Progress Report
December 13 - December 16, 2010

Raj K. Kumar, Amir Hesam Salavati
E-mail: raj.kumar@epfl.ch, hesam.salavati@epfl.ch

Supervisor: Prof. Amin Shokrollahi
E-mail: amin.shokrollahi@epfl.ch

Algorithmics Laboratory (ALGO)
Ecole Polytechnique Federale de Lausanne (EPFL)

December 16, 2010

1 Introduction

In this report we will get back one more time to the Gold sequences! This time, we consider applying our old idea on using diagonal matrices in deriving weights with Gold sequences. We suggest a scheme based on the original Hopfield network and this modified weighting scheme which is able to to store $M = n$ patterns with a network of n nodes with the capability of correcting a number of erroneous bits (4 for the case of $n = 127$). The proposed method is iterative, meaning that we let it converge to a stable state, in contrast to our previous attempts which was mainly based on a single shot approach. This probably gives us more degrees of freedom in addition to the new weighting scheme. We have simulated the performance of the proposed method and found some good results.

In what follows, we briefly describe our approach and partially analyze it. Although it is not possible yet to derive a closed-from relationship, simulation results are quite encouraging.

2 Model

We assume that we have a Hopfield network and would like to memorize M patterns. Neural update rule is fixed as before. However, we use a modified weighting rule, as shown in equation (1).

$$w_{ij} = \frac{1}{n} \sum_{\mu=0}^{M-1} \Omega_m x_i^\mu x_j^\mu \tag{1}$$

We assume that the pattern μ corresponds to $a = \alpha^\mu$ in the generation of Gold sequences, as given by equation (2) [1]. Here, m is an *odd* integer and $d = 2^l + 1$, where l is another integer such that l and m are relatively prime. α is a primitive element of $F(2^m)$.

$$g_a^i = (-1)^{T(a\alpha^i)+T(\alpha^{id})} \tag{2}$$

In which g_a^i is the i^{th} bit of the Gold pattern g_a and $T(.)$ is the finite field trace function from $F(2^m)$ to $F(2)$. The length of each pattern is equal to $n = 2^m - 1$, i.e. i varies from 1 to n. For each $a \in F(2^m)$ we will get a binary sequence with length n.

The key idea here would be then how to choose Ω_μ such that we can memorize as many Gold sequences as possible *in presence of noise*. We show that if we select Ω_μ according to equation (**??**), then the Hopfield network can memorize upto $M = n$ sequences with the ability to correct at least $6 bits of error$.

$$\Omega_\mu = (-1)^{T(\alpha^{-\mu})} \tag{3}$$

There is one subtlety here, however, and that's the fact that we assume the input to the network is $\Omega_\mu \underline{x}^\mu$, i.e. we multiply pattern μ by its corresponding weighting term Ω_μ and then network will give us $\underline{x}^\mu$ in return. This is not an unrealistic assumption though since it is just the same as to say that we are doing *hetero-association* between $\Omega_\mu \underline{x}^\mu$ and $\underline{x}^\mu$. Then, the weighting rule (1) is exactly the same as the original Hopfield weighting rule for the hetero-association network (as illustrated in figure 1).

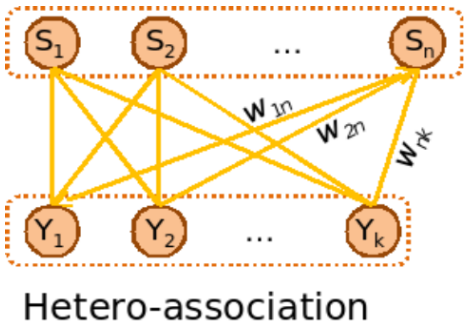

Figure 1: Hetero-associative Hopfield Network

3 Simulation Results

In this section, we briefly overview simulation results.

Key Assumptions

For clarity, we have listed our assumptions as follows:

- We choose $M = n$.
- We assume that neural update rule is fixed. However, we use stochastic neurons, i.e. they update probabilistically according to their input weighted sum h_i, as given by equation s(4).

$$s_i = \begin{cases} 1, & \text{with probability } \frac{1}{1+e^{-2\beta h_i}} > \theta \\ -1, & \text{Otherwise} \end{cases} \tag{4}$$

- In contrast to previous cases, we do multiple iterations of neural update rule. In fact, we wait until neurons converge to a stable state or a maximum of 1000 iteration is done.
- Simulated annealing: Furthermore, we assume that the temperature parameter in the stochastic neural update rule is high in the beginning of iterations and reduces gradually. In terms of the β parameter, we assume it is relatively small in the beginning and grows gradually. We fix the initial β to some value β_0 and then increase it as $\beta = \beta_0 \times itr$, where itr is the number of neural update iteration.
- As mentioned before, the input pattern to the network in the recall phase is $\Omega_\mu \underline{x}^\mu$ and not $\underline{x}^\mu$, for some μ. We expect the network to converge to an state which gives us $\underline{x}^\mu$ in the output.

Having the set of above assumptions, we have summarized the performance of Hopfield networks in the table below for the case of $M = n = 127$ and 400 random patterns fed to the network. The elements inside each box show the error rate, i.e. the number of erroneously recalled patterns divided by the number of patterns simulated (equal o 1200 for this report).

Initial Number of bit errors	3	4	5	6	7	8
$\beta_0 = 10$	0	0.0008	0.0008	0.0042	0.0108	0.0625
$\beta_0 = 100$	0	0.0008	0.0017	0.0050	0.0117	0.0483
$\beta_0 = 1000$	0.0008	0	0.0017	0.0058	0.0183	0.0542

As you can see from the figures, the scheme is able to recover at least 4 bits of errors and as the number of initial bit flips increase, so does the number of final erroneous patterns. Furthermore, the initial temperatures affect the error rate and one must fine-tune this parameter to get the best result.

4 Partial Analysis of the Proposed Method

In this section, we make an effort to mathematically analyze the simulation results. Nevertheless, as will be seen shortly, the analysis is not complete and needs further investigation. We start by replacing the explicit form of Gold sequences, as given by equation (2), and the weight generation of equation (1), into the formula to calculate the weighted input sum of neuron i, assuming that the input to the network is $\underline{s} = \Omega_p \underline{x}^p$:

$$\begin{aligned}
h_i &= \sum_{j=1}^{n} w_{ij} s_j = \sum_{j=1}^{n} w_{ij} \Omega_p x_j^p \\
&= \frac{\Omega_p}{n} \sum_{\mu=0}^{M-1} \Omega_\mu (-1)^{T(\alpha^{i+\mu})+T(\alpha^{di})} \sum_{j=1}^{n} (-1)^{T(\alpha^{j+\mu})+T(\alpha^{dj})} (-1)^{T(\alpha^{j+p})+T(\alpha^{dj})} \\
&= \frac{\Omega_p (-1)^{T(\alpha^{di})}}{n} \sum_{\mu=0}^{M-1} \Omega_\mu (-1)^{T(\alpha^{i+\mu})} \sum_{j=1}^{n} (-1)^{T(\alpha^{j+\mu})+T(\alpha^{j+p})} \\
&= \frac{(-1)^{T(\alpha^{di})}}{n} \left[n(-1)^{T(\alpha^{i+p})} + \Omega_p \sum_{\mu=0\neq p}^{M-1} \Omega_\mu (-1)^{T(\alpha^{i+\mu})} \sum_{j=1}^{n} (-1)^{T(\alpha^{j+\mu})+T(\alpha^{j+p})} \right] \\
&= x_i^p + \frac{(-1)^{T(\alpha^{di})}}{n} \Omega_p \sum_{\mu=0\neq p}^{M-1} \Omega_\mu (-1)^{T(\alpha^{i+\mu})} \sum_{j=1}^{n} (-1)^{T(\alpha^{j+\mu})+T(\alpha^{j+p})} \qquad (5)
\end{aligned}$$

The summation $\sum_{j=1}^{n} (-1)^{T(\alpha^{j+\mu})+T(\alpha^{j+p})}$ is equal to -1 since we are doing a summation over the whole finite field except zero as we made sure to exclude the case of $\mu = p$. Hence, by replacing $\Omega_\mu = (-1)^{T(\alpha^{-\mu})}$, the above equation reduces to:

$$h_i = x_i^p - \frac{(-1)^{T(\alpha^{di})} \Omega_p}{n} \sum_{\mu=0\neq p}^{M-1} (-1)^{T(\alpha^{-\mu})} (-1)^{T(\alpha^{i+\mu})} \qquad (6)$$

Therefore, to be able to mathematically analyze the behavior of the system in presence or absence of noise, we have to analyze the term $\sum_{\mu=0\neq p}^{M-1} (-1)^{T(\alpha^{-\mu})} (-1)^{T(\alpha^{i+\mu})}$, which is not as easy at seems because of the $(-1)^{T(\alpha^{-\mu})}$ element. However, as it turns out, this the crucial factor in determining the good performance of the new scheme. The analysis of the input sum in presence of noise reduces to investigating similar summations as well.

5 Conclusions and Future Works

So far, we have suggested a scheme which is able to to store $M = n$ patterns with a network of n nodes using a modified weighting rule [1] with the ability to correct a number of erroneous bits (5 for the case of $n = 127$).

The proposed scheme has some favorable properties:

- It is iterative and the performance is enhanced over iterations.
- If we consider the hetero-association case, it seems that we will have more degrees of freedom and then we can find good set of *pairs* to memorize with the usual Hopfield rule.
- The degree of stochasticity (the parameter β) can be fine-tuned to give us the best performance.

To mathematically analyze the behavior of the system, we shall investigate equations of the form (6), which is not easy to deal with. This is the subject of our future research.

References

[1] R. Gold, "Optimal binary sequences for spread spectrum multiplexing", IEEE Transactions on Information Theory, Vol. 13, No. 4, 1967, pp. 619 −621.

[1] If one likes to think of the suggested scheme as *hetero-association*, then we have $2n$ nodes. However, the weighting method is exactly the Hopfield version so we have used the same weighting scheme to increase the capacity in this case.