Hidato Solving
Noam Gianella, Noa Finkelshtein
Advisors: Prof. Daniel Berend

The problem:

Original Hidato: In Hidato, a grid of \( n \) cells is given, some cells are labeled with a natural number from 1 to \( n \). The goal is to find the label of each cell, so that there's a path of adjacent from label 1 to label \( n \).

Example:

<table>
<thead>
<tr>
<th></th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>8</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Our project consider the following generalization of Hidato: Given is a graph on \( n \) vertices, where some of the vertices are labeled by distinct numbers between 1 and \( n \). The goal is to find a Hamiltonian path which passes through the labeled vertices at the times (labels) showed at those vertices.

Graph Generators:

Adding edges is done by two methods:
- \( G(n,p) \) Generator: Every edge is added with probability \( p \), received as input.
- Path plus Generator: First an Hamilton path is added to the graph, guaranteeing at least one solution. Then \( c \times n \) edges are added randomly, when \( c \) is constant input.

Solvers:

Naive Solver: split to intervals according to labels \( \Rightarrow \) find all possible solutions for each interval \( \Rightarrow \) combine the intervals’ solutions.

Matching Solver: search for a certainly match according to the game rules, until none can be deduced \( \Rightarrow \) guess a match and try to find a solution.

SAT Reduction Solver: transform graph instance to CNF expression \( \Rightarrow \) solve with miniSAT solver \( \Rightarrow \) transform the output to a solution (if such exist).

Statistical analysis:

For graphs that were generated with the \( G(n,p) \) generator we analyzed the influence of the probability, \( p \), on the number of solutions.

For graphs that were generated with the Path plus generator we analyzed the influence of the constant, \( c \), on the number of solutions.