06-35324 *Mathematical and Logical Foundations of Computer Science* University of Birmingham
Autumn Semester 2021 School of Computer Science

Solutions to Exercise Sheet 4

Exercise 4.1

$$A \cup B = \{-1,0,1,2,4\} \quad A \cap B = \{0\} \quad A \setminus B = \{-1,1\}$$
$$A \times B = \{(-1,0),(-1,2),(-1,4),(0,0),(0,2),(0,4),(1,0),(1,2),(1,4)\}$$

Exercise 4.2

This is the same as the set of perfect cubes below 100; there are five of them: $\{0,1,8,27,64\}$

Exercise 4.3

To a subset A of $\mathbb{N}$ we associate the infinite list whose i-th entry is 1 if $i \in A$ and 0 otherwise. It is clear that from a given 0-1-list one can reconstruct the corresponding subset.

Exercise 4.4

(a) There are 2^{32} many possible bit patterns in a register, and they are all elements of this Boolean algebra.

(b)
$$\mathbf{x} \wedge \mathbf{y} = 0000\,0000\,0000\,0000\,0000\,0000\,0000\,0000$$
$$\mathbf{x} \vee \mathbf{y} = 1111\,1111\,1111\,1111\,1111\,1111\,1111\,1111$$
$$\overline{\mathbf{x}} = \mathbf{y} = 0000\,0001\,0010\,0011\,0100\,0101\,0110\,0111$$

(c) The laws hold for each of the 32 positions in the bit vectors, since the operations don't mix the contents of different positions, and since in each position we simply have the operations of the basic Boolean algebra of truth values. Since the laws hold for that two-element Boolean algebra, they also hold for the whole vectors. For example:

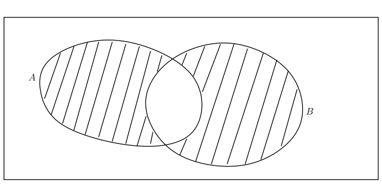
$$\mathbf{x} \wedge \mathbf{y} = (x_{31} \wedge y_{31})\ (x_{30} \wedge y_{30})\ (\ldots\ x_1 \wedge y_1)\ (x_0 \wedge y_0) = (y_{31} \wedge x_{31})\ (y_{30} \wedge x_{30})\ (\ldots\ y_1 \wedge x_1)\ (y_0 \wedge x_0) = \mathbf{y} \wedge \mathbf{x}$$

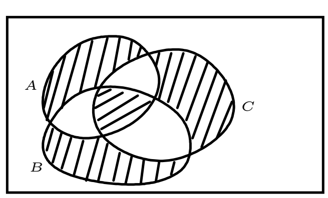
(d) Java uses & for $\wedge$, | for $\vee$, and $\sim$ for $\overline{}$.

(e) Here is one use: To test whether a number is even, you can perform logical "and" with the number 1. If the result is 0, then the number was even, if it is 1, then the number was odd.

The logical operations also exist for `char` and can be used for converting uppercase characters into lowercase ones, and vice versa.

Exercise 4.5





The operation $\triangle$ corresponds to "exclusive or."

Exercise 4.6

(a) $A \bigtriangleup B = (A \wedge \neg B) \vee (\neg A \wedge B)$

(b) We use the laws of Boolean algebras:

$$\begin{array}{rcll}
(A\wedge C)\bigtriangleup(B\wedge C) & = & ((A\wedge C)\wedge\neg(B\wedge C))\vee(\neg(A\wedge C)\wedge(B\wedge C)) & \text{the definition of } \bigtriangleup \\
& = & ((A\wedge C)\wedge(\neg B\vee\neg C))\vee((\neg A\vee\neg C)\wedge(B\wedge C)) & \text{de Morgan} \\
& = & ((A\wedge C)\wedge\neg B)\vee((A\wedge C)\wedge\neg C))\vee((\neg A\wedge(B\wedge C))\vee(\neg C\wedge(B\wedge C)) & \text{distributivity} \\
& = & ((A\wedge C)\wedge\neg B)\vee(A\wedge(C\wedge\neg C))\vee((\neg A\wedge(B\wedge C))\vee((\neg C\wedge C)\wedge B) & \text{associativity and communtativity} \\
& = & ((A\wedge C)\wedge\neg B)\vee(A\wedge\mathsf{false})\vee((\neg A\wedge(B\wedge C))\vee(\mathsf{false}\wedge B) & \text{complements} \\
& = & ((A\wedge C)\wedge\neg B)\vee\mathsf{false}\vee((\neg A\wedge(B\wedge C))\vee\mathsf{false} & \text{annihilation} \\
& = & ((A\wedge C)\wedge\neg B)\vee((\neg A\wedge(B\wedge C)) & \text{neutral element} \\
& = & ((A\wedge\neg B)\wedge C)\vee((\neg A\wedge B)\wedge C) & \text{associativity and commutativity} \\
& = & ((A\wedge\neg B)\vee(\neg A\wedge B))\wedge C & \text{distributivity} \\
& = & (A\bigtriangleup B)\wedge C & \text{definition of } \bigtriangleup
\end{array}$$

(c) We test the equation in the minimal Boolean algebra $\{\mathsf{false},\mathsf{true}\}$ where we let $A = \mathsf{false}$ and $B = C = \mathsf{true}$. Then the left hand side evaluates to

$$(A\wedge B)\bigtriangleup C = (\mathsf{false}\wedge\mathsf{true})\bigtriangleup\mathsf{true} = \mathsf{false}\bigtriangleup\mathsf{true} = \mathsf{true}$$

while the right hand side evaluates to

$$(A\bigtriangleup C)\wedge(B\bigtriangleup C) = (\mathsf{false}\bigtriangleup\mathsf{true})\wedge(\mathsf{true}\bigtriangleup\mathsf{true}) = \mathsf{true}\wedge\mathsf{false} = \mathsf{false}$$

(It can be shown that an equation that fails in any Boolean algebra must also fail in the minimal one.)

(d) For 0 take false and for 1 take true. The ring laws can now be checked by the indefatigable...