6.4:

a. Rectilinear floor-planning

Let R represent a large rectangle, S represent the smaller rectangles and. R has two variables (W)idth and (H)eight and S has (x) position, (y) position, (h)eight, and (w)idth. Different instances of the same type of rectangles will be denoted by a number as a subscript IE are two different small rectangles.

The constraints are as follows:

≥ 0: x position of the smaller rectangles is non-negative

≥ 0: y position of the smaller rectangles is non-negative

: the smaller rectangles fit in the larger rectangle, on the x axis

: the smaller rectangles fit in the larger rectangle on the y axis

: rectangle 1 does not overlap with rectangle 2, on the x axis

: rectangle 1 does not overlap with rectangle 2, on the y axis

: rectangle 2 does not overlap with rectangle 1, on the x axis

: rectangle 2 does not overlap with rectangle 1, on the y axis

b. Class scheduling

According to the book we have four different variables that we need to consider. We have (T)eacher, (C)lassroom, Course , and Time. We will denote a specific instance of a course as the letter ‘a’ or the letter ‘b’ and a specific time as the letter ‘t’.

Our constraints are as follows:

the teacher only teaches one classes during a specific time slot

: Two different teachers aren’t using the same classroom at the same time

c. Hamiltonian tour

Each city should be represented as the letter c with a subscript, i.e for each city we represent

the connected cities as the subscript ‘a’ so if we wanted to view the next possible cities after

we can use . To keep track of our visited cities we’ll use a set denoted by the letter ‘v’. If we restrict the set of connected cities of every city node by removing the cities listed in the visited node then we can potentially come up with a path with no repeated cities. We can tell if all cities are visited if the amount of cities that is in our visited set is equal to the amount of total city nodes.

6.11 AC-3 algorithm for detecting partial assignment inconsistencies given {WA = green, V = red}

If we start at the SA node we can remove the WA/SA constraint and V/SA constraint and remove green and red from the SA domain. That leaves blue as the only possible color. Next we can remove the SA/NT and the WA/NT constraint and remove blue and green from the domain of NT which means red is the only possible color for NT. Now we can remove the WA/Q constraint and the SA/Q restraint and remove red and blue from the domain of Q which means green is the only possible color for Q.

Then finally we can remove the Q/NSW constraint, the SA/NSW constraint, and the V/NSW constrain and remove red, blue, and green from the domain of NSW. That leaves NSW with an empty domain which means there was an inconsistency.