Lecture Notes for

Logic and Computability

Course Number: IND04033UF

Contact

Bettina Könighofer
Institute for Applied Information Processing and Communications (IAIK)
Graz University of Technology, Austria
bettina.koenighofer@iaik.tugraz.at

Graz University of Technology

Table of Contents

10
Modelling Systems and Symbolic Encoding

Many digital circuits and programs are examples of reactive systems. Such systems typically exhibit frequent interactions with their environment and often do not terminate. The most important feature of a reactive system that we need to capture is its *state*. A state is a snapshot of the system that captures the values of the variables of the system at a particular instant of time. To analyze system behavior we also need to know how the state of the system changes as the result of some action of the system. Such a pair of states determines a *transition* of the system. Consequently, the behaviors of a reactive system can be defined in terms of its transitions.

In this chapter, we use a type of state transition graph called a *transition system* to model the behavior of reactive systems. A transition system consists of a set of states and a set of transitions between states. Paths in transition systems correspond to behaviors of the system. Although transition systems are very simple models, they are sufficiently expressive to capture many aspects of temporal behavior that are most important for reasoning about reactive systems.

Transition systems are often very large. Therefore, we will use formulas to symbolically represent transition systems. We will see that it is straightforward to translate a transition system to a formula, and vise versa.

10.1 Transition Systems

Definition - Transition system. *A transition system* $\mathcal{T}$ *is a triple* (S, S_0, R) *where*

1. S *is a set of states.*
2. $S_0 \subseteq S$ *is the set of initial states.*
3. $R \subseteq S \times S$ *is a transition relation.*

Transition systems are frequently visualized by means of directed graphs.

Example. Draw the graph of a transition system $\mathcal{T}_1$ with:

$$S = \{s_1, s_2, s_3\}, \quad S_0 = \{s_1\}, \quad R = \{(s_1, s_2), (s_2, s_1), (s_3, s_2)\}$$

.

Solution.

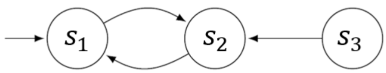


Figure 10.1: Graph for transition system $\mathcal{T}_1$.

Example. Consider the example of a traffic light. Initially the red light is on. After some time, the traffic light switches from the red light to a state where both the red and the yellow light are on. From red/yellow, it switches to green, from green to yellow, and from yellow back to red, and so on. Model the traffic light as transition system.

Solution. We use the following states:

- s_r indicates that the red light is on.
- s_y indicates that the yellow light is on.
- s_g indicates that the green light is on.
- s_{ry} indicates that the red and yellow lights are on.

The transition system is then given by: $\mathcal{T}_2 = (S, S_0, R)$ with $S = \{s_r, s_y, s_g, s_{ry}\}$, $S_0 = \{s_r\}$, $R = \{(s_r, s_r), (s_r, s_{ry}), (s_{ry}, s_{ry}), (s_{ry}, s_g), (s_g, s_g), (s_g, s_y), (s_y, s_y), (s_y, s_r)\}$. The transition system $\mathcal{T}_2$ is represented as graph in Figure 10.2:

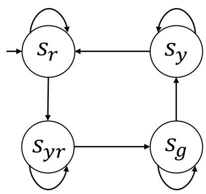


Figure 10.2: Graph for transition system $\mathcal{T}_2$.

10.2 Symbolic Encoding

Our goal is to represent transition systems via formulas in propositional logic. This gives us an efficient way to store and to work with transition systems. Therefore, we need to be able to encode sets of states and transitions symbolically, this means to express them with propositional logic formulas.

10.2.1 Symbolic Representation of Sets of States

Since states are instantaneous descriptions of a system, it is natural to identify states with valuations of the system variables. To this end, let $V = \{v_1, \ldots, v_n\}$ be the set of (Boolean) system variables. Thus, a state s defines a valuation for all variables $v \in V$; it assigns truth values to all variables. For a given state, we can write a formula that is true for exactly that valuation of state variables.

Symbolic Representation of States via Binary Encoding. Given a state space $S = \{s_0, s_1, \ldots\}$ of the size $|S| \leq 2^n$, we need n Boolean variables $\{v_0, \ldots, v_{n-1}\}$ to symbolically represent the state space. We use an Binary encoding to represent states in the state space.

- Using a single Boolean variable v_0, we can symbolically represent two states $S = \{s_0, s_1\}$.
  - The formula $\neg v_0$ symbolically represent the state s_0.
  - The formula v_0 symbolically represent the state s_1.
- Using 2 Boolean variables v_0 and v_1, we can symbolically represent four states $S = \{s_0, s_1, s_2, s_3\}$.
  - The formula $\neg v_1 \wedge \neg v_0$ symbolically represent the state s_0.
  - The formula $\neg v_1 \wedge v_0$ symbolically represent the state s_1.
  - The formula $v_1 \wedge \neg v_0$ symbolically represent the state s_2.
  - The formula $v_1 \wedge v_0$ symbolically represent the state s_3.
- Using 3 Boolean variables v_0, v_1 and v_2, we can symbolically represent 8 states $S = \{s_0, s_1, s_2, s_3 \ldots, s_7\}$.
  - The formula $\neg v_2 \wedge \neg v_1 \wedge \neg v_0$ symbolically represent the state s_0.
  - ....
  - The formula $v_2 \wedge v_1 \wedge v_0$ symbolically represent the state s_7.

For every Boolean variable we add, the number of possible combinations and therefore states that we can encode, doubles. In general with n bits we can encode up to 2^n states.

Example. Given a state space $S = \{s_0, s_1, s_2, s_3, s_4\}$. How many Boolean variables to you need to represent all states in the state space symbolically?

Solution. The number of Boolean variables required to encode the set of five states is $n \geq log_2\ 5 = 2.3219281$. Therefore, we need at least *3 Boolean variables* v_2, v_1 and v_0.

Example. Given a state space of the size $|S| = 2^{12} = 4096$. Give the symbolic encoding for the state s_{457}.

Solution. For the symbolic encoding we need 12 Boolean variables, $\{v_{11}, \ldots, v_0\}$. Let v_{11} be the most significant bit, and v_0 the least significant bit.

We have that

$$(457)_{10} = (0001\ 1100\ 1001)_2$$

therefore, we get the symbolic encoding

$$\neg v_{11} \wedge \neg v_{10} \wedge \neg v_9 \wedge v_8 \wedge v_7 \wedge v_6 \wedge \neg v_5 \wedge \neg v_4 \wedge v_3 \wedge \neg v_2 \wedge \neg v_1 \wedge v_0.$$

Symbolic Representation of Sets of States. The strength of symbolic encoding is that we are often able to represent huge sets with relatively small formulas. Therefore, symbolic encoding gives us a method to handle extremely large state spaces.

A formula precisely represents the set of all valuations that make it true and each valuation represents a state.

Thus, a formula can be viewed as a *symbolic representation* of a set of states.

Example. Given a state space of the size $|S| = 8$. Give the symbolic encoding for the set of states $\{s_5, s_1\}$.

Solution. We encode s_5 with $(v_1 \wedge \neg v_2 \wedge v_3)$ and s_1 with $(\neg v_1 \wedge \neg v_2 \wedge v_3)$. The set $\{s_5, s_1\}$ is represented with the formula:

$$(v_1 \wedge \neg v_2 \wedge v_3) \vee (\neg v_1 \wedge \neg v_2 \wedge v_3) = (\neg v_2 \wedge v_3)$$

Example. Given a state space of the size $|S| = 1024$ and two sets of states $B = \{s_0, s_1, s_2, \ldots, s_{511}\}$ and $C = \{s_{256}, s_{257}, \ldots, s_{767}\}$. Find a propositional logic formula b that symbolically encodes the set B, and find a formula c that encodes the set C.

Solution. Using the variables $\{v_9, \ldots, v_0\}$ the encoding of B is simply given by $b = \neg v_9$.

In order to find a simple formula c for C, we can analyze the Binary representations of the contained states.

- The binary encoding of 256 is 0100000000.
- The binary encoding of 257 is 0100000001.
- ...
- The binary encoding of 767 is 1011111111.

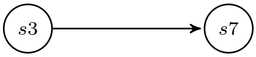
We can see that only the values of v_9 and v_8 are important to distinguish whether a state is included in C or not. We get the final formula:

$$c = (\neg v_9 \wedge v_8) \vee (v_9 \wedge \neg v_8) = (v_9 \oplus v_8).$$

10.2.2 Symbolic Representation of the Transition Relation

To represent the transition relation of transition system symbolically, we extend the idea used for the symbolic representation of sets of states from above. This time, we use a formula to represent *a set of ordered pairs of states.* Therefore, we create a second set of variables V'. We think of the variables in V as *present state variables* and the variables in V' as *next state variables.* Each variable $v \in V$ has a corresponding next state variable $v' \in V'$. A transition is a valuation for the variables in $V \cup V'$.

Example. Given a state space of the size $|S| = 2^3 = 8$. Give the formula for the transition from state s_3 to state s_7 as shown below.



Solution. The current state s_3 is encoded via $\neg v_2 \wedge v_1 \wedge v_0$. The next state s_7 is encoded via $v_2' \wedge v_1' \wedge v_0'$. This results in the following formula encoding the transition:

$$\neg v_2 \wedge v_1 \wedge v_0 \wedge \ v_2' \wedge v_1' \wedge v_0'.$$

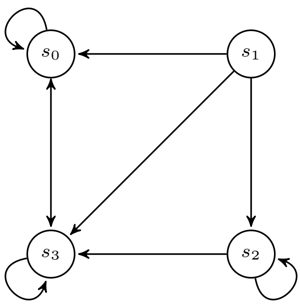
Example. Which transition is encoded by the following formula:

$$(v_1 \wedge v_2 \wedge v_3 \wedge v_1' \wedge v_2' \wedge v_3').$$

Solution. The transition which is a self loop from state s_7 to s_7.

Sets of transitions can be formed using disjunctions to connect the encodings for the individual transitions, which can then be further simplified. For transition systems with dense transition relations, a simple way to get a short formula for the transition relation is to encode the edges that are *not* contained in the transition relation, and to *negate* the resulting formula.

Example. Consider the following transition system with $|S| = 4$. Give the formula that symbolically represents the transition relation.



Solution: Using the variables v_1 and v_0, we can define the transition relation using the following formula:

$$\begin{aligned} \neg\Big((\neg v_1 \wedge \neg v_0 \wedge \neg v_1' \wedge v_0')\vee \\ (\neg v_1 \wedge v_0 \wedge \neg v_1' \wedge v_0')\vee \\ (v_1 \wedge \neg v_0 \wedge \neg v_1' \wedge v_0')\vee \\ (v_1 \wedge v_0 \wedge \neg v_1' \wedge v_0')\vee \\ (\neg v_1 \wedge \neg v_0 \wedge v_1' \wedge \neg v_0')\vee \\ (v_1 \wedge \neg v_0 \wedge \neg v_1' \wedge \neg v_0')\Big) \end{aligned}$$

We can simplify the formula to:

$$\begin{aligned} \neg\Big((\neg v_1' \wedge v_0')\vee \\ (\neg v_1 \wedge \neg v_0 \wedge v_1' \wedge \neg v_0')\vee \\ (v_1 \wedge \neg v_0 \wedge \neg v_1' \wedge \neg v_0')\Big) \end{aligned}$$

10.2.3 Symbolic Encoding of arbitrary Sets

Aside using binary encoding to efficiently represent transition systems, we want to be able to efficiently represent and manipulate sets of arbitrary symbols.

In order to encode arbitrary sets, you need to perform the following steps.

1. Determine the cardinality of the set of symbols and the number of Boolean variables to use in the encoding.
2. Fix the correspondence between each symbol in the set and the valuation of the Boolean variables representing it.

With n Boolean variables, a maximum of 2^n elements can be encoded. Therefore, if N is the cardinality of the set of symbols, we need $n \geq \lceil log_2 N \rceil$ Boolean variables for the symbolic encoding.

Once the number of bits has been decided, the relationship between each element in the set and a concrete validation of the Boolean variables needs to be fixed. Each element must have at least one binary representation, and each sequence of bits must correspond with one element in the set.

Example. Suppose you have to encode the symbols in the set $A = \{Yellow, Orange, Green, Blue\}$. Find a symbolic representation for A. Give the formulas b and c representing the set $B = \{Yellow, Orange\}$ and $C = \{Green, Orange\}$ respectively.

Solution. We use 2 Boolean variables for the encoding.

Symbol	**Encoding** v_1, v_0
Yellow	00
Orange	01
Green	10
Blue	11

Using this encoding, we end up in the following formulas for b and c.

$$b = (\neg v_1); \quad c = (v_1 \oplus v_0)$$

10.2.4 Set Operations on Symbolically Encoded Sets

When using formulas to characterize sets, we can perform the usual set operations by appropriate transformations to symbolic operations. Let A and B denote subsets of a set S and a and b denote the respective symbolic representations of A and B. Then we have:

Intersection	$A \cap B$	$a \wedge b$
Union	$A \cup B$	$a \vee b$
Difference	$A \backslash B$	$a \wedge \neg b$

Similar transformations apply in the case of relational operators over sets. For example, we can check $A \subseteq B$ by determining whether the formula $a \rightarrow b$ evaluates to true.

Equality	$A = B$	$a \equiv b$
Subset	$A \subseteq B$	$a \rightarrow b$

Example. Given the sets of states S with $|S| = 1024$, $B = \{s_0, s_1, s_2, \ldots, s_{511}\}$ and $C = \{s_{256}, s_{257}, \ldots, s_{767}\}$ from the example above. Compute the symbolic representation for the sets $D = B \cup C$, $E = B \cap C$, and $F = S \setminus E$.

Solution. We first compute the formulas $b = \neg v_9$ and $c = (v_9 \oplus v_8)$. Using b and c, we compute:

- $d = b \vee c \;\equiv\; (x_9 \oplus x_8) \vee \neg x_9 \;\equiv\; (x_9 \wedge \neg x_8) \vee \neg x_9$
- $e = ((x_9 \wedge \neg x_8) \vee (\neg x_9 \wedge x_8)) \wedge \neg x_9 \;\equiv\; \neg x_9 \wedge x_8$
- $f = \mathit{true} \wedge \neg(\neg x_9 \wedge x_8) \;\equiv\; x_9 \vee \neg x_8$

Declaration of Sources

Chapter 8 is based on the following book.

Edmund M. Clarke Jr., Orna Grumberg, Daniel Kröning, Doron Peled, Helmut Veith: *Model Checking*. Second edition. MIT Press. ISBN-13: 978-0262038836. ISBN-10: 0262038838