state $q$ at least once. If we subtract from $L_2$ all the languages $L(E_q^*)$ for $q$ in $Q$, then we have the accepting computations of $A$ that visit all the states. Call this language $L_4$. By Theorem 4.10 we know $L_4$ is also regular.

Our final step is to construct $L$ from $L_4$ by getting rid of the state components. That is, $L = h(L_4)$. Now, $L$ is the set of strings in $\Sigma^*$ that are accepted by $A$ and that visit each state of $A$ at least once during their acceptance. Since regular languages are closed under homomorphisms, we conclude that $L$ is regular.

**4.2.5 Exercises for Section 4.2**

**Exercise 4.2.1:** Suppose $h$ is the homomorphism from the alphabet \{0, 1, 2\} to the alphabet \{a, b\} defined by: $h(0) = a$; $h(1) = ab$, and $h(2) = ba$.

* a) What is $h(0120)$?

b) What is $h(21120)$?

c) If $L$ is the language $L(01^*2)$, what is $h(L)$?

d) If $L$ is the language $L(0 + 12)$, what is $h(L)$?

* e) Suppose $L$ is the language \{ababa\}, that is, the language consisting of only the one string $ababa$. What is $h^{-1}(L)$?

f) If $L$ is the language $L(abab\ast)$, what is $h^{-1}(L)$?

**Exercise 4.2.2:** If $L$ is a language, and $a$ is a symbol, then $L/a$, the quotient of $L$ and $a$, is the set of strings $w$ such that $wa$ is in $L$. For example, if $L = \{a, aab, baa\}$, then $L/a = \{e, ba\}$. Prove that if $L$ is regular, so is $L/a$.

**Hint:** Start with a DFA for $L$ and consider the set of accepting states.

**Exercise 4.2.3:** If $L$ is a language, and $a$ is a symbol, then $a\backslash L$ is the set of strings $w$ such that $aw$ is in $L$. For example, if $L = \{a, aab, baa\}$, then $a\backslash L = \{e, ab\}$. Prove that if $L$ is regular, so is $a\backslash L$. **Hint:** Remember that the regular languages are closed under reversal and under the quotient operation of Exercise 4.2.2.

**Exercise 4.2.4:** Which of the following identities are true?

a) $(L/a)a = L$ (the left side represents the concatenation of the languages $L/a$ and \{a\}).

b) $a(a\backslash L) = L$ (again, concatenation with \{a\}, this time on the left, is intended).

c) $(La)/a = L$.

d) $a\backslash(aL) = L$. 

successors under input symbol \( a_1 \) are indistinguishable. Then, the successors of those states on input \( a_2 \) are indistinguishable, and so on, until we conclude that \( p \) and \( q \) are indistinguishable.

Since \( N \) has fewer states than \( M \), there are two states of \( M \) that are indistinguishable from the same state of \( N \), and therefore indistinguishable from each other. But \( M \) was designed so that all its states are distinguishable from each other. We have a contradiction, so the assumption that \( N \) exists is wrong, and \( M \) in fact has as few states as any equivalent DFA for \( A \). Formally, we have proved:

**Theorem 4.26:** If \( A \) is a DFA, and \( M \) the DFA constructed from \( A \) by the algorithm described in the statement of Theorem 4.24, then \( M \) has as few states as any DFA equivalent to \( A \). □

In fact we can say something even stronger than Theorem 4.26. There must be a one-to-one correspondence between the states of any other minimum-state \( N \) and the DFA \( M \). The reason is that we argued above how each state of \( M \) must be equivalent to one state of \( N \), and no state of \( M \) can be equivalent to two states of \( N \). We can similarly argue that no state of \( N \) can be equivalent to two states of \( M \), although each state of \( N \) must be equivalent to one of \( M \)'s states. Thus, the minimum-state DFA equivalent to \( A \) is unique except for a possible renaming of the states.

\[
\begin{array}{c|cc}
 & 0 & 1 \\
\hline
\rightarrow A & B & A \\
B & A & C \\
C & D & B \\
D & E & A \\
E & D & F \\
F & G & E \\
G & F & G \\
H & G & D \\
\end{array}
\]

Figure 4.14: A DFA to be minimized

### 4.4.5 Exercises for Section 4.4

* **Exercise 4.4.1:** In Fig. 4.14 is the transition table of a DFA.
  
  a) Draw the table of distinguishabilities for this automaton.
  
  b) Construct the minimum-state equivalent DFA.

**Exercise 4.4.2:** Repeat Exercise 4.4.1 for the DFA of Fig 4.15.
!! Exercise 4.4.3: Suppose that \( p \) and \( q \) are distinguishable states of a given DFA \( A \) with \( n \) states. As a function of \( n \), what is the tightest upper bound on how long the shortest string that distinguishes \( p \) from \( q \) can be?

4.5 Summary of Chapter 4

- **The Pumping Lemma:** If a language is regular, then every sufficiently long string in the language has a nonempty substring that can be “pumped,” that is, repeated any number of times while the resulting strings are also in the language. This fact can be used to prove that many different languages are not regular.

- **Operations That Preserve the Property of Being a Regular Language:** There are many operations that, when applied to regular languages, yield a regular language as a result. Among these are union, concatenation, closure, intersection, complementation, difference, reversal, homomorphism (replacement of each symbol by an associated string), and inverse homomorphism.

- **Testing Emptiness of Regular Languages:** There is an algorithm that, given a representation of a regular language, such as an automaton or regular expression, tells whether or not the represented language is the empty set.

- **Testing Membership in a Regular Language:** There is an algorithm that, given a string and a representation of a regular language, tells whether or not the string is in the language.

- **Testing Distinguishability of States:** Two states of a DFA are distinguishable if there is an input string that takes exactly one of the two states to an accepting state. By starting with only the fact that pairs consisting