- Lecture 9

- Difference logic
- DPLL(T)
- Theories for E/EUF.

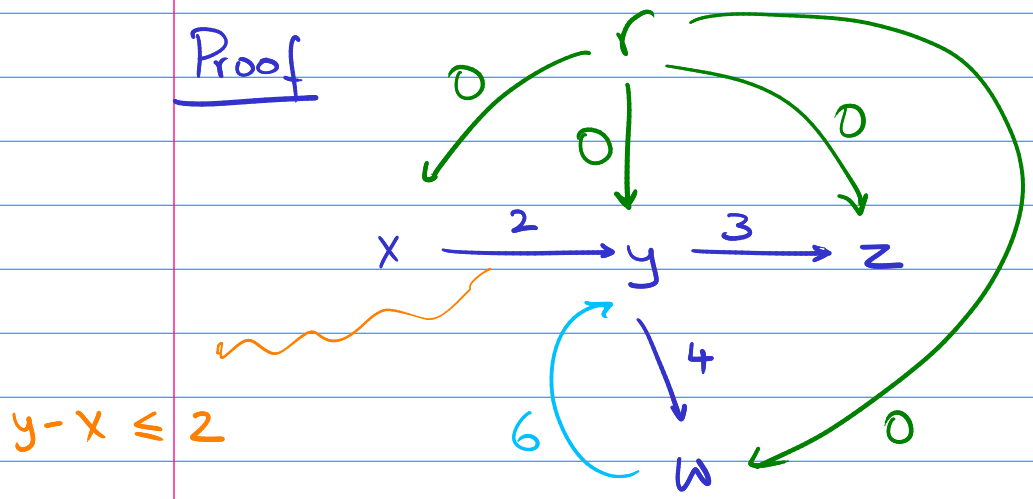
- LRA / LIA
- Difference
- Bit vectors, Strings
-

Claim: A set of difference logic constraints φ is satisfiable iff G_φ does not contain a negative weight cycle.

Part 1: If negative weight cycle exists, then a satisfying assignment cannot exist.

Part 2: If a negative weight cycle does not exist, then a satisfying assignment exists.

Proof



$w - y \leq 4$

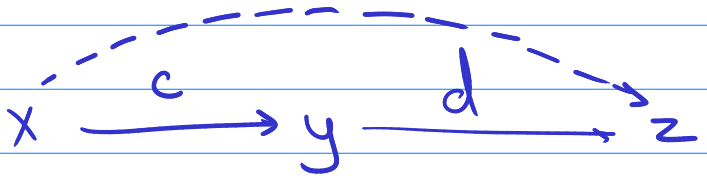
$y - w \leq 6$

$-6 \leq w - y$

$-6 \leq w - y \leq 4$

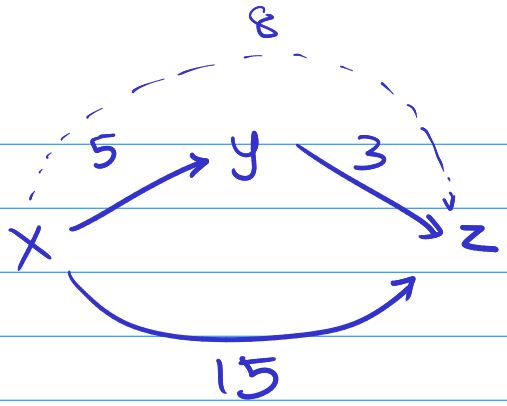
$x - r \leq 0$	$x \leq r$
$y - r \leq 0$	$y \leq r$
$z - r \leq 0$	$z \leq r$
$w - r \leq 0$	$w \leq r$

Why is introducing the root node r such a great idea? Because every node now has a shortest path (from r)



$$\frac{(y - x \leq c) \quad (z - y \leq d)}{z - x \leq c + d}$$

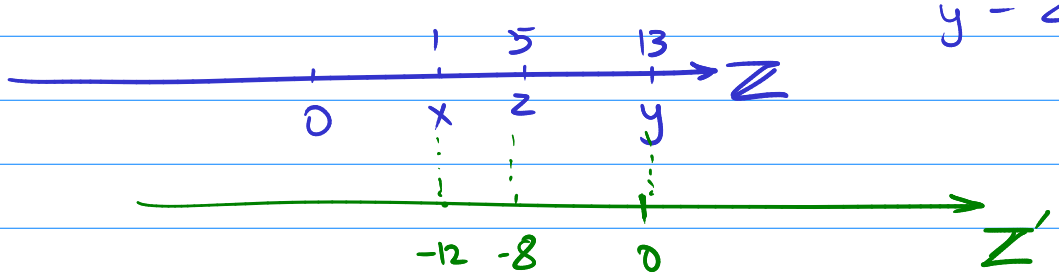
Adding constraints corresponds to walking through the graph.



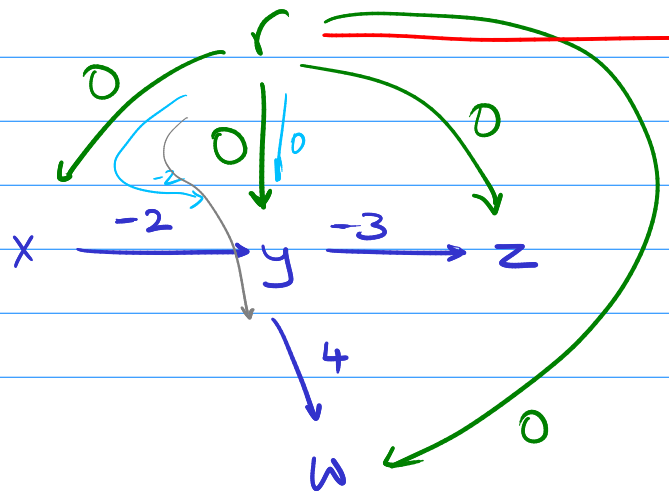
Shortest paths are the ultimate constraints that we should worry about.

$$\left.\begin{array}{l} y - x \le 5 \\ z - y \le 3 \end{array}\right] \; z - x \le 8$$
$$z - x \le 15$$

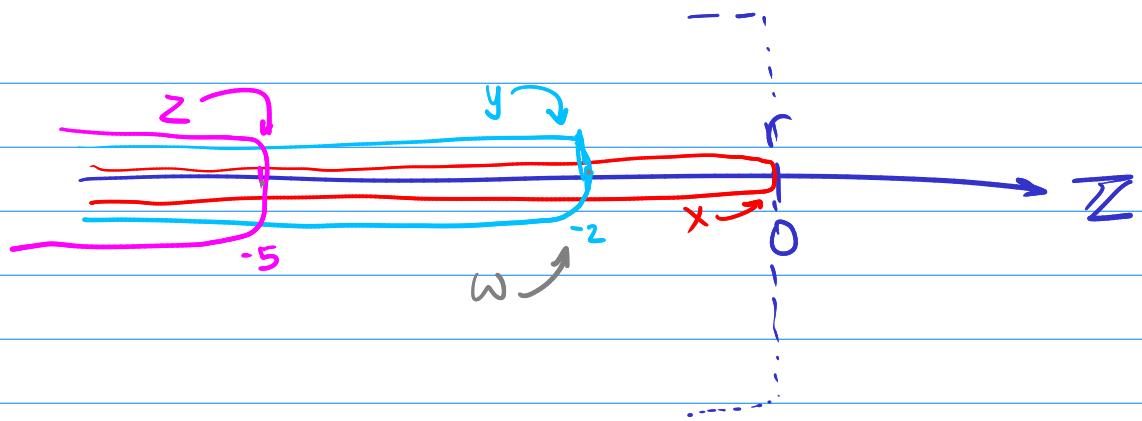
$z - x \le$ ___
$y - x \le$ ___
$y - z \le$ ___



Physically offsetting the number line preserves satisfying assignments.



Insight: Why not let this r be the origin of the number line?



x, y, z, w : all of them confined to live left of r.

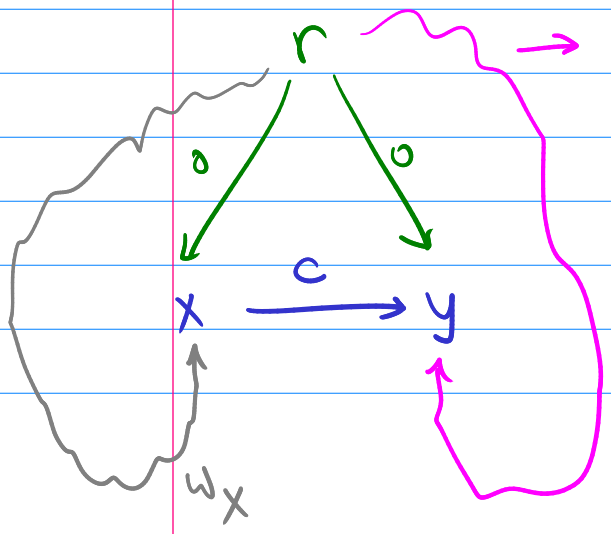
Idea. Let the variables live as much to the right as they possibly can.

$\{x \mapsto 0,\ y \mapsto -2,\ z \mapsto -5,\ w \mapsto -2\}$

Claim: This object is a satisfying assignment.

Proof: Assume not.

Say some constraint $y - x \leq c$ is violated



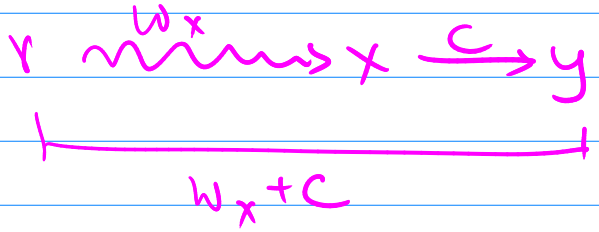
→ of all paths, this path has the shortest weight.

By assumption, $w_y - w_x \not\leq c$

$w_y \not\leq w_x + c$

$w_y > w_x + c$

i.e the weight of the shortest path could not have been w_y.



is a shorter path

Contradiction!

DPLL(T)

- Theory solvers only solve a conjunction of literals

$\underbrace{y - x \leq 3}_{a}$ and $\underbrace{z - y \leq 4}_{b}$ and ...

- SAT solvers can handle complex combinatorial reasoning

$$a \wedge (b \vee c) \wedge (\overline{b} \wedge \overline{c})$$

- Can we build a constraint solver where the propositions "mean something" in an underlying theory? ↑ interpreted

---

- DPLL(T) Assumes that all propositions mean something in the same theory.

- $(x - y \leq 2)$ or $(y - z \leq 3$ and $x - z \leq 4)$

Single theory

- $n \geq 4$ or $(w = \text{"abc"} \cdot w'$ and $n = |w'|)$

↑ Theory of LIA ↑ Theory of strings

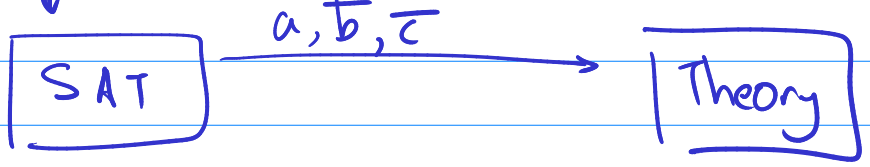
Multiple theories / Theory combination

Nelson Oppen Procedure

(SMT)

$\underbrace{(x-y \leq 2)}_{a}$ or $\underbrace{(y-z \leq 3}_{b}$ and $\underbrace{x-z \leq 4)}_{c}$

$a \vee (b \wedge c)$



$y - z \neq 3$

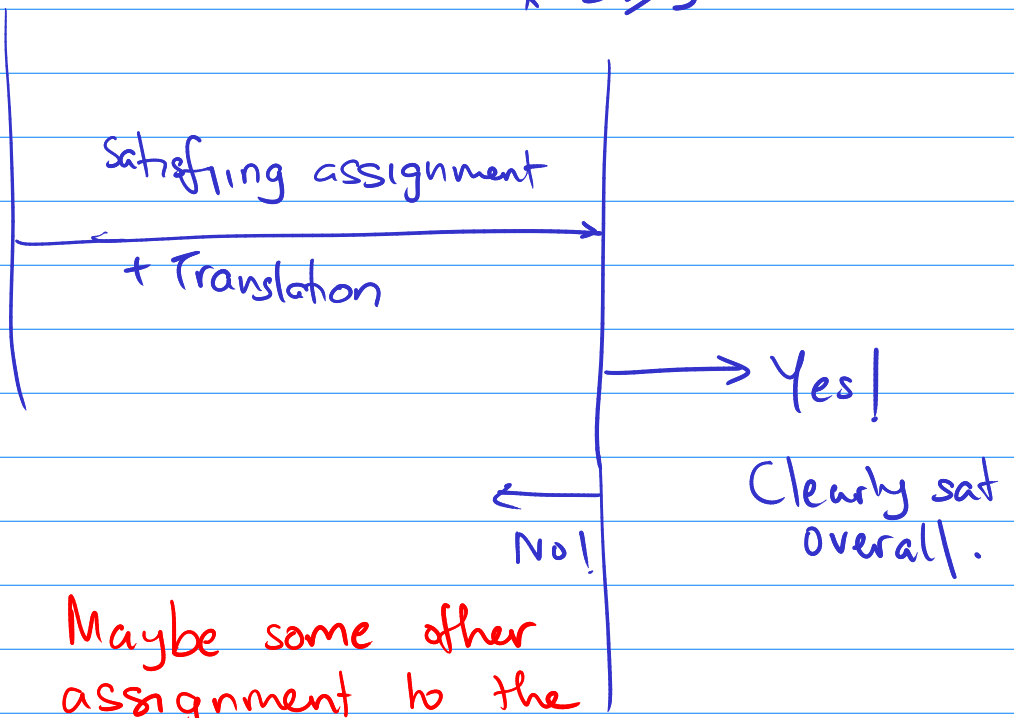
$y - z > 3$

$y - z > 4$

$x - y \leq 2$

$y - z \geq 4$

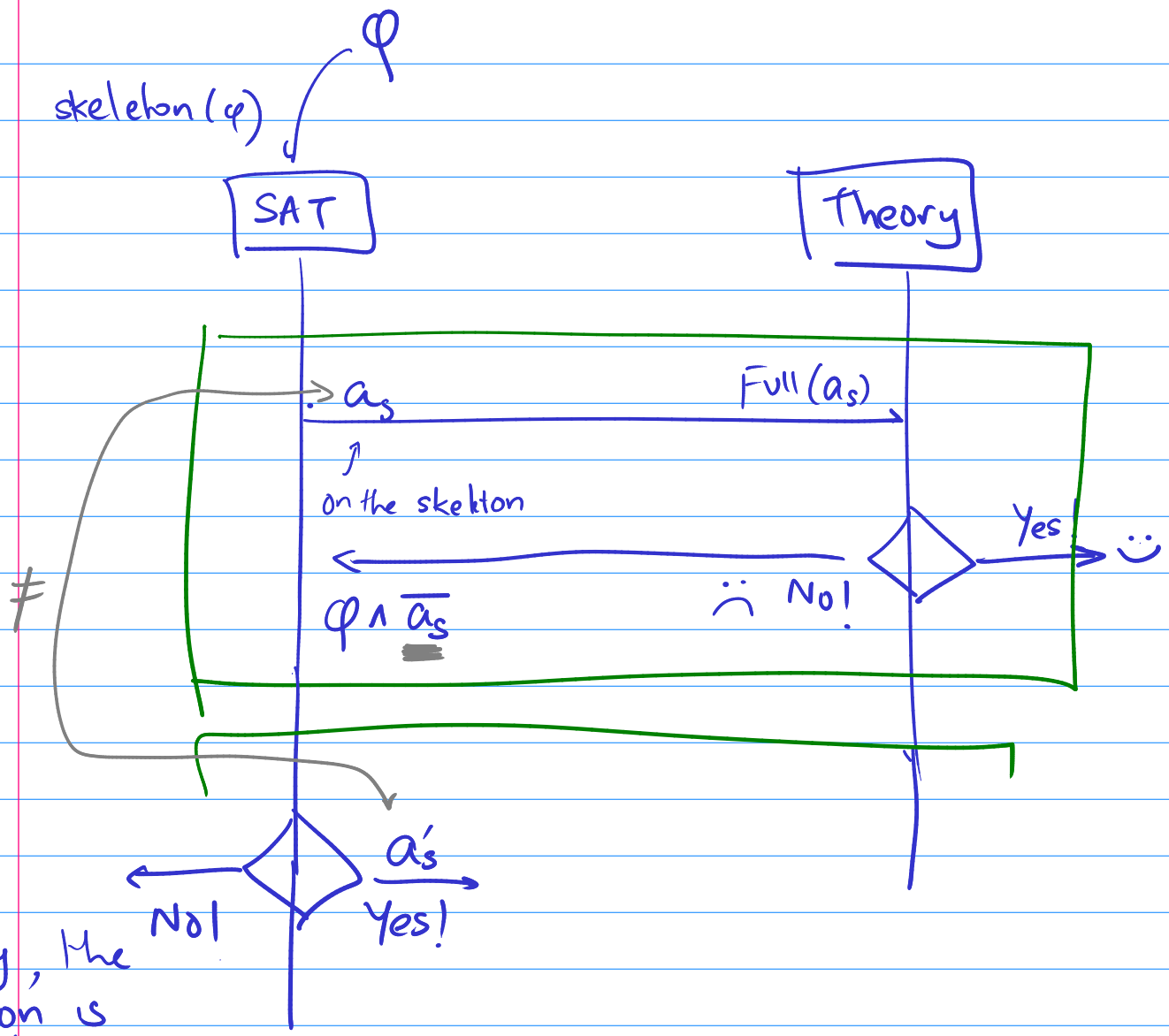
$x - z \geq 5$



Maybe some other assignment to the propositional skeleton works?

$\bar{a}\ b\ c$, $\bar{a}\ b\ \bar{c}$,

$a\ \bar{b}\ c$



Clearly, the situation is hopeless!

Overall UNSAT.

- Eventually, the SAT solver will run out of models.

- So the procedure is guaranteed to terminate

This procedure will eventually ~~do~~ say the right thing ∴

- How do we make this procedure fast?

Reason for previous procedure to be inefficient.

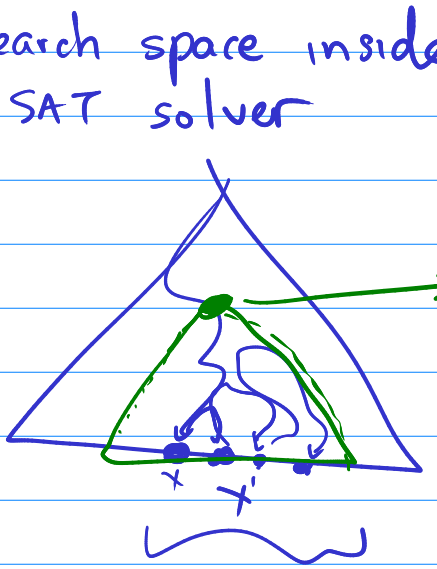
Too many models.

What can we do about it?

a - SAT solver which can return multiple models?

b - What if the SAT solver can do something more?

c - What if the theory solver could do something more?

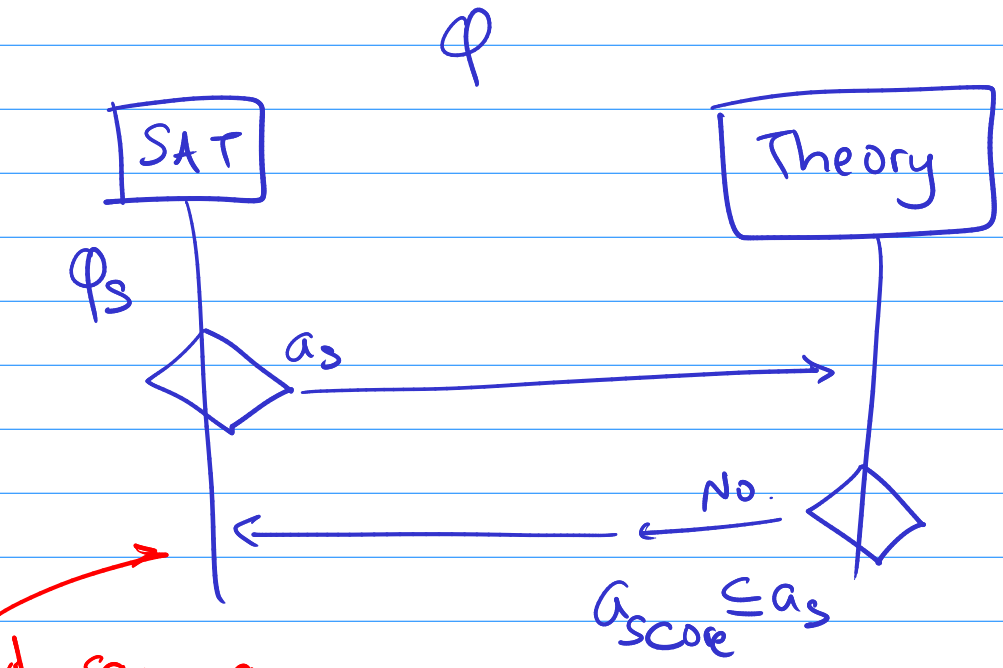


Question: Why not submit these to theory solver?

c. What more could the theory solver do?

It could provide unsat cores (minimal)

($x - y \leq 3$ and $x - y \geq 5$ and $w - x \leq 3$ and $u - w \leq 5$)



Instead of saying

$\varphi_s \wedge \overline{a_s}$

we can say $\varphi_s \wedge \overline{a_{score}}$

- Theory of Equality

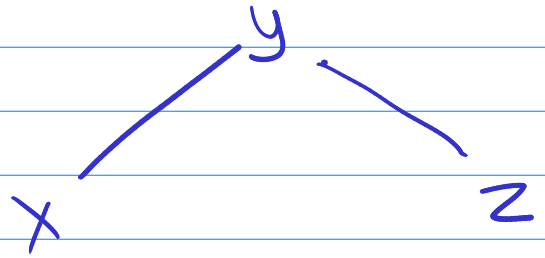
$\exists x\, y\, z \in \mathbb{Z} \;.\; x = y \;\; x \neq z \;\; y = z$

Disequality Constraint

Equality Constraint

Transitivity is your friend.

- Step 1: From all equality constraints, construct undirected graph.



- Step 2: For each disequality constraint

$x \neq y$

check if x & y are in the same SCC.

- If yes: formula is unsat

- Step 3: Formula is sat.