CSE 260 Fall 2022
Programming Project

Dr. Borzoo Bonakdarpour

In this assignment, you are to solve the n-Queens problem by reducing the problem to the propositional satisfiability problem. Recall that a propositional formula φ is satisfiable if there is an assignment of truth values to propositional variables in φ that makes φ true.

1 Description

The *n-queens problem* asks for placement of n queens on an $n \times n$ chessboard so that no queen can attack another queen. One can encode the n-Queens problem as a satisfiability problem as follows (more details can be found in your textbook and on lecture slides):

- We introduce n^2 propositions. Let them be $p(i,j)$ for $i = 1, 2, \ldots, n$ and $j = 1, 2, \ldots, n$, which indicates whether there is a queen in row i and column j.
- There has to be *at least* one queen in each *row*:

$$Q_1 = \bigwedge_{i=1}^{n} \bigvee_{j=1}^{n} p(i,j)$$

- There has to be *at most* one queen in each *row*:

$$Q_2 = \bigwedge_{i=1}^{n} \bigwedge_{j=1}^{n-1} \bigwedge_{k=j+1}^{n} \Big(\neg p(i,j) \vee \neg p(i,k)\Big)$$

- No *column* contains *more than one* queens:

$$Q_3 = \bigwedge_{j=1}^{n} \bigwedge_{i=1}^{n-1} \bigwedge_{k=i+1}^{n} \Big(\neg p(i,j) \vee \neg p(k,j)\Big)$$

- No *diagonal* has two queens:

$$Q_4 = \bigwedge_{i=2}^{n} \bigwedge_{j=1}^{n-1} \bigwedge_{k=1}^{\min(i-1,n-j)} \Big(\neg p(i,j) \vee \neg p(i-k,k+j\Big)$$

and

$$Q_5 = \bigwedge_{i=1}^{n-1} \bigwedge_{j=1}^{n-1} \bigwedge_{k=1}^{\min(n-i,n-j)} \Big(\neg p(i,j) \vee \neg p(i+k, k+j)\Big)$$

- Putting all these together:

$$Q = Q_1 \wedge Q_2 \wedge Q_3 \wedge Q_4 \wedge Q_5$$

- Thus, if Q is satisfiable, then the n-queens problem has a solution given by $p(i,j)$ for $i = 1, 2, \ldots, n$ and $j = 1, 2, \ldots, n$.

2 Assignment

You are to solve the problem for $n = 3$ and $n = 4$ using the python API of the SMT-solver Z3. Z3 is a tool that can solve the proposition satisfiability problem[1]. The tool allows declaring propositional variables and including propositional formulas for checking their satifiability. If Z3 returns `unsat`, it means the input formula is not satisfiable. Otherwise, it returns `sat` with "a model", which the truth assignments. A tutorial video is available in D2L (Programming Project Unit). Here's also the repository for that: `https://github.com/oyendrila-dobe/IntroToZ3`.

You will develop two Z3 files one for $n = 3$ and one for $n = 4$. If your Z3 submission has syntax error, you will not receive any credit. You will receive partial credit if you make a meaningful attempt at the problem.

You are required to add comments to indicate formulas $Q, Q_1, \ldots, Q_5$. Name your propositional variables as `pij`, which represents $p(i,j)$, as described above. For example, `p23`, represents $p(2,3)$. In case of `sat`, the TAs will check if the truth assignments to each $p(i,j)$ is indeed a valid solution to the problem.

A tutorial is av https://github.com/oyendrila-dobe/IntroToZ3

3 Extra Credit

You will receive 100% extra credit if you write a program in `python` that solves the problem for any input value n. To this end, you will have to use the Z3 API to write a program that (1) receives n as input, (2) generates the corresponding propositional formulas, and (3) invokes the Z3 engine to determine whether the generated formula is satisfiable.

Deliverable

Your solutions must be submitted by **11:59pm on Friday, December 2,** via D2L. The first two lines of your code should include the name of both students in the group. Z3 does have a function to directly solve the n-Queens problem. You are *not* allowed to use that: you should implement the formulas in Section 1.

[1] `https://www.microsoft.com/en-us/research/project/z3-3/`