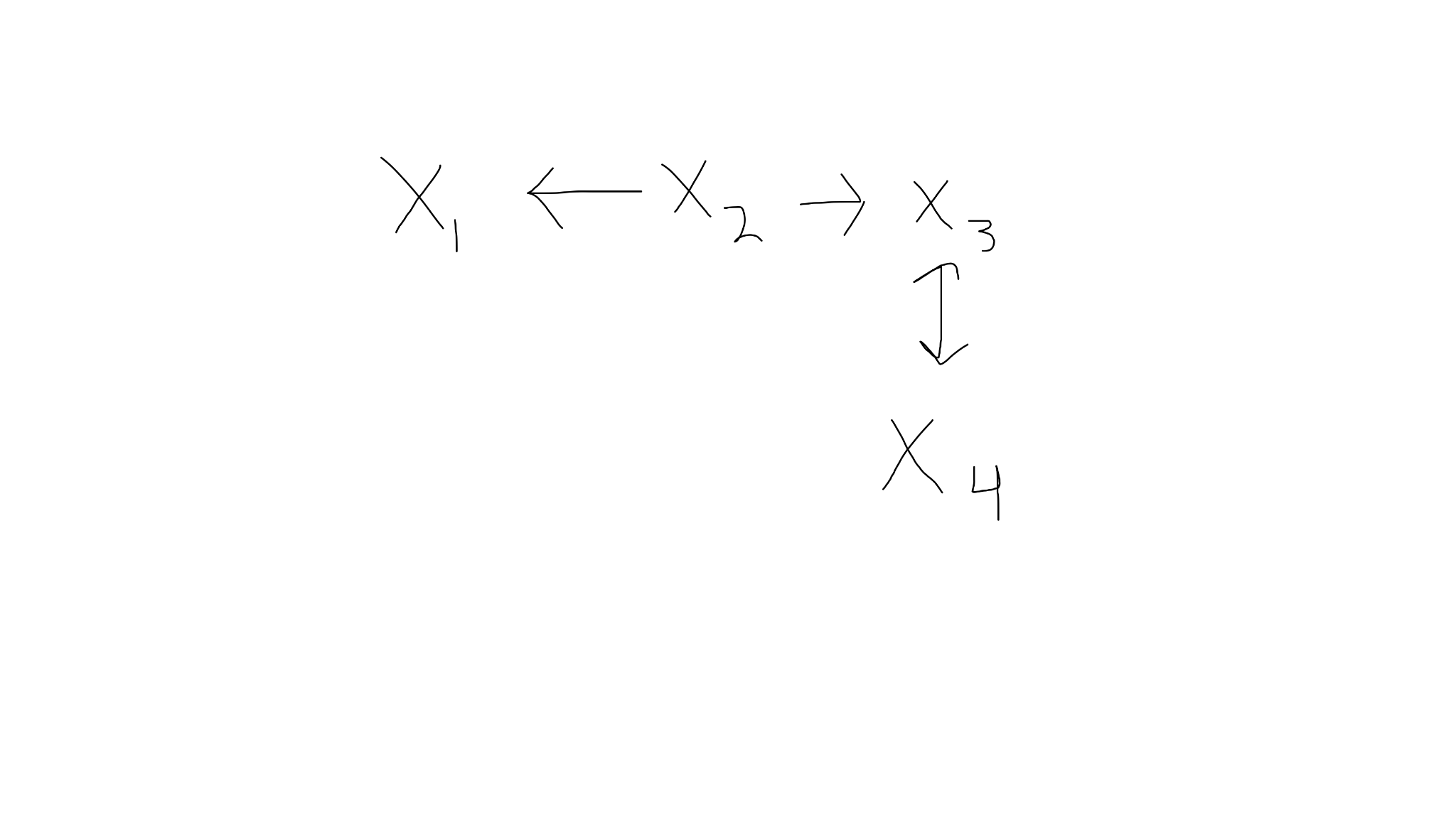
### **Constraint Graph:**

The constraint graph consists of four nodes representing X1 ,X2, X3 and X4​. The directed edges represent constraints:

* X1 ≥ X2​ (directed from X2​ to X1​)
* X2 > X3 (directed from X3​ to X2​)
* X3 ≠ X4​ (undirected between X3​ and X4)

The graph would look like this:



### **Applying AC-3 Algorithm:**

The initial queue contains arcs:

{(X2, X1),(X3, X2),(X3, X4),(X4, X3)}

Processing each arc:

**(X₂, X₁):**

* X1 ≥ X2​ means X1​ must be at least as large as X2​.
* Check domain pairs:
  + If X2 = 3, valid X1​ values: {3,4}
  + If X2 = 4, valid X1​ values: {4}
  + If X2 = 5,8,9, no valid X1​ values → **remove 5,8,9 from X2**
* Updated domains:
  + X1 = {3,4}
  + X2 = {3,4}

**(X₃, X₂):**

* X2 > X3 means X2​ must be strictly greater than X3​.
* Check domain pairs:
  + If X3 = 2, valid X2​ values: {3,4}
  + If X3 = 3, valid X2 values: {4}
  + If X3 = 5, 6, 7, 9, no valid X2​ values → **remove 5,6,7,9 from X3​**
* Updated domains:
  + X3 = {2,3}

**(X₃, X₄):**

* X3 ≠ X4​ means they cannot have the same value.
* Since X3 = {2,3} and X4={3,5,7,8,9}, → **remove 3 from X4​**
* Updated domains:
  + X4= {5,7,8,9}

### **Arc Consistency:**

The final domains are:

X1 = {3, 4}, X2 = {3, 4}, X3 = {2, 3}, X4 = {5, 7, 8, 9}

* X1 ≥ X2​ → Holds since both share {3,4}.
* X2 > X3 → Holds since X2 = {3,4} X2​ = {3,4} and X3 = {2,3}.
* X3 ≠ X4​ → Holds as X3 = {2,3} and X4​ doesn’t contain 2 or 3.

**Therefore, the network is arc-consistent.**

### **Finding a Valid Schedule:**

Assigning values satisfying all constraints:

* X1 = 4
* X2 = 3
* X3 = 2
* X4 = 5

All constraints are satisfied:

* X1 ≥ X2 → 4 ≥ 3 (VALID)
* X2 > X3​ → 3 > 2 (VALID)
* X3 ≠ X4​ → 2 ≠ 5 (VALID)

**Therefore, a valid schedule exists.**

### **Adding New Constraint X1 ≠ X4:**

The constraint X1 ≠ X4​ means that X1​ and X4​ cannot have the same value.

* Since X1 = {3,4} and X4 = {5,7,8,9}, there is no overlap, so no revision is needed.

**Therefore, the network remains arc-consistent.**