The Animated Abacus

Pieter Jordaan

[1] LIACS, Leiden University, Niels Bohrweg 1, Leiden, The Netherlands
pieter_jordaan@hotmail.com

supervised by Fons Verbeek.
fverbeek@liacs.nl

Abstract. There are many programs to learn kids the basics of numbers. However all these programs do this by examination and no program explains what addition, subtraction, division and multiplication mean. Kids are too young to understand abstract terms. We found a way to learn kids what addition, subtraction, division and multiplication that is more persuasive than the traditional black board.

1 Introduction

Programs that try to learn children mathematic operations are very common, but the problem is that all these programs are focused on examination and not on learning the abstract operators. Many children that are about 5 years old have trouble understanding abstract terms.

2 Problem definition

The mathematical operators are very hard to explain to a kid, and so far the best way on primary schools is to teach it in class verbally. Some kids experience this as very boring, making it hard to teach them maths. This is mainly the case because most operators are learned by memorization and not by explaining why. For example, for a 3rd grade kid 5x10=50, just because that is said to be the answer. No logics are explained making it harder for kids to understand any advanced mathematics.
The computer is only used to examine if the kid knows the answers. The computer asks a question (like 1+1) and the child provides an answer. There is no program that tries to explain why 1+1 is equal to 2. It only says if you had the correct answer or not. That is why we think our program could explain the basic operators in a child-friendly program and examine at the same time. It is especially the best solution for children who find listening to the teacher very boring. For those children a computer can be used to aid them. A computer program that teaches children mathematics should be as persuasive as possible. Children often like cheery, colorful objects, which is indeed

used in many computer programs for children. However without the teacher telling you what plus, minus, division and subtraction is, the kid could never use the programs that were originally made for teaching math. The combination is what makes this program different from all the programs that already exist.

3 Problem Solving

For every operator a different approach is taken to show how the basic mathematic operators work. As said in the introduction we want to explain the basic mathematic operators to the child and give that more priority than examination.

Addition

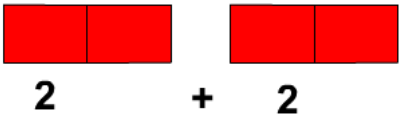

Figure 1: An example of addition as explained by the program.

Addition uses the simple fact that you can count both together to get the answer. For example if we want to explain 2+2, we do this by making two pairs of any type of object (like a car, a banana or a monkey). The child can count both pairs together and find out that if he/she counts them together the answer might be four. After the child gives the answer, the computers shows the answer by counting all objects one by one and tells you if what you answered was correct or not with a funny animation.

Subtraction

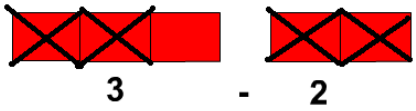

Figure 2: An example of a subtraction as explained by the program.

Subtraction is explained by saying that for some object (like a monkey) every object is picking one other object (like a banana) and counting the objects that were left over. There are for example 3 monkeys and 2 bananas. Every monkey picks one banana and walks away. Once the bananas are gone, only one monkey is left over giving you the answer, which is 1. To make it easier for children to solve the answer they can drag the objects with the mouse and move every monkey to a banana and find out themselves that one monkey can't have a banana.

Multiplication

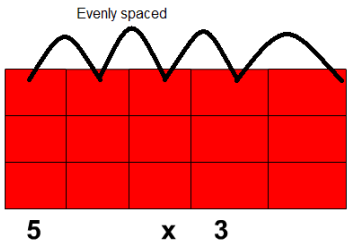

Figure 3: An example of multiplication.

Multiplication is the hardest to explain to a child. That's why teachers teach multiplication by memorization and do not explain very thoroughly why. We chose to explain multiply by making rows and columns of characters that are evenly spaced. In the example we explain 5x3 by making 5 columns of 3 (or 3 rows of 5). Counting one by one is slow and boring plus a child might not see that the characters form rows and columns. So we decided that the computer counts them column by column so the computer says: '3, 6, 9, 12, 15' which is the answer. Multiplication is learned by children once they understand addition so it can be assumed children understand that we find the answer by saying 3+3+3+3+3=15, so 5x3=15.

Division

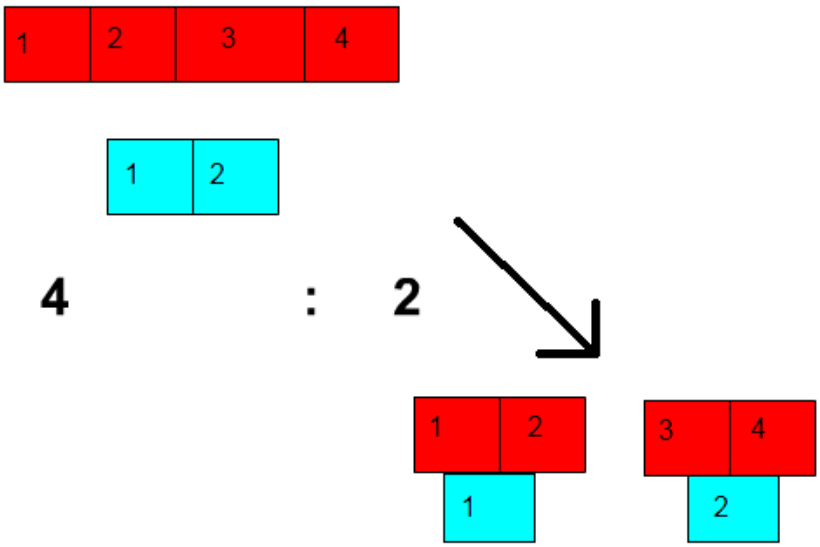

Figure 4: An explanation how division is explained for children.

Division is learned with the explanation of sharing. For example there could be 4 bananas and 2 monkeys. All the monkeys should eat the same amount of bananas. The computer shows that 4 divided by 2 is equal to 2, because the 2 monkeys will both eat one banana at the same time. After every monkey eats one banana, they go to the next banana and both monkeys eat another banana. After that, no bananas are left over so the answer is 2. Figure 4 shows how it is explained. To help the child find the answer he/she can drag the objects so he/she can drag them in multiple groups all with the same amount of characters to find the answer.

To work on the interface, we have to look at what the user groups want. There are 2 user groups: the children and the teachers. They both have their own demands.

Children	**Teacher**
• Children want a simple, colorful interface. • Since children can't read very well on that age the program needs to have as little text as possible. • Children seem to have trouble with the interface if the interface differs too much compared to programs they already use.	• Teachers want to see how well the children did their tests. • Teachers want to control what children can do and what they can 't do with the program(supervision) • Children need to get the questions they have more trouble with more often.

For the children user group we made the interface with cheery colors and also by adding big buttons for the operators. The first screen they see is a screen where the child can select her name. Since this involves text we added a photo of the pupil and the button for going to the next step is just an arrow. Then we'll see the menu as shown on figure 5. On the button is the operator symbol. Since we want to make clear what this button does and a symbol is abstract, a mouse over event will start an animation in the button that shows the explanation of the operator in detail just like the way our program explains a sum.

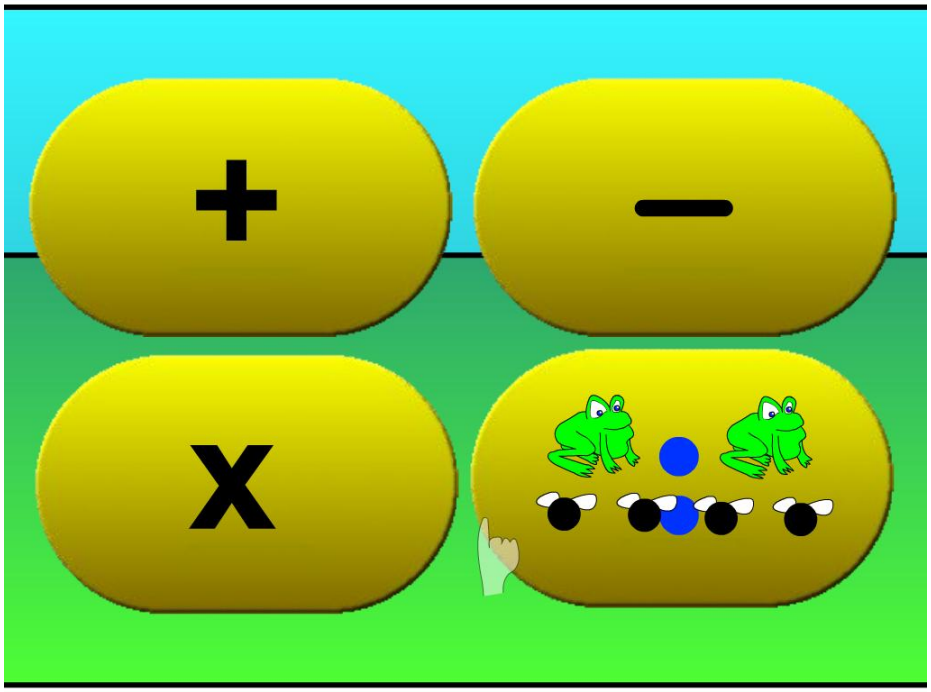

Figure 5: The interface the child will see if the teacher enabled all operators for the child. As you can see that when the mouse cursor is hovered over division that it will show division in action.

Once the child selects the operator he or she wants she will see the screen like on figure 6. Any button that will not do anything will disappear or will be grayed out. This is the same in existing programs that children will use. If the child presses the ok button the program shows not only what the answer is but will also give an explanation why. If the child answered it correctly the child will be able to make the next question. If the child makes a mistake the child can redo the question (unless the teacher disabled this option) or go to the next question. After answering all questions the child will see a congratulations screen once they're done.

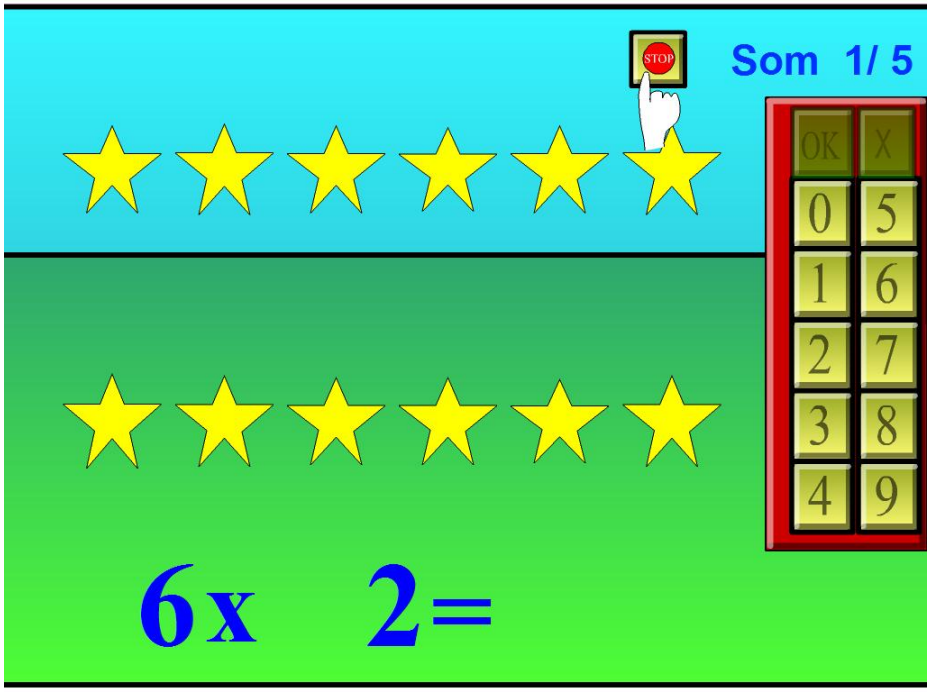

Figure 6: The interface asking the question 6 times 2 with the sum mentioned on the bottom, the stop button at the top and the number buttons on the right.

For the teacher different aspects are needed and the most important one is that the teacher needs to know how the children did the tests. This will need a data structure which saves all information about the child and also the options the child can use.

This will give us the next data structure:

- **Pupil name:** String with maximum of 80 characters
- **Number of correct answers:** Unsigned integer
- **Number of wrong answers:** Unsigned integer
- **What sums the pupil can make:** 4 booleans
- **A distribution of all the addition sums:** an array of unsigned integers
- **A distribution of all the subtraction sums:** an array of unsigned integers
- **A distribution of all the division sums:** an array of unsigned integers
- **A distribution of all the multiplication sums:** an array of unsigned integers

I will cover the distribution of the sums later. The data structures of the children are saved in Flash local shared objects. The first pupil's local shared object name is 'data0' and the second pupil's local shared object name is 'data1', etcetera. The picture of the pupil covers this too by naming them like 0.jpg and 1.jpg.

The distribution of all the sums is used for giving the teacher 2 extra possibilities. For every possible sum (1+1, 1+2, 1+3… till 50:10, 60:10) there is a number in this array. The chance for a the sum i in the array to appear as the current sum can be calculated with:

$$P(i) = \frac{V_i}{\Sigma V_n}$$

In this formula V_i is the value of the array on position i. and is ΣV_n the sum of all values in the array of the used operator. When a child starts this program for the first time, this array is filled with the value 10 meaning every sum has the same chance to appear.

If a child answers a question the values change in the array.

If a child answered sum i correctly V_i will decrease by 1(when V_i>1) and all the other V_n values will increase their value by 1.

If a child answered sum i incorrectly V_i will increase it's value and the inverse sum (so for 1+2 this is 2+1) increase its value too.

This system has some characteristics. For example if a child is very well with answering the questions the values will increase which means that answering one question wrong will have less impact on the distribution then if a child does very bad answering the questions. The reason is that it's very likely that the child made a typing error if it is used to answer everything correctly. Another characteristic is that if a child is very bad answering a question correct will have a high impact on the appearance of that question again. This is what teachers want of the program.

Another good thing about this approach is that it's easy to disable a sum without needing a bigger data structure, since we can disable a question by putting its value on 0. The downside is the fact that the values will keep increasing over time making the impact on the distribution smaller and smaller. Also if a child keeps answering the same question wrong all over and over again that eventually it will be the only question that will be asked. Even though this applies in theory, in practice this never happens as you need to do this intentional to get to that point.

4 Results and Evaluation

Evaluating it on real kids seems to show that this program is more interesting than the traditional courses they have now. With just a slight push in the right direction some children from class 3 (and early class 4) who haven't learned division could answer the questions. These haven't learned division yet, but can solve them by just dragging the characters. However for some children the learning doesn't work on the long run as it becomes just an 'algorithm' to get an answer. For example for addition it will become just a counting game instead.

5 Conclusion and Discussion

The evaluation shows that the program does what it was supposed to do. Even older children seem to like the program, which is mainly because of the cheery animations. Still this program can not replace a human teacher as using just this program can still

have a negative impact on it since how silly as it seems, memorization is still needed if children want to calculate faster, because extensive use of the program will change it into a routine and will remove the teaching aspect for some of the children. Also on some schools the program ran too slow since Flash is not very fast. The reason Flash was used was that it does not need any new components that needs an install on a school system.

6 Future Work

An improvement might be needed for addition and multiplication so it becomes more than a counting assignment. Also a conversion from Flash to C++ can be used for gaining speed, even though the development time of a C++ version will take some time.

References

1. Human-Computer Interaction, third edition, Dix Finlay, Abowd Beale
2. Teaching with games, Richard Sandford, Mary Ulicsak, Keri Facer, Tim Rudd
3. Affordances and Design, D. Norman
4. Social Contect in HCI: A New Framework for Mental Models, Cooperation, and Communication, University of Padova

Appendix

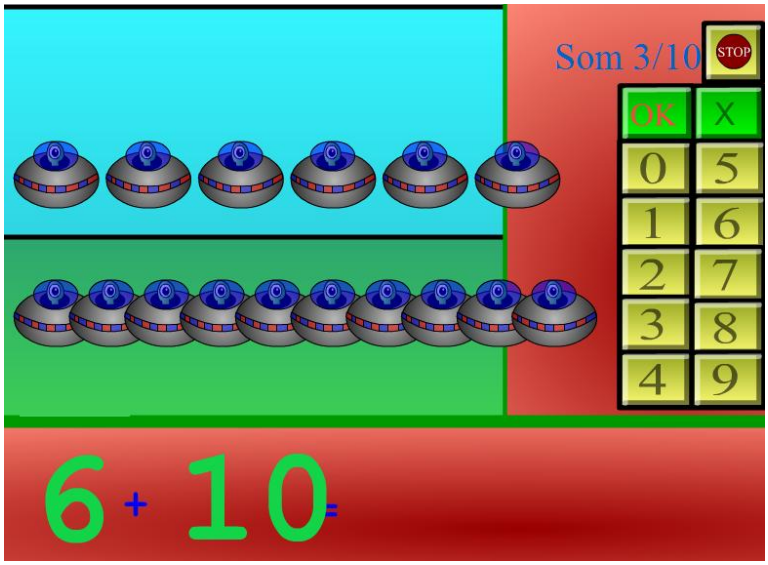

Figure 7: The final prototype when it was used for HCI course, which has changed now.

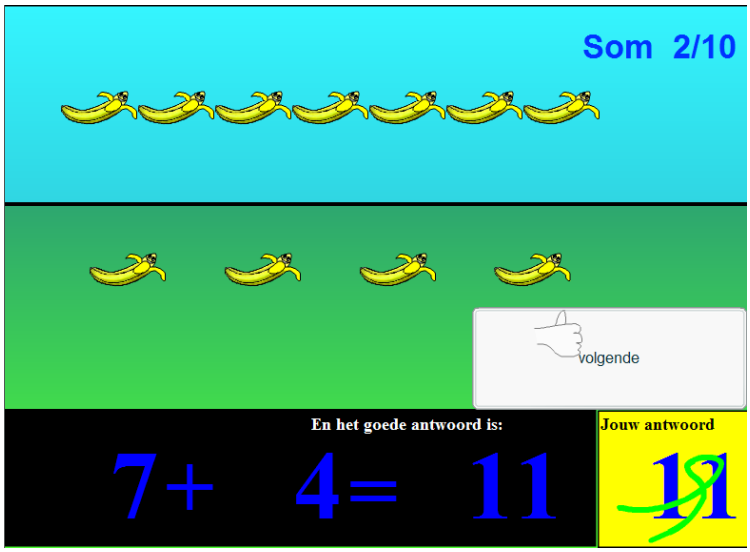

Figure 8: The new version: less crowded and everything perfectly aligned.