Home assignment #2

Uladzislau Bohdan / Владислав Богдан

Task 1

To compute all formal concepts.

The formal context is given to us using a table. Airlines are objects; continents and countries are attributes.

The following code is to compute all formal concepts and to output a number of them:

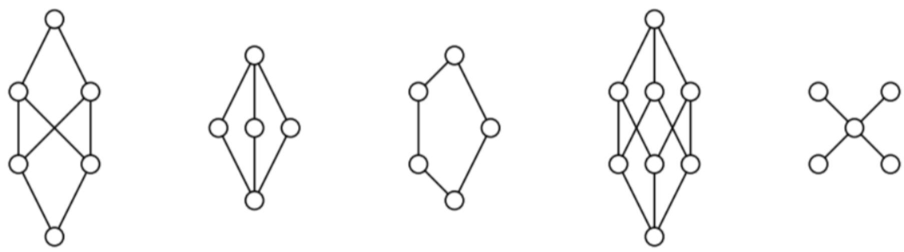
In [1]:

```
import numpy as np
import itertools

data = np.array([[1, 1, 1, 1, 1, 0, 1, 1, 1],
                 [0, 1, 0, 1, 0, 0, 0, 0, 1],
                 [0, 1, 0, 1, 0, 0, 0, 0, 1],
                 [0, 0, 0, 1, 0, 0, 0, 0, 0],
                 [0, 1, 1, 1, 1, 1, 0, 0, 1],
                 [0, 1, 0, 0, 0, 0, 0, 0, 0],
                 [1, 1, 1, 1, 1, 1, 1, 0, 1],
                 [1, 0, 1, 0, 0, 0, 1, 1, 1],
                 [1, 1, 0, 1, 0, 1, 0, 0, 1],
                 [0, 1, 1, 1, 1, 1, 0, 0, 1],
                 [1, 1, 0, 1, 0, 0, 0, 1, 1],
                 [1, 1, 1, 1, 0, 0, 1, 1, 1],
                 [1, 1, 0, 1, 0, 1, 1, 0, 1]])

n = data.shape[0]

m = data.shape[1]
all_attribute_subsets = list(map(list, itertools.product([0, 1], repeat=m)))

formal_concepts = []

def check_subsets(attribute_subset):
    a = np.ones(n, dtype=int)
    for j in range(m):
        if attribute_subset[j] == 1:
            a = np.logical_and(a, data.transpose()[j])

    object_subset = a

    b = np.ones(m)
    for i in range(n):
        if object_subset[i] == 1:
            b = np.logical_and(b, data[i])

    if not np.array_equal(object_subset, a) or not np.array_equal(attribute_subs
et, b):
        return

    formal_concepts.append([object_subset.astype(int), np.asarray(attribute_subs
et, dtype=int)])

for attribute_subset in all_attribute_subsets:
    check_subsets(attribute_subset)

print("Number of formal concepts found: ", len(formal_concepts))
print("The formal concepts found are: ")
for fc in formal_concepts:
    print("  objects: %s  attr: %s" % (fc[0], fc[1]))
```

```
Number of formal concepts found:  26
The formal concepts found are:
  objects: [1 1 1 1 1 1 1 1 1 1 1 1 1]  attr: [0 0 0 0 0 0 0 0 0]
  objects: [1 1 1 0 1 0 1 1 1 1 1 1 1]  attr: [0 0 0 0 0 0 0 0 1]
  objects: [1 1 1 1 1 0 1 0 1 1 1 1 1]  attr: [0 0 0 1 0 0 0 0 0]
  objects: [1 0 0 0 1 0 1 1 0 1 0 1 0]  attr: [0 0 1 0 0 0 0 0 1]
  objects: [1 1 1 0 1 1 1 0 1 1 1 1 1]  attr: [0 1 0 0 0 0 0 0 0]
  objects: [1 1 1 0 1 0 1 0 1 1 1 1 1]  attr: [0 1 0 1 0 0 0 0 1]
  objects: [0 0 0 0 1 0 1 0 1 1 0 0 1]  attr: [0 1 0 1 0 1 0 0 1]
  objects: [1 0 0 0 1 0 1 0 0 1 0 1 0]  attr: [0 1 1 1 0 0 0 0 1]
  objects: [1 0 0 0 1 0 1 0 0 1 0 0 0]  attr: [0 1 1 1 1 0 0 0 1]
  objects: [0 0 0 0 1 0 1 0 0 1 0 0 0]  attr: [0 1 1 1 1 1 0 0 1]
  objects: [1 0 0 0 0 0 1 1 1 0 1 1 1]  attr: [1 0 0 0 0 0 0 0 1]
  objects: [1 0 0 0 0 0 0 1 0 0 1 1 0]  attr: [1 0 0 0 0 0 0 1 1]
  objects: [1 0 0 0 0 0 1 1 0 0 0 1 1]  attr: [1 0 0 0 0 0 1 0 1]
  objects: [1 0 0 0 0 0 1 1 0 0 0 1 0]  attr: [1 0 1 0 0 0 1 0 1]
  objects: [1 0 0 0 0 0 0 1 0 0 0 1 0]  attr: [1 0 1 0 0 0 1 1 1]
  objects: [1 0 0 0 0 0 1 0 1 0 1 1 1]  attr: [1 1 0 1 0 0 0 0 1]
  objects: [1 0 0 0 0 0 0 0 0 0 1 1 0]  attr: [1 1 0 1 0 0 0 1 1]
  objects: [1 0 0 0 0 0 1 0 0 0 0 1 1]  attr: [1 1 0 1 0 0 1 0 1]
  objects: [0 0 0 0 0 0 1 0 1 0 0 0 1]  attr: [1 1 0 1 0 1 0 0 1]
  objects: [0 0 0 0 0 0 1 0 0 0 0 0 1]  attr: [1 1 0 1 0 1 1 0 1]
  objects: [1 0 0 0 0 0 1 0 0 0 0 1 0]  attr: [1 1 1 1 0 0 1 0 1]
  objects: [1 0 0 0 0 0 0 0 0 0 0 1 0]  attr: [1 1 1 1 0 0 1 1 1]
  objects: [1 0 0 0 0 0 1 0 0 0 0 0 0]  attr: [1 1 1 1 1 0 1 0 1]
  objects: [1 0 0 0 0 0 0 0 0 0 0 0 0]  attr: [1 1 1 1 1 0 1 1 1]
  objects: [0 0 0 0 0 0 1 0 0 0 0 0 0]  attr: [1 1 1 1 1 1 1 0 1]
  objects: [0 0 0 0 0 0 0 0 0 0 0 0 0]  attr: [1 1 1 1 1 1 1 1 1]
```

Task 2

Which are the lattices? Are there any complete lattices?



(i) (ii) (iii) (iv) (v)

An ordered set is a lattice if any pair of elements has infimum and supremum. This is true for samples (ii), (iii) and (iv) (this is easy to check due to a small size of the sets).

However, it is not true for two other sets.

(i). Let's enumerate vertices from top to bottom, left to right, and consider subset {4, 5}. Upper bound for {4, 5} is {1, 2, 3} then, but because {1, 2, 3} has no minimum (2 and 3 are incomparable) - there's no supremum. Therefore, (i) is not a lattice.

(v). Let's enumerate vertices in the same way again: top to bottom, left to right. Two upmost vertices (1, 2) are incomparable between each other and there's no supremum for a subset {1, 2} - drawing is not a lattice.

All sets which are lattices: (ii) - (iv), are also complete lattices because they are finite.

Task 3

$(L, \leq)$ - a lattice with supremum and infimum defined. $x, y \in L$

a) $x \vee x = sup(x, x) =$ [sup is the smallest value of an upper bound for {x}, which is a set of elements which are greater or equal than $x \Rightarrow$ sup of an element equals to the element] $= x$

b) $x \vee (x \wedge y) = sup(x, inf(x, y)) =$ [$inf(x, y) \leq x$ by definition of infimum, then by defition of sup] $= x$

c) The definition of sup on a set has nothing to do with the order of the element; therefore for two-element set (x, y) operation is commutative: $x \vee y = sup(x, y) = sup(y, x) = y \vee x$.