Automata and Formal Languages — Homework 2

Due Friday 23rd October (TA: Christopher Broadbent)

Exercise 2.1

Consider two alphabets Σ_1 and Σ_2. Let h be a *homomorphism* $h : \Sigma_1^* \to \Sigma_2^*$—that is a map such that

$$\text{(i) } h(\epsilon) = \epsilon \qquad \text{and} \qquad \text{(ii) } h(w_1 w_2) = h(w_1)h(w_2) \text{ for all } w \in \Sigma_1^*$$

(a) Prove that if $L \subseteq \Sigma_1^*$ is regular, then $h(L) \subseteq \Sigma_2^*$ is also regular.

(b) Prove that h is injective if and only if the following holds:

For all $L \subseteq \Sigma_1^*$ it is the case that if $h(L)$ is regular, then L is also regular.

(c) Show that for every finite alphabet Σ, there exists an injective homomorphism $h : \Sigma \to \mathbb{B}^*$, where $\mathbb{B} = \{ 0, 1 \}$.

(d) Let Σ be a finite alphabet such that $|\Sigma| > 1$. Let $\mathbb{U} = \{ \bullet \}$ be the alphabet containing just one element. Prove that there exists *no* homomorphism $\phi : \Sigma^* \to \mathbb{U}^*$ that is injective.

Exercise 2.2

Recall the definition of *residual*: Given a language $L \subseteq \Sigma^*$ and $w \in \Sigma^*$, the w-residual of L is the language $L^w = \{ u \in \Sigma^* \mid wu \in L \}$. A language $L' \subseteq \Sigma^*$ is a *residual* of L if it is a w-residual of L for some $w \in \Sigma^*$.

Determine the residuals of the following languages over $\Sigma = \{a, b\}$: $(ab + ba)^*$, $(aa)^*$, and $\{a^n b^n c^n \mid n \geq 0\}$.

Exercise 2.3

Given a language $L \subseteq \Sigma^*$ and $w \in \Sigma^*$, we denote ${}^{w}L = \{u \in \Sigma^* \mid uw \in L\}$. A language $L' \subseteq \Sigma^*$ is an *inverse residual* of L if $L' = {}^{w}L$ for some $w \in \Sigma^*$.

(a) Determine the inverse residuals of the first two languages in Exercise 2.2.

(b) Show that a language is regular iff it has finitely many inverse residuals.

(c) Does a language always have as many residuals as inverse residuals?

Exercise 2.4

We consider encodings of the natural numbers $\mathbb{N} = \{ 0, 1, 2, \dots \}$ in respectively $\mathbb{B}^*$ and $\mathbb{U}^*$ (where $\mathbb{B}$ and $\mathbb{U}$ are as in Exercise 2.1). Observe that the *binary encoding* $\mathbf{B}(n)$ for each $n \in \mathbb{N}$ can be seen as an element of $\mathbb{B}^*$ where trailing 0s are suppressed. (E.g. $\mathbf{B}(0) = \epsilon$, $\mathbf{B}(1) = 1$, $\mathbf{B}(2) = 10$, $\mathbf{B}(6) = 110$). The *unary encoding* $\mathbf{U}(n)$ can be seen as an element of $\mathbb{U}^*$ where $\mathbf{U}(n)$ is the word consisting of n $\bullet$s. (E.g. ($\mathbf{U}(0) = \epsilon$, $\mathbf{U}(1) = \bullet$, $\mathbf{U}(2) = \bullet\bullet$, $\mathbf{U}(6) = \bullet\bullet\bullet\bullet\bullet\bullet$).

(a) Consider a language $L \subseteq \mathbb{U}^*$ encoding the set of natural numbers $S := \mathbf{U}^{-1}(L) \subseteq \mathbb{N}$. Describe the sets of the form $T = \mathbf{U}^{-1}(L') \subseteq \mathbb{N}$ where L' is a residual of L.

Do the same for $L \subseteq \mathbb{B}^*$ and $\mathbf{B}$.

(b) Prove that there exists a set of natural numbers $S \subseteq \mathbb{N}$ such that $\mathbf{B}(S)$ is regular but $\mathbf{U}(S)$ is *not* regular.

[**Hint:** Recall that regular languages have a finite number of residuals. Consider using exponentiation to define a candidate S.]

(c) Prove that for every $S \subseteq \mathbb{N}$ such that $\mathbf{U}(S)$ is regular, it is also the case that $\mathbf{B}(S)$ is regular.

Exercise 2.5

An NFA $A = (Q, \Sigma, \delta, Q_0, F)$ is *reverse-deterministic* if $(q_1, a, q) \in \delta$ and $(q_2, a, q) \in \delta$ implies $q_1 = q_2$, i.e., no state has two input transitions labelled by the same letter. Further, A is *trimmed* if every state accepts at least one word, i.e., if $L_A(q) \neq \emptyset$ for every $q \in Q$.

Let A be a reverse-deterministic, trimmed NFA with one single final state q_f. Prove that *NFAtoDFA*(A) is a minimal DFA.

[**Hint:** Show that any two distinct states of *NFAtoDFA*(A) recognize different languages.]

Exercise 2.6

Let us fix an alphabet $\Sigma = \{ a_i \mid i \in [1, n] \}$ of size n. For each $a_i \in \Sigma$ and $w \in \Sigma^*$ we define $\#_{a_i}(w)$ to be the number of occurrences of a_i in w. (E.g. $\#_{a_2}(a_1a_2a_1a_2a_2) = 3$ and $\#_{a_2}(\epsilon) = \#_{a_2}(a_1a_1) = 0$). The *Parikh vector* $\mathcal{P}(w)$ associated with a word $w \in \Sigma^*$ is the vector $\vec{v} \in \mathbb{N}^n$ that counts the number of occurrences of each symbol in w. That is: $\mathcal{P}(w) = \langle \#_{a_1}(w), \ldots, \#_{a_n}(w) \rangle$. For a language $L \subseteq \Sigma^*$ we call $\mathcal{P}(L) := \{ \mathcal{P}(w) \mid w \in L \}$ *the Parikh image* of L.

(a) Where $a := a_1$ and $b := a_2$, characterise the sets $\mathcal{P}((ab)^*)$ and $\mathcal{P}(\{ a^n b^n \mid n \geq 0 \})$.

(b) For arbitrary languages $L_1, L_2 \subseteq \Sigma^*$ (not necessarily regular) describe how $\mathcal{P}(L)$ relates to $\mathcal{P}(L_1)$ and $\mathcal{P}(L_2)$ in each of the following cases: (i) $L = L_1 \cup L_2$, (ii) $L = L_1 \cap L_2$, (iii) $L = L_1 \cdot L_2$, (iv) $L = L_1^*$, (v) $L = L_1^+$.

(c) A set of vectors $V \subseteq \mathbb{N}^n$ is *linear* if it takes the form $V = \{ \vec{v} = \vec{v}_0 + \lambda_1\vec{v}_1 + \cdots + \lambda_k\vec{v}_k \mid \lambda_1, \ldots, \lambda_k \in \mathbb{N} \}$ for some vectors $\vec{v}_0, \vec{v}_1, \ldots, \vec{v}_k \in \mathbb{N}^n$.

Prove that for every linear set $V \subseteq \mathbb{N}^n$ there exists a regular language $L \subseteq \Sigma^*$ such that $V = \mathcal{P}(L)$.

(d) A set of vectors $U \subseteq \mathbb{N}^n$ is called *semi-linear* if it is of the form $U = V_1 \cup \cdots \cup V_m$ for some linear sets $V_1, \ldots, V_m$.

Prove that for every semi-linear set $U \subseteq \mathbb{N}^n$ there exists a regular language $L \subseteq \Sigma^*$ such that $U = \mathcal{P}(L)$.

(e) Prove that for all regular expressions e_1, e_2 it is the case that $\mathcal{P}((e_1 + e_2)^*) = \mathcal{P}(e_1^* e_2^*)$.

(f) We inductively define two operations on regular expressions $\hat{e}$ that *do not contain union (addition)*. Intuitively $Ext_*(\hat{e})$ ('*ext*ract $*$') is the regular expression formed by deleting all sub-expressions that are *not* in the scope of an $*$. Intuitively $Str_*(\widehat{e})$ ('*str*ip $*$') is the regular expression formed by deleting all sub-expressions that *are* in the scope of an $*$.

$$\begin{aligned} Ext_*(a_i) &= \epsilon \qquad & Ext_*(\widehat{e_1}\widehat{e_2}) &= Ext_*(\widehat{e_1})Ext_*(\widehat{e_2}) \qquad & Ext_*(\widehat{e}^*) &= \widehat{e}^* \\ Str_*(a_i) &= a_i \qquad & Str_*(\widehat{e_1}\widehat{e_2}) &= Str_*(\widehat{e_1})Str_*(\widehat{e_2}) \qquad & Str_*(\widehat{e}^*) &= \epsilon \end{aligned}$$

Prove that for all regular expressions $\widehat{e}$ that do not contain union it is the case that

$$\mathcal{P}(\widehat{e}^*) = \mathcal{P}(Ext_*(\widehat{e})\, Str_*(\widehat{e})^+ + \epsilon)$$

(g) Prove that $\mathcal{P}(L)$ is semi-linear for every regular language $L \subseteq \Sigma^*$.

(Observe that combining (g) and (d) tells us that the semi-linear sets are precisely *the Parikh images of the regular languages. How cool is that?)*

Solution 2.1

We introduce some additional notation that is used throughout this solution. Suppose that Q is a finite set of states and Σ is a finite alphabet.

Given a(n ϵ-free) transition relation $\Delta \subseteq Q \times \Sigma \times Q$ and word $w \in \Sigma^*$, we write $q_1 \xrightarrow[\Delta]{w} q_2$ to mean that an NFA with transition relation Δ has a run on w from state q_1 to state q_2. Formally we can define $\xrightarrow[\Delta]{w}$ by induction on the structure of w:

$$q \xrightarrow[\Delta]{\epsilon} q \qquad \text{and} \qquad q_1 \xrightarrow[\Delta]{w\,a} q_2 \text{ if for some } p \in Q \text{ it is the case that } q_1 \xrightarrow[\Delta]{w} p \text{ and } (p, a, q_2) \in \Delta$$

for all $q, q_1, q_2 \in Q$.

In a similar vein, for a relation of the form $\Delta' \subseteq Q \times \Sigma^+ \times Q$, which we call a *non-empty word transition relation*, we also write $q_1 \xrightarrow[\Delta']{w} q_2$ to mean that there is a run on w from q_1 to q_2 in a regular automaton with transition relation Δ'. Formally this overloads notation since the inductive definition must be modified to reflect the fact that Δ' labels its transitions over Σ^* instead of Σ:

$$q \xrightarrow[\Delta']{\epsilon} q \qquad \text{and} \qquad q_1 \xrightarrow[\Delta']{w\,w'} q_2 \text{ if for some } p \in Q \text{ it is the case that } q_1 \xrightarrow[\Delta']{w} p \text{ and } (p, w', q_2) \in \Delta'$$

By definition, if $A = (\Sigma, Q, \Delta, Q_0, F)$ is an ϵ-free NFA (resp. regular automaton whose transition relation is a non-empty word transition relation) it is the case that

$$\mathcal{L}(A) = \{\ w \in \Sigma^* \mid q_0 \xrightarrow[\Delta]{w} q_f \quad \text{for some } q_0 \in Q_0 \text{ and } q_f \in F\ \}$$

(a) Suppose that $L \subseteq \Sigma_1^*$ is regular. There must be an ϵ-free finite automaton $A_1 = (\Sigma_1, Q, \Delta_1, Q_0, F)$ such that $\mathcal{L}(A_1) = L$. It suffices to show that there is a regular automaton A_2 such that $\mathcal{L}(A_2) = h(L)$. In fact we will only use the special case of regular automata in which the transition relation is a non-empty word transition relation.

We claim that the regular automaton A_2 is as required, where $A_2 = (\Sigma_2, Q, \Delta_2, Q_0, F)$ with Δ_2 defined by

$$\Delta_2 = \{\ (q_1, h(a), q_2) \mid (q_1, a, q_2) \in \Delta_1\ \}$$

We now prove that A_2 is indeed as required

We argue by induction on the length of $w \in \Sigma_2^*$ that for all $q_1, q_2 \in Q$ it is the case that

$$q_1 \xrightarrow[\Delta_2]{w} q_2 \qquad \text{if and only if} \qquad w = h(w_0) \text{ for some } w_0 \in \Sigma_1^* \text{ such that } q_1 \xrightarrow[\Delta_1]{w_0} q_2.$$

- The base case is when $w = \epsilon$.
  Since A_1 is ϵ-free and A_2 has a *non-empty* word transition relation, it must be the case that

  $$q_1 \xrightarrow[\Delta_2]{\epsilon} q_2 \qquad \text{iff} \qquad q_1 = q_2 \qquad \text{iff} \qquad q_1 \xrightarrow[\Delta_1]{\epsilon} q_2$$

  Since h is a homomorphism, $h(\epsilon) = \epsilon$. Thus taking $w_0 = \epsilon$ shows us that the hypothesis holds in the base case.
- For the induction step consider $w \in \Sigma_2^+$ and $q_1, q_2 \in Q$. By definition

  $$q_1 \xrightarrow[\Delta_2]{w} q_2 \qquad \text{iff} \qquad \text{there exist } w_1 \in \Sigma_2^*, w_2 \in \Sigma_2^+ \text{ and } p \in Q \text{ s.t. } w = w_1 w_2 \text{ and } q_1 \xrightarrow[\Delta_2]{w_1} p \text{ and } (p, w_2, q_2) \in \Delta_2$$

  By the induction hypothesis, for $w_1 \in \Sigma_2^*$ such that $|w_1| < |w|$ it must be the case that

  $$q_1 \xrightarrow[\Delta_2]{w_1} p \qquad \text{iff} \qquad w_1 = h(w_0) \text{ for some } w_0 \in \Sigma_1^* \text{ such that } q_1 \xrightarrow[\Delta_1]{w_0} p$$

  Moreover, by the definition of Δ_2, $(p, w_2, q_2) \in \Delta_2$ iff there exists $a \in \Sigma_1$ such that $w_2 = h(a)$ and $(p, a, q_2) \in \Delta_1$. Combining all of the above gives us

  $$q_1 \xrightarrow[\Delta_2]{w} q_2 \quad \text{iff} \quad \text{there exist } w_0 \in \Sigma_1^* \text{ and } a \in \Sigma_1 \text{ and } p \in Q \text{ s.t. } w = h(w_0)h(a) \text{ and } q_1 \xrightarrow[\Delta_1]{w_0} p \text{ and } (p, a, q_2) \in \Delta_1$$

  Since h is a homomorphism, $h(w_0)h(a) = h(w_0 a)$, and so by additionally considering the inductive definition of $\xrightarrow[\Delta_1]{w_0 a}$ we get the required conclusion:

  $$q_1 \xrightarrow[\Delta_2]{w} q_2 \qquad \text{iff} \qquad q_1 = q_2 \qquad \text{iff} \qquad q_1 \xrightarrow[\Delta_1]{w_0 a} q_2 \text{ where } w = h(w_0 a)$$

In particular we have for every $q_0 \in Q_0$ and $q_f \in F$ and $w \in \Sigma_2^*$ that

$$q_0 \xrightarrow[\Delta_2]{w} q_f \qquad \text{iff} \qquad q_0 \xrightarrow[\Delta_1]{w_0} q_f \text{ for some } w_0 \in \Sigma_1^* \text{ s.t. } w = h(w_0)$$

That is to say, $w \in \mathcal{L}(A_2)$ iff $w \in h(\mathcal{L}(A_1))$, in other words $\mathcal{L}(A_2) = h(L)$, as required.

$\square$

(b) $\Rightarrow$ Suppose that h is an injective homomorphism and that $L \subseteq \Sigma_1^*$ is such that $h(L)$ is regular. There must then be a finite automaton $A_2 = (\Sigma_2, Q, \Delta_2, Q_0, F)$ such that $\mathcal{L}(A_2) = h(L)$. We construct a finite automaton $A_1 = (\Sigma_1, Q, \Delta_1, Q_0, F)$ by defining Δ_1 by:

$$\Delta_1 := \{\, (q_1, a, q_2) \mid a \in \Sigma_1 \text{ and } q_1 \xrightarrow[\Delta_2]{h(a)} q_2 \,\}$$

We claim that $\mathcal{L}(A_1) = L$ (and hence that L is indeed regular).
By induction on the length of w (which looks similar to the proof of (a)) we can get that for every $w \in \Sigma_1^*$ and $q_1, q_2 \in Q$ it is the case that

$$q_1 \xrightarrow[\Delta_1]{w} q_2 \qquad \text{iff} \qquad q_1 \xrightarrow[\Delta_2]{h(w)} q_2$$

Thus in particular, for every $q_0 \in Q_0$, and $q_f \in Q_f$, and $w \in \Sigma_1^*$

$$q_0 \xrightarrow[\Delta_1]{w} q_f \qquad \text{iff} \qquad q_0 \xrightarrow[\Delta_2]{h(w)} q_f$$

We now can finish the proof of the claim that $\mathcal{L}(A_1) = L$.
Suppose first that $w \in L$. Then, of course, $h(w) \in h(L)$ and so by assumption $h(w) \in \mathcal{L}(A_2)$, which is to say that $q_0 \xrightarrow[\Delta_2]{h(w)} q_f$ whence $q_0 \xrightarrow[\Delta_1]{w} q_f$ and so $w \in \mathcal{L}(A_1)$. Thus we have $L \subseteq \mathcal{L}(A_1)$.
Note that so far we have not used the assumption that h is injective. We now use this assumption to prove that $\mathcal{L}(A_1) \subseteq L$, which combined with the inclusion above completes the proof.
Let $w \in \mathcal{L}(A_1)$. Then $q_0 \xrightarrow[\Delta_1]{w} q_f$ for some $q_0 \in Q_0$ and $q_f \in F$. It follows that $q_0 \xrightarrow[\Delta_2]{h(w)} q_f$ and so $h(w) \in \mathcal{L}(A_2) = h(L)$. It must thus be the case that there exists some $w_0 \in L$ such that $h(w_0) = h(w)$. Since h is injective, $w_0 = w$ and so it is also the case that $w \in L$. Thus $\mathcal{L}(A_1) \subseteq L$, as required. $\square$

$\Leftarrow$ We prove the contrapositive by showing that if h is not injective then there exists a language $L \subseteq \Sigma_1^*$ that is not regular but is also such that $h(L)$ *is* regular.
Suppose that h is *not* injective. Then there must exist *distinct* $a, b \in \Sigma_1$ such that $h(a) = h(b)$. Let us define $w := h(a) = h(b) \in \Sigma_2^*$. The language $L = \{\, (a^n b^n) \mid n \in \mathbb{N} \,\}$ is irregular. However, $h(L) = \{\, (w^n w^n) \mid n \in \mathbb{N} \,\} = \{\, (ww)^n \mid n \in \mathbb{N} \,\}$. This is just the regular language given by $(ww)^*$. $\square$

(c) Suppose that $\Sigma = \{\, a_1, \ldots, a_n \,\}$. Let us write $\mathbf{B}(i)$ to denote the binary representation of the natural number i for $1 \leq i \leq n$. Thus $\mathbf{B}(i) \in \mathbb{B}^*$. Let us further define k to be the maximum number of digits appearing in $\mathbf{B}(i)$ for any $1 \leq i \leq n$. We can then define $\hat{h} : \Sigma \to \mathbb{B}^*$ by $\hat{h}(a_i) := 0^{k-|\mathbf{B}(i)|}\mathbf{B}(i)$. Observe that for every $1 \leq i \leq n$ it is the case that $|a_i| = k$ (each letter maps to a word in $\mathbb{B}^*$ of the same length).

$\hat{h}$ induces a unique homomorphism $h : \Sigma^* \to \mathbb{B}^*$ defined inductively by:

$$h(\epsilon) := \epsilon \text{ and } h(w\, a) := h(w)\hat{h}(a)$$

We need to check that h is injective. We prove by induction on the total length of words w_1 and w_2 in Σ_1^* that for all such words it is the case that $h(w_1) = h(w_2)$ implies that $w_1 = w_2$.

The base case is when $w_1 = w_2 = \epsilon$, which is immediate. For the induction step, suppose that $w_1 = w_1' a$ for some letter $a \in \Sigma_1$ and that $h(w_1' a) = h(w_1')\hat{h}(a) = h(w_2)$. Since $\hat{h}(a) \neq \epsilon$ and so $h(w_1) \neq \epsilon$, it must be the case that $h(w_2) \neq \epsilon$ and so $w_2 \neq \epsilon$. Thus for some $w_2' \in \Sigma_1^*$ and letter $b \in \Sigma_1$ it is the case that $w_2 = w_2' b$.

Thus we have $h(w_1')\hat{h}(a) = h(w_2')\hat{h}(b)$. Since $\hat{h}$ maps letters to words of length k, $|\hat{h}(a)| = |\hat{h}(b)| = k$. Thus it must be the case that $\hat{h}(a) = \hat{h}(b)$. Since $\hat{h}$ is, by construction, injective, it follows that $a = b$. (Let us set $c := a = b$). Moreover, we have $h(w_1') = h(w_2')$ and so by the induction hypothesis, $w_1' = w_2'$. Let us say $w := w_1' = w_2'$.

Thus $w_1 = w_2 = wc$, as required. $\square$

(d) Suppose for contradiction that such an injective homomorphism does exist. Since $|\Sigma| > 1$, there must exist distinct $a, b \in \Sigma$. It must be the case that for some $m, n \in \mathbb{N}$ we have $h(a) = \bullet^m$ and $h(b) = \bullet^n$. Thus $h(a)h(b) = h(b)h(a) = \bullet^{m+n}$. Since h is a homomorphism, we thus get $h(ab) = h(a)h(b) = h(b)h(a) = h(ba)$, which contradicts injectivity, since by assumption $ab \neq ba$. $\square$

Solution 2.2

- For $(ab + ba)^*$. We give the residuals as regular expressions: $(ab + ba)^*$ (residual of ε); $b(ab + ba)^*$ (residual of a); $a(ab + ba)^*$ (residual of b); $\emptyset$ (residual of aa). All other residuals are equal to one of these four.
- For $(aa)^*$. We give the residuals as regular expressions: $(aa)^*$ (residual of ε); $a(aa)^*$ (residual of a); $\emptyset$ (residual of b). All other residuals are equal to one of these three.
- For $\{a^n b^n c^n \mid n \geq 0\}$: Every prefix of a word of the form $a^n b^n c^n$ has a different residual. For all other words the residual is the empty set. There are infinitely many residuals.

Solution 2.3

(b) Let L^R be the reverse of L. Since $uw \in L$ iff $w^R u^R \in L^R$, we have $u \in {}^wL$ iff $u^R \in \left(L^R\right)^w$. So K is an inverse residual of L iff K^R is a residual of L^R. In particular, the number of inverse residuals of L is equal to the number of residuals of L^R.

Now we have:

$$\begin{array}{ll} & L \text{ is regular} \\ \text{iff} & L^R \text{ is regular} \\ \text{iff} & L^R \text{ has finitely many residuals} \\ \text{iff} & L \text{ has finitely many residuals} \end{array}$$

(c) No. Consider the language L over $\{a, b\}$ containing all words ending with a. The language has two residuals:

$$L^w = \begin{cases} \varepsilon + (a+b)^* a & \text{if } w = w'a \text{ for some } w \in \{a, b\}^* \\ (a+b)^* a & \text{if } w = w'b \text{ for some } w \in \{a, b\}^* \text{ or } w = \varepsilon \end{cases}$$

However, it has three inverse residuals:

$${}^wL = \begin{cases} (a+b)^* a & \text{if } w = \varepsilon \\ (a+b)^* & \text{if } w = w'a \text{ for some } w \in \{a, b\}^* \\ \emptyset & \text{if } w = w'b \text{ for some } w \in \{a, b\}^* \end{cases}$$

Solution 2.4

(a)
- For the unary encoding the residuals represent sets of numbers of the form $T_m = \{ n \in \mathbb{N} \mid m + n \in L \}$ for each $m \in \mathbb{N}$.
- For the binary encoding, the residuals represent sets of numbers of the form $T_m = \{ n \in \mathbb{N} \mid m.2^{\lfloor \log_2' n \rfloor + 1} + n \in L \}$ where we define

$$\log_2' k = \begin{cases} \log_2 k & \text{if } k \geq 1 \\ -1 & \text{if } k = 0 \end{cases}$$

Note that $|\mathbf{B}(n)| = \lfloor \log_2' n \rfloor + 1$ so that $\mathbf{B}(m.2^{\lfloor \log_2' n \rfloor + 1}) = \mathbf{B}(m) \underbrace{0 \cdots 0}_{|\mathbf{B}(n)|\text{-times}}$ and $\mathbf{B}(m.2^{\lfloor \log_2' n \rfloor + 1} + n) = \mathbf{B}(m)\mathbf{B}(n)$.

(b) Let $S = \{ 2^n \mid n \in \mathbb{N} \}$. Then $\mathbf{B}(S) = 10^*$, and so is regular.

We now prove that $\mathbf{U}(S)$ is irregular. It suffices to show that $\mathbf{U}(S)$ has infinitely many residuals.

The residuals of $\mathbf{U}(S)$ take the form $R_m = \{ \bullet^k \mid k + m = 2^n \text{ for some } n \in \mathbb{N} \}$ for each $m \in \mathbb{N}$. Since we are working over a unary alphabet, words are uniquely determined by their length, and so as in part (a) it is helpful to consider residuals as the set of numbers $\mathbf{U}^{-1}(S)$ that they define:

$$T_m = \{ |w| \mid w \in U_m \} = \{ k \mid k + m = 2^n \text{ for some } n \in \mathbb{N} \}$$

It suffices to show that there are infinitely many such sets T_m. Consider the special cases of the form $V_r := T_{2^{r+1} - 2^r}$ for each $r \in \mathbb{N}$.

Let $r \geq 1$. Since $2^r + (2^{r+1} - 2^r) = 2^n$ for $n = r + 1$, it must be the case that $2^r \in V_r$.

Now let $r' \in \mathbb{N}$ be such that $0 \le r' < r$. Then $2^{r'} + (2^{r+1} - 2^r) = 2^{r'}(1 + 2^{r+1-r'}2^{r-r'})$ where $r + 1 - r' > 0$ and $r - r' > 0$. Dividing this number by 2 thus leaves remainder 1 whence it cannot be of the form 2^n for $n \in \mathbb{N}$ (since numbers of the latter form leave 0 remainder upon division by 2). We can thus infer that $r' \neq V_r$.

Putting this together tells us that amongst the sets T_m is the infinite collection of sets: $V_1, V_2, V_3, \ldots, V_r, \ldots$ for each $r \geq 1$. To see that this collection is indeed infinite we show that $V_r \neq V_{r'}$ for every $r \neq r'$.

Suppose for contradiction that there exist $r \neq r'$ such that $V_{r'} = V_r$. Without loss of generality assume that $r' < r$. Then as we have previously seen $2^{r'} \in V_{r'}$, but $2^{r'} \notin V_r$, which implies that $V_{r'} \neq V_r$ after all, a contradiction. □

(c) I am going to save this question for a subsequent problem sheet. You will learn some techniques in subsequent lectures that will make for a much more elegant proof than using the apparatus currently at your disposal. (Look out for Presburger Arithmetic). □

Solution 2.5

Let $B = \mathit{NFAtoDFA}(A)$ and let Q_1, Q_2 be two distinct states of B. Then Q_1 and Q_2 are sets of states of A, and we have $L_B(Q_i) = \bigcup_{q \in Q_i} L_A(q)$ for $i = 1, 2$. We prove $L_B(Q_1) \neq L_B(Q_2)$. Assume the contrary. Then, since $Q_1 \neq Q_2$, there is $q_1 \in Q_1 \setminus q_2$. Since A is trimmed, the $L_A(q)$ contains at least one word w. Since $L_B(Q_1) = L_B(Q_2)$, we have $w \in L(q_2)$ for some $q_2 \in Q_2$, and further $q_1 \neq q_2$. Since q_f is the unique final state of A, the NFA has two paths $q_1 \delta w q_f$ and $q_2 \delta w q_f$. Since these paths start at different states and end at the same state, there is a prefix $w'a$ of w, two different states q_1', q_2', and a state q such that $q_1 \delta w' q_1' \delta a q$ and $q_2 \delta w' q_2' \delta a q$. So A is not reverse-deterministic, contradicting the assumption.

Solution 2.6

(a) Both languages have the same Parikh images namely the set

$$\{ (n, n) \mid n \in \mathbb{N} \}$$

(b) (i) $\mathcal{P}(L) = \mathcal{P}(L_1) \cup \mathcal{P}(L_2)$, (ii) $\mathcal{P}(L) = \mathcal{P}(L_1) \cap \mathcal{P}(L_2)$, (iii) $\mathcal{P}(L) = \mathcal{P}(L_1) + \mathcal{P}(L_2)$, (iv) $\mathcal{P}(L) = \bigcup_{k \in \mathbb{N}} \sum_{i=1}^{k} L_1 \cup \{ (0, \ldots, 0) \}$ (v) $\mathcal{P}(L) = \bigcup_{k \in \mathbb{N}} \sum_{i=1}^{k} L_1$

(c) Suppose that $\vec{v}_i = (j_1^i, \ldots, j_n^i)$ for each $0 \le i \le k$. Let $w_i := a_1^{j_1^i} \cdots a_n^{j_1^n}$ for each i. By construction $\mathcal{P}(w_i) = v_i$. We thus have for each $1 \le i \le k$ that $\mathcal{P}(w_i^\star) = \{ \lambda_i w_i \mid \lambda_i \in \mathbb{N} \}$. Moreover $\mathcal{P}(w_0 w_1^\star \cdots w_n^\star) = V$. □

(d) This follows from the fact that every linear set is the Parikh image of a regular language and the fact that regular languages are closed under union. That is, for each V_i there must exist a regular language L_i such that $\mathcal{P}(L_i) = V_i$. Then $\mathcal{P}\left(\bigcup_{i=1}^{m} L_i\right) = U$. □

(e) We have $w \in (e_1 + e_2)^*$ iff $w = e_{i_1} \cdots e_{i_k}$ for some $0 \le k$ such that $i_1, \ldots, i_k \in \{ 1, 2 \}$. To compute the Parikh vector for w we must sum the Parikh vectors for each of the i_j. That is:

$$\mathcal{P}(w) = \sum_{j=1}^{k} \mathcal{P}(e_{i_j}) = p_1 \mathcal{P}(e_1) \ + \ p_2 \mathcal{P}(e_2) = \mathcal{P}(e_1^{p_1} e_2^{p_2}) \in \mathcal{P}(e_1^* e_2^*)$$

taking $p_1 := |\{ r \in [1, k] \mid i_r = 1 \}|$ and $p_2 := |\{ r \in [1, k] \mid i_r = 2 \}|$, where we take the empty sum to be $(0, \ldots, 0)$ (and consider the sum to be empty when $k = 0$).

Thus $\mathcal{P}((e_1 + e_2)^*) \subseteq \mathcal{P}(e_1^* e_2^*)$.

A very similar argument in the opposite direction gives the reverse inclusion and thus establishes the required result. □

(f) Recall that $\widehat{e}^* = \sum_{k=0}^{\infty} \widehat{e}^k$. Thus $\widehat{e}^* = \epsilon + \sum_{k=1}^{\infty} \widehat{e}^k$. It thus suffices to prove that

$$\mathcal{P}\left(\sum_{k=1}^{\infty} \widehat{e}^k \right) \ = \ \mathcal{P}\left(\mathit{Ext}_*(\widehat{e}) \sum_{k=1}^{\infty} \mathit{Str}_*(\widehat{e}^k) \right)$$

This in turn follows from the claim that for every $k \geq 1$ it is the case that

$$\mathcal{P}(\widehat{e}^k) = \mathcal{P}(\mathit{Ext}_*(\widehat{e}) \mathit{Str}_*(\widehat{e}^k))$$

We prove this claim by induction on the structure of $\widehat{e}$.

- One base case is when $\widehat{e} = a_i$ for some $1 \leq i \leq n$ (i.e. when it is a letter). Trivially $a_i^k = \epsilon\, a_i^k = Ext_*(a_i) Str_*(a_i)^k$. The situation is similar for the other base cases (when $\widehat{e} \in \{\, \epsilon, \emptyset \,\}$.
- Suppose $\widehat{e} = \widehat{e_1}\widehat{e_2}$. Then (by properties of $\mathcal{P}(\_)$ and the induction hypothesis):

$$\begin{aligned}\mathcal{P}(\widehat{e}^k) \;&=\; \mathcal{P}(\widehat{e_1}^k) + \mathcal{P}(\widehat{e_2}^k) \;=\; \mathcal{P}(Ext_*(\widehat{e_1})Str_*(\widehat{e_1})^k) + \mathcal{P}(Ext_*(\widehat{e_2})Str_*(\widehat{e_2})^k) \\ &=\; \mathcal{P}(Ext_*(\widehat{e_1})) + \mathcal{P}(Ext_*(\widehat{e_2})) + \mathcal{P}(Str_*(\widehat{e_1})^k) + \mathcal{P}(Str_*(\widehat{e_2})^k) \\ &= \mathcal{P}(Ext_*(\widehat{e_1})Ext_*(\widehat{e_2})(Str_*(\widehat{e_1})Str_*(\widehat{e_2}))^k) = \mathcal{P}(Ext_*(\widehat{e})Str_*(\widehat{e})^k)\end{aligned}$$

- Suppose $\widehat{e} = \widehat{e}_0^\star$. Then since $k \geq 1$, $\widehat{e}^k = (\widehat{e_0}^*)^k = \widehat{e_0}^* = Ext_*(\widehat{e}) = Ext_*(\widehat{e})\epsilon^k = Ext_*(\widehat{e})Str_*(\widehat{e})^k$.

□

(g) Let L be a regular language. Then there must be some regular expression e for L. We first show that for every regular expression e. We will show that for every regular expression e there exists a regular expression e' such that $\mathcal{P}(e) = \mathcal{P}(e')$ where e' is of the form

$$e' = W_1 + W_2 + \cdots + W_k$$

where each W_i has the form

$$W_i = u_1^i \cdots u_l^i (v_1^i)^* \cdots (v_{l'}^i)^*$$

where the u_j^i and v_j^i are just words in Σ^*. It follows quickly from definitions that the Parikh image of a W_i of such a form is linear. It thus follows that $\mathcal{P}(e') = \mathcal{P}(e)$ is semi-linear.

In order to prove the existence of such an e', we will argue by induction on the following properties of e ordered lexicographically: (i) the *star height* of e [defined below], and (ii) the structure of e.

The principle of induction allows us to apply the induction hypothesis to a structurally bigger term (e.g. a term including more + symbols) so long as the star height (which we give a greater priority) decreases.

The *star height* $sh(e)$ of a regular expression e intuitively measures the depth of nesting of $*$. More precisely:

$$sh(a_i) = 0 \quad sh(e^*) = sh(e) + 1 \quad sh(e_1 e_2) = sh(e_1 + she_2) = \max(sh(e_1), sh(e_2))$$

So let us consider the structure of e

- If e is a letter, ϵ or $\emptyset$, then we just take $e' := e$.
- If $e = e_1 e_2$, then by the induction hypothesis there exist e_1' and e_2' of the required form such that $\mathcal{P}(e) = \mathcal{P}(e_1' e_2')$. Since in general language concatenation and union are associative, we can just 'multiply out the brackets' in the expression $e_1' e_2'$ to get e' of the required form.
- If $e = e_1 + e_2$, then by the induction hypothesis there must be e_1' and e_2' of the required form such that $\mathcal{P}(e) = \mathcal{P}(e_1' + e_2')$. But then we can just take $e' = e_1' + e_2'$.
- If $e = e_0^*$, then we apply part (f), which tells us that $\mathcal{P}(e) = \mathcal{P}(Ext_*(e_0)Str_*(e_0)^+ + \epsilon)$. Notice that $sh(Ext_*(e_0)) = sh(e_0) = sh(e) - 1$. We may thus apply the induction hypothesis to $Ext_*(e_0)$ (even though it is not necessarily a subterm of e). Let e_0' be the term of the required form obtained from the induction hypothesis.

  Then $\mathcal{P}(e) = \mathcal{P}(e_0'(Str_*(e_0)^* + \epsilon) + \epsilon)$. Observe that $Str_*(e_0)$ is just a word (in Σ^*). Thus by associativity of concatenation and union it must be possibly to multiply out the brackets to get an expression e' of the required form.

□