Technical Report AC-TR-18-013

July 2018

QBF Encodings of Chess Problems

Tomáš Peitl

This is the author’s copy of a paper that was presented at the *QBF Workshop 2018*, see `https://fmv.jku.at/qbf18/program.html`

QBF Encodings of Chess Problems

Tomáš Peitl

July 2018
QBF Workshop@FLoC

1 Introduction

Chess has long held a prominent position as a benchmarking application of AI—whenever computers gained some significant ground against humans, or even against other computers lately, news of it quickly spread all around the world. Since a QBF can also be seen as a game between the universal and the existential player, it seems natural to apply it to chess. However due to the unbounded nature of chess (practically unbounded, there are rules that in fact prevent games of unbounded length, but games can still be very long), simply playing chess does not seem to be a good fit for QBF. Instead, QBF can very well be used to model *chess problems*.

Chess problems are special positions with a designated challenge, typically for White, such as to deliver a mate in a given number of moves. Not all chess problems have a bound on the number of moves in which the challenge must be fulfilled, but most do, and we will focus exactly on such ones. More precisely, we will work with *directmates*, i.e. problems in which White has to deliver a mate in n moves (to every response of Black in every move), or $\#n$ for short.

There are two different things that can be done with chess problems. Given a chess problem, the question of course is whether it has the required solution, and also what the solution is. However, even more interestingly, one can ask whether given some constraints (such as up to 4 white pawns, or fixed position of the black king) say a #2 that satisfies those constraints exists. The latter is known as the *composing* of chess problems, and while there is good software to *solve* existing chess problems, computer-aided composition of problems is still very much in its infancy. It would therefore be particularly interesting if QBF solvers could compose nice problems. Trivial problems can usually be created easily, but the real challenge comes when we impose constraints on the aesthetics of the solution, such as uniqueness of White's moves. Our composing instances incorporate some of the more challenging constraints that, even though they do not guarantee perfect compositions, increase the quality of the results significantly, and also make the resulting QBFs very tough to solve.

2 The Instances

The submitted instances are all intended for the prenex non-CNF track, they are in the QCIR format, and there is a total of 185 of them. They can be downloaded from `https://www.ac.tuwien.ac.at/files/resources/instances/chess_qbf/chess_instances.tar`, and fall in two categories.

The first category are encodings of a random sample of real human-composed directmates that have been used in chess-solving competitions in the past decades. These range from #2 to #10, there are 20 problems of each of #2 to #7, four #8, two #9, and finally one #10, a total of 127 instances. All of these formulas should be true, and the difficulty should increase with the number of moves, although particularly in the higher numbers, it can happen that a longer problem is easier than a shorter one. Each file name consists of a prefix that indicates that this is a chess problem encoded for solving (as opposed to composing), the number of moves in which mate is to be delivered (always by White by convention), and a string that identifies in which competition and which round the instance appeared. If say 20 instances were to be used, we would recommend to use five of each #2 and #3, two of each #4 to #6, and one of each of the remaining ones. Empirically, #2 seem to be generally easy, #3 rather easier than harder, #4 rather harder, but still solvable in some cases, #5+ hard, but we of course look forward to learning about this from the competition, as well as to seeing some interesting surprises.

The second category are encodings of some hand-crafted composing templates for #2. These templates impose hard constraints on the problem being composed such as fixing the position of some pieces, or limiting the number of pieces available. In addition to that, all composing instances enforce uniqueness and some other aesthetic properties of White's first move. As a result, these instances contain roughly twice the number of variables, quantifier alternations, and constraints, and are much harder than solving #2. Because these instances turned out so hard in our experiments, we also introduced a simpler version, namely composing on a chessboard of size 6×6. The submitted instances consist of 50 classical composing templates (i.e. 8×8 board), and 8 ones on a 6×6 board. All of these instances should be true as well, although it is possible that a false one has crept in by accident. Each file name consists of a prefix that indicates that this is a chess problem encoded for composing, the dimension of the board (6 or 8), and the number of the composing template. If 30 instances were to be picked in total, we would suggest picking 20 solving instances as described in the previous paragraph, seven 8×8 composing instances, and three 6×6 composing instances.