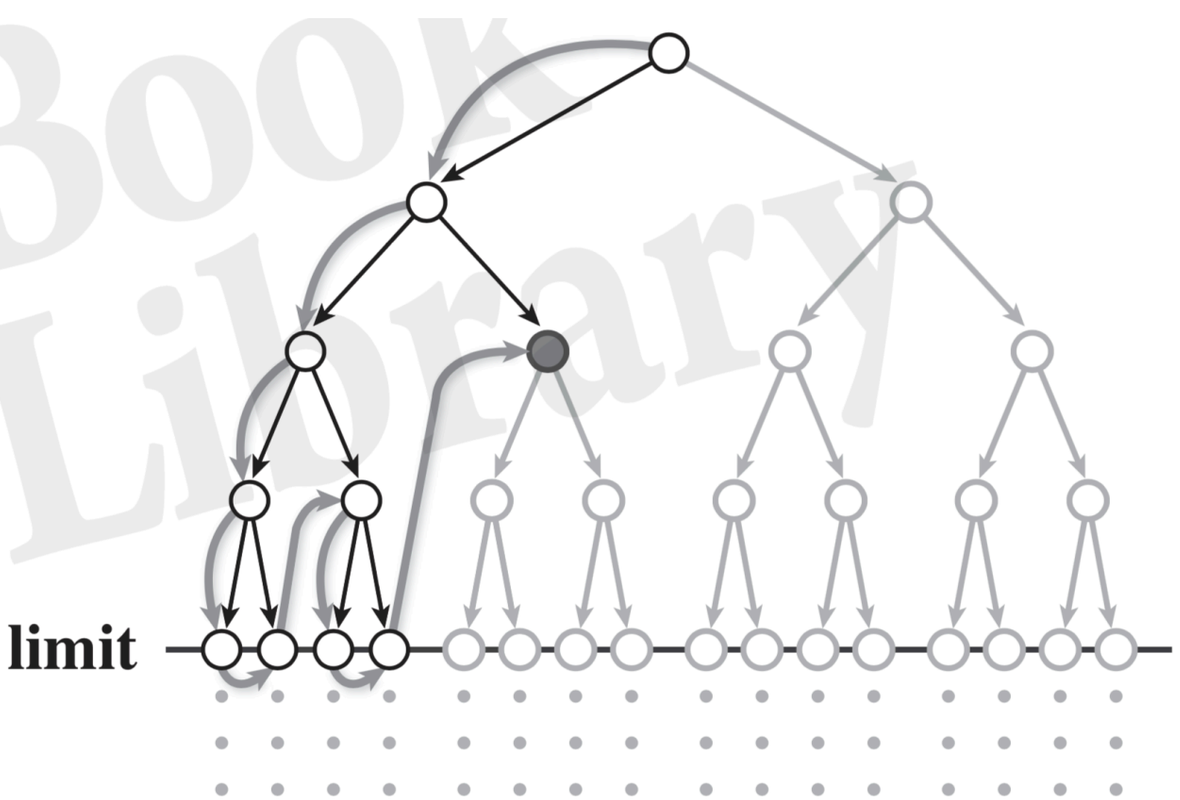
▷状態遷移が一度生成した状態を生成してしまった場合

▷問題の性質からこれ以上探索しても明らかに無駄があると断定できる場合

8 Puzzle

・深さに制限を持たせ，定められた値に達したら探索を打ち切る

→探索を高速化

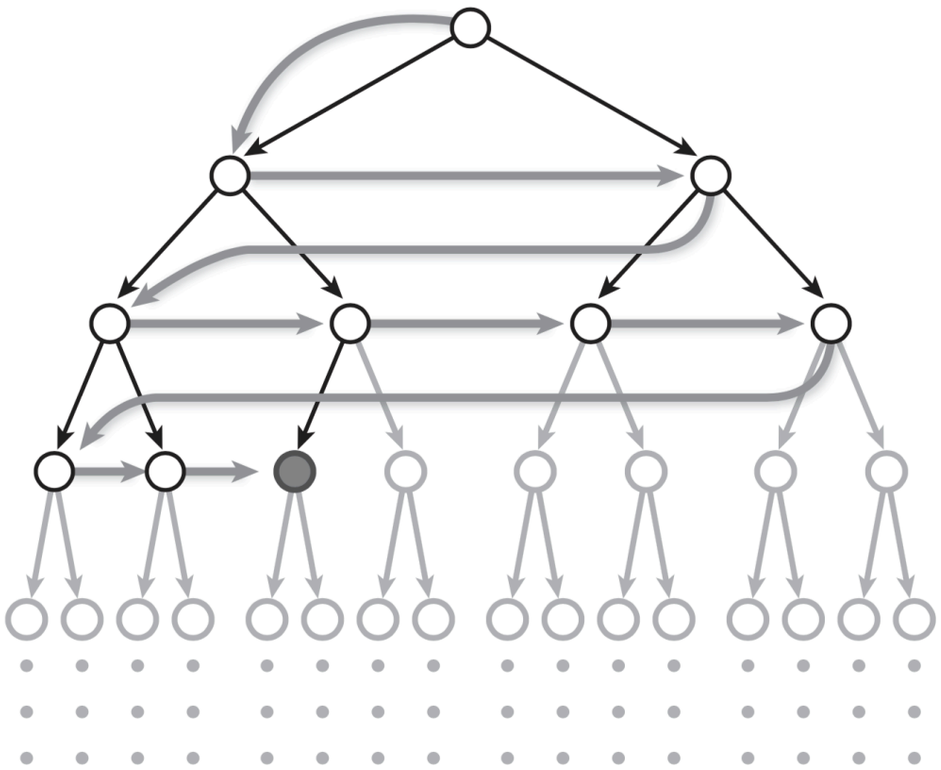


Book Library

8 Puzzle

幅優先探索

- 現在の状態から全ての状態遷移を作る
- 遷移状態で作られた新しい状態をキューに追加
- 探索を展開するためにキューを先頭の状態からさらに状態遷移を繰り返す
- 一度生成した状態を再び作らないよう，状態をメモリに記録



→幅優先探索はメモリを多く必要とするが
問題の解が存在すれば，最短の経路を
比較的早く求めることが可能