CSC444

October 26, 2014

Contents

1 Review: Relations

A *binary relation* on a set A is a subset of $A \times A$.

Example 1.1. Let $A = \{1, 2, 3, 4\}$ and $R = \{(1, 2), (1, 1), (3, 4), (4, 1)\}$.

Let A be a set. The following are two important properties a binary relation R on $A \times A$ can have:

- R is *reflexive* if $(a, a) \in R$ for each $a \in A$
- R is *transitive* if whenever $(a, b) \in R$ and $(b, c) \in R$ then $(a, c) \in R$

Definition 1.2. Let A be a set. The *reflexive, transitive closure* of a relation R on $A \times A$ is the relation $R^* = \{ (a, b) \in A \times A \mid$ there is a path from a to b in $R\}$.

Intuitively, R^* is the smallest possible relation that contains all the pairs of R and is reflexive and transitive.

Thus, to get R^* from R we need to add in just enough arcs to get both properties.

Example 1.3. Let $A = \{1, 2, 3, 4\}$ and let $R = \{(1, 2), (2, 3), (2, 4), (3, 4)\}$.

2 State diagrams

Example 2.1. $K = \{q_0, q_1, q_2, q_3\}$, $s = q_0$, $F = \{q_0, q_1, q_2\}$

δ is given by the following table:

q	σ	δ(q,σ)
q_0	0	q_0
q_0	1	q_1
q_1	0	q_0
q_1	1	q_2
q_2	0	q_0
q_2	1	q_3
q_3	0	q_3
q_3	1	q_3

$\mathcal{L}(M) = \{ w \mid w$ does not contain three consecutive 1's $\}$

Note. Is it possible to leave state q_3?

Definition 2.2. q_3 is called a *dead state* and if M reached q_3 it is said to be *trapped* since no further input can cause it to escape from the state.

3 Nondeterministic finite automata

We now want to introduce a useful notational generalization of a deterministic finite automaton.

Recall. In the definition of a DFA we required that δ be a function, that is, given an input symbol $a \in \Sigma$ and a state $q \in K$, $\delta(a, q)$ is uniquely determined.

We now introduce a new model very similar to a DFA except that δ is not required to be a function.

Instead it will be a *relation*.

How does this change the model? Consider an example.

3.1 An example

Let's suppose that we want to find a DFA M where $\mathcal{L}(M)$ is either $\{01\}*$ or $\{010\}*$, that is, $\mathcal{L}(M)$ is either a string of 01's or a string of 010's.

How can we design an automaton to accept this language?

It would be easy enough if we were asked to construct an automaton accepting either one:

- An automaton for $\{01\}*$.

  $M_1 = (K_1, \{0, 1\}, \delta_1, s_1, F_1)$ where $K_1 = \{q_0, q_1, q_2, q_3\}$, $s_1 = q_0$, $F_1 = \{q_0, q_3\}$ and δ_1 is given below:

q	σ	δ_1(q,σ)
q_0	0	q_1
q_0	1	q_2
q_1	0	q_2
q_1	1	q_3
q_2	0	q_2
q_2	1	q_2
q_3	0	q_1
q_3	1	q_2

- An automaton for { 010 }*.

  $M_2 = (K_2, \{0, 1\}, \delta, s_2, F_2)$ where $K_2 = \{p_0, p_1, p_2, p_3, p_4\}$, $s_2 = p_0$, $F_2 = \{p_0, p_4\}$ and δ_2 is given below:

q	σ	δ_2(q,σ)
p_0	0	p_1
p_0	1	p_2
p_1	0	p_2
p_1	1	p_3
p_2	0	p_2
p_2	1	p_2
p_3	0	p_4
p_3	1	p_2
p_4	0	p_1
p_4	1	p_2

If we consider these automata "subrountines", what we want, to answer the original question, is an automaton that "calls" these two.

How can we do that?

Design a new start state s that on input 0 enters both q_1 or p_1 and on input 1 enters either q_2 or p_2 (but not both).

Draw the composite automaton M.

Observe that this automaton is *not deterministic*. There are two possible transitions in the state s when reading a 0 or a 1.

A *string is accepted* by the automaton if there is some way to get from the initial state s to a final state while following arrows labeled with the symbols of the string.

Example 3.1. Let $w = 01010$. We can *simplify this automaton* if we do two things:

- Remove the dead states

  Just as we can have more than one transition with the same label out of a state, we may have no transition with a given label out of a state.

- Use arrows labeled by the empty string ε.

  If there is one of these transitions, called an ε-*transition*, between two states q and p, then you can move from q to p without consuming any input.

These can simplify M_1, M_2, and the composite automaton M as follows:

- The new M_1.

  $K_1 = \{q_0, q_1, q_3\}$, $s = q_0$, $F = \{q_0\}$ and δ is given below:

q	σ	δ(q,σ)
q_0	0	q_1
q_1	1	q_3
q_3	ε	q_0

- The new M_2.

  $K_2 = \{p_0, p_1, p_3, p_4\}$, $s = p_0$, $F = \{p_0\}$, and δ is given below:

q	σ	δ(q,σ)
p_0	0	p_1
p_1	1	p_3
p_3	0	p_4
p_4	ε	p_0

- The new M

  $K = K_1 \cup K_2 \cup \{s\}$ where $\delta(s, \varepsilon) = \{q_0, p_0\}$.

3.2 Definition of NFA

The generalization of finite automata we considered in the previous examples is called a nondeterministic finite automaton (NFA).

Nondeterminism is the ability to change state in a way that is only partially determined by the current state and the input symbol scanned.

Once nondeterminism is introduced, we have to change what we mean by acceptance of a string by the finite automaton.

Definition 3.2. A *nondeterministic finite automaton* is a quintuple $M = (K, \Sigma, \Delta, s, F)$ where

- K is the set of states,
- Σ is the alphabet,
- $s \in K$ is the starting state,
- $F \subseteq K$ is the set of final states, and
- $\Delta \subseteq K \times (\Sigma \cup \{\varepsilon\}) \times K$ is the transition relation.

Each triple $(q, u, p) \in \Delta$ is called a *transition* of M.

If M is in state q and the next input symbol to be read is a then M may follow any transition of the form (q, a, p) or (q, ε, p). If the latter is followed then the reading head does not move and a remains the next input symbol.

As with DFA, we can formally define what it means for a NFA to accept a string:

A *configuration* of M is again an element of $K \times \Sigma^*$.

The relation $\mapsto_M$ (*yields in one step*) between configurations is given by: $(q, w) \mapsto_M (q', w')$ iff $\exists u \in \Sigma \cup \{\varepsilon\}.\ w = uw'$ and $(q, u, q') \in \Delta$.

Note. $\mapsto_M$ is a relation and not a function, meaning that for a given (q, w) there may be several or no pairs (q', w') so that $(q, w) \mapsto_M (q', w')$.

$\mapsto_M^*$ is the *reflexive, transitive closure of* $\mapsto_M$, and a string $w \in \Sigma^*$ is *accepted by* M iff there is a state $q \in F$ s.t. $(s, w) \mapsto_M^* (q, \varepsilon)$.

As before the *language accepted by* M is $\mathcal{L}(M) = \{ w \in \Sigma^* \mid w \text{ is accepted by } M \}$.

Example 3.3.

q	σ	Δ(q,σ)
q_0	0	q_0, q_3
q_0	1	q_0, q_1
q_1	1	q_2
q_2	0	q_2
q_2	1	q_2
q_3	0	q_4
q_4	0	q_4
q_4	1	q_4

$\mathcal{L}(M) = \{ w \mid w$ has either two consecutive 0's or two consecutive 1's$\}$

Example 3.4. Let M be given by K = $\{ q_0, q_1, q_2 \}$, $\Sigma = \{ 0,1,2 \}$, s = q_0, F = $\{ q_2 \}$ with transition relation Δ given below:

q	σ	Δ(q,σ)
q_0	0	q_0
q_0	ε	q_1
q_1	1	q_1
q_1	ε	q_2
q_2	2	q_2

$\mathcal{L}(M) = \{0\}^*\{1\}^*\{2\}^*$

Example 3.5. Let $\Sigma = \{a_1, ..., a_n\}$ where $n \geq 2$. Suppose we want to create a NFA to accept the language $L = \{ w \mid \exists i.\ a_i \text{ does not appear in } w \}$.

For example, If $\Sigma = \{a_1, a_2, a_3\}$ then $a_1a_1a_2 \in \Sigma$ but $a_1a_2a_3 \notin \Sigma$.

Intuitively, the NFA would work in the following manner:

- Guess the symbol a_i that is missing from the input.
- If no symbol is missing, move to a dead state.
- If a symbol a_i is missing, go to state q_i.
- If in state q_i you ever encounter a_i, move to a dead state.
- Otherwise eat the remaining symbols and accept.

For the construction of the NFA we need one starting state q_0, one state for each symbol in the alphabet, $q_1, \ldots, q_n$ and a dead state q_{n+1}.

There are ε-transitions from q_0 into each of $q_1, \ldots, q_n$, self-loops on each of $q_1, \ldots, q_n$ labeled with the states that are legal, and a transition out of each of $q_1, \ldots, q_n$ into q_{n+1} labeled with the illegal state.

Note. Assume that if you ever encounter a symbol and there is no arc leading out of that symbol labeled with the current input symbol then the string is automatically rejected.

4 The relationship between DFA and NFA

Fact 4.1. *A DFA is just a special type of NFA.*

This is because in a DFA the transition relation just happens to be a function.

More formally, a NFA is deterministic iff there are no transitions of the form $(q, \varepsilon, p) in \Delta$ and for each $q \in K$ and $a \in \Sigma$ there is exactly one $p \in K$ s.t. $(q, a, p) \in \Delta$.

Corollary 4.2. *The set of languages accepted by DFA is a subset of the set of languages accepted by NFA.*

The surprising thing is that the reverse is also true.

The set of languages accepted by NFA is the same as the set of languages accepted by DFA.

Note. It is important that we have not bounded the space needed by FA in their definition or this would not be true. We will see that there are NFA that are exponentially smaller than any equivalent DFA. (Size is measured by the number of states of the FA).

Definition 4.3. Two finite automata M_1 and M_2 (deterministic or nondeterministic) are *equivalent* iff $\mathcal{L}(M_1) = \mathcal{L}(M_2)$.

Theorem 4.4. *For each nondeterministic finite automaton there is an equivalent deterministic finite automaton.*

We will prove this theorem by taking an arbitrary NFA, constructing an equivalent DFA and then proving that what we did worked.

Key idea: At any point in time we will view a NFA as occupying not a single state, but a set of states.

That set of states will be *all states reachable* from the initial state given the input consumed so far.

If we force our DFA to keep track of these super-states then if we reach the end of the string and our set of possible states includes at least one final state, then the NFA would have accepted the string. Otherwise it would have rejected the string.

Example 4.5. Consider the NFA from Example 1.

Substring read so far	Set of possible states
1	$\{ q_0, q_1 \}$
10	$\{ q_0, q_3 \}$
100	$\{ q_0, q_3, q_4 \}$
etc.	etc

Suppose that the NFA had n states to start out with. How *many possible states* could a DFA constructed in this manner have?

The set of states of the DFA is simply all possible subsets of the NFA, that is, the power set of the n states.

So the DFA can have as many as 2^n states.

Let $M = (K, \Sigma, \Delta, s, F)$ be an arbitrary NFA. The idea of the proof is to construct a DFA $M' = (K', \Sigma, \delta', s', F')$ equivalent to M.

The set of states of M' will be $K' = 2^K$, the power set of the set of states of M.

The set of final states of M' will consist of all those subsets of K that contain at least one final state of M.

Defining the transition function δ' is more complicated, but the basic idea is the following:

A move of M' on an input symbol $a \in \Sigma$ simulates a move of M on input symbol a, possibly followed by any number of ε-moves of M.

To formalize this we need a definition:

Definition 4.6. For any state $q \in K$, let $\mathcal{E}(q)$ be the set of all states of M that are reachable from state q without reading any input.

So $\mathcal{E}(q) = \{ p \in K \mid (q, \varepsilon) \mapsto_M^* (p, \varepsilon) \}$

Another way to state this is that $\mathcal{E}(q)$ is the closure of the set $\{q\}$ under the relation:

$$\{ (p, r) \mid (p, \varepsilon, r) \in \Delta \}$$

We can *compute* $\mathcal{E}(q)$ using the following algorithm:

- Initialize $\mathcal{E}(q) = \{q\}$
- while $\exists (p, \varepsilon, r) \in \Delta.\ p \in \mathcal{E}(q)$ and $r \notin \mathcal{E}(q)$ do:
- $\mathcal{E}(q) \Leftarrow \mathcal{E}(q) \cup \{r\}$

How much time does this algorithm take in the worst case?

Since there $|K|$ states and one state is added at each step, it will terminate after no more than $|K|$ steps.

Example 4.7. Consider the NFA M given by $K = \{q_0, q_1, q_2\}$, $\Sigma = \{0, 1, 2\}$, $s = q_0$, $F = \{q_2\}$ with transition relation Δ given below:

q	σ	Δ(q,σ)
q_0	0	q_0
q_0	ε	q_1
q_1	1	q_1
q_1	ε	q_2
q_2	2	q_2

Recall. $\mathcal{L}(M) = \{0\}^*\{1\}^*\{2\}^*$.

$\mathcal{E}(q_0) = \{q_0, q_1, q_2\}$, $\mathcal{E}(q_1) = \{q_1, q_2\}$ and $\mathcal{E}(q_2) = \{q_2\}$.

We can now formally give the DFA M'.

Let $M = (K, \Sigma, \Delta, s, F)$ be an arbitrary NFA. We will construct a DFA $M' = (K', \Sigma, \delta', s', F')$ equivalent to M. The DFA M' is given by:

- $K' = 2^K$,
- $s' = \mathcal{E}(s)$,
- $F' = \{ Q \subseteq K \mid Q \cap F \neq \emptyset \}$, and
- For each $Q \subseteq K$ and each symbol $a \in \Sigma$, define $\delta'(Q, a) = \cup\{ \mathcal{E}(p) \mid p \in K$ and $(q, a, p) \in \Delta$ for some $q \in Q \}$

Thus, $\delta'(Q, a)$ is the set of all states of M to which M can go by reading input a, possibly following some ε-transitions.

Example 4.8. Consider the NFA M given in the last example.

$\delta'(\{q_0\}, 0) = \mathcal{E}(q_0) = \{q_0, q_1, q_2\}$.

If we added an arc from q_0 to q_1 with label 0, then $\delta'(\{q_0\}, 0) = \mathcal{E}(q_0) \cup \mathcal{E}(q_1) = \{q_0, q_1, q_2\}$.

We still have to show that M' is deterministic and equivalent to M.

We will show that M' is equivalent to M after we understand the construction presented better.

4.1 The construction

The formal proof of correctness for the construction makes more sense if you have seen an example.

This is because the proof that any NFA has an equivalent DFA is *constructive*, which means that it not only tells you that an equivalent DFA exists, but it also gives an algorithm for finding one.

4.1.1 Unreachable states

If we are constructing an equivalent DFA M' for an NFA M with n states, the set of possible states for M' has size 2^n.

Some of these states, however, may have no effect on the computation of M'.

This is because any state that is not reachable from the starting state s' of M' is irrelevant.

We will ensure that when we apply the algorithm, we will only build the reachable states into the DFA.

This is done by:

- Creating the appropriate starting state s' for the DFA M'
- Adding a new state only when it is needed as the value of $\delta'(q, a)$ for some state $q \in K'$ already introduced and some $a \in \Sigma$

4.1.2 Example

Consider the NFA M given by $K = \{q_0, q_1, q_2\}$, $\Sigma = \{0, 1, 2\}$, $s = q_0$, $F = \{q_2\}$ with transition relation Δ given below:

q	σ	Δ(q,σ)
q_0	0	q_0
q_0	ε	q_1
q_1	1	q_1
q_1	ε	q_2
q_2	2	q_2

Recall. $\mathcal{L}(M) = 0^*1^*2^*$.

To construct a DFA M' equivalent to M we need to determine $\mathcal{E}(q)$ for each $q \in K$.

$\mathcal{E}(q_0) = \{q_0, q_1, q_2\}$

$\mathcal{E}(q_1) = \{q_1, q_2\}$

$\mathcal{E}(q_2) = \{q_2\}$

The starting state s' of M' is $\mathcal{E}(s) = \mathcal{E}(q_0) = \{q_0, q_1, q_2\}$.

All the transitions $(q, 0, p)$ for some $q \in s'$ s.t. $(q_0, 0, q_0)$

All the transitions $(q, 1, p)$ for some $q \in s'$ s.t. $(q_1, 1, q_1)$

All the transitions $(q, 2, p)$ for some $q \in s'$ s.t. $(q_2, 2, q_2)$

$\delta'(s', 0) = \mathcal{E}(q_0) = \{q_0, q_1, q_2\}$

$\delta'(s', 1) = \mathcal{E}(q_1) = \{q_1, q_2\}$

$\delta'(s', 2) = \mathcal{E}(q_2) = \{q_2\}$

Repeat this process for each new state introduced in the last step:

$\delta'(\{q_1, q_2\}, 0) = \emptyset$

$\delta'(\{q_1, q_2\}, 1) = \mathcal{E}(q_1) = \{q_1, q_2\}$

$\delta'(\{q_1, q_2\}, 2) = \mathcal{E}(q_2) = \{q_2\}$

$\delta'(\{q_2\}, 0) = \emptyset$

$\delta'(\{q_2\}, 1) = \emptyset$

$\delta'(\{q_2\}, 2) = \mathcal{E}(q_2) = \{q_2\}$

We now have to repeat the process for the new state introduced:

$\delta'(\emptyset, 0) = \delta'(\emptyset, 1) = \delta'(\emptyset, 2) = \emptyset$

The *resulting DFA* M' has $K' = \{\{q_0, q_1, q_2\}, \{q_1, q_2\}, \{q_2\}, \emptyset\}$, $s' = \{q_0, q_1, q_2\}$, $F = \{\{q_0, q_1, q_2\}, \{q_1, q_2\}, \{q_2\}\}$ and transition function δ' given below:

q	σ	δ′ (q,σ)
$\{ q_0, q_1, q_2 \}$	0	$\{ q_0, q_1, q_2 \}$
$\{ q_0, q_1, q_2 \}$	1	$\{ q_1, q_2 \}$
$\{ q_0, q_1, q_2 \}$	2	$\{ q_2 \}$
$\{ q_1, q_2 \}$	0	$\emptyset$
$\{ q_1, q_2 \}$	1	$\{ q_1, q_2 \}$
$\{ q_1, q_2 \}$	2	$\{ q_2 \}$
$\{ q_2 \}$	0,1	$\emptyset$
$\{ q_2 \}$	2	$\{ q_2 \}$
$\emptyset$	0,1,2	$\emptyset$

4.1.3 Example

Let $\Sigma = \{a_1, ..., a_n\}$ where $n \geq 2$. Suppose we want to create a NFA to accept the language $L = \{ w \mid \exists i.\ a_i \notin w \}$.

Sample: If $\Sigma = \{a_1, a_2, a_3\}$ then $a_1 a_1 a_2 \in \Sigma$ but $a_1 a_2 a_3 \notin \Sigma$.

Intuitively, the NFA would work in the following manner:

- Guess the symbol a_i that is missing from the input.
- If a symbol a_i is missing, go to state q_i.
- If in state q_i you ever encounter a_i, reject
- Otherwise consume the remaining symbols and accept.

For the construction of the NFA we need one starting state q_0 and one state for each symbol in the alphabet, $q_1, \ldots, q_n$.

There are ε-transitions from q_0 into each of $q_1, \ldots, q_n$, and self-loops on each of $q_1, \ldots, q_n$ labeled with the states that are legal.

What happens when we use the construction to produce a DFA accepting this language?

The equivalent DFA M' has initial state $s' = \mathcal{E}(q_0) = \{q_0, q_1, q_2, q_3, ..., q_n\}$.

Good news: Half of the possible states of M' are irrelevant and do not need to be used.

This includes:

- The state $\{q_0\}$ since it cannot be reached from s'
- Any state that contains some q_i for $i \neq 0$ but not q_0.

Bad news: Half of the possible states of M' are necessary.

All the states of the form $\{q_0\} \cup Q$ for some non-empty subset Q of $\{q_1, q_2, ..., q_n\}$ are reachable from s'

The depressing fact: No amount of optimization will eliminate any more states. The equivalent DFA requires 2^n states.

This is simply a language that does not have a small, equivalent DFA.

4.2 The formal proof

We now show that the DFA M' is equivalent to the original NFA M, thus completing the proof of the theorem. We do so by proving the following claim.

Claim 4.9. *For any string $w \in \Sigma^*$ and any states $p, q \in K$, $(q, w) \mapsto_M^* (p, \varepsilon)$ iff $(\mathcal{E}(q), w) \mapsto_{M'}^* (P, \varepsilon)$ for some set P containing p.*

Claim $\Rightarrow$ Theorem. Why?

To show that M and M' are equivalent, consider any string $w \in \Sigma^*$.

$w \in \mathcal{L}(M)$ iff $(s, w) \mapsto_M^* (f, \varepsilon)$ for some $f \in F$ (by definition) iff $(\mathcal{E}(s), w) \mapsto_{M'}^* (Q, \varepsilon)$ for some Q containing f (by the claim).

This means that $w \in \mathcal{L}(M)$ iff $(s', w) \mapsto_{M'}^* (Q, \varepsilon)$ for some $Q \in F'$. The latter condition is simply the definition of $w \in \mathcal{L}(M')$.

Proof. By induction on $|w|$.

Base: For $|w| = 0$, that is, $w = \varepsilon$, we must show that $(q, \varepsilon) \mapsto_M^* (p, \varepsilon)$ iff $(\mathcal{E}(q), \varepsilon) \mapsto_{M'}^* (P, \varepsilon)$ for some set P containing p.

$\Rightarrow$ Suppose $(q, \varepsilon) \mapsto_M^* (p, \varepsilon)$. Then $p \in \mathcal{E}(q)$ by definition of $\mathcal{E}(q)$. So $(\mathcal{E}(q), \varepsilon) \mapsto_{M'}^* (P, \varepsilon)$ for $P = \mathcal{E}(q)$.

$\Leftarrow$ Suppose $(\mathcal{E}(q), \varepsilon) \mapsto_{M'}^* (P, \varepsilon)$ for some set P containing p. Then since M' is deterministic, $P = \mathcal{E}(q)$. So $p \in \mathcal{E}(q)$. Thus $(q, \varepsilon) \mapsto_M^* (p, \varepsilon)$.

Inductive hypothesis: Suppose that the claim is true for all strings w of length k or less for some $k \geq 0$.

Inductive step: We will prove the claim for any string w of length $k + 1$. Let $w = va$ where $a \in \Sigma$ and $v \in \Sigma^*$.

$\Rightarrow$ Suppose that $(q, w) \mapsto_M^* (p, \varepsilon)$. Then there are states r_1 and r_2 s.t.

$(q, w) \mapsto_M^* (r_1, a) \mapsto_M (r_2, \varepsilon) \mapsto_M^* (p, \varepsilon)$

Now $(q, va) \mapsto_M^* (r_1, a)$ means that $(q, v) \mapsto_M^* (r_1, \varepsilon)$. Since $|v| = k$, by the inductive hypothesis, $(\mathcal{E}(q), v) \mapsto_{M'}^* (R_1, \varepsilon)$ for some set R_1 containing r_1.

Since $(r_1, a) \mapsto_M (r_2, \varepsilon)$ there is a triple $(r_1, a, r_2) \in \Delta$, and thus by the construction of M', $\mathcal{E}(r_2) \subseteq \delta'(R_1, a)$.

But since $(r_2, \varepsilon) \mapsto_M^* (p, \varepsilon)$, it follows that $p \in \mathcal{E}(r_2)$ and therefore $p \in \delta'(R_1, a)$.

Thus $(R_1, a) \mapsto_{M'} (P, \varepsilon)$ for some P containing p, and thus $(\mathcal{E}(q), va) \mapsto_{M'}^* (R_1, a) \mapsto_{M'} (P, \varepsilon)$ as required.

$\Leftarrow$ Suppose that $(\mathcal{E}(q), w) \mapsto_{M'}^* (P, \varepsilon)$ for some set P containing p.

This is equivalent to saying that $(\mathcal{E}(q), va) \mapsto_{M'}^* (R_1, a) \mapsto_{M'} (P, \varepsilon)$ for some P containing p and some R_1 s.t. $\delta'(R_1, a) = P$.

By definition of δ', $\delta'(R_1, a)$ is the union of all sets $\mathcal{E}(r_2)$ where for some state $r_1 \in R_1$, (r_1, a, r_2) is a transition of M.

Since $p \in P = \delta'(R_1, a)$, there is some r_2 s.t. $p \in \mathcal{E}(r_2)$, and, for some $r_1 \in R_1$, (r_1, a, r_2) is a transition of M.

Then $(r_2, \varepsilon) \mapsto_M^* (p, \varepsilon)$ by the definition of $\mathcal{E}(R_2)$. By the inductive hypothesis, $(q, v) \mapsto_M^* (r_1, \varepsilon)$.

Therefore, $(q, va) \mapsto_M^* (r_1, a) \mapsto_M (r_2, \varepsilon) \mapsto_M^* (p, \varepsilon)$ which means that $(q, w) \mapsto_M^* (p, \varepsilon)$ as required.

This proves the claim and the theorem. ? $\square$