Solving Sokoban screens using ROBDDs

S. Kikitamara P.T. Jager

April 21, 2016

1 Introduction

2 Implementation

2.1 Encoding

In our implementation Sokoban screens are encoded using variables denoting the x and y position of each block and the man. Initial, error and goal properties are then defined in terms of these variables, as is the transition relation. Below we will show this more clearly and more formally and we will also link the formal specification to the implementation in Sylvan.

Ultimately we will combine these parts to check that there is a path from the initial configuration to a configuration which satisfied the goal property without ever satisfying the error property. More formally, we will check the following formula:

$$\mathsf{EG}(\neg\mathsf{error} \cup \mathsf{goal}) \tag{1}$$

Screens We interpret a screen as a grid of x columns and y rows (position (0,0) is considered the position in the top left). Each block and the man has a position in the screen identified by which row and column it is in. Goal positions and walls are not explicitly encoded in the screen but are instead encoded in goal and error properties.

Positions For the man and for each block variables are defined which indicate in which row and in which column the block/man is. For a screen with x columns, y rows and n blocks we define $(n+1)*(x+y)$ variables.

For each block we define

$bx_{i,x}$ which indicates if block i is in the column numbered x.

$by_{i,y}$ which indicates if block i in in the row numbered y.

For the man we define similar variables, but named just mx_x for the columns and my_y for the rows as a screen only has one man.

The actual implementation defines just a sequence of BDDvars, one for each of these variables.

Transition relation Before showing the different properties of the system, we will first show the transition relation which describes how the man moves around the board and how this affects blocks around him.

The main idea behind the transition relation is rather simple. Given that the man is in a certain position (x, y) he can move either up, down, left or right, and if there is a block in that position that block moves in the same direction. If the man is currently at an edge then he can not move in the direction which would make him fall of the board, and if the man is separated from the edge of the board by only a block, then he also can not move in that direction as that would push the block outside of the board.

Any transitions which result in either the man or a block overlapping with a wall, or which result in two blocks overlapping are considered erroneous and their resulting states will satisfy the error property.

Equation 2 shows the formal transition relation. blocks is considered the set of block(numbers) in this screen, $rows$ the set of row numbers and $cols$ the set of column numbers. max_r will be the maximum column number and max_c the maximum column number. In this relation the primed version of each variable denotes that variable in the next state, i.e. the state after the transition has happened. We will only fully show the formula for moving right, the formulas for moving left, up and down are defined similar.

$$
\begin{aligned}
\bigwedge_{x \in cols, y \in rows} (mx_x \wedge my_y) \implies \Big(\\
&\Big(x \neq max_c \wedge \big(x \neq (max_c - 1)(\wedge_{b \in blocks} \neg(bx_{i,x+1}) \wedge by_{b,y})\big) \\
&\qquad \implies \big(\neg mx'_x \wedge mx'_{x+1} \wedge my'_y \\
&(\wedge_{b \in blocks} bx_{b,x+1} \wedge by_{b,y} \implies bx'_{b,x+1} \wedge bx'_{b,x+2} \wedge by'_{b,y}))\Big) \\
&\vee \Big(\text{similar for left}\Big) \vee \Big(\text{similar for up}\Big) \vee \Big(\text{similar for down}\Big)\Big)
\end{aligned} \tag{2}
$$

For standard screens this encoding is actually more complex than necessary. Since all screens are bordered by a layer of walls it is not necessary to check if a block would be pushed of screen and if the man would walk of screen, since states in which a transition could result in this would already satisfy the error property, since a block or the man would overlap with the outside walls. However defining the transition like this allows for an optimization in which all layers of outside wall have been removed from a board, reducing the state space.

Error property A state is considered to be an error state if either the man or one of the blocks is overlapping with a wall, or when two blocks are overlapping. Equations 3 shows this relation formally. In this $walls$ is the set of walls on the screen, and w^x denotes the column wall w is in and w^y the row it is in.

$$\bigvee_{w \in walls, b \in blocks} (bx_{b,w^x} \wedge by_{b,w^y})$$
$$\vee \left(\bigvee_{x \in cols, y \in rows} \bigvee_{b \in blocks} \bigvee_{b' \in blocks \setminus \{b\}} (bx_{b,x} \wedge bx_{b',x} \wedge by_{b,y} \wedge by_{b',y}) \right) \quad (3)$$

Goal property A state is a goal state if all blocks are on a goal position and no two blocks occupy the same goal position. Equation 4 shows this formally. In this *goals* is the set of goal positions and g^x denotes the column goal g is in, and g^y the row it is in.

$$\bigwedge_{g \in goals} \bigvee_{b \in blocks} bx_{b,g^x} \wedge by_{b,g^y}$$
$$\wedge \left(\bigvee_{x \in cols, y \in rows} \bigvee_{b \in blocks} \bigvee_{b' \in blocks \setminus \{b\}} (bx_{b,x} \wedge bx_{b',x} \wedge by_{b,y} \wedge by_{b',y}) \right) \quad (4)$$

Initial property The initial property describes a state in which all blocks and the man are in their initial position on the board (and nowhere else). Equation 5 shows this. In this $init^x_b$ denotes the the initial column of block b, $init^y_b$ denotes the initial column of block b. $init^x_m$ denotes the initial column of the man and $init^y_m$ the initial row of the man.

$$\bigwedge_{x \in cols, y \in rows} \bigwedge_{b \in blocks} \Big(((x = init^x_b \implies bx_{b,x}) \wedge (x \neq init^x_b \implies \neg bx_{b,x})) \wedge ((y = init^y_b \implies by_{b,y}) \wedge (y \neq init^y_b \implies \neg by_{b,y})) \Big)$$
$$\wedge \bigwedge_{x \in cols, y \in rows} \Big(((x = init^x_m \implies mx_x) \wedge (x \neq init^x_m \implies \neg mx_x)) \wedge ((y = init^y_m \implies my_y) \wedge (y \neq init^y_m \implies \neg my_y)) \Big) \quad (5)$$

3 Results

4 Usage