 ISSN: 2277-9655
(I2OR), Publication Impact Factor: 3.785

IJESRT
INTERNATIONAL JOURNAL OF ENGINEERING SCIENCES & RESEARCH TECHNOLOGY

INTELLIGENT OPTIMIZATION AND SCHEDULING OF *NETWORKED CONTROL SYSTEMS* USING *NEURAL NETWORK*

Fatin I. Telchy

ABSTRACT

This paper presents the use of Neural Networks *(NN)* for transmission time scheduling for the Networked Control System (*NCS)* where a network is widely used to connect sensors and actuators to the control systems. The need to respect typical timing constraints of the applications supported in these systems requires suitable scheduling strategies in order to devise an appropriate sequence for transmission of the information produced by the processes using the communication system.

The proposed model for *NCS* scheduling assesses its computational complexity, pointing out the drastic reduction in the time needed to generate a schedule as compared with the algorithmic scheduling solutions. The applied approach allows real-time *NCS* scheduling and makes it possible for the scheduling table to adapt the changes in process control features. Finally an on-line scheduling strategy is developed based on the neural model which can achieve real-time adaptation of the scheduling table changes in the manufacturing environment.

KEYWORDS: Scheduling, Network Control System, Neural Network, Rate Monotonic Algorithm.

INTRODUCTION

Major advancements over the last decades in wired and wireless communication networks gave rise to the new paradigm of Networked Control Systems (*NCS*). Within this paradigm, sensing and actuation signals are exchanged among various parts of a single system or among many subsystems via communication networks [1]. With the development of *NCS*, more and more researchers focus on the scheduling of network to realize the cooperation between network bandwidth requirement and control performance and can improve the Quality of Service *(QoS)* of network and reduce the chance of collision and congestion in network, then it can reduce the network induced time delay and the rate of data packet loss, so scheduling has great signification on improving the performances of NCS [2]. The most important part of network scheduling issue is how often a plant should be scheduled to transmit the data and with what priority the packet should be sent out regardless how the packet gets to the destination from the source efficiently, and what to do if the route is congested, these problems are up to the routing algorithms and congestion control algorithms [3].

The use of the communication network in the feedback control systems (wherein the control loops are closed through a real-time network) makes the analysis and design of *NCS* complex. Scheduling of the network tasks has to be involved when a set of *NCSs* are connected to the network which competes for network bandwidth [4].

The problem of network scheduling of *NCS* is finding an optimal/feasible schedule that can minimize a given performance measure. Network scheduling in *NCSs* is comparable to *CPU* scheduling in hard real time computing systems, where a set of concurrent *CPU* tasks are executed on a single *CPU* with timing constraints. Both cases involve allocating a shared resource to a set of a concurrent tasks; both involve frequent invocations of concurrent tasks, and both tasks have real time constraints and have deadlines to be met. However, in the case of network scheduling in *NCS*, the shared resource becomes the network instead of the *CPU* processor, and the execution of a real time task has been replaced by the transmission of a data packet [5].

Many contributions have been accomplished in this field; in Zhang (2001) [6] considered the scheduling of a set of controls system when their feedback control loops are closed through a communication network using Rate Monotonic Scheduling *(RMS)* algorithm. The optimal scheduling with *RMS* schedulability constraints with *NCS* stability constraints had been considered, Branicky et. al. (2002) [4] applied *RMS* algorithm for optimal scheduling of set of *NCSs*. They worked on scheduling when a set of NCSs are connected to the network and arbitrating for network bandwidth. They formulated the optimal scheduling

ISSN: 2277-9655
(I2OR), Publication Impact Factor: 3.785

problem under both *RMS* schedulability constraints and *NCS*-stability constraints using Sequential Quadratic Programming *(SQP)* optimization algorithm, Lin et. al (2009) [7] worked on co-design of scheduling and control of *NCSs*. The sampling periods are scheduled for multiple-control loops of *NCSs* depending on TrueTime toolbox and non-preemptive *RMS* algorithm. It was found that *NCS* scheduling enhances the performance of control systems, but also improves the network efficiency, and Jie and Wei-dong (2011) [2] worked on control and scheduling co-design of *NCSs* by approximate response-time analysis under fixed-priority scheduling to improve the control performance of *NCS* and enhance utilization rate of network resource.

The Proposed Intelligent *NFS*

The proposed Neural Feedback Scheduler *(NFS)* technique consist of two intelligent stages: The first stage produces optimal sampling periods by using Feedforward Neural Network (*FFNN*) named as Neural Network Optimizer *(NNO)* which replaces traditional optimization algorithm. The second stage of the *NFS*, schedules the *NCS* tasks using another Feedforward Neural Network (*FFNN*) named as Neural Network Scheduler *(NNS)*, which works online and replaces the traditional offline *RMS* algorithm. This leads to improvement in the overhead (optimize the required time for a task to be completed) and computational complexity.

The developed framework of the intelligent *NFS* is shown in Figure 1. The highlighted block illustrates the proposed technique which effectively provides high efficiency and low overhead with respect to the convenient applied methods as can be seen in [4, 7, 2].

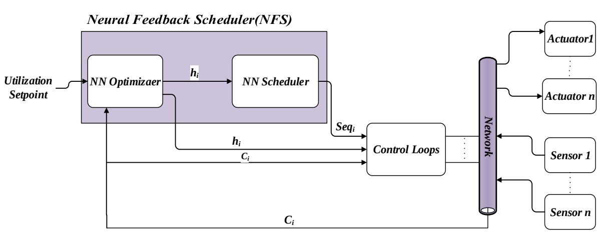

Figure 1: Neural Feedback Scheduler "NFS"

Optimized Sampling Periods

Liu and Layland [8], showed that *RMS* is optimal among all fixed priority assignments in the sense that no other fixed priority algorithm can schedule a task set that cannot be scheduled by *RMS*. Accordingly, *RMS* has been chosen as scheduling method for *NCS*, and to be developed to overcome the issues of finding an optimized sampling periods and overhead issue, i.e. develop the system performance by employing an intelligent technique.

The performance measure function of the *NCS* is associated with the control cost function $J_i(h_i)$, as function of transmission period (h_i), the selection of the performance measure function is crucial in the optimization problem. It directly relates the control cost to the *NCS* transmission period h_i [6].

The formulation of the optimization problem is [6]:

$$minimize \quad J=\sum_{i=1}^{n} J_i\,(h_i) \qquad (1)$$

Subjected to:

a) RMS Algorithm schedulability constraints:

$$h_1 \le h_2 \le h_3 \qquad (2)$$

$$\frac{C_1}{h_1}+\cdots+\frac{C_i}{h_i}+\frac{b_{l,i}}{h_i}\le i\left(2^{\frac{1}{i}}-1\right), \quad i=1,\dots,n \qquad (3)$$

b) And to: *NCS* stability constraints:

$$h_i \le \frac{h_{bw}}{20}-2\tau_i, \quad i=1,\dots,n \qquad (4)$$

$$h_i \le h_{true,i}-\overline{b_l}, \quad i=1,\dots,n \qquad (5)$$

where: $$\overline{b_{l,\iota}} \le \max_{j=i+1} C_j \qquad (6)$$

The worst-case blocking time of each *NCS* transmission, b_l, needs to be taken into account when considering h_i. The minimization process is carried out using *MATLAB 2012* function *fmincon*, which finds the minimum constrained of a nonlinear multivariable scalar function starting at an initial estimate. This is generally referred to as constrained nonlinear optimization or nonlinear programming.

According to previously described equations (1-6), the implemented algorithm will be applied as illustrated in the block diagram of Figure 2.

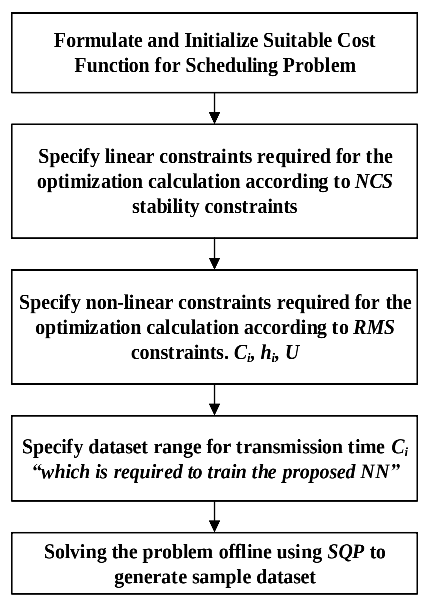

Figure 2: Block Diagram for Offline Creating Optimized Sampling Periods' Dataset

Intelligent Optimization of Sampling Period *"NNO"*

ISSN: 2277-9655
(I2OR), Publication Impact Factor: 3.785

In this section an intelligent technique will be developed based on *FFNN*. It will be named Neural Network Optimizer *(NNO)* to replace the traditional optimization method of *SQP* that applied to obtain optimal sampling period.

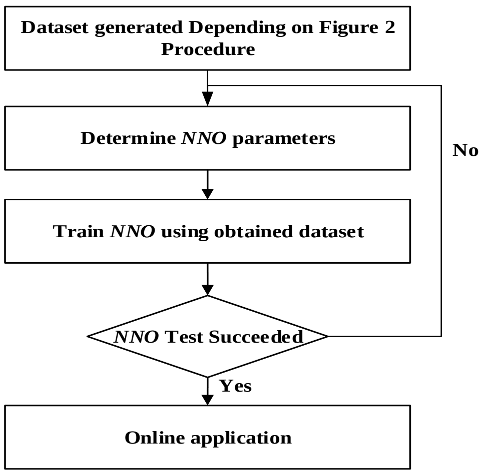

Figure 3: Neural Network Optimizer Procedure

In traditional applied scheduling methods, an optimal sampling periods is usually obtained offline by using non-linear optimization method to get a value that is suitable for the system stability constraint and *RMS* constraints and this repeated in each feedback iteration. But in the proposed method, the required dataset for *NNO* training is obtained offline for one time by using traditional optimization method (*SQP*), then suitable *NNO* has been carefully chosen to be trained based on the previously obtained dataset, later on, the obtained *NNO* can be used online as adaptive standalone unit to obtain the optimal sampling period as shown in Figure 3.
Figure 4 shows the proposed *NNO*, there is only one hidden layer apart from the input and output layers in the *NNO*. Since *FFNN* with only one hidden layer are able to approximate arbitrary functions with arbitrary precision that are continuous on closed intervals, one hidden layer is sufficient for guaranteeing solution accuracy [9].

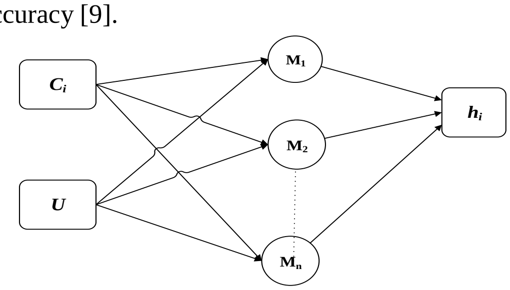

Figure 4: The Developed NNO Structure

The inputs for the *NNO* is the Transmission Time *(C)* and the Utilization *(U)* which processed by the hidden layer and iteratively produced the Sampling Periods *(h)*, using this intelligent technique will overcome complexity raped application of the optimization.

Intelligent Scheduling by Neural Network *"NNS"*

When a set of Control System *(CS)* plants are connected to the network and arbitrate for network bandwidth, based on priority scheduling algorithm such as *RMS* algorithm, a "faster" plant (i.e., requiring higher transmission rate) is given higher priority over a slower plant. The *RMS* algorithm can be implemented on priority-based networks, such as Controller Area Network (*CAN*) and DeviceNet, where the priority of the message can be incorporated into the message identifier [6].
In this work, another intelligent technique will be developed using *FFNN*. It will be named Neural Network Scheduler (*NNS*) to replace traditional *RMS* algorithm to schedule *NCS* tasks.
In *RMS* algorithm, transmission time *(C)*, sampling period *(h)*, and tasks priority *(P)* are required for offline scheduling *NCS* packet transmission.
Accordingly the required dataset has been manually obtained offline by applying *RMS* on set of tasks to prepare dataset which is used later as training data for *NNS*. The priority condition depended on the sampling period $h_1 < h_2 < h_3$. Suitable *NNS* will be chosen with proper number of neurons, based on previous obtained dataset for *NNS* training. In turn *NNS* can be used online as adaptive standalone unit to schedule the *NCS* tasks as shown in Figure ***5***.

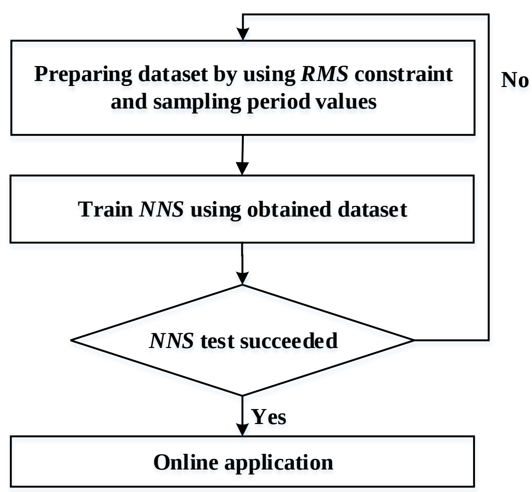

Figure 5: The Developed NNS Technique

The *NNS* structure consist of only one hidden layer apart from the input and output layers, as shown in

ISSN: 2277-9655
(I2OR), Publication Impact Factor: 3.785

Figure 6. The sampling period *(h)* forms the *NNS*'s input and the scheduling sequence *(Seq.)* forms the output.

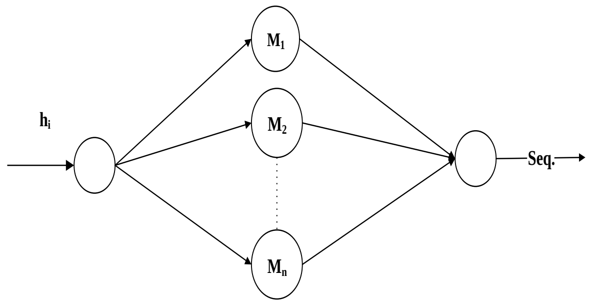

Figure 6: The Developed NNS Technique.

The inputs for the *NNS* is *h* which processed by the hidden layer and iteratively produced task Sequence (*Seq*), using this intelligent technique will schedule the *NCS* tasks instantaneously.
In order to determine the suitable number of hidden neurons, i.e. the value of *M*, neural networks of different sizes (4, 8, and 16) has been compared and it was found that the most applicable one was with *M=8*, as the performance and Gradient values equals 5.49×10^{-7} and 7.64×10^{-7} respectively.
The *FFNN* has been used to replace the *RMS* method to scheduling periodic tasks for this case, the proposed *NN* has been trained by using the last obtained optimal sampling period and the scheduling constraints to determine the priority $h_1<h_2<h_3$, then the proposed *NN* employed to work online and stand alone to schedule *NCS* tasks.

Verification of *NN* Scheduling Techniques for *NCS*

The power and effectiveness of the developed Neural Network techniques have been examined through applications to solve *NCS* problems, the following examples illustrate both creating optimal sampling periods using *SQP* optimization method and the *NNO* followed by *NNS*.
The illustrated examples present the effectivity of the developed *NFS* by employing different cost functions, constraints, and transmission times.

Example 1: Linear Cost Function with Constant Transmission Time

A set of scalar plants have been considered and represented by the state space equations are shown below and the systems properties are shown in Table 1 [2].

$\dot{x}_1=20x_1+u_1,\qquad u_1=-40x_1,$
$\dot{x}_2=15x_2+u_2,\qquad u_1=-35x_2$
$\dot{x}_3=10x_3+u_3,\qquad u_1=-30x_3$

so, the closed loop system will be; Ā=-20. Let the tasks transmission times are known $C_1=C_2=C_3=0.004s$, also the priorities are given $P_1=1$, $P_2=2$, *and* $P_3=3$, $h_{wb}=900ms$. The goal of the design is to assign task periods such that the overall system cost is minimized. The overall cost *J* is defined as [2]:

$$J_{all}=\frac{3+\sqrt{3}}{6}(p_1h_1+p_2h_2+p_3h_3)+p_1\tau_1+p_2\tau_2+p_3\tau_3$$

where p_i is weight coefficient, corresponding to the priority of control system. The greater value, the higher priority of the corresponding control system. And J_i is performance index function of each control loop. τ is the input-output latency, and h is the sampling interval. The optimal sampling period can be estimated by the optimization analysis and the previous function which is minimized subject to utilization constraint in Equations (3 and 4):

Table 1: Information Data for Examples 1

	CS_1	CS_2	CS_3
C*	4	4	4
$\boldsymbol{b_{l,i}}$ *	4	8	8
τ_i *	4	8	8
h_{bw} *	900	900	900
Linear Constraint* $h_i\leq$	37	29	29
Initial h_0*	16.36	9.65	15.58

**The measure unit for time is miliseconds*

In order to create the required dataset for *NNO's* training process, transmission times C_1, C_2, and C_3 were selected from 1*ms* to 10*ms* with increments of 1*ms*. For all possible values of these parameters, applying *SQP* to solve the cost equation offline results in totally 1000 sets of sample data.

Simulation Results Using NNO

In order to determine the number of hidden neurons, i.e. the value of *M*, *FFNN* of different sizes (2,4,6,8,12,16, and 20) has been compared and the most applicable one was with *M=20*, as the performance and Gradient values equals 4.71×10^{-7} and 9.89×10^{-6} respectively. Given that the performance is comparable. From this perspective, it is set that *M = 20* because of the good performance of corresponding neural network and the fast reaching to the required results.
Several number of neurons *(M)* has been tested to reach to the best *NN* performance, the performance and the gradient form the most important keys to evaluate the best *NN* structure, the best two choices of many attempts was 16 and 20 Neuron and according to below reasons 20 Neuron has been chosen due to the minimum performance value (4.71×10^{-7}), Gradient (9.89×10^{-6}), Mu (1×10^{-6}) which are lower than appears in 16 neurons *NN*.

A comparison has been made between the optimal sampling periods gained from regular *SQP* method and between the proposed *FFNN* are shown in Table 2, also schedulability test has been performed on the neural networks results to approve the effectivity of the proposed technique as shown from the calculations below. And as seen from the results all the selected transmission periods are schedulable using *RMS* algorithm.

Table 2: Results from Traditional SQP and NNO

	h_1 (ms)	h_2 (ms)	h_3 (ms)	Min J	U <0.7798	Overhead (s)
SQP	0.0213	0.0150	0.0123	40.0695	0.7797	0.6334
NNO	0.0209	0.0156	0.0125	40.0707	0.7678	0.011036

The above results show that overhead improved in *NNO* by 98.26% than *SQP* and utilization improved by 1.54%, although the *J* value of *NNO* is greater than *SQP*, but *NN* gave schedulable *NCS* tasks while *SQP* gave tightly schedulable tasks.

NNS to Schedule Transmission Time

After the training process, the proposed *NNS* is ready to do scheduling task, Figure 7 shows scheduling of Example 1 tasks', where sampling periods are $h_1=0.0212$, $h_2=0.0153$, $h_3=0.0124$, and transmission time $C_1=C_2=C_3=0.004$, where h_3 is higher priority as it has smaller sampling period and h_1 is the lower priority. It is clear that *NNS* could schedule the three tasks according to *RMS* conditions which bring new intelligent technique that minimize the overhead and use low memory.

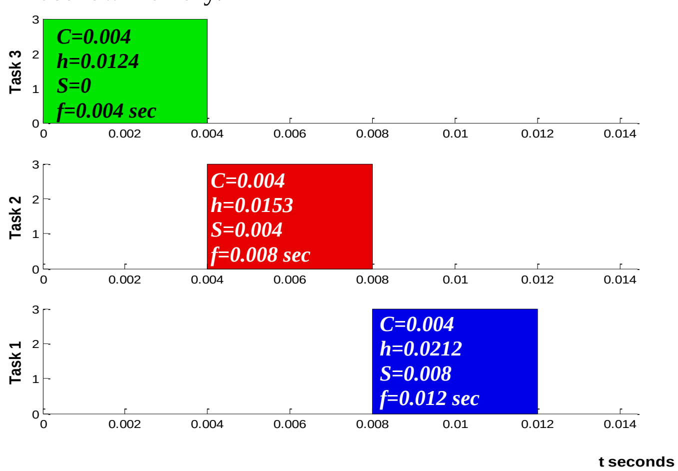

Figure 7: Scheduling Results by NNS

Example 2: Exponential cost function with constant transmission time

To evaluate the performance of *CS* with their feedback control loops which are closed through a communication network, a set of scalar plants has been considered, represented by $\dot{x}=Ax+Bu$, with $A = 25, 20, 5$, $B=1$, and $K = 50,45, 30$, respectively. The state space equations of the systems are shown below and the systems information is listed in Table 3 [4].

$\dot{x}_1=25x_1+u_1,\quad u_1=-50x_1,$
$\dot{x}_2=20x_2+u_2,\quad u_1=-45x_2$
$\dot{x}_3=5x_3+u_3,\quad u_1=-30x_3$

So, the closed loop system will be $\bar{A}$=-25. Note that all three *NCSs* have the same closed-loop performance which represent that the closed loop control systems are stable, below performance function has been implemented to optimize the sampling period [4]:

$$\min J(h) = e^{25*h_1} + 1.25e^{20*h_2} + 5e^{5*h_3}$$

Equations 3 & 5 have been used as *RMS* for *NCS* stability constraints. The upper bounds on these plants' transmission periods that preserves their stability, $h_{true,i}$, can be calculated as [6]:

$$\boldsymbol{h_{true} = \frac{1}{A} ln \frac{\frac{K}{A}+1}{\frac{K}{A}-1}}$$

Table 3: Information Data for Example 2

	CS_1	CS_2	CS_3
A	25	20	5
K	50	45	30
C*	0.004	0.004	0.004
$\bar{b_i}$*	0.004	0.004	0
h_{true}*	0.0439	0.0478	0.0673
Linear Constraint h_i*≤	0.0399	0.0438	0.0673
Initial h_0*	0.026	0.03	0.034

**The measure unit for time is seconds*

As the transmission time according to DeviceNet specification is *4 ms* [12], so, for the purpose of creating sample dataset for *NNO* training process, the ranges of C_1, C_2, C_3 has been specified from *1ms* to *10ms* with increments of *1ms*, applying *SQP* to solve the cost equation offline results in totally 1000 sets of sample data.

Simulation Results Using NNO

Same procedure steps for Example 1 has been used for this case to get efficient *NNO* with 3 layers (one input layer, one hidden layer with 20 neuron, one output layer).

A comparison has been made between the optimal sampling periods gained from regular *SQP* method and between the proposed *NNO* Table 4, also schedulability test has been performed on the neural networks results to approve the effectivity of the proposed techniques as shown from the calculation below. By applying *NNO* with *M=20*, the following analysis have been obtained:

Table 4: Results from Traditional SQP and NNO

	h_1 (ms)	h_2 (ms)	h_3 (ms)	Min J	U <0.7798	Overhead (s)
SQP	**0.0146**	**0.0150**	**0.0167**	**8.5639**	**0.7802**	**1.2491**
NNO	**0.0146**	**0.0150**	**0.0168**	**8.5646**	**0.7787**	**0.009785**

The above results show that overhead improved in *NN* by 99.22% than *SQP* and utilization improved by 0.14%, although the *J* value of *FFNN* is greater than *SQP*, but *FFNN* gave schedulable *NCS* tasks while *SQP* gave tightly schedulable tasks as clear from the above utilization values.
As seen from the above results all the selected transmission periods are schedulable according to *RMS* algorithm constraints.

Adapting Neural Network for Task Scheduling by NNS

After training *NNS* is ready process to do scheduling task, Figure 8 shows scheduling of Example 2 tasks, where sampling periods are $h_1=0.0146$, $h_2=0.0150$, $h_3=0.0168$, and transmission time $C_1=C_2=C_3=0.004$, where h_1 is higher priority as it has smaller sampling period and h_3 is the lower priority.

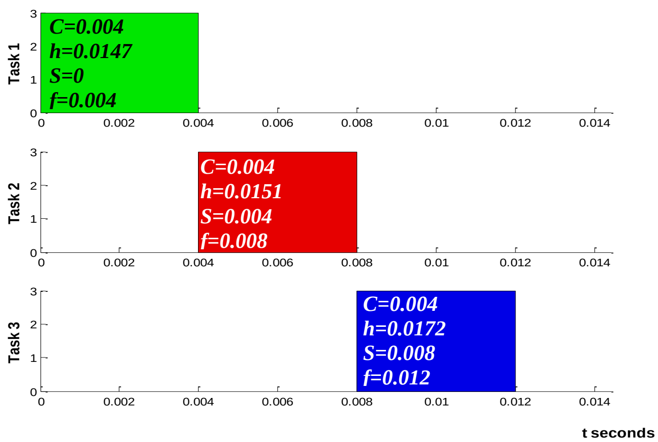

Figure 8: Scheduling Results by NNS

Example 3:Linear cost function with variable transmission time

The assumed design goal for this example is to select sampling periods h_1, h_2,... h_i such that a weighted sum of the cost function:

$$J_{all} = \frac{3+\sqrt{3}}{6}(p_1 h_1 + p_2 h_2 + p_3 h_3) + p_1\tau_1 + p_2\tau_2 + p_3\tau_3$$

The *NCS* stability constraints [7] is $h_i \leq h_{bw} - 2\tau_i$, $i=1, \ldots, n$, where h_{bw} is the received by the control system's bandwidth, which is assumed equal to *800ms*. The full Example 3 data information is shown in Table 5.

Table 5: Information Data for Example 3

	CS_1	CS_2	CS_3
C^*	2	5	5
p_i	1	2	3
$\boldsymbol{b_{l,i}}^*$	5	5	0
τ_i^*	3	4	6
h_{bw}^*	800	800	800
Linear Constraint $h_i^* \leq$	34	32	28
Initial h_o^*	7	31	28

**The measure unit for time is miliseconds*

The optimal sampling period can be estimated by the optimization analysis and the previous function which is minimized subject to below utilization constraint as in Equation 3.

Simulation Results Using *NNO*

A comparison has been made between the optimal sampling periods gained from regular *SQP* method and between the *NNO* Table 6, also schedulability test has been performed on the *NNO* results to approve the effectivity of the proposed technique as shown from the calculation below.

Table 6: Results from Traditional SQP and NNO

	h_1 (ms)	h_2 (ms)	h_3 (ms)	Min J	U <0.7798	Overhead (s)
SQP	**0.0153**	**0.0171**	**0.0140**	**29.0722**	**0.7803**	**0.7204**
NNO	**0.0156**	**0.0176**	**0.0142**	**29.0736**	**0.7644**	**0.009602**

The above results show that overhead improved in *NN* by 98.67% than *SQP* and utilization improved by 1.97%, although the *J* value *of NNO* is greater than *SQP*, but *NN* gave schedulable *NCS* tasks while *SQP* gave tightly schedulable tasks.

NNS to Schedule Tasks

In order to determine the suitable number of hidden neurons, i.e. the value of *M*, neural networks of different sizes (4, 8, and 16) has been compared and it was found that the most applicable one was with $M=8$, as the performance and Gradient values equals 1.91×10^{-7} and 1.54×10^{-6} respectively. Also it can be seen that h_3 is higher priority as it has smaller sampling period and h_2 is the lower priority. The *NCS* results is shown in Figure 9.

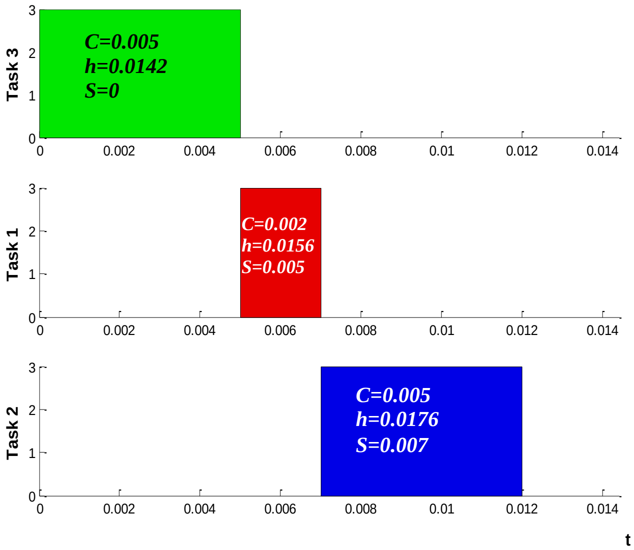

Figure 9: Scheduling Results by NNS

CONCLUSION

This paper presents the neural model for scheduling process and assesses its computational complexity, pointing out the drastic reduction in the time needed to generate a schedule as compared with the algorithmic scheduling solution. New scheduling technique has been proposed as on-line scheduling strategy based on the neural model which can achieve real-time adaptation of the *RMS* algorithm.

The traditional scheduling methods complicates the problem of scheduling as it requires the use of computationally complex algorithms to optimize sampling periods.

In this work, an alternative approach to scheduling based on a Feedforward Neural Network model and show how it overcomes the problem of the computational complexity of the algorithmic solution. It can be noticed that the results of 98.26% , 99.22%, and 98.67% in overhead improving for mentioned cases by using *NNO* with respect to traditional *SQP*, in addition to the improvement of traditional *RMS* algorithm by using *NNS* which act as stand-alone technique for *NCS* scheduling tasks.

The developed optimization and scheduling provide more flexibility, minimum overload and lower utilization than the traditional methods.

REFERENCES

[1] ETH Zurich and Peter Al Hokayem. (2010, April) Networked Control Systems Group. http://control.ee.ethz.ch/~ncs/

[2] WANG Jie and LIU Wei-dong, "Control and Scheduling Co-Design of Networked Control System," in *Signal Processing, Communications and Computing (ICSPCC), IEEE International Conference on*, Xi'an, 2011, pp. 1-4.

[3] Pedro Macedo and José A. Afonso, "Simulation Analysis of IEEE 802.15.4 for Wireless Networked Control Systems," in *Industrial Electronics, 2009. IECON '09. 35th Annual Conference of IEEE*, Porto, 3-5 Nov. 2009, pp. 2482 - 2487.

[4] Michael S. Branicky, Stephen M. Phillips, and Wei Zhang, "Scheduling and Feedback Co-Design for Networked Control Systems," in *Decision and Control, 2002, Proceedings of the 41st IEEE Conference on (Volume:2)*, 10-13 Dec. 2002, pp. 1211 - 1217.

[5] Dr. Kang Li, "Bio-Inspired Computational Intelligence and Applications," in *International Conference on Life System Modeling, and Simulation*, Shanghai, 2007, pp. 14-17.

[6] Wei Zhang, Stability Analysis of Networked Control Systems, Augest 2001.

[7] Tao Lin, Yao-Han Ran, and Jie-Hua Liu, "Research of Scheduling and Control Co-Design of Networked Control Systems," in *Second International Conference on Intelligent Networks and Intelligent Systems*, Tianjin, China, 1-3 Nov. 2009, pp. 201 - 204.

[8] C.L. Liu and J.W. Layland, "Scheduling Algorithm for multiprogramming in a Hard-Real-Time Environment," *J.ACM*, vol. 20(1), pp. 46-61, January 1973.

[9] Feng Xia, Yu-Chu Tian, Youxian Sun, and Jinxiang Dong, "Neural Feedback Scheduling Of Real-Time Control Tasks," *International Journal of Innovative Computing, Information and Control, ICIC*, vol. 4, no. 11, pp. 2965–2975, December 2007.

[10] Emil Michta. (2005, July) Scheduling Systems. http://eu.wiley.com/legacy/wileychi/hbmsd/pdfs/mm823.pdf

[11] Lui Sha, Ragunathan Rajkumar, and Shirish S. Sathaye, "Generalized Rate-Monotonic Scheduling Theory: A Framework for Developing Real-Time Systems," *Proceesings of The IEEE*, vol. 82, no. 1, pp. 68-82, January 1994.

[12] Richard McLaughlin and Eva D'souza. (2002 , May) CAN/EtherNet Research Centre. [Online]. HYPERLINK

ISSN: 2277-9655
(I2OR), Publication Impact Factor: 3.785

"http://www.webring.org/l/rd?ring=cancontrollerare;id=1;url=http%3A%2F%2Fweb%2Ewarwick%2Eac%2Euk%2Fdevicenet%2F"
http://www.webring.org/l/rd?ring=cancontrollerare;id=1;url=http%3A%2F%2Fweb%2Ewarwick%2Eac%2Euk%2Fdevicenet%2F