# On the Role of Time and Space in Neural Computation

Wolfgang Maass

Institute for Theoretical Computer Science
Technische Universität Graz
Klosterwiesgasse 32/2
A-8010 Graz, Austria
maass@igi.tu-graz.ac.at

**Abstract.** We discuss structural differences between models for computation in biological neural systems and computational models in theoretical computer science.

## 1 Introduction

One of the most interesting scientific developments during the next two decades will be the unraveling of the structure of computation in living organisms. Since the information processing capabilities of living organisms are in many aspects superior to those of our current artificial computing machinery, this is likely to have significant consequences for the way in which computers and robots will be designed in the year 2020.

Traditionally theoretical computer science has played the role of a scout that explores novel approaches towards computing well in advance of other sciences. Curiously enough, this did not happen so far in the case of computation in living organisms, and it may be worthwhile to ponder for a moment about the possible reasons for that. One obstacle may result from the fact that theoretical computer science has become to a large extent "technique-driven", i.e., one typically looks for new problems that can be solved by variations and extensions of a body of fascinating mathematical tools that one has come to like, and that form the heart of current theoretical computer science. In contrast, to have a serious impact on theoretical research in neural computation, a theoretical researcher has to be to a larger extent "problem-driven", i.e., he/she has to employ and develop those mathematical concepts and tools that are most adequate for the problem at hand.

On the positive side, I would like to mention a success story regarding an earlier very fruitful interaction between the areas which are nowadays called compuational neuroscience and theoretical computer science. McCulloch and Pitts [McCulloch and Pitts, 1943] developed an abstract mathematical model for computation in living organisms: circuits of *"McCulloch-Pitts neurons"* or *threshold gates*, as they are now called in theoretical computer science. Kleene [Kleene, 1956] proved that they were equivalent to his notion of a *finite automaton*. Thus "historically finite automata were first used to model neuron nets"

[Hopcroft and Ullman, 1979]. As we can all see, both models turned out to be a very fruitful for the subsequent development of both fields involved.

Within the scientific discipline of neuroscience a new subfield has emerged during the 90's that is called *computational neuroscience.* However a look at any recent issue of leading journals (e.g. Neural Computation, Network: Computation in Neural Systems, Computational Neuroscience) or conference proceedings in this new area (e.g. of the Annual Conference on Computational Neuroscience) may have a sobering effect on a theoretical computer scientist who is ready to develop adequate computational theories for computational neuroscience: There exists a large amount of interdisciplinary work in computational neuroscience and a fair number of theoretical work has already been done in this area. But so far the theoretical work in computational neuroscience has been dominated by approaches from theoretical physics, information theory, and statistics.[1]

One of the main obstacles for a theoretical computer scientist who is ready to tackle theoretical problems about computing in biological neural systems is the diversity of models for neural computation that are proposed by neuroscientists, and the diversity of opinions among leading neuroscientists regarding the right way to understand computations in the brain. This has the effect that it is hardly possible to identify abstract models and theoretical problems in computational neuroscientists that are of undisputed significance. In fact, it is hard to identify solid empirical facts regarding computation in biological neural systems on which most researchers and laboratories agree.[2] This concerns especially the first questions that a theoretical computer scientist is likely to ask:

- How is information encoded in biological neural systems?
- What are the computational units of biological neural systems, and what functions can they compute?
- How are biological computations organized and programmed?
- Which maps from inputs to outputs are computed by specific neural systems?

We will focus in this short article on one issue on which most neuroscientists seem to be able to agree, and which may point to a fruitful area for future contributions from theoretical computer science to this field: that time and space appear to play a different role for computations in biological neural systems than for computations in currently existing computational models in theoretical computer science. This may provide some "food for thought" for the development of new models that are both of interest from the point of view of theoretical

[1] For a theoretical computer scientist from Europe a survey of the current state of computational neuroscience may have an additional sobering effect: most leading journals and conferences are based in the USA.

[2] This is a consequence of the fact that most of the basic questions about computations in living organisms cannot be answered directly through suitable experiments. Hence the available answers are typically based on indirect empirical evidence, that often varies with details of the experimental setup, the specific neural system and species that is studied, and the methods for data-analysis that are employed. Bad tongues say that the answers also depend on the theoretical hypothesis that the researcher wants to support through the experiment.

computer science, and which also provide fruitful new hypotheses regarding the organization of computations in living organisms.

In the remainder of this article I will illustrate in a few examples specific directions into which currently existing computational models from theoretical computer science need to be evolved in order to take into account the different role that time and space play in biological neural systems. In view of the space constraints for this article we cannot give here a survey of relevant results and theories. But we will give in the last section some pointers to up-to-date survey articles and books.

## 2 On the Role of Time in Neural Computation

The most biology-like computational model that we traditionally consider in theoretical computer science are circuits consisting of threshold gates or sigmoidal gates. These gates compute functions from $\mathbf{R}^n$ into $\mathbf{R}$ of the form

$$\langle x_1, \ldots, x_n \rangle \mapsto \sigma(\sum_{i=1}^{n} w_i x_i + w_0) \qquad (1)$$

for some fixed "activation function" $\sigma : \mathbf{R} \to \mathbf{R}$ (e.g., $\sigma(y) = \text{sign}\ (y)$ in the case of a threshold gate, or $\sigma(y) = 1/(1 + e^{-y})$ in the case of a sigmoidal gate) and suitable parameters $w_0, \ldots, w_n \in \mathbf{R}$. A computation in such circuit is defined with the help of some input-dependent schedule, which decides which gates carry out their computational operation (1) at which discrete time step. This computation schedule is particularly obvious in the case of a layered feedforward circuit, where all gates on level $\ell$ carry out their computational operation at time step $\ell$.

In contrast, the *output* of a biological neuron consists of "action potentials" or *"spikes"* (see Fig. 1). In other words: a *biological neuron* does not output any number or bit, instead it simply *marks points in time.* The *input* to a biological

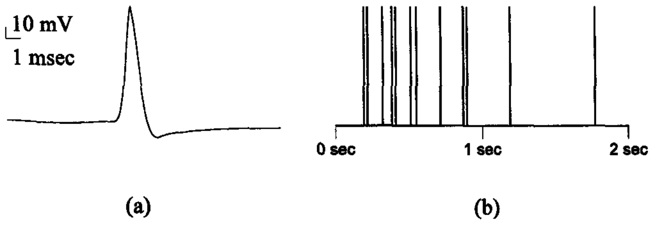


**Fig. 1.** a) Typical action potential (spike). b) A typical spike train produced by a neuron (each firing time marked by a bar)

neuron $v$ consists of trains of pulses, socalled excitatory postsynaptic potentials (EPSP's) or inhibitory postsynaptic potentials (IPSP's) of a shape as indicated

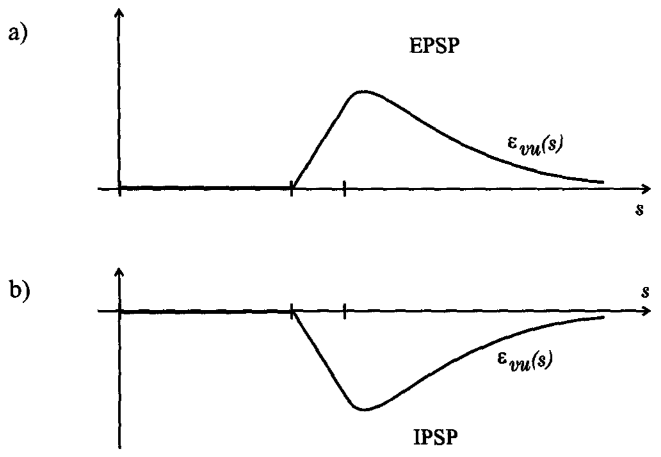


**Fig. 2.** a) Typical time course of an excitatory postsynaptic potential (EPSP). b) Typical time course of an inhibitory postsynaptic potential (IPSP). The vertical axis indicates the membrane voltage of the neuron $v$

in Fig. 2. More precisely: About 1000 to 10000 other neurons $u$ are each connected to $v$ by a *synapse.* The synapse from neuron $u$ to neuron $v$ transforms the output spike train of neuron $u$ (which is of a type as illustrated in Fig. 1 b)) into a train of EPSP's or IPSP's in neuron $v$. One usually assumes that neuron $u$ only causes EPSP's or only causes IPSP's in other neurons $v$.

According to the *spike response model* (see [Gerstner and van Hemmen, 1994] and [Gerstner, 1998] one can model the response of the membrane potential of neuron $v$ at time $t$ to a spike train with spikes at times $t_1, t_2, \ldots$ from a presynaptic neuron $u$ by a function of the form

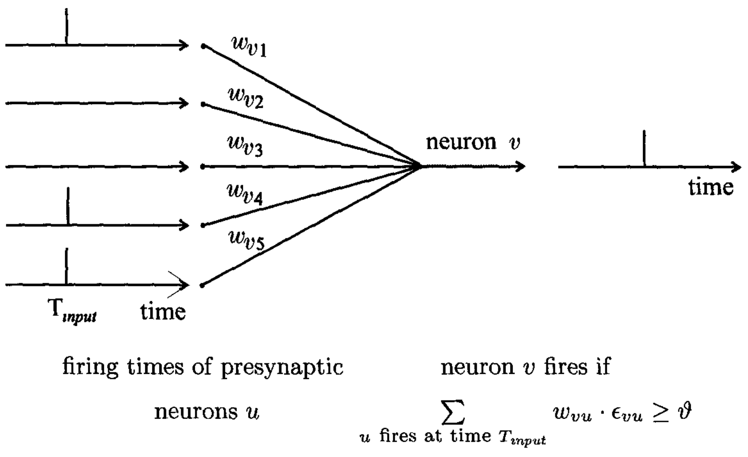
$$\text{response}_{vu}(t) := \sum_i w_{vu}(t) \cdot \varepsilon_{vu}(t - t_i) \ .$$

One assumes in this model that neuron $v$ *"fires"* – and thereby emits a spike – at time $t$ whenever the resulting total membrane potential

$$h_v(t) := \sum \{\text{response}_{vu}(t) \mid u \text{ has a synapse to } v\}$$

at neuron $v$ reaches the *firing threshold* $\vartheta_v(t)$ of neuron $v$. One refers to this model as a *leaky integrate-and-fire neuron* or *spiking neuron.*

Let us first assume for simplicity that the synaptic "weights" $w_{vu}(t)$ and the firing threshold $\vartheta_v(t)$ do not depend on the time $t$. Then the spiking neuron $v$ can in principle simulate a threshold gate, provided that all presynaptic neurons $u$ that represent an input variable with value 1 fire at a common time $T_{input}$, and all presynaptic neurons $u$ that represent an input variable with value 0 do not fire at all during a certain time interval; see Fig. 3. To be precise, one also has to make an assumption about the shapes of the response functions, for example that



$w_{v1}$
$w_{v2}$
$w_{v3}$
$w_{v4}$
$w_{v5}$
neuron $v$
time
$T_{input}$
time
firing times of presynaptic neurons $u$
neuron $v$ fires if
$\sum_{u \text{ fires at time } T_{input}} w_{vu} \cdot \epsilon_{vu} \geq \vartheta$

**Fig. 3.** Simulation of a threshold gate by a spiking neuron.

the response functions $\varepsilon_{vu}(s)$ for different $u$ are identical except for their sign. On the other hand empirical data show that an input of the type indicated in Fig. 4 is more typical. In order to illustrate that in such asynchronous mode a spiking

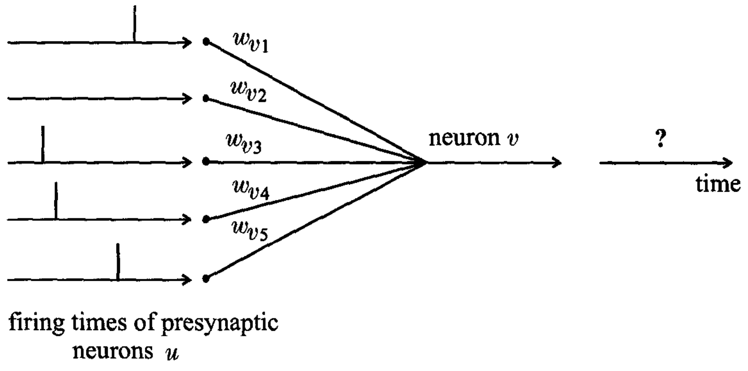


**Fig. 4.** Typical input for a biological spiking neuron $v$, where its output cannot be easily described in terms of conventional computational units.

neuron can carry out computational operations that are not at all reflected in the model of a threshold gate or sigmoidal gate (1), we consider for some arbitrary fixed parameters $0 < c_1 < c_2$ the following function $ED_n : \mathbf{R}^n \to \{\mathbf{0}, \mathbf{1}\}$:

$$ED_n(x_1, \ldots, x_n) = \begin{cases} 1 \text{ , if there are } j \neq j' \text{ so that } |x_j - x_{j'}| \leq c_1 \\ 0 \text{ , if } |x_j - x_{j'}| \geq c_2 \text{ for all } j \neq j' \text{ .} \end{cases}$$

Note that this function $ED_n(x_1, \ldots, x_n)$ (where $ED$ stands for "element distinctness") is in fact a partial function, which may output arbitrary values in case that $c_1 < \min\{|x_j - x_{j'}| : j \neq j' \text{ and } j, j' \in \gamma_i\} < c_2$. Therefore hair-trigger situations can be avoided, and a single spiking neuron can compute this function $ED_n$ even if there is a small amount of noise on its membrane potential $h_v(t)$. We assume here that the inputs $x_1, \ldots, x_n$ to the spiking neuron $v$ are given through the firing times $x_1, \ldots, x_n$ of $n$ presynaptic neurons. On the other hand

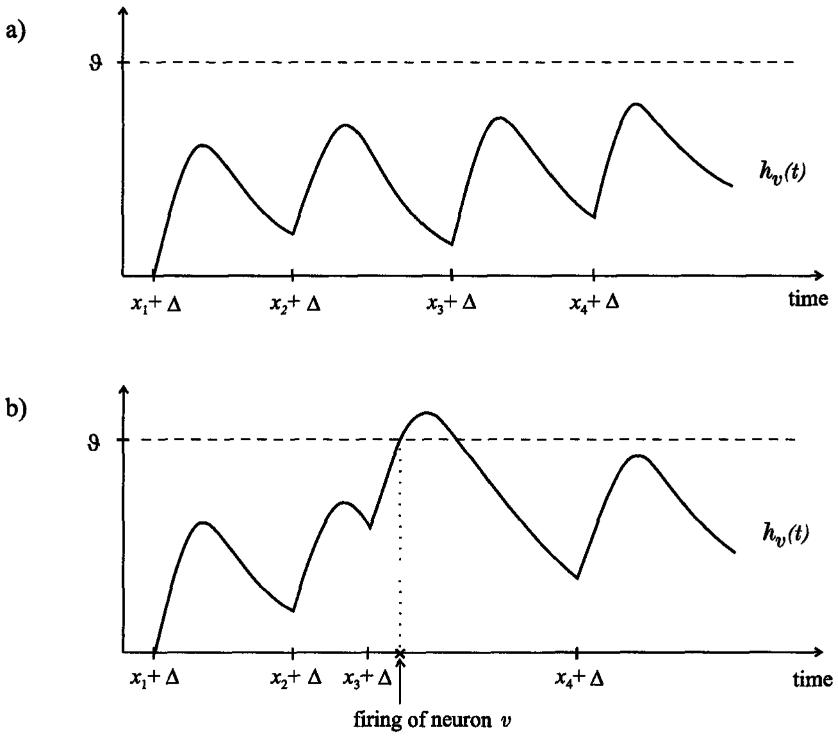


**Fig. 5.** a) Typical time course of the membrane potential $h_v(t)$ if $ED_4(x_1, x_2, x_3, x_4) = 0$. b) Time course of $h_v(t)$ in the case where $ED_4(x_1, x_2, x_3, x_4) = 1$ because $|x_3 - x_2| \leq c_1$.

the following results show that the same partial function $ED_n$ requires a substantial number of gates if computed by circuits consisting of McCulloch-Pitts neurons (threshold gates) or sigmoidal gates.

**Theorem 1.** *Any layered threshold circuit that computes $ED_n$ needs to have at least* $\log(n!) \geq \frac{n}{2} \cdot \log n$ *threshold gates on its first layer.*

The *proof* of Theorem 1 relies on a geometrical argument, see [Maass, 1997].

**Theorem 2.** *Any feedforward circuit consisting of arbitrary sigmoidal gates needs to have at least* $\frac{n-4}{2}$ *gates in order to compute* $ED_n$.

The *proof* of Theorem 2 is more difficult, since sigmoidal gates output *analog numbers* rather than *bits*. Therefore a multilayer circuit consisting of sigmoidal gates can have larger computational power than a circuit consisting of threshold gates (see [Maass et al., 1991,DasGupta and Schnitger, 1996]). The proof procedes in an indirect fashion by showing that any sigmoidal neural net with $m$ gates that computes $ED_n$ can be transformed into another sigmoidal neural net that "shatters" *every* set of $n-1$ different inputs with the help of $m+1$ programmable parameters. According to [Sontag, 1997] this implies that $n-1 \leq 2(m+1)+1$. We refer to [Maass, 1997] for further details. ∎

*Remark 1.* This is the largest lower bound for *any* concrete function in $P$ that has been achieved to date for the size of circuits consisting of sigmoidal gates.

So far we have assumed that the firing threshold $\vartheta_v(t)$ and the synaptic weights $w_{vu}(t)$ are independent of the time $t$. This is certainly not the case for a biological neuron. In a first approximation one may assume that $\vartheta_v(t)$ shoots up to an extremely high value for a few ms after each firing of $v$ and then returns to a "resting value" $\vartheta$. This threshold dynamics enforces an upper bound on the maximal firing rate of a neuron, which usually is in the range of a few hundred Hz (although *typical* firing rates in the cortex are well below 100 Hz).

The dependence of synaptic weights $w_{vu}(t)$ on the time $t$ is substantially more complex. It has been shown that different synapses exhibit quite heterogeneous dependencies on the preceding firing times of the presynaptic neurons [Dobrunz and Stevens, 1997]. Therefore one has to view synapses as another type of *active computational units* in neural computation. They cannot really be viewed as passive "registers" that store a single parameter – the "synaptic weight" $w_{vu}$ – that remains fixed during a computation. Traditionally one views the "synaptic weights" $w_{vu}$ as parameters that collectively contain the "program" of a computation in a neural circuit. Hence the fact that in biological neurons the values $w_{vu}(t)$ of these parameters are highly dynamic has drastic consequences: It is no larger clear which parameters *store* the program of a neural computation. Obviously that makes it even less clear how *learning algorithms* (i.e., algorithms that adjust the parameters that store the "program" of a neural computation) operate in biological neural systems.

These issues lead us to another significant structural difference between computations in biological neural systems and those computations that are usually studied in theoretical computer science. The inputs and outputs of computations in biological neural systems are typically vectors of *time-series*, rather than vectors of *numbers*. The processing of the $t$-th input $\underline{x}(t)$ may depend on the preceding inputs $\underline{x}(1), \ldots, \underline{x}(t-1)$. In that respect biological neural computation corresponds to computations carried out by finite state transducers (Mealy- or Moore-machines) or *filters* (as considered in signal processing and systems theory), rather than to computation carried out by feedforward circuits or Turing machines. For analyzing the computational power of neural circuits for computations on time series it is essential for the model that in reality the thresholds $\vartheta_v(t)$ and weights $w_{vu}(t)$ are functions of time that may depend on the preceding history of the computation. The following example [Maass and Zador, 1998b] illustrates that even on the abstract level of threshold circuits one can observe an increase in computational power resulting from history-dependent weights in connection with a sequential input presentation

Consider a threshold gate with $n$ inputs, that receives an input $\boldsymbol{xy}$ of $2n$ bits in two subsequent batches $\boldsymbol{x}$ and $\boldsymbol{y}$ of $n$ bits each. We assume that the $n$ weights $w_1, \dots, w_n$ of this gate are initially set to 1, and that the threshold of the gate is set to 1. We adopt the following very simple rule for changing these weights between the presentations of the two parts $\boldsymbol{x}$ and $\boldsymbol{y}$ of the input: the value of $w_i$ is changed to 0 during the presentation of the second part $\boldsymbol{y}$ of the input if the i-th component $x_i$ of the first input part $\boldsymbol{x}$ was non-zero. If we consider the output bit of this threshold gate after the presentation of the second part $\boldsymbol{y}$ of the input as the output of the whole computation, this threshold gate with "dynamic synapses" computes the boolean function $F_n : \{0,1\}^{2n} \to \{0,1\}$ defined by $F_n(\boldsymbol{x},\boldsymbol{y}) = 1 \iff \exists i \in \{1,\dots,n\}(y_i = 1 \text{ and } x_i = 0)$. One might associate this function $F_n$ with some novelty detection task since it detects whether an input bit has changed from 0 to 1 in the two input batches $\boldsymbol{x}$ and $\boldsymbol{y}$.

It turns out that this function cannot be computed by a small circuit consisting of threshold gates of the usual type, that receives all $2n$ input bits $\boldsymbol{xy}$ as one batch. In fact, one can prove that any feedforward circuit consisting of the usual type of "static" threshold gates, which may have arbitrary weights, thresholds and connectivity, needs to consist of at least $\frac{n}{\log(n+1)}$ gates in order to compute $F_n$. This lower bound can easily be derived from the lower bound from [Maass, 1997] for another boolean function $CD_n(\boldsymbol{x},\boldsymbol{y})$ from $\{0,1\}^{2n}$ into $\{0,1\}$ which gives output 1 if and only if $x_i + y_i \geq 2$ for some $i \in \{1,\dots,n\}$ , since $CD_n(\boldsymbol{x},\boldsymbol{y}) = F_n(\boldsymbol{1}-\boldsymbol{x},\boldsymbol{y})$.

The article [Maass and Zador, 1998a] surveys empirical data on the temporal dynamics of synapses and theoretical investigations of their possible computational role.

One fundamental open problem for the theory of neural computation is the question how information is encoded in spike trains. There appears to be no unique answer. Rather, the *neural code* seems to vary from system to system, and even the same neural system may apply different neural codes for different computational tasks (see [Rieke et al., 1997] and [Recce, 1998]). It is shown in Fig. 6 that typical interspike intervals are relatively long in comparison with the total computation time of some neural systems. Furthermore interspike intervals tend to be highly irregular. Hence it is rather difficult for a biological neuron to find out within the given computation time the current firing rates of presynaptic neurons (especially in view of the relatively short time duration of most EPSP's). Therefore the popular assumption that information is primarily communicated between neurons through their firing rates is somewhat dubious.

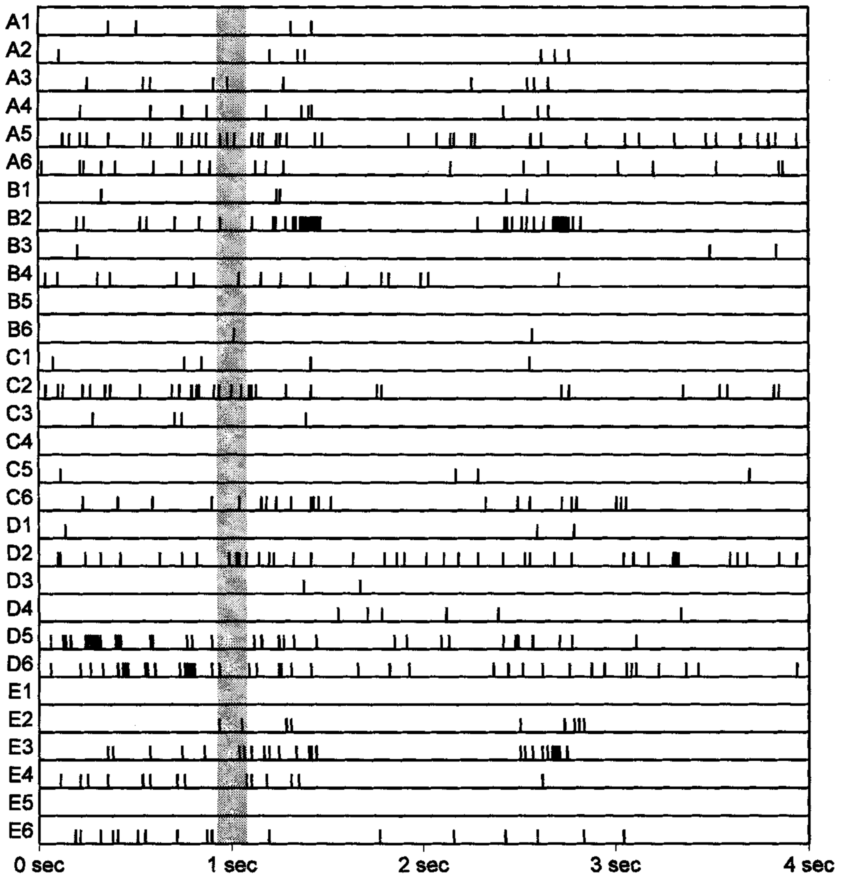


**Fig. 6.** Simultaneous recordings (over 4 seconds) of the firing times of 30 neurons from monkey striate cortex by Krüger and Aiple [Krüger and Aiple, 1988]. Each firing is denoted by a vertical bar, with a separate row for each neuron. For comparison we have shaded an interval of 150 msec. This time span is known to suffice for the completion of some complex multilayer cortical computations.

One completely different neural code that has been suggested as being relevant is the socalled *correlation code* (see [Recce, 1998] and [Maass, 1998b] for references). This code appears to be of particular interest from the point of view of *computer science logic*, since it hypothesizes that information is not transmitted between neurons in the form of numbers, but in the form of *second order objects*: (graded) *relations* or *sets*: Neurons whose firing time is statistically correlated during a neural computation communicate to other neurons the fact that they currently belong to the same set. A neuron can detect whether a critical number of presynaptic neurons belong to the same *set* because it can detect coincident firing times (as in our preceding discussion of the function $ED_n$, but now possibly with a higher firing threshold that requires coincident firing times of more than two presynaptic neurons).

## 3 On the Role of Space in Neural Computation

In the preceding section we had discussed examples for the different role that *time* plays in biological neural computation. In this section we will briefly discuss the role of another resource for neural computation: space, or more precisely the geometrical layout of neural circuits. The number of neurons to which a biological neuron has synaptic connections is by several orders of magnitude smaller than the number of neurons that participate in a typical computation. Hence "edges" or "wires" are a sparse resource in biological neural computation. This is not reflected in currently existing investigations of computational complexity issues for threshold circuits in theoretical computer science. There one typically wants to minimize the number of layers and the number of gates, with no charge for wires between adjacent layers. In addition in biological neural systems a large number of synapses connect neurons that are located quite close to each other. Obviously such architecture tends to keep *total wire length* small. This resource has also not yet been investigated in the context of threshold circuits in theoretical computer science. Its investigation appears to be of interest also from the point of view of related electronic hardware (for example cellular neural networks, see [Roska, 1997]). We refer to [Maass, 1998a] for some results in this direction.

[Valiant, 1994] addresses another important problem regarding the role of the spatial layout of neural circuits: Which local algorithms enable the computational units to carry out reliable information processing in a circuit whose layout is given by a random graph – hence not by a precise top-down design?

## 4 Outlook

Concurrently with the investigation of neural computation in living organisms one has started to design electronic hardware that captures particular aspects of the special role that time and space play in biological neural computation (see [Mead, 1989] and [Murray, 1998]). Examples are artificial retinas [Mead, 1989] schemes for low power analog communication between chips via pulses (address-event-representation, see [Douglas and Whatley, 1998,Mortara and Venier, 1998]) and programmable analog filters that employ pulses in a mix of analog and digital circuit techniques (see [Hamilton and Papathanasiou, 1998]) and cellular neural networks [Roska, 1997]. This approach is sometimes referred to as *Neuromorphic Engineering* (see [Smith, 1998] for the Proceedings of the first European Workshop on this topic). Obviously this area is still at a very early stage, and one might hope that theoretical computer science will play a role in its future development.

In principle theoretical computer science might be useful in modelling essential aspects of biological neural systems in a simplified mathematical framework, thereby providing a platform for extracting "portable" computational mechanisms and principles that can potentially be transported to novel *artificial* computing machinery. Unfortunately up to now, theoretically computer science has contributed very little in this direction (notable expections are for example [von Neumann, 1958] and [Valiant, 1994]). Perhaps one obstacle has been the difficulty for a non-expert to get an overview of the current state of the art in neurophysiology and neuromorphic engineering. This situation is now improving since a number of books with quite accessible surveys of most relevant topics have recently appeared (or will appear shortly): [Churchland and Sejnowski, 1992,Arbib, 1995] [Rieke et al., 1997,Ballard, 1997,Maass and Bishop, 1998,Koch, 1998].

Details to some of the specific models and results discussed in this article are available from http://www.cis.tu-graz.ac.at/igi/maass/ .

## References

[Arbib, 1995] Arbib, M. A., editor (1995). *The Handbook of Brain Theory and Neural Networks.* MIT Press, Cambridge.

[Ballard, 1997] Ballard, D. H. (1997). *An Introduction to Natural Computation.* MIT-Press.

[Churchland and Sejnowski, 1992] Churchland, P. and Sejnowski, T. (1992). *The Computational Brain.* MIT Press, Cambridge.

[DasGupta and Schnitger, 1996] DasGupta, B. and Schnitger, G. (1996). Analog versus discrete neural networks. *Neural Computation,* 8(4):805–818.

[Dobrunz and Stevens, 1997] Dobrunz, L. and Stevens, C. (1997). Heterogenous release probabilities in hippocampal neurons. *Neuron,* 18:995–1008.

[Douglas and Whatley, 1998] Douglas, R. J. and Whatley, A. M. (1998). A pulse-coded communications infrastructure for neuromorphic systems. In Maass, W. and Bishop, C., editors, *Pulsed Neural Networks.* MIT-Press, Cambridge.

[Gerstner, 1998] Gerstner, W. (1998). Spiking neurons. In Maass, W. and Bishop, C., editors, *Pulsed Neural Networks.* MIT-Press, Cambridge.

[Gerstner and van Hemmen, 1994] Gerstner, W. and van Hemmen, L. (1994). How to describe neuronal activity: spikes, rates or assemblies? In *Advances in Neural Information Processing Systems,* volume 6, pages 463–470. Morgan Kaufmann.

[Hamilton and Papathanasiou, 1998] Hamilton, A. and Papathanasiou, K. (1998). Preprocessing for pulsed VLSI systems. In Maass, W. and Bishop, C., editors, *Pulsed Neural Networks.* MIT-Press, Cambridge.

[Hopcroft and Ullman, 1979] Hopcroft, J. E. and Ullman, J. D. (1979). *Introduction to automata theory, languages and computation.* Addison-Wesley, Reading Mas.

[Kleene, 1956] Kleene, S. C. (1956). Representation of events in nerve nets and finite automata. In *Automata Studies,* pages 3–42. Princeton University Press, Princeton N.J.

[Koch, 1998] Koch, C. (1998). *Biophysics of Computation: Information Processing in Single Neurons.* Oxford University Press, Oxford.

[Krüger and Aiple, 1988] Krüger, J. and Aiple, F. (1988). Multielectrode investigation of monkey stritate cortex: Spike train correlations in the infragranular layers. *Neurophysiology,* 60:798–828.

[Maass, 1997] Maass, W. (1997). Networks of spiking neurons: The third generation of neural network models. *Neural Networks*, 10:1659–1671.

[Maass, 1998a] Maass, W. (1998a). A model for universal analog computation in neural circuits with local connectivity. in preparation.

[Maass, 1998b] Maass, W. (1998b). A simple model for neural computation with firing rates and firing correlations. submitted for publication.

[Maass and Bishop, 1998] Maass, W. and Bishop, C., editors (1998). *Pulsed Neural Networks*. MIT-Press, Cambridge.

[Maass et al., 1991] Maass, W., Schnitger, G., and Sontag, E. (1991). On the computational power of sigmoid versus boolean threshold circuits. In *Proc. of the 32nd Annual IEEE Symposium on Foundations of Computer Science 1991*, pages 767–776.

[Maass and Zador, 1998a] Maass, W. and Zador, A. (1998a). Computing and learning with dynamic synapses. In Maass, W. and Bishop, C., editors, *Pulsed Neural Networks*. MIT-Press, Cambridge.

[Maass and Zador, 1998b] Maass, W. and Zador, A. M. (1998b). Dynamic stochastic synapses as computational units. In *Advances in Neural Processing Systems*, volume 10. MIT Press, Cambridge (to appear).

[McCulloch and Pitts, 1943] McCulloch, W. S. and Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *Bull. Math. Biophysics*, 5:115–133.

[Mead, 1989] Mead, C. (1989). *Analog VLSI and Neural Systems*. Addison-Wesley (Reading).

[Mortara and Venier, 1998] Mortara, A. and Venier, P. (1998). Analog VLSI pulsed networks for perceptive processing. In Maass, W. and Bishop, C., editors, *Pulsed Neural Networks*. MIT-Press, Cambridge.

[Murray, 1998] Murray, A. F. (1998). Pulse-based computation in VLSI neural networks. In Maass, W. and Bishop, C., editors, *Pulsed Neural Networks*. MIT-Press, Cambridge.

[Recce, 1998] Recce, M. (1998). Encoding information in neuronal activity. In Maass, W. and Bishop, C., editors, *Pulsed Neural Networks*. MIT-Press, Cambridge.

[Rieke et al., 1997] Rieke, F., Warland, D., Bialek, W., and de Ruyter van Steveninck, R. (1997). *SPIKES: Explorıng the Neural Code*. MIT-Press, Cambridge.

[Roska, 1997] Roska, T. (1997). Implementation of cnn computing technology. In W. Gerstner, A. Germond, M. H. and Nicoud, J.-D., editors, *Proc. of ICANN 1997*, pages 1151–1155. Springer Verlag, Berlin.

[Smith, 1998] Smith, L. (1998). *Neuromorphic Systems: Engineering Silicon from Neurobiology*. World Scientific.

[Sontag, 1997] Sontag, E. D. (1997). Shattering all sets of 'k' points in "general position" requires (k-1)/2 parameters. *Neural Computation*, 9(2):337–348.

[Valiant, 1994] Valiant, L. G. (1994). *Circuits of the Mind*. Oxford University Press, Oxford.

[von Neumann, 1958] von Neumann, J. (1958). *The Computer and the Brain*. Yale University Press, New Haven.