A Lagrangian Formulation of Neural Networks II: Clocked Objective Functions and Applications

Willard L. Miranker[1] and Eric Mjolsness[2]

[1] Department of Computer Science
and Neuroengineering and Neuroscience Center
Yale University New Haven CT 06520
and
Research Staff Member, Emeritus,
IBM T.J. Watsou Research Center, Yorktown Heights, NY 10598

[2]Jet l'repulsion Laboratory
California Institute of Technology
4800 Oak Grove Drive
Pasadena CA 9110W3099

Abstract

In Part I of this work we showed how a tradeoff between measures of neural net cost (of operation) and functionality (efficacity) could be used to derive the dynamics of the net, and in particular, optimize thereby a class of objective functions. Here we extend that methodology; a Lagrangian formulation and greedy variation to treat more ramified problems. We introduce a notion of clocking and a class of clocked objective functions to do this. A kind of switching dynamics occurs which is suitable for many applications. This notational clocking calculus makes for time-scaled computational techniques employing a "focus of attention" (similar to saccading, foveation, and covert attention in biological vision). Experiments dealing with applications are referenced.

1 INTRODUCTION

In Part I of this work [MM] (to be referred to hereafter, simply as Part 1) we introduced a Lagrangian formulation of the dynamics of a class of relaxation-based analog neural networks. These Lagrangians incorporate a trade-off between measures of the operational cost and the functionality (efficacy) of neural networks employed to optimize a given objective function E. Because of the need for nonconservative or dissipative dynamics, our Lagrangians are to be varied in a nonstandard way using the so-called "greedy variation". This results in dissipative analog circuit dynamics described by first-order systems of differential equations. Within a class of candidate I,agrangians, we proved the near-optimality (under a suitable meta-objective function) of a particular Lagrangian corresponding to the Hopfield/Grossberg analog circuit dynamics. However, for efficiency, elaborate computations may require more complex dynamics specified at a coarser scale of temporal resolution, and this is a theme of the present work.

Here (in Part II) we proceed to consider more elaborate Lagrangians which are capable of specifying non autonomous dynamics. For example the dynamics may depend on which subset of the problem variables is currently being optimized, as well as the subset next to be optimized. This kind of '[switching" dynamics occurs in many applications and requires a more general formulation of the Lagrangian which we develop [n section 2 we introduce a time-varying or switched version of the problem objective function E, *called* a "clocked objective function". We relate it to our Lagrangian formulation of dynamics, producing so-called cyclic Lagrangians We develop suitable notation for expressing a number of existing optimization methods in terms of such clocked objectives. Reference is made to a number of experiments, application and computation, which utilize this clocking calculus. In section 3 we show how to specialize these ideas to the case of a computational "focus of attention" (similar to saccading, foveation, and covert attention in biological vision) which iteratively and opportunistically selects a subset of the problem's variables for optimization, and optimizes them. We show how to develop Lagrangians on different problem scales, Greedy variation then leads to the dynamics relevant to each scale. The working of the clocking or switching in the problem development and its solution is worked out, In section 4 we derive and relate various particular focus of attention mechanisms, inducting several which have been tested in previously reported computer experiments. These include priority queue attention, multiscale attention, jumping and rolling windows of attention, spreading activation (of neurons) and orthogonal windows. Section 5 provides a summary.

2 DYNAMICS WITH SWITCHING: VIRTUAL NETWORKS

Suppose we have hardware capable of switching different sets of neuron output values from a static (backup) memory into an active neural network, where they can be updated. With such hardware it is possible to implement a computation which would require a much larger neural network if every neuron were to be actively updated at all times. This situation would be analogous to the use of virtual memory in a conventional computer, in which one has a limited amount of physical memory (Random Access Memory) augmented by a much larger amount of secondary storage (magnetic disks). Equally, it is analogous to the distinction between the small cache memory associated with a central processing unit, and the larger physical memory (RAM). In either part of the memory hierarchy a relatively small and fast memory, in concert with a relatively large and slow memory, simulates a large fast memory (with occasional slowdowns due to page faults or cache misses). In like manner, we seek to design a switching mechanism for obtaining the computational power of a large neural network with a small neural network plus a large, slow and relatively inexpensive memory. Furthermore, in some cases it will prove possible to dispense with the slow memory entirely.

Such a system would be useful not only for making space-time tradeoffs in situations where only a limited amount of spatial resources (neurons and connections) are available, but also for formulating search algorithms (such as binary search) which can't be fully parallelized due to their unpredictable total resource requirements.

What kind of cost and functionality terms would model this situation? This is a hierarchical design problem. At a coarse time scale, we have just two kinds of decisions to make: what the active set of neurons (the *focus of* attention) is to be at any given time, and what their new values are to be after some period of active dynamics. (In the memory hierarchy analogy, one would like to decide which part of slow memory to bring into fast memory as some computation progresses.) At a fine time scale we must repeatedly make circuit-implementable state changes which move towards answering these two coarse-scale questions.

A strong constraint on the system is that, under reasonable cost metrics such as network space-time volume, no savings will be realized unless the focus-of-attention decision has converged to a definite answer by the time a switch of attention is to be made (i.e. by the time that decision is to be implemented); partial answers as to which neurons should be active would just force all the candidate neurons to be active. (An attentive neural network which unhappily violates this constraint is described in [Mjo87].) Of course one can contemplate dynamics in which by way of example a linear combination of neuron values is made active, but such a system should be designed by introducing new variables for the linear combinations and a discrete switching circuit which still, to be physically cost-effective, makes definite decisions about the active set of neurons.

So, our problem is to find both coarse-scale and fine-scale cost and functionality terms to model a focus-of-attention mechanism which switches many stored neuron values into and out of a small active network, where the neural values are updated. We will not consider all aspects of this problem. Rather wc shall show how the Lagrangian formalisms provide a tractable framework for

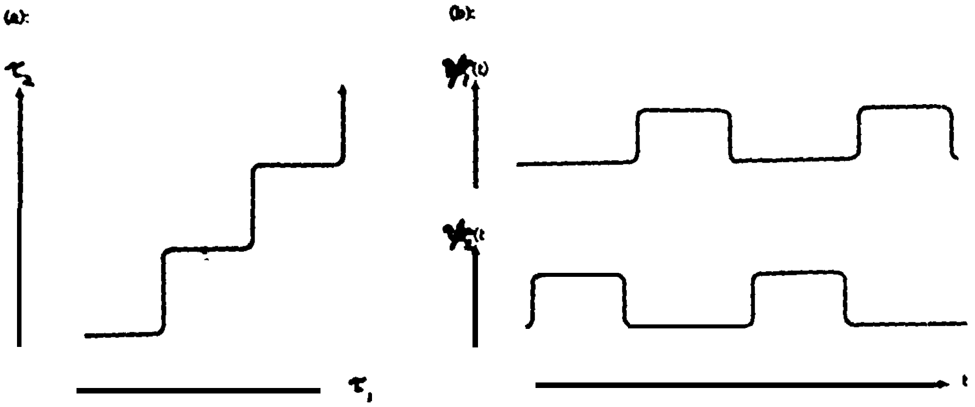

Figure 1: Two time variables τ_1 and τ_2 may increase during nonoverlapping intervals of an underlying physical time variable, t. For example $\tau_1 = \int dt\psi_1$ (t) and $\tau_2 = \int dt\psi_2(t)$ where $\psi_1 = d\tau_1/dt$ and $\psi_2 = d\tau_2/dt$ are nonoverlapping clock signals. (a)' The parametric curve $(\tau_1$ (t), $\tau_2(t))$. **(b)** The functions $\psi_1(t)$ and $\psi_2(t)$.

our approach. This is illustrated through derivation of a few plausible Lagrangians in the form of clocked objective functions. Related work appears in [Coo89, Mjo87, MM91, BSB+ 91].

2.1 Cyclic Lagrangians

In discussing coarse-scale cost and functionality terms, the idea of a repeating cycle of a fixed set of dissimilar coarse-scale decisions will be fundamental. This idea is analogous to a "loop" in programming, or to the use of cyclic clock signals to control an electronic circuit. The idea may be expressed in terms of Lagrangians in several different contexts which we will discuss here. In all cases we will find a simple formulation in terms of a "clocked objective function" [MGM91]: a version of the AE functionality term of the Lagrangian in which the structure of E is regarded as time-dependent according to a temporal cycle corresponding to the fixed cycle of coarse-scale decisions. The possibility of formulating a cyclic Lagrangian in terms of a clocked objective function was introduced in section 2.1.1 of Part 1, equations (17) and (18).

As an example, consider a line-minimization algorithm for local optimization. Repeatedly, one calculates the gradient at a current location x, does a one-dimensional minimization of the objective function along the gradient direction, and updates x. During the cycle it is necessary to store an old configuration x$^{\text{old}}$ for use in updating x, and to reset to zero the scalar parameter s which measures displacement in the gradient direction.

To express these ideas we recall the clocked objective function notation [MGM91]: Suppose that we have a small set of objective functions { E_α } which are to be partially relaxed (i.e. partially optimized) in a cycle. We define one nonoverlapping clock function, $\psi_\alpha(t)$ = O or 1 (with $\sum_\alpha \psi_\alpha(t) \leq 1$), for each phase $\alpha = 1,2,\ldots, A$ of the cycle. 'l'he clocked objective function is written as

$$E_{\text{clocked}}[\mathbf{x}, t] = \sum_\alpha \psi_\alpha(t) E_\alpha[\mathcal{X}_\alpha^{\text{free}} | \mathcal{X}_\alpha^{\text{fixed}}], \tag{1}$$

where $\mathcal{X}_\alpha^{\text{free}}$ and $\mathcal{X}_\alpha^{\text{fixed}}$ arc subsets of variables from the entire set $\{x_i\}$. During phase α (i.e. when $\psi_\alpha(t)$ = 1), $E_{\text{clocked}} = E_\alpha[\mathcal{X}_\alpha^{\text{free}}|\mathcal{X}_\alpha^{\text{fixed}}]$ is to be extremized with respect to all variables in $\mathcal{X}_\alpha^{\text{free}}$, while all variables in $\mathcal{X}_\alpha^{\text{fixed}}$ are to be held fixed or clamped. Figure 1 shows one interpretation of the nonoverlapping clock functions ψ_α (t).

For example, a simple clocked objective function for line minimization would be

$$\begin{aligned} E_{\text{clocked}} \quad = \quad & \psi_1(t)\frac{1}{2}\left(||\mathbf{x}^{\text{old}} - \mathbf{x}||^2 + s^2\right)[\mathbf{x}^{\text{old}}, s|\mathbf{x}] && \text{(initialize } \mathbf{x}^{\text{old}} \text{ and s)} \\ & +\psi_2(t)\left(E[\mathbf{x} + s\nabla E[\mathbf{x}^{\text{old}}]]\right)[s|\mathbf{x}, \mathbf{x}^{\text{old}}] && \text{(line minimization)} \qquad (2) \\ & +\psi_3(t)\frac{1}{2}\left(||\mathbf{x} - \mathbf{x}^{\text{old}} - \text{Sv } E[\mathbf{x}^{\text{old}}]||^2\right)[\mathbf{x}|s, \mathbf{x}^{\text{old}}] && \text{(update } \mathbf{x}\text{).} \end{aligned}$$

Since the $\alpha = 1$ and $\alpha =$ 3 phases are especially easy quadratic optimizations, one could arrange that these terms are relaxed almost to zero during the clock phase interval appropriate to each. Then equation (2) is a continuous-time refinement of the coarsf-scale Lagrangian's decision cycle, which partially relaxes Ein a gradient direction and then resets the variables for the next partial relaxation. At the end of phase 2 in each cycle, the clocked objective function takes the value of E at the new point. So the clocked objective function may be interpreted as a refinement of the functionality term of the coarse-scale decision-cycle Lagrangian. This interpretation also requires that the appropriate variables be held fixed at the correct times; this may be achieved with a cost term C_α which strongly penalizes any change in the clamped variables for the relevant clock phase.

Many variations on equation (2) are possible; the clocked objective could interpolate an extra cycle for the calculation of the gradient vector, and the x used to calculate the gradient could be taken as the $\mathbf{u} = g^{-1}$ **(v)** rather than v variables for E, and soon.

2.1.1 R elation of E_{clocked} to E

So far we have only argued that clocked objective functions provide an interesting special case of the distributed Lagrangian equation (5) in Part I; we have not shown how they can be related to the static objective function E or the dynamic objective function equation (4) in Part I with functionality term $F = E_{\text{final}} - E_{\text{initial}}$. Here we will discuss three different classes of clocked objective functions, each of which can be used to make some progress on minimizing E in every complete clock cycle so that AE $\leq$ Ofor each cycle even though the functionality term is not simply equal to ΔE. In this section we refer to such a clocked objective as "valid" for objective E.

Transient Terms For the first claw of clocked objective functions, of which the line minimization objective (2) is an example, E_{clocked} is valid if one of its components E_β is equal to E itself, perhaps with restricted arguments, and if the other components can each be expected to relax to near-zero values within their own phase of the cycle. These other components will be referred to as *transient terms* of a clocked objective function, since they approach zero quickly. Then progress is definitely made during phase β, and at least no harm is done (i.e. no increase in E_β is suffered) in the other phases a. Generally these other phases are used to ensure the suitability of the arguments of $E_\beta = E$.

Subspace Terms In the second class **of** valid clocked objective **functions,** E_α **is** equal to E during all clock phases, except that it is a function of different sets of variables (or more generally, is a function on different submanifolds) during different clock phases. We will refer to this type of term a-s a *subspace term* of a clocked objective function. There can be no significant calculation required to decide what subset of variables E depends on during each phase (otherwise we'd need a further phase to make that calculation). One simple arrangement is to partition all variables into a few blocks $\mathcal{X}_\alpha$, with the variables in one block allowed to change during each phase of the clock. Then equation (1) simplifies (since every E_α is just E) to

$$E_{\text{clocked}}[\mathbf{x}, t] = \sum_{a} \psi_\alpha(t) E[\mathcal{X}_\alpha^{\text{free}} | \mathcal{X}_\alpha^{\text{fixed}}]. \tag{3}$$

This permits concise expression of blockwise coordinate descent algorithms.

It 1S perhaps surprising that $E_{\text{clocked}}[\mathbf{x}, t]$ is not numerically equal to $E[\mathbf{x}(t)]$ 'or all t in this case,owing to the nonoverlapped clock factors $\psi_\alpha(t)$ ● [0, 1] whose sum varies between O (between phases) and 1 (during a clock phase). As we will see in the next section (2.1.2), this is necessary so that the continuous-time Lagrangian will force all variables to completely stop changing between clock phases, as they should.

We note that the second class of clocked objective functions can be used for the discrete parallelization scheme mentioned at the beginning of section 2.3.1 of Part I. l'here we postulated a partition of the network variables into a small number of "colored" blocks, with neighboring variables in the network having different colors. (Colors are in a correspondence with phases.) Such a partition can be used to ensure noninterference of discrete-time parallel update dynamics. Clearly equation (3) is the correct clocked objective for this situation, and F would just be $\Delta E_{\text{clocked}}$.

Control Terms For the third class of valid clocked objective functions which perform optimization, onc constituent objective E_β is agai n taken to be E with restricted arguments (a subspace term, as in the first and second classes), ancl the other phases either relax to nearly zero (being composed of transient terms as in the first class) or serve to determine the choice of active arguments for phase β without directly changing any of the original variables x. Since this last type of objective is a sum of terms that only involve variables that control membership in the active set of arguments for β, its constituent terms will be referred to as *control terms* in a clocked objective function. Clocked objective functions with control terms are the class of objective functions most relevant to the attention mechanisms of section 4. In that section we will have occasion to usc clocked objectives containing a variety of subspace terms, transient terms and control terms.

2.1.2 Lagrangians for Clocked Objective Functions

We have seen in equation (17) of Part I how clocked objective functions may arise from coarse-scale Lagrangians, in which the the functionality term takes on a cyclic sequence of different forms. Our purpose now is to relate such clocked objective functions (as in (l)) to continuous-time Lagrangians.

The essential feature of a single term $E_\alpha[\mathcal{X}_\alpha^{\prime\text{free}}|\mathcal{X}_\alpha^{\prime\text{fixed}}]$ in a clocked objective function E is that it depends only on some of the variables, the rest being held constant at their earlier values. This gives a property expressible in terms of derivatives:

$$\frac{\partial E_\alpha}{\partial x_i}[\mathcal{X}_\alpha^{\prime\text{free}}|\mathcal{X}_\alpha^{\prime\text{fixed}}] = \chi_{\alpha i}\frac{\partial E_\alpha}{\partial x_i}, \tag{4}$$

where $\chi_{\alpha i} \in$ {O, 1 } is a *constant* which indicates the presence (χ = 1) or absence (χ = O) of x_i in $\mathcal{X}_\alpha^{\prime\text{free}}$. (For fixed $\alpha, \chi_{\alpha i}$ is an indicatrix for $\mathcal{X}_\alpha^{\prime\text{free}}$). Consequently, $E_\alpha[\mathcal{X}_\alpha^{\prime\text{free}}|\mathcal{X}_\alpha^{\prime\text{fixed}}]$ is a low-dimensional *slice* (restriction) of the higher-dimensional function $E_\alpha[\mathcal{X}'_\alpha]$, evaluated at values of the fixed parameters which are dictated by the state vector x at the beginning of the α-th phase.

¿From equations (3) and (4) we may now calculate $\partial E_{\text{clocked}}/\partial x_i$:

$$\frac{\partial E_{\text{clocked}}}{\partial x_i} = \sum_\alpha \psi_\alpha(t)\chi_{\alpha i}\frac{\partial E_\alpha}{\partial x_i}, \tag{5}$$

which is nonzero at any given time *t* only if x_i is in the free set of variables at that time.

We can take the final continuous-time Lagrangian to be

$$L = \sum_i \left(K[\dot{x}_i, x_i] + \frac{\partial E_{\text{clocked}}}{\partial x_i}\dot{x}_i \right), \tag{6}$$

where K is a cost-of-motion term (see section 1.1 of Part I). To see that this is consistent with the desired pattern of fixed variables as a function of time, we examine the resultant dynamics. As in equation (30) of Part 1, varying $\dot{x}_i$ and using $\sum_\alpha \psi_\alpha \chi_{\alpha i} \in$ {O, 1 }, and defining $\tilde{K}[w, x]$ as the inverse of $K[\dot{x}, x]_{,\dot{x}}$ with respect to its first argument, the equations of motion are

$$\dot{x}_i = \tilde{K}[-\sum_\alpha \psi_\alpha(t)\chi_{\alpha i}\frac{\partial E_\alpha}{\partial x_i}, x_i] = \sum_\alpha \psi_\alpha(t)\chi_{\alpha i}\tilde{K}[-\frac{\partial E_\alpha}{\partial x_i}, x_i]. \tag{7}$$

Here we have used equation (5) and $\tilde{K}[0, x]$ = O to simplify the equations of motion. The factor of $\psi_\alpha(t)\chi_{\alpha i}$ ensures that the correct variables are clamped at the correct times.

Equation (6) is appealing because it has the same form as the continuous-time Lagrangian for unclocked objective functions, equation (22) of Part 1. This is the desired relationship between continuous-time Lagrangians and clocked objectives. Because of equation (6) it will often suffice to give the clocked objective alone, omitting the Lagrangian, in order to specify a network's dynamics.

2.1.3 Notation for Clocked Objective Functions

Equations such as (4) can be expressed in a more convenient notation for algebraic calculations (by human or computer). From an algebraic point of view, (4) may be regarded as the x_i derivative of E_α, a version of E_α in which all fixed variables $x_j \in \mathcal{X}_\alpha^{\prime\text{fixed}}$ are simply replaced by *clamped variables* (or "fixed variables") $\hat{x}_j$ for which

$$\frac{\partial \hat{x}_j}{\partial x_i} = () \text{ despite the fart that } \frac{\partial x_j}{\partial x_i} = \delta_{ij}. \tag{8}$$

'l'he actual value of x_j is updated to the current value of x_j only at the (otherwise irrelevant) time intervals between the nonoverlapped clock phases, when $\sum_\alpha \psi_\alpha(t) = 0$. Equation (4) follows directly from this interpretation of $E_\alpha[\mathcal{X}_\alpha^{\text{free}}|\mathcal{X}_\alpha^{\text{fi xed}}]$ in terms of $\hat{E}_\alpha$.

In fact, we can design notation for the substitution that relates $\hat{E}$ to E. Define

$$x\{\chi\} = \chi x + (1-\chi)\dot{x} \quad \text{s o} \quad \mathbf{x}\{\chi_\alpha\} = \chi_\alpha \cdot \mathbf{x} + (1-\chi_\alpha)\cdot\dot{\mathbf{x}} \tag{9}$$

where χ is a binary (zero- or one-valued) scalar (or can easily be rounded to zero or one) and $\boldsymbol{\chi}_\alpha$ is just the constant array $\chi_{\alpha i}$ which specifics with its zero-valued entries which variables are clamped in each phase α. With this notation, $\hat{E}_\alpha$ is just $E_\alpha[\mathbf{x}\{\chi_\alpha\}]$, i.e.

$$E_\alpha[\mathcal{X}_\alpha^{\text{free}}|\mathcal{X}_\alpha^{\text{fixed}}] = E_\alpha[\mathbf{x}\{\chi_\alpha\}]. \tag{1o}$$

We will use $E_\alpha[\mathbf{x}\{\chi_\alpha\}]$ as the preferred notation. Furthermore $\boldsymbol{\chi}$ need not be a constant; it can be replaced with any vector-valued expression $\pi(\xi)$ involving variables ξ. Equation (9) would still define

$$\mathbf{x}\Big\{\pi(\xi)\Big\} = \Theta(\pi(\bar{\xi}) - \mathbf{1/2}) \cdot \mathbf{x} + \Theta(1/2 - \pi(\bar{\xi})) \;\; \mathbf{x}, \tag{11}$$

where

$$\mathbf{c)(r)} = \begin{cases} 1 & \text{if } x > 0 \\ 0 & \text{otherwise.} \end{cases} \tag{12}$$

Θ is defined componentwise on vectors. 'l'he purpose of the Θ function in (11) is to round $\pi(\xi)$ to zero or one, with a boundary at 1/2. Note that, in agreement with equation (9) in which χ is a constant, $\boldsymbol{\chi}$ is clamped in equation (1 1). That is because x's focus of attention cannot shift during the phase in which x is being relaxed without incurring excessive and uncontrolled switching costs.

As a further notational refinement, we may drop the explicit $\psi(t)$ functions from our notation by defining a *clocked* SUITI,

$$\bigoplus_\alpha E_\alpha \equiv \sum_\alpha \psi_\alpha(t) E_\alpha \tag{13}$$

which may be written out term-by-term as

$$E_1 \oplus E_2 \oplus \ldots E_A. \tag{14}$$

(The "$\oplus$" symbol is evocative both of a rolling "+" sign, and of an analog clock face.) Of course the periodic functions $\psi_\alpha(t)$ still have to be specified before the clocked sum is a well-defined quantity. The clocked sum is neither commutative nor associative, but we may take it to associate over the ordinary sum:

$$\sum_a \bigoplus_\alpha E_{a\alpha} \equiv \bigoplus_\alpha \sum_a E_{a\alpha} \tag{15}$$

Moreover, parenthesized expressions such as $E_1 \oplus (E_2 \oplus E_3)$ may be used to denote nested loops in which for example E_2 and E_3 are repeatedly relaxed in an inner loop, within one phase of an outer loop, and E_1 is relaxed once during the other phase of the outer loop. Again the timing would be controlled by external functions ψ_α (t), which must still be specified separately.

Note that the use of clocked objective functions is reminiscent of time ordering of operators in quantum physics. See also the so-called Feynman entangling calculus [MW66].

Perhaps the most important algebraic property of the clocked sum, for the purpose of formulating descent algorithms, is its commutation with partial differentiation:

$$\frac{\partial}{\partial x_i} \bigoplus_\alpha E_\alpha = \bigoplus_\alpha \frac{\partial}{\partial x_i} E_\alpha. \tag{16}$$

This follows directly from the definition of the clocked sum. The right hand side of equation (16) could be used as the time-dependent descent direction in a gradient-descent algorithm.

We may conventionally expect to find the $\oplus$ signs outside the + signs in a clocked objective function, and accordingly we assign $\oplus$ a lower grammatical precedence than + in otherwise ambiguous expressions. So by convention, $E_1 \oplus E_2 + E_3$ means $E_1 \oplus (E_2 + E_3)$.

With the addition of clamped variables $\bar{x}$, conditional variables $x\{\setminus\}$, and clocked sums $\bigoplus_\alpha E_\alpha$, we are able to concisely express a wide variety of cloc'keel objective functions. For example the line minimization objective (2) becomes

$$\begin{aligned} E_{\text{clocked}} \;=\;& s^2/2 + ||\mathbf{x}^{\text{old}} - \bar{\mathbf{x}}||^2/2 && \text{(initialize s, } \mathbf{x}^{\text{old}}) \\ & \oplus E[\bar{\mathbf{x}} + s\nabla E[\bar{\mathbf{x}}]] && \text{(line minimization)} \\ & \oplus ||\mathbf{x} - \bar{\mathbf{x}}^{\text{old}} - \bar{s}\nabla E[\bar{\mathbf{x}}]||^2/2 && \text{(update x)}, \end{aligned} \tag{17}$$

or what may be easier to implement as a circuit,

$$\begin{aligned} E_{\text{clocked}} \;=\;& s^2/2 + ||\mathbf{x}^{\text{old}} - \bar{\mathbf{x}}||^2/2 + ||\mathbf{w} - \nabla E[\bar{\mathbf{x}}]||^2/2 \\ & \quad \text{(initialize s, } \mathbf{x}^{\text{old}}\text{; find gradient w)} \\ & \oplus E[\bar{\mathbf{x}} + s\bar{\mathbf{w}}] && \text{(line minimization)} \\ & \oplus ||\mathbf{x} - \bar{\mathbf{x}}^{\text{old}} - \bar{s}\bar{\mathbf{w}}||^2/2 && \text{(update x).} \end{aligned} \tag{18}$$

Furthermore, clocked objective functions make new algebraic transformations possible. For example, equation (11) may be implemented for X-expressions (assuming only that we can implement it for O/l-valued variables) by introducing new variables η as follows:

$$E[\mathbf{x}\left\{\pi(\xi)\right\}] \to \sum_i \left[\eta_i(\pi_i(\bar{\xi}) - 1/2) + \phi_{0/1}(\eta_i)\right] \oplus E[\mathbf{x}\{\eta\}]. \tag{19}$$

Here $\phi_{0/1}$ is a two-sided barrier function which limits its argument to values between zero and one.

2.1.4 Experiments

The clocked objective function notation has been used to derive and express a number of experimentally validated relaxation-based neural networks, including networks for multiscale image segmentation [Tsi97], visual pose estimation [LM94], point matching [GLR+95], and invariant learning of point-set and graph models of visual objects [RGM96]. In these applications, the problem variables were divided into an exhaustive collection of subsets each of which received an exclusive clock phase. During the clock phase for any subset of the variables, all other variables were clamped and the optimization of the free subset was relatively easy or even analytically solvable. This situation is described by equation (3), which may be rewritten as a clocked objective function using *(13).* It occurs sufficiently often that we provide another notation for it:

$$E\langle \mathcal{X}_1^{\text{free}}, \mathcal{X}_2^{\text{free}}, \ldots \mathcal{X}_A^{\text{free}} \rangle_\oplus \equiv \bigoplus_\alpha E[\mathcal{X}_\alpha^{\text{free}} | \mathcal{X}_\alpha^{\text{fixed}}] = \sum_\alpha \psi_\alpha(t) E[\mathcal{X}_\alpha^{\text{free}} | \mathcal{X}_\alpha^{\text{fixed}}] \tag{20}$$

2.1.5 Clocked Circuits

Clocked objective functions can also be used to specify circuits at the analog level. The simplest way to do this is to assign to each clock phase the dynamics of an analog neural network in which some variables have been clamped. The clamping is under the control of the clock signals and/or other variables. That is the effect of equation (6), either under the original definition of clocked objective (5) or under the more powerful and convenient notation defined in equations (8), (11), and (1 3); it is also a basic idea behind the design of clocked pipelines of combinatorial logic in the data paths of simple CPU chips [MC80] where clamping is determined only by the clock signals. We take it as clear, then, that such clocked objective functions can be implemented as analog circuits provided that each phase can be so implemented, and provided that the objective includes $\bar{x}$ expressions (cf. (3)) but does not include $x\{y\}$ expressions (cf. (8)). For example, the line minimization clocked objective of equation (18) can be implemented this way, as can the multiscale optimization objective found in [MGM91].

In the next subsection wc show another such example: a clocked objective function which incorporates one or more general feed-forward neural networks inside a relaxation-based neural net, in a hybrid that may be of usc for combining relatively efficient learning algorithms (from feed-forward nets) wit h expressive power (from relaxation nets).

Later, we will discuss a set of applications that require the more powerful $x\{y\}$ notation, without speculating on the hidden circuit- level implementation of the switching mechanism. Thus the problem of eliminating $x\{y\}$ expressions in favor of x expressions remains for future work; it is related to the "neural network routing problem" discussed in [MG90], section 2.6. A further open problem is to replace global clock signals in a Lagrangian circuit formulation with a system of self-timed subcircuits in which the ψ_α control functions arc replaced by relatively local variables with independent dynamics. Solutions to analogous problems are implicit in the design of many distributed computer systems but not within a circuit-level Lagrangian framework. The $x\{y\}$ notation represents a substantial escalation in expressive power, and section 4 is devoted to some of its uses in designing computational attention mechanisms.

2.1.6 Feed-Forward Networks as Constraint Projection

A feed-forward network inside of a relaxation network can be regarded as a set of *constraints* on the relaxation network:

$$E_{\mathrm{FF/relax}}[\mathbf{x}] = E_{\mathrm{relax}}[\mathbf{x}] \text{ -t} \sum_{l\,\text{(layers)}} \mathrm{FF}[\mathbf{v}^l, T^l, \mathbf{v}^{l-}\ 1], \tag{21}$$

where FF is the functional dependency constraint of a layer's output neurons on its input neurons (here taken to be in the previous layer, though neurons in any previous layer may be inputs without causing problems for the following algorithm). Various methods are available for enforcing constraints within a neural network optimization [PB87,MG90,PS89], but the feed-forward network constraints have a natural ordering determined by the feed-forward pattern of connections. So in this special-case we can use a nonlinear *projection* method to enforce all the constraints. As mentioned in section 2.3.1 of Part I, related algorithms are discussed in [BT'89], for example, under the name of "gradient projection algorithms" or "scaled gradient projection algorithms".

Any incremental relaxation of the objective E_{relax} is followed by a series of projections which reestablish the feed-forward constraints, layer by layer (i.e. from earlier to later neurons in the feed-forward neuron order), in preparation for further relaxation. The clocked objective is

$$E_{\mathrm{FF-projection}}[\mathrm{x}]\ \mathbf{v}-\ \bigoplus_{l\,\text{(layers)}} \left\{\sum_{\mathrm{i}} \left\{ -v_i^l \sum_{\mathrm{j}} T_{ij}^l \bar{v}_j^{l-1} + \phi_i(v_i^l)\right\}\right\} \oplus E_{\mathrm{relax}}[\mathbf{x}]. \tag{22}$$

Note the especially simple form of each layer's objective:

$$\sum_i \left\{ -v_i^l \sum_j T_{ij}^l \bar{v}_j^{l-1} + \phi_i(v_i^l) \right\}. \tag{23}$$

Every neuron v_i^l in layer 1 is independent of every other in this objective, and the minimization of this objective is best achieved just by assigning values to all layer-l variables in parallel:

$$v_i^l = g_i\left(\sum_j T_{ij}^l v_j^{l-1}\right), \text{ where } g_i^{-1}(v) = \phi_i'(v). \tag{24}$$

This is the projection operation which immediately enforces the layer-l constraints. Later layers' projection operations do not disrupt earlier ones. So, at the beginning of the relaxation phase of every cycle, all the FF constraints will have been consistently satisfied.

3 FOCUS OF ATTENTION THEORY

A particular kind of clocked objective function formalizes the idea of a computational focus of attention. We will derive this clocked objective by first considering the functionality and cost terms of a coarse-scale greedy Lagrangian, and then developing the associated fine-scale greedy Lagrangian which specifies circuit-level dynamics,

3.1 Formulation of the Lagrangian at the Coarse Scale

Let χ be a set of discrete-valued variables which determine, directly or indirectly, which components of the neuron vector v are actively updated at any given time. In other words, χ determines a characteristic function $\pi_i(\chi)$ for the *focus of attention* or active set of v_i's. Thus

$$\pi_i(\chi) = \begin{cases} 1 \text{ if } v_i \text{ is active, i.e. in the focus of attention,} \\ 0 \quad \text{otherwise,} \end{cases} \tag{25}$$

with

$$\sum_i \pi_i(\chi) = n. \tag{26}$$

For example, we could have as many components of χ as of v and set $\pi_i(\chi) = \chi_i$. Or instead, we could introduce a partition of the components of v into blocks indexed by cc, with a 0/1 partition matrix B_{ia}; this is a form of aggregation applied to χ. (For now we will take n to be constant, though a variable n is sometimes useful) Then we would have one component of χ to switch each block of the partition, and $\pi_i(\chi) = \sum_a B_{ia}\chi_a$. (That is, a variable v_i isin the focus of attention if and only if its course-scale block a is in the focus of attention as determined by χ_a.) Usually $\pi_i(\chi)$ can be made linear in χ.

Regardless of the actual formula for $\pi_i(\chi)$, there will be some sparseness constraint on χ to ensure that only a small fraction of the neurons v are in the focus of attention at any one time. For example one might impose $\sum_i \pi_i(\chi) = n$, where n is the optimal size of the focus of attention (and n<< N = the total number of neurons vi). In the case of a partition matrix B with blocks of roughly equal size *b (so* $\sum_i B_{ia} \approx b$), the sparseness constraint would become $\sum_a \chi_a = n/b$.

Whatever the sparseness constraint on χ is, we will express it as a summand $\hat{\Phi}(\chi)$ in an objective function. Φ may be a penalty function, a barrier function, a Lagrange multiplier times the constraint, orsomecornbinatio nof these possibilities. Thus, we could choose froma variety of "k-winner" objective functions (k winners allowed in a competitive group). Assuming $\hat{\Phi}(\chi) = \Phi(e)$ where $e \equiv \sum_i \pi_i(\chi) - n$, we can enforce or at least favor satisfaction of the constraint e ≤ O with

$$\Phi(e) = \left\{ \begin{array}{rl} (c/2)e^2 & \text{(a penalty term), or} \\ \lambda e + \text{cue} - (c/2)\sigma^2 & \text{(Lagrange multiplier+ effective penalty [MG90],} \\ & \text{ with } \sigma \text{ an appropriate auxiliary variable), or} \\ c\int_{-\infty}^{e} g(x)dx & g \text{ monotonic and odd (a barrier term), or} \\ c\sigma - \int_{\min g(y)}^{\sigma} g^{(-1)}(x)dx, & \text{(effective barrier, linear in e),} \end{array} \right. \tag{27}$$

Stricter sparseness terms are also permissible, such as a sum of many k-winner terms on different sets of variables. And for a variable-size focus of attention, in which *71* is variable, one would also need a cost term for n.

All components of v will be assumed to take continuous values, even if they are ultimately supposed to converge to discrete values, Then the coarse-time-scale update rule implied by the action S will be of the form

$$\text{v}' = \mathbf{v}'(\mathbf{v},\chi). \tag{28}$$

For example

$$v_i' - v_i = \pi_i(\chi)G_i(\mathbf{v}), \tag{29}$$

where G is the cumulative effect determined by the fine-scale dynamics within an active-v clock phase. 'l'his update rule is to be derived from the greedy variation of a multiphase dynamical objective of the form

$$S = \sum_{\substack{\text{coarse scale} \\ \text{decision times } t, \\ L_{\text{cycle}} \geq 0}} L(t) = \sum_{t \mid \sum_\alpha L_\alpha \geq 0} \quad \sum_{\alpha \in \left\{ \begin{array}{c} \text{coarse-}\mathbf{v}, \\ \text{coarse-}\chi \end{array} \right\}} \psi_\alpha(t)\left[C_\alpha(t) + F_\alpha(t)\right], \tag{30}$$

where ψ_α is defined as in section 2.1. *The principle feature of equation* (30) *is that it has two clocIc phases, one* during which the v variables are free to move and the χ variables are clamped, and one in which the roles are reversed, During the active-χ phase the focus of attention is determined for the next active-v phase of the cycle.

Notice also that we have assumed a simple stopping criterion, $\sum_\alpha L_\alpha < 0$, which means that the coarse-scale dyuarnics continues only as long as its benefits (decrease in F) outweigh the costs (given by C'), and this decision is made at the end of each complete cycle. We must now find suitable functions $C_{\text{coarse-v}}$, $F_{\text{coarse-v}}$, $C_{\text{coarse-}\chi}$, and $F_{\text{coarse-}\chi}$.

3.2 Coarse-Scale F

To find the F terms, wc must decompose F_{total} = A E into a sum of roars(~-scale causal terms. We would like F_{coarse} to measure the improvement in E due to choosing a configuration χ and then updating v accordingly:

$$F(t) = F_{\text{coarse}-\mathbf{v}} + F_{\text{coarse}-\chi} = E[\mathbf{v}'(\mathbf{v},\chi)] - E[\mathbf{v}] + \Phi(\chi) \tag{31}$$

How can wc decompose this combined effect of v and χ into separate F terms for each coarse-scale decision? As previously mentioned, the difficulty is that the coarse-scale decision step which chooses values for χ cannot be made simultaneously with the decision of v values whose presence in the focus of attention is determined by that particular χ· One obvious way to accomplish this is to stage alternating coarse-scale decision phases, updating the two sets of variables, each based on the most recent value of the other:

$$\begin{aligned} X' &= \chi'(\chi,\mathbf{v}) \\ v' &= \mathbf{v}'(\mathbf{v},\chi'). \end{aligned} \tag{32}$$

Then, to decompose $F_{\mathbf{v}} + F_{\chi} = E[\mathbf{v}'] - E[\mathbf{v}]$, we may interpose some especially low cost *estimate* $\tilde{\mathbf{v}}$ of v' which could even be computed analytically given any candidate χ':

$$\begin{aligned} F_{\text{coarse}\,\chi'}[X' \text{ Iv}] &= E[\tilde{\mathbf{v}}(\mathbf{v},\chi')] - E[\mathbf{v}] + \Phi(\chi) \\ F_{\text{coarse }\mathbf{v}'}[\mathbf{v}'|\mathbf{v},\ X'] &= E[\mathbf{v}'|\chi'] - E[\tilde{\mathbf{v}}(\mathbf{v},\chi')|\mathbf{v},\chi']. \end{aligned} \tag{33}$$

The optima of these two expressions with respect to their free arguments then determine the functions in equation (32). Note that $F_{\text{coarse }\mathbf{v}'}$“[v] . ..] is independent oft, though the constant $E[\tilde{\mathbf{v}}(\mathbf{v},\chi')]$ is subtracted off to satisfy equation (31).

The F functions of equation (33) may be understood in the terminology of section 2.1.1 as a control term $(\Delta E)_{\text{est}}[\chi|\mathbf{v}] = E[\tilde{\mathbf{v}}(\mathbf{v},\chi')] - E[\mathbf{v}]$, a transient term $\Phi(\chi)$, and a subspace term $E[\mathbf{v}'|\chi']$. However, the subspace term is carefully normalized by subtracting the constant $E[\tilde{\mathbf{v}}(\mathbf{v},\chi')]$ in order to apportion credit for a given ΔE (equation (31)) between the χ and v phases of the dynamics. By equations (9) and (25), the subspace term $E[\mathbf{v}'|\chi']$ may be written as $E[\mathbf{v}'\{\pi(\chi)\}]$. So the objective function of equation (33) is equivalent to the clocked objective function

$$F_{\text{atten}} = (\Delta E)_{\text{est}}[\chi|\mathbf{v}] + \Phi(\chi) \oplus E[\mathbf{v}\{\pi(\chi)\}]. \tag{34}$$

It remains to specify the parameterization $\pi(\chi)$ of the focus of attention, the cost $\Phi(\chi)$ for a given focus of attention, and the estimation formula for the ΔE that would accrue from a given focus of attention $\pi(\chi)$·Each can be specified in a variety of ways. $\Phi(\chi)$ may be a k-winner constraint. Also the estimation formula $(\Delta E)_{\text{est}}$ may be meta-optimized to provide more accurate estimations as judged by their effect on the performance of the attention algorithm.

In summary, once we are given the function $\tilde{\mathbf{v}}(\mathbf{v},\chi')$ and the cost terms C_α, there is a Lagrangian (the sum of cost and functionality terms) and an associated optimization principle ($\delta_G L$ = O, as in section 2.2 of Part I) that determines the discrete-time dynamics of v and χ·The action is given by (30) for S and (33) for F.

3.2.1 **Criteria for Estimating the Effects of a Focus**

It remains to find suitable expressions or dynamics for $\tilde{\mathbf{v}}(\mathbf{v},\chi')$·These have the function of estimating the influence of alternative χ vectors (hence of different foci of attention) on v without actually performing the minimization of $F_{\text{coarse }\mathbf{v}'}$ [v] ỵ., Fhis problem is closely analogous to the recta-optimization problem posed in section 3.2 of Part I . ‘l’here we sought a functional form $K(\dot{\mathbf{v}}$, v) for the kinetic energy which resulted in the “optimal” dynamical system, where optimality was defined to depend on behavior in many different trials of the network. Likewise we must first define meta-optimality and then seek it, in the determination of a formula for $\tilde{\mathbf{v}}$ which will be used in many different trials of the network.

For any such functional $\tilde{\mathbf{v}}$, the required network computation must be very *inexpensive* compared to that of v’ for this reason: the cost of optimizing $F_{\text{coarse}\,\chi}$ is expected to be some large number of fine-scale iterations times the cost of finding $\tilde{\mathbf{v}}$ and is to be added to (and therefore balanced with) the cost of finding v’, As always we must weigh functionality against cost. What makes an estimator $\tilde{\mathbf{v}}(\mathbf{v}$, X') effective? For a single neural network trajectory, the obvious choice is to consider the $\tilde{\mathbf{v}}$ function effective to the extent that the resulting $\mathbf{v}(t)$ trajectory minimizes the action S in (30). After all, the Lagrangian already contains the correct balance of cost and benefit terms for judging the $\mathbf{v}$ dynamics, complete with a stopping criterion. The only remaining question is how to aggregate over many trials of the network which share the same formula for $\tilde{\mathbf{v}}$, i.e. many starting points, inputs, and possibly connection matrices. One could attempt a worst-case analysis as in the determinat ion of $K(\dot{\mathbf{v}}$, v), but we have not succeeded in that, Alternatively we consider an average case measure of action, averaged just over some probability distribution on starting points.

We have already proposed a recta-objective, (35), for this type of problem, Here we are averaging over starting points (and perhaps also over inputs h and connection matrices T):

$$\mathcal{M}[\tilde{\mathbf{v}}] = \langle S \rangle = \Big\langle \sum_{t|L(t)\geq 0} L(t) \Big\rangle_{\mathbf{v}(0)} \approx \frac{1}{P}\sum_{p=1}^{P} \sum_{t|L(t)\geq 0} L(t|\mathbf{v}_p(0)) = \mathcal{M}_p[\tilde{\mathbf{v}}], \tag{35}$$

where {VP(0)} are P starting points sampled from the same random distribution over initial conditions.

Generally, predictive accuracy in $\tilde{\mathbf{v}}$ is rewarded by this objective because of the term $E[\mathbf{v}'|\chi']$ in (33): x' is optimized for $E[\tilde{\mathbf{v}}(\mathbf{v}$, x)] and then used as a constraint in optimizing $E[\mathbf{v}'|\chi']$ with respect to v'.

The sampling procedure converts the infinite sum into a computable and optimizable quantity $\mathcal{M}_p$ at the expense of introducing a *learning* and *generalization* problem. As in theoretical approaches to learning [Vap82,BH89], we must ensure a sample size sufficient not only to approximate the infinite sum, but to continue to do so even after the sampled objective has been optimized (by tuning $\tilde{\mathbf{v}}$) to that particular sample (so that it is no longer a random sample of the infinite sum). In this way, a nontrivial predictive learning problem enters into the design of the switched neural network dynamics.

$\mathcal{M}_\infty$ may also be regarded as an average over all configurations along a trajectory, rather than just over the starting points, since every decision point along the trajectory contributes to the summed action. But to do this we must define a suitable probability distribution of configurations, and the distribution itself is a function of $\tilde{\mathbf{v}}$. This may limit its usefulness for simplifying the objective.

The connection between the optimization of $\tilde{\mathbf{v}}$ and a learning problem demonstrates one advantage of the derivation in section 3.2 of Part I of optimal kinetic energy terms from a worst-case meta-objective (equation (60) in Part I) rather than an average-case meta-objective (equation (35)): by this means analysis could be substituted for a large and (in general) recurring training computation.

3.2.2 Candidate $\tilde{\mathbf{v}}$ Estimators

We now present several possible forms for $\tilde{\mathbf{v}}(\mathbf{v},\chi)$, which are to be optimized and evaluated according to the criteria of the previous section. In the simplest form, $\tilde{\mathbf{v}}$ is to be computed by hypothesizing a small, constant time At between course scale decisions, during which $\tilde{\mathbf{v}}$ and therefore $E[\mathbf{v}]$ change according to Taylor's formula:

$$\tilde{v}_i = v_i + \Delta t \frac{dv_i}{d\,T_v} \tag{36}$$

(cf. (29)) where $T_v = \int \psi_{\mathbf{v}}(t)dt$ as in Figure 2.1.

We may also introduce, for each variable v_i, a hypothetical time axis τ_i which increases linearly with real time t when neuron vi is in the focus of attention (equivalently, when $\psi_{\mathbf{v}}(t) = 1$ and x allows vi to be actively updated, i.e. when $\psi_{\mathbf{v}}(t)\pi_i(\chi) = 1$) and stays constant otherwise. So

$$\tau_i(t) = \int dt\psi_{\mathbf{v}}(t)\pi_i(\chi), \text{ and } d\tau_i/d\tau_{\mathbf{v}} = \pi(\chi). \tag{37}$$

Then

$$\tilde{v}_i = v_i + \Delta t \frac{d\,v_i}{(171}\frac{d\tau_i}{d\,\tau_{\mathbf{v}}} \tag{38}$$

and

$$\begin{aligned} F_{\mathrm{coarse}-\chi'}[\chi|\mathbf{v}] &= E[\tilde{\mathbf{v}}(\mathbf{v},\chi')] - E[\mathbf{v}] + \Phi(\chi) \\ &\simeq (\Delta E)_{\mathrm{est}}[\chi|\mathbf{v}] \; + \; \Phi(\chi), \end{aligned} \tag{39}$$

where

$$(\Delta E)_{\mathrm{est}}[\chi|\mathbf{v}] = \Delta t \sum_i \left(\frac{\partial E}{\partial v_i} \frac{dv_i}{d\tau_i} \frac{d\tau_i}{d\tau_{\mathbf{v}}} \right) [\mathbf{v}(t_{\text{beginning of v phase}})|\chi]. \tag{40}$$

We introduce the useful quantity

$$E_{;i}\,[\mathrm{v}] \equiv \frac{\partial E}{\partial v_i} \frac{\mathrm{d}\, v_i}{d\tau_i}, \tag{41}$$

which for Hopfield/Grossberg dynamics becomes (cf. equation (30) of Part I)

$$E_{;i}[\mathbf{v}] = -g_i'(g_i^{-1}(v_i)) \left(\frac{\partial E}{\partial v_i} \right)^2 \equiv -g_i'(u_i)(E_{,i})^2, \tag{42}$$

first proposed as an objective function for driving a focus of attention in [Mjo87]. With these definitions, $(\Delta E)_{\mathrm{est}}$ becomes

$$(\Delta E)_{\mathrm{est}}[\chi|\mathbf{v}] = \Delta t \sum \pi_i(\chi) E_{;i}[\mathbf{v}] + \Phi(\chi), \tag{43}$$

and the associated $\tilde{\mathbf{v}}$ becomes, from (38),

$$\tilde{v}_i = v_i + \Delta t \pi_i(\chi) \dot{v}_i, \tag{44}$$

where now $\dot{v}_i \equiv dv_i/d\tau_i$ and $\dot{v}_i$ will take bounded values determined by the v-phase Lagrangian.

‘l’he optimizing parameter here (for the prediction objective M) *is* At, which will also enter into the coarse-scale cost term, since the cost of switching can be amortized only over the time At. Note that the variables χ_a are still discrete, and the cost of partly or completely minimizing $F_{\mathrm{coarse}\,\chi'}$ depends on the relation between $\pi_i(\chi)$ and χ_a to be specified.

Naturally the partial relaxation cost associated with $\pi_i(\chi)$ will only increase if we take the natural step of expanding $\tilde{\mathbf{v}}$ and E to second order in At. One good reason for doing this second-order expansion is that the optimal At will not be small if switching costs are sufficiently high, so a second order approximation may be more accurate. The second-order expansion proceeds as before:

$$\tilde{v}_i = v_i + \Delta t \pi_i(\chi) \dot{v}_i + \frac{\Delta t^2}{2} \pi_i(\chi) \ddot{v}_i \tag{45}$$

and

$$(\Delta E)_{\mathrm{est}}[\chi|\mathbf{v}] = \mathrm{At} \sum_{\mathrm{i}} \pi_i(\chi) E_{;i}[\mathbf{v}] + \frac{\Delta t^2}{2} \sum_{ij} \pi_i(\chi) \pi_j(\chi) E_{;ij}[\mathbf{v}] + \Phi(\chi), \tag{46}$$

where $E_{;i}$ has been defined in equation (43) and where $E_{;ij}[\mathbf{v}]$ is the quadratic form given by

$$\begin{aligned} E_{;ij}[\mathbf{v}] &\equiv \frac{\partial^2 E}{\partial \tau_i \tau_j} \\ &= \frac{\partial^2 E}{\partial v_i v_j} \frac{dv_i}{d\tau_i} \frac{dv_j}{d\tau_j} + \delta_{ij} \frac{\partial E}{\partial v_i} \frac{d^2 v_i}{d\tau_i^2} \\ &= E_{,ij} \dot{v}_i \dot{v}_j + \delta_{ij} E_{,i} \ddot{v}_i . \end{aligned} \tag{47}$$

For example under Hopfield/Grossberg dynamics, $E_{;ij}$ can be calculated as

$$\tau_H^2 E_{;ij}[\mathbf{v}] = g'(u_i)g'(u_j) E_{,i} E_{,j} E_{,ij} + \delta_{ij} g'(u_i) E_{,i} \left(\sum_k g'(u_k) E_{,ik} + \frac{g''(u_i)}{g'(u_i)} (E_{,i})^2 \right) \tag{48}$$

Because $\pi_i(\chi)^2 = \pi_i(\chi)$, any diagonal terms in the quadratic form $\sum_{ij} E_{;ij} \pi_i(\chi) \pi_j(\chi)$ (cf. (46)), in particular all those terms with δ_{ij} factors as in (48), can be absorbed into the π-linear part of $F_{\mathrm{coarse}\;\chi'}$ For example, in a quadratic neural net objective $E[v] = -(1/2) \sum_{ij} T_{ij} v_i v_j - \sum_i h_i v_i + \sum_i \phi(v_i)$, the coefficient of the quadratic form for χ could be taken as

$$E_{;ij}[\mathbf{v}] = -T_{ij} g'(u_i) g'(u_j) E_{,i} E_{,j} . \tag{49}$$

In this case the π-quadratic part of (46) becomes

$$(\Delta E)_{\text{estimate-quadratic}} = -\sum_{ij} \pi_i(\chi)\pi_j(\chi) g'(u_i) g'(u_j) E_{,i} E_{,j} T_{ij}, \tag{50}$$

and a corresponding connection matrix would have the opposite sign.

The essential ncw feature of objective (46) is that it involves quadratic infraction.s between the χ expressions corresponding to different neurons This introduces a nontrivial *scheduling* problem as part of the determination of the next focus of attention: separate neurons must not only be capable of making progress individually, but also those neurons likely to cooperate should be scheduled into the same focus of attention. This point will be elaborated in section 4.2.

3.2.3 Cost Terms

At the coarse scale, the cost of one cycle of computation is the cost of running the v network for time Δt_{v}, plus the cost of switching to the χ network, plus the cost of running the x network for a period At$_x$, plus the cost of switching back to the v network to start the next cycle.

These considerations may be expressed in the following cost terms for a coarse-scale clocked Lagrangian:

$$C_{\text{coarse-v}} \cdot C_{\text{switch}} + N_1(n)\Delta t_{\mathrm{v}} + \text{Clamp}(\Delta\chi, \{\Delta v_{i|\pi_i(\chi)=0}\}) \tag{51}$$

and

$$C_{\text{coarse-}\chi} = C_{\text{switch}} + N_2(n)\Delta t_\chi + \text{Clamp}(\Delta \mathrm{v}), \tag{52}$$

where "Clamp" is a penalty or barrier function which enforces the constancy of v or x as needed. Both of the cost terms here are constant if we regard $n, \Delta t_{\mathrm{v}}$, and At$_x$ as constant within a run, although in that case the constant values of the n and the At's probably should be chosen by a meta-optimization procedure using the same action, averaged over many trials, as the meta-objective.

Such a meta-optimization procedure could also be generalized to produce a simple rule, rather than a constant value, for each At and for 71; when such a rule produces the result Δt_{v} = At$_x$ = O, the computation stops, In that way the cornrnon problem of choosing a stopping criterion, as well as the more specialized problem of switching between optimization of v and of χ, fall naturally in the purview of meta-optimization. Of course such a rule could be given in the form of a Lagrangian for Δt_α, or equivalently for ψ_α, but we will not pursue this case here.

3.3 L at the Fine Scale

Since the v are analog variables, finding fine-scale C and F terms which act to minimize the coarse-scale ones is *now easy.* We proceed as in sections 2.1.2 of Part I and 3 of Part I, except that the Lagrangian functional of equation (22) in Part I is generalized to integrate each variable v_i according to its own internal time variable $\tau_i = \int \psi_{\mathrm{v}}(t)\pi_i(\chi)(t)dt$ as in Figure 1:

$$S^{(1)}_{\text{fine-v}} = \sum_i \int d\tau_i \left(K[\frac{dv_i}{d\tau_i}, v_i] + \frac{\partial E}{\partial v_i}\frac{dv_i}{d\tau_i} \right) \tag{53}$$

We may convert this into an integral of a single Lagrangian over a single time variable by using the formula for τ_i and the fact that $\psi_{\mathrm{v}}(t)$ and $\pi_i(\chi)(t)$ are each approximately zero or one almost all the time:

$$\begin{aligned}
S^{(1)}_{\text{fine-v}} &= \sum_i \int dt \frac{d\tau_i}{dt} \left(K[\frac{dv_i}{d\tau_i}, v_i] + \frac{\partial E}{\partial v_i}\frac{dv_i}{d\tau_i} \right) \\
&\approx \int dt \sum_i (\frac{d\tau_i}{dt})^2 \left(K[\frac{dv_i}{d\tau_i}, v_i] + \frac{\partial E}{\partial v_i}\frac{dv_i}{d\tau_i} \right) \\
&= \int dt \sum_i \psi_{\mathrm{v}}(t)\pi_i(\chi) \left(\frac{d\tau_i}{dt} K[\frac{dv_i}{d\tau_i}, v_i] + \frac{\partial E}{\partial v_i}\frac{dv_i}{d\tau_i}\frac{d\tau_i}{dt} \right) \\
&\approx \int dt \psi_{\mathrm{v}}(t) \sum_i \pi_i(\chi) \left(K[\frac{dv_i}{d\tau_i}\frac{d\tau_i}{dt}, v_i] + \frac{\partial E}{\partial v_i}\frac{dv_i}{dt} \right) \\
&\qquad \text{(using } K[0, v] = 0 \text{ and } d\tau_i/dt \approx 0 \text{ or } 1) \\
&= \int dt \psi_{\mathrm{v}}(t) \left(\sum_i \pi_i(\chi) K[\frac{dv_i}{dt}, v_i] + \sum_i \pi_i(\chi) \frac{\partial E}{\partial v_i}\frac{dv_i}{dt} \right).
\end{aligned} \tag{54}$$

But this is not quite the whole fine-scale Lagrangian for the active-v clock phase, because of the coarse cost terms of equation (51). The "Clamp" terms may be refined by adding appropriate cost-of-movement terms $K[\dot{x}, x]$ (where K is minimal at $\dot{x} = 0$) for each of the clamped variables:

$$S^{(2)}_{\text{fine-v}} = \int dt \psi_{\mathbf{v}}(t) \left(\sum_{\text{all non-v variables } x} K[\dot{x}, x] + \sum_{i} (1 - \pi_i(\chi)) K[\frac{dv_i}{dt}, v_i] \right) \qquad (55)$$

Adding $S^{(1)}$ and $S^{(2)}$ together, we get the part of the action that pertains to the active-v phase:

$$S_{\text{fine-}\mathbf{v}} = \int dt \psi_{\mathbf{v}}(t) \left(\sum_{\text{all variables } x} K[\dot{x}, x] + \sum_{i} \pi_i(\chi) \frac{\partial E}{\partial v_i} \frac{dv_i}{dt} \right) \qquad (56)$$

Comparing this action to the Lagrangian in equation (6), we see that the fine-scale dynamics is that of a clocked objective function governed by the focus of attention characteristic function $\pi_i(\chi)$.

Note that, as far as the Lagrangian is concerned, this refinement amounts to an algebraic substitution

$$\psi_{\mathbf{v}}(t) \; [C_{\mathbf{v}} + F[\mathbf{v}]] \; \rightarrow \; \psi_{\mathbf{v}}(t) \Big(\sum_{\text{all variables } x} K[\dot{x}, x] + \sum_{i} \pi_i(\chi) \frac{\partial F}{\partial v_i} \dot{v}_i \Big), \qquad (57)$$

which is justified since at the end of a coarse-scale step, F is just a constant starting value plus a coarse-scale change $\Delta_{\text{coarse}} F$, and the coarse-scale change is equal to a sum of fine-scale changes $\int dt \sum_i (\partial F / \partial v_i) \dot{v}_i$. Also, the K terms for the clamped variables (some v_i and all other variables) serve as penalty terms which, in the absence of other $\dot{x}$ terms, enforce $\dot{x}$ = O when $\psi_{\mathbf{v}}$ = 1 and thereby refine the "Clamp" terms of $C_{\mathbf{v}}$.

The hard part of refining a focus-of-attention Lagrangian is to find fine-scale C and F terms for the variable-z phase, because our coarse-scale terms assume discrete-valued $\boldsymbol{\chi}$ variables and the previous refinement techniques don't apply to that case. Indeed, a general, N variable, discrete-valued optimization may be the goal of the entire neural computation (at the coarsest time scale of all) so we surely can't assume that much capability at the tine time scale. On the other hand we have already accepted an approximation in $F_{\text{coarse-}\chi}$ on the grounds that it is not global convergence but merely the order of neural updates that is at stake. Additional simplifying approximations may also be acceptable if optimized through training and verified through testing.

Unless $F_{\text{coarse-}\chi}$ is linear in χ_a, (for example by being linear in At with $\pi_i(\chi)$ linear in x), this F is a nonlinear objective which will require many steps of analog relaxation dynamics, implying an uncertain time to convergence to a nearly discrete- valued $\boldsymbol{\chi}$. Since we only have an intermediate, fixed time At available for relaxation, some additional mechanism will be required to find discrete values for $\boldsymbol{\chi}$ after a possibly incomplete analog optimization of $F[\boldsymbol{\xi}]$, where ξ_a are continuous-valued versions of χ_a.

3.3.1 Two Phases of Switching

The computational **savings we** seek accrues through the actual switching from one active set of neurons to the next. For switching to occur, however, we need a "digital restoration phase" in which the $\boldsymbol{\chi}$ variables are restored to definite 0/1 values. This phase could be left implicit in our modeling, as part of the unspecified switching hardware, but then we would be unable to analyze possible failures of the mechanism such as too little time to converge to discrete values, or too many $\pi_i(\chi) = 1$. By contrast it is easy to leave purely digital circuit switching details unspecified, since accumulated experience makes it relatively easy to engineer such circuit mechanisms outside of our methodology. We will however explicitly model a third phase, in which analog variables $\hat{\chi}_a$ are restored to nearly discrete values χ_a, as close to O or 1 as any physical circuit quantity ever gets.

Then we will have a global cycle through one phase that relaxes the analog v variables and two phases that optimize the discrete $0/1 \boldsymbol{\chi}$ variables by first optimizing analog variables $\boldsymbol{\xi}$ and then restoring them to nearly discrete values $\hat{\boldsymbol{\chi}}$ which can substitute for actual discrete values $\boldsymbol{\chi}$ in any circuit implementation. Of course in a digital implementation medium (such as a general-purpose software environment) which exists as an abstraction of some analog physical system, one should instead move directly from $\boldsymbol{\xi}$ to $\boldsymbol{\chi}$.

With this addition the fine-scale Lagrangian becomes

$$L_{\text{fine}} = \sum_{\text{all variables } x} K[\dot{x}, x] + \sum_{\text{phases } \alpha} \psi_\alpha(t) \sum_{\alpha\text{-variables}, \mathbf{x}_\alpha} \left(\frac{\partial E_\alpha}{\partial \mathbf{x}_\alpha} \dot{\mathbf{x}}_\alpha \right) \tag{63}$$

which, as we showed with equation (6), is exactly the Lagrangian corresponding to a clocked objective function

$$E_{\text{fine}} = \sum_{a} \psi_\alpha(t) E_\alpha[\mathbf{x}_\alpha| \ .] = \bigoplus_{\alpha} E_\alpha[\mathbf{x}_\alpha, \bar{\mathbf{x}}_{\beta \neq \alpha}]. \tag{64}$$

More particularly (substituting from equations (57) and (60)) we get the clocked objective function for three-phase attentive dynamics:

$$\begin{aligned} E_{\text{3-phase}} = \ & \sum_i \pi_i[\xi] E_{;i}[\tilde{\mathbf{v}}] + \Phi\left(\sum_i \pi_i(\xi) - n\right) + \sum \phi_{0/1}(\xi_a) \ \text{(control terms)} \\ & \oplus - \sum_a \chi_a(\bar{\xi}_a - \theta) + \sum_a \phi_{0/1}(\chi_a) \qquad \text{(transient terms)} \\ & \oplus \ E[\mathbf{v}\{\pi(\chi)\}]. \qquad \text{(subspace term)} \end{aligned} \tag{65}$$

This clocked objective function for a focus of attention is a more elaborated version of equation (34). Note that, from equation (57), we have

$$\frac{\partial E[\mathbf{v}\{\pi(\chi)\}]}{\partial v_i} = \pi_i(\chi) \frac{\partial F}{\partial v_i} = \pi_i(\chi) \frac{\partial E}{\partial v_i}, \tag{66}$$

which is the essential feature of a clocked objective function, as derived in (5).

Various special case expressions for $\pi_i(\chi)$ will be explored in the next section. In the resulting networks we will often omit the digital resetting phase for a simple kWTA network, on the understanding that it should be restored as part of an analog circuit design.

4 APPLICATIONS TO COMPUTATIONAL ATTENTION

Here we present several possible applications of the forgoing computational attention mechanisms and notation. The first two (sections 4.1and 4.2) have been employed to good effect in [Tsi97] where substantial savings in computational cost are documented.The rest of the applications below may be considered as design examples.

4.1 Priority Queue Attention

The simplest possible expression for $\pi_i(\chi)$ is the identity function, in which each variable v_i has its own attention indicator χ_i:

$$\pi_i(\chi) = \chi_i \in \{O, 1\}, \text{ where } \sum_i \chi_i = n << N. \tag{67}$$

We have previously reported on this case in [MM91]. "i'he objective function for χ would be transformed into a clocked objective, as in (30) (again using the notation of section 2. 1.3):

$$E[\mathbf{v}] \rightarrow \left(\text{kWTA}(\chi, n) + \sum_{\text{i}} \chi_i E_{;i}[\tilde{\mathbf{v}}]\right) \oplus E[\mathbf{v}\{\chi\}]. \tag{68}$$

This representation of $\pi_i(\chi)$ looks expensive, since any savings obtained by leaving most v_i 's out of the focus of attention could be lost by updating all the χ_i variables each iteration. From equation (65) this update would also require computing $E_{;i}$ for every i, in the focus or not. But in fact $E_{;i}$ is unchanged unless v_i is in the focus of attention, or has a network neighbor in the focus; so for efficiency we can store this gradient information in a variable w_i which is only updated in those

These considerations can be formalized as a slight modification of the Lagrangian transformation point of view used in section 2.1 of Part I to derive a fine-scale Lagrangian for v. Now we are required to *partially* optimize an objective $F_{\text{coarse-}\chi}[\chi|\mathbf{v}]$, while guaranteeing the discreteness of χ. We will adapt the same three transformations as before. First we switch from discrete to constrained continuous optimization, accomplished in two successive phases using clocked objective function notation (2.1):

$$\psi_{\chi}(t)\left[C_{\chi} + F[\chi] + \Phi\left(\sum \pi_i(\chi) - n\right)\right] \rightarrow \psi_{\xi}(t)\left[\sum_{\text{all variables } x} K[\dot{x}, x] + F[\xi] + \Phi\left(\sum_i \pi_i(\xi) - n\right)\right] + \psi_{\hat{\chi}}(t)\left[\sum_{\text{all variables } x} K[\dot{x}, x] + \sum_a \hat{\chi}_a(\xi_a - \theta)\right], \tag{58}$$

where $\xi_i \in [0, 1]$, $\hat{\chi}_i \in [0, 1]$, θ is a threshold, and Φ is a sparseness term such as those of equation (27). Second, replace all constraints with penalty functions added to the objectives:

$$\begin{array}{c} F[\xi] \rightarrow E_{\chi\text{-opt}}[\xi] \equiv F[\xi] + \Phi(\sum_i \pi_i(\xi) - n) + \sum_a \phi(\xi_a), \\ \sum_a \hat{\chi}_a(\xi_a - \theta) + E_{\text{restore}}[\hat{\chi}] \equiv \sum_a \hat{\chi}_a(\xi_a - \theta) + \sum_a \phi(\hat{\chi}_a). \end{array} \tag{59}$$

Here the threshold θ is usually taken to be 1/2, but other values may be used if the analog $\hat{\chi}$ dynamics would thereby be sped up without losing accuracy. Also $\xi(i) = \pi_i(\xi)$, as in equation (25). Note that the objective $E_{\text{restore}}[\hat{\chi}]$ is especially well-behaved among those wc have considered, since the only way a large condition number or delay can arise is through the potential terms. The third transformation is to refine these coarse-scale objectives, and the usual volumetric cost terms, into fine-scale Lagrangians (cf. (57)):

$$\begin{array}{l} C_{\xi} + F[\xi] + \sum_a \phi(\xi_a) \rightarrow \sum_{\text{all variables } x} K[\dot{x}, x] + \nabla_{\xi}\left[F[\xi] + \Phi(\sum_i \pi_i(\xi) - n) + \sum_a \phi(\xi_a)\right] \cdot \dot{\xi} \\ C_{\hat{\chi}} + \sum_a \hat{\chi}_a(\xi_a - \theta) + \sum_a \phi(\hat{\chi}_a) \rightarrow \sum_{\text{all variables } x} K[\dot{x}, x] + \nabla_{\hat{\chi}}\left[\sum_a \hat{\chi}_a(\xi_a - \theta) + \sum_a \phi(\hat{\chi}_a)\right] \cdot \dot{\hat{\chi}} \end{array} \tag{60}$$

These two Lagrangians, along with the usual one for v, must be reassembled into a full three-phase Lagrangian by multiplying by nonoverlapping clocks $\psi_\alpha(t)$ and summing over α as in section 2.1; that is the only way to express the action as a sum over algorithm time t (some $\int \cdot dt$ or some $\sum_t \cdot$) rather than over the intra-phase time variables τ_α.

3.3.2 Complete Multiphase Dynamics

We now have a 3-phase dynamics: First, choose the focus of attention using analog χ variables so as to optimize their estimated effect on ΔE subject to resource limitations. Second, discretize x. Third, relax $E[\mathbf{v}|\chi]$, using the chosen focus of attention. The analog χ phase includes a global k-winner constraint for $\pi(\chi)$. We will assemble the previously derived fine-scale cost and functionality terms for this net into an action functional and an associated clocked objective function.

Adding the partial Lagrangians of equations (57) and (60), we get a preliminary Lagrangian

$$\hat{L}_{\text{fine}} = \sum_{\text{phases } \alpha} \psi_\alpha(t)\left\{\sum_{\text{all variables } x} K[\dot{x}, x] + \sum_{\text{o-variables, } \mathbf{x}_\alpha} \frac{\partial E \partial E_\alpha}{\partial \mathbf{x}_\alpha} \dot{\mathbf{x}}_\alpha\right\} \tag{61}$$

This Lagrangian presents a problem for times t between a-phases, when $\sum_\alpha \psi_\alpha(t) = 0$, because at such times no dynamics is specified. The desired dynamics between phases is that all variables should be clamped. This can be ensured by adding a penalty term for movement of any variable between phases, in the form of a kinetic energy term K:

$$L_{\text{extra}} = \left(1 - \sum_\alpha \psi_\alpha(t)\right) \sum_{\text{all variables } x} K[\dot{x}, x]. \tag{62}$$

Note that in physics, a Lagrangian consisting only of a kinetic energy term corresponds to a particle moving along a geodesic such as a straight line ($\ddot{x} = 0$), whereas here it corresponds to a variable clamped to a particular value.

circumstances. Also, the n-winner circuit can be implemented digitally as an incremental priority queue of w_i values. SO the clocked objective function becomes

$$\begin{aligned} E_{\text{queue}} &= \sum_i \left(w_i \left\{ \text{start} + \chi_i + \sum_j \text{Nbr}_{ij} \chi_j \right\} - E_{;i}[\bar{\mathbf{v}}] \right)^2 /2 && \text{(transient terms)} \\ &\oplus \ \text{start}^2/2 + \sum_i \chi_i \bar{w}_i + \Phi \left(\sum_i \chi_i - n \right) + \sum_i \phi_{0/1}(\chi_i) && \text{(transient + control terms)} \\ &\oplus \ E[\mathbf{v}\{\chi\}]. && \text{(subspace terms)} \end{aligned} \tag{69}$$

Here "start" is initialized to unity and almost immediately changed to zero (in the second phase of the first clock cycle), and Nbr_{ij} is a constant 0/1 matrix recording whether neurons v_i and v_j are adjacent in the network or not:

$$\text{N b r}_{ij} = \left\{ \begin{array}{ll} 0 & \text{if } \max_{\mathbf{v}} |\frac{\partial^2 E}{\partial v_i \partial v_j}[\mathbf{v}]| = \mathbf{0}, \\ & \text{i.e. if } \max_{\mathbf{v}}(|\phi_i''(v)|) + |T_{ij}| + \sum_k |T_{ijk}| = 0; \\ 1 & \text{otherwise.} \end{array} \right. \tag{70}$$

Note that at the end of the first phase, $w_i = -E_{;i}[\bar{\mathbf{v}}]$. That's because (a) in the first cycle, start_i = 1, and every variable w_i is initialized to $-E_{;i}$; and (b) in subsequent cycles, either w_i is again set to the proper value, or else χ_i = O and $\sum_j \text{Nbr}_{ij}\chi_j$ = O. In the latter case we know that w_i is unchanged from the previous cycle (since it is only changed in the first phase of any cycle), and also that $E_{;i}$ is unchanged from the previous cycle because it is unchanged by the dynamics of $E[\mathbf{v}\{\chi\}]$'s relaxation:

$$\begin{aligned} \left| \frac{d}{dt} E_{;i} \right| &= \left| \frac{d}{dt} \left(\frac{\partial E}{\partial v_i} \hat{K} \left(-\frac{\partial E}{\partial v_i}, v_i \right) \right) \right| \\ &= \left| \frac{d}{dt} \left(\frac{\partial E}{\partial v_i} \right) \left(\hat{K} - \frac{\partial E}{\partial v_i} \hat{K}_{,w} \right) + \frac{\partial E}{\partial v_i} \frac{dv_i}{dt} \hat{K}_{,v} \right| \\ &= \left| \left(\sum_j \frac{\partial^2 E}{\partial v_i \partial v_j} \frac{dv_j}{d\tau_j} \frac{d\tau_j}{dt} \right) \left(\hat{K} - \frac{\partial E}{\partial v_i} \hat{K}_{,w} \right) + \frac{dv_i}{d\tau_i} \frac{d\tau_i}{dt} \frac{\partial E}{\partial v_i} \hat{K}_{,v} \right| \\ &\leq \left(\sum_i \left| \frac{\partial^2 E}{\partial v_j \partial v_j} \right| \chi_j \left| \frac{dv_j}{d\tau_j} \right| \right) \left| \hat{K} - \frac{\partial E}{\partial v_i} \hat{K}_{,w} \right| + \chi_i \left| \frac{dv_i}{d\tau_i} \right| \left| \frac{\partial E}{\partial v_i} \hat{K}_{,v} \right| \\ &= \text{O (since } \chi_i + \sum_j \text{Nbr}_{ij} \chi_j = \text{ O).} \end{aligned} \tag{71}$$

So throughout the second phase when χ is being determined, $\bar{w}_i = -E_{;i}[\bar{\mathbf{v}}]$.

Also note that in accordance with the definition in equation (11), the expression that controls the clamping of a variable such as w_i is implicitly held constant and need not be explicitly clamped. Only the second phase of equation (69) above has O(N) variables, and it can be replaced by a priority queue data structure with update cost $O(n \log N + cN)$, where k depends on digital hardware details and where c <<1 reflects the cost of storing w_i in inactive memory for future use, presumed to be relatively small.

Equation (69) assumes that n is constant. This assumption may be removed, if the coarse-scale cost of each n is modeled explicitly as mentioned in section 3.2.3. To a first approximation we may take the cost of a focus of attention to be proportional to its size, n, and ignore the effects of various different border shapes on the actual cost (these effects would tend to favor a focus with a small-boundary.) But what should the proportionality factor be between cost and benefit (ΔE) terms? To get sensible results we'll answer this question in an ad hoc way, not (yet) derived from fundamental considerations. Suppose that the cost of updating a neuron is dominated, not by space and time costs, but by the A E benefit *foregone* by not saving those same space-time resources to update some other neuron in the following iteration. To estimate that cost, per focal neuron, we multiply the average available ΔE per neuron by a constant f which must be meta-optimized. Then we have the following functionality expression.

$$F[\chi, n] = \sum_i \chi_i E_{;i}[\mathbf{v}] + \text{kWTA}(\chi, n) + \frac{nf}{N} \sum_i |E_{;i}[\bar{\mathbf{v}}]| + \phi_{0/1}(n/N). \tag{72}$$

Optimizing this F may be achieved by (a) *sorting* i according to $E_{;i}$, for example incrementally with a priority queue data structure, and **(b)** turning on all χ_i for which $|E_{;i}|/(N^{-1}\sum_i |E_{;i}[\mathbf{v}]|) \geq f$. The focus of attention then consists of neurons whose single-neuron estimated contribution to ΔE is more than f times the average; it can range from none to all of the neurons. The potential function $\phi_{0/1}(n/N)$ can also be chosen so that the minimum focus size is one, rat her than none, of the neurons.

The focus of attention equation (67) provides maximal flexibility, since any subset of n out of N neurons in the network can be in the focus at one time. however, efficiency requires a hidden priority queue representation of $\pi(\chi)$, so that χ can be represented with only a marginal increment of space to encode this focus over that required by the n actual neurons in the focus at any time.

Generally such a representation is based on the binary addressing capabilities of a general-purpose computer. In fact the number of bits required in χ to specify such a focus is $\log_2 \left({N \atop 71} \right)$“ For large N and $n \ll$ N, this is approximately $n \log_2 N$ bits. \Ve can easily encode χ with this many bits, for example using the binary addresses of the n neurons in the unrestricted focus of attention. (Other efficient addressing schemes, such as Gray codes, would work too.) In radix (e.g. binary) notation for which $i = i_1 \ldots i_l$:

$$\chi(i) = \sum_a \prod_{b=1}^{1} \delta^K(\chi_{ab} - i_b) \tag{73}$$

(where χ_{ab} are binary-valued and δ^K is the Kronecker delta), or equivalently,

$$\chi(i) = \sum_a \prod_{b=1}^{l} \chi_{abi_b}, \quad \text{with} \quad \sum_{i_b} \chi_{abi_b} = 1. \tag{74}$$

If such a representation is substituted directly into a neural network objective function, rather than used in a hidden digital implementation of a stereotyped objective function such as the priority queue, then we get relatively intractable high-order objectives for χ (see [MG90] for an example of a sorting network using a similar high-order representation). Until this problem is solved by expressing some special- or general-purpose addressing and communication algorithms with simple clocked objective functions, we must appeal to non-neural switching circuits as necessary, taking care to estimate their costs. The clocked objective with brace notation $v\{\chi\}$ still specifies the *use* we make of such switching hardware, and would remain a useful notation even if we knew how to eliminate it in terms of clocked objectives without brace notation.

4.2 Multiscale Attention

The π_i (x) = χ_i representation of a focus of attention has the disadvantages of requiring a hidden, digital implementation (e.g. a priority queue) in order to be efficient, ancl of allowing foci without any coherent structure that might decrease the number of border neurons that are outside the focus but involved in the computational decision to move the focus, Both of these problems may be eliminated by restricting the focus of attention to a choice of one or several blocks of neurons, from a fixed partition of all the neurons into equal-sized blocks with low connectivity between the Mocks. An example of such a partition would be the division of the 2-d grid of the region-segmentation network (equation (19) in Part I) into $A \ll N$ uniform rectangular sub-grids. Any such partition can be represented by a sparse, non-square 0/1 matrix B for which $\sum_a B_{ia} = 1$. Given such a partition, only one focus indication neuron χ_a is needed for each block $a \in \{1, \ldots .$ A $\ll N\}$, rather than one per neuron index $i \in \{1, \ldots .$ N$\}$. In return for increased efficiency in the attention mechanism as compared with the previous case, one gives up flexibility in the shape of the focus of attention. Some of that flexibility can be reacquired by generalizing the partition scheme described below to many levels in a recursive algorithm.

For a single level of partitioning, in which neurons v_i are grouped into fixed blocks a which enter or leave the focus together according to indicator neurons χ_a,

$$\pi_i(\chi) = \sum_a B_{ia}\chi_a, \tag{75}$$

where B is the constant partition matrix.

We could just substitute this expression for χ_i (or $\pi_i(\chi)$) into equation (69) (or (65)), in which case the most active blocks of the partition B would be the focus of attention. Attention would be a very affordable computation, a k-wi[lncr-take-all (kWTA) network. One clocked objective is simply

$$\begin{aligned} E_{\text{block}} &= \sum_a \chi_a \sum_i B_{ia} E_{;i}[\bar{\mathbf{v}}] + \Phi\left(\sum_a \chi_a - n(A/N)\right) + \sum_a \phi_{0/1}(\chi_a) \\ \oplus \quad & E[\mathbf{v}\{\sum_a B_{ia}\chi_a\}], \end{aligned} \tag{76}$$

which can again be improved by storing $E_{;i}$ as w_i, to be recalculated only as necessary, and which can be further improved by storing $\omega_a = \sum_i B_{ia} w_i$.

But here we will push the method a little farther, by choosing the k blocks not only based on their internal gradients but also on their predicted synergies with each other. The synergy is predicted by using the second order expansion for E, equation (46), which may be affordable now that we have only A focus-control neurons:

$$E[\chi] = \Delta\tau_{\mathbf{v}}(\frac{dE}{d\tau_{\mathbf{v}}})[\mathbf{v}\{B\chi\}] + \frac{\Delta\tau_{\mathbf{v}}^2}{2}(\frac{d^2E}{d\tau_{\mathbf{v}}^2})[\mathbf{v}\{B\chi\}]. \tag{77}$$

Then the clocked objective analogous to (69) is

$$\begin{aligned} E_{\text{block}} &= \sum_i \left(w_i\left\{\text{start} + \sum_a B_{ia}\chi_a + \sum_b \text{Nbr}_{ib}\chi_b\right\} - E_{;i}[\bar{\mathbf{v}}]\right)^2/2 \\ & \quad + \sum_{ij} \left(w_{ij}\left\{\text{start} + \sum_a (B_{ia} + B_{ja})\chi_a + \sum_c (\text{Nbr}_{ic} + \text{Nbr}_{ic})\chi_c\right\} - E_{;ij}[\bar{\mathbf{v}}]\right)^2/2 \\ \oplus \quad & \sum_a \left(\omega_a\left\{\text{start} + \chi_a + \sum_b \text{Nbr}_{ab}\chi_b\right\} - \sum_i B_{ia}\bar{w}_i\right)^2/2 \\ & \quad + \sum_{ab} \left(\omega_{ab}\left\{\text{start} + \chi_a + \chi_b + \sum_c (\text{Nbr}_{ac} + \text{Nbr}_{bc})\chi_c\right\} - \sum_{ij} B_{ia}B_{jb}\bar{w}_{ij}\right)^2/2 \\ \oplus \quad & \text{start}^2/2 + \sum_a \xi_a\bar{\omega}_a - \frac{\Delta\tau_{\mathbf{v}}}{2}\sum_{ab}\xi_a\xi_b\bar{\omega}_{ab} + \Phi\left(\sum_a \xi_a - n(A/N)\right) + \sum_a \phi_{0/1}(\xi_a) \\ \oplus \quad & -\sum_a \eta_a\bar{\chi}_a + \Phi\left(\sum_a \eta_a - n(A/N)\right) + \sum_a \phi_{0/1}(\eta_a) \\ \oplus \quad & -\sum_a \chi_a(\bar{\eta}_a - \theta) + \sum_a \phi_{0/1}(\chi_a) \\ \oplus \quad & E[\mathbf{v}\{\sum_a B_{ia}\chi_a\}], \end{aligned} \tag{78}$$

where we have introduced constant sparse matrices

$$\text{Nbr}_{ib} = \Theta\Big(\sum_j B_{jb}\text{Nbr}_{ij} - 1/2\Big) \tag{79}$$

ant]

$$\text{Nbr}_{ab} = \Theta\Big(\sum_{ij} B_{ia}B_{jb}\text{Nbr}_{ij} - 1/2\Big). \tag{80}$$

In (78), as in its prototype (46), the main departure from other clocked objective functions for attention is the quadratic objective function for ξ which expresses a nontrivial scheduling problem: which k neuron-blocks should be active simultaneously in order to maximize the expected sum of single-block and block-pair contributions to $|\Delta E|$? This quadratic optimization could be as hard as the original optimization problem E, were it not for the fact that it involves far fewer variables ξ_a. So it is crucial to have a separate restoration phase for χ in case the ξ analog scheduling optimization does not finish within its clock phase. In fact if the convergence time of the scheduling network isn't known well enough, we may need two restoration phases: one which restores ξ to an analog kWTA solution η, and a subsequent phase to ensure discrete 0/1 values χ for the attention control variables. This conservative approach to restoration is incorporated in equation (78).

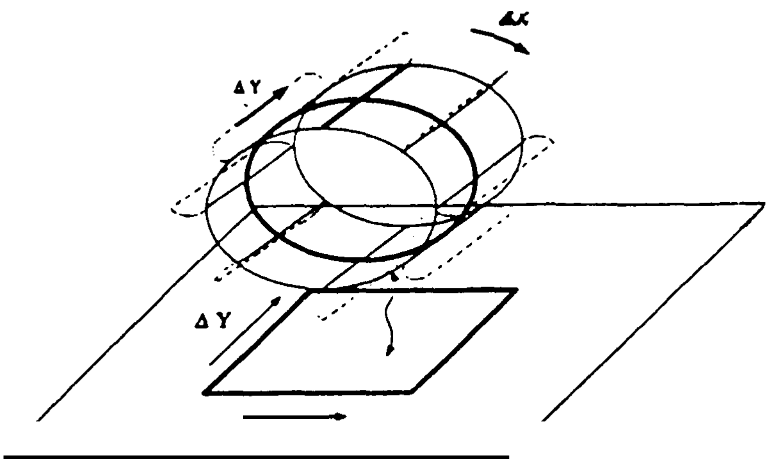

Figure 2: A rolling window of attention.

The scheduling network is a kind of auxiliary, coarse-scale network which controls attention at the level of blocks. Its connection matrix is surprisingly similar to part of a previously studied multiscale optimization neural network [M GM91], which also had an auxiliary coarse-scale network at the level of blocks of neurons. In that case the coarse-scale network was not for the purpose of control, but rather to accelerate the convergence of the much more expensive fine-scale network (which was simulated without any attention mechanism). In this regard the coarse-scale attention-control connection matrix ω_{ab} may be taken (as discussed in section 3.2.2) to be the negative of equation (50) after substituting (75) for $\pi_i(\chi)$; then it becomes identical to the coarse-scale acceleration connection matrix from [MGM91],

$$\hat{T}_{ab} = \sum_{ij} B_{ia} B_{jb} g'(u_i) g'(u_j) E_{,i} E_{,j} T_{ij}. \tag{81}$$

4.3 Jumping and Rolling Windows of Attention

The block-attentive neural network algorithm of equation (78) is equipped with a focus of attention that *jumps* from one block or combination of blocks to another in successive clock cycles. These jumps are rather expensive, since they involve storing the values of whole blocks of neurons which used to be in the focus of attention but no longer are, and retrieving from static memory the blocks of neurons which are newly promoted to the focus. A more gradual migration of neurons to and from the focus of attention is studied in this section, for networks with such a regular topology that the focus of attention can *roll* (i.e. move incrementally) from one region to another as well as jump.

A rolling focus of attention is one which moves incrementally, keeping most of its neurons assigned to the same implementation hardware. For example, consider a two-dimensional mesh of neurons with local connectivity, as occurs for example in the region-segmentation objective function (19) of Part I. A small piece of such a mesh could be implemented by a two-dimensional VLSI chip in which a fraction of the chip area is devoted to end-around connections, giving the circuit the topology of a torus, together with some form of secondary storage for the many neuron values which are clamped and stored off-chip. The torus can roll in any direction. The situation is illustrated in figure 2. Consider also the assignment of physical (chip-implemented) neurons to the much larger set of *virtual neurons* comprising the neural network. A rolling motion allows this assignment to remain unchanged everywhere except at the boundaries of the chip, or equivalently the boundaries of the focus of attention. This minimizes the need for off-chip communication and on-chip analog shifting circuitry everywhere in the chip, at the expense of requiring dynamic boundary circuitry (probably digital) throughout the chip. An alter-native would be to allow the focus of attention to "slide" around the neural net instead, in which case the dynamic boundary circuitry may be eliminated in favor of the analog shifting circuitry. Our clocked objective function can be implemented either way. For clarity we will discuss the rolling case.

‘1’0 describe the focus of attention mathematically, we just need $\boldsymbol{\pi}(\chi)$. We want to use a set of blocks of neurons as in section 4.2, so that they can jump under the control of $\{y_G\}$, except that the blocks also roll (or slide) around the mesh. Each block’s position can be characterized by its center. Block a has center $\mathbf{c}_a + \mathbf{x}_a$, in which $\mathbf{c}_a$ is a home position for block *a defined* by a fixed coarst’-scalc grid, and $\mathbf{x}_a$ is a dynamical displacement variable. The reason for including the home positions is to allow unused blocks to stay near their home positions, providing coverage of the alternative locations that the focus of attention can jump to. (This capability would not be necessary if blocks were only allowed to roll, but that would introduce spurious local minima into the attenti on mechanism, for example when a rolling window encounters its own or another window’s path.) Then $\boldsymbol{\pi}(\chi)$ is as in section 4.2, with $B_{ia} = b_i(\mathbf{c}_a + \mathbf{x}_a)$:

$$\pi_i(\chi) = \sum_a b_i(\mathbf{c}_a + \mathbf{x}_a)\chi_a. \tag{82}$$

We may scale our two-dimensional coordinates so that a block is a unit square, and we may assign addresses c_i in this coordinate system to each neuron i. We take $\mathbf{c}_a$ and $\mathbf{x}_a$ to be measured in this coordinate system also. Then the window boundary function b_i becomes

$$b_i(\mathbf{c}_a + \mathbf{x}_a) = b(\mathbf{c}_a + \mathbf{x}_a - c_i), \tag{83}$$

where

$$b(x) = \prod_{\alpha=1}^{\dim x} \Theta(1/2 - |x_\alpha|). \tag{84}$$

We will also have occasion to use a soft (differentiable) version of this window boundary function,

$$\tilde{b}_i(\mathbf{c}_a + \mathbf{x}_a) = \tilde{b}(\mathbf{c}_a + \mathbf{x}_a - \mathbf{c}_i), \tag{85}$$

where

$$\tilde{b}(\mathbf{x}) = \prod_{\alpha=1}^{\dim x} \tilde{\Theta}(1/2 - |x_\alpha|) \tag{86}$$

and

$$\tilde{\Theta}(x) = \begin{cases} 0, & x \leq -w/2 \\ x/w + 1/2, & -111/2 \leq x \leq w/2 \\ 1, & x \leq w/2 \end{cases} \tag{87}$$

Then a clocked objective function for the rolling and jumping window of attention is

$$\begin{aligned} E_{\mathrm{j\,\&\,r}}[\mathbf{v}, \boldsymbol{\xi}, \chi] = & \sum_i \Big(w_i\Big\{\text{start} + \sum_a b_i(\mathbf{c}_a + \mathbf{x}_a)\chi_a + \sum_b \text{Nbr}_{ib}\chi_b\Big\} - E_{;i}[\bar{\mathbf{v}}]\Big)^2/2 \\ & \text{(compute the gradients)} \\ \oplus \; & \sum_a \Big[-\eta_a\Big(\bar{\text{start}} + \bar{\chi}_a + \sum_b \text{Nbr}_{ab}\bar{\chi}_b - 1/2\Big) + \phi_{0/1}(\eta_a)\Big] \\ & \text{(clamp unaffected windows)} \\ \oplus \; & \sum_a \Big(\omega_a\{\eta_a\} - \sum_i b_i(\mathbf{c}_a + \mathbf{x}_a)\bar{w}_i\Big)^2/2 \\ & \text{(aggregate the gradients)} \\ \oplus \; & \text{start}^2/2 + \sum_a \Big[H(x_a\{\eta_a\}) + \sum_j \text{Nbr}_{aj}\tilde{b}_j(x_a\{\eta_a\})\bar{w}_j\Big] \\ & \text{(roll unclamped windows)} \\ \oplus \; & \sum_a \chi_a\Big[H(\bar{x}_a) + \sum_j b_j(\mathbf{c}_a + \bar{\mathbf{x}}_a)\bar{w}_j\Big] + \text{kWTA}(\chi, nA/N) \\ & \text{(select } k \text{ best windows \& jump there),} \\ \oplus \; & E[\mathbf{v}\Big\{\sum_a \chi_a b_i(\mathbf{c}_a + \mathbf{x}_a)\Big\}] \qquad \text{(descent within windows),} \end{aligned} \tag{88}$$

where as before

$$\text{kWTA}(\chi, k) = \Phi\left(\sum_a \chi_a - k\right) + \sum_a \phi_a(\chi_a). \tag{89}$$

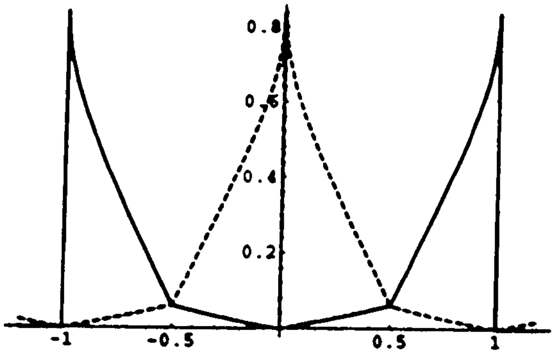

Figure 3: Spring function $\hat{H}(x) = \epsilon|x| + c_1\rho(|x| - 1/2) + \phi_{\pm 1}(\frac{x}{1-w/2})$, solid curve. First term restores $|x|$ to zero when block is out of the focus of attention. Second term favors hand-of^{f} to a neighboring block (neighboring block spring functions shown in dotted curves.) The third term is a barrier term, limiting the number of blocks that can be attracted to an attractive focal region of the network.

A crucial ingredient is the spring potential function H which allows a block o to move freely away from its home position until it is more than halfway into another block's territory, then to *hand* off the rolling window to a neighboring block 6 by turning off χ_a and turning on χ_b, and then to return to the borne position $\mathbf{x}_a$ = O to compute its expected ΔE and compete for another chance in the focus of attention.

A spring function that makes this possible is illustrated in figure 3. An explicit expression for H is

$$H(\mathbf{x}) = \sum_{\alpha=1}^{\dim X} \hat{H}(x_\alpha), \tag{90}$$

where

$$\hat{H}(x) = \epsilon|x| + c_1\rho(|x| - 1/2) + 4 * 1(\frac{x}{1 - w/2}), \tag{91}$$

and where

$$p(r) = \int_{-\infty}^{x} \Theta(x)dx = \{ \begin{matrix} o, & x \leq 0 \\ a, & 0 \leq x \end{matrix} \text{``} \tag{92}$$

4.4 Sparse Networks and Spreading Activation

'l'he attention mechanisms of the previous sections are designed to limit the number of active variables at any time, including both problem variables v and attention-control variables $\mathcal{X}\cdot$ However there is no attempt to limit the number of inactive variables whose values must still be stored and which therefore still occupy some hardware at all times. By imposing such a limit, we may be able to achieve far greater efficiency for optimization problems whose solutions are constrained to be sparse. What is required is that most of the variables outside the focus of attention should take on default values, such as zero, which need not be stored at all. The strategy is to enforce sparseness of v at every phase in every cycle, not just at the end of the computation. 'l'o achieve this we will allow mild expansions in the number of active neurons at some phases within a cycle, and enforce counterbalancing contractions in the number of active neurons at other phases in the cycle.

Suppose v is a set of N variables, constrained to be sparse in the sense that all but $n << N$ of them take (possibly identical) default values default(i) at any valid configuration. 'l'he default values may be zero or any number easily computed from the index r' alone, without the use of a large table of values (which would have to be stored). Let E(v) be an objective which includes penalty terms for sufficient sparseness constraints on at least some of the variables v, and which has the property that at any sparse configuration in which cn variables are unclamped in a focus of attention, all but n of the variables must approximate their default values at any local minimum. (Here c $\geq$ 1 is a constant,) Also suppose it is possible to initialize the network so that the focus of attention contains all non-default variables (of which there are $\leq n$) and also all neighbors of such variables (of which there arc $\leq cn$).

Then at the beginning of a relaxation phase for $E[\mathbf{v}\{\chi\}]$, all $\leq n$ non-default variables and all their $\leq cn$ neighbors are included in the focus of attention At the end of the relaxation phase, some new set of $\leq n$ variables have non-default values; the rest have near default values which can be reset to their default values without introducing much error, and which therefore do not need to be stored explicitly. In this way a limited front of activation relaxation, will propagate through the network of possible neurons which we shall refer to as *latent* neurons. The dynamics is reminiscent spreading activation or "marker propagation" algorithms in artificial intelligence [Fah79, '1ou86], and could perhaps be developed in that direction by using objective functions proposed in [MGA89]. Latent neurons are to be distinguished from the virtual neurons of previous sections (e.g. section 4.1), the latter requiring storage even when out of the focus of attention.

A suitable clocked objective function for such a spreading activation network, with many latent neurons, is

$$
\begin{aligned}
E_{\text{spread}} \;=\; & \sum_i \chi_i\{\chi_i\} + \sum_i \phi_{0/1}(\chi_i\{\chi_i\}) \\
& \oplus -\sum_i \chi_i\{s_i + \mathrm{Nbr}_{ij} s_j\} + \sum_i \phi_{0/1}(\chi_i\{s_i + \mathrm{Nbr}_{ij} s_j\}) \\
& \oplus E[\mathbf{v}\{\chi\};\ 71,\ \epsilon] \\
& \oplus -\!- \sum_i s_i\{\chi_i\}(\bar{v}_i - \mathrm{default}(i))^2/\epsilon^2 - 1) + \sum_i \phi_{0/1}(s_i\{\chi_i\}) \\
& \qquad + \gamma\Phi(\sum_i s_i\{\chi_i\} - n) + \gamma \sum_i s_i\{\chi_i\} E_{;i}[\bar{\mathbf{v}}; n] \\
& \oplus -\!-(1/2) \sum_i (v_i\{\chi_i(s_i - 1)\} - \mathrm{default}(i))^2 + \sum_i \phi_{0/1}(v_i\{\chi_i(s_i - 1)\}).
\end{aligned}
\tag{93}
$$

Here the first phase serves simply to find all nonzero X's and to set their values to zero. The second phase sets the focus of attention to include all non-default v_i's (for which $s_i = 1$) and their neighbors in the network topology. The third phase relaxes the network within the focus of attention, which we assume produces a new set of $\leq n$ variables v_i's which are not close to their default values. The fourth phase finds these variables and updates s_i to record them. Optionally, we can set $\gamma > 0$ to ensure what is already supposed to be guaranteed by E, that s = 1 for nonzero gradients and that $\sum_i s_i \leq n$. The fifth phase truncates near default values to exact default values, because neurons taking their default values do not need to be stored. (So in an implementation the fifth phase would not physically perform a truncation; it would simply de-allocate the hardware used to support the affected neurons.) The five phases together constitute one iteration of sparsity-preserving dynamics.

As an example of a suitable objective function E, we discuss a simple network for finding roots of a continuous function $f(x)$ of one variable $x \in [0, 1]$, by the bisection method. This network dynamically constructs a tree of at most n nonzero indicator neurons a_i, taken from an infinitely large tree of latent neurons. The network seeks large negative values of $f(x)f(x+\epsilon)$, and then bisects the interval $[x, x+\epsilon]$. Using multiple index notation $i \equiv i_1 i_2 i_l$, the search tree consists of all the latent 0/1 neurons ail.. i_l which take a value close to one if the search currently includes that node of the tree; also each node has a census neuron $m_{i_1 .. i_l} \in [0, n]$ which counts the number of neurons (including a's and m's) active at or below that node in the tree. These sets of variables would include the l = O versions, a and m without any indices, which are associated with the root of the search tree. The bisection search interval boundaries arc x_0 = O, x_1 = 1, x_{00} = O, $x_{01} = x_{10}$ = .5, x_{11} = 1, and in general, $x_{i_1 .. i_n b} = \sum_{p=1}^{l} i_p 2^{-p} + b 2^{-l}$.

Then a sparse objective function for this problem is

$$
\begin{aligned}
E_{\text{tree}} \;=\; & \sum_{l=1}^{\infty} \sum_{i_1 \ldots i_l = 0,1} a_{i_1 \ldots i_{l-1}} a_{i_1 \ldots i_l} g_{\pm}(C^l f(x_{i_1 \ldots i_l 0}) f(x_{i_1 \ldots i_l 1})) \\
& + (A/2) \sum_{l=0}^{\infty} \sum_{i_1 \ldots i_l = 0,1} (a_{i_1 \ldots i_l} + \hat{g}_{\pm 1}(m_{i_1 \ldots i_l}) + m_{i_1 \ldots i_l 0} + m_{i_1 \ldots i_l 1} - m_{i_1 \ldots i_l})^2 \\
& + (A/2)(m/n - 1)^2 + \sum_{l=0}^{\infty} \sum_{i_1 \ldots i_l = 0,1} \phi_{0/1}(a_{i_1 \ldots i_l}) + \sum_{l=0}^{\infty} \sum_{i_1 \ldots i_l = 0,1} \phi_{0/1}(m_{i_1 \ldots i_l}/n),
\end{aligned}
\tag{94}
$$

where $g_\pm$ is an odd monotonic function with slow asymptotic growth, e.g. logarithmic growth. The network could be initialized with al] a, m and s variables taking near-zero $(O(\epsilon) << 1)$ values, except at the root where s = 1. At initialization all the non-zero gradients of E (which arise from the k-winner-take-all terms) are concentrated at the root and its immediate children $i = 0$ and i = 1.

A noteworthy property of the objective (94) is that the sparseness constraints are not global, but rather distributed over the topology of the network in such a way that an actual neuron a is involved in every term of the sparseness constraint. This prevents many census variables m from being given non-zero values in an effort to find one non-zero a variable. Instead, only as many census variables will be activated as needed. The $a_i + \hat{g}_{\pm 1}(m_i)$ summand in the k-winner term serves to include both a_i and m_i in the count of activated variables: $\hat{g}_{\pm 1}(m)$ is a sigmoid with values $\approx$ 771 for $m <<$ 1, and $\approx$ 1 for $m \geq 1$. The $\hat{g}_{\pm 1}(m)$ expression could be replaced by another O/l-valued neuron whose sole connection is to m.

We speculate that it may be possible to give a similar treatment of the conventional objective functions for inexact graph matching, such as [11'1'86]

$$\begin{aligned} E_{\text{match}}[M] \;\; = \;\; & -\sum_{ijab} G_{ij} g_{ab} M_{ia} M_{jb} + (A/2) \sum_i (\sum_a M_{ia} - 1)^2 + (A/2) \sum_a (\sum_i M_{ia} - 1)^2 \\ & + B \sum_{ia} M_{ia}(1 ^- M_{ia}) + \sum_{ia} \phi_{0/1}(M_{ia}\ ). \end{aligned} \tag{95}$$

However it is again necessary to localize the winner-take-all constraints, for example by embedding them in spanning trees for both G and g, in which each variable M_{ia} enters into each WTA constraint at its own location in the spanning tree. An additional attraction of such a sparse graph-matching network is that the E-relaxation phase of the clocked objective could actually be a nested loop which performs deterministic annealing in order to avoid local minima, Since successive cycles would have different foci of attention, the successive annealing procedures would be different - the high-temperature part of an annealing relaxation would not erase the progress towards a solution encoded in the focus of attention. A related technique for accelerating the convergence of matching networks by exploiting their sparseness was used in [LM94, GLR$^+$95].

4.5 Orthogonal Windows

As suggested in [Mjo87], we can take advantage of the fact that some or all of the neurons in many hand-designed neural nets fall into natural cross-products, e.g. $v_i \equiv v_{i_1, i_2}$. An example is the graph-matching objective function of equation (95). In such cases we can greatly decrease the cost term by decomposing $\boldsymbol{\chi}$ and hope to retain functionality since it is only $\boldsymbol{\chi}$, not v, whose information content is thereby reduced. An obvious decomposition to try is:

$$\pi_i(\chi) = \chi_{i_1}^{(1)} \chi_{i_2}^{(2)}, \tag{96}$$

i.e.

$$\boldsymbol{\pi}(\boldsymbol{\chi}) = \boldsymbol{\chi}_{i_1}^{(1)} \otimes \boldsymbol{\chi}_{i_2}^{(2)}, \tag{97}$$

where

$$\sum_i \pi_i(\chi) = (\sum_{i_1=1}^{N_1} \chi_{i_1}^{(1)})(\sum_{i_2=1}^{N_2} \chi_{i_2}^{(2)}) \leq n. \tag{98}$$

'l'he last may be ensured by constraining

$$\sum_{i_1=1}^{N_b} \chi_{i_b}^{(b)} \leq n_b \;\; (b \in \{1,2\} \text{ and } n_1 n_2 \leq n). \tag{99}$$

For more than two terms in the cross product, all this generalizes to

$$\pi_i(\chi) = \prod_b \chi_{i_b}^{(b)}, \tag{100}$$

where

$$\sum_{i_b=1}^{N_b} \chi_{i_b}^{(b)} \leq n_b \text{ and } \sum_i \pi_i(\chi) = \prod_b n_b \leq n. \tag{101}$$

Following equation (68), we can use the clocked objective function

$$E_{\mathrm{orthog}} = E_\chi[\chi, \mathbf{v}] \oplus E[\mathbf{v}\{\chi^{(1)} \otimes \chi^{(2)}\}], \tag{102}$$

where

$$\begin{aligned} E_\chi[\chi, \bar{\mathbf{v}}] \equiv & \sum_i \chi^{(1)}_{i_1}\chi^{(2)}_{i_2} E_{;i}[\bar{\mathbf{v}}] + \Phi(\sum_{i_1} \chi^{(1)}_{i_1} - n_1) + \Phi(\sum_{i_2} \chi^{(2)}_{i_2} - n_2) \\ & + \sum_{i_1} \phi_{0/1}(\chi^{(1)}_{i_1}) + \sum_{i_2} \phi_{0/1}(\chi^{(2)}_{i_2}). \end{aligned} \tag{103}$$

A major problem with this scheme is that all the $E_{;i}[\bar{\mathbf{v}}]$ derivatives must be calculated, even though we want a small window of attention. A simple solution is to window the control variables χ also, and only calculate the few that are necessary. There may be only $O(N_1 + N_2)$ of those, rather than $O(N)$. One possibility is the disjoint union focus of attention $\pi(\chi) = (\eta^{(1)}, \eta^{(2)})$ for χ. We will apply transformation (68) twice: first to v, substituting $\pi_i(\chi) = \chi^{(1)}_{i_1}\chi^{(2)}_{i_2}$ for χ_i, and then to X itself, using a straightforward focus of attention:

$$\pi_{(b,i_b)}(\eta) = \eta^{(b)}_{i_b}, \text{ where } \sum_{i_b} \eta^{(b)}_{i_b} \leq cn_b. \tag{104}$$

¿From equations (41) and (42), we can calculate

$$E'^{(1)}_{\chi;i_1}[\chi, \bar{\mathbf{v}}] = E_{\chi,i}[\chi, \bar{\mathbf{v}}]\dot{\chi}^{(1)}_{i_1} = -g'_\chi(g_\chi^{-1}(\chi^{(1)}_{i_1}))\Big(\sum_{i_2} \chi^{(2)}_{i_2} E_{;i}[\bar{\mathbf{v}}] + \;..\Big) \tag{105}$$

and

$$E'^{(2)}_{\chi;i_2}\;[X, \;`] = E_{\chi,i}[\bar{\chi}, \bar{\mathbf{v}}]\dot{\chi}^{(2)}_{i_1} - g'_\chi(g_\chi^{-1}(\chi^{(2)}_{i_2}))\Big(\sum_{i_2} \chi^{(1)}_{i_1} E_{;i}[\bar{\mathbf{v}}] + \;.\Big) \tag{106}$$

Then the doubly attentive clocked objective function becomes

$$\begin{aligned} E_{\mathrm{orthog}} = & E'^{(1)}_{\chi;i_1}[\bar{\chi}, \bar{\mathbf{v}}] + E'^{(2)}_{\chi;i_2}[\bar{\chi}, \bar{\mathbf{v}}] + \Phi(\sum_{i_1} \nu^{(1)}_{i_1} - cn_1) + \Phi(\sum_{i_2} \nu^{(2)}_{i_2} \quad cn_2) \\ & + \sum_{i_1} \phi_\nu(\nu^{(1)}_{i_1}) + \sum_{i_2} \phi_\nu(\nu^{(2)}_{i_2}) \\ \oplus & \sum_i \chi^{(1)}_{i_1}\{\nu^{(1)}_{i_1} + \chi^{(1)}_{i_1}\}\chi^{(2)}_{i_2}\{\nu^{(2)}_{i_2} + \chi^{(2)}_{i_2}\}E_{;i}[\bar{\mathbf{v}}] \\ & + \Phi(\sum_{i_1} \chi^{(1)}_{i_1}\{\nu^{(1)}_{i_1} + \chi^{(1)}_{i_1}\} - n_1) + \Phi(\sum_{i_2} \chi^{(2)}_{i_2}\{\nu^{(2)}_{i_2} + \chi^{(2)}_{i_2}\} - n_2) \\ & + \sum_{i_1} \phi_{0/1}(\chi^{(1)}_{i_1}\{\nu^{(1)}_{i_1} + \chi^{(1)}_{i_1}\}) + \sum_{i_2} \phi_{0/1}(\chi^{(2)}_{i_2}\{\nu^{(2)}_{i_2} + \chi^{(2)}_{i_2}\}) \\ \oplus & E[\mathbf{v}\{\chi^{(1)} \otimes \chi^{(2)}\}]. \end{aligned} \tag{107}$$

The first phase may be traded in as before for a priority queue implementation; but the space cost of the default circuit implementation is already so small $(O(n_1 + n_2)$ for the kWTA network) that the priority queue is not necessary. In the second phase at most (c+ 1)$^2 n^2$ gradients $E_{;i}$ must be calculated, As in previous networks, one could make the efficient calculation of all gradients explicit by adding extra phases and variables.

The focus of attention introduced in this section applies when the neuron index i takes values in some domain which is a cross product of other domains, domain(i) $=$ domain$(i_1) \times$ domain(i_2). 'I'his is of interest for building complex network architectures by composing simpler elements. Another natural operation on index domains is the disjoint union $i = (b, i_b)$. The E_χ example above showed how to compose a focus of attention for this case as well (see equation (104), with $\sum_b cn_b \leq$ the number $\hat{n}$ of active neurons allowed), though that case is much simpler than for the cross product.

5 DISCUSSION AND CONCLUSIONS

In part I of this work wc introduced a Lagrangian formulation of the relaxation dynamics of neural networks which compute try optimizing an objective function in a standard neural network form. The Lagrangian formulation u iakes novel use of *a greedy functional derivative*, which we defined and computed. With these tools we demonstrated the use of three levels of optimization in the design of relaxation neural network dynamics: the original objective E, the Lagrangian L, and a Illcta-objective $\mathcal{M}$ which measures cost and functionality over many trials of the network.

In part II here we deal with a second group of more ramified applications. For these we introduced a clocked objective function and an associated notation. These constructs have the capability to clamp or unclamp net variables depending on the values of other of the net variables. This notation and tile stepwise refinement strategy for designing clocked objective functions sufficed to obtain computational attention *mechanisms*. Analogous to virtual memory or virtual processors in digital computers, such computational attention mechanisms have a focus of attention quality which can take a variety of forms. These include a priority queue, a set of coarse-scale blocks of neurons which could be scheduled according to their expected synergies in optimization, a set of jumping and rolling rectangular windows in a two-dimensional network, a sparse set of active neurons for which the excluded *latent* **neurons** require no memory, and the cartesian product of several simpler foci of attention. Each of these cases **was** concisely expressed using simple analytic notation with clocked objective functions. Reference was made to a number of experiments, application aucl computation, which employ the greedy variational and clocking calculus which we have introduced here.

Acknowledgements

We wish to acknowledge Charles Garrett and K. Srinivas for unpublished simulations; also discussions with Roger Smith, Chien-Ping Lu, Anand Rangarajan, Paul Cooper, Stan Eisenstat and Alain Martin; also the hospitality of the Institute for Theoretical Physics at Santa Barbara. This research was supported in part by AFOSR grant AFOSR-88-0240 and by ONR grant N00014-92-J-4048.

References

[111189] Eric B. Baum and David Haussler. What size net gives valid generalization? *Neural Computation*, 1(1):151–160, Spring 1989.

[BSB+91] Bernhard E. Boser, Eduard Sackinger, Jane Bromley, Yann LeCun Richard E. Howard, and Larry D. Jackel. An analog neural network processor and its application to high-speed character recongnition. In *International Joint Conference on Neural Networks*, pages I- 415 to 1-421. IEEE, July 1991.

[BT89] Dimitri P. Bertsekas and John N. Tsitsiklis. *Parallel and Distributed Computation*, chapter 3, pages 210-217. Prentice Hall, 1989.

[Coo89] Paul R. Cooper. *Parallel Object Recognition from Structure (The Tinkertoy Project)*. PhD thesis, University of Rochester Department of Computer Science, July 1989. Technical Report 301.

[Fah79] S. E. Fahlman. *NETL: A System for Representing and Using Real- World Knowledge*. MIT Press, 1979.

[GLR+95] Steven Gold, Chien-Ping Lu, Anand Rangarajan, Suguna Pappu, and Eric Mjolsness. New algorithms for 2d and 3d point matching: Pose estimation and correspondence. In G. Tesauro, D. Touretzky, arid T. Leen, editors, *Advances in Neural Information Processing System 7*. MIT Press, 1995.

[HT86] J. J. Hopfield and D. W. Tank. Collective computation with continuous variables. In *Disordered Systems and Biological Organization*, pages 155--170. Springer-Verlag, 1986.

[LM94] Chien Ping Lu and Eric Mjolsness. Two-dimensional object localizat ion by coarse-to-fine correlation matching. In J. Cowan, G. Tesauro, and J. Alspector, editors, *Advances in Neural Information Processing Systems 6*. Morgan -Kaufmann, 1994.

[MC80] Carver Mead and Lynn Conway. *introduction to VLSI Systems*. Addison-Wesley, 1980.

[MG90] Eric Mjolsness and Charles Garrett. Algebraic transformations of objective functions Neural ***Networks, 3:651-669,*** *1990*.

[MGA89] Eric Mjolsness, Gene Gindi, and P. Anandan. Optimization in model matching and perceptual organization. *Neural Computation*, 1, 1989.

[MGM91] Eric Mjolsness, Charles D. Garrett, and Willard L. Miranker. Multiscale optimization in neural nets. *IEEE Transactions* on *Neural Networks*, 2(2), March 1991.

[Mjo87] Eric Mjolsness. Control of attention in neural networks. In *Proc. of First International Conference on Neural Networks, volume vol.* 11, pages *567-574.* IEEE, 1987.

[MM] Eric Mjolsness and Willard L. Miranker. A Lagrangian formulation of neural networks I: Theory and analog dynamics. Part I of this paper.

[MM91] Eric Mjolsness and Willard 1,. Miranker. A Lagrangian approach to fixed points. In Richard P. Lippmann, John E. Moody, and David S. Touretzky, editors, *Neural Information Processing System 3.* Morgan Kaufmann, 1991.

[MW66] W.L. Miranker and B. Weiss. 'l'he Feynman operator calculus. *SIAM Review*, 8:224-232, 1966.

[PB87] John C. Platt and Alan H. Barr. Constrained differential optimization. In Dana Z. Anderson, editor, *Neural information Processing Systems.* American Institute of Physics, 1987.

[1'S89] C. Peterson and B. Soderberg. A new method for mapping optimization problems onto neural networks. *International Journal of Neural Systems,* 1 (3), 1989.

[RGM96] Anand Rangarajan, Steven Gold, and Eric Mjolsness. A novel optimizing network architecture with applications. *Neural Computation, 8(5), 1996.*

[Tou86] David S Touretzky. *The Mathematics of Inheritance Systems.* Morgan Kaufmann Pulishers, 1986.

[Tsi97] Dimitris Ioannis Tsioutsias. *Multiscale Attention as a Gloabally Convergent Framework for Large-Scale* Nonlinear *Optimization.* Phi) thesis, Yale University Computer Science Department, May 1997. See chapters 3 and 4.

[Vap82] V. N. Vapnik. *Estimation of Dependences Based* on *Empirical Data.* Springer Verlag, 1982.