4.1. (a) $a^*b^* = (\Sigma^* \setminus \Sigma^*.b.\Sigma^*).(\Sigma^* \setminus \Sigma^*.a.\Sigma^*)$ is $FO[<, suc]$-defined by $\forall x.\forall y.\ P_a(x) \wedge P_b(y) \to x<y$.

(b) $\{\varepsilon, a, b\} \cup a\Sigma^*a \cup b\Sigma^*b$ describes the set of words over $\Sigma := \{a, b\}$ that contain the subword ab as often as the subword ba.

This language is clearly star-free and $FO[<, suc]$-defined by $\forall x.\forall y.\ first(x) \wedge last(y) \to (P_a(x) \wedge P_a(y)) \vee (P_b(x) \wedge P_b(y))$.

One could give an inductive argument of why each word in the above language contains subword $\underline{ab}$ as often as $\underline{ba}$.

4.2. Problem A: $A \models \varphi$? Input: A, φ

Problem B: s reachable in A? Input: $A = (\Sigma, Q, q_0, \to, Q_F), s \in Q$

Turing reduction proving $A \leq_T B$:

procedure satisfy (A, φ)
  construct $A_{\neg\varphi}$ with the Büchi construction
  construct $A \cap A_{\neg\varphi}$ as the intersection automaton A'
  construct A'' from A by adding state $\bar{q}$ and
    transitions $q \xrightarrow{a} \bar{q}$ for all $q \xrightarrow{a} q'$ in A' such that $q' \in Q_F$
  return $\neg$reachable $(\bar{q}, A'')$.

Notice that $A \models \varphi \iff \underbrace{A \cap A_{\neg\varphi}}_{=: A'} = \emptyset$

$\iff$ reachable $\bar{q}$ in A'' = false

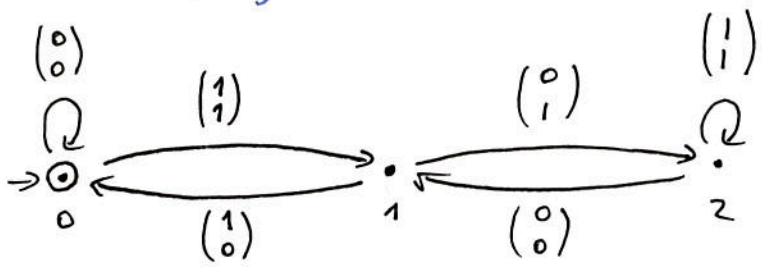
4.3. Clearly $\Diamond\varphi := \text{true}\ \mathcal{U}\ \varphi$ and $\Box\varphi := \neg\Diamond\neg\varphi$. For the other constructors of LTL we have the following reduction:

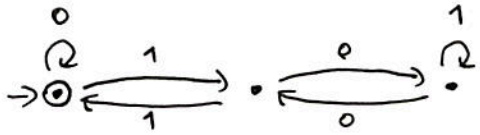
$$\lfloor \text{true} \rfloor(x) := \bigvee_{a\in\Sigma} P_a(x) \quad (\text{or } \forall y.\, y=y, \text{ etc...})$$

$$\lfloor \bigcirc\varphi \rfloor(x) := \exists y.\ \text{suc}(x,y) \to \lfloor\varphi\rfloor(y)$$

$$\lfloor \varphi\,\mathcal{U}\,\psi \rfloor(x) := \exists y.\ x\le y \wedge \lfloor\psi\rfloor(y) \wedge \forall z.\ x\le z \wedge z<y \to \lfloor\varphi\rfloor(z).$$

4.4. One can construct $A_{\exists y} \cap A_{x-3y\le 0} \cap A_{-x+3y\le 0}$ using the construction for $\bar{a}\cdot\bar{x} \le b$ given in class or can use the slight modification to the algorithm described in section 10.2.1. of Esparza's notes to compute directly $A_{x-3y=0} := A_{x-3y\le 0} \cap A_{-x+3y\le 0}$. Then $A_{x-3y=0}$ is:



By projecting away y we get 

.

The difference between the $\bar{a}\cdot\bar{x} \le b$ and $\bar{a}\cdot\bar{x} = b$ algorithms consists in taking only b as a final state in the 2nd case and by defining

$$q \xrightarrow{\overline{bit}} q' := \frac{1}{2}(q - \bar{a}\cdot\overline{bit})$$

only when $q - \bar{a}\cdot\overline{bit}$ is even.