Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Computer Engineering and Networks Laboratory
Computer Engineering Group (TEC)

Prof. L. Thiele / X.He

Discrete Event Systems

Solution to Exercise: Verification of Finite Automata (Part 1)

A classic complex system design involve many steps. Often enough, various components are designed independently, and then assemble together to shape the overall system. For example, a car manufacturer essentially assembles various parts produced by specialized suppliers. The combined behavior of the assembled parts results in the overall system behavior.

However, if some combined behavior are part of the specification (e.g., one can listen to the radio while driving), some others must be prevented (e.g., the engine must not start if the gas tank is open). The interaction between components must be controlled. That is the goal of *system specification.*

In this exercise, we look at two of the steps in this process. In the first exercise, we use representation of sets (and their translation into boolean expressions) in order to express a system property. Simple properties can be combined together to finally capture complex and high level ones, like the famous "*The system is safe*"... what ever that means, depending on the context. But, almost always, there is more than one option to satisfy all specifications. Otherwise, we would all have the same car, phone... In the second exercise, we use Binary Decision Diagrams to compare two boolean equations. In our context, this can be useful to guarantee that our system design (the choices we made) satisfies indeed the system's specifications, formulated as a boolean equation.

1 Sets Representation

1.1 Warm-up

In the early the design phase of a system, it is common to qualify some of the accessible states as *faulty*, or *error* states. One goal is to ensure that the system won't enter such states. Let us define a few sets of states:

- X: the whole set of states,
- N: the set of nominal states,
- E: the set of error states,
- O: the set of state where there is a memory overflow.

We denote by ψ_Q the characteristic function of the set Q, i.e., $x \in Q \Leftrightarrow \psi_Q(\sigma(x)) = 1$ where $\sigma(x)$ is the binary encoding of the state x.

For each sub-question bellow, first draw a 2D-representation of the sets of states before answering.

a) What is ψ_X, the characteristic function of the whole set of states?

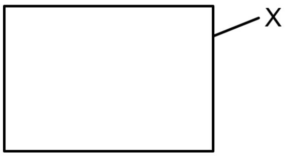
b) "*Each state is either a nominal or an error state or both*". Express this property in term of sets and characteristic functions.

c) *"If a state is in the overflow set, it is not a nominal state"*. Express this property in term of sets and characteristic functions.

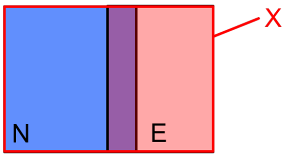
d) Describe Q_1, the set of error states which are not an overflow, in term of sets and characteristic functions.

e) Describe Q_2, satisfying "$O \Rightarrow E$", i.e., the set of state for which this property holds, in term of sets and characteristic functions.
Hint: "$O \Rightarrow E$" reads "O implies E", in other words, if a state is in O, then it is in E. Beware that we look for the states for which this property holds true, not a relation between O and E.

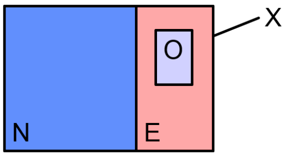
a) $\psi_X = 1$



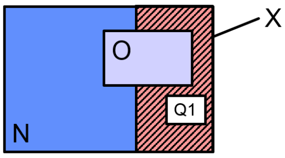
b) $N \cup E = X \quad \Leftrightarrow \quad \psi_N + \psi_E = 1$



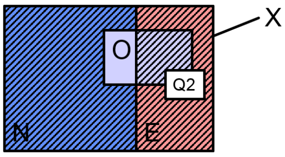
c) $N \cap O = \emptyset \quad \Leftrightarrow \quad \psi_N \cdot \psi_O = 0$



d) $Q_1 = E \backslash O \quad \Leftrightarrow \quad \psi_{Q_1} = \psi_E \cdot \overline{\psi_O}$



e) $$Q_2 = (O \cap E) \cup \overline{O} \quad \begin{aligned} &= (O \cup \overline{O}) \cap (E \cup \overline{O}) \\ &= X \cap (E \cup \overline{O}) \\ &= E \cup \overline{O} \end{aligned} \quad \Leftrightarrow \quad \psi_{Q_2} = \psi_E + \overline{\psi_O}$$



1.2 Specification composition

Bellow are a set of constraints, expressed in a textual form, and a set of boolean variables that encode the system states. Your task consists in **(i) expressing each of the constraints** independently, **(ii) combining them** to express the overall system specification.

The system we consider is a sensor network composed of 3 sensor nodes, a bus and a sink (a node where data is collected). The network can be on nominal or bootstrapping mode. In order to save energy, nodes are put in sleep mode whenever possible.

C1 When one node in using the bus, the sink must be awake to receive data.

C2 No more than one node can use the bus at the same time.

C3 In bootstrapping mode, the sink must be awake and the nodes cannot use the bus.

We consider the following encoding:

$x_s = 1$	The sink is awake.
$x_b = 1$	The network is in bootstapping mode.
$x_i = 1$	Node i is using the bus ($i \in \{1, 2, 3\}$).

a) Express the specification of **C1**, **C2** and **C3**.

b) What is the specification of the desired behavior? **Hint:** All constraints should be satisfied.

C1 $\psi_{C1} = (x_1 + x_2 + x_3)x_s + \overline{x_1} \cdot \overline{x_2} \cdot \overline{x_3} = x_s + \overline{x_1} \cdot \overline{x_2} \cdot \overline{x_3}$

C2 $\psi_{C2} = x_1 \cdot \overline{x_2} \cdot \overline{x_3} + \overline{x_1} \cdot x_2 \cdot \overline{x_3} + \overline{x_1} \cdot \overline{x_2} \cdot x_3 + \overline{x_1} \cdot \overline{x_2} \cdot \overline{x_3}$

C3 $\psi_{C3} = x_b \cdot x_s \cdot \overline{x_1} \cdot \overline{x_2} \cdot \overline{x_3} + \overline{x_b} = x_s \cdot \overline{x_1} \cdot \overline{x_2} \cdot \overline{x_3} + \overline{x_b}$

The specification consists in satisfying all constraints at all times:

$$\psi_N = \psi_{C1} \cdot \psi_{C2} \cdot \psi_{C3}$$

2 Binary Decision Diagrams

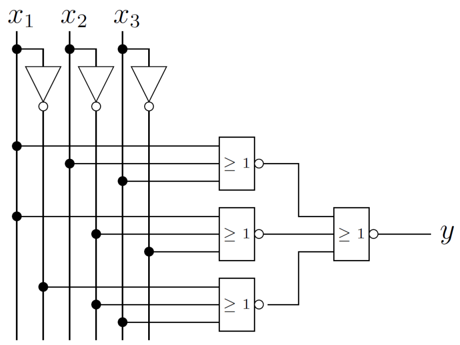
For an Reduced Ordered Binary Decision Diagram (ROBDD), we denote by $\Pi : x_1 < x_2 < \ldots < x_n$ the variable order, where x_1 is the highest variable of the tree, x_2 the second highest, and so on. An ordering Π_1 is said to be better than Π_2 for an ROBDD G if G contains less nodes when using Π_1 rather than Π_2 (eventually after merging equivalent nodes).
In the following, use the following notation to represent BDDs: A solid arc (———) if the variable labeling the parent node evaluates to 1, and the dashed arc (– – – –) otherwise. **Do not** use color (it is a bad habit to take. . .).

2.1 Verification using BDDs

You are in the process of designing a processing architecture. Your specification analysis results in the following desired function to implement:

$$f_1 : (x_1\overline{x_2} + x_1x_3 + \overline{x_2}x_3 + \overline{x_1}x_2\overline{x_3})$$

For practical reason, you only dispose of invert- and NOR- gates to implement your circuit. Considering those constraints, your automated synthesis program returns this schematics:



Your team-leader is quite old fashion and does trust these new fancy software so much (or maybe he/she is just testing you!). He/She asks that you verify the schematic circuit does indeed implement the same function as f_1. This can be done efficiently using ROBDDs.

a) Express the function f_2 realized by the circuit.

b) Draw and compare the ROBDDs of f_1 and f_2 using the ordering of variables $\Pi : x_1 < x_2 < x_3$. Do they implement the same behavior?

a) $f_2 : y = \overline{\overline{x_1 + x_2 + x_3} + \overline{x_1 + \overline{x_2} + \overline{x_3}} + \overline{\overline{x_1} + \overline{x_2} + x_3}}$

b) For f_1, we have

Fall $x_1 = 0$

$y_{|x_1=0} = \overline{x_2}x_3 + x_2\overline{x_3}$

Fall $x_2 = 0$

$y_{|x_1=0,x_2=0} = x_3$

Fall $x_2 = 1$

$y_{|x_1=0,x_2=1} = \overline{x_3}$

Fall $x_1 = 1$

$y_{|x_1=1} = \overline{x_2} + x_3 + \overline{x_2}x_3$

Fall $x_2 = 0$

$y_{|x_1=1,x_2=0} = 1$

Fall $x_2 = 1$

$y_{|x_1=1,x_2=1} = x_3$

For f_2, we have

Fall $x_1 = 0$

$y_{|x_1=0} = \overline{\overline{x_2 + x_3} + \overline{\overline{x_2} + \overline{x_3}}}$

Fall $x_2 = 0$

$y_{|x_1=0,x_2=0} = \overline{\overline{x_3} + \overline{1 + \overline{x_3}}} = x_3$

Fall $x_2 = 1$

$y_{|x_1=0,x_2=1} = \overline{\overline{1} + \overline{\overline{x_3}}} = \overline{x_3}$

Fall $x_1 = 1$

$y_{|x_1=1} = \overline{\overline{1} + \overline{1} + \overline{\overline{x_2} + x_3}} = \overline{x_2} + x_3$

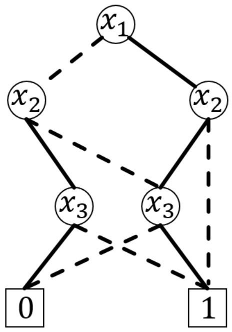
Fall $x_2 = 0$

$y_{|x_1=1,x_2=0} = 1$

Fall $x_2 = 1$

$y_{|x_1=1,x_2=1} = x_3$

Both ROBDDs have the same falls. They are equivalent.



2.2 BDDs with respect to different orderings

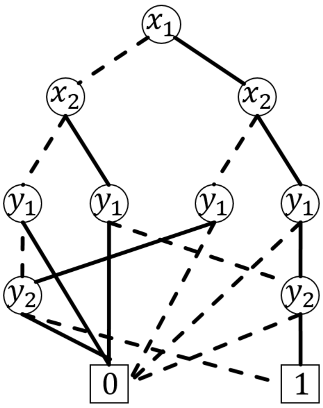
a) Consider the boolean function $g(x_1, x_2, y_1, y_2) = (x_1 == y_1) \cdot (x_2 == y_2)$ and the ordering of variables $\Pi : x_1 < x_2 < y_1 < y_2$. Give the Boole-Shannon decomposition of g with respect to Π.
Hint: "$(x == y)$" is a short for the boolean expression: $x \cdot y + \overline{x} \cdot \overline{y}$

b) Draw the corresponding ROBDD for g.

c) Let us now consider the new ordering $\Pi' : x_1 < y_1 < x_2 < y_2$. Use it to reconstruct the ROBDD of g. Is Π' a better ordering than Π for g?

a) $g = x_1\{x_2[y_1(y_2) + \overline{y_1}(0)] + \overline{x_2}[y_1(\overline{y_2}) + \overline{y_1}(0)]\} + \overline{x_1}\{x_2[y_1(0) + \overline{y_1}(y_2)] + \overline{x_2}[y_1(0) + \overline{y_1}(\overline{y_2})]\}$

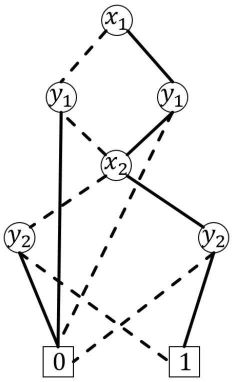
b)



c) With the new ordering Π', the Boole-Shannon decomposition becomes

$$g = x_1\{y_1[x_2(y_2) + \overline{x_2}(\overline{y_2})] + \overline{y_1}[0]\} + \overline{x_1}\{y_1[0] + \overline{y_1}[x_2(y_2) + \overline{x_2}(\overline{y_2})]\}$$

This is a better ordering as it leads to a ROBDD with fewer nodes as with Π (6 instead of 9).


x_1
y_1
y_1
x_2
y_2
y_2
0
1