WORKSHEET FOR EXERCISES FROM CHAPTER 21

RECORD YOUR DATA FROM THE FIRST STUDY IN TABLE 21-1

Run Number	Number Of Hidden Units	Converged?	Total Sweeps
1	5	Yes	362
2	5	Yes	1236
3	5	Yes	1120
4	5	Yes	594
5	5	Yes	450
6	4	Yes	927
7	4	Yes	603
8	4	Yes	794
9	4	Yes	489
10	4	Yes	405
11	3	No	5000
12	3	Yes	713
13	3	Yes	683
14	3	Yes	976
15	3	Yes	1177
16	2	No	5000
17	2	No	5000
18	2	No	5000
19	2	No	5000
20	2	Yes	3432

Table 21-1. Record for the results of the first simulation.

EXERCISE 21.1

1. **On the basis of the results that you have recorded in Table 21-1, what would you conclude about the number of hidden value units that are required to solve this problem? What evidence would you use to defend this conclusion?**

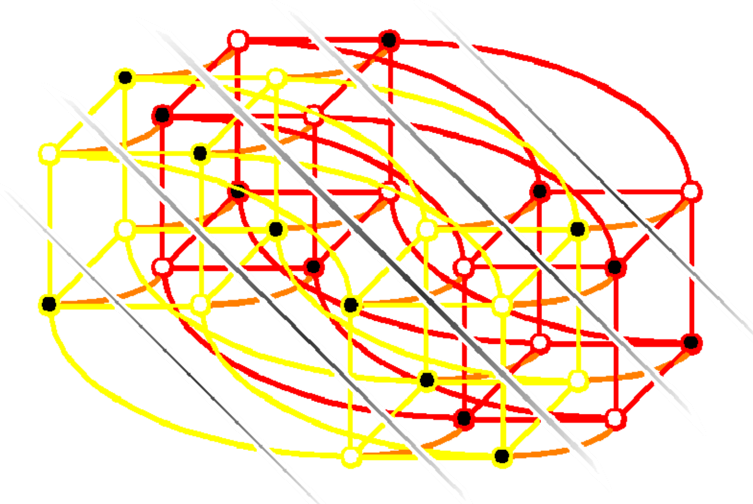
This problem can be very reliably solved by a network with 3 hidden value units. It is also possible to solve the problem with as few as 2 hidden units, because one did. However, for such a small network, this is a much more difficult problem – it might be difficult to get a decent number of successful replications with this few hidden units.

2. **Examine the "sweeps to converge" column for the simulations that converged in Table 21-1. Can you make any conclusions about what the effect of reducing the number of hidden units is on the amount of time that is required to learn the problem? If you believe that there is an effect, then why do you think this effect emerges?**

From the data above, the 5 hidden unit network appeared to be the fastest, and there was a definite slowing through the table. However, the trend is not particularly strong, as quick solutions were obtained in all conditions except for the 2 hidden unit one. Several more replications in each condition would be required to statistically verify this trend. If it exists, though, it suggests that as a network becomes more "optimal", it is more difficult for the network to solve the problem.

3. **Imagine that you wanted to determine how many hidden units are required to solve this problem without conducting an experiment like the one that you have just performed. Speculate about how you might predict the minimum number of hidden units that are required.**

For a small dimensional problem, one might draw a pattern space for the training set, and draw cuts through the pattern space that separated the on patterns from the off patterns. A pattern space for 5-parity is given below. Note from this pattern space that one could use 3 hidden units to capture the white patterns in the space, or the black patterns in the space.



CREATE YOUR VERSION OF TABLE 21-2 BY RECORDING DATA FOR THE SECOND SIMULATION

Run Number	Total Sweeps	Converged?	SSE At End	Misses At End
1	15,000	No	3.01	13

2	15,000	No	3.35	13
3	15,000	No	0.88	5
4	7175	Yes!	0.05	0
5	15,000	No	4.77	22
6	15,000	No	3.6	4
7	15,000	No	3.78	17
8	15,000	No	0.94	6
9	15,000	No	0.89	5
10	15,000	No	4.54	20

Table 21-2. Record of your data for the second exercise.

EXERCISE 21.2

1. **From your simulation records, what would you estimate the probability of having a network converge to a solution when these settings are used?**

Again, there is a very low probability of achieving a solution. From the data above, I would have to estimate a 10% chance of convergence.

2. **Can you explain why in the majority of runs the network fails to solve this problem, but in some cases a solution is found? What is the difference between runs that might affect whether the network will converge or not?**

The problem that the network has is that there are a number of different local minima that it can fall into. As can be seen from the table above, there are several different network states that are repeated when the network stalls. It would appear that the network needs a very precise set of starting weights – which randomly only occur 10% of the time – to avoid these local minima and find the global error minimum for the problem.

3. **How many hidden value units would you now say are required to solve the 5-bit parity problem?**

While I have demonstrated above that 2 units can solve the problem, this is rare. If I was running many networks on this problem, and needed a high number of convergences, I would be tempted to use a 3 hidden unit network.

4. **Assuming that you were able to train a network to solve the problem, examine its connection weights and biases, as well as the responses of the two hidden units to each pattern that was presented. Can you explain how this network is solving the parity problem? If you cannot come up with an explanation – and it might be difficult! – can you describe the source of the problem?**

The one 2-hidden unit network that I found has a very interesting structure. The connection weights that feed into the network are very regular, as shown in the table below:

Pattern	HID 1	HID 2
Bias	-0.04	1.47
IN 1	-0.52	-0.65
IN 2	0.48	-0.96
IN 3	0.52	0.65
IN 4	0.54	2.27
IN 5	-0.54	-2.29

The one 2-hidden unit network that I found has a very interesting structure. Note for hidden unit 1, all of the connection weights are nearly equal in weight, but some are negative. The bias is zero. This means that for many patterns in which an even number of bits are turned on – but not all – the unit will turn on. It will turn on when the "negative" inputs cancel the "positive" inputs. A similar organization is evident in hidden unit 2, but it is a bit more complicated, because the bias is much larger, and the pairing of weights is more intricate – some weights are repeated, but not all of the weights have the same value. Together, these units will generate either a 1 or a 0 to any even parity pattern. Interestingly, an odd parity pattern will generate a weak intermediate response in both hidden units. As a result, if you plot the patterns in a space defined by hidden unit activity, the odd parity patterns lie in the middle of two areas of even parity patterns. A single output value unit can easily make the two required cuts in this hidden unit space to correctly classify the patterns, as is shown in the figure below: