ETH zürich

Alexander J. Summers

Program Verification

Exercise Solutions 2: Encoding Problems to SAT

Assignment 1 (Eliminating Equality)

The variation of Ackermannization shown in the lectures to eliminate equality potentially generates many variables and extra conjuncts to express instances of the reflexivity, symmetry and transitivity properties of equality.

1. We generate n^2 new boolean variables; one for each pair of original term variables.
2. The number of possible instances is n instances of reflexivity, n^2 of symmetry and n^3 of transitivity.
3. Here are a few ideas; if you have others, I'd be interested to hear them.
   - We don't need boolean variables to represent reflexive cases (equating a term with itself); we could just translate such equalities to $\top$. In this way, we also avoid the need for adding instances of the reflexivity property entirely.
   - For transitivity, we can avoid instances which only differ by symmetry with one of the three axioms already chosen (however, this idea is subsumed by the next one).
   - We can "build in" symmetry properties as part of our Ackermannization process in the following way: choose any total ordering $<$ on the original term variables in the problem. Now, only introduce boolean variables $eq_{x,y}$ for pairs of variables x and y such that $x < y$. When translating the original formula, translate equalities between x and y to this, regardless of the order of the terms in the equality (i.e. $y = x$ also gets translated to $eq_{x,y}$). Note that the case of reflexive equalities is already dealt with above. In this way, we avoid the need for symmetry axioms entirely.
   - For transitivity, we need to consider triples of term variables x,y and z. But there is no need to consider choices in which any of these three variables are the same (the property is then logically trivial). Similarly (by symmetry), once we instantiate the property for x,y and z, there is no need to instantiate it also for z,y and x. We could implement this by only instantiating for $x < z$ combinations (according to the ordering above).

- We can further optimize transitivity with the following idea: let $\equiv_{eq}$ be the smallest equivalence relation on the original term variables, such that $x \equiv_{eq} y$ if x and y form the two sides of some equality in the original formula. We only need generate transitivity instances for triples of variables all in the same $\equiv_{eq}$-equivalence class. The intuition behind this idea is that some variables might never (even via transitivity) get compared with one another in constructing models for the original formula.

Assignment 2 (Eliminating Uninterpreted Functions)

We work on the inner terms first. Starting from:

$$f(g(x)) = x \wedge f(y) = x \wedge \neg(y = g(x))$$

we generate a new term variable g_x to replace $g(x)$, and a new term variable f_y to replace $f(y)$. Since, so far there are no pairs of newly-introduced terms regarding the same top-level function (just one for f and one for g), we don't need to add any congruence constraints. The resulting formula is:

$$f(g_x) = x \wedge f_y = x \wedge \neg(y = g_x)$$

Now, we generate a new term variable f_{g_x} to replace the function application $f(g_x)$. Since we have already introduced the term variable f_y (for the same function f), we have to add the congruence constraint: $y = g_x \Rightarrow f_y = f_{g_x}$, conjoining this to the new formula. We obtain:

$$f_{g_x} = x \wedge f_y = x \wedge \neg(y = g_x) \wedge (y = g_x \Rightarrow f_y = f_{g_x})$$

This formula is satisfiable (in fact, we can almost "read off" a model from the conjunction of equalities and inequalities): in any model M for which $M(f_{g_x}) = M(x) = M(f_y)$ and where $M(y)$ and $M(g_x)$ are interpreted as two different values. Note that the congruence conjunct is satisfied, since the left-hand-side of the implication is false in such a model.

This also tells us how to build models for the original formula: take any model in which $M(g)$ is a mathematical function which does *not* map $M(x)$ to $M(y)$; i.e. $M(g)(M(x))$ should be some value v in the model different from $M(y)$. Furthermore, the interpretation of f in the model (i.e. $M(f)$) must be a function which maps v to the same value as $M(x)$, and which also maps $M(y)$ to this same value.

Assignment 3 (Encoding the Eight Queens Puzzle)

One way to represent a problem state would be to specify for each place if it is occupied by a rook or not. We can do this by having a variable for each place $o_{r,c}$ ($0 \leq r < 8$ and $0 \leq c < 8$), where a variable $o_{r,c} = \top$ iff the place at the row r and the column c is occupied by a rook. Now we need to encode two constraints: there are 8 rooks on the chessboard and no two rooks threaten each other. One could make the observation that the second constraint already implies that there are at most 8 rooks on the board, because if there were more, at least two of them would threaten each other. Therefore, to ensure the first constraint we only need to make sure that there are at least 8 rooks on the board, which means that in each row and each column there is at least one rook. We can encode this by requiring that at least one variable in each row

and each column is true. For example, the disjunct that expresses that in a row i there is at least one rook would be:

$$o_{r,0} \wedge o_{r,1} \wedge o_{r,2} \wedge o_{r,3} \wedge o_{r,4} \wedge o_{r,5} \wedge o_{r,6} \wedge o_{r,7}$$

Similarly, the disjunct that expresses that in a column j there is at least one rook, would be:

$$o_{0,j} \wedge o_{1,j} \wedge o_{2,j} \wedge o_{3,j} \wedge o_{4,j} \wedge o_{5,j} \wedge o_{6,j} \wedge o_{7,j}$$

Two rooks threaten each other iff they are on the same row or the same column. We can express that no two rooks are on the same row (column) by requiring that each pair of places in that row (column) contains at most one rook. This can be encoded as follows for a specific row i:

$$\bigwedge_{0 \leq j < 8,\ 0 \leq k < j} \neg(o_{i,j} \wedge o_{i,k})$$

This is equivalent to:

$$\bigwedge_{0 \leq j < 8,\ 0 \leq k < j} \neg o_{i,j} \vee \neg o_{i,k}$$

Similarly, the constraint for the column j would be:

$$\bigwedge_{0 \leq i < 8,\ 0 \leq k < i} \neg o_{i,j} \vee \neg o_{k,j}$$

These finitely bounded quantifiers can be expanded to give the actual set of clauses used.

1. The presented encoding uses $8 \times 8 = 64$ variables.

2. We need 2×8 clauses of size 8 to express that there is at least 1 rook in each row and each column. To express that there is at most one rook on a specific row (column) we need $8 \times 7/2 = 28$ clauses of size two, which means that for the second constraint we would need in total $2 \times 8 \times 28 = 448$ clauses of size two.

3. We would need to capture that there is at most one queen in each diagonal, which we can do in the same way as we did for rows and columns. That would require an additional $2 \times \left(0 + \frac{2\times1}{2} + \frac{3\times2}{2} + \cdots + \frac{8\times7}{2} + \cdots + \frac{3\times2}{2} + \frac{2\times1}{2} + 0\right) = 240$ clauses of size 2.