2.1 (a) WMSO[<, suc] = WMSO[suc] because $S(w), \dot{I} \models x < y$ iff

$$S(w), \dot{I} \models \overbrace{\neg \forall X.\ X(x) \leftrightarrow X(y)}^{x \neq y}$$
$$\wedge\ \forall X.\ \underbrace{X(x) \wedge (\forall z \forall z'.\ X(z) \wedge suc(z,z') \rightarrow X(z'))}_{z' > x \text{ then } X(z')} \rightarrow X(y)$$

(b) WMSO[suc] = WMSO_0 because one can substitute every first-order variable $\underline{x}$ by a second-order variable $\underline{X_x}$ describing the singleton $\underline{\{x\}}$ to get an equivalent formula.

The following rules desribe the reduction more concretely:

$$\begin{aligned}
\lfloor P_a(x) \rfloor &:= X_x \subseteq P_a \\
\lfloor suc(x,y) \rfloor &:= SUC(X_x, X_y) \\
\lfloor X(x) \rfloor &:= X_x \subseteq X \\
\lfloor \neg\varphi \rfloor &:= \neg \lfloor \varphi \rfloor \\
\lfloor \varphi_1 \vee \varphi_2 \rfloor &:= \lfloor \varphi_1 \rfloor \vee \lfloor \varphi_2 \rfloor \\
\lfloor \exists x.\varphi \rfloor &:= \exists X_x.\ Sing(X_x) \wedge \lfloor \varphi \rfloor \\
\lfloor \exists X.\varphi \rfloor &:= \exists X.\ \lfloor \varphi \rfloor
\end{aligned}$$

E.g. $X(x) \wedge \forall z \forall z'.\ X(z) \wedge suc(z,z') \rightarrow X(z')$ is equivalent to $X_x \subseteq X \wedge \forall X_z \forall X_{z'}.\ X_z \subseteq X \wedge SUC(X_z, X_{z'}) \rightarrow X_{z'} \subseteq X$.

2.2. (a) The empty set has only itself as a subset, hence

$$\varphi(X) := \forall Y.\ Y \subseteq X \rightarrow X \subseteq Y$$

is a WMSO_0 formula which expresses that $X = \emptyset$.

(b) The singleton set has only itself and $\emptyset$ as subsets, hence

$$\psi(X) := \forall Z.\ Z \subseteq X \wedge \neg\varphi(Z) \longrightarrow X \subseteq Z$$

is a WMSO_0 formula which expresses that $|X| = 1$.

2.3. (a) The following WMSO[<, suc]-formula defines $\{w \in \{a,b\}^* \mid |w| \in 3\mathbb{N}\}$

$$\underset{\substack{\uparrow \\ 3\mathbb{N}+2 \\ \text{positions}}}{\exists X} .\ \underset{\substack{\nwarrow \\ \text{voidly true} \\ \text{for } \varepsilon}}{\forall x} .\ first(x) \to \underbrace{(\exists y \exists z .\ suc(x,y) \wedge suc(y,z) \wedge \neg X(x) \wedge \neg X(y) \wedge X(z))}_{\text{each word's position 2 is in } X \text{ (if word} \neq \varepsilon)}$$

$$\wedge \underbrace{(\exists t .\ last(t) \wedge X(t))}_{\text{each word's last position is in } X \text{ (if word} \neq \varepsilon)}$$

$$\underbrace{\begin{aligned} \wedge\ \forall x \forall y \forall z .\ suc(x,y) \wedge suc(y,z) \to (&X(x) \leftrightarrow \neg X(y) \wedge \neg X(z)) \\ \wedge\ (&X(z) \leftrightarrow \neg X(x) \wedge \neg X(y)) \end{aligned}}_{\text{every third position in the word belongs to } X}$$

(b) Let $\varphi_a(x) := \exists y \exists z .\ suc(x,y) \wedge suc(y,z) \wedge P_a(x) \wedge P_a(y) \wedge P_a(z)$

$\varphi_b(x) := \exists y \exists z .\ suc(x,y) \wedge suc(y,z) \wedge P_b(x) \wedge P_b(y) \wedge P_b(z)$.

Moreover, define

$$\varphi(X) := \forall x \forall y \forall z \forall t .\ X(x) \wedge suc(x,y) \wedge suc(y,z) \wedge suc(z,t) \to X(t) \wedge (\varphi_a(t) \vee \varphi_b(t)).$$

A WMSO[<, suc]-formula defining $\{aaa, bbb\}^*$ is then

$$\underset{\substack{\uparrow \\ 3\mathbb{N} \text{ positions}}}{\exists X} .\ (\forall x .\ \underbrace{first(x) \to X(x) \wedge (\varphi_a(x) \vee \varphi_b(x))}_{\text{base case}}) \wedge \underset{\substack{\nwarrow \\ \text{inductive principle}}}{\varphi(X)}$$

(c) The different parts of the formula are interpreted as:

$\forall x .\ X(x) \to P_a(x)$

↳ if x belongs to X then position x in the word contains a

$\exists x \exists y .\ x < y \wedge X(x) \wedge X(y)$

↳ there are two positions x and y in X

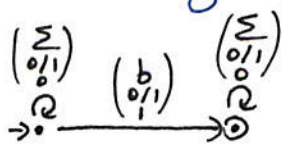
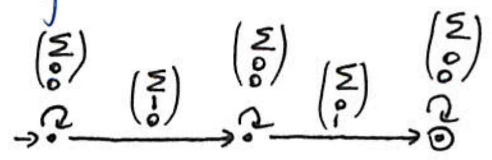
$\forall x \forall y \forall z .\ X(x) \wedge X(y) \wedge x < z \wedge z < y \to X(z)$

↳ if x and y are in X then every z inbetween belongs to X

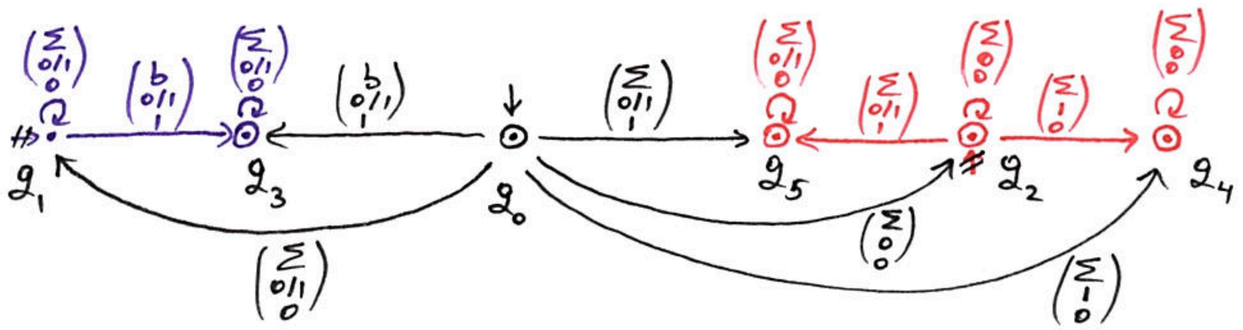
The language described by the given formula then is

$$(\Sigma \setminus \{a\})^* a a^* a (\Sigma \setminus \{a\})^*.$$

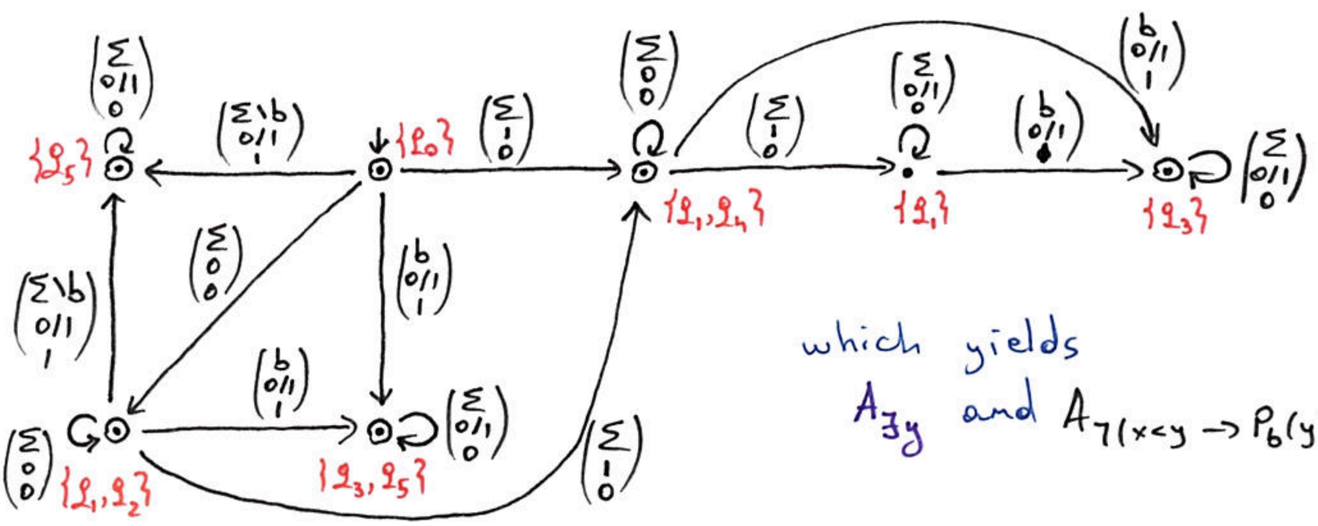
2.4. The regular representation of the language defined by φ is $(\Sigma\setminus a)^*(ab^*+\varepsilon)$. Using the construction from class we have

$A_{P_b(y)}$ over $\Sigma_{\{x,y\}}$:  $A_{x<y}$ over $\Sigma_{\{x,y\}}$: 

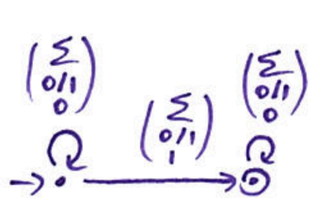
We then get $A_{x<y \to P_b(y)} := A_{P_b(y)} \cup A_{\neg(x<y)}$:

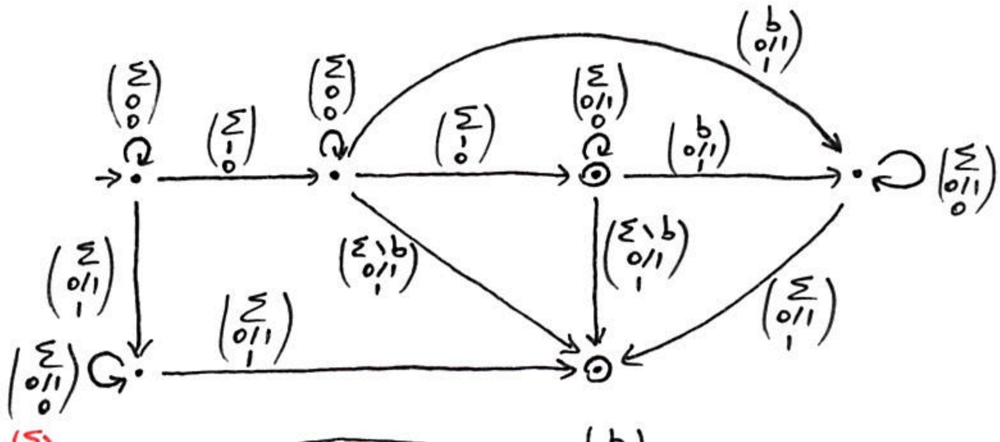


Using the powerset construction we then get:

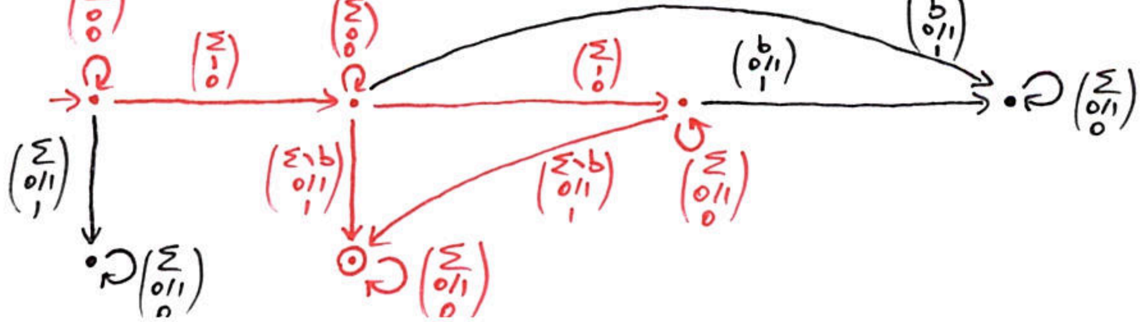


which yields $A_{\exists y}$ and $A_{\neg(x<y \to P_b(y))}$:



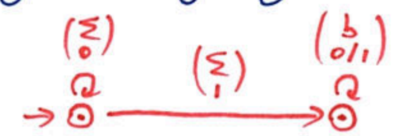


so $A_{\exists y} \cap A_{\neg(x<y \to P_b(y))}$ is:

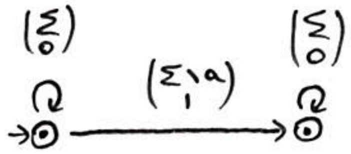


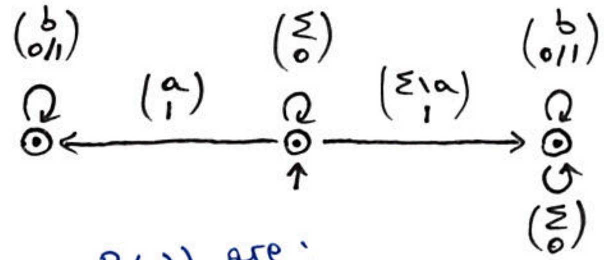
By projecting away y in the previous automaton we have $A_{\exists y.\, \neg(x<y \rightarrow P_b(y))}$:

Then, by determinizing and negating we have $A_{\forall y.\, x<y \rightarrow P_b(y)}$:

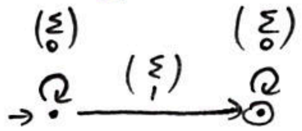


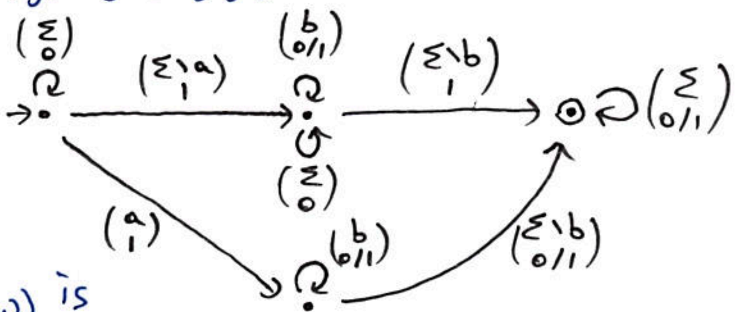
Furthermore, $A_{\neg P_a(x)}$ over $\Sigma_{\{x\}}$ and $A_{P_a(x) \rightarrow \forall y.\, x<y \rightarrow P_b(y)}$ are:



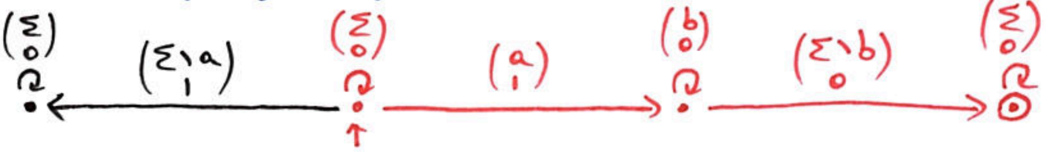


Then $A_{\exists x}$ and $A_{\neg(P_a(x) \rightarrow \forall y.\, x<y \rightarrow P_b(y))}$ are:

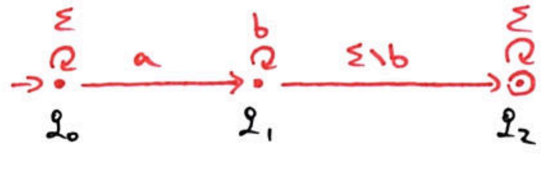




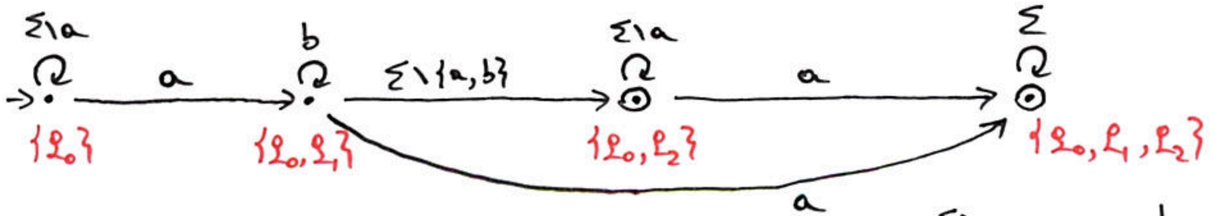
so $A_{\exists x} \cap A_{\neg(P_a(x) \rightarrow \forall y.\, x<y \rightarrow P_b(y))}$ is



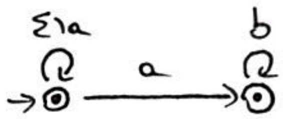
thus $A_{\exists x.\, \neg(P_a(x) \rightarrow \forall y.\, x<y \rightarrow P_b(y))}$ is



which, by determinization is the same as:



and then, by negation A_φ is as expected



2.5. Let $Var := \{x_1, \dots, x_n\} \cup \bigcup_t \{r_1^t, \dots, r_m^t\}$ as in Exercise 1.2. Statements are then interpreted using reads/writes as follows:

- $\ell: x \leftarrow y$
  $\ell': \_\_\_$

  $\equiv$ $\ell: r(y,0)$ // goto next line
  $\quad w(x,0)$; goto ℓ'
  $\ell: r(y,1)$ // goto next line
  $\quad w(x,1)$; goto ℓ'
  $\ell': \_\_\_$

- ℓ: if x then ℓ_1
  $\quad$ else ℓ_2
  $\ell_1: \_\_\_$
  $\ell_2: \_\_\_$

  $\equiv$ $\ell: r(x,1)$; goto ℓ_1
  $\ell: r(x,0)$; goto ℓ_2
  $\ell_1: \_\_\_$
  $\ell_2: \_\_\_$

- $\ell: r_1 := \neg r_2$
  $\ell': \_\_\_$

  $\equiv$ $\ell: r(r_2, 0)$ // goto next line
  $\quad w(r_1, 1)$; goto ℓ'
  $\ell: r(r_2, 1)$ // goto next line
  $\quad w(r_1, 0)$; goto ℓ'
  $\ell': \_\_\_$

- $\ell: r_1 := r_2 \wedge r_3$
  $\ell': \_\_\_$

  $\equiv$ $\ell: r(r_2, 1)$ // goto next line
  $\quad r(r_3, 1)$ // goto next line
  $\quad w(r_1, 1)$; goto ℓ'
  $\ell: r(r_2, 1)$ // goto next line
  $\quad r(r_3, 0)$ // goto next line
  $\quad w(r_1, 0)$; goto ℓ'
  $\ell: \dots$
  $\ell: \dots$
  $\ell': \_\_\_$

- ℓ: goto ℓ'
  $\ell': \_\_\_$

  $\equiv$ $\ell: r(x,0)$; goto ℓ' // for some $x \in Var$
  $\ell: r(x,1)$; goto ℓ'
  $\ell': \_\_\_$