Activity 9

The muddy city—*Minimal spanning trees*

Age group	Middle elementary and up.
Abilities assumed	Need to be able to count to about 50.
Time	About 20 to 30 minutes.
Size of group	From individuals to the whole classroom.

Focus

Puzzle solving.

Optimization.

Planning.

Summary

Our society is linked by many networks: telephone networks, utility supply networks, computer networks, and road networks. For a particular network there is usually some choice about where the roads, cables, or radio links can be placed. The following activity explores one way to optimize the choice of links between objects in a network.

Technical terms

Minimal spanning trees; greedy algorithms; graph algorithms.

Materials

Each child will need:

a copy of the blackline master on page 96 (it is easier for the children to use if it is enlarged onto double-sized paper), and

counters or squares of cardboard (approximately 40 per child).

What to do

Once upon a time there was a city that had no roads. Getting around the city was particularly difficult after rainstorms because the ground became very muddy—cars got stuck in the mud and people got their boots dirty. The mayor of the city decided that some of the streets must be paved, but didn't want to spend more money than necessary because the city also wanted to build a swimming pool. The mayor therefore specified two conditions: (1) Enough streets must be paved so that it was possible for everyone to travel from their house to anyone else's house by a route consisting only of paved roads, possibly via other houses, and (2) the paving should be accomplished at a minimum total cost.

The map on page 96 shows the layout of the city. The number of paving stones between each house represents the cost of paving that route. The problem comes down to figuring out the least number of paving stones needed to allow people to get from any house to any other.

1. Hand out a copy of the blackline master to each child and explain the muddy city problem using the story above (you will need to adapt the story to suit the age of the children).

2. Give out the counters and let the children try to find efficient solutions by putting counters where they think paving stones should be laid. Keep the class informed as children come up with better and better solutions. Figure 9.1 shows two optimal solutions, both with a cost of just 23 paving stones. As this example illustrates, there can be more than one solution to this kind of problem, each with the same total cost.

3. Discuss the strategies that the children used to solve the problem.

   Some of the children will have developed the strategy of starting with an empty map, and gradually adding counters as necessary until all of the houses are linked. This is a good strategy; in fact, you are guaranteed to find the optimal solution if you add the paths in increasing order of length, but don't add any paths that link houses that are already linked. Different solutions are generated by changing the order in which paths of the same length are added.

   Another strategy that the children might develop is to start with all of the paths paved, and then remove redundant paths. This can lead to an optimal solution, but requires considerably more effort.

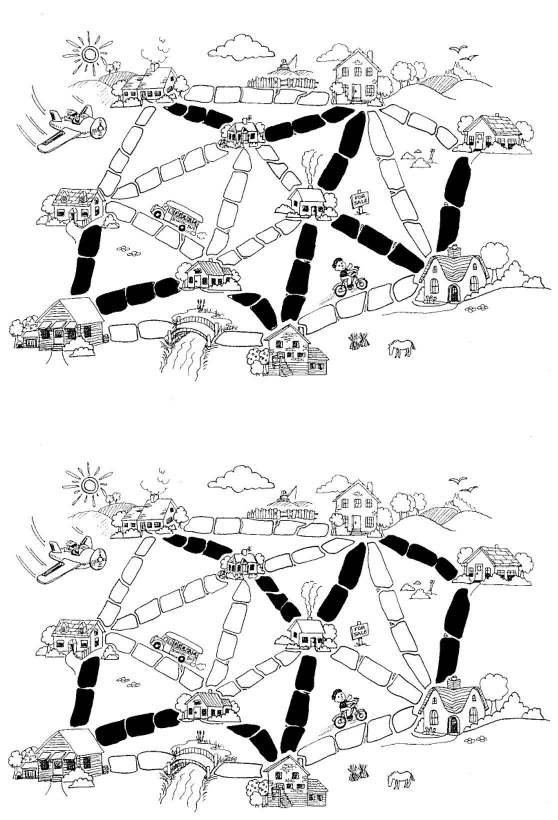


Figure 9.1: Two solutions to the muddy city problem on page 96

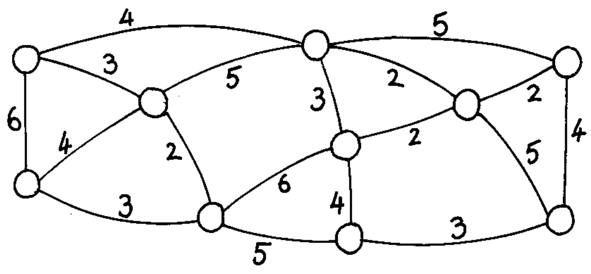


Figure 9.2: An abstract representation of a (different) muddy city

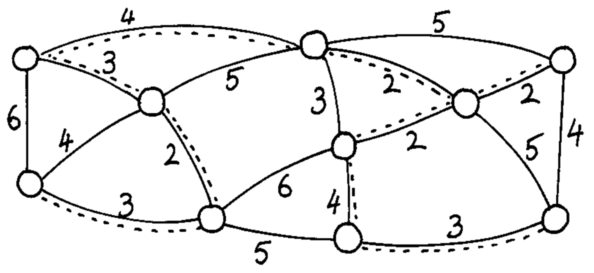


Figure 9.3: A solution to the muddy city problem in Figure 9.2

Variations and extensions

The children could experiment with other cities. Figure 9.2 shows a more abstract representation of a city that is quicker to draw and easier to work with. The houses are represented by circles, the muddy roads by lines, and the length of a road is given by the number beside the line. Computer scientists and mathematicians often use this kind of abstract representation of relationships for such problems. They call it a *graph*. This may be confusing at first because the term "graph" is used in mathematics to mean a chart displaying numerical data, such as a bar graph, but the graphs that computer scientists use are not related to these. Notice that the lengths in Figure 9.2 are not to scale. The dotted lines in Figure 9.3 show a solution.

You could have the children think about how many roads or connections are needed if there are n houses in the city. It turns out that an optimal solution will always have exactly n 1 connections in it, as this is always sufficient to link up the n houses, and adding one more would create unnecessary alternative routes between houses.

Another extension is to have the children look for real networks that can be represented by a graph, such as the network of roads between local cities, or airplane flights around the country. The muddy city algorithm may not be much use for these networks, because it simply minimizes the total length of the roads or flight paths. It guarantees that you can get between any two points, but does not take into account the convenience of the route. However, there are many other algorithms that can be applied to graphs, such as finding the shortest distance between two points, or the shortest route that visits all the points. Representing the network abstractly as a graph is a useful first step towards solving these problems.

What's it all about?

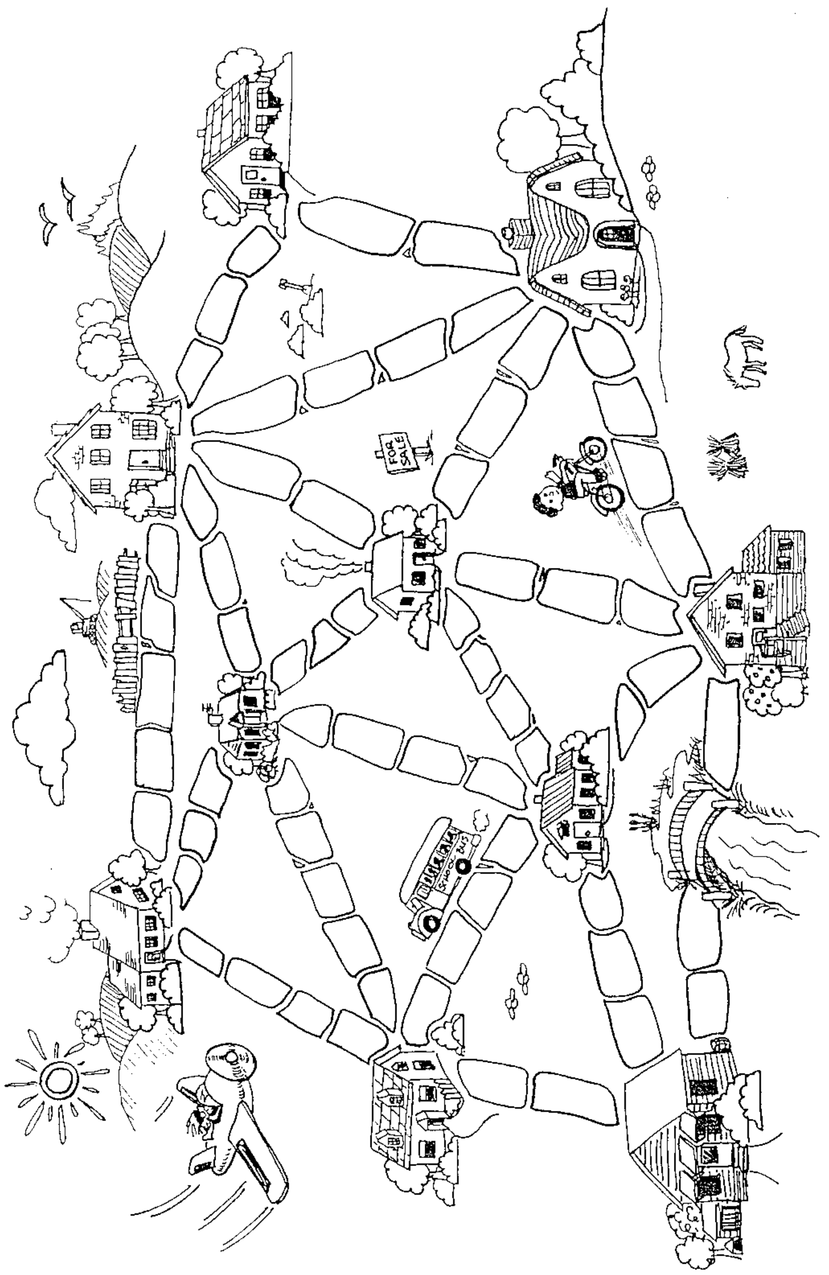
Suppose you are designing how a utility such as electricity, gas, or water should be delivered to a new community. A network of wires or pipes is needed to connect all the houses to the utility company. Every house needs to be connected into the network at some point, but the route taken by the utility to get to the house is not usually very critical, just so long as a route exists. The task of designing a network with a minimal total length is called the *minimal spanning tree* problem.

Minimal spanning trees aren't only useful in gas and power networks; they help us solve problems in applications as diverse as computer networks, telephone networks, oil pipelines, and airline routes—although, as noted above, when routing people you have to be careful to take into account the convenience of the route as well as its cost. They are also useful as one of the steps for solving other problems on graphs, such as the "traveling salesperson problem" which seeks the shortest route that visits every point in the network.

There are efficient algorithms (methods) for solving minimal spanning tree problems. A simple but method that gives an optimal solution is to start with an no connections, and add them in increasing order of size, only adding connections that join up part of the network that wasn't previously connected. This approach was mentioned earlier, and we have known children to discover this algorithm for themselves. It is called Kruskal's algorithm after J.B. Kruskal, who published it in 1956.

Further reading

Harel discusses minimal spanning trees in *Algorithmics*, and also shows how they can be used to help with the "traveling salesperson." Dewdney's *Turing Omnibus* has a section on minimal spanning trees, which discusses Kruskal's algorithm.



Instructions: *Find the minimum number of paving stones that need to be used so that you can get from any house to any other house. (The bridge doesn't need to be paved.)*