ISRN SICS-R--95/10--SE

Evaluation of a Fast Activation Mechanism for the Kanerva SDM Memory

by

Roland Karlsson

SICS Research Report
R95:10
ISSN 0283-3638

Report R95:10

ISRN : SICS-R--95/10-SE
ISSN : 0283-3638

Evaluation of a Fast Activation Mechanism for the Kanerva SDM Memory

Roland Karlsson (roland@sics.se)

RWCP[1] Neuro SICS[2] Laboratory
Box 1263, S-164 28 Kista, Sweden

December 1995

Abstract

THE Kanerva Sparse Distributed Memory (SDM) is a mechanism for implementing a memory with a huge address space. The physical memory consists of substantially fewer locations than the virtual address space. In earlier implementations of the SDM memory the addresses of all physical locations had to be scanned to determine which locations to activate. This is time-consuming, and massive parallel hardware support is needed for other than small applications. This paper introduces a new mechanism for finding the active locations. Using this access mechanism, the access time may be speeded up several orders of magnitude.

Keywords: *SDM, sparse distributed memory, selected-coordinate design.*

Contents

---

[1]Real World Computing Partnership
[2]Swedish Institute of Computer Science

1 Introduction

THE Kanerva Sparse Distributed Memory (SDM) [4, 5] is a clever mechanism for storing/retrieving data in/from a memory with a huge address space. The memory addresses, also called input addresses, are N-bit vectors (N is generally large). The actual memory contains substantially fewer memory locations (called hard locations) than there are possible input addresses. When accessing the memory, a number of hard locations "close to" the input address are activated. In the basic SDM, each hard location has an N-bit address, and the activation criterion is: "close" according to Hamming distance (i.e., the number of mismatched address bits) between the input address and the location's address.

This paper really concerns only the activation mechanism for the SDM, but briefly, the write and read operations work as follow: The contents of a hard location is a vector of integers where each integer corresponds to one bit in the input data. When writing to the memory, the data bits (from the input data vector) are added (adding -1 when the data bit is 0 and adding $+1$ when the data bit is 1) to the contents of all activated hard locations. When reading, the content vectors of the activated hard locations are summed and the sum is converted to the output data by some threshold mechanism. The key idea is that data written sufficiently "far away" (i.e., having few common activated hard locations) from the address can be considered as noise that will more or less cancel out. To implement a good threshold mechanism is somewhat involved but details can be found in [3].

Several activation mechanisms have been used, e.g., Hamming distance or the Jaeckel Selected-Coordinate Design [1, 2]. The activation mechanisms have all been based on matching the input address with some "matching pattern" for each hard location. This makes the implementations impractical without some massive parallel hardware support (except for small problems).

This paper introduces an efficient mechanism for finding the hard locations that are "close to" the input address. The proposed mechanism may be viewed as an efficient implementation of a restricted Jaeckel design. It is suitable for execution on sequential machines or machines with a moderate level of parallelism (say, tens to hundreds of processors). The implementation, apart from being much faster, gives approximately the same signal-to-noise ratio.

The rest of the paper is organized as follows. Section 2 describes the activation mechanism in the Jaeckel Selected-Coordinate Design. Section 3 describes the activation mechanism in the new proposed design. Section 4 compares the performance of the Jaeckel and the new design. And finally Section 5 concludes the paper.

3 bits in the address are used as an index to one of 8 positions in a block. The hard-location memory consists of A such blocks, i.e., the size of the hard-location memory is therefore $M = A \cdot 2^k$.

Compared to the earlier designs, the number of access calculations is decreased by a factor 2^k. The number of bits (k) is usually in the range $5 - 20$, making the decrease in access time between 2 and 6 orders of magnitude.

The activation patterns in the table are substantially fewer than in the Jaeckel design (A patterns vs. M patterns). It is not a good idea to generate the activation patterns in the table at random. A bad choice of activation patterns might lead to very unsatisfactory results. In order to remove this problem, a restriction is introduced for choosing bits in the activation patterns: a bit cannot be chosen if there exists any other bit that have been used less times. Note that this optimization is neither necessary nor used in the Jaeckel model.

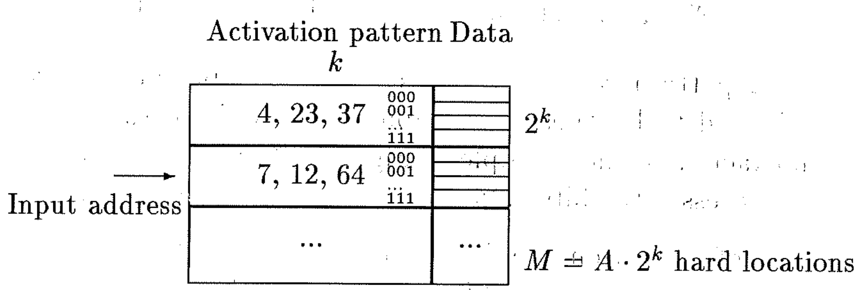

Figure 3: The new design implemented as a restricted Jaeckel Design.

The new design can be viewed as a restricted version of Jaeckel's design (see Figure 3), with identical functionality (but much slower) since the new design could be implemented by dividing Jaeckel's hard-location memory into A blocks each with 2^k locations, and by using in each block all combinations of 0s and 1s in a fixed subset of the address bits.

The two main advantages of the new design are: (1) it is much faster and (2) the number of activations is fixed to A. The latter result is used in [3] to get better convergence. One disadvantage may be that it is less random, e.g., there may be problems with interference between the choice of activation pattern and some input data. One constraint on the basic version of the new design is that the number of hard locations is fixed to $M = A \cdot 2^k$. This can of course be remedied by not using all 2^k locations.

References

[1] Jaeckel L. A. An Alternative Design for a Sparse Distributed Memory. Technical report, Research Institute for Advanced Computer Science, NASA Ames Research Center, 1989.

[2] Jaekel L. A. A Class of Designs for a Sparse Distributed Memory. Technical report, Research Institute for Advanced Computer Science, NASA Ames Research Center, 1989.

[3] Sjödin G. Improving the Capacity and Capabilities of SDM. Paper in preparation.

[4] Kanerva P. *Sparse Distributed Memory.* The MIT Press, 1988.

[5] Kanerva P. Sparse distributed memory and related models. In Mohamad H. Hassoun, editor, *Associative Neural Memories.* Oxford University Press, 1993.