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 $a \in \Sigma$: $\delta'(q, a) = r_{[\delta(q,a)]_\equiv}$
- $c' = c\restriction_{Q'}$

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 non-empty 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.

We show that for all i, $\min \mathrm{Occ}(c(\rho[k_i, k_{i+1}+1]))$ and $\min \mathrm{Occ}(c'(\rho'[k_i, k_{i+1}+1]))$ are the same. The claim then follows immediately.

Directly observe that $c'(\rho'[k_i, k_{i+1}+1]) = c(\rho'[k_i, k_{i+1}+1])$ and that $\min \mathrm{Occ}(c(\rho[k_i, k_{i+1}+1])) = \min \mathrm{Occ}(c(\rho'[k_i, k_{i+1}] \cdot \delta(\rho'(k_{i+1}-1), \alpha(k_{i+1})))$ because $\rho(k_i) \equiv_{\mathrm{PR}}^{\lambda} \rho'(k_i)$.

Now $\mathrm{Occ}(c(\rho[k_i, k_{i+1}+1])) = \mathrm{Occ}(c(\rho[k_i, k_{i+1}])) \cup \{c(\rho(k_{i+1}))\}$ and $\mathrm{Occ}(c(\rho'[k_i, k_{i+1}] \cdot \delta(\rho'(k_{i+1}-1), \alpha(k_{i+1}))) = \mathrm{Occ}(c(\rho[k_i, k_{i+1}])) \cup \{c(\delta(\rho'(k_{i+1}-1), \alpha(k_{i+1})))\}$.

The rest of the claim follows because $c(\rho'(k_i)) = c(\rho'(k_{i+1})) \leq \delta(\rho'(k_{i+1}, \alpha(k_{i+1}))$. □

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* $\mathcal{G}_{\mathrm{PR}}^{R,\lambda}(p, q) = (Q_{\mathrm{PR}}, \Sigma, q_{0,\mathrm{PR}}^{p,q}, \delta_{\mathrm{PR}}^{\lambda}, F_{\mathrm{PR}})$, which is a DFA.

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

  where $p' = \delta(p, a)$ and $q' = \delta(q, a)$.

  $\eta_k(k_p, k_q, k) = \min_{\leq_\checkmark} \{k_p, k_q, k\}$

  $\eta_x(k_p, k_q, k, x) = \begin{cases} < & \text{if } (k_p <_\checkmark k_q \text{ and } k_p <_\checkmark k) \text{ or } (k < k_q \text{ and } (x = <)) \\ > & \text{if } (k_p >_\checkmark k_q \text{ and } k >_\checkmark k_q) \text{ or } (k_p > k \text{ and } (x = >)) \\ = & \text{else} \end{cases}$
- $F_{\mathrm{PR}} = Q_{\mathrm{PR}} \setminus \{\perp\}$

Lemma 0.1.2. *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 $(\delta_{PR}^{\lambda})^*(q_{0,PR}, v)$ 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 0.1.3. *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 $p \equiv_{PR}^{R} q$ iff $L(\mathcal{G}_{PR}^{R,\lambda}(p, q)) = \Sigma^*$.*

Proof. **If** Let $p \not\equiv^R_{\mathrm{PR}} q$. Similarly to the proof of Lemma **??**, we use the inductive definition of $R_\kappa \subseteq \equiv^R_{\mathrm{PR}}$ using f and the sets X_i here. Let m be the smallest index at which $(p,q) \notin X_m$. Let $\rho = (p_i, q_i, k_i, x_i)_{0 \le i \le |w|}$ be the run of $\mathcal{G}^{R,\lambda}_{\mathrm{PR}}(p,q)$ 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 0.1.2, $x_{|w|-1} = <$. 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 $p' \not\equiv^R_{\mathrm{PR}} q'$. By induction, $w \notin L(\mathcal{G}^{R,\lambda}_{\mathrm{PR}}(p',q'))$; since that run is a suffix of ρ, ρ itself is also a rejecting run.

Only If Let $L(\mathcal{G}^{R,\lambda}_{\mathrm{PR}}(p,q)) \neq \Sigma^*$. Since ε is always accepted, there is a word $w \in \Sigma^+ \setminus L(\mathcal{G}^{R,\lambda}_{\mathrm{PR}}(p,q))$, 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 $p \not\equiv^R_{\mathrm{PR}} q$ by induction on m. Let $\rho = (p_i, q_i, k_i, x_i)_{0 \le i < |w|}$ be the run of $\mathcal{G}^{R,\lambda}_{\mathrm{PR}}(p,q)$ 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 0.1.2, $\min\{c(\delta^*(p,u)) \mid u \sqsubseteq w\} < \min\{c(\delta^*(q,u)) \mid u \sqsubseteq w\}$. Therefore, $p \not\equiv^R_{\mathrm{PR}} q$.

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

The differences between different $\mathcal{G}^{R,\lambda}_{\mathrm{PR}}$ for different λ are minor and the question whether the accepted language is universal boils down to a simple question of reachability. Thus, $\equiv^R_{\mathrm{PR}}$ can be computed in $\mathcal{O}(|\mathcal{G}^{R,\lambda}_{\mathrm{PR}}|)$ 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 $\equiv^\lambda_{\mathrm{PR}}$ 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 $\mathcal{A}^\lambda_{\mathrm{visit}} = (Q^\lambda_{\mathrm{visit}}, \Sigma, \delta^\lambda_{\mathrm{visit}})$ as follows:

- $Q^\lambda_{\mathrm{visit}} = ((Q \setminus \lambda) \times c(Q) \times \{\perp\}) \cup (\lambda \times c(Q) \times c(Q))$
  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.

- $\delta^\lambda_{\mathrm{visit}}((q,k,k'),a) = \begin{cases} (q', \min\{k, c(q')\}, \perp) & \text{if } q' \notin \lambda \\ (q', c(q'), \min\{k, c(q')\}) & \text{if } q' \in \lambda \end{cases}$, where $q' = \delta(q,a)$.

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

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

We abbreviate the state $(q, c(q), k)$ for any $q \in \lambda$ and $k \in c(Q)$ by ι_q^k.

Lemma 0.1.4. *For all $p, q \in \lambda$ and $l, k \in c(Q)$: $(\iota_p^l, \iota_q^k) \in V$ iff $l = k$.*

Proof. Follows directly from the definition. □

Lemma 0.1.5. *Let $q \in \lambda$, $k \in c(Q)$, $w \in L_{\lambda\hookleftarrow}$, and $\varepsilon \sqsubset v \sqsubset w$. Then $(\delta_{\text{visit}}^{\lambda})^*(\iota_q^k, v) = (\delta^*(q, v), x_v, \perp)$, where $x_v = \min\{c(\delta^*(q, u)) \mid u \sqsubseteq v\}$.*

Proof. We provide this proof by using induction on v. First, consider $v = a \in \Sigma$. Since $v \notin L_{\lambda\hookleftarrow}$, we know $\delta(q, a) \notin \lambda$ and thus $(\delta_{\text{visit}}^{\lambda})^*(\iota_q^k, v) = \delta_{\text{visit}}^{\lambda}(\iota_q^k, a) = (\delta(q, a), c(\min\{c(q), c(\delta(q, a))\}, \perp)$. This is exactly what we had to show, as $\min\{c(q), c(\delta(q, a))\} = x_a$.

For the induction step, let $v = v'a \in \Sigma^+ \cdot \Sigma$. Then $(\delta_{\text{visit}}^{\lambda})^*(\iota_q^k, v) = \delta_{\text{visit}}^{\lambda}((\delta^*(q, v'), x_{v'}, \perp), a)$ by induction. Again, $\delta^*(q, v) \notin \lambda$, so $(\delta_{\text{visit}}^{\lambda})^*(\iota_q^k, v) = (\delta^*(q, v), \min\{x_{v'}, c(\delta^*(q, v))\}, \perp)$. This is our goal, as $\min\{x_{v'}, c(\delta^*(q, v))\} = x_v$. □

Lemma 0.1.6. *Let $q \in \lambda$, $k \in c(Q)$, and $w \in L_{\lambda\hookleftarrow}$. Then $(\delta_{\text{visit}}^{\lambda})^*(\iota_q^k, w) = \iota_{q'}^x$, where $q' = \delta^*(q, w)$ and $x = \min\{c(\delta^*(q, u)) \mid u \sqsubseteq w\}$.*

Proof. For all $v \sqsubseteq w$, let $x_v = \min\{c(\delta^*(q, u)) \mid u \sqsubseteq v\}$ (i.e. $x = x_w$). Let $w = va \in \Sigma^* \cdot \Sigma$ (since words in $L_{\lambda\hookleftarrow}$ are non-empty). Then $(\delta_{\text{visit}}^{\lambda})^*(\iota_q^k, v) = (\delta^*(q, v), x_v, \perp)$ by Lemma 0.1.5 and $(\delta_{\text{visit}}^{\lambda})^*(\iota_q^k, w) = \delta_{\text{visit}}^{\lambda}((\delta^*(q, v), x_v, \perp), a)$.

Let $q' = \delta^*(q, w)$. Since $w \in L_{\lambda\hookleftarrow}$, $q' \in \lambda$ and definition tells us $\delta_{\text{visit}}^{\lambda}((\delta^*(q, v), x_v, \perp), a) = (q', c(q'), \min\{x_v, c(q')\})$. The fact that $\min\{x_v, c(q')\} = x_w$ finishes our proof. □

Lemma 0.1.7. *For every $q \in \lambda$, $l, k \in c(Q)$, and $w \in \Sigma^+$: $(\delta_{\text{visit}}^{\lambda})^*(\iota_q^k, w) = (\delta_{\text{visit}}^{\lambda})^*(\iota_q^l, w)$.*

Proof. If suffices to consider the case $w = a \in \Sigma$. If the statement is true for any one-symbol word, then it is for words of any length, as $\mathcal{A}_{\text{visit}}^{\lambda}$ is deterministic.

For $w \in \Sigma$, w is always in $L_{\lambda\hookleftarrow}$ or a prefix of a word in that set. Thus we can apply Lemma 0.1.5 and 0.1.6 to obtain our wanted result. □

Lemma 0.1.8. *For all $p, q \in \lambda$, and $l, k \in c(Q)$, $(\iota_p^k, \iota_q^k) \in V_M$ if and only if $(\iota_p^l, \iota_q^l) \in V_M$.*

Proof. As l and k are chosen symmetrically, it suffices for us to prove on direction of the bidirectional implication. Assume towards a contradiction that $(\iota_p^k, \iota_q^k) \in V_M$ but $(\iota_p^l, \iota_q^l) \notin V_M$, so there is a word $w \in \Sigma^*$ such that $((\delta_{\text{visit}}^{\lambda})^*(\iota_p^l, w), (\delta_{\text{visit}}^{\lambda})^*(\iota_q^l, w)) \notin V$.

It must be true that $w \neq \varepsilon$; otherwise, $(\iota_p^l, \iota_q^l) \notin V$, which would contradict Lemma 0.1.4.

By Lemma 0.1.7, we have $(\delta_{\text{visit}}^{\lambda})^*(\iota_p^l, w) = (\delta_{\text{visit}}^{\lambda})^*(\iota_p^k, w)$ and analogously for q. Therefore, $((\delta_{\text{visit}}^{\lambda})^*(\iota_p^k, w), (\delta_{\text{visit}}^{\lambda})^*(\iota_q^k, w)) \notin V$ and thus $(\iota_p^k, \iota_q^k) \notin V_M$, which contradicts our assumption. □

Lemma 0.1.9. *Let $q \in \lambda$ and $w \in \Sigma^+$ such that $\delta^*(q, w) \in \lambda$. Then there is a decomposition of w into words $v_1 \cdots v_m$ such that all $v_i \in L_{\lambda\hookleftarrow}$.*

Proof. Let $\rho \in Q^*$ be the run of $\mathcal{A}$ starting in q on w. Let $i_1 < \cdots < i_m$ be those positions at which $\rho(i_j) \in \lambda$. For every $1 \leq j < m$, we define $v_j = w[i_j, i_{j+1}]$. By choice of the i_j, all of those words are elements of $L_{\lambda\hookleftarrow}$.

Since $q \in \lambda$ and $\delta^*(q, w) \in \lambda$, we have $i_0 = 0$ and $i_m = |w| + 1$, so $w = v_1 \cdots v_m$. $\square$

Theorem 0.1.10. *Let $\mathcal{A}$, R, and λ be as before. Let $\hat{c} = \max c(Q)$. Then for all $p, q \in \lambda$, we have $p \equiv^\lambda_{\mathrm{PR}} q$ iff $(\iota^{\hat{c}}_p, \iota^{\hat{c}}_q) \in V_M$.*

Proof. We have $(\iota^{\hat{c}}_p, \iota^{\hat{c}}_q) \in V_M$ if and only if for all $w \in \Sigma^*$: $(\delta^*(\iota^{\hat{c}}_p, w), \delta^*(\iota^{\hat{c}}_q, w)) \in V$. Thus, we want to show that his property holds for all w iff $p \equiv^\lambda_{\mathrm{PR}} q$.

If Assume there is a w such that $((\delta^\lambda_{\mathrm{visit}})^*(\iota^{\hat{c}}_p, w), (\delta^\lambda_{\mathrm{visit}})^*(\iota^{\hat{c}}_q, w)) \notin V$. Choose this w to have minimal length. By definition of V, both $(\delta^\lambda_{\mathrm{visit}})^*(\iota^{\hat{c}}_p, w)$ and $(\delta^\lambda_{\mathrm{visit}})^*(\iota^{\hat{c}}_q, w)$ must be in λ. By Lemma 0.1.9, there is a decomposition of w into words $v_1 \cdots v_m$ that are in $L_{\lambda\hookleftarrow}$. We perform a proof of induction on m.

If $m = 1$, then $w \in L_{\lambda\hookleftarrow}$. From Lemmas 0.1.4 and 0.1.6, we know that $\min\{c(\delta^*(p, u)) \mid u \sqsubseteq w\} \neq \min\{c(\delta^*(q, u)) \mid u \sqsubseteq w\}$ and therefore $p \not\equiv^\lambda_{\mathrm{PR}} q$.

If $m + 1 > 1$, consider $(\delta^\lambda_{\mathrm{visit}})^*(\iota^{\hat{c}}_q, v_1) = \iota^x_{q'}$ and $(\delta^\lambda_{\mathrm{visit}})^*(\iota^{\hat{c}}_q, v_1) = \iota^y_{p'}$ as stated in Lemma 0.1.6 (with $p' = \delta^*(p, v_1)$ and q' analogously). w was chosen to have minimal length, so $(\iota^x_{q'}, \iota^y_{p'}) \in V$, which means that $x = y$.

As $(\iota^{\hat{c}}_p, \iota^{\hat{c}}_q) \notin V_M$, we also have $(\iota^x_{p'}, \iota^y_{q'}) \notin V_M$ and by Lemma 0.1.8, $(\iota^{\hat{c}}_{p'}, \iota^{\hat{c}}_{q'}) \notin V_M$ with the word $v_2 \cdots v_m$ being a witness. We can therefore argue with induction to deduce $p' = \delta^*(p, v_1) \not\equiv^\lambda_{\mathrm{PR}} q' = \delta^*(q, v_1)$. As $v_1 \in L_{\lambda\hookleftarrow}$, the definition of path refinement tells us $p \not\equiv^\lambda_{\mathrm{PR}} q$.

Only If Assume $p \not\equiv^\lambda_{\mathrm{PR}} q$. Let f_{PR} and $(X_i)_i$ be the function and sets used in the construction of the path refinement. Let n be minimal s.t. $(p, q) \notin X_n$. We use induction on n to prove the claim.

If $n = 0$, there is a word $w \in L_{\lambda\hookleftarrow}$ such that $\min\{c(\delta^*(p, u)) \mid u \sqsubseteq w\} \neq \min\{c(\delta^*(q, u)) \mid u \sqsubseteq w\}$. Let $x, y \in c(Q)$ be the third components of $(\delta^\lambda_{\mathrm{visit}})^*(\iota^{\hat{c}}_p, w)$ and $(\delta^\lambda_{\mathrm{visit}})^*(\iota^{\hat{c}}_q, w)$ respectively. By Lemma 0.1.6, $x = \min\{c(\delta^*(p, u)) \mid u \sqsubseteq w\}$ and $y = \min\{c(\delta^*(q, u)) \mid u \sqsubseteq w\}$, so $x \neq y$. By definition of $\mathcal{A}^\lambda_{\mathrm{visit}}$, this means that $((\delta^\lambda_{\mathrm{visit}})^*(\iota^{\hat{c}}_p, w), (\delta^\lambda_{\mathrm{visit}})^*(\iota^{\hat{c}}_q, w)) \notin V$.

For $n + 1 > 0$, there is a $w \in L_{\lambda\hookleftarrow}$ such that $(\delta^*(p, w), \delta^*(q, w)) \notin V$. Let $p' = \delta^*(p, w)$ and q' analogously. By induction, $(\iota^{\hat{c}}_{p'}, \iota^{\hat{c}}_{q'}) \notin V_M$. Lemma 0.1.6 tells us that there are k' and l' such that $(\delta^\lambda_{\mathrm{visit}})^*(\iota^{\hat{c}}_p, w) = \iota^{k'}_{p'}$ and $(\delta^\lambda_{\mathrm{visit}})^*(\iota^{\hat{c}}_q, w) = \iota^{l'}_{q'}$. Since n was chosen to be minimal, it must be true that $k' = l'$; otherwise, we would already have $p, q \notin X_0$. From Lemma 0.1.8, we know that $(\iota^{\hat{c}}_{p'}, \iota^{\hat{c}}_{q'}) \notin V_M$ if and only if $(\iota^{k'}_{p'}, \iota^{l'}_{q'}) \notin V_M$, which is false. Thus, finally, $(\iota^{\hat{c}}_p, \iota^{\hat{c}}_q) \notin V_M$. $\square$

The automaton has size $|\mathcal{A}^\lambda_{\mathrm{visit}}| \in \mathcal{O}(|Q| \cdot |c(Q)|^2)$ and the computation of V_M brings the runtime up to $\mathcal{O}(|\mathcal{A}^\lambda_{\mathrm{visit}}| \cdot \log |\mathcal{A}^\lambda_{\mathrm{visit}}|)$.