and are run in parallel. The same applies when we consider input strings that are not accepted by the given DFA. In such cases, we may need to modify the DFA to ensure that it accepts all input strings and rejects none that are not accepted.

We have seen that DFAs can recognize regular languages, but what about more complex languages? In the next section, we will explore the concept of context-free grammars and learn how they can be used to recognize languages that are not regular.

2.4.3 Non-context-free Languages

In this section, we will explore the concept of non-context-free languages, which are languages that cannot be recognized by any context-free grammar. These languages are more complex than regular languages and require more powerful computational models to recognize.

2.4.4 Applications of Text Searching

Text searching has numerous applications in various fields, including biology, computer science, and information retrieval. In this section, we will explore some of the key applications of text searching.

2.4.5 An Application: Text Searching

In this section, we will discuss an application of text searching in the field of bioinformatics. We will explore how text searching can be used to identify sequences of DNA or protein that are similar to a given query sequence.
Example 2.1: The construction of a DFA from the NFA of Fig. 2.16 is shown.
finite automata with epsilon-transitions

Exercise 2.4.2: Consider each of your NFA’s from Exercise 2.4.1 to try.

Exercise 2.4.1: Design an NFA to recognize the following set of strings.

2.4.4. Exercise for Section 2.4.

finite automata with epsilon-transitions

2.5 Finite Automata With Epsilon-Transitions

Example 2.16: In Fig. 2.16 is an NFA that accepts decimal numbers.