0.1 Congruence Path Refinement

In this section we present an algorithm that uses an existing congruence relation and refines it to the point where equivalent states can be "merged".

Definition 0.1.1. Let $\mathcal{A} = (Q, \Sigma, q_0, \delta, c)$ be a DPA and let $\equiv \subseteq Q \times Q$ be an equivalence relation on the state set. For every equivalence class $\kappa \subseteq Q$, let $r_\kappa \in \kappa$ be an arbitrary representative of that class. For a DPA $\mathcal{A}' = (Q', \Sigma, q'_0, \delta', c')$, we say that $\mathcal{A}'$ is a *representative merge of $\mathcal{A}$ w.r.t.* $\equiv$ if it satisfies the following:

- $Q' = \{r_{[q]_\equiv} \subseteq Q \mid q \in Q\}$
- $q'_0 = r_{[q_0]_\equiv}$
- For all $q \in Q'$ and $a \in \Sigma$: $\delta'(q, a) = r_{[\delta(q,a)]_\equiv}$
- $c' = c \upharpoonright_{Q'}$

Definition 0.1.2. Let $\mathcal{A} = (Q, \Sigma, q_0, \delta, c)$ be a DPA and let $R \subseteq Q \times Q$ be a congruence relation on the state space. Let $\lambda \subseteq Q$ be an equivalence class of R. We define $L_{\lambda\hookleftarrow}$ as the set of words w such that for any $u \sqsubseteq w$, $(\delta(p, u), q) \in R$ iff $u \in \{\varepsilon, w\}$. In other words, the set contains all minimal words by which the automaton moves from λ to λ again.

Let $f_{\mathrm{PR}} : 2^{\lambda\times\lambda} \to 2^{\lambda\times\lambda}$ be a function such that $(p, q) \in f(X)$ iff for all $w \in L_{\lambda\hookleftarrow}$, $(\delta^*(p, w), \delta^*(q, w)) \in X$. Then let $X_0 \subseteq \lambda \times \lambda$ such that $(p, q) \in X_0$ iff for all $w \in L_{\lambda\hookleftarrow}$, $\min\{c(\delta^*(p, u)) \mid u \sqsubseteq w\} = \min\{c(\delta^*(q, u)) \mid u \sqsubseteq w\}$, i.e. the minimal priority when moving from p or q to λ again is the same.

Using both, we set $X_{i+1} = f_{\mathrm{PR}}(X_i)$. f_{PR} is monotone w.r.t. $\subseteq$, so there is an $X_n = X_{n+1}$ by Kleene's fixed point theorem. We define the *path refinement of* λ, called $\equiv^\lambda_{\mathrm{PR}}$, as

- For $p \in Q \setminus \lambda$, $p \equiv^\lambda_{\mathrm{PR}} q$ iff $p = q$.
- For $p, q \in \lambda$, $p \equiv^\lambda_{\mathrm{PR}} q$ iff $(p, q) \in X_n$.

Theorem 0.1.1. *Let $\mathcal{A} = (Q, \Sigma, q_0, \delta, c)$ be a DPA and let $R \subseteq Q \times Q$ be a congruence relation that implies language equivalence. Let $\mathcal{A}'$ be a representative merge of $\mathcal{A}$ w.r.t. $\equiv^\lambda_{\mathrm{PR}}$ for some equivalence class λ of R such that each representative r_κ is chosen to have minimal priority. Then $L(\mathcal{A}) = L(\mathcal{A}')$.*

Proof. Let $\alpha \in \Sigma^\omega$ be a word with runs $\rho \in Q^\omega$ and $\rho' \in (Q')^\omega$ of $\mathcal{A}$ and $\mathcal{A}'$ respectively. Let $k_0, \dots \in \mathbb{N}$ be exactly those positions (in order) at which ρ reaches λ, and analogously $k'_0, \dots$ for ρ'.

Claim 1: For every i, $k_i = k'_i$ and $\rho(k_i) \equiv^\lambda_{\mathrm{PR}} \rho'(k_i)$. For all $j < k_0$, we know that $\rho(j) = \rho'(j)$, as no redirected edge is taken. Thus, $\rho'(k_0) = r_{[\rho(k_0)]_{\equiv^\lambda_{\mathrm{PR}}}} \equiv^\lambda_{\mathrm{PR}} = \rho(k_0)$.

Now assume that the claim holds for all $i \leq n$. By definition, $w = \alpha[k_n, k_{n+1}] \in L_{\lambda\hookleftarrow}$ and therefore $\rho(k_{n+1}) = \delta^*(\rho(k_n), w) \equiv^\lambda_{\mathrm{PR}} \delta^*(\rho'(k_n), w) = \rho'(k_{n+1})$.

Claim 2: If λ only occurs finitely often in ρ and ρ', then ρ is accepting iff ρ' is accepting.

Let $k_n \in \mathbb{N}$ be the last position at which $\rho(k_n)$ and $\rho'(k_n)$ are in λ. From this point on, $\rho'[k_n, \omega]$ is also a valid run of $\mathcal{A}$ on $\alpha[k_n, \omega]$. $\rho(k_n), \rho'(k_n) \in \lambda$, so $(\rho(k_n), \rho'(k_n)) \in R$. As R implies language equivalence, reading $\alpha[k_n, \omega]$ from either state in $\mathcal{A}$ leads to the same acceptance status. This also means that $\rho'(k_n)$ has the same acceptance status as $\rho(k_n)$.

Claim 3: If λ occurs infinitely often in ρ and ρ', then ρ is accepting iff ρ' is accepting.

For each i, $\alpha[k_i, k_{i+1}] \in L_{\lambda \hookleftarrow}$ by choice of the k_i. Hence, $\min \mathrm{Occ}(c(\rho[k_i, k_{i+1}])) = \min \mathrm{Occ}(c'(\rho'[k_i, k_{i+1}]))$ follows directly from the definition of $\equiv^{\lambda}_{\mathrm{PR}}$. Extending that result gives us $\min \mathrm{Inf}(c(\rho)) = \min \mathrm{Inf}(c'(\rho'))$.

□

Lemma 0.1.2. *Let $\mathcal{A} = (Q, \Sigma, q_0, \delta, c)$ be a DPA and let $R \subseteq Q \times Q$ be a congruence relation that implies language equivalence. Let $\mathcal{A}'$ be a representative merge of $\mathcal{A}$ w.r.t. $\equiv^{\lambda}_{\mathrm{PR}}$ for some equivalence class λ of R. Then $R\restriction_{Q'}$ still is a congruence relation that implies language equivalence in $\mathcal{A}'$.*

Proof. □

0.1.1 Algorithmic Definition

The definition of path refinement that we introduced is useful for the proofs of correctness. It however does not provide one with a way to actually compute the relation. That is why we now provide an alternative definition that yields the same results but is more algorithmic in nature.

Definition 0.1.3. Let $\mathcal{A} = (Q, \Sigma, q_0, \delta, c)$ be a DPA and let $R \subseteq Q \times Q$ be a congruence relation. For each equivalence class λ of R, we define the *path refinement automaton* $\mathcal{G}^{R,\lambda}_{\mathrm{PR}}(p, q) = (Q_{\mathrm{PR}}, \Sigma, q^{p,q}_{0,\mathrm{PR}}, \delta^{\lambda}_{\mathrm{PR}}, F_{\mathrm{PR}})$, which is a DFA.

- $Q_{\mathrm{PR}} = (Q \times Q \times c(Q) \times \{<, >, =\}) \cup \{\bot\}$
- $q^{p,q}_{0,\mathrm{PR}} = (p, q, \eta_k(c(p), c(q), \checkmark), \eta_x(c(p), c(q), \checkmark, =))$
- $\delta^{\lambda}_{\mathrm{PR}}((p, q, k, x), a) = \begin{cases} (p', q', \eta_k(c(p'), c(q'), k), \eta_x(c(p'), c(q'), k, x)) & \text{if } p' \notin \lambda \\ q^{p',q'}_{0,\mathrm{PR}} & \text{if } p' \in \lambda \text{ and } (\eta_x(c(p'), c(q'), k, x) = =) \\ \bot & \text{else} \end{cases}$

  where $p' = \delta(p, a)$ and $q' = \delta(q, a)$.
  $\eta_k(k_p, k_q, k) = \min_{\leq_\checkmark}\{k_p, k_q, k\}$
  $\eta_x(k_p, k_q, k, x) = \begin{cases} < & \text{if } (k_p <_\checkmark k_q \text{ and } k_p <_\checkmark k) \text{ or } (k < k_q \text{ and } (x = <)) \\ > & \text{if } (k_p >_\checkmark k_q \text{ and } k >_\checkmark k_q) \text{ or } (k_p > k \text{ and } (x = >)) \\ = & \text{else} \end{cases}$
- $F_{\mathrm{PR}} = Q_{\mathrm{PR}} \setminus \{\bot\}$

Lemma 0.1.3. *Let $\mathcal{A}$ be a DPA with a congruence relation R. Let λ be an equivalence class of R, $p, q \in \lambda$, and $w \in L_{\lambda \to \lambda}$. For every $v \sqsubset w$ and $\oplus \in \{<, >, =\}$, the fourth component of $(\delta^{\lambda}_{PR})^*(q_{0,PR}, v)$ is $\oplus$ if and only if $\min\{c(\delta^*(p, u)) \mid u \sqsubseteq v\} \oplus \min\{c(\delta^*(q, u)) \mid u \sqsubseteq v\}$.*

The proof of this Lemma is a very formal analysis of every case in the relations between the different priorities that occur and making sure that the definition of η_x covers these correctly. No great insight is gained, which is why we omit the proof at this point.

Theorem 0.1.4. *Let $\mathcal{A}$ be a DPA with a congruence relation R. Let λ be an equivalence class of R and $p, q \in \lambda$. Then $p \equiv^{R}_{PR} q$ iff $L(\mathcal{G}^{R,\lambda}_{PR}(p, q)) = \Sigma^*$.*

Proof. **If** Let $p \not\equiv_{\mathrm{PR}}^{R} q$. Similarly to the proof of Lemma **??**, we use the inductive definition of $R_\kappa \subseteq \equiv_{\mathrm{PR}}^{R}$ using f and the sets X_i here. Let m be the smallest index at which $(p,q) \notin X_m$. Let $\rho = (p_i, q_i, k_i, x_i)_{0 \leq i \leq |w|}$ be the run of $\mathcal{G}_{\mathrm{PR}}^{R,\lambda}(p,q)$ on w. We prove that $\rho(|w|) = \perp$ and therefore ρ is not accepting by induction on m.

If $m = 0$, then $(p,q) \notin Y_\lambda$, meaning that there is a word w such that $\min\{c(\delta^*(p,u)) \mid u \sqsubset w\} \neq \min\{c(\delta^*(q,u)) \mid u \sqsubset w\}$. Without loss of generality, assume $\min\{c(\delta^*(p,u)) \mid u \sqsubset w\} < \min\{c(\delta^*(q,u)) \mid u \sqsubset w\}$. By Lemma 0.1.3, $x_{|w|-1} = <$. Furthermore, $\delta(p_{|w|-1}, w_{|w|-1}) \in \lambda$, as $w \in L_{\lambda \to \lambda}$. Thus, $\rho(|w|) = \perp$ and the run is rejecting.

Now consider $m+1 > 1$. Since $(p,q) \in X_m \setminus f(X_m)$, there must be a word $w \in L_{\lambda \to \lambda}$ such that $(p',q') \notin X_m$, where $p' = \delta^*(p,w)$ and $q' = \delta^*(q,w)$. As $R_\kappa \subseteq X_m$, $(p',q') \notin R_\kappa$ and therefore $p' \not\equiv_{\mathrm{PR}}^{R} q'$. By induction, $w \notin L(\mathcal{G}_{\mathrm{PR}}^{R,\lambda}(p',q'))$; since that run is a suffix of ρ, ρ itself is also a rejecting run.

Only If Let $L(\mathcal{G}_{\mathrm{PR}}^{R,\lambda}(p,q)) \neq \Sigma^*$. Since ε is always accepted, there is a word $w \in \Sigma^+ \setminus L(\mathcal{G}_{\mathrm{PR}}^{R,\lambda}(p,q))$, meaning that $\delta_{\mathrm{PR}}^*(q_{0,\mathrm{PR}}, w) = \perp$. Split w into sub-words $w = u_1 \cdots u_m$ such that $u_1, \ldots, u_m \in L_{\lambda \to \lambda}$. Note that this partition is unique. We show $p \not\equiv_{\mathrm{PR}}^{R} q$ by induction on m. Let $\rho = (p_i, q_i, k_i, x_i)_{0 \leq i < |w|}$ be the run of $\mathcal{G}_{\mathrm{PR}}^{R,\lambda}(p,q)$ on w.

If $m = 1$, then $w \in L_{\lambda \to \lambda}$. Since $\rho(|w|) = \perp$, it must be true that $x_{|w|-1} \neq =$. Without loss of generality, assume $x_{|w|-1} = <$. By Lemma 0.1.3, $\min\{c(\delta^*(p,u)) \mid u \sqsubset w\} < \min\{c(\delta^*(q,u)) \mid u \sqsubset w\}$. Therefore, $p \not\equiv_{\mathrm{PR}}^{R} q$.

Now consider $m+1 > 1$. Let $p' = \delta^*(p, u_1)$ and $q' = \delta^*(q, u_1)$. By induction on the word $u_2 \cdots u_m$, $p' \not\equiv_{\mathrm{PR}}^{R} q'$. Since $u_1 \in L_{\lambda \to \lambda}$, that also means $p \not\equiv_{\mathrm{PR}}^{R} q$. □

The differences between different $\mathcal{G}_{\mathrm{PR}}^{R,\lambda}$ for different λ are minor and the question whether the accepted language is universal boils down to a simple question of reachability. Thus, $\equiv_{\mathrm{PR}}^{R}$ can be computed in $\mathcal{O}(|\mathcal{G}_{\mathrm{PR}}^{R,\lambda}|)$ which is $\mathcal{O}(|Q|^2 \cdot |c(Q)|)$.

0.1.2 Alternative Algorithmic Definition

The computation presented in the previous section was a straight-forward description of $\equiv_{\mathrm{PR}}^{\lambda}$ in an algorithmic way. We can reduce the complexity of that computation by taking a more indirect route, as we will see now.

Definition 0.1.4. Let $\mathcal{A} = (Q, \Sigma, q_0, \delta, c)$ be a DPA. Let R be a congruence relation on Q and let $\lambda \subseteq Q$ be an equivalence class of R. We define a deterministic transition structure $\mathcal{A}_{\mathrm{visit}}^{\lambda} = (Q_{\mathrm{visit}}^{\lambda}, \Sigma, \delta_{\mathrm{visit}}^{\lambda})$ as follows:

- $Q_{\mathrm{visit}}^{\lambda} = ((Q \setminus \lambda) \times c(Q)) \cup (\lambda \times c(Q) \times c(Q))$
  These states "simulate" $\mathcal{A}$ and use the second component to track the minimal priority that was seen since a last visit to λ. The states in λ itself also have a third component that is used to distinguish their classes, as is explained below.

- For $q \in Q \setminus \lambda$: $\delta_{\mathrm{visit}}^{\lambda}((q,k),a) = \begin{cases} (q', \min\{k, c(q')\}) & \text{if } q' \notin \lambda \\ (q', c(q'), \min\{k, c(q')\}) & \text{if } q' \in \lambda \end{cases}$.
  For $q \in \lambda$: $\delta_{\mathrm{visit}}^{\lambda}((q,k,k'),a) = (q', \min\{k, c(q')\})$.
  Where $q' = \delta(q,a)$.

Definition 0.1.5. Consider $\mathcal{A}^{\lambda}_{\text{visit}}$ of a DPA $\mathcal{A}$ and a congruence relation R. We define an equivalence relation $V \subseteq Q^{\lambda}_{\text{visit}} \times Q^{\lambda}_{\text{visit}}$ as:

- For every $p, q \in Q \setminus \lambda$ and $l, k \in c(Q)$, $((p, l), (q, k)) \in V$.
- For every $p, q \in \lambda$ and $l, k \in c(Q)$, $((p, l, l'), (q, k, k')) \in V$ iff $l' = k'$.

The Moore-refinement of V is then called V_M.

Theorem 0.1.5. *Let $\mathcal{A}$, R, and λ be as before. For $p, q \in \lambda$, we have*

$$p \equiv^{\lambda}_{PR} q \Leftrightarrow ((p, c(p), \max c(Q)), (q, c(q), \max c(Q))) \in V_M$$

.

Proof. □

Constructing $\mathcal{A}^{\lambda}_{\text{visit}}$ and V requires time $\mathcal{O}(nk)$ and the computation of V_M brings it up to $\mathcal{O}(nk \log(nk))$, where $n = |Q|$ and $k = |c(Q)|$.