# Simulating logic formulas

Let $\varphi(x_1, \dots, x_n)$ be a formula on an automatic structure $\langle E, r_1, \dots, r_k \rangle$ where the only free variables the $x_i$ (first order variables)

We design an $\omega$-automaton $\mathcal{A}$ such that $\mathcal{L}(\mathcal{A})$ is $\mathcal{I}(\varphi)$: $\mathcal{I}(\varphi) = \{ (u_0^{(0)}, \dots, u_0^{(n-1)}) \dots (u_k^{(0)} \dots u_k^{(n-1)}) \dots \mid$

$u^{(0)} = \mathcal{I}(e_0)$
$\vdots$
$^{(n-1)} = \mathcal{I}(e_{n-1})$
$(e_0, \dots, e_{n-1})$ is true $\}$

Firstly, one changes $\varphi$:

- remove all functions turning them into predicate
- remove all universal quantifications ($\forall$) replacing them by a $\neg \exists \neg$
- change all "$\exists v$" by "$\exists v \; v \in E \wedge$"
- replace all implications $A \to B$ by $\neg A \vee B$

Secondly, one build the $\omega$-automaton recursively:

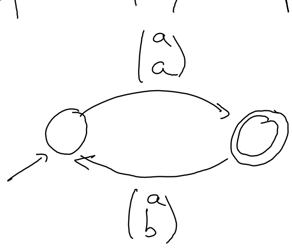
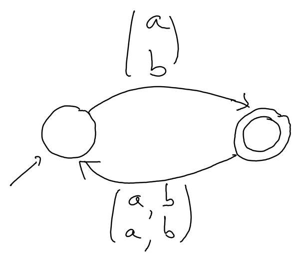
- If $\varphi$ is an atomic formula (hence a predicate) it corresponds to an $\omega$-automaton
- If $\varphi$ is $\varphi_1 \vee \varphi_2$ [resp $\varphi_1 \wedge \varphi_2$] construct the

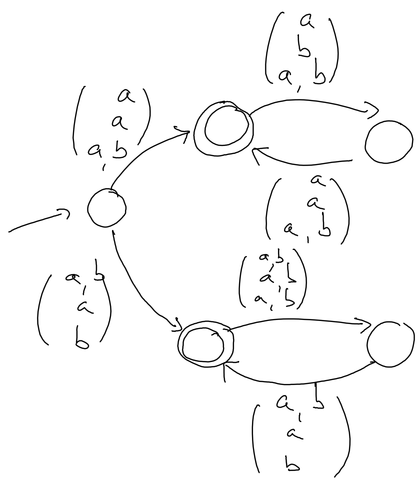
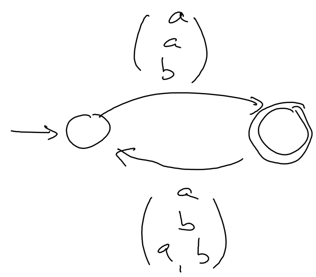
automaton which recognize the "union" [resp. "intersection"] of the automata for $\varphi_1$ and $\varphi_2$

⚠ beware alphabets: $\varphi_1(x,y) \vee \varphi_2(y,z)$

alphabet $= \Sigma^2$ alphabet $= \Sigma^2$

but for $\varphi$, alphabet $= \Sigma^3$

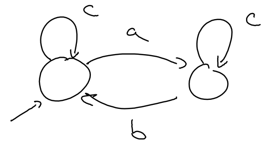
Ex $\varphi_1$:  ; $\varphi_2$ 

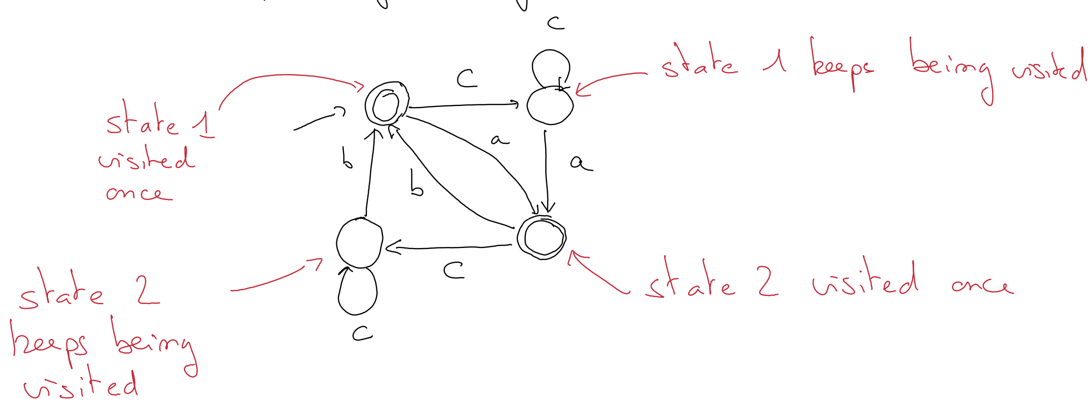
$\varphi_1 \vee \varphi_2$ :  ; $\varphi_1 \wedge \varphi_2$ 

- If $\varphi$ is $\neg \psi$, construct the $\omega$-automaton which recognizes the complement language
  - ↳ firstly turn the $\omega$-automaton into a Muller deterministic $\omega$-automaton (Safrah algorithm) ← the costly procedure
  - ↳ secondly, reverse the set of accepting set of states

↳ then, turn it back to an Büchi ω-automaton

[ guess the set of states which are visited infinitely many times and check it with the Büchi condition

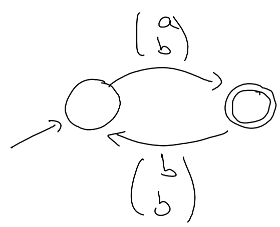
ex:  and both states infinitely many times visited :

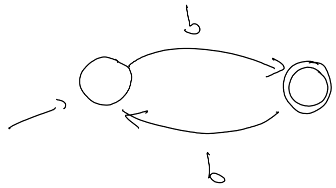



⟹ ensure cycle state 1 visited

state 2 visited

- If φ is $\exists x\ \varphi(x)$, erase the letter corresponding to x in the alphabet

Ex $\varphi(x, y)$: 

$\exists x\ \varphi(x, y)$ 

For close formulas, alphabet is unary. The formula is true if the language is not empty ($\Leftrightarrow$ full)

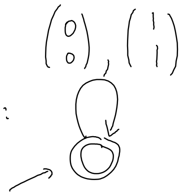
<u>Ex</u> : $E = \mathbb{N}$ $\quad$ $J =$ "lsb first binary representation"

$\sigma = \langle +, = \rangle$

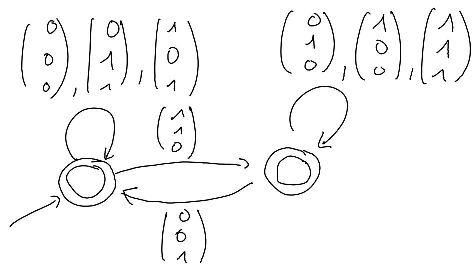
$\varphi = \forall x \ \exists y \ \exists z \quad x + y = z$

Transforming $\varphi$ : $\neg \exists x, x \in \mathbb{N} \wedge \neg \exists y \ y \in \mathbb{N} \wedge \exists z \ z \in \mathbb{N} \wedge \exists t \ t \in \mathbb{N}$
$\wedge \ Plus(x, y, t) \wedge t = z$

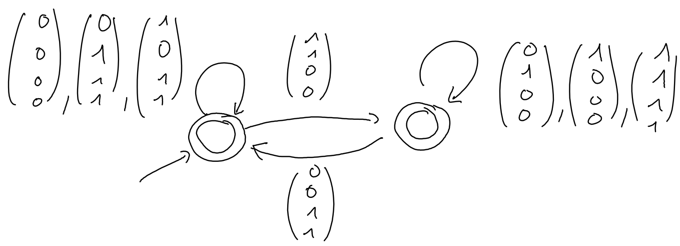
$t = z$ : $\begin{pmatrix} z \\ t \end{pmatrix}$



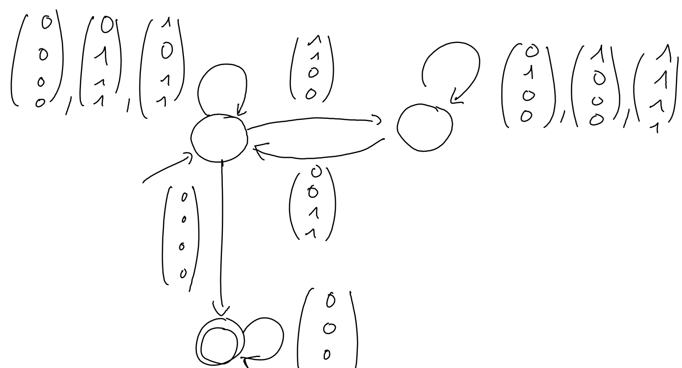
$Plus(x, y, t)$ : $\begin{pmatrix} x \\ y \\ t \end{pmatrix}$



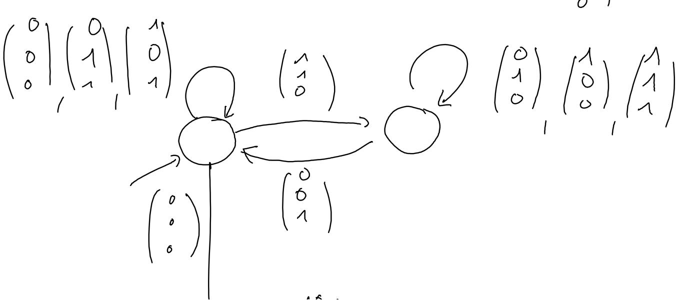
$t = z \wedge Plus(x, y, t)$ : $\begin{pmatrix} x \\ y \\ z \\ t \end{pmatrix}$



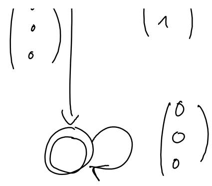
$t \in \mathbb{N} \wedge t = z \wedge Plus(x, y, t)$ : $\begin{pmatrix} x \\ y \\ z \\ t \end{pmatrix}$



$\exists t, t \in \mathbb{N} \wedge t = z \wedge \ldots$ : $\begin{pmatrix} x \\ y \\ z \end{pmatrix}$

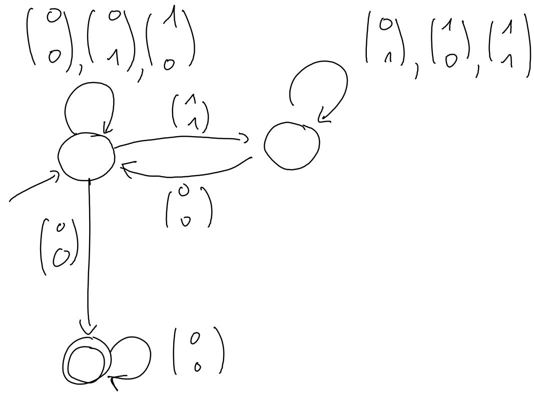


$z$



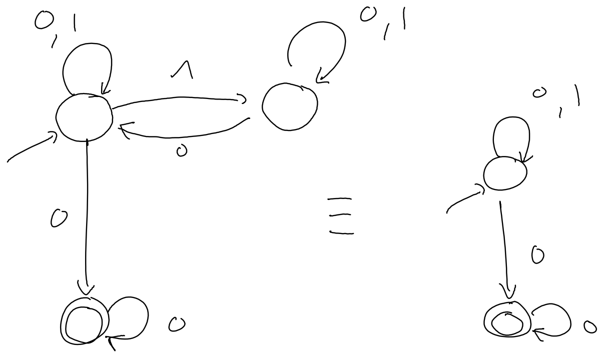
$\exists z\ z \in \mathbb{N} \wedge \exists t,\ t \in \mathbb{N} \wedge t = z \wedge \dots :$

$\begin{pmatrix} x \\ y \end{pmatrix}$



$\exists y\ \ y \in \mathbb{N} \wedge \exists z\ z \in \mathbb{N} \wedge \exists t \dots :$

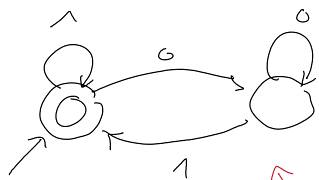
$(x)$



finitely many 1

$\neg \exists y\ \ y \in \mathbb{N} \wedge \exists z\ z \in \mathbb{N} \dots :$

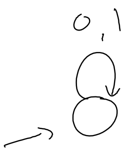
$(x)$



infinitely many 1

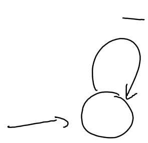
$x \in \mathbb{N} \wedge \neg \exists y \dots :$

$(x)$

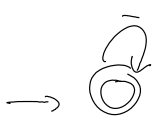


empty

$\exists x,\ x \in \mathbb{N} \wedge \dots :$



$\neg \exists x, \dots :$



full the formula is true

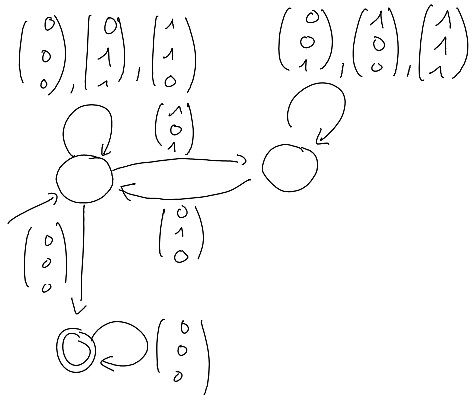
Ex2: $\varphi = \forall x\ \forall y\ \exists z\ \ x + z = y$

Transforming $\varphi$: $\neg \exists x\ x \in \mathbb{N} \wedge \exists y\ y \in \mathbb{N} \wedge \neg \exists z\ z \in \mathbb{N} \wedge \exists t\ t \in \mathbb{N}$

$\wedge\ Plus(x,z,t) \wedge y=t$

$\exists t\ t \in \mathbb{N} \wedge Plus(x,z,t) \wedge y=t$

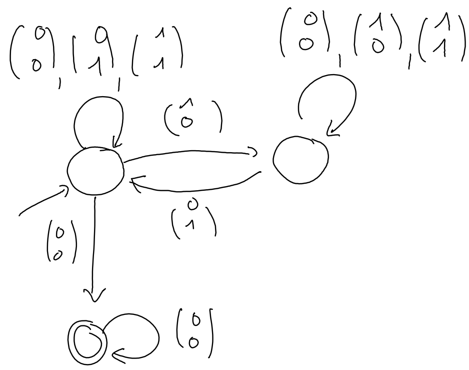
$$\begin{pmatrix} x \\ y \\ z \end{pmatrix}$$



$\exists z\ z \in \mathbb{N} \wedge \exists t\ t \in \mathbb{N} \wedge Plus(x,z,t) \wedge y=t$

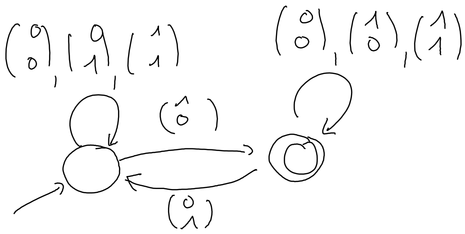
$$\begin{pmatrix} x \\ y \end{pmatrix}$$

basically $x \leq y$



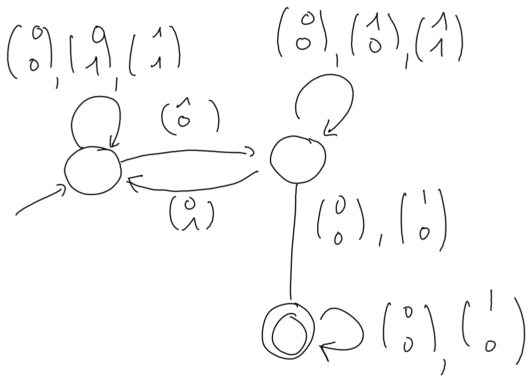
$\neg \exists z\ z \in \mathbb{N} \wedge \exists t\ t \in \mathbb{N} \wedge \ldots$

$$\begin{pmatrix} x \\ y \end{pmatrix}$$



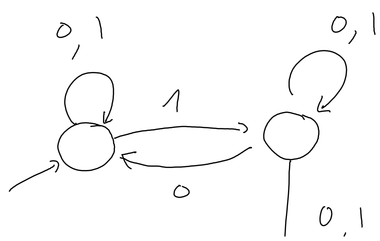
$y \in \mathbb{N} \wedge \neg \exists z\ z \in \mathbb{N} \wedge \ldots$

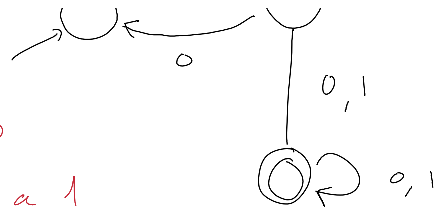
$$\begin{pmatrix} x \\ y \end{pmatrix}$$



$\exists y\ y \in \mathbb{N} \wedge \neg \exists z\ z \in \mathbb{N} \wedge \ldots$

$$\begin{pmatrix} x \end{pmatrix}$$

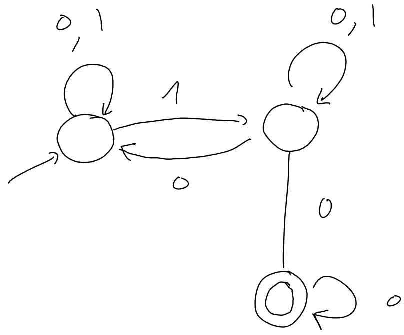




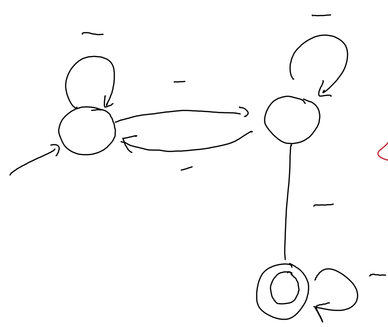

words with a 1

↳ only 0 is such that $\forall y \; \exists z \; 0+z=y$

$x \in \mathbb{N} \land \exists y \; y \in \mathbb{N} \land \ldots$
$(x)$

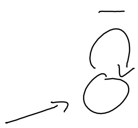



$\exists x, x \in \mathbb{N} \land \exists y \ldots$



full

$\varphi = \neg \exists x, x \in \mathbb{N} \land \ldots$



empty

The formula is false; the previous automata could give counter examples $(x \neq 0)$: this is called synthesis