算术乘法定义方式的推广

参赛队员：赵易非　指导老师：肖恩利　学校：复旦大学附属中学

Generalization of the Definition Method of the Arithmetic Multiplication

Abstract

According to Van der Waerden, the sum of two natural numbers is defined as

$x\cdot 1=x$

$x\cdot y^{+}=x\cdot y+x$ (for every x and every y)

The product of two natural numbers is recursively defined as

$x\cdot 1=x$

$x\cdot y^{+}=x\cdot y+x$ (for every x and every y)

$x^{+}(y^{+})$ here means the successor(consequent) of $x(y)$ in the set of natural numbers. (*Algebra*)

Apparently, the *product* here uniquely depends on how the *sum* works. The generalization of this *product* is thus based on the substitution of addition, the *sum* here. Then the question arose. What result will emerge if we change the addition of natural numbers into a normal arithmetic binary operation?

Inspired by the substitution of original addition, we are going to define an order of arithmetic binary operation in the following text (Sometimes we call the original addition of natural numbers of 1-order, and therefore the multiplication of 2-order and so on. Actually it is an instinctive perception rather than a well-defined mathematical theory. However, to change the addition into a binary operation and generalize the multiplication provides a tool to clarify this so-called order. If we call one binary operation of i-order, we may treat it as an addition and hence define a multiplication according to the mentioned recursive definition and then call it (i+1)-order. By using the same method, we can define an operation of (i+2)-order based on the former (i+1)-order one.) and then concentrate our attention on the properties of the operations we create by using the generalized multiplication definition.

What is really surprising is if the order of a binary operation is equal to or larger than 2, the commutation, association, and a particular distribution of the binary operation require one same property of its lower-ordered binary operation, and this property leads the lower-ordered operation to be of exactly 1-order. The bulk of the following text is composed of these theorems and their extrapolations.

Furthermore, a discussion on how the demonstrated results can work on the sequence of objects (not only natural numbers) would be presented in the last paragraph.

The motivation of creating this whole paper is from pure mathematical curiosity. Thereby, the whole text is, at least presently, lack of applications.

ZHAO Yi-Fei

In modern mathematics, multiplication in natural numbers(the set of natural numbers doesn’t include 0 in this thesis) has a definition as following

For every x and every y

$$x\cdot 1 = x$$

$$x\cdot y^{+} = x\cdot y + x$$

Here y^{+} means the successor of y

Actually, we can define a new binary operation for every binary operation in natural numbers. We will no longer use the symbol of addition(or multiplication) in this thesis in order to avoid the ambiguity.

Definition 1: for a binary operation in natural numbers $\left[a,b\right]_i$, a new binary operation $\left[a,b\right]_{i+1}$ can be defined as

$$[a,1]_{i+1} = a$$

$$[a,b+1]_{i+1} = [[a,b]_{i+1},a]_i$$

$$\left(\forall a,b\in N\right)$$

We call the binary operation $\left[a,b\right]_{i+1}$ is **one-order higher** than $\left[a,b\right]_i$ ($\left[a,b\right]_i$ is **one-order lower** than $\left[a,b\right]_{i+1}$), while the subscript is one of the orders of the binary operation(it can have several orders). Thus we can define a binary operation **two-orders higher(lower)** than a given one as an operation one-order higher(lower) than “an operation one-order higher(lower) than the given one”. In this manner, we will reasonably have an operation **n-orders higher(lower)** than a given binary operation.

(Note that we have defined “one-order higher” in this recursive manner, but not the order of a binary operation, which we will discuss below.)

If a given binary operation is one-order higher than no operations, it is said to be of **one-order**.
A binary operation has an order *i*, if and only if it is (*i*-1)-orders higher than a one-order binary operation.

It is easy to obtain that a one-order binary operation has the only order: one.

Sometimes, we could find the one-order higher operation of a given operation by mathematical induction.

Examples:

If $\left[a,b\right]_i = a\cdot b$ then $\left[a,b\right]_{i+1} = a^b$

If $\left[a,b\right]_i = a\cdot b + a + b$ then $\left[a,b\right]_{i+1} = \left(a+1\right)^b - 1$

Obviously, definition 1 is a recursive one. It is possible to give a definition equivalent to definition 1 which includes ellipsis, but because of which, it is not so rigid.

Definition 2:

$$[a,b]_{i+1}=\begin{cases}[...[[a,a]_i,a]_i...,a]_i(b\neq 1)\\ \\ a(b=1)\end{cases}$$

Of course it satisfies $\begin{cases}[a,1]_{i+1}=a\\ [a,b+1]_{i+1}=[[a,b]_{i+1},a]_i\end{cases}$ $\left(\forall a,b\in N\right)$

On the other hand, if definition 1 is true, let $b=2$ we have $[a,2]_{i+1}=[a,a]_i$

If $[a,n]_{i+1}=[...[[a,a]_i,a]_i...,a]_i$ is true when $b=n$, then let $b=n+1$ we obtain

$$[a,n+1]_{i+1}=[[a,n]_{i+1},a]_i=[[...[[a,a]_i,a]_i...,a]_i,a]_i=[...[[a,a]_i,a]_i...,a]_i$$

Hence definition 2 is equivalent to definition 1. Definition 2 shows that a one-order higher operation expresses the times an element operates with itself by the lower-ordered binary operation.

If a binary operation is not one-order, it is possible to find all of its one-order lower binary operations by the following method:

Let $b=1$ in equation $[a,b+1]_{i+1}=[[a,b]_{i+1},b]_i$, then $[a,a]_i(\forall a\in N)$ is known.

Let $b=2,3,...$ then all expressions which can be written as $[...[[a,a]_i,a]_i...,a]_i$ is known,

while expressions that could not be written as $[...[[a,a]_i,a]_i...,a]_i$ (which remain unknown) is useless in terms of deciding $[a,b]_{i+1}$. So, the results of these expressions can be chosen arbitrarily or even not defined, which means sometimes a binary operation $f:D\to N(D\subseteq N\times N)$ also has its one-order higher operation.

The significance of introducing the orders of binary operations is to study the properties of one operation by another binary operation which is related to the original one in the orders.

Since one given binary operation can define an operation one-order higher than itself, this given operation must decide all the properties of the one-order higher one. In the next part, 3 theorems about how this "decision" works will be given.

We are going to talk about operations under one condition.

Condition 1: There **doesn't** exist 2 different numbers $a,b\in N$ which satisfy: for $\forall n\in N$, binary operation $[x,y]_i$ satisfies$[n,a]_i=[n,b]_i$

Since the choice of *n* is arbitrary, condition 1 is quite “weak”. It helps us to exclude some irregular binary operations like $[a,b]_i=a$

Theorem 1: If a binary operation $[x,y]_i$ is **commutative** and is not one-order, then $[n,1]_{i-1}=n+1\left(\forall n\in N\right)$

Proof: in formula $[a,b]_i=[b,a]_i\left(\forall a,b\in N\right)$

Let $a=1$, $b=n\left(\forall n\in N\right)$.

Regarding $[a,1]_i=a$, we have $[1,n]_i=[n,1]_i=n$

Let both sides of the equation operate with 1 in (*i*-1)-order, $[[1,n]_i,1]_{i-1}=[n,1]_{i-1}$

Again using the definition and $[1,n]_i=n$, we have $[[1,n]_i,1]_{i-1}=[1,n+1]_i=n+1$

So, $[n,1]_{i-1}=n+1\left(\forall n\in N\right)$

Theorem 2: If a binary operation $[x,y]_i$ **satisfying condition 1** is **associative** and is not one-order, then $[n,1]_{i-1}=n+1\left(\forall n\in N\right)$

Proof: in formula $[a,[b,c]_i]_i=[[a,b]_i,c]_i\left(\forall a,b,c\in N\right)$

Let $b=1,c=n\left(\forall n\in N\right)$

We have $[a,[1,n]_i]_i=[[a,1]_i,n]_i\left(\forall a,n\in N\right)$

Because binary operation $[x,y]_i$ is not one-order, $[a,1]_i=a$

Thus, for $\forall a\in N$, $[a,[1,n]_i]_i=[a,n]_i$. Regarding condition 1, $[1,n]_i=n$

Let both sides of the equation operate with 1 in (*i*-1)-order, $[[1,n]_i,1]_{i-1}=[n,1]_{i-1}$

Using the definition and $[1,n]_i=n$, $[[1,n]_i,1]_{i-1}=[1,n+1]_i=n+1$

So $[n,1]_{i-1}=n+1\left(\forall n\in N\right)$

Theorem 3: If a binary operation $[x,y]_i$ **satisfying condition 1** is **left-distributive to its one-order lower binary operation** and is not one-order, then $[n,1]_{i-1}=n+1\left(\forall n\in N\right)$

Proof: in formula $[a,[b,c]_{i-1}]_i=[[a,b]_i,[a,c]_i]_{i-1}\left(\forall a,b,c\in N\right)$

Let $c=1,b=n\left(\forall n\in N\right)$

We have $[a,[n,1]_{i-1}]_i=[[a,n]_i,[a,1]_i]_{i-1}\left(\forall a,n\in N\right)$

Because binary operation $[x,y]_i$ is not one-order, $[a,1]_i=a$

The former equation becomes $[a,[n,1]_{i-1}]_i=[[a,n]_i,a]_{i-1}$

Using the definition $[[a,n]_i,a]_{i-1}=[a,n+1]_i$, $[a,[n,1]_{i-1}]_i=[a,n+1]_i\left(\forall a,n\in N\right)$

Regarding condition 1 , we finally have $[n,1]_{i-1}=n+1\left(\forall n\in N\right)$

In these 3 theorems, we may find that under condition 1, no matter a not one-order binary operation $[x,y]_i$ is **commutative** or **associative** or **left-distributive to its one-order lower binary operation**, a common prerequisite $[n,1]_{i-1}=n+1\left(\forall n\in N\right)$ must hold.

Considering the definition of addition of natural numbers,

$$\begin{matrix} x+1=x^{+} \\ x+y^{+}=\left(x+y\right)^{+} \end{matrix}\left(\forall x,y\in N\right)$$

It is easy to discover that $[n,1]_{i-1}=n+1\left(\forall n\in N\right)$ is exactly the first part of addition. Thus we could define binary operations with this property as following:

Definition 3:A binary operation $[x,y]_i$ is called a **semi-addition**, if and only if it satisfies $[n,1]_i=n+1\left(\forall n\in N\right)$

Since it is proved that semi-addition plays an important role in deciding the property of its one-order higher binary operation, it is quite natural for us to study the properties of a semi-addition itself.

Theorem 4:Semi-addition is a one-order binary operation.

Proof: If there exists a binary operation $[x, y]_i$ which has its one-order higher operation $[x, y]_{i+1}$ be a semi-addition, then $[n,1]_{i+1} = n+1\left(\forall n \in N\right)$, which is contradictive to definition 1: $[n,1]_{i+1} = n$. Hence $[x, y]_i$ does not exist.

So, semi-addition is a one-order binary operation.

Theorem 5:Addition is equivalent to a semi-addition which is associative

Proof: Addition is obviously a semi-addition which is also associative.

On the other hand, if semi-addition $[x, y]_1$ is associative, first we have

$$[x,1]_1 = x+1\left(\forall x \in N\right)$$

And if $[x,n]_1 = x+n\left(\forall x \in N\right)$ holds for $y = n$, then for $y = n+1$

$$[x,n+1]_1 = [x,[n,1]_1]_1 = [[x,n]_1,1]_1 = [x+n,1]_1 = x+n+1\left(\forall x \in N\right)$$

By using mathematical induction,

we have $[x, y]_1 = x+y\left(\forall x, y \in N\right)$, which is equivalent to addition of natural numbers.

Since addition of natural numbers is a semi-addition, the multiplication of natural numbers, which is one-order higher than addition, has an order 2. The exponential has an order 3, while the binary operation which is one-order higher than exponential is $[a,b]_4 = a^{a^{b-1}}$

Because

$$[a,1]_4 = a^{a^0} = a$$

$$[a,b+1]_4 = a^{a^b} = (a^{a^{b-1}})^a = [[a,b]_4, a]_3$$

$$\left(\forall a,b \in N\right)$$

The most interesting property of this 4-order operation is that it can be expressed by a combination of its lower-order binary operations, which does not hold for multiplication, exponential or all its higher-order binary operations.

Here we have several beautiful extrapolations derived from former theorems:

Corollary: if a binary operation(in natural numbers) has **an** order larger than 2, it will never be associative or commutative or left-distributive to its one-order lower binary operation.

This corollary, together with the first 3 theorems, offers us a new way to prove a given binary operation does not satisfy certain property through its order.

Corollary: if a given 2-order binary operation(in natural numbers) is commutative, or associative, or left-distributive to its one-order lower binary operation, and this one-order lower binary operation is associative, then this given operation is multiplication.

Besides natural numbers, we can also use this theory to solve some problems about sequences(not only number sequences).
It is quite apparent that natural numbers have the same algebraic structure with sequences, but it is necessary to construct our theory in sequences clearly because the isomorphism here is from sets to sets.

On sequence $\left\{x_i\left(i\in N\right)\right\}$,an binary operation $[x_i,x_j]_p$ is given.We can define its one-order higher operation as:

$$\begin{array}{l}[x_i,x_1]_{p+1}=x_i\\ [x_i,x_{j+1}]_{p+1}=[[x_i,x_j]_{p+1},x_i]_p\end{array}\left(\forall i,j\in N\right)$$

A mapping $f:x_i\to i$ can be defined from this sequence to the set of natural numbers.

Define the operation of subscripts. $[i,j]_p=k$, if and only if $[x_i,x_j]_p=x_k$

It will be proved that $[x_i,x_j]_{p+1}=x_{[i,j]_{p+1}}\left(\forall i,j\in N\right)$

First $[x_i,x_1]_{p+1}=x_i=x_{[i,1]_{p+1}}\left(\forall i\in N\right)$ holds for $j=1$

If $[x_i,x_n]_{p+1}=x_{[i,n]_{p+1}}\left(\forall i\in N\right)$ holds for $j=n$

Then for $j=n+1$,

$$[x_i,x_{n+1}]_{p+1}=[[x_i,x_n]_{p+1},x_i]_p=[x_{[i,n]_{p+1}},x_i]_p=x_{[[i,n]_{p+1},i]_p}=x_{[i,n+1]_{p+1}}\left(\forall i\in N\right)$$

By mathematical induction, we obtain:

For $\forall i,j\in N$, $[x_i,x_j]_{p+1}=x_{[i,j]_{p+1}}$

Thus the one-to-one mapping *f* satisfies both $[f(x_i),f(x_j)]_p=f([x_i,x_j]_p)$ and

$[f(x_i),f(x_j)]_{p+1}=f([x_i,x_j]_{p+1})$

which means this mapping is an isomorphism.

It’s necessary for one to be familiar with common binary operations’ one-order higher(or lower) binary operations if he/she wants to use the “order theory” to solve problems. The following table about common binary operations’ one-order higher operations is thus given.

$[a,b]_i=a+b(\operatorname{mod}k)$

$$[m,n]_{i+1} = mn(\operatorname{mod} k)$$

$$[a,b]_i = ab(\operatorname{mod} k)$$

$$[m,n]_{i+1} = m^n(\operatorname{mod} k)$$

$$[a,b]_i = ab + a + b$$

$$[m,n]_{i+1} = (m+1)^n - 1$$

$$[a,b]_i = \lambda a + f(b)$$

$$[m,n]_{i+1} = \lambda^{n-1} m + \frac{1-\lambda^{n-1}}{1-\lambda} f(m)$$

$$[a,b]_i = \lambda a f(b)$$

$$[m,n]_{i+1} = \lambda^{n-1} m f^{n-1}(m)$$

$$[a,b]_i = \lambda a^{f(b)}$$

$$[m,n]_{i+1} = \lambda^{\frac{1-f^{n-1}(m)}{1-f(m)}} m^{f^{n-1}(m)}$$

$$[a,b]_i = \lambda a + (1-\lambda)b$$

$$[m,n]_{i+1} = m$$

Thanks:
Xiao En-li supported and directed my work.
Zhao Dong-hua pointed out that the order of an binary operation is sometimes not unique.

References:

B.L. van der Waerden *Algebra*