International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

Improvement of Intrusion Decision for I^2D^2RS with BPNN

Hongmei Kai [1], **Hongbing Zhu** [2], **Xiaoli Lin** [1], **Kei Eguchi** [3]

[1] *Hiroshima Kokusai Gakuin University, 6-20-1 Nakano Aki-ku, Hiroshima 739-0321, Japan*
[2] *Wuhan University of Science and Technology, Wuhan 430081, China*
Hiroshima Kokusai Gakuin University, 6-20-1 Nakano Aki-ku, Hiroshima 739-0321, Japan
[3] *Shizuoka University, 836, Ohya, Shizuoka 422-8529, Japan*

Abstract: Neural networks have good learning and associative memory abilities and have been widely applied to various fields. We employed the Backpropagation Neural Network (BPNN) to replace the fuzzy methods of the Intelligent Intrusion Detection, Decision, Response System (I^2D^2RS) [5] to decide the intrusion. Through this improvement the processing of the system was simplified and the performance of the system was enhanced in the intrusion decision. The effectivities of these improvements were confirmed with the experiments.

Keywords: Neural network, Learning, Associative memory, BPNN, Fuzzy, I^2D^2RS

1. Introduction

An intrusion detection system (IDS) generally detects unwanted manipulations of computer systems, mainly through the Internet. The manipulations may take the form of attacks by crackers. Several types of malicious behaviors include network attacks against vulnerable services, data driven attacks on applications, host based attacks such as privilege escalation, unauthorized logins and access to sensitive files, and malware (viruses, trojan horses, and worms), which can compromise the security and trust of a computer system.

Over the years, there have been intelligent methods and technologies widely applied to IDS, for example the Self-Organizing Networks[1], the Intelligent Agents[2], the Fuzzy Cognitive Maps[3], the Fuzzy Logic Agent[4] *etc*. Through these methods, the performance of the systems have been improved at some extent. However, some problems still remain:

- each database must be built and updated by an administrator
- intrusive decisions and responses depend on the administrator
- the system's structures have become more complicated and enlarged.

These aforementioned problems damage the celerity, dynamicity, reliability and robustity of the systems and limit the effectiveness of the systems. In the Reference [5], we proposed one novel intelligent intrusion detection, decision, response system (I^2D^2RS) with fuzzy rule-base method. This I^2D^2RS utilizes the two essential informations of the failed login's users: the failed login's times and time, to decide automatically who is a normal user and who is a intrusive user from the failed login's users using the fuzzy rules built on the measures and skills of the experienced system/security administrators. Figure 1 shows the procedure of the I^2D^2RS. The failed login surveillance monitors the system's log file to detect the failed login event, and the data processing picks up the necessary data: the failed login's times and the periodicity of time intervals, from the information of the failed

[2] Corresponding author.
Email address: kohe@wuchang.cs.hkg.ac.jp

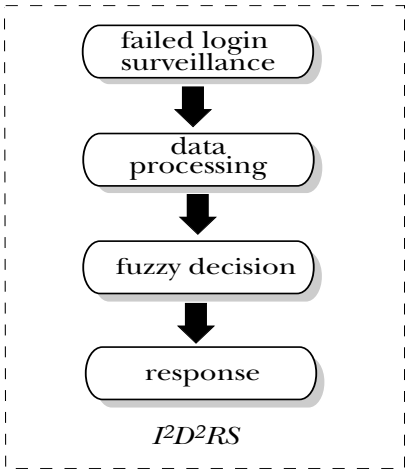

Figure 1. Flowchart of processing

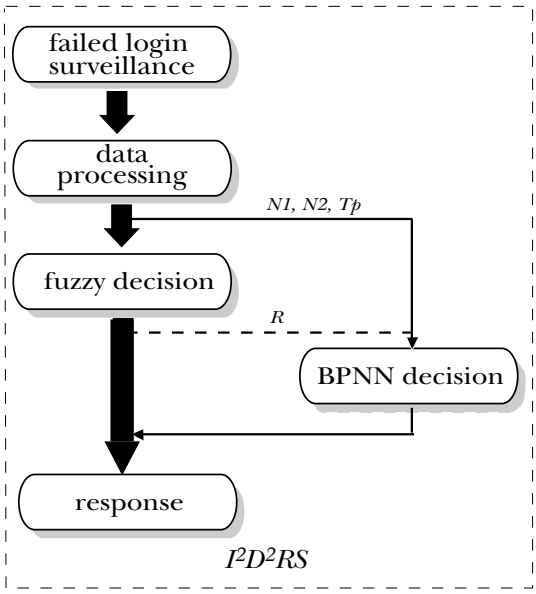

Figure 2. Flowchart of the process with BPNN

login. The fuzzy decision utilizes the fuzzy rules of the times and the periodicity of time intervals to decide the intrusion. And then the response makes the accurate response to the intrusion.

Though the fuzzy decision perfectly decides the intrusion, the results of the decision aren't used for recognizing the same or similar intrusions that occur again, for the fuzzy logic is the lack of the abilities of learning and memory. So the performance of the system is limited. Neural networks are the intelligent approaches that have been successfully applied to solve the pattern recognition and classification problems with their learning and associative memory abilities. So that, we replaced the fuzzy decision with the Backpropagation Neural Network (BPNN) that was structured with the Neural Network Toolbox of MATLAB [6]. With this approach, the procedure of the intrusive decision was simplified and the efficiency was the same as the I^{2}D^{2}RS with fuzzy method, which was confirmed through experiments.

The rest of this paper is organized as follows. The processing of the improvement I^{2}D^{2}RS with BPNN is described in Section 2. Section 3 explains what is the BPNN's Syntax of MATLAB and how to select the training function for the BPNN, the numbers of layers and the numbers of neurons at each layer. Next gives the results of intrusion decision using the trained BPNN. A lot of experimental results are shown in Section 4. The conclusion and the future study are given in the last section.

2. Improvement

Figure 2 shows the flowchart of the processing of the improvement I^{2}D^{2}RS with BPNN. While the announcement of the failed login comes from the failed login surveillance, the data processing picks up $N1$ – the times of the failed login during the short specific time, $N2$ – during the long time and T_p – the periodicity of time intervals of the failed login's user, and then transfers these informations into the fuzzy decision for deciding the intrusion. With the antecedent clauses $\{N1, N2, T_p\}$ of the fuzzy inference, the fuzzy decision utilizes the following fuzzy IF - THEN rules to evaluate the fuzzy consequence – the danger degree R.

$$IF\ N1\ is\ A_i,\ N2\ is\ B_i\ and\ T_p\ is\ C_i$$
$$THEN\ R\ is\ D_i\ \ (i = 1, 2, ..., n) \qquad (1)$$

As the mentioned shortcomings of fuzzy method in Section 1, the BPNN decision is employed to replace the fuzzy decision to decide the intrusions. In order to train the BPNN decision the inputs of the training patterns $(N1, N2, T_p)$ are obtained from the data processing and the target output R of the training patterns is get from the fuzzy decision of the original I^{2}D^{2}RS with fuzzy method. When the training BPNN is finished the fuzzy decision will be replaced with the trained BPNN decision, which will be illustrated with the thin real line of the Figure 2. Through the improvement the procedure of the intrusion decision is simplified and the performance of data processing ability is enhanced.

3. Backpropagation Neural Network

The backpropagation (BP) algorithm is utilized to train multilayer networks on the basis of computing

the derivatives of the squared error with respect to the weights and biases in the hidden layers. It is called backpropagation because the derivatives are computed first at the layer – the output layer of the network, and then propagated backward through the network to compute the derivatives in the hidden layers. The structure of BPNN is shown in Figure 3.

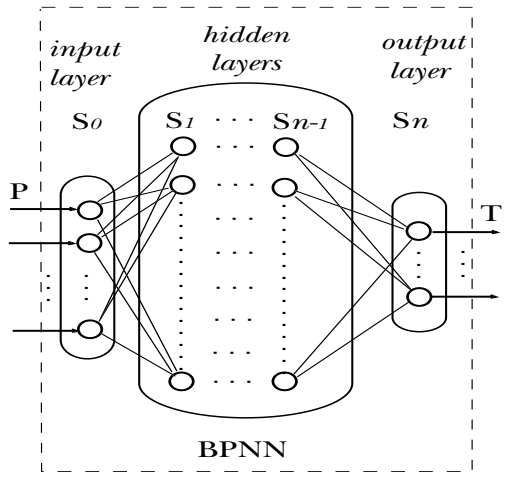

Figure 3. Structure of BPNN

3.1 BPNN of MATLAB

With the structure of Figure 3 the BPNN's Syntax[6] that is the rules governing construction of neural network, of MATLAB is represented in the following.

Syntax:

$$net = newff(P, T, [S_1 S_2 ... S_n], TF_1 TF_2 ... TF_n, BTF, BLF, PF) \quad (2)$$

where

- $newff$: Function for creating a feedforward backpropagation neural network
- P : $R \times Q$ matrix of Q sample R-element input vectors
- T : $M \times N$ matrix of N sample M-element target vectors
- S_i : Size (the number of neurons) of the i-th layer for n layers
- TF_i : Transfer Function of the i-th layer
- BTF : Backpropagation Training Function of neural network
- BLF : Backpropagation weight/bias Learning Function of neural network
- PF : Performance Function of neural network

3.1.1 Experiments' Environments

On the above mentioned **Syntax**, BTF and S_i are selected for structuring BPNN with a lot of experiments that of environments are shown in Table 1.

Table 1. Experiments' environments

CPU	Intel D950(3.4GHz)
RAM	2GB
OS	Fedora Core 6
PR	0.00∼1.00
$Epochs$	5000
$Goal$	0.0001
TF_i	logsig

PR : the range of input P
Epochs : maximum loops number of training
Goal : error goal of neural network
logsig : Log-Sigmoid transfer function

3.1.2 Select BTF

The training function (BTF) trains the network to eventually converge to a solution. For the origin BP algorithm is too slow in converging, some acceleration of convergence methods are used for MATLAB's training functions. With some decision experiments in the different BTF (see Table 2), the function $trainlm$ was finally decided.

Table 2. Results of various BTF

BTF	$\boldsymbol{C}$	$\boldsymbol{E}$	$\boldsymbol{interpretation}$
$traingd$	×	6.50	steepest descent BP
$trainbfg$	×	0.43	Quasi-Newton algorithms
$trainrp$	×	0.82	resilient back-PROPagation
$trainscg$	×	19.33	scaled conjugate gradient
$trainlm$	○	0.43	levenberg-marquardt

$\boldsymbol{E}$: network error
× : no convergence
$\boldsymbol{C}$: convergence
○ : convergence

3.1.3 Select S_i

A lot of experiments for selecting S_i were done and the results of the 3-layers' BPNN are shown in Table 3 and the 4-layers' BPNN are shown in Table 4. The Table 3 and 4 indicate that the increase of the numbers of the neurons can not improve the performance of network. The 3-layers' network wasn't converged to a solution without the large numbers of neurons, so the 4-layers' network with the small neurons was chosen. The 4-layers' 3-21-7-1 (the number of neurons for each layer), which has the smaller network

Table 3. 3-layers' BPNN

S_0	3	3	3	3	3
S_1	20	40	60	80	100
S_2	1	1	1	1	1
$\boldsymbol{E}$	1.0693	0.5915	0.4840	0.4480	0.4266
$\boldsymbol{C}$	×	×	×	×	○

Table 4. 4-layers' BPNN

S_0	3	3	3	3	3
S_1	15	18	21	24	27
S_2	5	6	7	8	9
S_3	1	1	1	1	1
$\boldsymbol{E}$	0.2997	0.3008	0.2854	0.3014	0.3015
$\boldsymbol{C}$	×	○	○	○	○

error, is employed for deciding the intrusions in our system.

Base on the probability theory and statistics, the total number of the input-output instances $\{(N1, N2, T_p), R\}$ that the training patterns are picked up from, are computed as the following equation.

$$(C^1_{n_1} \times C^1_{n_2} - \sum_{i=1}^{n_1-1} i) \times C^1_{t_p} \quad (n_2 \geq n_1) \tag{3}$$

In this paper, the n_1 is set into 10, n_2 is 50 and the t_p is 4 respectively, so the total numbers is 1820. The results for the different numbers of training patterns are shown in Table 5. From that Table, it is seen that

Table 5. Results of intrusion decision

S_0	S_1	S_2	S_3	$\boldsymbol{NP}$	$\boldsymbol{T}(\%)$	$\boldsymbol{R}(\%)$
3	21	7	1	1214	66.7	96.20
				910	50.0	96.81
				606	33.3	94.81
				455	25.0	92.09
				364	20.0	94.64
				182	10.0	84.98

NP : the numbers of input P
T : the percentage of the total numbers of instances
R : the percentage of correct decision

though the numbers of training patterns are increased largely, the change of the percent of the correct decision rate is a little. So one BPNN with high correct decision rate would be taken by the small number of training patterns.

3.2 Training Time

Training time is the point that the fuzzy decision is replaced with the trained BPNN decision at. Figure 4

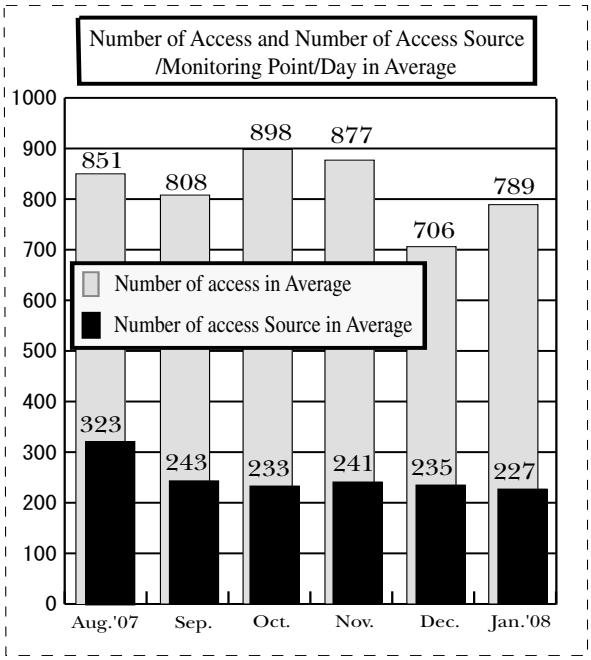

Figure 4. Number of access unwanted (one-sided) and number of access source/monitoring point/day

shows the number of access unwanted (one-side) and of access source for one monitoring point at one day during August 2007 – January 2008, which are from the Internet Monitoring (called TALOT2) of the IPA (Information - technology Promotion Agency, Japan) [23] and the average of the 10 monitoring points. According to the report of IPA, since each monitoring environment for the TALOT2 is nearly equal to the general connection environment used for the Internet, it can be considered that the same amount of access unwanted (one-sided) can be monitored for the general Internet users' connection environment. In another word, your computer is being accessed from 227 unknown source addresses in average at one day or you are being accessed about 3 times from one source address unauthorized. From the Figure 4, the least number of access unwanted is over 700 at one day, so the training time can be estimated with the percent of updated training patterns at one day. For example, if the percentage of updated is 5% then the new training patterns are 35 at the 700 access unwanted of one day, the training time of obtaining 364 training patterns is about 11 days. The training time is estimated with the

following equation.

$$Days = \frac{NP}{n\% \times NA} \quad (4)$$

where NA is the number of access unwanted and $n\%$ is the percentage of updated training patterns at one day. With the equation the training times for the different number of training patterns were estimated and the results are shown in Table 6.

Table 6. Training time

NP	Percentage of update						
	1%	3%	5%	7%	9%	11%	13%
	Days						
910	130	44	26	19	15	12	10
364	52	18	11	8	6	5	4

From the table it is seen that:

- with the increase of the number of training patterns the training time becomes long.
- with the increase of the percentage of updated training patterns at one day the training time becomes short.

4. Experiment

With the same experiment environments as that of the Reference [5], the two kinds of the experiments were done and are shown in Table 7.

Table 7. Two kinds of experiment

Name		$N1$	$N2$	T_p
Exp1	old	1∼$N2_{max}$	1∼$N2_{max}$	$PG1{\sim}PG4$
	new			
Exp2	old	1∼$N1_{max}$	1∼$N2_{max}$	$PG1{\sim}PG4$
	new			

where "Exp1-old" and "Exp2-old" are the experiments with the fuzzy method, "Exp1-new" and "Exp2-new" are the experiments with the BPNN approach, "1∼ $N2_{max}$" is the $N1$ taking value from 1 to $N2$'s maximum that was five times $N1_{max}$, "1∼$N1_{max}$" is the $N1$ taking value from 1 to $N1$'s maximum that was taken as 10 and being reset at $N1 = N1_{max}$ in our experiments, "$PG1{\sim}PG4$" are four grades of periodicity of five time intervals. The short specific time and the long specific time were set into the 24 hours and 30 days respectively. The responses were divided into four level corresponding to the different degrees of danger as following[5]:

Level 1 : corresponding to the degrees 1∼3 where the accident is neglected by the system

Level 2 : corresponding to the degrees 4∼5 where the user receives the warning alarm

Level 3 : corresponding to the degrees 6∼7 where the system gives the prohibition for the username

Level 4 : corresponding the degree 8 and over where the system passes the user's machine IP address to the firewall

Figure 5 ∼ 12 show the experiments' results of the NP being 910 and Figure 13 ∼ 20 show the experiments' results of the NP being 364. From these results the following things are seen:

- the results of intrusion decision for the two methods completely match at the above mentioned dangerous degree levels.
- the results of intrusion decision for the 910 and 364 training patterns are stupendously similar.

5. Conclusion

In this paper one Backpropagation Neural Network was utilized to replace the fuzzy decision of the I^2D^2RS for deciding whether the failed login user is an intrusion or not. From the results of experiments the decision of the both methods was wonderfully similar and the processing was simplified with the BPNN decision. The future studies are to enhance the performance of the system to decide the various intrusions.

References

[1] Richard A. Wasniowski, "Using Self-Organizing Networks for Intrusion Detection ", *Proceedings of the 6th WSEAS Int. Conf. on Neural Networks*, pp.90-94, 2005.

[2] Curtis A. Carver, Jr., John M.D. Hill, John R. Surdu, Udo W. Pooch, "A Methodology for Using Intelligent Agents to Provide Automated Intrusion Response", *Proceedings of the 2000 IEEE Workshop on Information Assurance and Security United States Military Academy*, 2000.

[3] Ambareen Siraj, Susan M.Bridges, Rayford B.Vaughn, "Fuzzy Cognitive Maps for Decision Support in an Intelligent Intrusion Detection System", *http://www.security.cse.msstate.edu/docs/Publications/asiraj/nafips2001.pdf*.

[4] Wasniowski R A, "Intrusion Detection System with Fuzzy Logic Agent", *Wseas Transactions on Systems Issue10*, vol.3, pp.1109-2777, 2004.

[5] H. Kai, H. Zhu, K. Eguchi, N. Sun and T. Tabata, "A Novel Intelligent Intrusion Detection, Decision, Response System", *Ieice Trans. Fundamentals*, Vol.E89-A, NO.6, pp.1630-1637, 2006.

[6] http://www.mathworks.com/

[7] H. Kai, H. Zhu, K. Eguchi, Z. Guo, J. Wang, "Classification of Grades of Intervals with BPNN", *ITC-CSCC 2007 The 22th Commemorative International Technical Conference on Circuits/Systems, Computers and Communications*, Vol.3, pp.1057-1058, 2007.

[8] H. Kai, H. Zhu, K. Eguchi, Z. Guo, J. Wang, H. Zheng, "Application of Neuro-Fuzzy Approach for I^2D^2RS", *ICICIC2007 Second International Conference on Innovative Computing, Information and Control*, cd-rom (A11-5), 2007.

[9] J.T. Yao, S.L. Zhao and L.V. Saxton, "A Study on Fuzzy Intrusion Detection", *Data Mining, Intrusion Detection Information Assurance, and Data Networks Security 2005* Vol.5812, pp.23-30, 2005.

[10] Kazuo Tanaka, *Advanced Fuzzy Control*, Kyoritsu Shuppan Co.,Ltd, Japan, pp.9-50, 1994.

[11] Hagiwara Masahumi, *Neuron, Fuzzy, Genetic Algorithm*, Sangyoutosyo, Japan, 1994.

[12] Peng Ning, Sushil Jajodia, X. Sean Wang, *Intrusion Detection in Distributed Systems: An Abstraction-Based Approach*, Kluwer Academic Publishers, 2004.

[13] Tim Crothers, *Implementing Intrusion Detection Systems: A Hands-On Guide for Securing the Network*, Wiley Publishing, INC., 2003.

[14] Martin T. Hagan, Howard B. Demuth, Mark Beale, *Neural Network Design*, PWS Publishing Company, 1996.

[15] Amit Konar, *Computational Intelligence*, Springer-Verlag Berlin Heidelberg, 2005.

[16] Yoko Uwate, Yoshifumi Nishio, "Flexibility of an Affordable Neural Network for Back Propagation learning", *The 18th Workshop on Circuits and Systems in Karuizawa* pp.339-342, 2005.

[17] Y.Hitaka, M.Shirakata, H.Sato, M.Yokomichi, M.Kono and KEK PS, "Optimal Beam Orbit Generation Using Neural Network Algorithm", *Proceedings of the 1st Annual Meeting of Particle Accelerator Society of Japan and the 29th Linear Accelerator Meeting in Japan*, pp.408-410, 2004.

[18] Christian Kanzow, Nobuo Yamashita and Masao Fukushima, "Levenberg-Marquardt Methods for Constrained Nonlinear Equations with Strong Local Convergence Properties", *http://citeseer.ist.psu.edu/596808.html*, 2002.

[19] Yuehui Chen, Ajith Abraham and Bo Yang, "Hybrid Flexible Neural-Tree-Based Intrusion Detection Systems", *International Journal of Intelligent Systems* VOL.22, pp.337-352, 2007.

[20] Zheng Zhang, Jun Li, C.N. Manikopoulos, Jay Jorgenson and Jose Ucles, "a Hierarchical Network Intrusion Detection System Using Statistical Preprocessing and Neural Network Classification", *Workshop on Information Assurance and Security* pp.85-90, 2001.

[21] Jake Ryan, Meng-Jang Lin and Risto Miikkulainen, "Intrusion Detection with Neural Networks", *http://nn.cs.utexas.edu/downloads/papers/ryan.intrusion.pdf*.

[22] Raymond S.T.Lee, *Fuzzy-Neuro Approach to Agent Applications: From the AI Perspective to Modern Ontology*, Springer, 2006.

[23] http://www.ipa.go.jp/security/english/virus/press/200801/E_PR200801.html.

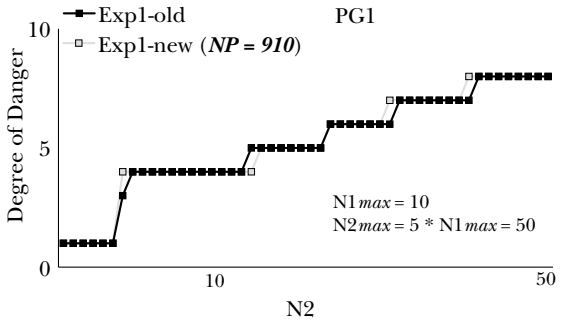

Figure 5. Results of Exp1

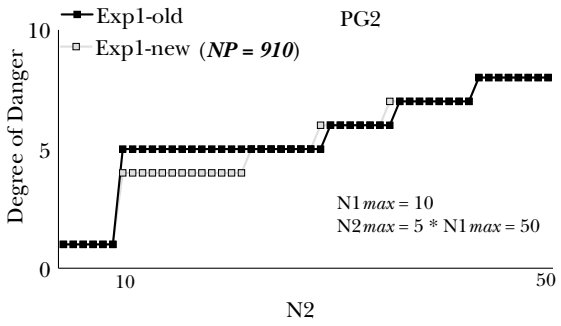

Figure 6. Results of Exp1

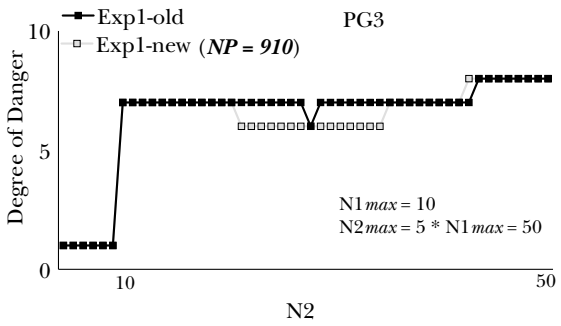

Figure 7. Results of Exp1

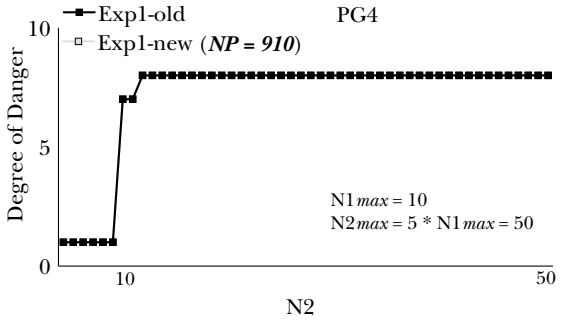

Figure 8. Results of Exp1

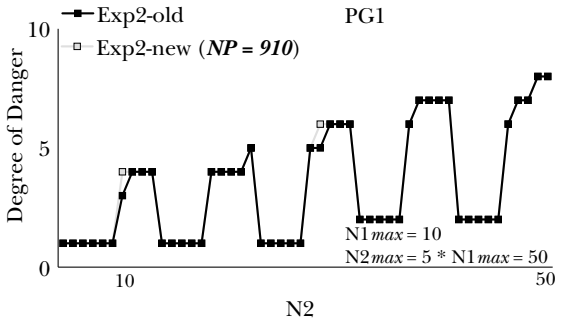

Figure 9. Results of Exp2 with reset

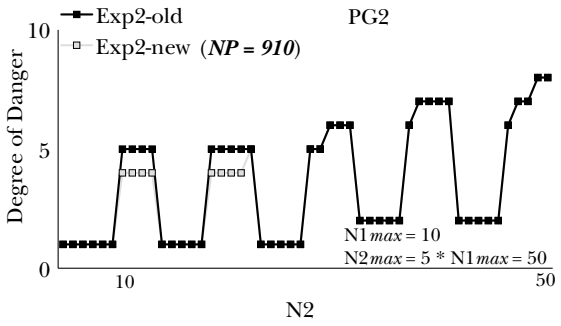

Figure 10. Results of Exp2 with reset

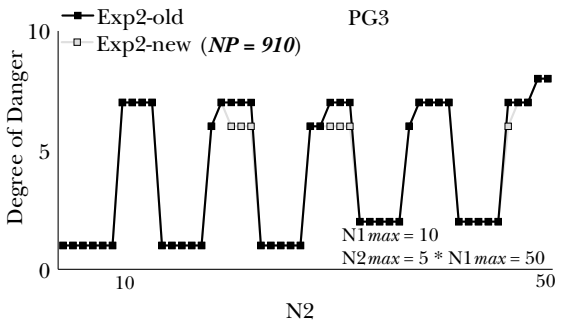

Figure 11. Results of Exp2 with reset

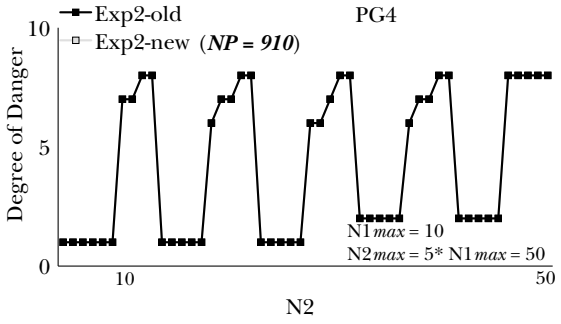

Figure 12. Results of Exp2 with reset

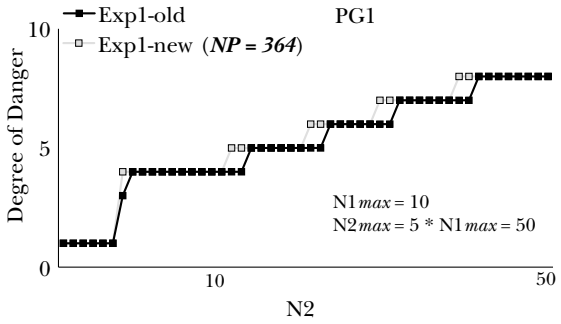

Figure 13. Results of Exp1

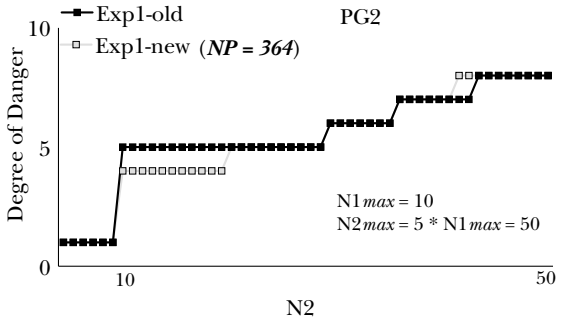

Figure 14. Results of Exp1

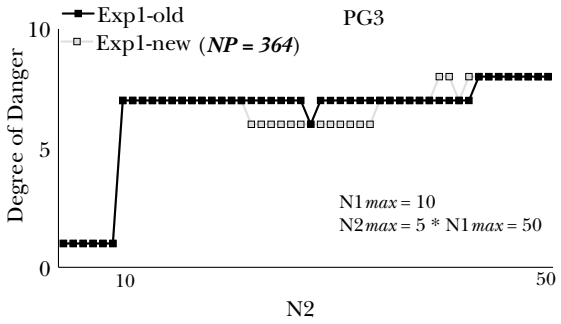

Figure 15. Results of Exp1

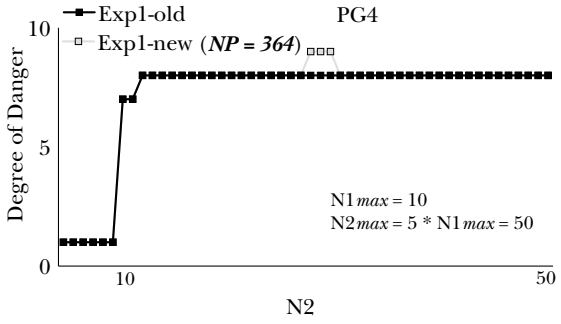

Figure 16. Results of Exp1

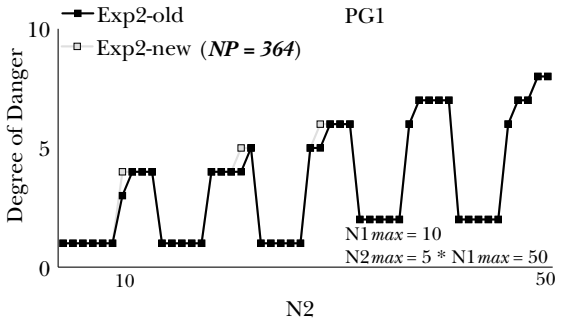

Figure 17. Results of Exp2 with reset

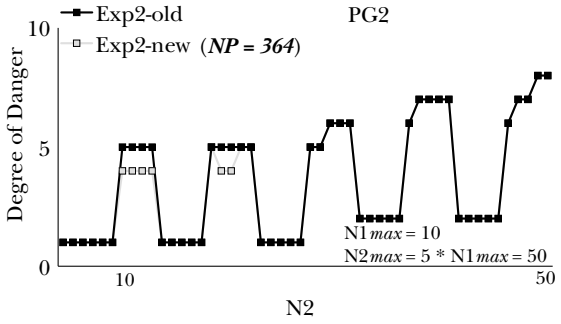

Figure 18. Results of Exp2 with reset

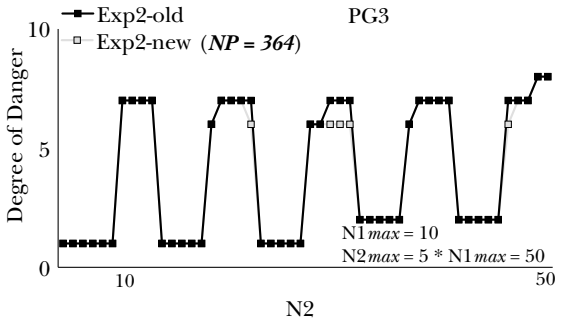

Figure 19. Results of Exp2 with reset

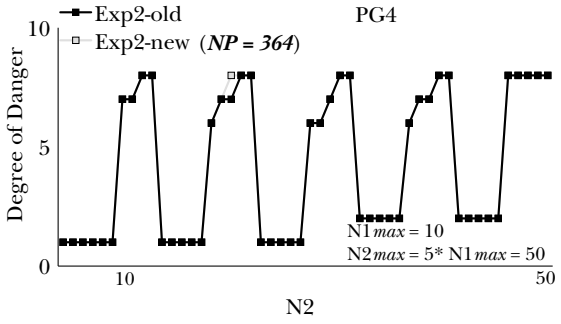

Figure 20. Results of Exp2 with reset