IDL (Integer Difference Logic)

# General instruction

Reduce the literals of the kind x - y ≤ n

Rewrite x ≤ n as x - 0 ≤ n

Rewrite x - y = n as x - y ≤ n ^ - x + y ≤ -n

Rewrite x - y ≰ n as x - y > n ⟹ x - y ≥ n+1 ⟹ y - x ≤ -n-1

Rewrite x ≰ n as x > n ⟹ x ≥ n + 1 ⟹ 0 - x ≤ - n - 1

Rewrite x - y ≠ n as x - y < n **v** x - y > n ⟹ x - y ≤ n - 1 **v** y - x ≤ -n - 1

!!! since in the last example there is an or there is a **case split** !!!

# How to graph it

In general x - y ≤ n can be graphed as y --(n)--> x (an arrow that goes from y to x with value n )

# Satisfiability

The constraint is satisfiable iff in such graph there are no negative weight cycle, that is if all the n sum up together are ≥ 0

A cycle is a path beginning and ending up to the same node

# Example 1

y ≤ 3 ^ z - y ≤ -2 ^ z ≮ - 5 ^ u - y ≤ -4 v - u = 1

Rewrite y ≤ 3 as y - 0 ≤ 3

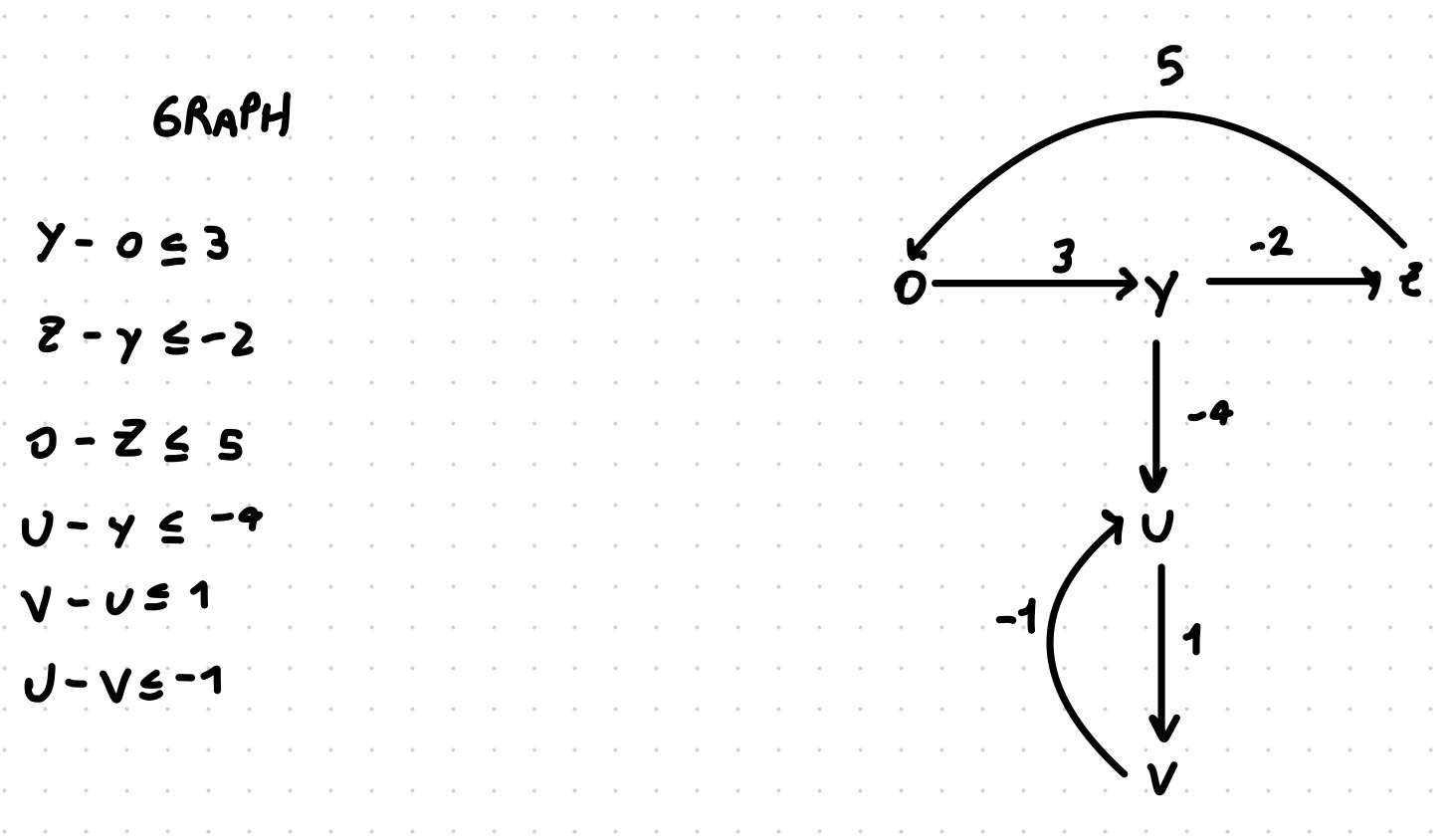
Rewrite z - y ≤ -2 as z - y ≤ -2

Rewrite z ≮ - 5 as 0 - z ≤ 5

Rewrite u - y ≤ -4 as u - y ≤ -4

Rewrite v - u = 1 as v - u ≤ 1 ^ u - v ≤ -1

## Graph



In this graph there are two cycle

The first cycle is composed by 0,y,z

The second cycle is composed by u,v

We need to check if both cycle are non-negative, that is

3 + (-2) + 5 ≥ 0 TRUE, the first cycle is non-negative

1 + (-1) ≥ 0 TRUE, the second cycle is non-negative

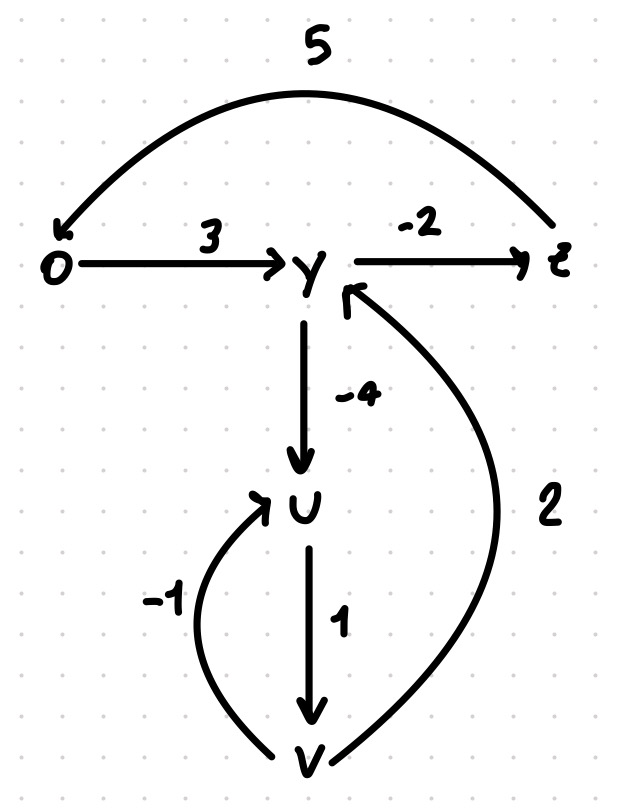
Since there is no negative cycle, that it is SAT

## Example 2

Like the example one with one more constraint y - v ≤ 2

## 

## Graph



Now there are three cycles, all cycles must be non negative, that is

5+3+(-2) ≥ 0 TRUE, so the first cycle is non negative

1+(-1) ≥ 0 TRUE, so the second cycle is non negative

(-4)+1+2 ≥ 0 FALSE, so the third cycle is NEGATIVE

Since there is a negative cycle, than it is UNSAT

≰ ≱