ISRN SICS-R--95/13--SE

Convergence and new operations in SDM

by

Gunnar Sjödin

SICS Research Report
R95:13
ISSN 0283-3638

Contents

1 Introduction **4**

1.1 The SDM model 4

1.2 Notations 6

1.2.1 Variables and parameters 6

1.2.2 The parameter values for cerebellum 7

2 Good reading and convergence **8**

2.1 The stopping criterion 8

2.2 How to improve the address 8

2.3 Masking out the faulty part of the address 9

2.3.1 The radical method 9

2.3.2 Some estimates 11

3 New operations on the memory **12**

3.1 Avoiding multiple storings 12

3.2 (Partially) erasing a stored value 13

3.3 SDM as a short term memory 13

3.4 Spurious states 13

1 Introduction

1.1 The SDM model

The Kanerva Sparse Distributed Memory (SDM) is a memory with a set of M locations of U-dimensional vectors of integers. The memory is addressed within a binary address space of dimension N, where $M \ll 2^N$.

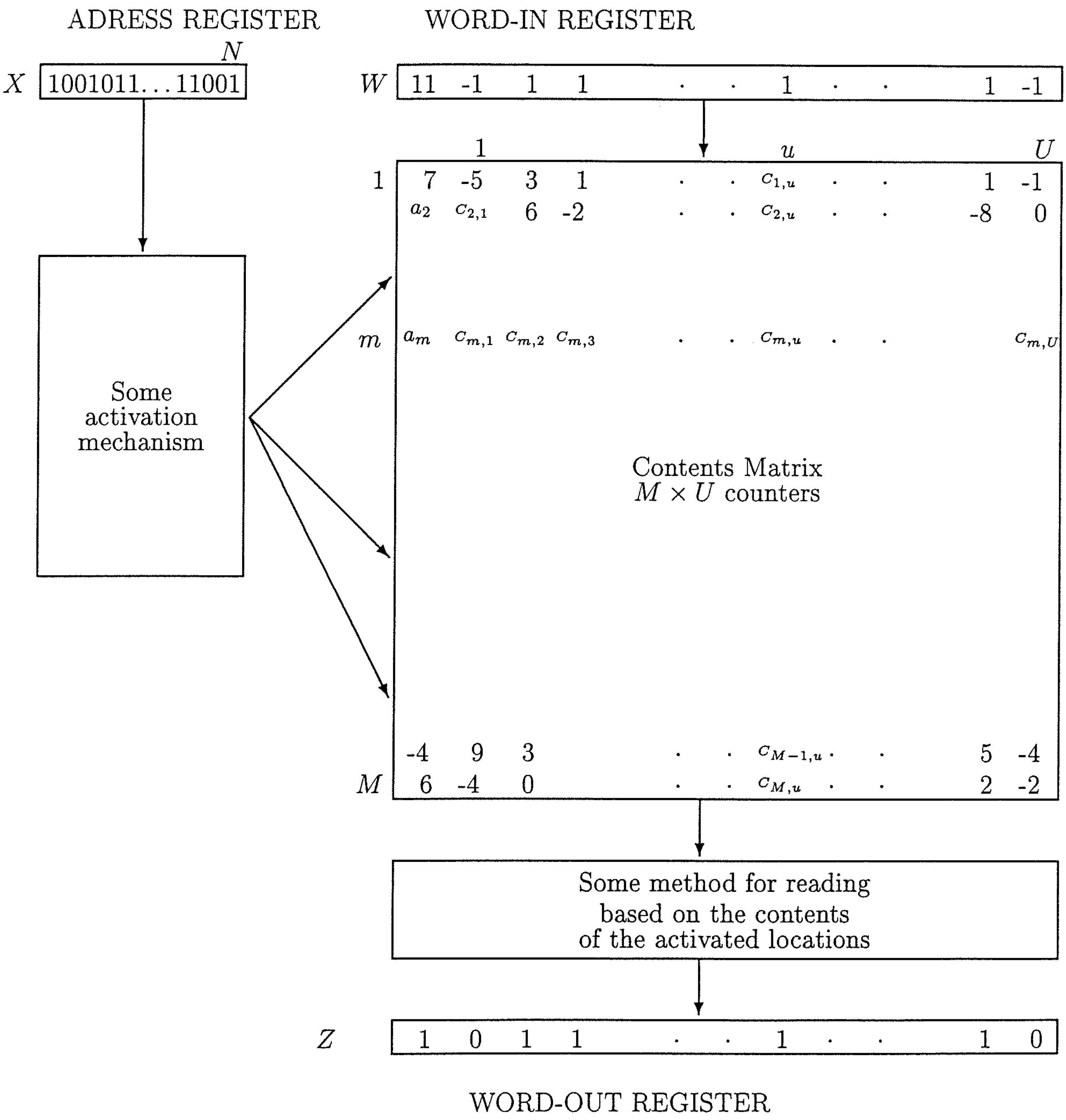
A picture of the general structure is given in **figure 1**. For relevant literature on SDM cf. [3], [4], [5], [2], [7], [1].

When a U-dimensional binary vector W is stored "at" an address X an activation mechanism activates a set of the locations and W is added to the contents of these after first transforming 0:s to -1:s.

In the original SDM model every location was given a binary address and those locations that had a sufficiently small Hamming distance to X were activated. In the Jaeckel/Karlsson model each location m is given a set **mask**(m) of K coordinates, i.e. a set of indices ranging from 1 to N. To each of those indices is associated either 0 or 1. If, for a given location, the associated values in this "mask" coincides with the values of the corresponding mask for X then the location is activated. In general an activation mechanism should have the robustness property that two addresses, with small Hamming distances from each other, should activate roughly the same set of locations.

When we want to read the memory at an address X, the same activation mechanism as when storing is at work. In the original SDM model the sum of the activated locations was thresholded for each index, or position as we will call it in this paper, at 0; i.e if a value was smaller than 0 it was transformed to 0 and otherwise to 1. This reading procedure can be considerably improved, as is demonstrated in Sjödin [1]. The involved methods are independent of the activation mechanism.

A major concern of this paper is the problem of converging. That is, if we read at an address X which is close to an address X_0, at which we have stored a datum, we would like to move X towards the unknown X_0 in order to get a better reading. This could be done with any of the above activation mechanisms but with the Karlsson/Jaeckel we are at an advantage to construct an efficient procedure. In the sequel, we will assume that one of this models are used.



ADRESS REGISTER
WORD-IN REGISTER
N
X
1001011...11001
W
11 -1 1 1 . . 1 . . 1 -1
1
u
U
1
7 -5 3 1 . . $c_{1,u}$ . . 1 -1
a_2 $c_{2,1}$ 6 -2 . . $c_{2,u}$ . . -8 0
m
a_m $c_{m,1}$ $c_{m,2}$ $c_{m,3}$ . . $c_{m,u}$ . . $c_{m,U}$
Some activation mechanism
Contents Matrix
$M \times U$ counters
-4 9 3 . . $c_{M-1,u}$ . . 5 -4
M
6 -4 0 . . $c_{M,u}$ . . 2 -2
Some method for reading based on the contents of the activated locations
Z
1 0 1 1 . . 1 . . 1 0
WORD-OUT REGISTER

Figure 1: SDM

1.2 Notations

The notations follow closely those of [1].

An *address* is a binary vector of 0:s and 1:s. Its indices are called *coordinates*.

A *datum* is a vector of -1:s and 1:s. Its indices are called *positions*.

A *mask* is a subset of the coordinates in the address vectors.

A *location*, in other papers usually called *hard location*, is a storage vector for data.

We also use the generic notations

$$
\begin{aligned}
\widehat{xx} &= \text{a stochastic variable with typical outcome } xx \\
a &\asymp b \\
&\text{means that } \log a \sim \log b \\
&\text{i.e. } a = e^{\alpha}, b = e^{\beta} \\
&\text{where } \lim \frac{\alpha}{\beta} = 1 \\
&\text{when some variable tends to } \infty
\end{aligned}
$$

1.2.1 Variables and parameters

$$
\begin{aligned}
M &= \text{the number of locations} \\
N &= \text{the length of an address} \\
U &= \text{the length of a datum} \\
T &= \text{the number of data stored in the memory} \\
\lambda &= \frac{T}{M}\text{, i.e. the ``load'' of the memory} \\
K &= \text{the number of chosen coordinates in a mask} \\
\mathbf{mask}(m) &= \text{the coordinates used in the mask belonging to } m \\
\rho &= \text{the probability that an arbitrary location is activated} \\
&\quad\ \text{in a read/write operation.} \\
&\quad\ \text{Usually we will assume that the activation mechanism}
\end{aligned}
$$

is such that

$\rho = (2\lambda M^2)^{-\frac{1}{3}}$

(cf. e.g. [4] and [6])

In particular, in the Jaeckel/Karlsson model,

we will have

$$2^{-K} = \rho = (2\lambda M^2)^{-\frac{1}{3}} \tag{1}$$

X^{t_0} = the address used at "time t_0"

X = the address close to X^{t_0} at which we read the memory

$\mathcal{H}$ = the locations activated by X

H = the number of locations in $\mathcal{H}$

This is actually a fixed size parameter in the Karlsson model

$\mathbf{h}$ = the locations activated by both X and X^{t_0}

h = the number of locations in $\mathbf{h}$

h_Q = the, by methods in [1], estimated value of h

1.2.2 The parameter values for cerebellum

Recall that in the SDM-interpretation of the cerebellum

M = the number of locations = 10^{10-11}

= number of granule cells

N = the length of an address = $3 \cdot 10^7$

= number of mossy fibers

= $M^{0.699} - M^{0.707}$

U = the length of a datum = 10^6 or $1.5 \cdot 10^7$

= number of Purkinje cells

= $M^{0.6} - M^{0.55}$ or $M^{0.74} - M^{0.67}$ respectively

2 Good reading and convergence

For the "good" reading the procedure is:

1 Use the address to activate the locations $\mathcal{H}$.

2 Find the locations $\mathbf{h}$ as in 6 in Sjödin [1].

3 Use only these locations to read the data
with some method from 2 in [1].

The general procedure for convergence towards the correct address is:

1 Use the address to activate the locations $\mathcal{H}$.

2 Find the locations $\mathbf{h}$ as in as in 6 in [1].

3 If the "stopping criterion" (cf. **section** 2.1) is satisfied, use $\mathbf{h}$ to read the data with some method from 2 in [1]. Otherwise continue to **4.**

4 Use the locations in $\mathbf{h}$ to find a better address (cf. **section** 2.2) to read from and go back to **1.** with this address.

2.1 The stopping criterion

In the Jaeckel/Karlsson model we may use the stopping criterion

$$h_Q \approx H \tag{2}$$

However, in the Jaeckel model, $h_Q = h = H$, may be satisfied although we have not reached the correct address. With the Karlsson activation method, the number of activated locations is always the same, and thus we do not have this problem.

2.2 How to improve the address

In the basic, Hamming distance activation model we may take the addresses, corresponding to the $\mathbf{h}$-locations, to construct an "average address" by majority vote for each coordinate in an address. This turns out to work if h is very big and the activation probability is small. Then the constructed address will end up between the used and the correct address. This requires a very big number of locations to work properly.

Instead we will concentrate on the Jaeckel/Karlsson model. In this case we can say that the bit-values in the selected coordinates corresponding to a location in $\mathbf{h}$ are definitely correct. On the other hand the activated non-$\mathbf{h}$-locations give sets of coordinates containing at least one coordinate with an incorrect bit in the given address. Using this, it is possible to improve our address. If the memory is big the number of activated locations for a reading will be big and so will the number of $\mathbf{h}$-locations. *It may then be possible to directly change all the bits in the coordinates of the address not contained in the union of coordinates corresponding to the* $\mathbf{h}$*-locations.* If there is an error in the calculation of the $\mathbf{h}$-locations this will impair the performance of the algorithm but not destroy it.

Below, we will describe these principles in more detail.

2.3 Masking out the faulty part of the address

In the following, for a set of locations $\mathcal{A} \subset [1, M]$, let

$$\mathbf{coord}(\mathcal{A}) \;=\; \bigcup_{m \in \mathcal{A}} \mathbf{mask}(m) \subset [1, N]$$

Now, assume that

$$m \;\in\; \mathcal{H} - \mathbf{h}$$

Then at least one of the coordinates in $\mathbf{mask}(m)$ must have different bit-values in X and X^{t_0}. In particular, if only one of the coordinates of $\mathbf{mask}(m)$ is outside $\mathbf{coord}(\mathbf{h})$ then that coordinate has different bit-values in X and X^{t_0}. How to best utilize the information, in order to move closer to X^{t_0} from X is an interesting issue; it *does* depend on the parameters of the memory and the size of h.

We will now describe a method that works for big memories.

2.3.1 The radical method

Let d be the Hamming distance between X and X^{t_0}. Let κ be the number of coordinates not in $\mathbf{coord}(\mathbf{h})$ but for which X and X^{t_0} have the same bit-values. Or, in other words, κ is the number of misses of the bits, in the part of the addresses that coincide. Then, if $\kappa < d$, flipping all the X-bits with coordinates in

$[1, N] - \mathbf{coord(h)}$

will bring X closer to X^{t_0}. Of course, we will not know whether the inequality is satisfied or not. However, as we will see, given h_Q it is possible to state this with a certain confidence.

One should really only flip the relevant bits with coordinates in $\mathbf{coord}(\mathcal{H})$. However, in the Karlsson design the set $\mathbf{coord}(\mathcal{H})$ is independent of X and thus only this set is relevant anyway. In fact, it may be arranged that this constant set equals $[1, N]$.

Let us make this a little more precise. Let $\varepsilon = \frac{d}{N}$, i.e. the proportional error in the address. Now, replace the expected number of common locations h, given $\widehat{H} = H$, by h. Then we get the following rough estimate (cf. equation 37 in [1])

$$h \approx H \cdot e^{-K\varepsilon} \tag{3}$$

and hence

$$\varepsilon \approx \frac{1}{K} \log \frac{H}{h} \tag{4}$$

Given $\widehat{h} = h$ we get that the expected value of κ is

$$\kappa = (N-d)(1 - \frac{K}{N-d})^h \tag{5}$$

$$\approx (N-d)e^{-\frac{Kh}{N-d}} \tag{6}$$

and hence $\kappa < d$ is roughly transformed to

$$(1-\varepsilon) \log(\frac{1}{\varepsilon} - 1)e^{K\varepsilon} < \frac{KH}{N} \tag{7}$$

when $d > 0$. The stopping criterion handles the case $d = 0$. Solving (7) for h, by using the relation in (4), and then replacing h by h_Q we get a criterion

$$\mathbf{const1} < h_Q < \mathbf{const2} \tag{8}$$

for when to assume that $\kappa < d$ and thus when to use the "radical" flipping method. It is also easy to see that if M is big the right-hand restriction may be removed, i.e. $\mathbf{const2} = H$.

2.3.2 Some estimates

Let a fixed ε and a fixed K be given. Then when increasing M, $H \asymp M$ and the largest N satisfying (7) will also be of size $\asymp M$. On the other hand, using the usual formula (1) for optimal activation probability we get

$$K = \frac{1}{3}\log_2(2\lambda M^2) \tag{9}$$

and this gives that the maximal N is of order $M^{\frac{1}{3}(1-\frac{2\varepsilon}{\log 2})}$. In the cerebellum, interpreted as an SDM (cf. **section** 1.2.2), the memory is not fully connected, the K:s are small and $N = M^{\approx 0.7}$. The investigation of how the convergence method and other methods in this paper and in Sjödin [1] should be made to work for the "thin" SDM-structure of cerebellum is an interesting issue.

3 New operations on the memory

3.1 Avoiding multiple storings

When **storing**:
We can figure out whether we are "intruding" upon another storage area by calculating the "h_Q" as in Sjödin [1]. Set limits "**lim1**" and "**lim2**" and proceed as follows:

If, case 1:

$h_Q \ < \ \mathbf{lim1}$

then store in the usual way.

If, case 2:

$\mathbf{lim1} \ \leq \ h_Q \ < \mathbf{lim2}$

then change the address and store in this new address (possibly recursively). The change of address is according to some sequence of of "robust" methods, moving it far away, as e.g. flipping all the bits.

If, case 3:

$\mathbf{lim2} \ \leq \ h_Q$

then *don't store* or issue a warning or do something else as e.g. erase the old value, partly or completely (cf. **section** 3.2), and store the new value.

When **reading**:

If, case A

$h_Q \ << \ \mathbf{lim2}$

then change the address and read again. If (iteratively) no big h_Q can be found, then probably nothing is stored.

If, case B

$h_Q \ >> \ \mathbf{lim2}$

Then, read in the ordinary way

If, case C

$$h_Q \approx \mathbf{lim2}$$

then there are two options to try.

This can of course be done recursively and also result in a number of choices for the read data. These procedures spreads the stored data and thus gives the memory better storing capacity. It furthermore gets rid of the problems with multiply stored data (cf. section 5 in [1]).

3.2 (Partially) erasing a stored value

Given a set of locations $\mathcal{A} \subset [1, M]$, we may use these to read the memory to obtain a datum $\widetilde{W}$ and then "erase" this by adding $-\widetilde{W}$ to the data in in the $\mathcal{A}$-locations. Note that we do not set the values at these locations to zero. Using this with $\mathcal{A} = \mathbf{h}$, we may partially erase W when reading at the address X. If we read at the correct address, possibly after converging to it (cf. **section** 2), W will be completely erased.

3.3 SDM as a short term memory

Use as addresses a cyclic counter of time. When a new value is to be stored, we erase the old one as above and store the new one. If for some reason the erasing starts failing (e.g. since we calculate the h_Q-locations badly) the memory becomes senile.

3.4 Spurious states

In the ordinary iterative reading of addresses stored in addresses, one may end up in spurious states. These are somewhat mysterious, but if the h_Q-value is small, one may conclude that

> "we don't know - this is a spurious state".

One reason for a state to be a spurious state could be that it is a double/multiple reading. If this is the case it could be possible to sort out the different addresses by the methods described in this paper (cf. section 5 in [1]).

References

[1] Gunnar Sjödin. Improving the Capacity of SDM, 1995. SICS Research Report, to appear.

[2] L. A. Jaeckel. An alternative Design for a Sparse Distributed Memory. Technical Report TR-89.28, RIACS, 1989.

[3] Pentti Kanerva. *Sparse Distributed Memory.* The MIT Press, 1988.

[4] Pentti Kanerva. Sparse distributed memory and related models. In Mohamad H. Hassoun, editor, *Associative Neural Memories*, chapter 3, pages 50–76. Oxford University Press, 1993.

[5] Roland Karlsson. A Fast Activation Mechanism for the Kanerva SDM Memory, 1995. SICS Research Report R95:10, Swedish Institute of Computer Science.

[6] Jan Kristoferson. Best Probabilities of Activation and Performance Comparisons for Several Designs of Kanerva's SDM (Sparse Distributed Memory), 1995. SICS Research Report R95:09, Swedish Institute of Computer Science.

[7] Jan Kristoferson. Some comments on the Information Stored in Sparse Distributed Memory, 1995. SICS Research Report, to appear.