Questionnaire “Logic and Computability”

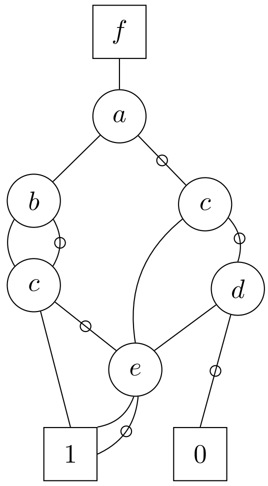
Summer Term 2023

Contents

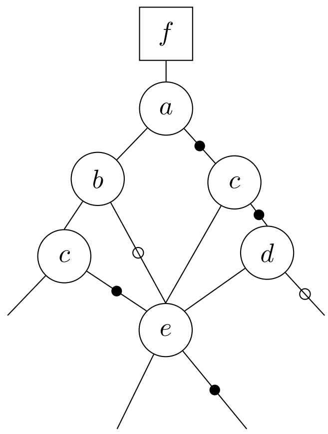
3 Binary Decision Diagrams

3.1 Reduced Ordered Binary Decision Diagrams

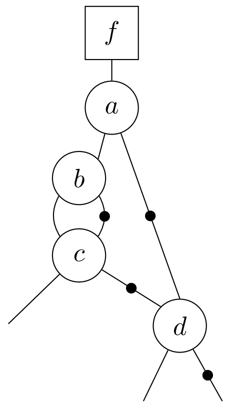
3.1.1 Given the *Binary Decision Diagram (BDD)* below, label and explain the different elements of the diagram.



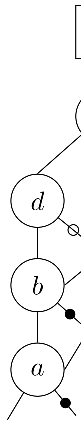
3.1.2 Given the following Binary Decision Diagram that represents the formula f. Compute its disjunctive normal form DNF(f).



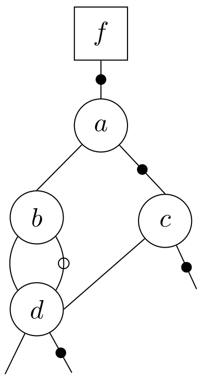
3.1.3 Given the following Binary Decision Diagram that represents the formula f. Compute its disjunctive normal form DNF(f).



3.1.4 Given the following Binary Decision Diagram that represents the formula f. Compute its disjunctive normal form DNF(f).



3.1.5 For the following binary decision diagram:

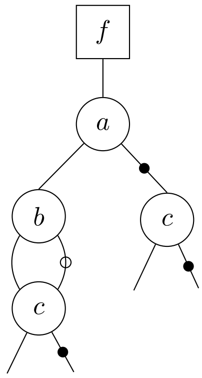


Check if the following models are satisfying:

$$\mathcal{M}_1 = \{a = \top, b = \top, c = \bot, d = \bot\},$$
$$\mathcal{M}_2 = \{a = \bot, b = \bot, c = \top, d = \top\}, \text{ and}$$

compute DNF(f).

3.1.6 For the following binary decision diagram:



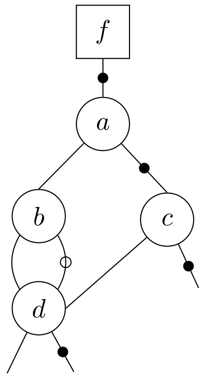
Check if the following models are satisfying:

$$\mathcal{M}_1 = \{a = \top, b = \top, c = \bot\},$$
$$\mathcal{M}_2 = \{a = \bot, b = \bot, c = \bot\}, \text{ and}$$

compute DNF(f).

3.1.7 For the following binary decision diagram:

Note: Else-edges are marked with circles. Filled circles represent the *complemented* attribute. Dangling edges are assumed to point to the constant node `true`.

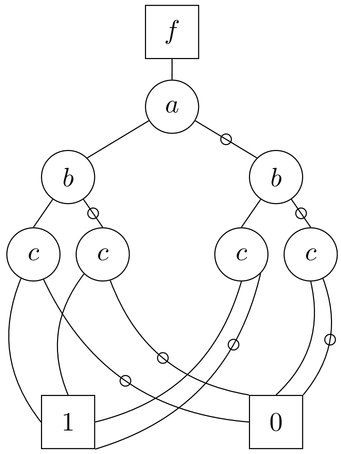


Check if the following models are satisfying:

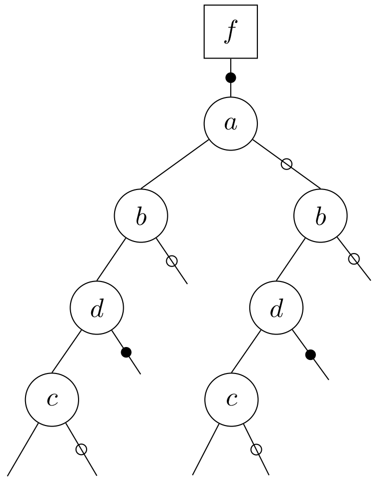
$$\mathcal{M}_1 = \{a = \top, b = \top, c = \bot, d = \bot\},$$
$$\mathcal{M}_2 = \{a = \bot, b = \bot, c = \top, d = \top\}, \text{ and}$$

compute DNF(f).

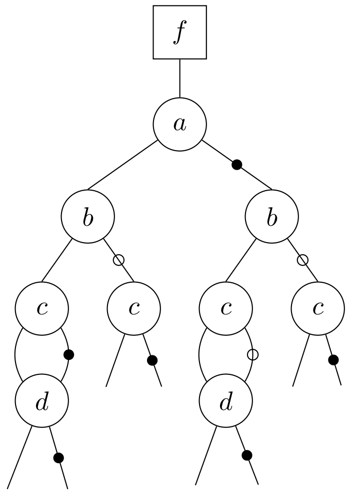
3.1.8 Transform the given Binary Decision Diagram into a reduced and ordered BDD.



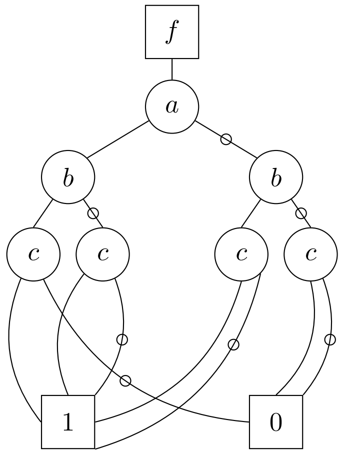
3.1.9 Transform the given Binary Decision Diagram into a reduced and ordered BDD.



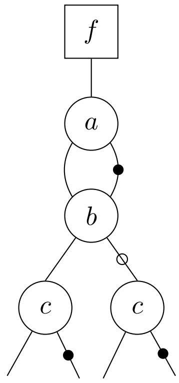
3.1.10 Transform the given Binary Decision Diagram into a reduced and ordered BDD.



3.1.11 Transform the given Binary Decision Diagram into a reduced and ordered BDD.



3.1.12 Transform the given Binary Decision Diagram into a reduced and ordered BDD.



3.1.13 A reduced and ordered BDD is a canonical representation of a propositional formula. Explain the term *canonical* in the context of propositional formulas and explain why reduced and ordered BDDs give canonical representations.

3.1.14 Give the definition of a cofactor of a formula f with respect to an assignment A.

3.1.15 What is the worst-case size of a binary decision diagram? What is the advantage of computing a reduced and ordered BDD to represent a formula compared to using a truth table?

3.1.16 Tick all properties that apply to a reduced and ordered binary decision diagram.

- ☐ A reduced and ordered BDD is a canonical representation of the formula it represents, for any fixed variable order.
- ☐ Since it is reduced, the number of nodes in the reduced and ordered BDD does not exceed $2n^2$, where n is the number of variables.
- ☐ The graph of an BDD may contain cycles.
- ☐ A BDD represents a propositional formula as directed acyclic graph (DAG).
- ☐ Every node with two non-complemented outgoing edges has two distinct child nodes.
- ☐ No two nodes in an reduced and ordered BDD represent the same cofactor.

3.1.17 Given you have computed the reduced and ordered BDD for a formula f. How can you compute the BDD representation for $\neg f$ in *constant* time?

3.1.18 How can you compute the propositional formula f represented by a given BDD?

3.1.19 Give the definition of redundant nodes in a BDD. Give an example for a BDD that contains at least one redundant node.

3.1.20 In the context of *Binary Decision Diagrams (BDDs)*, how does the variable order impact the BDD?

3.1.21 Tick all properties that apply to a reduced and ordered BDD.

- ☐ If the *else*-edge of a node is complemented, it may point to the same child node as the *then*-edge.
- ☐ Using the reduced and ordered BDD representation of formula f, it is possible to whether f is valid in constant time.
- ☐ The size of a BDD is independent on the variable order.
- ☐ Using complemented edges, negation can be performed in constant time.
- ☐ The size of a BDD is independent of the variable order.

3.1.22 When do we consider a BDD to be reduced? Explain the types of redudancies that are not allowed to appear in a reduced and ordered BDD.

3.1.23 Explain how a reduced and ordered BDD can be used to determine the satisfiability of the formula f it is representing.

3.1.24 Explain how a reduced and ordered BDD can be used to determine whether the formula f it is representing is valid.

3.2 Construction of Reduced Ordered BDDs

3.2.1 Construct a Reduced Ordered Binary Decision Diagram (ROBDD) for the formula

$$f = (\neg x \vee \neg y) \wedge (x \wedge (y \vee z)),$$

using *variable order* $y < z < x$. Use complemented edges and a node for `true` as the only constant node. To simplify drawing, you may assume that *dangling edges* point to the constant node. Write down all cofactors that you compute to obtain the final result and mark them in the graph.

3.2.2 Construct a Reduced Ordered Binary Decision Diagram (ROBDD) for the formula

$$f = (\neg x \wedge \neg y) \vee (x \wedge y),$$

using *variable order* $z < x < y$. Use complemented edges and a node for `true` as the only constant node. To simplify drawing, you may assume that *dangling edges* point to the constant node. Write down all cofactors that you compute to obtain the final result and mark them in the graph.

3.2.3 Construct a Reduced Ordered Binary Decision Diagram (ROBDD) for the formula

$$f = (\neg p \vee r) \wedge (q \vee \neg p) \wedge (\neg q \vee p)$$

using *variable order* $r < q < p$. Use complemented edges and a node for `true` as the only constant node. To simplify drawing, you may assume that *dangling edges* point to the constant node. Write down all cofactors that you compute to obtain the final result and mark them in the graph.

3.2.4 Construct a Reduced Ordered Binary Decision Diagram (ROBDD) for the formula

$$f = (q \wedge \neg s) \vee (s \wedge (\neg r \vee p)) \vee (p \wedge q \wedge r)$$

using *variable order* $p < q < r < s$. Use complemented edges and a node for `true` as the only constant node. To simplify drawing, you may assume that *dangling edges* point to the constant node. Write down all cofactors that you compute to obtain the final result and mark them in the graph.

3.2.5 Construct a ROBDD for the formula

$$f = (a \wedge d \wedge c) \vee (b \wedge \neg d \wedge \neg a) \vee (c \rightarrow \neg d) \vee (a \rightarrow \neg b)$$

using *variable order* $b < a < d < c$. Use complemented edges and a node for `true` as the only constant node. To simplify drawing, you may assume that *dangling edges* point to the constant node. Write down all cofactors that you compute to obtain the final result and mark them in the graph.

3.2.6 Construct a reduced ordered binary decision diagram (ROBDD) for the formula

$$f = (p \oplus q) \wedge \neg r$$

using *variable order* $p < q < r$. Use complemented edges and a node for `true` as the only constant node. To simplify drawing, you may assume that *dangling edges* point to the constant node. Write down all cofactors that you compute to obtain the final result and mark them in the graph.

3.2.7 Construct a ROBDD for the formula

$$f = (p \leftrightarrow q) \wedge (r \leftrightarrow s)$$

using *variable order* $r < s < p < q$. Use complemented edges and a node for `true` as the only constant node. To simplify drawing, you may assume that *dangling edges* point to the constant node. Write down all cofactors that you compute to obtain the final result and mark them in the graph.

3.2.8 (a) Construct a Reduced Ordered Binary Decision Diagram (ROBDD) for the formula

$$f = (a \vee b \vee c) \wedge \neg d$$

using *variable order* $a < b < c < d$. Use complemented edges and a node for `true` as the only constant node. To simplify drawing, you may assume that *dangling edges* point to the constant node. Write down all cofactors that you compute to obtain the final result and mark them in the graph.

(b) Construct a Reduced Ordered Binary Decision Diagram (ROBDD) for f with a different variable order. The ROBDD should result in a *smaller* ROBDD, w.r.t. the number of nodes.

3.2.9 Construct the reduced and ordered BDD for the formula

$$f = ((a \wedge b) \vee \neg a \vee (c \leftrightarrow d)$$

using *alphabetic variable order*. Compute the needed cofactors. You may add function nodes representing all cofactors to the final BDD. Use complemented edges and one terminal node representing the truth value **T**. To simplify drawing, you may assume that dangling edges point to the constant node.

3.2.10 Construct the reduced and ordered BDD for the formula

$$f = (r \wedge p) \vee (\neg r \wedge \neg p) \vee (s \wedge \neg r) \vee (\neg s \wedge r) \vee (\neg r \wedge q)$$

using *variable order* $p < q < r < s$. Compute the needed cofactors. You may add function nodes representing all cofactors to the final BDD. Use complemented edges and one terminal node representing the truth value **T**. To simplify drawing, you may assume that dangling edges point to the constant node.

3.2.11 Construct the reduced and ordered BDD for the formula

$$f = (r \wedge \neg p) \vee (\neg r \wedge p) \vee (s \wedge \neg r) \vee (\neg s \wedge r) \vee (r \wedge q)$$

using *variable order* $p < q < r < s$. Compute the needed cofactors. You may add function nodes representing all cofactors to the final BDD. Use complemented edges and one terminal node representing the truth value **T**. To simplify drawing, you may assume that dangling edges point to the constant node.

3.2.12 Construct a Reduced Ordered Binary Decision Diagram (ROBDD) for the formula

$$f = (p \wedge q) \vee (r \wedge s) \vee (\neg p \wedge \neg r)$$

using *alphabetic variable order*. Use complemented edges and a node for `true` as the only constant node. To simplify drawing, you may assume that *dangling edges* point to the constant node. Write down all cofactors that you compute to obtain the final result and mark them in the graph.

3.2.13 Construct a Reduced Ordered Binary Decision Diagram (ROBDD) for the formula

$$f = (\neg p \wedge q \wedge r) \vee (p \wedge \neg s)$$

using *alphabetic variable order*. Use complemented edges and a node for `true` as the only constant node. To simplify drawing, you may assume that *dangling edges* point to the constant node. Write down all cofactors that you compute to obtain the final result and mark them in the graph.

3.2.14 Construct a Reduced Ordered Binary Decision Diagram (ROBDD) for the formula

$$f = (p \wedge q \wedge \neg r) \vee (\neg q \wedge s) \vee (\neg p \wedge \neg s)$$

using *alphabetic variable order*. Use complemented edges and a node for `true` as the only constant node. To simplify drawing, you may assume that *dangling edges* point to the constant node. Write down all cofactors that you compute to obtain the final result and mark them in the graph.

3.2.15 Construct a Reduced Ordered Binary Decision Diagram (ROBDD) for the formula

$$f = (p \wedge q) \vee (r \wedge s) \vee (\neg p \wedge \neg r)$$

using *reverse alphabetic variable order*. Use complemented edges and a node for `true` as the only constant node. To simplify drawing, you may assume that *dangling edges* point to the constant node. Write down all cofactors that you compute to obtain the final result and mark them in the graph.

3.2.16 [3 points] Construct a reduced ordered binary decision diagram (ROBDD) for the formula

$$f = (p \vee q) \wedge \neg(p \wedge q) \wedge r$$

using *variable order* $q < p < r$. Use complemented edges and a node for `true` as the only constant node. To simplify drawing, you may assume that *dangling edges* point to the constant node. Write down all cofactors that you compute to obtain the final result and mark them in the graph.

3.2.17 [3 points] Construct a reduced ordered binary decision diagram (ROBDD) for the formula

$$(a \vee \neg b) \wedge \neg(c \vee d) \vee (a \wedge b),$$

using *variable order* $a < b < c < d$. Use complemented edges and a node for `true` as the only constant node. To simplify drawing, you may assume that *dangling edges* point to the constant node. Write down all cofactors that you compute to obtain the final result and mark them in the graph.