CS 360
Naomi Nishimura

Equivalence of languages

Note: Credit for many of the ideas and examples used in the context-free language section goes to J. Shallit, who developed an earlier handout on that material.

Throughout the course, you will be asked to prove statements of the form $L_1 \subseteq L_2$ for various languages L_1 and L_2, where the languages may be expressed using a recursive definition, a regular expression, a context-free grammar, a machine, or English. This handout is intended to give ideas about how to approach such problems.

The sections below are based on the way L_1 is expressed. In each case, our goal is to show that for every string $w \in L_1$, $w \in L_2$. We can think of constructing each proof out of two pieces, where the pieces depend on how L_1 and L_2 have been expressed. The two pieces consist of making sure that we are considering every string in L_1 and showing that a string is in L_2.

We first briefly consider various techniques used for each of these pieces and then give further details when at least one of the languages is described using a context-free grammar.

1 Considering every string in L_1

The goal here is to find a methodological way of ensuring that each string has been considered. This may entail considering various cases or using various proof methods, depending on how L_1 is described.

1.1 Various descriptions of L_1

English Depending on the language and how it is described, there may be structure inherent in the language that allows you to use one of the other methods listed below. As a default, you can also use proof by induction on the length of the string to consider each string in the language.

Recursive definitions When the language is defined using a recursive definition, using structural induction is a natural way of considering each string in the language. That is, you will first show that each string derived as a base case is in L_2 and then you will use structural induction to show that for each general case, a string formed from one or more smaller strings in L_1 is in L_2. Such a proof will make use of the assumption that the substrings in L_1 are also in L_2.

Machines Various strings in the language may be divided into cases in various ways, depending on the machine. Options include considering the accepting state reached or the use of specific transitions.

Context-free grammars When L_1 is defined by a grammar, in order to consider each string in L_1, we can use any of the following: induction on the length of the string (discussed above), induction on the length of the derivation of the string (discussed in Section 1.2, and using an invariant that holds for each sentential form (discussed in Section 1.3).

1.2 Using induction on the length of the derivation

In our first example, the grammar has a single variable. In our proof by induction, we use a statement that refers explicitly to the length of the derivation. As we are concerned with strings of terminals, we state the condition that $w \in T^*$.

Example 1: A grammar with a single variable
$L_1 = L(G)$ for G defined as $S \to aSb \mid ab$
$L_2 = \{a^i b^i \mid i \geq 1\}$
Basis: For each string $w \in T^*$ such that $S \Rightarrow^1 w$, $w \in L_2$.

The only such string is ab, which is clearly in L_2 for $i = 1$.
Induction hypothesis: For each string $w \in T^*$ such that $S \Rightarrow^k w$, $w \in L_2$.
Statement to prove: For each string $w \in T^*$ such that $S \Rightarrow^{k+1} w$, $w \in L_2$.
Inductive step:

We suppose we have a string $w \in T^*$ such that $S \Rightarrow^{k+1} w$. We know that the first step of the derivation must use the rule $S \to aSb$, since the derivation is of length greater than one. Hence, w was derived as $S \Rightarrow aSb \Rightarrow^k axb = w$. We can apply the induction hypothesis to x to conclude that $x \in L_2$, and hence $x = a^j b^j$ for some $j \geq 1$. This implies that $w = a^{j+1} b^{j+1}$, which is in L_2 for $i = j + 1$, completing the proof.

Our proof of Example 1 used the fact that we could find in w a substring x that was also derived from S. However, in more complex grammars with multiple variables, we might wish to apply the induction hypothesis to variables other than S. This typically results in two different strategies, depending on how the uses of variables are interconnected.

When it is easy to order variables in terms of dependence of variables on each other, it is possible to prove a series of statements about the variables in turn, using earlier results to prove later results. This suggests the following steps.

- Order the variables depending on what uses what.
- Create a series of statements about the variables.
- Prove the statements.

Example 2: A grammar with "orderable" variables
$L_1 = L(G)$ for G defined as
$S \to BC$
$B \to aBb \mid ab$
$C \to bCc \mid bc$
$L_2 = \{a^i b^{i+j} c^j \mid i, j \geq 1\}$
Order the variables depending on what uses what.

In this case, S depends on B and C, whereas B and C each only depends on itself. We can then choose either of the following orders: B, C, S or C, B, S.
Create a series of statements about the variables.

For each string $u \in T^*$ such that $B \Rightarrow^* u$, $u \in \{a^i b^i \mid i \geq 1\}$.
For each string $v \in T^*$ such that $C \Rightarrow^* v$, $v \in \{b^j c^j \mid j \geq 1\}$.
For each string $w \in T^*$ such that $S \Rightarrow^* w$, $w \in L_2$.

Prove the statements.

We omit the proofs of the first two statements, as the first is the same as Example 1 and the second is analogous.

To prove the third statement, we simply argue that any string derived from S uses the rule $S \to BC$, and hence w is of the form $S \Rightarrow BC \Rightarrow^* uC \Rightarrow^* uv$. We can then use the results of the first two statements to complete the proof.

Sometimes, however, there is interdependence between statements. In this case, the statements about the variables need to be proved all at once, using the technique of **mutual induction**. In mutual induction, each step (basis, induction hypothesis, statement to prove, inductive step) is used for all statements at the same time. This allows us to prove a statement about one variable using a statement that uses a shorter derivation for a different variable.

Example 3: A grammar with mutually dependent variables
$L_1 = L(G)$ for G defined as
$S \to aB \mid a$
$B \to bS \mid b$
$L_2 = L(\alpha)$ for $\alpha = (ab)^*(a + ab)$

We need a statement about B, namely, that for each string $x \in \{a, b\}^*$ such that $B \Rightarrow^* x$, $x \in L(\beta)$ for $\beta = (ba)^*(b + ba)$.

Basis:

1. For each string $w \in \{a, b\}^*$ such that $S \Rightarrow^1 w$, $w \in L(\alpha)$.
2. For each string $x \in \{a, b\}^*$ such that $B \Rightarrow^1 x$, $x \in L(\beta)$.

The only string over $\{a, b\}$ derivable from S in one step is a, which is clearly in $L(\alpha)$ (we choose no instances of ab and choose a instead of ab).

The only string over $\{a, b\}$ derivable from B in one step is b, which is clearly in $L(\beta)$ (we choose no instances of ba and choose b instead of ba).

Induction hypothesis:

1. For each string $w \in \{a, b\}^*$ such that $S \Rightarrow^k w$, $w \in L(\alpha)$.
2. For each string $x \in \{a, b\}^*$ such that $B \Rightarrow^k x$, $x \in L(\beta)$.

Statement to prove:

1. For each string $w \in \{a, b\}^*$ such that $S \Rightarrow^{k+1} w$, $w \in L(\alpha)$.
2. For each string $x \in \{a, b\}^*$ such that $B \Rightarrow^{k+1} x$, $x \in L(\beta)$.

Inductive step:

To prove the first statement, we observe that the first step in the derivation of w uses $S \to aB$, and hence the derivation is of the form $S \Rightarrow aB \Rightarrow^k ax$. We can apply the second statement of the induction hypothesis to x to conclude that $x \in L(\beta)$. This implies that w is in $L(\gamma)$ for $\gamma = a\beta = a(ba)^*(b + ba)$. We wish to show that $L(\gamma) \subseteq L(\alpha)$, which we do as follows.

Using identity 13 with $x = a$ and $b = y$, we know that $(ab)^*a = a(ba)^*$. Thus, $\gamma = a(ba)^*(b + ba) = (ab)^*a(b + ba)$. We now use identity 7, where $L = a$, $M = b$ and $N = ba$ to show that $a(b + ba) = ab + aba$, or $\gamma = (ab)^*(ab + aba)$. Using identity 8, we can rewrite this as $\gamma = (ab)^*(ab) + (ab)^*aba$. We consider each of the two pieces ($\gamma_1 = (ab)^*(ab)$ and $\gamma_2 = (ab)^*aba$); it suffices to show that $L(\gamma_1) \subseteq L(\alpha)$ and $L(\gamma_2) \subseteq L(\alpha)$. For γ_1, we can see that $L(\alpha)$ contains any string made up of zero or more strings ab followed by ab. We observe that γ_2 consists of one or more strings ab followed by a, which is clearly contained in the set of zero or more strings ab followed by a, or $(ab)^*a$, which is part of α.

To prove the second statement, we know that the first step in the derivation of x uses $B \to bS$, and hence the derivation is of the form $B \Rightarrow bS \Rightarrow^k bw$. We can apply the second statement of the induction hypothesis to w to conclude that $w \in L(\alpha)$. That implies that x is in $L(\delta)$ for $\delta = b\alpha = b(ab)^*(a + ab)$. We wish to show that $L(\gamma) \subseteq L(\delta)$, which we can do in a manner analogous to that given above.

1.3 Using an invariant on sentential forms

To use an invariant, we wish to find a property that is true for every sentential form and, for a sentential form in T^*, implies what we wish to prove (that is, that the sentential form is in L_2).

Example 4: Use of an invariant
$L_1 = L(G)$ for G defined as
$S \to AASB \mid AAB$
$A \to a$
$B \to bbb$
$L_2 = \{a^{2n}b^{3n} \mid n \geq 1\}$

Invariant attempt 1: If $S \Rightarrow^* \alpha$, then $3n_a(\alpha) = 2n_b(\alpha)$.

This is not true for every sentential form, such as aAB, which can be derived as $S \Rightarrow AAB \Rightarrow aAB$.

Invariant attempt 2: If $S \Rightarrow^* \alpha$, then $3(n_a(\alpha) + n_A(\alpha)) = 2(n_b(\alpha) + 3n_B(\alpha))$.

This is true for $bbbaa \in T^*$, which is not in L_2.

Invariant attempt 3: If $S \Rightarrow^* \alpha$, then $3(n_a(\alpha) + n_A(\alpha)) = 2(n_b(\alpha) + 3n_B(\alpha))$ and all a's and A's (if any) precede S (if any) which precedes all b's and B's (if any).

Use induction on the length of the derivation.

Basis: If $S \Rightarrow^0 \alpha$, then $\alpha = S$. Clearly, $3(n_a(\alpha) + n_A(\alpha)) = 2(n_b(\alpha) + 3n_B(\alpha))$ and, all a's and A's (if any) precede S (if any) which precedes all b's and B's (if any).

Induction hypothesis: If $S \Rightarrow^k \alpha$, then $3(n_a(\alpha) + n_A(\alpha)) = 2(n_b(\alpha) + 3n_B(\alpha))$ and, all a's and A's (if any) precede S (if any) which precedes all b's and B's (if any).

Statement to prove: If $S \Rightarrow^{k+1} \alpha$, then $3(n_a(\alpha) + n_A(\alpha)) = 2(n_b(\alpha) + 3n_B(\alpha))$ and, all a's and A's (if any) precede S (if any) which precedes all b's and B's (if any).

Inductive step:

Since $S \Rightarrow^{k+1} \alpha$, we know that $S \Rightarrow^k \beta \Rightarrow \alpha$, and by the induction hypothesis, β satisfies the condition.

We consider the following cases, depending on which rule was applied to transform β to α.

Case 1 The rule $S \to AASB$ was used to transform β to α.

We use the difference between the numbers of symbols in α and β and the induction hypothesis on β to show that $3(n_a(\alpha) + n_A(\alpha)) = 3(2 + n_a(\beta) + n_A(\beta)) = 6 + 3(n_a(\beta) + n_A(\beta)) = 6 + 2(n_b(\beta) + 3n_B(\beta)) = 2(n_b(\beta) + 3(n_B(\beta) + 1)) = 2(n_b(\alpha) + 3n_B(\alpha))$.

Moreover, since we know that the order of symbols in β matches the condition in the invariant, the same holds for α, as the new symbols follow the order in the condition.

Case 2 The rule $S \to AAB$ was used to transform β to α.

This proof is almost identical to that of Case 1.

Case 3 The rule $A \to a$ was used to transform β to α.

We use the fact that α and β differ only in the replacement of an A by an a and the induction hypothesis on β to show that $3(n_a(\alpha) + n_A(\alpha)) = 3((n_a(\beta) + 1) + (n_A(\beta) - 1)) = 3(n_a(\beta) + n_A(\beta)) = 2(n_b(\beta) + 3n_B(\beta)) = 2(n_b(\alpha) + 3n_B(\alpha))$

Since the only change from β to α is between A and a, both of which appear in the same section in the ordering condition, the fact that β satisfies the condition implies that α does as well.

Case 4 The rule $B \to bbb$ was used to transform β to α.

We use the fact that α and β differ only in the replacement of a B by bbb and the induction hypothesis on β to show that $3(n_a(\alpha) + n_A(\alpha)) = 3(n_a(\beta) + n_A(\beta)) = 2(n_b(\beta) + 3n_B(\beta)) = 2((n_b(\beta) + 3) + 3(n_B(\beta) - 1)) = 2(n_b(\alpha) + 3n_B(\alpha))$

Since the only change from β to α is between B and b's, both of which appear in the same section in the ordering condition, the fact that β satisfies the condition implies that α does as well.

2 Showing a string is in L_2

The requirements to show that a string w is in L_2 depend on how L_2 is described.

2.1 Various descriptions of L_2

Depending on the description of L_2, one can show that $w \in L_2$ by showing that w

- satisfies the conditions explained in English,
- satisfies a condition in the recursive definition,
- is in the language of strings accepted by the regular expression,
- can be derived using the grammar, or
- is accepted by the machine.

2.2 L_2 described as a context-free grammar

Here we use induction on the length of strings in L_1, with a statement such as "For each string $w \in L_1$, $S \Rightarrow^* w$."

Example 5:
$L_1 = \{a^i \mid i \geq 1\}$
$L_2 = L(G)$ for G defined as $S \to aS \mid a \mid bbS$
Basis: For each string $w \in L_1$ such that $|w| = 1$, $S \Rightarrow^* w$.

The only string in L_1 of length one is a, which can be derived as $S \Rightarrow a$.
Induction hypothesis: For each string $w \in L_1$ such that $|w| < k$, $S \Rightarrow^* w$.
Statement to prove: For each string $w \in L_1$ such that $|w| = k > 1$, $S \Rightarrow^* w$.
Inductive step: We suppose we have a string $w \in L_1$ such that $|w| = k > 1$. In particular, $w = a^k$. To show that we can derive w from S, we observe that by the induction hypothesis, since $a^{k-1} \in L_1$ and $|a^{k-1}| < k$, $S \Rightarrow^* a^{k-1}$.

We can then form the derivation $S \Rightarrow aS \Rightarrow^* aa^{k-1} = a^k$.

Notice that we did not use the production $S \to bbS$; remember that in this example we are only showing $L_1 \subseteq L_2$, not $L_2 \subseteq L_1$.

Example 6:
$L_1 = \{w \in \{a, b\}^* \mid n_a(w) > n_b(w)\}$
$L_2 = L(G)$ for G defined as $S \to a \mid aS \mid Sa \mid bSS \mid SSb \mid SbS$

Basis: For each string $w \in L_1$ such that $|w| = 1$, $S \Rightarrow^* w$.
Induction hypothesis: For each string $w \in L_1$ such that $|w| < k$, $S \Rightarrow^* w$.
Statement to prove: For each string $w \in L_1$ such that $|w| = k > 1$, $S \Rightarrow^* w$.
Inductive step: We suppose we have a string $w \in L_1$ such that $|w| = k > 1$.

Like in the previous example, we wish to find a substring of w that is in L_1 so that we can apply the induction hypothesis to the substring.

This raises the question of which production to use. The answer is to use every one that is needed to cover one of the possible cases of how w is structured. We choose the cases with the possible rules in mind.

Choosing cases, attempt 1

The rules $S \to aS$ and $S \to Sa$ might suggest that we wish to consider the following cases:

1. w starts with an a
2. w ends with an a
3. w does not start or end with an a

In each case, we want to be sure that we can find a smaller string in L_1 to which to apply the induction hypothesis.

However, if we remove an a from the start or end of a string that has more a's than b's, the resulting string may not have more a's than b's.

Choosing cases, attempt 2

Suppose instead we wish to use the rule $S \to bSS$. We then wish to show that if w starts with a b, then it is of the form bxy where x and y both are in L_1.

Since $n_a(w) - n_b(w) \geq 1$, if we remove b to form z (that is, $w = bz$), then $n_a(z) - n_b(z) \geq 2$. We can divide z into x and y such that each have at least one more a than b; we scan from left to right, "cutting" off x as soon as one more a than b has been seen. To see that the cut point must exist, we observe that the imbalance between number's of a and b's changes by one

upon seeing a new symbol, and since we start with an imbalance of zero and end up with an imbalance of at least two, there must be a point in between where the imbalance is one.

We then use the following cases:

1. w starts with a b
2. w ends with a b
3. w contains only a's
4. w contains at least one b but does not start or end with a b

For the second case we use an argument like that for the first case, but using $S \to SSb$.

For the third we use $S \to aS$ and $S \to \epsilon$.

We know that w has at least one b that can be used to rewrite w as ybz; by showing that both y and z are in L_1, we can use the induction hypothesis and the rule $S \to SbS$ to complete the proof. We consider two possible cases.

Case 4A The string of symbols up to but not including the last b is in L_1.

In this case, we can clearly use the last b to form y and z, where by assumption y is in L_1. Moreover, since z does not contain any b's but is of nonzero length (since w does not end with a b) z is also in L_1. We can now apply the induction hypothesis and use $S \to SbS$ to form a derivation of w.

Case 4B The string of symbols up to but not including the last b is not in L_1.

We know that "cutting" at the last b results in a choice of y that does not have an excess of a's. We also know that cutting at the first b results in a choice of y that has an excess of a's (since y contains no b's and, since w does not start with b, contains at least one a). Using the idea of scanning left to right, we wish to find a choice of b such that both y and z have an excess of at least one a, resulting in both y and z in L_1, as needed.

To find the correct choice of b, we find the leftmost b that creates a y that does not have an excess of a's (which could be the last b if there are no other such b's); let's call the y thus created y_{bad}. We then consider the b to the left of that b, and call the y created by splitting at that b y_{good}.

We know that y_{good} has an excess of a's, by the choice of b's. We also know that $n_b(y_{\text{bad}}) = 1 + n_b(y_{\text{good}})$ and that $n_a(y_{\text{bad}}) \geq n_a(y_{\text{good}})$, which together imply that the number of excess a's in y_{good} is at most one greater than the number of excess a's in y_{bad}, that is, at most 1. Putting these two facts together, we conclude that y_{good} has an excess of exactly one a, and thus the corresponding z has an excess of at least one a (since $w = ybz$ has an excess of at least one a), as required to complete the proof.