Activity 8

Beat the clock—*Sorting networks*

Age group	Middle elementary and up.
Abilities assumed	Identifying the larger and smaller of two numbers. Activity 7 on sorting algorithms is helpful, but not essential, preparation.
Time	10 to 30 minutes.
Size of group	At least six children.

Focus

Comparing

Ordering

Developing algorithms

Cooperative problem solving

Summary

Even though computers are fast, there is a limit to how quickly they can solve problems. One way to speed things up is to break a job up into pieces and have a different computer process each piece simultaneously, a strategy that is called "parallel computing." This activity shows how sorting items into numerical order, normally thought of as a sequential kind of process, can be carried out more quickly using parallelism.

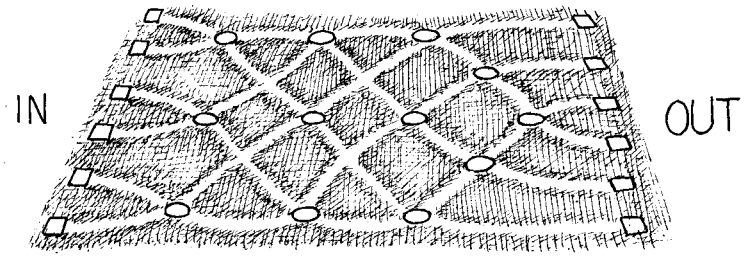


Figure 8.1: A sorting network

Technical terms

Parallel computation; concurrency; sorting.

Materials

Chalk or other material for marking on the ground,

two sets of six cards, such as the ones in the blackline master on page 89, and

a stopwatch.

What to do

1. Before working with the children, mark out the network shown in Figure 8.1 on the ground. It needs to be large enough for children to run around it following the lines, and the circles and squares must be obvious. There are several ways of marking the network. Possibilities include

   - drawing with chalk on an asphalt playground,
   - construction site ribbon “stapled” to the grass with small pieces of masking tape,
   - masking tape on the floor,
   - string or streamers attached to the ground with tape, and
   - mowing paths through long grass.

2. If the children have not already done the activity on sorting (Activity 7), discuss the frequent need for computers to sort lists of items into order (Activity 7 gives information about this).

3. Show the children the sorting network (Figure 8.1) drawn on the ground, and explain how they will use it, as follows.

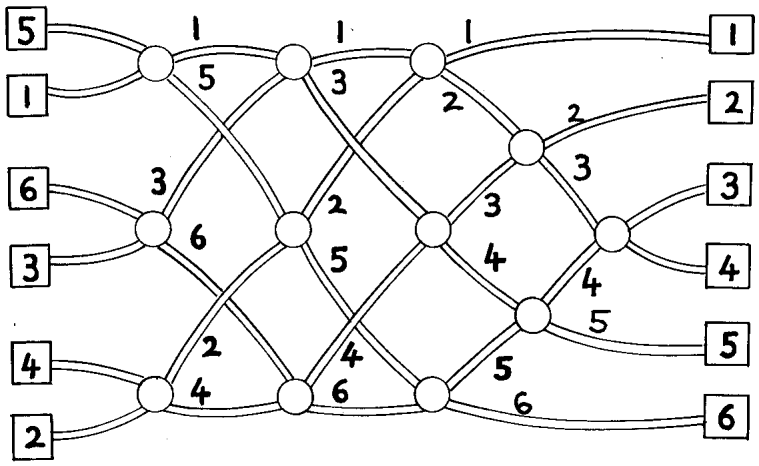


Figure 8.2: An example flow of numbers through a sorting network

Divide the children up into teams of six. One team uses the network at a time. Give each member of the team a card with a number from 1 to 6 on it (see the blackline master on page 89). The members of a team start at the six squares on the left-hand side of the network, in a random order. Children advance along the lines marked, and when they reach a circle they wait for someone else to arrive. When there are two children in a circle they compare cards. The one with the smaller number takes the exit to their left, and the other to their right (it can be helpful if the line to the right is thicker to remind them that the larger number goes that way). When they arrive at the six squares on the right they should have ended up in ascending numerical order of cards. Figure 8.2 shows an example of the numbers flowing through the network of Figure 8.1, with the sorted numbers coming out on the right.

4. Once the children have understood the goal, and have had a trial run, use the stopwatch to time how long each team takes to get through the network. Now use cards with larger numbers (such as the three-digit ones in the blackline master on page 89). For older children, make up cards with even larger numbers that will take some effort to compare—or with words, to be compared alphabetically. The idea is to see which team can get through the network quickest. In the rush it is possible that the numbers will end up unsorted, or that a lone child will be left standing in the middle of the network. In both cases the team has made an error, and must start again.

Variations and extensions

Once the children have understood the operation of a node (circle) in the network, where the smaller value goes left and the other goes right, all sorts of variations can be explored using this as a building block. One simple question that they should be able to answer is “what happens

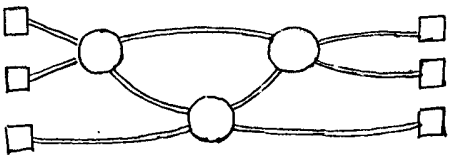


Figure 8.3: A smaller sorting network

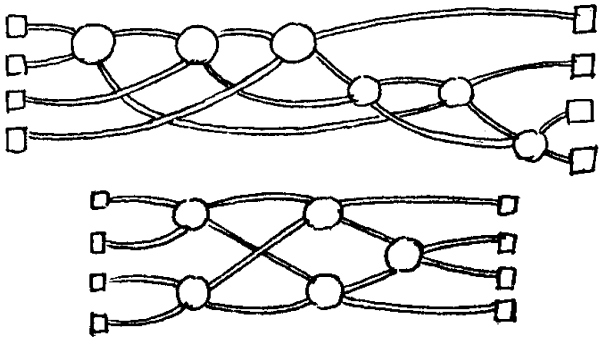


Figure 8.4: Two different networks for sorting four input values

if the smaller one goes right instead of left and vice versa?" (The numbers will be sorted in reverse order.) Another question for them to try to answer is "does it work if the network is used backwards?" (It will not necessarily work, and the children should be able to find an example of an input that comes out in the wrong order.)

The children can try to design smaller and larger networks. For example, Figure 8.3 shows a network that sorts just three numbers, which children should try to come up with on their own.

The network for sorting a particular number of inputs is not unique. For example, Figure 8.4 shows two different networks that will sort four inputs. Which is the faster? (The second one is. Whereas the first requires all comparisons to be done serially, one after the other, the second has some being performed at the same time. The first network is an example of serial processing, whereas the second uses parallel processing to run faster.)

Figure 8.5 shows a larger sorting network which can be used if a group is keen to tackle something a bit more challenging.

The networks can also be used to find the minimum or maximum value of the inputs. For example, Figure 8.6 shows a network with eight inputs, and the single output will contain the minimum of the inputs (the other values will be left at the dead ends in the network).

Discuss processes from everyday experience that can (and can't) be accelerated using parallelism. For example, cooking a meal would be a lot slower using only one cooking element, because the items would have to be cooked one after the other. Likewise, when building a house, it is possible to have several people work on different tasks at the same time. What kinds of jobs can be completed more quickly by employing more people? What kinds of jobs can't?

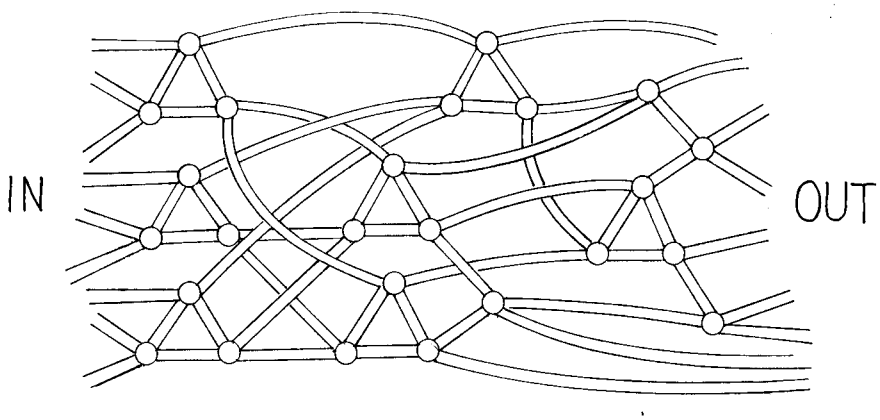


Figure 8.5: A larger sorting network

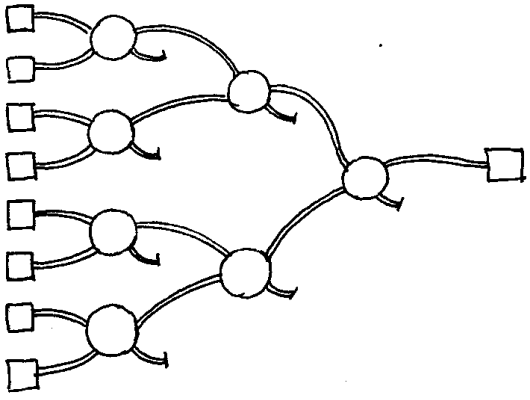


Figure 8.6: A network that finds the smallest of eight numbers

What's it all about?

The power of computers is increasing at a mind-boggling rate. Their speed doubles approximately every two years, and their memory capacity doubles even more often, yet their price remains roughly constant. With this exponential rate of growth of readily available computing power, it is hard to understand why people persistently complain that their computer is too slow. Yet they do, and those who can afford to are quick to upgrade when faster models come out. As we see what computers are capable of and find more uses to put them to, our expectations rise dramatically. Today's lowly word processor would have been unimaginable only 25 years ago, and a kid's drawing program would have been a research project back then.

Some people need faster computers so badly that they can't even wait a couple of years until computers are faster. Motivated by these needs, computer scientists explore ways to make computations happen more quickly without having to get faster computers. One way of doing this is to write programs that do their task using fewer computational steps. This approach is explored in the searching and sorting activities (Activities 6 and 7).

Another approach to solving problems faster is to have several computers work on different parts of the task at the same time. This can only be done if the task can be broken up into independent parts. This is not always possible, but computer scientists have found that a lot of everyday tasks can be parallelised, including searching a database and sorting data into order. This activity shows how the problem of sorting a list into order can be decomposed into a number of comparison operations, many of which can be carried out at the same time.

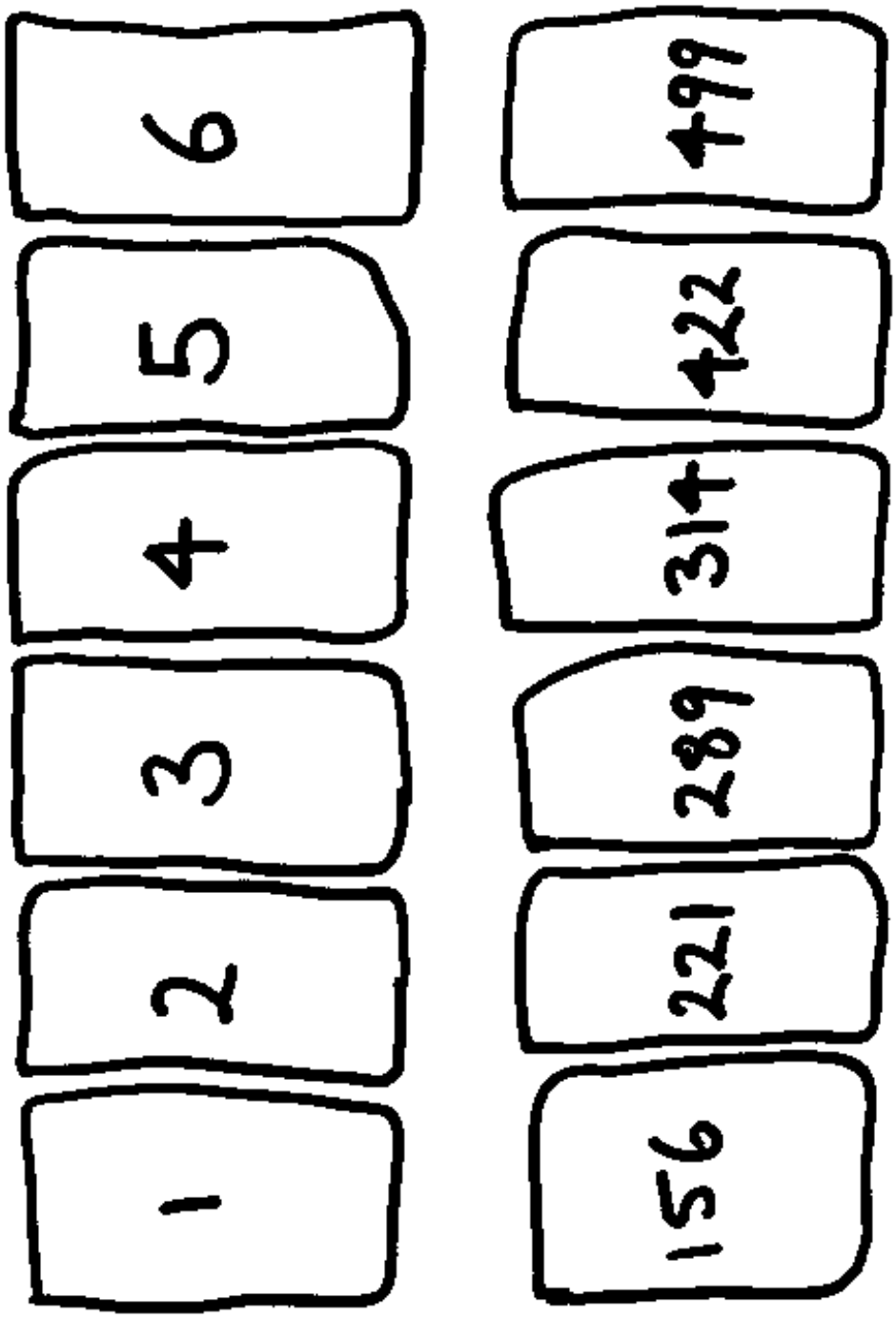
For example, in Figure 8.2, up to three comparisons are made in parallel. Even though a total of twelve comparisons will be made in the process of sorting the numbers, many of them are performed simultaneously, and the overall time will be that required for just five comparison steps. This parallel network sorts the list more than twice as quickly as a system that can only perform one comparison at a time.

Isn't parallel processing expensive? Well, yes, it is a bit. But most of the money you spend on a computer goes into the screen, the keyboard, the box, the power supply, the disk unit, ... even the desk you put it on. The processor chip is really quite cheap, and having a dozen (or even a hundred) of them inside the box instead of just one isn't going to affect the price of the computer as much as you might think.

Not all tasks can be completed more quickly by using parallel computation. As an analogy, imagine one person digging a ditch ten feet long. The task could be completed almost ten times as quickly if ten people each dug one foot of the ditch. However, the same strategy could not be applied to a ditch ten feet deep—the second foot is not accessible until the first foot has been dug. And although one woman can bear a child in nine months, it is not possible to speed up the process and have nine women work together to bear a child in one month!

Further reading

Chapter 10 of *Algorithmics* by David Harel discusses parallelism and concurrency, including sorting networks, and the analogies with ditch-digging and pregnancy.



Instructions: *These are two sets of six cards for use with the sorting networks activity.*