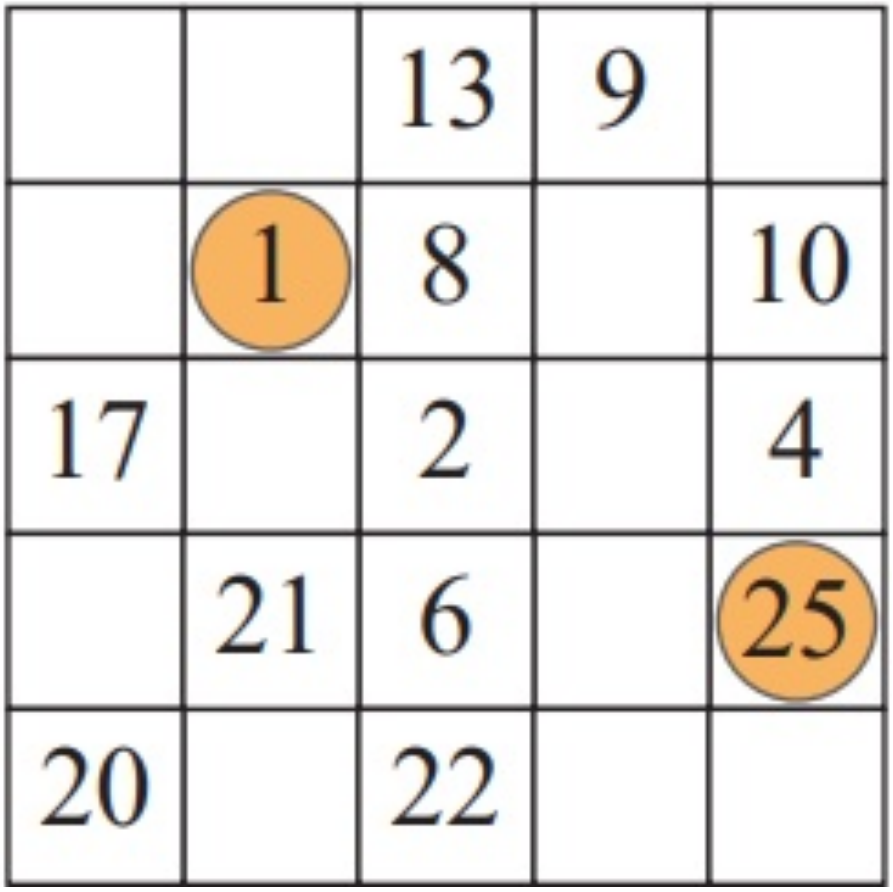
HIDATO PUZZLE COM BACKTRACKING

Alunos: Wagner Silva e Wagner Lucena
Professor: Leonardo Nogueira
Disciplina: PAA


13
9
1
8
10
17
2
4
21
6
25
20
22

O QUE É O HIDATO?

- Puzzle em grade (NxN ou MxN)
- Preencher números consecutivos (1 até K).
- Cada número deve ser adjacente ao anterior e ao próximo.
- Adjacência em 8 direções
- Algumas células vem pré-preenchidas.

MODELO FORMAL DO PROBLEMA HIDATO

- Preencher sequência completa
- Cada número em célula distinta
- Movimentos permitidos: 8 adjacências
- Números fixos restringem a busca

(DUARTE; CAMPOS; COSTA, 2020; DUARTE; ALMEIDA; CAMPOS, 2022)

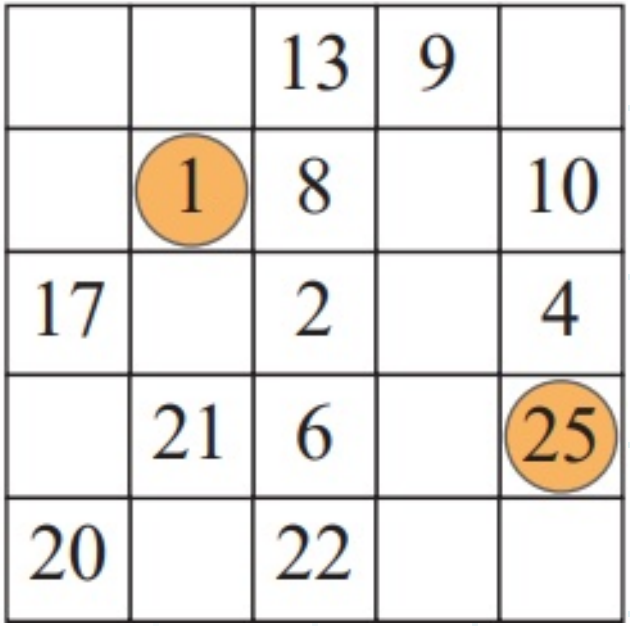
ENTRADA DO HIDATO

- Tamanho de grade.
- Células preenchidas previamente
- Células vazias
- Objetivo: completar 1…K

(DUARTE; CAMPOS; COSTA, 2020; DUARTE; ALMEIDA; CAMPOS, 2022)

EXEMPLO DE INSTÂNCIA

- **Objetivo:** conectar todos os números consecutivos via adjacência.



POR QUE HIDATO ESTÁ EM NP

Hidato $\in$ NP

- Fácil verificar uma solução completa
- Checagem polinomial
  - números consecutivos
  - adjacências corretas
  - respeita posições fixas

POR QUE É NP-COMPLETO

Hidato é NP-Completo

- Redução do Hamiltonian Path
- Problema equivalente: encontrar caminho Hamiltoniano na grade

POR QUE USAR BACKTRACKING?

- Permite explorar todas as possibilidades
- Verifica caminho número por número
- Retorna quando uma escolha é inválida
- Combina com a natureza NP-completa do problema

Solução

```python
if value in self.grid.positions:
    return self._backtrack(value + 1)
```

2	3		5
3	1		6
11			

2	3		5
~~3~~	1		6
11			

```python
future = self._next_fixed_after(value)
if future is not None:
    future_value, future_pos = future
    steps_available = future_value - value
    candidates = [
        cell for cell in candidates if chebyshev_distance(cell, future_pos) <= steps_available
    ]

return sorted(candidates, key=self._neighboring_free_cells)
```

Solução

2	3	4	5
	1	4	6
11			

2	3	X	5
	1	4	6
11			

```
for cell in self._candidate_cells(value):
    self.grid.set_value(cell, value)

    if self._backtrack(value + 1):
        return True

    self.grid.clear_value(value)
```

Solução

2	3	4	5
	1		6
11			

2	3	4	5
	1	7	6
		7	7
11			

Solução

2	3	4	5
13	1	7	6
12	14	9	8
11	10	15	16