Assignment 1

Hamiltonian paths meet SAT

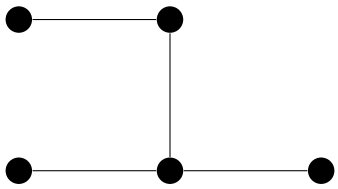
Static Program Analysis and Constraint Solving
Master's Degree in Formal Methods in Computer Science
Year 2021/22

Submission deadline: October, 1st

Submission instructions: Students are required to submit the source code of the program and (if applicable) the generated SMT-LIB script. The program has to contain comments explaining how the problem is formalized in propositional logic, and which constraints have been generated. Students might be required to present and defend their work after submission.

Given a graph G, a *Hamiltonian path* is a path in G that visits every vertex exactly once. For example, the following graph shown on the left-hand side has several Hamiltonian paths, one of them highlighted in the figure on the right-hand side:

However, there are some graphs in which no Hamiltonian path exists. For example:



The problem of finding Hamiltonian paths in a graph can be reduced to SAT. Firstly, let us assume that the set of vertices in the graph is $V = \{v_1, \ldots, v_n\}$. Therefore, a Hamiltonian path is a sequence of exactly n vertices. Assume a set $\{p_{ij} \mid 0 \leq i \leq n, 0 \leq j \leq n\}$ of propositional variables, so that p_{ij} is true if and only if the vertex v_i appears in the j-th position of the sequence.

Given this encoding, our solution has to meet the following constraints:

1. For each $i \in \{1..n\}$, v_i appears in the sequence.
2. For each $i \in \{1..n\}$, v_i does not appear in two different positions of the sequence.
3. For each $j \in \{1..n\}$, the j-th element of the sequence contains a vertex.

4. For each $i, j \in \{1..n\}$ such that $i \neq j$, v_i and v_j do not appear in the same position of the sequence.

5. For each $i, j \in \{1..n\}$ such that $i \neq j$: if v_i and v_j are not adjacent in the graph, then they do not appear together in the sequence.

Formalize these restrictions as sentences in propositional logic. These sentences should mention only the p_{ij} variables, and cannot contain quantifiers. Then, in your favorite programming language, implement a program that receives a graph as input and determines whether it contains a Hamiltonian path. You can use one of these approaches:

- Your program generates an SMT-LIB script that is subsequently executed by Z3, **or**
- Your program uses language-specific bindings to Z3 API. Currently supported languages are C, C++, Java, Python, ML/OCaml, and those targeting the .NET platform.