The Robustness of Relaxation Rates in Constraint Satisfaction Networks

D. Randall Wilson
Dan Ventura
Brian Moncur
fonix corporation
WilsonR@fonix.com

Tony R. Martinez
Computer Science Department
Brigham Young University
martinez@cs.byu.edu
http://axon.cs.byu.edu

Abstract

Constraint satisfaction networks contain nodes that receive weighted evidence from external sources and/or other nodes. A relaxation process allows the activation of nodes to affect neighboring nodes, which in turn can affect their neighbors, allowing information to travel through a network. When doing discrete updates (as in a software implementation of a relaxation network), a goal net or goal activation can be computed in response to the net input into a node, and a relaxation rate can then be used to determine how fast the node moves from its current value to its goal value. An open question was whether or not the relaxation rate is a sensitive parameter. This paper shows that the relaxation rate has almost no effect on how information flows through the network as long as it is small enough to avoid large discrete steps and/or oscillation.

1. Introduction

Constraint satisfaction neural networks [1, 2, 3, 4] contain nodes that are connected by excitatory (positive) or inhibitory (negative) weights. Nodes are given initial activation values, after which a relaxation process causes nodes to change their activation value in response to the net input of all of the weighted connections coming into each node. When doing discrete updates on a Hopfield network [1], one way of computing activation for each node y at each *relaxation step*, s, is as follows. First the current "goal" net input into node y is computed as

$$GoalNet_y(s) = \sum_{x=1}^{m} A_x(s\text{-}1) \cdot W_{xy} \quad (1)$$

where s is the number of relaxation steps taken so far, m is the number of nodes connected to node y, A_x is the activation of node x, and W_{xy} is the weight from node x to node y. The current input net for node y is moved a fraction dt ($0 < dt \leq 1$) from its present value to the goal net value:

$$Net_y(s) = Net_y(s\text{-}1) + dt \cdot (GoalNet_y(s) - Net_y(s\text{-}1)) \quad (2)$$

Finally, the activation of y is computed by running its net through a squashing function such as a sigmoid function:

$$A_y(s) = \frac{1}{2}\left(\tanh\left(\frac{Nety(s)}{\mu}\right)+1\right) \quad (3)$$

Where μ is an amplification parameter controlling the steepness of the activation function.

A new relaxation procedure for Hopfield networks called *Controlled Relaxation* (CR) was presented in [4] and [5]. CR uses a *goal activation*, $GoalA_y$, rather than a goal net value, and uses a *relaxation rate*, r, to control the speed and smoothness of relaxation. CR can also use a new bipolar (*i.e.*, -1..1) *Evidence-Based Activation* (EBA) function EBA(x) that is similar in shape to a sigmoid function except that it is flat near $x = 0$ so as to de-emphasize noise and amplify real evidence. For details on this activation function, see [6].

The net input value, goal activation and current activation using controlled activation and EBA are computed as follows:

$$Net_y(s) = \sum_{x=1}^{m} A_x(s\text{-}1) \cdot W_{xy} \quad (4)$$

$$GoalA_y(s) = \mathrm{EBA}(Net_y(s)) \quad (5)$$

$$A_y(s) = A_y(s\text{-}1) + r \cdot (GoalA_y(s) - A_y(s\text{-}1)) \quad (6)$$

Zeng & Martinez [4] applied CR to the 10-city traveling salesman problem (TSP) and were able to increase the percentage of valid tours by 195% while reducing the error rate by 35% when compared to standard Hopfield

relaxation applied using the same architectures. When the CR method used the EBA activation function instead of the standard sigmoid function, it was able to increase valid tours by 245% and reduce the error rate by 64% compared to corresponding Hopfield networks using sigmoid activation functions [5].

An open question was whether or not the relaxation rate was a sensitive parameter, *i.e.*, whether or not it determined how much the activation of one node can affect the activations of other nodes during relaxation. This paper shows that the relaxation rate has almost no effect on how information flows through the network as long as it is small enough to avoid large discrete steps and/or oscillation. Section 2 explains why the relaxation rate is theoretically robust, and Section 3 presents empirical results on a simple network illustrating the robustness of sufficiently small relaxation rates as well as the potential dangers of large ones. Section 4 then concludes and provides future research directions.

2. Relaxation Rates

In Equation 2, the parameter *dt* is used to determine how fast the current net value of a node moves towards a new net value. In Equation 6, the parameter *r* is used to determine how fast the current activation value of a node moves towards a new activation value. Though applied at different points in the process, both parameters do essentially the same thing, *i.e.*, they control the speed and smoothness of relaxation. For simplicity, this section discusses only the effect of *r* on the relaxation process in the CR relaxation approach, but the discussion applies to *dt* in Hopfield relaxation as well.

During each relaxation step *s*, the activation of node *y*, $A_y(s)$, moves a fraction r from its old activation $A_y(s\text{-}1)$ to its goal activation at *s*, $GoalA_y(s)$. For example, if $A_y(0) = 0$, $GoalA_y(1) = 1.0$, and $r = 0.2$, then $A_y(1) = 0.2$. If $GoalA_y(s)$ remains fixed at 1.0, $A_y(s)$ will asymptotically approach $GoalA_y(s)$ with values of 0, 0.2, 0.36, 0.48, ..., reaching 0.99 after about 20 relaxation steps. If instead we use $r = 0.02$, then $A_y(s)$ approaches $GoalA_y(s)$ more slowly with values of 0, 0.02, 0.0396, 0.0588, ..., and reaches 0.99 after about 200 relaxation steps. In other words, it takes approximately 10 times as many relaxation steps to reach the same value when using a relaxation rate that is 10 times as small.

The speed of relaxation can be normalized by taking both the number of steps and relaxation rate into account. Let the variable *t* indicate the elapsed *relaxation time* during relaxation. The relaxation rate can then be viewed as the change in relaxation time per relaxation step, *i.e.*, $r = \Delta t / \Delta s$, and thus we can define the relaxation time

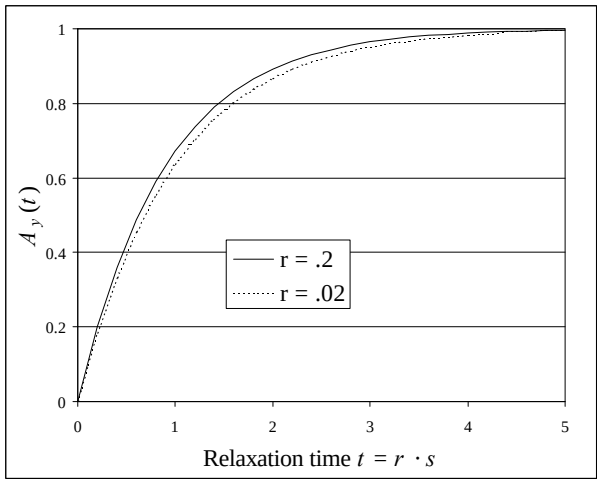

Figure 1. Simple relaxation with $r = .2$ and $r = .02$.

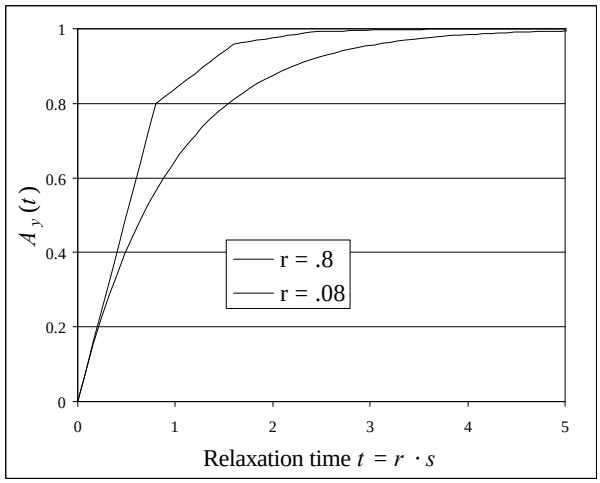

Figure 2. Simple relaxation with $r = .8$ and $r = .08$.

as $t = r \cdot s$. In the above example the activation reached 0.99 at about $t = r \cdot s = 0.2 \cdot 20 = 4.0$ when $r = 0.2$, and at $t = 0.02 \cdot 200 = 4.0$ when $r = 0.02$. While the smaller relaxation rate requires more relaxation steps, they both require approximately the same amount of relaxation time.

Figure 1 shows the activation with respect to *t* using $r = 0.2$ and $r = 0.02$ as *t* goes from 0 to 5. The smaller relaxation rate does cause a slightly slower change in $A_y(t)$ with respect to *t*, but the difference is fairly small. Larger relaxation rates cause the difference to be larger, while smaller relaxation rates cause the difference to be much smaller. For example, Figure 2 shows the relaxation behavior when relaxation rates of $r = 0.8$ and $r = 0.08$ are used. Note that these relaxation rates differ by a factor of 10 just like in Figure 1, but the difference in relaxation behavior is much more exaggerated. In the limit, using $r = 1$ reaches the goal activation at $t = 1$, *i.e.*, in one step,

while any value of $r < 1$ will never quite reach the goal activation but will only asymptotically approach it. Using a smaller relaxation rate not only smooths the relaxation path, but also reduces sensitivity to the size of the relaxation rate itself. Thus, the smaller the relaxation rate is, the less it matters exactly what value is used. However, since using a smaller relaxation rate requires more iterations through the relaxation process, using too small of a relaxation rate is inefficient.

3. Relaxation Experiments

In the above examples, the goal activation remained fixed at 1.0. When the goal activation for each node depends on the current activations of other nodes, the situation is more complicated. An open question was whether the relaxation rate influences to what degree information can flow through a network. Section 3 addresses this question.

If the relaxation rate influences the degree to which one node can affect other nodes in the network (including nodes to which it is not directly connected), then it can alter the outcome of the network and thus affect accuracy. To see what effect the relaxation rate has on direct and indirect interactions between nodes, several architectures were used with various weight settings between nodes and different initial activation values. Relaxation rates from 0.00001 to 2.0 were used to see what effect the relaxation rate has on the relaxation process and the final state of the network in each case.

One of the architectures used in these experiments is illustrated in Figure 3. The five nodes in the network receive initial activation values of 0.5, 0.6, 0.8, 1.0, and -1.0, respectively, as shown in the figure 3. For simplicity, all weights are set to 10.0. Relaxation proceeds as described in Equations 4-6 until convergence. In these experiments the network is considered to have *converged* if the maximum difference between any node's activation and its goal activation is less than 0.001.

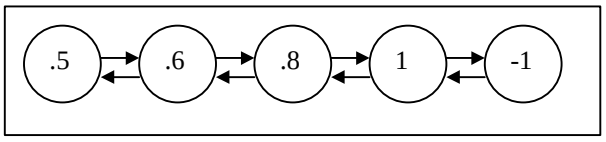

Figure 3. Relaxation network with 5 nodes. Numbers indicate initial activation values, and all weights are equal.

Figures 4-8 show the results for relaxation rates of 0.0001, 0.1, 0.5, 0.8, and 1.0, respectively. The activations of the five nodes are plotted with respect to the relaxation time $t = r \cdot s$, where r is the relaxation rate, and s is the number of relaxation steps.

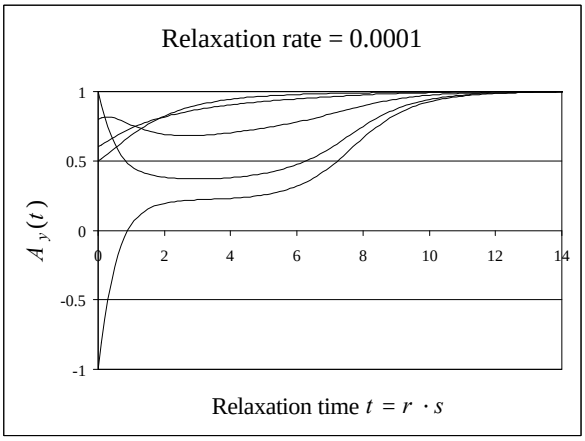

Figure 4. Relaxation with $r = 0.0001$.

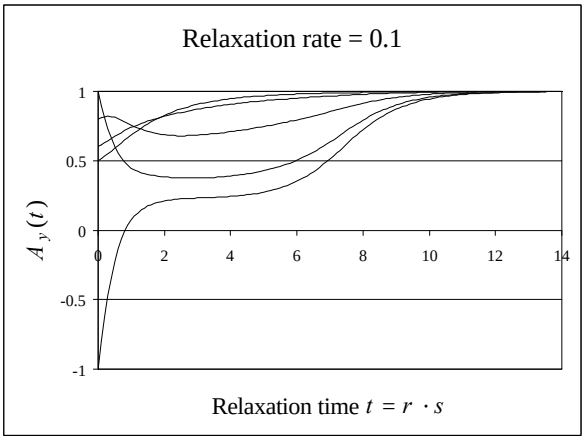

Figure 5. Relaxation with $r = 0.1$.

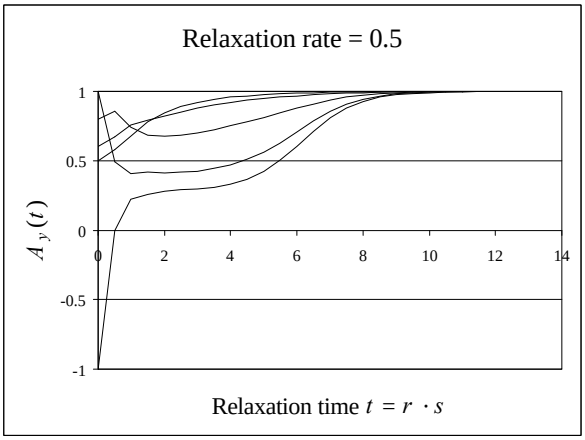

Figure 6. Relaxation with $r = 0.5$.

Figures 4 and 5 appear identical even though the relaxation rate in Figure 4 is 1,000 times smaller than the one in Figure 5. This supports the hypothesis that the relaxation rate is a robust parameter and does not affect how far information travels through a relaxation network (when it is fairly small). Convergence occurred after 149,609 steps for $r = 0.0001$, and after 145 steps for $r = 0.1$, corresponding to $t = 14.9$ and $t = 14.5$, respectively, indicating that the relaxation time is not affected much by smaller relaxation rates.

Figure 6 uses $r = 0.5$, and some discrete steps in the relaxation process are visible. These large steps do have a small effect on the relaxation process, causing it to move towards convergence somewhat more quickly, though less smoothly, similar to the effect shown in Figures 1 and 2 when the relaxation rate was increased.

Figure 7 uses $r = 0.8$, and with this large of a relaxation rate, oscillations occur. At each step, every node y moves more than halfway between its current activation and its goal activation. The fifth node, for example, begins at $A_y = -1.0$, and at the very next step its activation is increased to $A_y = 0.6$. Meanwhile, node 5 has caused node 4 to go negative, which in turn causes node 5 to go negative again on the third relaxation step, and so on, until convergence is reached. When the relaxation rate grows too large, such oscillations can become infinite, as shown in Figure 8 using $r = 1.0$. After a few relaxation steps, the activations of the nodes in this case flip-flop between 1.0 and 0 indefinitely.

These results illustrate that using a smaller relaxation rate does not appreciably change the final activations of the nodes in the network *nor the path taken to get to the final state*. For reasonably small values of r, this simple network always converged at approximately $t = r \cdot s = 14$. If the relaxation rate is too large, e.g., over $r = 0.3$ in this network, the large discrete steps taken can produce slight variations, as shown in Figure 6 when $r = 0.5$; or minor oscillations can occur in the path taken to the final state, as shown for $r = 0.8$ in Figure 7. If a sufficiently large value of r is used, infinite oscillations can occur, as illustrated in Figure 8.

In the simple five-node network illustrated in this example, the relaxation rate did not have much effect on the final activations of the nodes in the network. In more complex networks [7, 4, 5], however, using too large a relaxation rate can have a significant effect on the outcome of relaxation due to lack of smoothness and/or oscillation. In [4], for example, CR networks were used for the 10-city TSP problem, and the percentage of valid tours dropped quickly as r rose above 0.14. However, the performance was quite stable with $r < 0.14$, indicating that even on

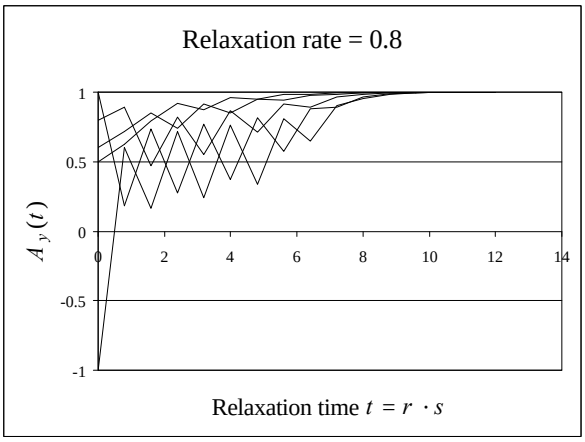

Figure 7. Relaxation with $r = 0.8$.

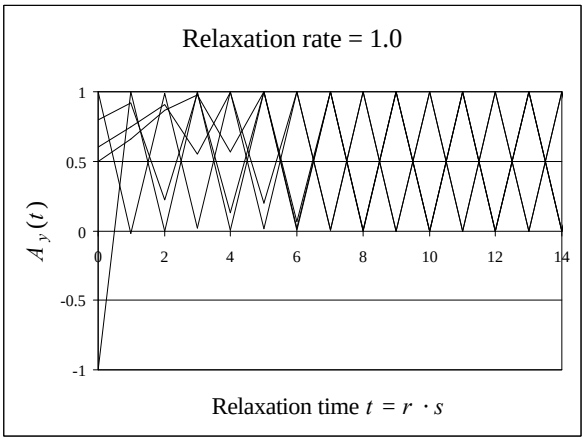

Figure 8. Relaxation with $r = 1.0$.

complex problems, values of $r < 0.1$ appear to be reasonably robust. It should be noted that [4] reports a reduction in valid tours for very small values of r, *e.g.,* $r < 0.06$, but this occurred because they limited their networks to 500 relaxation steps (*iterations*) instead of allowing the network to relax until convergence.

4. Conclusions

The relaxation rate in constraint satisfaction networks of the type discussed in this paper does not significantly affect how much information propagates through the network, as long as it is reasonably small. Very small values of r do not substantially affect the relaxation process and are thus not worth the extra computation they require.

While the results presented in Section 3 use the *Controlled Relaxation* method [4] and the *Evidence-Based Activation* function [5], experiments run using sigmoidal activation

functions yielded the same conclusions. Furthermore, the same conclusions appear to apply to standard Hopfield networks [1], namely, that the parameter *dt* is a robust parameter as long as it is sufficiently small to allow relaxation to be fairly smooth. In our experiments as well as those of Zeng & Martinez [4], values less than 0.1 have typically been sufficiently small, though this value should not be trusted as more than a rule of thumb until more experience on a broad range of applications is tested.

Future research will address the question of whether analytical proofs can be derived to show more precisely what effect the relaxation rate has on activation values, as well as applying CR networks to complex tasks such as speech recognition [8].

References

[1] Hopfield, J. J., and D. W. Tank, (1985). "Neural Computations of Decisions in Optimization Problems." Biological Cybernetics, vol. 52, pp. 141-152, 1985.

[2] Ackley, David H, Goeffrey E. Hinton, and Terry J. Sejnowski, (1985). "A Learning Algorithm for Boltzmann Machines," *Cognitive Science*, vol. 9, pp. 147-169.

[3] Kosko, B., (1988). "Bidirectional Associative Memories." *IEEE Transactions on Systems, Man, and Cybernetics,* vol. 18, no. 1.

[4] Zeng, Xinchuan and Tony R. Martinez, (1999). "A New Relaxation Procedure in the Hopfield Network for Solving Optimization Problems." To appear in *Neural Processing Letters*.

[5] Zeng, Xinchuan and Tony R. Martinez, (1999). "Extending the Power and Capacity of Constraint Satisfaction Networks." In *Proceedings of the IEEE International Joint Conference on Neural Networks* (*IJCNN'99*) (this proceedings), 1999.

[6] Zeng, Xinchuan and Tony R. Martinez, (1999). "A New Activation Function in the Hopfield Network for Solving Optimization Problems." In *Proceedings of the International Conference on Neural Networks and Genetic Algorithms*.

[7] Zeng, Xinchuan and Tony R. Martinez, (1999). "Improving the Performance of the Hopfield Network by Using a Relaxation Rate." In *Proceedings of the International Conference on Neural Networks and Genetic Algorithms*.

[8] Ventura, Dan, D. Randall Wilson, Brian Moncur, and Tony R. Martinez, (1999). "A Neural Model of Centered Tri-gram Speech Recognition." In *Proceedings of the International Joint Conference on Neural Networks* (*IJCNN'99*) (this proceedings).

Publications for D. Randall Wilson, Dan Ventura, Tony R. Martinez and Xinchuan Zeng can be found on the internet at http://axon.cs.byu.edu.