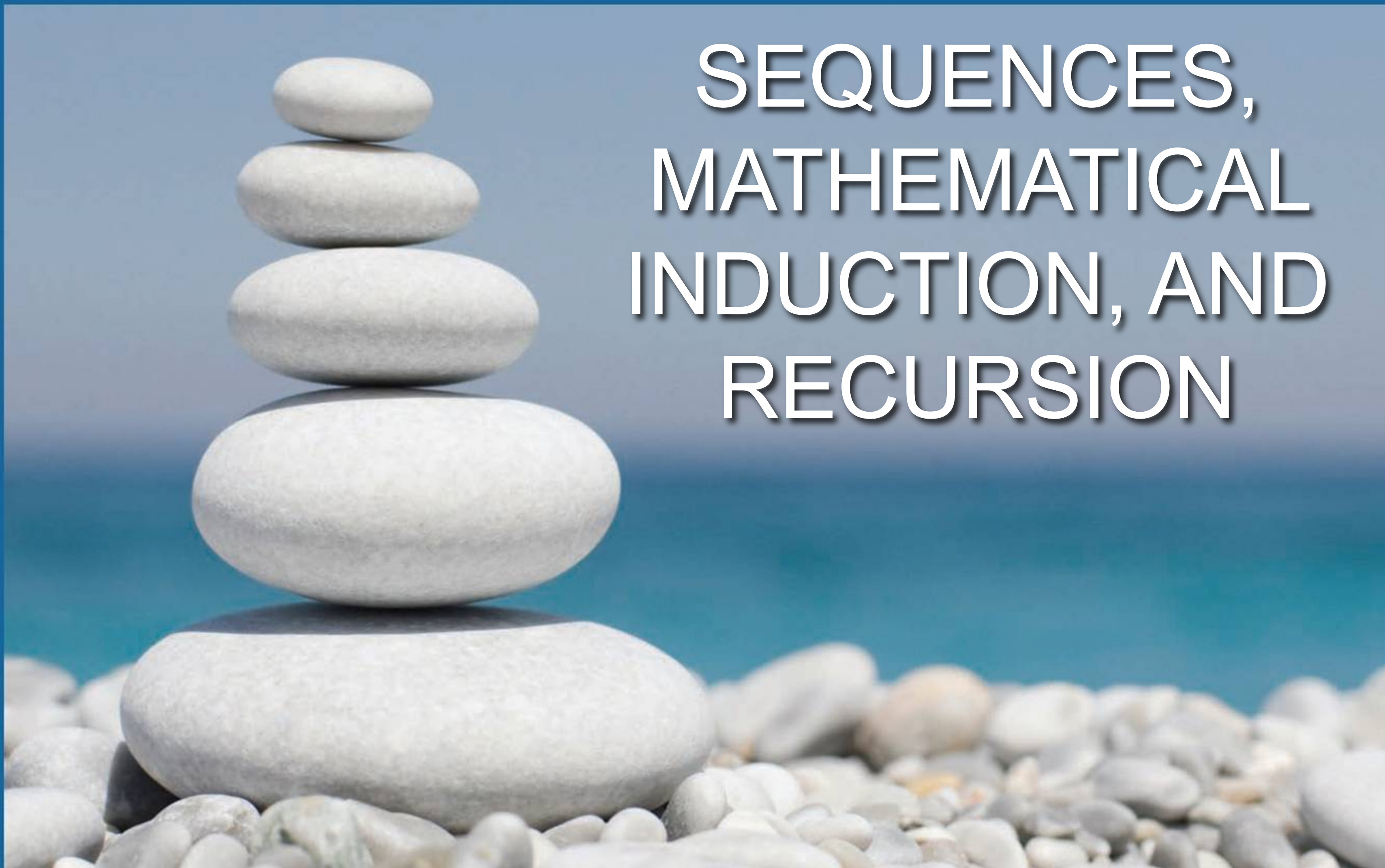
CHAPTER 5

SEQUENCES, MATHEMATICAL INDUCTION, AND RECURSION



Alessandro Artale – UniBZ - http://www.inf.unibz.it/~artale/

SECTION 5.9

General Recursive Definitions and Structural Induction

Recursively Defined Sets

Recursively Defined Sets

To define a set of objects recursively, you identify a few core objects as belonging to the set and give rules showing how to build new set elements from old.

A recursive definition for a set consists of the following:

I. BASE: A statement that certain objects belong to the set.

II. RECURSION: A collection of rules indicating how to form new objects from those already known to be in the set.

III. RESTRICTION: A statement that no objects belong to the set other than those defined in I and II.

Example 1 – *Recursive Definition of Boolean Expressions*

An expression composed of boolean variables, as letters from the alphabet L = {a, b, c, . . . , x, y, z}, and the connectives $\wedge$, $\vee$, and ~ is called a boolean expression and are defined recursively.

I. BASE: Each symbol of the alphabet L is a Boolean expression.

II. RECURSION: If P and Q are Boolean expressions, then so are

(**a**) $(P \wedge Q)$ and (**b**) $(P \vee Q)$ and (**c**) $\sim P$.

III. RESTRICTION: There are no Boolean expressions over the alphabet L other than those obtained from I and II.

Example 1 – *Recursive Definition of Boolean Expressions*

cont'd

Derive the fact that the following is a Boolean expression over the English alphabet $\{a, b, c, \ldots, x, y, z\}$:

$$(\sim(p \wedge q) \vee (\sim r \wedge p)).$$

(1) By **I**, p, q, and r are Boolean expressions.

(2) By (1) and **II**(**a**) and (**c**), $(p \wedge q)$ and $\sim r$ are Boolean expressions.

(3) By (2) and **II**(**c**) and (**a**), $\sim(p \wedge q)$ and $(\sim r \wedge p)$ are Boolean expressions.

(4) By (3) and **II**(**b**), $(\sim(p \wedge q) \vee (\sim r \wedge p))$ is a Boolean expression.

Example 4 – *Parenthesis Structures*

Certain configurations of parentheses in algebraic expressions are "legal" *[such as*(())() *and*()()()*]*, whereas others are not *[such as*)())) *and*()))(((*]*.

Here is a recursive definition to generate the set *P* of legal configurations of parentheses.

I. BASE: () is in *P*.

II. RECURSION:

a. If *E* is in *P*, so is (*E*).

b. If *E* and *F* are in *P*, so is *EF*.

Example 4 – *Parenthesis Structures* cont'd

III. RESTRICTION: No configurations of parentheses are in *P* other than those derived from **I** and **II** above.

Derive the fact that (())() is in *P*.

Solution:

(1) By **I**, () is in *P*.

(2) By (1) and **II**(**a**), (()) is in *P*.

(3) By (2), (1), and **II**(**b**), (())() is in *P*.

Proving Properties about Recursively Defined Sets

Proving Properties about Recursively Defined Sets

When a set has been defined recursively, a version of mathematical induction, called **structural induction**, can be used to prove that every object in the set satisfies a given property.

Proving Properties about Recursively Defined Sets

Structural Introduction for Recursively Defined Sets

Let S be a set that has been defined recursively, and consider a property that objects in S may or may not satisfy. To prove that every object in S satisfies the property:

1. Show that each object in the BASE for S satisfies the property;
2. Show that for each rule in the RECURSION, if the rule is applied to objects in S that satisfy the property, then the objects defined by the rule also satisfy the property.

Because no objects other than those obtained through the BASE and RECURSION conditions are contained in S, it must be the case that every object in S satisfies the property.

Example 5 – *A Property of the Set of Parenthesis Structures*

Consider the set *P* of all grammatical configurations of parentheses defined in Example 4. Prove that every configuration in *P* contains an equal number of left and right parentheses.

Solution:

Proof (by structural induction):

Given any parenthesis configuration, let the property be the claim that it has an equal number of left and right parentheses.

Show that each object in the BASE for *P* satisfies the property: The only object in the base for *P* is (), which has one left parenthesis and one right parenthesis, so it has an equal number of left and right parentheses.

Example 5 – *Solution*

cont'd

Show that for each rule in the RECURSION for *P*, if the rule is applied to an object in *P* that satisfies the property, then the object defined by the rule also satisfies the property: The recursion for *P* consists of two rules denoted II(a) and II(b).

Suppose *E* is a parenthesis configuration that has an equal number of left and right parentheses.

When rule II(a) is applied to *E*, the result is (*E*), so both the number of left parentheses and the number of right parentheses are increased by one.

Since these numbers were equal to start with, they remain equal when each is increased by one.

Example 5 – *Solution*

cont' d

Suppose E and F are parenthesis configurations with equal numbers of left and right parentheses. Say E has m left and right parentheses, and F has n left and right parentheses.

When rule II(b) is applied, the result is EF, which has an equal number, namely $m + n$, of left and right parentheses.

Thus when each rule in the RECURSION is applied to a configuration of parentheses in P with an equal number of left and right parentheses, the result is a configuration with an equal number of left and right parentheses.

Therefore, every structure in P has an equal number of left and right parentheses.