Chapter 5

Myhill-Nerode

The last characterization of regular languages that we consider is given by the Myhill-Nerode theorem.

What to write here?

5.1 Definition

The following definitions (roughly following [19]) will lead us to the statement of the Myhill-Nerode theorem.

Definition 5.1.1. *The* ***equivalence class*** *of $u \in \Sigma^*$ w.r.t. $\equiv$ is the set of all v such that $u \equiv v$. It is denoted by $[u]_\equiv$.*

Definition 5.1.2. *$\equiv$ is of* ***finite index*** *if and only if the set of $\{[u]_\equiv \mid u \in \Sigma^*\}$ is finite.*

Due to the lack of native support for quotient types in Coq, we formalize equivalence relations of finite index as functions from Σ^* to a finite type X.

Definition 5.1.3. *Let $f : \Sigma^* \mapsto X$ be such a function. The relation $\equiv_f$ is defined as*

$$\{(u, v) \mid u, v \in \Sigma^* \wedge f(u) = f(v)\}.$$

For all $w \in \Sigma^*$, $f(w)$ can be seen as an equivalence class of $\equiv_f$.

It is easy to see that $\equiv_f$ is an equivalence relation. Furthermore, from the finiteness of F, it follows that $\equiv_f$ is of finite index.

Lemmas for this?

Definition 5.1.4. *Let f be as above. Let $x \in X$. $w \in \Sigma^*$ is a* ***representative*** *of x if and only if $f(w) = x$. We write $cr(x)$ to denote any representative of x.*

Our formalization of equivalence relations of finite support requires the function f to be surjective. Mathematically, this is not a restriction since empty equivalence classes can be disregarded. In Coq, however, it is required in order to be able to give a representative of every equivalence class.

```
Record Fin_Eq_Cls :=
  { fin_type : finType;
    fin_f :> word -> fin_type;
    fin_surjective : surjective fin_f }.
```

Myhill Relations

Definition 5.1.5. *Let $\equiv$ be an equivalence relation.*

*(i) $\equiv$ is **right congruent** if and only if for all $u, v \in \Sigma^*$ and $a \in \Sigma$,*

$$u \equiv v \Rightarrow u \cdot a \equiv v \cdot a.$$

*(ii) $\equiv$ **refines** L if and only if for all $u, v \in \Sigma^*$,*

$$u \equiv v \Rightarrow (u \in L \iff v \in L).$$

*(iii) $\equiv$ is of **finite index** if and only if it has finitely many equivalence classes, i.e.*

$$\{[u]_\equiv \mid u \in \Sigma^*\} \text{ is finite}$$

Definition 5.1.6. *An equivalence relation is a **Myhill relation**[1] if and only if it satisfies (i), (ii) and (iii) [19].*

Building on our formalization of equivalence relations of finite support, we only need to give formalizations of (i) and (ii).

```
Definition right_congruent {X} (f: word -> X) :=
    forall u v a, f u = f v -> f (rcons u a) = f (rcons v a).
Definition refining {X} (f: word -> X) :=
    forall u v, f u = f v -> u \in L = (v \in L).
Record Myhill_Rel :=
    { myhill_func :> Fin_Eq_Cls;
      myhill_congruent : right_congruent myhill_func ;
      myhill_refining : refining myhill_func }.
```

Myhill relations correspond to the equivalence relations defined as the pairs of words (u, v) whose runs on a DFA A end in the same state. These relations are right congruent, refine $\mathcal{L}(A)$ and of finite index as A has finitely many states. We will later give a formal proof of this.

[1] Myhill relations are commonly referred to as "Myhill-Nerode relations". In this thesis, it makes sense to split the concept of a Myhill relation from that of Nerode relation.

Nerode Relations

Definition 5.1.7. *Let $u, v \in \Sigma^*$. We say that u and v are **invariant under concatenation** w.r.t. L if and only if*

$$\forall w \in \Sigma^*.\ uw \in L \Leftrightarrow vw \in L.$$

We write $u \doteq_L v$ when u and v are invariant under concatenation w.r.t L.

Definition 5.1.8. *Let $\equiv$ be an equivalence relation. We say that $\equiv$ is a **weak Nerode relation** if and only if*

$$\forall u, v \in \Sigma^*.\ u \equiv v \implies u \doteq_L v.$$

```
Definition equal_suffix u v :=
        forall w, u++w \in L = (v++w \in L).
Definition imply_suffix {X} (f: word -> X) :=
        forall u v, f u = f v -> equal_suffix u v.
Record Weak_Nerode_Rel :=
        { weak_nerode_func :> Fin_Eq_Cls;
          weak_nerode_imply: imply_suffix weak_nerode_func }.
```

The notion of a weak Nerode relation is not found in the literature. We will later prove them weaker than Myhill relations, in the sense that every Myhill relation is also a weak Nerode relation.

Definition 5.1.9. *Let $\equiv$ be an equivalence relation. We say that $\equiv$ is a **Nerode relation**[2] if and only if*

$$\forall u, v \in \Sigma^*.\ u \equiv v \iff u \doteq_L v.$$

```
Definition equiv_suffix {X} (f: word -> X) :=
        forall u v, f u = f v <-> equal_suffix u v.
Record Nerode_Rel :=
        { nerode_func :> Fin_Eq_Cls;
          nerode_equiv: equiv_suffix nerode_func }.
```

5.2 Minimizing Equivalence Classes

We will prove that weak Nerode relations can be converted into Nerode relations. For this purpose, we employ the table-filling algorithm to find indistinguishable states under the Myhill-Nerode relation [14]. However, we do not rely on an automaton. In fact, we use the finite type X, i.e., the equivalence classes, instead of states.

[2] The Nerode relation is sometimes referred to as the "coarsest Myhill-Nerode relation".

Given a weak Nerode relation f, we construct a fixed-point algorithm. The algorithm initially outputs the set of equivalence classes that are distinguishable by the inclusion of their class representative in L. We call the corresponding predicate *dist* and define it such that We denote this initial set $dist_0$.

$$dist_0 := \{(x, y) \in F \times F \,|\, cr(x) \in L \Leftrightarrow cr(y) \notin L\}.$$

```
Definition distinguishable := [ fun x y => (inv f x) \in L != ((inv f y) \in L) ].
Definition distinct0 := [set x | distinguishable x.1 x.2 ].
```

To find more distinguishable equivalence classes, we have to identify equivalence classes that lead to distinguishable equivalence classes.

Definition 5.2.1. *We say that an equivalence class x **transitions** to y with $a \in \Sigma$ if and only if $f(cr(x) \cdot a) = y$ We denote y by $ext_a(x)$.*

Definition 5.2.2. *A pair of equivalence classes (x, y) **transitions** to (x', y') with a if and only if x transitions to x' with a and y transitions to y' with a. We denote (x', y') by $pext_a(x, y)$.*

The fixed-point algorithm tries to extend the set of distinguishable equivalence classes by looking for a pair of equivalence classes that transitions to a pair of distinguishable equivalence classes. Given a set of of equivalence classes *dist*, we define the set of distinguishable equivalence classes they transition to as

$$distinct_S(dist) := \{(x, y) \mid \exists a.\, pext_a(x, y) \in dist\}.$$

Definition 5.2.3. *Let dist be a subset of $X \times X$. We define unnamed such that*

$$unnamed(dist) := dist_0 \cup dist \cup distinct_S(dist).$$

```
Definition ext := [ fun x a => f ((inv f x) ++ [::a]) ].
Definition pext := [ fun x y => [ fun a => (ext x a, ext y a) ] ].
Definition distinctS (distinct : {set X*X}) :=
    [set (x,y) | x in X, y in X & [ exists a, pext x y a \in distinct ] ].
Definition unnamed distinct :=
      distinct0 :|: distinct :|: ( distinctS distinct ).
```

Lemma 5.2.1. *unnamed is monotone and has a fixed-point.*

Proof. Monotonicity follows directly from the monotonicity of $\cup$. The number of sets in $F \times F$ is finite. Therefore, *unnamed* has a fixed point.

$\square$

Let ***distinct*** be the fixed point of *unnamed*. We write ***equiv*** for the complement of *distinct* and denote it $\cong$. We denote *distinct* $\ncong$.

Lemma 5.2.2. $\cong$ *is an equivalence relation.*

Proof. It suffices to show that *distinct* is anti-reflexive, symmetric and ????. We do a fixed-point induction.

a name for this

1. For $unnamed(dist) = \emptyset$ we have anti-reflexivity, symmetry and ???? by the properties of of $\emptyset$.

2. For $unnamed(dist) = dist'$ we have *dist* anti-reflexive, symmetric and ????. The set of anti-reflexive, antisymmetric and ???? sets is closed under union. It remains to show that $dist_0$ and $distinct_S(dist)$ are anti-reflexive, symmetric and ????.

   too much ????

   $dist_0$ is anti-reflexive and symmetric by definition.

   $distinct_S(dist)$ can be seen as an intersection of a symmetric subset of $X \times X$ defined by $pext_a$ and the anti-reflexive, symmetric *dist*. Thus, $distinct_S(dist)$ is anti-reflexive and symmetric.

   The set of anti-reflexive and antisymmetric sets is closed under union. Therefore, $dist'$ is anti-reflexive and symmetric.

□

```
Lemma equiv_refl x: x ~= x.
Lemma equiv_sym x y:  x ~= y -> y ~= x.
Lemma equiv_trans x y z: x ~= y -> y ~= z -> x ~= z.
```

Lemma 5.2.3. *Let* $u, v \in \Sigma^*$. *If* $f(u) \cong f(v)$, *then* u *and* v *are invariant under concatenation, i.e.* $f(u) \cong f(v) \implies u \doteq_L v$.

Proof. Let $w \in \Sigma^*$. We then show the contraposition of the claim:

$$uw \in L \not\Leftrightarrow vw \in L \implies f(u) \ncong f(v).$$

We do an induction on w and generalize over u and v.

1. For $w = \varepsilon$ we have $u \in L \not\Leftrightarrow v \in L$ which gives us $(f(u), f(v)) \in dist_0$. Thus, $f(u) \ncong f(v)$.

2. For $w = aw'$ we have $uaw \in L \not\Leftrightarrow vaw \in L$. We have to show $f(u) \ncong f(v)$, i.e. $(f(u), f(v)) \in distinct$. By definition of *distinct*, it suffices to show $(f(u), f(v)) \in unnamed(distinct)$.

   For this, we prove $(f(u), f(v)) \in distinct_S(distinct)$. By $uaw \in L \not\Leftrightarrow vaw \in L$ we have $(f(cr(u)a), f(cr(v)a)) \in dist_0$.

It remains to show that $f(cr(u)a) \ncong f(cr(v)a)$ which we get by inductive hypothesis. For this, we need to show that $cr(u)aw \in L \not\Leftrightarrow cr(v)aw$.

By the properties of f, we get $cr(u)aw \in L \Leftrightarrow uaw \in L$ and $cr(v)aw \in L \Leftrightarrow vaw \in L$. Thus, $cr(u)aw \in L \not\Leftrightarrow cr(v)aw$.

□

Lemma 5.2.4. *Let $u, v \in \Sigma^*$. If $f(u) \ncong f(v)$, then u and v are **not** invariant under concatenation, i.e. $f(u) \ncong f(v) \implies u \not\doteq_L v$.*

Proof. We do a fixed-point induction.

1. For $unnamed(dist) = \emptyset$ we have $(f(u), f(v)) \in \emptyset$ and thus a contradiction.

2. For $unnamed(dist) = dist'$ we have $(f(u), f(v)) \in dist'$. We do a case distinction on $dist'$.

   (a) $(f(u), f(v)) \in dist_0$. We have $u \in L \not\Leftrightarrow v \in L$. Thus, $u \not\doteq_L v$ as witnessed by $w = \varepsilon$.

   (b) $(f(u), f(v)) \in dist$. By inductive hypothesis, $u \not\doteq_L v$.

   (c) $(f(u), f(v)) \in distinct_S(dist)$. We have $a \in \Sigma$ with $pext_a(f(u), f(v))) \in dist$. By inductive hypothesis, we get $cr(u)a \not\doteq_L cr(v)a$ as witnessed by $w \in \Sigma^*$ such that $cr(u)aw \in L \not\Leftrightarrow cr(v)aw \in L$.

   By the properties of f, we get $cr(u)aw \in L \Leftrightarrow uaw \in L$ and $cr(v)aw \in L \Leftrightarrow vaw \in L$. Thus, we have $u \not\doteq_L v$ as witnessed by aw.

□

Corollary 5.2.1. *Let $u, v \in \Sigma^*$. We have that*

$$f(u) \cong f(v) \iff u \doteq_L v.$$

```
Lemma equiv_equal_suffix u v: f u ~= f v -> equal_suffix L u v.
Lemma distinct_not_equal_suffix u v:
    f u ~!= f v ->
    exists w, u ++ w \in L != (v ++ w \in L).
Lemma equivP u v:
    reflect (equal_suffix L u v)
            (f u ~= f v).
```

Definition 5.2.4. *Let $w \in \Sigma^*$. We define*

$$f_{min}(w) := \{x \mid x \in X,\ f(w) \cong x\}.$$

Note that the domain of f_{min} is finite (since f is finite) and contains no empty sets (due to reflexivity of $\cong$).

Lemma 5.2.5. *f_{min} is surjective.*

Proof. Let $s \in dom(f_{min})$. There exists $x \in X$ such that $x \in s$ since $s \neq \emptyset$. We have $f(x) = f(cr(x))$ and therefore $f(x) \cong f(cr(x))$ by reflexivity of $\cong$. Thus, $cr(x)$ is a representative of s since $f_{min}(x) = f_{min}(cr(x)) = s$. □

Lemma 5.2.6. *For all $u, v \in \Sigma^*$ we we have*

$$f_{min}(u) = f_{min}(v) \iff f(u) \cong f(v).$$

Proof. "⇒" We have $f_{min}(u) = f_{min}(v)$ and thus $f(u) \cong f(v)$.

"⇐" We have $f(u) \cong f(v)$. Let $x \in X$. It suffices to show that $f(u) \cong x$ if and only if $f(v) \cong x$. This follows from symmetry and transitivity of $\cong$. □

Theorem 5.2.1. *f_{min} is a Nerode relation, i.e. f_{min} is surjective and for all $u, v \in \Sigma^*$ we have*

$$f_{min}(u) = f_{min}(v) \iff u \doteq_L v.$$

Proof. We have proven surjectivity in lemma 5.2.5. By lemma 5.2.6 we have $f_{min}(u) = f_{min}(v)$ if and only if $f(u) \cong f(v)$. By corollary 5.2.1 we have $f(u) \cong f(v)$ if and only if $u \doteq_L v$. Thus, $f_{min}(u) = f_{min}(v)$ if and only if $u \doteq_L v$. □

The formalization of f_{min} is slightly more involved than the mathematical construction. We first need to define the finite type of f_{min}'s domain, which we do by enumerating all possible values of f_{min}.

```
Definition equiv_repr x := [set y | x ~= y].
Definition X_min := map equiv_repr (enum (fin_type f)).
Definition f_min w := SeqSub _ (equiv_repr_mem (f w)).
```

We can then prove lemmas 5.2.5, 5.2.6 and theorem 5.2.1.

```
Lemma f_min_surjective: surjective f_min.
Lemma f_minP u v:
      reflect (f_min u = f_min v)
              (f u ~= f v).
Lemma f_min_correct: equiv_suffix L f_min.
```

Finally, we give a function to explicitly convert the weak Nerode relation f to a Nerode relation.

```
Definition f_min_fin : Fin_Eq_Cls :=
      {| fin_surjective := f_min_surjective |}.
Definition weak_nerode_to_nerode: Nerode_Rel L :=
      {| nerode_func := f_min_fin ;
         nerode_equiv := f_min_correct |}.
```

5.3 Finite Automata and Myhill-Nerode

We prove theorem **??** by proving it equivalent to the existence of an automaton that accepts L.

5.3.1 Finite Automata to Myhill-Nerode

5.3.2 Myhill-Nerode to Finite Automata