add-on collection of *oriented* items

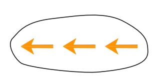
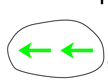
Chapter 8

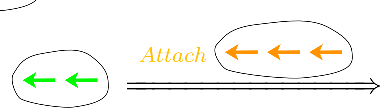
Signed Addition

8.1 Attaching Collections of Oriented Items

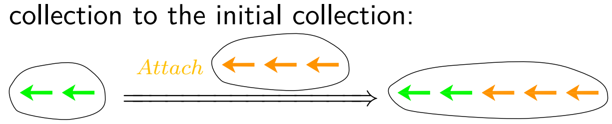
Attaching an **add-on collection of *oriented* items** to an initial collection of *oriented* items is almost as simple as attaching an add-on collection of *plain* items to an initial collection of *plain* items. The only difference is that we must check the orientation of the items in the add-on collection against the orientation of the items in the initial collection.

- When the orientation of the items in the add-on collection is *the same as* the orientation of the items in the initial collection, the attachment process is exactly the same as with collections of plain items.

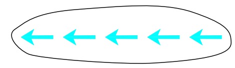
EXAMPLE 8.1. To perform the *action* of attaching  to the *initial collection* , that is:

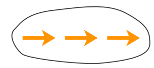
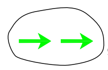


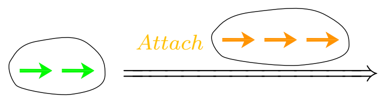
we attach the add-on collection to the initial collection:



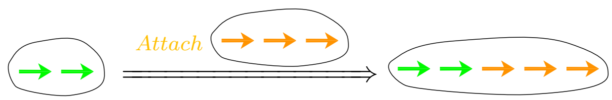
and, since the orientation of the initial items is *the same as* the orientation of the attached items, there is no cancellation and the *final collection* is

.

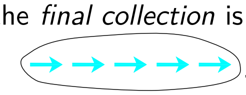
EXAMPLE 8.2. To perform the *action* of attaching  to the *initial collection* , that is:



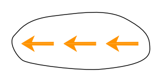
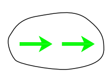
we attach the add-on collection to the initial collection:



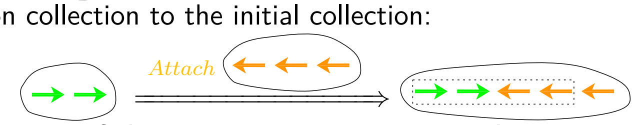
and, since the orientation of the initial items is *the same as* the orientation of the attached items, there is no cancellation and the *final collection* is

.

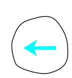
- When the orientation of the items in the add-on collection is *the opposite of* the orientation of the items in the initial collection, there is going to be *cancellation* of the items of *opposite* orientation.

EXAMPLE 8.3. To perform the *action* of attaching  to the *initial collection* , that is:

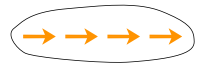
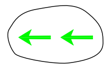
we attach the add-on collection to the initial collection:

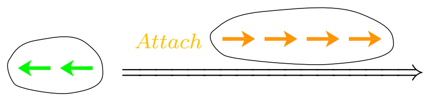


but, since the orientation of the initial items collection is *the opposite of* the orientation of the attached items, there are two cancellations and the *final collection* is

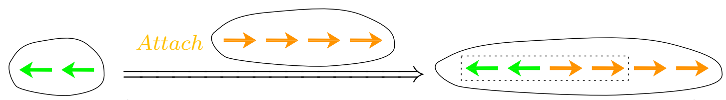
.

function
input-output rule
unspecified input
specific inputs
function name
output specifying code
specific outputs

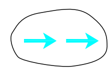
Example 8.4. To perform the *action* of attaching  to the *initial collection* , that is:



we attach the add-on collection to the initial collection:



but, since the orientation of the initial items collection is *the opposite of* the orientation of the attached items, there are two cancellations and the *final collection* is

.

8.2 Adding Signed Number Phrases

1. We saw in Chapter 5 that real world *agents of change* are represented on paper by **functions** which we specify with an **input-output rule** that consists of:
i. An **unspecified input** eventually to be replaced by **specific inputs**, that is the number phrases that represent the *initial collections*.
ii. A **function name**, that is the name of the function that represents the agent of change
iii. The **output specifying code** which is the code that specifies the *output* of the function in terms of the input. The **specific outputs** are the number phrases that represent the final collections.
Thus, the real world *action*

$$\text{Initial collection} \xRightarrow{\textit{Agent of Change}} \text{Final collection}$$

is represented on paper by the *input-output rule*

$$\underbrace{\text{Unspecified input}}_{\text{To be replaced by specific inputs}} \xrightarrow{\text{Name of function}} \underbrace{\text{Output-specifying code}}_{\text{Specifies what to do with the inputs to get the outputs}}$$

2. It will be most important to know what we are talking about:

plain adding
$\oplus$
oplussing
signed adding function
signed addition

LANGUAGE 8.1. (Plain versus Signed Adding) To make it clear which *adding* we are talking about, we will refer to the adding of plain numerators which were introduced in Chapter 5 as **plain adding**.
In fact, to keep things clear, we will use for signed addition the symbol $\oplus$, read "oplus", and, in the future, instead of saying that we are "*adding* a signed numerator", we will say that we are "**oplussing** that numerator".

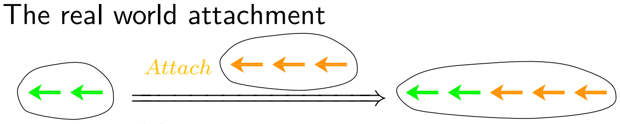
3. Just as with plain addition, we represent the action of attaching an add-on collection of *oriented* items to a collection of *oriented* items by the input-output rule of a **signed adding function**, often called **signed addition** for short.

$$\text{Input signed numerator} \xrightarrow{\oplus\ \text{signed numerator}} \text{Output numerator} = \text{Input signed numerator} \oplus \textit{signed}\ \text{numerator}$$

4. Just like, in the real world , we must check if the orientation of the items in the add-on collection is the same as or the opposite of the orientation of the items in the initial collection, in the paper world we must check if the sign of the add-on numerator is the same as or the opposite of the sign of the input numerator.

- When the sign of the add-on numerator is the same as the sign of the input numerator, then:
  - The *sign* of the output numerator is the sign common to the input numerator and the add-on numerator
  - The *size* of the output numerator is the plain addition of the size of the add-on numerator to the size of the input numerator

EXAMPLE 8.5. The real world attachment

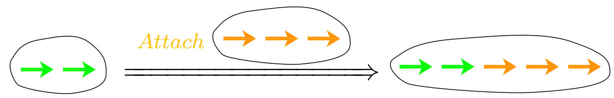


is represented in the paper world by

$$\begin{aligned} -2\,\textsf{Arrows} \xrightarrow{\oplus -3\,\textsf{Arrows}} & -2\,\textsf{Arrows} \oplus -3\,\textsf{Arrows} \\ &= -(2+3)\,\textsf{Arrows} \\ &= -5\,\textsf{Arrows} \end{aligned}$$

where $2+3=5$ is the *plain* addition of the sizes.

EXAMPLE 8.6. The real world attachment



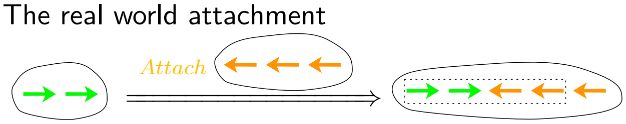
is represented in the paper world by

$$\begin{aligned} +2\,\textsf{Arrows} \xrightarrow{\oplus +3\,\textsf{Arrows}} & +2\,\textsf{Arrows} \oplus +3\,\textsf{Arrows} \\ &= +(2+3)\,\textsf{Arrows} \\ &= +5\,\textsf{Arrows} \end{aligned}$$

where $2 + 3 = 5$ is the *plain* addition of the sizes.

- When the sign of the add-on numerator is the *opposite of* the sign of the input numerator, we must *compare* the *size* of the add-on numerator to the *size* of the input numerator in order to know which way to *plain subtract* the sizes:
  - The *sign* of the output numerator will be the sign of the numerator with the *larger size*,
  - The *size* of the output numerator will be the result of *plain subtracting* the numerator with *smaller size* from the numerator with *larger size*.

EXAMPLE 8.7. The real world attachment

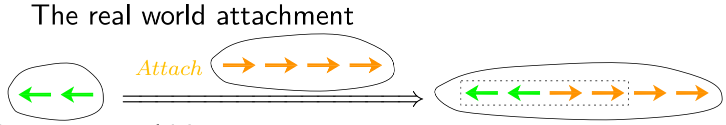


is represented in the paper world by

$$\begin{aligned} +2\,\textsf{Arrows} \xrightarrow{\oplus -3\,\textsf{Arrows}} & +2\,\textsf{Arrows} \oplus -3\,\textsf{Arrows} \\ &= -(3-2)\,\textsf{Arrows} \\ &= -1\,\textsf{Arrows} \end{aligned}$$

where $3 - 2 = 1$ is the *plain* subtraction of the smaller size from the larger size..

EXAMPLE 8.8. The real world attachment



is represented in the paper world by

prior knowledge

$$-2\,\mathsf{Arrows} \xrightarrow{\oplus +4\,\mathsf{Arrows}} -2\,\mathsf{Arrows} \oplus +4\,\mathsf{Arrows}$$
$$= +(4-2)\,\mathsf{Arrows}$$
$$= +2\,\mathsf{Arrows}$$

where $4-2=1$ is the *plain* subtraction of the smaller size from the larger size..

5. Here are a few examples that focus on the *numerators*.

Example 8.9. $-3 \oplus -5 = -(3+5) = -8$
where, since the two numerators have the *same* signs:
- the *sign* of the result, $-$, is the common sign,
- the *size* of the result, 8, is the the *plain addition*, $3+5$, of the sizes

Example 8.10. $+3 \oplus +5 = +(3+5) = +8$
where, since the two numerators have the *same* signs:
- the *sign* of the result, $+$, is the common sign,
- the *size* of the result, 8, is the the *plain addition*, $3+5$, of the sizes

Example 8.11. $-3 \oplus +5 = +(5-3) = +2$
where, since the two numerators have *opposite* signs:
- the *sign* of the result, $+$, is the sign of the numerator with the *larger* size,
- the *size* of the result, 2, is the *plain* subtraction, $5-3$, of the smaller size from the larger size.

Example 8.12. $+3 \oplus -5 = -(5-3) = -2$
where, since the two numerators have *opposite* signs:
- the *sign* of the result, $-$, is the sign of the numerator with the *larger* size,
- the *size* of the result, 2, is the *plain* subtraction, $5-3$, of the smaller size from the larger size.

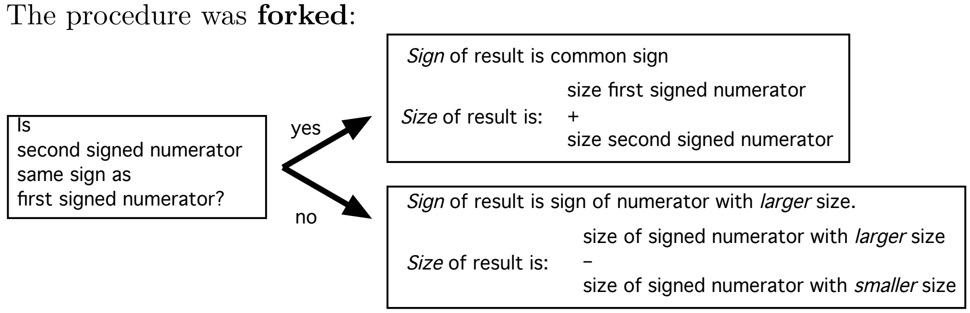
8.3 Architecture

Dealing with *signed* numerators was based on a **prior knowledge** of *plain* numerators. (To help us focus, we will deal here with *counting* numerators but things are exactly the same with *decimal* numerators.)

1. In order to represent collections of *plain items*, we used: tool
i. *Plain numerators*, $1, 2, 3, \ldots$ forked
ii. In order to *compare* a first plain numerator to a second plain numerator, the procedure was to count from the first plain numerator to the second plain numerator and if we had to *count up* then the verb was $<$ and if we had to *count down* the verb was $>$.
iii. In order to *add* to a first plain numerator a second plain numerator, the symbol was $+$ and the procedure was, starting from the first plain numerator, to count *up* on our fingers the second plain numerator. Then the plain numerator we arrive at was the result.
iv. In order to *subtract* from a first plain numerator a second plain numerator, the symbol was $-$ and the procedure was, starting from the first plain numerator, to count *down* on our fingers the second plain numerator. Then, *if we could do it*, the plain numerator we arrived at was the result.
So, our **tool** was *counting*.

2. In order to represent collection of *oriented items*, we used:
i. *Signed numerators* whose *sign* was one of two symbols, $+$ or $-$ (unfortunately already used above) and whose *size* was a plain numerator.
ii. In order to *compare* a first signed numerator to a second signed numerator, the procedure was still to count from the first signed numerator to the second signed numerator and if we had to *count up* then the verb was $<$ and if we had to *count down* the verb was $>$. Note, though, that this was *signed counting* since we also had to be able to count on *negative* numerators.
iii. In order to *oplus* to a first signed numerator a second signed numerator, the procedure was quite a bit more complicated because:

- The procedure was **forked**:



but also because

- We had to compare, add, subtract *plain numerators*, namely the *sizes* of the signed numerators.

Altogether then, *oplussing signed numerator* is *based* on prior knowledge of *plain numerators*.

Index