in element of itself. The satisfaction of the property is indicated by the complement of the diagonal. A set \( X_i \) is not an element of itself if, and only if, entry \([i, i]\) is 0.

Assume that \( S = \{X | X \not\in X\} \) is a set. Is \( S \) in \( S \)? If \( S \) is an element of itself, then it is not in \( S \) by the definition of \( S \). Moreover, if \( S \) is not in \( S \), then it must be in \( S \) since it is not in element of itself. This is an obvious contradiction. We were led to this contradiction by our assumption that the collection of sets that satisfy the property \( X \not\in X \) form a set.

We have constructed a describable property that cannot define a set. This shows that Cantor's assertion about the universality of sets is demonstrably false. The ramifications of Russell's paradox were far-reaching. The study of set theory moved from a foundation based in naive definitions to formal systems of axioms and inference rules and helped initiate the formalist philosophy of mathematics. In Chapter 12 we will use self-reference to establish a fundamental result in the theory of computer science, the undecidability of the Halting Problem.

### 1.6 Recursive Definitions

Many, in fact most, of the sets of interest in formal language and automata theory contain an infinite number of elements. Thus it is necessary that we develop techniques to describe, generate, or recognize the elements that belong to an infinite set. In the preceding section we described the set of natural numbers utilizing ellipsis dots (\ldots). This seemed reasonable since everyone reading this text is familiar with the natural numbers and knows what comes after 0, 1, 2, 3. However, this description would be totally inadequate for an alien unfamiliar with our base 10 arithmetic system and numeric representations. Such a being would have no idea that the symbol 4 is the next element in the sequence or that 1492 is a natural number.

In the development of a mathematical theory, such as the theory of languages or automata, the theorems and proofs may utilize only the definitions of the concepts of that theory. This requires precise definitions of both the objects of the domain and the operations. A method of definition must be developed that enables our friend the alien, or a computer that has no intuition, to generate and "understand" the properties of the elements of a set.

A recursive definition of a set \( X \) specifies a method for constructing the elements of the set. The definition utilizes two components: a basis and a set of operations. The basis consists of a finite set of elements that are explicitly designated as members of \( X \). The operations are used to construct new elements of the set from the previously defined members. The recursively defined set \( X \) consists of all elements that can be generated from the basis elements by a finite number of applications of the operations.

The key word in the process of recursively defining a set is generate. Clearly, no process can list the complete set of natural numbers. Any particular number, however, can be obtained by beginning with zero and constructing an initial sequence of the natural numbers. This intuitively describes the process of recursively defining the set of natural numbers. This idea is formalized in the following definition.
Following the construction given in Definition 1.6.2, the sum of any two natural numbers can be computed using 0 and s, the primitives used in the definition of the natural numbers. Example 1.6.1 traces the recursive computation of \(3 + 2\).

**Example 1.6.1**
The numbers 3 and 2 abbreviate \(s(s(s(0)))\) and \(s(s(0))\), respectively. The sum is computed recursively by

\[
\begin{align*}
& s(s(s(0))) + s(s(0)) \\
= & s(s(s(s(0))) + s(0)) \\
= & s(s(s(s(0))) + 0) \\
= & s(s(s(s(0)))) \quad \text{(basis case).}
\end{align*}
\]

This final value is the representation of the number 5. □

Figure 1.1 illustrates the process of recursively generating a set \(X\) from basis \(X_0\). Each of the concentric circles represents a stage of the construction. \(X_i\) represents the basis elements and the elements that can be obtained from them using a single application of an operation defined in the recursive step. \(X_i\) contains the elements that can be constructed with \(i\) or fewer operations. The generation process in the recursive portion of the definition produces a countably infinite sequence of nested sets. The set \(X\) can be thought of as the infinite union of the \(X_i’s\). Let \(x\) be an element of \(X\) and let \(X_j\) be the first set in which \(x\) occurs. This means that \(x\) can be constructed from the basis elements using exactly \(j\) applications of the operators. Although each element of \(X\) can be generated by a finite number of applications of the operators, there is no upper bound on the number of applications needed to generate the entire set \(X\). This property, generation using a finite but unbounded number of operations, is a fundamental property of recursive definitions.

The successor operator can be used recursively to define relations on the set \(\mathbb{N} \times \mathbb{N}\). The Cartesian product \(\mathbb{N} \times \mathbb{N}\) is often portrayed by the grid of points representing the ordered pairs. Following the standard conventions, the horizontal axis represents the first component of the ordered pair and the vertical axis the second. The shaded area in Figure 1.2(a) contains the ordered pairs \(\{i, j\}\) in which \(i < j\). This set is the relation LT, less than, that was described in Section 1.2.

**Example 1.6.2**
The relation LT is defined as follows:

i) **Basis:** \([0, 1] \in LT\).

ii) **Recursive step:** If \([m, n] \in LT\), then \([m, s(n)] \in LT\) and \([s(m), s(n)] \in LT\).

iii) **Closure:** \([m, n] \in LT\) only if it can be obtained from \([0, 1]\) by a finite number of applications of the operations in the recursive step.
Recursive generation of \( X \):
\[ X_0 = \{ x \mid x \text{ is a basis element} \} \]
\[ X_{i+1} = X_i \cup \{ x \mid x \text{ can be generated by } i + 1 \text{ operations} \} \]
\[ X = \{ x \mid x \in X_j \text{ for some } j \geq 0 \} \]

**FIGURE 1.1** Nested sequence of sets in recursive definition.

Using the infinite union description of recursive generation, the definition of LT generates the sequence \( LT_i \) of nested sets where
\[
\begin{align*}
LT_0 &= \{ [0, 1] \} \\
LT_1 &= LT_0 \cup \{ [0, 2], [1, 2] \} \\
LT_2 &= LT_1 \cup \{ [0, 3], [1, 3], [2, 3] \} \\
LT_3 &= LT_2 \cup \{ [0, 4], [1, 4], [2, 4], [3, 4] \} \\
& \quad \vdots \\
LT_i &= LT_{i-1} \cup \{ [j, i + 1] \mid j = 0, 1, \ldots, i \} \\
& \quad \vdots
\end{align*}
\]

The construction of LT shows that the generation of an element in a recursively defined set may not be unique. The ordered pair \( [1, 3] \in LT_2 \) is generated by the two distinct sequences of operations:

<table>
<thead>
<tr>
<th>Basis: ( [0, 1] )</th>
<th>( [0, 1] )</th>
<th>( [0, 1] )</th>
<th>( [0, 1] )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1: ( [0, s(1)] = [0, 2] )</td>
<td>( [s(0), s(1)] = [1, 2] )</td>
<td>( [s(0), s(1)] = [1, 2] )</td>
<td>( [s(0), s(2)] = [1, 3] )</td>
</tr>
<tr>
<td>2: ( [s(0), s(2)] = [1, 3] )</td>
<td>( [1, s(2)] = [1, 3] )</td>
<td>( [1, s(2)] = [1, 3] )</td>
<td>( [1, s(2)] = [1, 3] ).</td>
</tr>
</tbody>
</table>
Example 1.6.3

The shaded area in Figure 1.2(b) contains all the ordered pairs with second component 3, 4, 5, or 6. A recursive definition of this set, call it $X$, is given below.

i) Basis: $[0, 3], [0, 4], [0, 5], \text{ and } [0, 6]$ are in $X$.
ii) Recursive step: If $[m, n] \in X$, then $[s(m), n] \in X$.
iii) Closure: $[m, n] \in X$ only if it can be obtained from the basis elements by a finite number of applications of the operation in the recursive step.

The sequence of sets $X_i$ generated by this recursive process is defined by

$$X_i = \{[j, 3], [j, 4], [j, 5], [j, 6] \mid j = 0, 1, \ldots, i\}.$$

1.7 Mathematical Induction

Establishing relationships between the elements of sets and operations on the sets requires the ability to construct proofs that verify the hypothesized properties. It is impossible to prove that a property holds for every member in an infinite set by considering each element individually. The principle of mathematical induction gives sufficient conditions for proving that a property holds for every element in a recursively defined set. Induction uses the family of nested sets generated by the recursive process to extend a property from the basis to the entire set.