Design of Auto-Tuning PID Controller Methods Based on Genetic Algorithms for LR-PONs

Tamara Jiménez, Noemí Merayo, Juan C. Aguado, Ramón J. Durán, Ignacio de Miguel, Patricia Fernández, Rubén M. Lorenzo, Evaristo J. Abril

Optical Communications Group of the Department of Signal Theory, Communications and Telematic Engineering, E.T.S.I. Telecommunication, University of Valladolid, Valladolid, Spain

e-mail: tamara.jimenez@tel.uva.es, noemer@tel.uva.es, jaguado@tel.uva.es, rduran@tel.uva.es, ignacio.miguel@tel.uva.es, patfer@tel.uva.es, rublor@tel.uva.es, ejad@tel.uva.es

***Abstract*—In this paper, a new method to automate the tuning process of PID controllers is presented. The designed method, based on genetic algorithms, tunes PID systems that control the QoS requirements in Long-Reach PONs. This new tuning technique has been compared with the manual Ziegler-Nichols frequency response method. The simulation results have demonstrated that the new technique efficiently automates the tuning process, which leads to a reduction in the tuning time and a higher accuracy.**

Keywords- Proportional-Integral-Derivative (PID); Passive Optical Network (PON); tuning process; Dynamic Bandwidth Allocation (DBA); Class of Service (CoS); Service Level Agreement (SLA); bandwidth guarantees; delay guarantees.

I. INTRODUCTION

Passive Optical Networks (PONs) and Long-Reach Passive Optical Networks (LR-PONs) are the most preferable networks infrastructures in the today's access deployment [1]. In fact, the number of PON subscribers in Asia Pacific remains 80 million subscribers by the end of 2012, whereas in America this number is 11 million and in Europe near 16 million users [2]. However, in Europe, 41.5 million households are expected to be biber access subscribers by means of PON infrastructures at end of 2017. On the other hand, current access networks have to deal with different kind of users which contract a Service Level Agreement (SLA) with a provider and different kind of Class of Services (CoS) with different priorities. Therefore, users are guaranteed some network requirements, typically related to a minimum bandwidth level or a maximum delay for high priority CoS. Consequently, it is highly necessary that Dynamic Bandwidth Allocation (DBA) algorithms cover one or both premises. Even more, it is quite suitable that algorithms comply with the network requirements by means of a real time and automatic readjustment. Some algorithms take into account both objectives in a very efficient way [3]-[6]. Hence, one typical way to guarantee bandwidth or delay bounds to different priority subscribers is using fixed weights assigned to each ONU according to its SLA. Hence, ONUs that belong to a higher priority SLA, are assigned a larger weight, so they are given more bandwidth [7][8]. However, fixed factors do not adapt the PON performance to different traffic patterns or network conditions, so if service providers do not properly adjust the initial weights, the network should automatically evolve to the requirements established by the service provider. Thus, it is essential that the network becomes independent of the initial weights or conditions. Therefore, algorithms based on Proportional-Integral-Derivative (PID) controllers are able to robustly and efficiently manage the allocated bandwidth to comply with different guaranteed bandwidth levels [5] or maximum delay requirements [6]. Indeed, these algorithms based on PID controllers have demonstrated better performance than other existing algorithms that control these network parameters (bandwidth, delay) in PON networks [5][6]. This kind of controllers is extensively used due to its simple structure, robustness and good performance [9][10]. In connection with this type of control, PIDs require a tuning process in order to achieve a reliable response according to the established objectives. However, in contrast to previous existing algorithms based on PIDs to control network parameters in PON infrastructures [5][6], which use the well-known Ziegler-Nichols frequency response method, we propose to tune the PID controller using a Genetic Algorithm (GA). Although, the Ziegler-Nichols frequency response method is a very widespread technique, it is a manual method based on experiments. Thus, this manual nature may convert it into a very time-consuming and tedious technique. Contrary, the use of GA allows an automatic and fine tuning process, with less tuning time and better accuracy than manual techniques.

Therefore, in this paper a GA is developed to tune the PIDs of the previous developed algorithms [5] and [6] to automatize the tuning process and to improve their performance when controlling the bandwidth and the delay network parameters. The rest of this paper is organized as follows. Section II describes the genetic algorithms developed to auto-tune PID controllers to control network parameters in LR-PONs. Section III presents the simulation results and the discussion. Section IV addresses the conclusions of the paper.

II. GENETIC ALGORITHMS TO AUTO-TUNING PID CONTROLLERS TO CONTROL NETWOK PARAMETERS

A. *Genetic algorithm to tune a PID to control the bandwith allocation in PONs*

A PID controller is designed to keep the value of a variable close to a desired value [11]. Therefore, the PID calculates the error, defined as the difference between the current value of the variable and its reference value. According to this committed error ($e[n]$), it calculates the

 ISBN: 978-1-61208-349-0

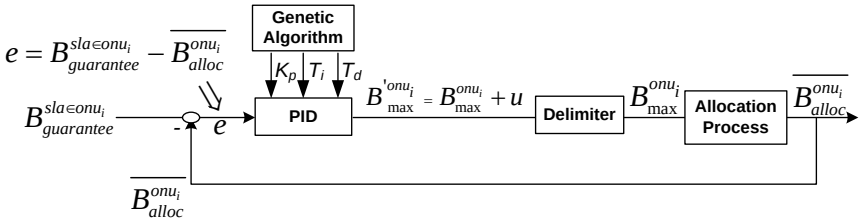

Figure 1. Block diagram of the proposed PID controler tuned with a GA to ensure guranteed bandwidth levels to differnet priority profiles.

control signal $u[n]$ following (1), which is the equation that models the PID in the discrete time. It is composed by three terms, the proportional term manages the current error, the integral one regards the accumulation of past errors, and the derivative term makes a prediction of future errors [11].

$$u[n] = K_p \cdot e[n] + K_p \cdot \frac{T_{sample}}{T_i} \sum_{m=0}^{n} e[m] + K_p \cdot \frac{T_d}{T_{sample}} (e[n] - e[n-1]) \quad (1)$$

The first proposed algorithm, called Genetic Algorithm Service level agreement PID (GA-SPID), keeps the mean allocated bandwidth of each ONU ($\overline{B^{onu_i}_{alloc}}$) close to its guaranteed bandwidth, which depends on its contracted SLA ($B^{sla \in onu_i}_{guarantee}$). GA-SPID assigns bandwidth to each Optical Network Unit (ONU) at every cycle, $B^{onu_i}_{alloc}$, using a polling policy with a limited scheme, defined as $B^{onu_i}_{alloc} = Minimum\{B^{onu_i}_{demand}, B^{onu_i}_{max}\}$, where $B^{onu_i}_{demand}$ is the bandwidth demanded by ONU *i* in one cycle (in bytes). To include the control of the PID in the bandwidth allocation ($B^{onu_i}_{alloc}$), the term $B^{onu_i}_{max}$ is updated by adding the control signal. Therefore, the maximum bandwidth allowed to each ONU, $B^{onu_i}_{max}$, is dynamically updated depending on the committed error, that is, the difference between the mean allocated bandwidth and the required bandwidth $e[n] = B^{sla \in onu_i}_{guarantee} - \overline{B^{onu_i}_{alloc}}$. In case that one ONU demands less bandwidth than its guarantee value, the PID only offers its demand because the remaining bandwidth up to its guarantee level will be unused by the ONU and the EPON performance could become inefficient. Finally, the system includes a delimiter which reduces the maximums proportionally to the ones calculated by the controller, to fit in the maximum cycle time of the Ethernet PON (EPON) standard (2 ms). Fig. 1 shows the block diagram of the proposed PID for the bandwidth assignment (GA-SPID).

On the other hand, the parameters K_p, T_i and T_d of (1) have to be tuned so that the control system will be stable and meet the established objectives. Among the existing techniques, the frequency response method proposed by Ziegler-Nichols [9][11] has become an easy and very high spread technique, especially when a mathematical model is not available, as in our system. It gives simple and experimental rules by only considering the proportional response ($T_i = \infty, T_d = 0$) and then the gain is increased until the process begins to oscillate. When this happens, the gain is defined as the ultimate gain (K_u) and the oscilation period is defined as the ultimate period (T_u). With both variables, it is possible to obtain K_p, T_i and T_d following a simple relation [11]. This method has been previously used in [5] to tune the PID for the bandwidth allocation process in a PON network. However, it can be noticed that it is a manual technique that sometimes may become a time-consuming and laborious method if the selected values are quite far of the suitable ones. Therefore, we propose an automatic method based on genetic algorithms to select the tuning parameters in a very efficient way. This method, as well as the Ziegler-Nichols method, is an offline tuning method, carried out just before the PON activates the PID which controls the network parameters. Indeed, genetic algorithms, which are efficient searching techniques used to optmize parameters and processes, have been included in the tuning process of PID controllers in many fields. In the literature, there are some proposals in different chemical and industrial applications [12][13]. In the Telecom field, it could be found one genetic algorithm that tunes a PID controller with the aim to improve the network utilization [14].

In order to design the novel tuning method, the main steps of genetic algorithms were followed. The first step regards the definition of the chromosome. In our system, each chromosome consists of the three tuning parameters (K_p, T_i, T_d) coded in a binary chain (16 bits per parameter), since this type of codification improves the efficiency of the genetic algorithm for this application [12]-[14] (Fig. 2).

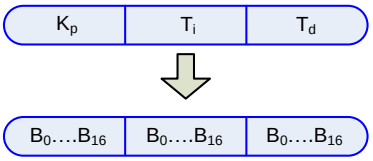

Figure 2. Appearance of a chormosome of the tuning parameters

After that, a random initial population is created. For the specific application of tuning a PID, the number of individuals that compose the population may become critical due to the strong dependence between the population size and the tuning time. Indeed, a low population size leads to a fast evolution of the algorithm towards the optimum tuning values. The next step consists of evaluating the fitness of each member of the population. Since the objective of the algorithm is to minimize the error between the desired output and the one obtained, to calculate the fitness of one specific individual, an objective function based on the committed error of the PID using this individual (K_p, T_i, T_d), during *m* iterations of the PID has

 ISBN: 978-1-61208-349-0

been used. Specifically, the objective function for one individual is defined according to (2), where N_{onus} is the number of ONUs in the PON and $e_i[m]$ is the error committed by ONU i in the m iteration of the PID.

$$F = \frac{1}{m} \cdot \frac{1}{N_{onus}} \sum_{m} \sum_{i=0}^{N_{onus}} \left| e_i[m] \right| \quad (2)$$

Therefore, those individuals with the lowest error (which are the fittest) have a high probability to be selected for the next iteration of the genetic algorithm, in which a new generation is created. Once the fitness evaluation of each individual is finished, the genetic algorithm checks the stop criterion. If the criterion is not satisfied, the algorithm repeats the process with the next generation. To generate a new population, the genetic algorithm selects individuals according to its fitness and it applies the crossover and mutation operators. Furthermore, in GA-SPID we have considered elitism, which means that the fittest individual in each generation is retained unchanged, so that the best solution is not lost. If the stop criterion is satisfied, then the best individual of the population is used to tune the PID. Specifically, the stop criterion selected for our proposal is to reach a maximum number of generations, which allow us to establish a fixed duration of the tuning process. A flow diagram with the steps of the genetic algorithm is shown in Fig. 3.

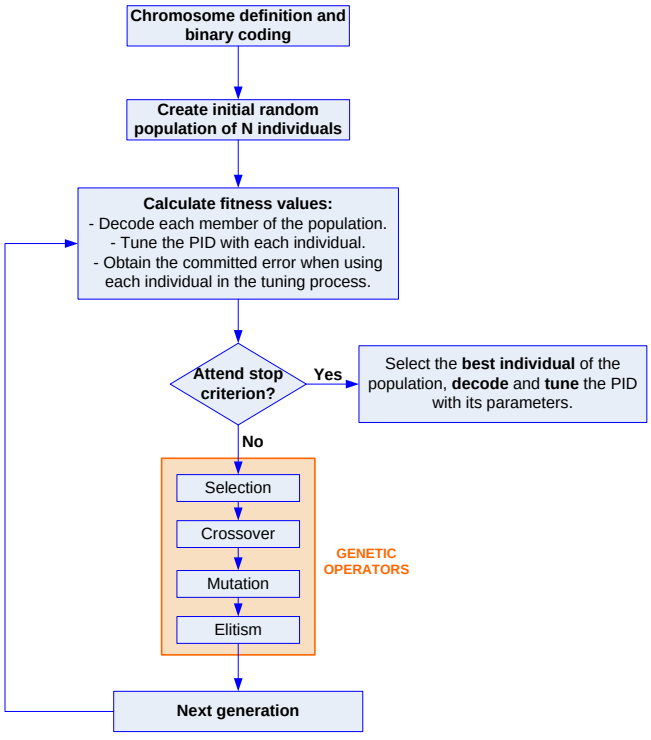

Figure 3. Flow diagram of the genetic algorithm used to tune the PID controller

B. *Genetic algorithm to tune a PID to control the mean packet delay of priority services*

The algorithm Genetic Algorithm Delay aware Service level agreement PID (GA-DaSPID) is able to control the maximum delay of different priority services by modifying the maximum bandwidth of each SLA using a simple P controller, which is a simplification of the PID controller which only considers the proportional term of (1). The block diagram of the algorithm is shown in Fig. 4. Since this algorithm controls the mean packet delay, the reference value is the maximum permitted delay stipulated by the service provider for the j classes of service with restrictive delay depending on the contracted k SLA $\left(R_{P_j}^{sla_k}\right)$. The term under control is the instantaneous mean packet delay for each j class of service of each k SLA $\left(\overline{r_{P_j}^{sla_k}[n]}\right)$. In order to calculate the instantaneous error ($e[n]$) of one ONU which belongs to the SLA k, it is necessary to carry out the sum of every individual committed error in each service j in order to guarantee the delay restrictions to each j class of service, that is, $e = \sum_{j} (R_{P_j}^{sla_k} - \overline{r_{P_j}^{sla_k}[n]})$. On the other hand, to calculate the control signal $u[n]$, (1) is applied with only the proportional term, since a P controller has demonstrated the best performance for this concrete application [6]. To obtain the new maximum permitted bandwidth ($B_{\max}^{onu_i}$), the control signal $u[n]$ is subtracted from the previous maximum permitted bandwidth. In this way, if for example the mean packet delays of all j services of ONU_i $\left(\overline{r_{P_j}^{sla_k}[n]}\right)$ are higher than their maximum packet delay $\left(R_{P_j}^{sla_k}\right)$, the error becomes negative, so the algorithm increments the maximum permitted bandwidth to allow ONU_i to decrease its mean packet delay so that it can comply with the delay restrictions. In contrast, if the mean packet delays of all j services of ONU_i $\left(\overline{r_{P_j}^{sla_k}[n]}\right)$ are lower than their maximum packet delay $\left(R_{P_j}^{sla_k}\right)$ the error becomes positive and the P controller reduces its maximum allocated bandwidth. As in the previous algorithm, the designed P controller is equipped with a delimiter (Fig. 4).

As in GA-SPID, a new method to efficiently tune the P controller by using a genetic algorithm is proposed. Since the controller only consists of the proportional term, the tuning parameters are reduced to K_p. Consequently, the chromosome is a binary code of 16 bits that represents only this parameter (K_p). Furthermore, the steps of the genetic algorithm are the same as those represented in the flow diagram of Fig. 3. Finally, the objective function of the algorithm is also (2).

III. RESULTS AND DISCUSSIONS

A. *Simulation scenario of the LR-PON*

We have designed a LR-EPON network with 16 ONUs and one user connected to each ONU using OPNET Modeler v.16 [15]. The upstream and downstream transmission rates are 1 Gbit/s whereas the transmission rate from users to each ONU is 100 Mbit/s. The distance between ONUs and the Optical Line Terminal (OLT) is 100 km. The maximum cycle time according to the EPON

 ISBN: 978-1-61208-349-0

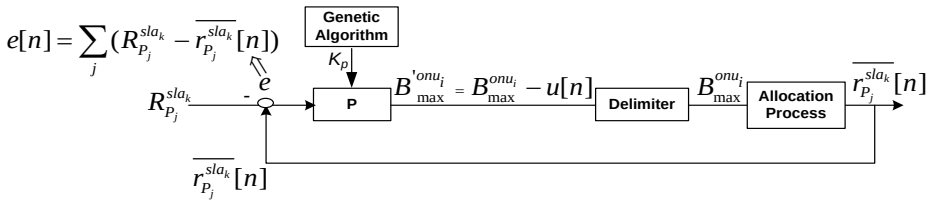

Figure 4. Block diagram of the proposed P controller tuned with a GA to control delay requirements in priority services in PONs

standard is 2 ms [16]. Traffic follows a Pareto distribution with a Hurst parameter H equal to 0.8, with variable packet length between 64 and 1500 bytes, plus the 38 bytes of the packet headers It has been assumed symmetry in the traffic load of every ONU, as in [1][3]-[8]. Furthermore, GA-SPID considers three SLAs (SLA_0, SLA_1, SLA_2) with their corresponding guaranteed bandwidth levels of 100 Mbps, 75 Mbps and 50 Mbps, to be controlled by the PID. Regarding GA-DaSPID, it takes into account three services, P_0 for the highest priority traffic (interactive), P_1 for the medium priority traffic (responsively) and P_2 for the non-critical traffic (best-effort). P_0 assumes the 20% of the total network load, and P_1 and P_2 the 40% of the total load, as in [6]. Furthermore, it considers three SLAs (SLA_0, SLA_1, SLA_2), so each delay-sensitive service (P_0, P_1) is set a different bound delay depending on the priority of the profile. In fact, Table I summarizes the delay bounds considered in GA-DaSPID, which are the same that those proposed by other algorithms [4][6].

TABLE I. DELAY BOUNDS FOR EACH CLASS OF SERVICE AND SERVICE LEVEL AGREEMENT CONSIDERED IN GA-DASPID

Class of Service	*Delay bound value*	*Applications*
P_0	1.5 ms	VoIP, videoconference, interactive games, Telnet
P_1	SLA_0: 5 ms	Voice Messaging Web-browsing HTML E-mail Transaction services
	SLA_1: 20 ms	
	SLA_2: 60 ms	
P_2	Not limited	Bulk data

The parameters related to the execution of the genetic algorithm for both algorithms are specified in Table II. These parameters have been selected by running previous simulations, and choosing those parameters which allows both, a good performance and a short tuning time. To justify the selection of these parameters, Fig. 5 represents the mean committed error (in bits) of the best individual in each generation when considering different population sizes and number of iterations in GA-SPID. As it can be observed, the error is reduced as the number of generations increases for every combination of population size and number of iterations. However, the worst performance is achieved for a population of 15 individuals and a number of iterations equal to 2. For the remaining combinations, the results are similar. Thus, a population of 20 individuals with 2 iterations is selected, since it leads to the lowest tuning time.

TABLE II. MOST IMPORTANT PARAMETERS OF THE GENETIC ALGORITHM IN GA-SPID AND GA-DASPID

Parameters of the genetic algorithm	*GA-SPID*	*GA-DaSPID*
Selection method	Roulette wheel	Roulette wheel
Threshold of tuning parameters	(0,5]	(0,5]
Cross probability	0.9	0.9
Mutation probability	0.01	0.01
Elitism	yes	yes
Population size	20 individuals	15 individuals
Stop criteria	10 generations	10 generations
Iterations of the PID to update fitnesss	2 iterations	5 iterations

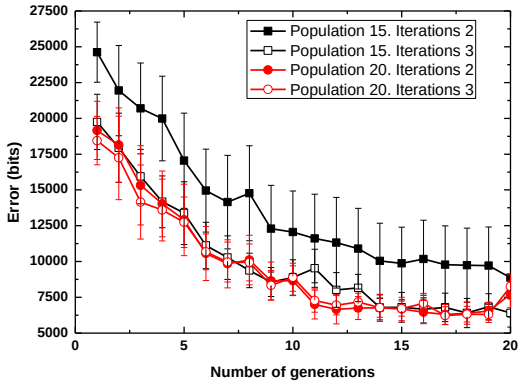

Figure 5. Evolution of the mean committed error of the best individual of each generation when considering different population sizes and number of iterations of the PID in GA-SPID.

A similar analysis has been carried out in the algorithm GA-DaSPID. Thus, Fig 6 represents the committed error (in seconds) of the best individual in each generation when considering different population sizes and number of iterations. It can be seen that the best performance is obtained with a population of 15 individuals and a number of iterations of the PID equal to 5.

Finally, in both algorithms the number of generations of the stop criterion is fixed to 10, because high number of generations increases the tuning time, but the reduction of the error (as it can be observed in Fig. 5 and Fig. 6) is not remarkably.

 ISBN: 978-1-61208-349-0

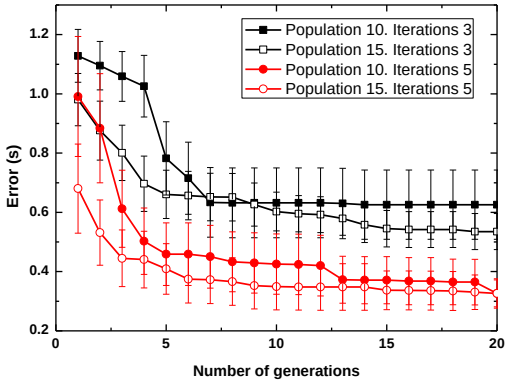

Figure 6. Evolution of the mean committed error of the best individual of each generation when considering different population sizes and number of iterations of the PID in GA-DaSPID.

B. *Simulation study of GA-SPID*

The objective of GA-SPID is to guarantee minimum bandwidth levels to different priority profiles using a PID controller. In contrast to SPID [5], designed for the same purpose but tuned with the Ziegler-Nichols method, GA-SPID incorporates a genetic algorithm to automatically tune the PID. In order to emphasize the importance of a correct tuning process of the PID controller, Fig. 7 and Fig. 8 show the real time evolution of $B_{\max}^{onu_i}$ and the mean value of $B_{alloc}^{onu_i}$ for the SLA_1 profile, respectively, considering different values for the tuning parameters. In particular, we compare the genetic algorithm solution, with the Ziegler-Nichols solution used in [5] and a random configuration of the tuning parameters. As it can be observed in Fig. 7, the genetic algorithm obtains more stability in the maximum permitted bandwidth than the Ziegler-Nichols or the random solution. Besides, Fig. 8 demonstrates the same performance for the evolution of the mean allocated bandwidth. In fact, the genetic algorithm achieves a more stable response than the other two tuning methods when approaching to the guaranteed bandwidth of SLA_1 profile (75 Mbps). Therefore, the importance of optimizing the tuning process to design a reliable PID can be stated by observing the bad performance of the random tuning parameter in both graphs.

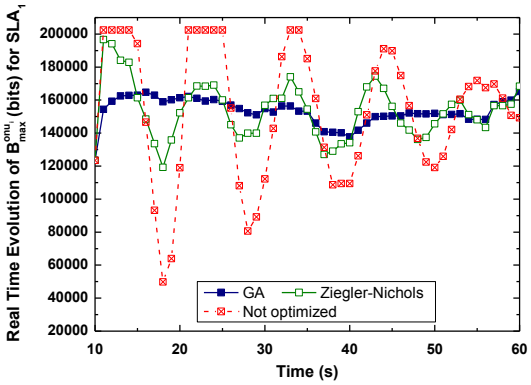

Figure 7. Real time evolution of the maximum permitted bandwidth for the SLA_1 profile considering different values for the tuning parameters

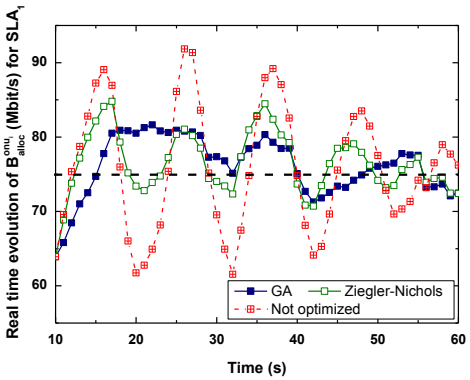

Figure 8. Real time evolution of the mean allocated bandwidth for the SLA_1 profile considering different values for the tuning parameters

On the other hand, the Ziegler-Nichols method is completely manual, based on visual oscillations of the controlled variable for different values of K_p. Therefore, if the chosen values are quite far from the suitable ones, the tuning process may become very slow. Once a value of K_p is selected, the performance when it tunes the PID during a regular interval have to be observed. In case fluctuations at the end of the interval keep high, another K_p value is necessary. In contrast, if fluctuations are low and kept inside a maximum and a minimum threshold, the selected value can be considered as a good K_u and it can be used to tune the PID. Therefore, to compare the tuning process time of the genetic algorithm and the Ziegler-Nichols method, we propose to automate this last method. This way, we consider a random initial value of K_p (between (0,5], as in the genetic algorithm) and its performance is observed during 300 seconds (a good interval to observe a more or less stable response). If fluctuations of the mean allocated bandwidth keep over the 10% above and below of the guarantee bandwidth (that is, the desired value for the variable under control), K_p moves to another value in steps of 0.1. Once the K_p value reaches the higher value of its interval (in this case 5), the following K_p values are obtained from the random initial value in descending steps of 0.1 until the end of the lower threshold of the interval (in this case 0). The selected value of the step affects the tuning time. In fact, if a higher precision is needed, the step of 0.1 can be smaller, but it implies a higher tuning time. In contrast, if the step value increases, it could be difficult to achieve a good tuning process. On the other hand, when the oscillations are within the margin of 10%, the tuning process is finished. As an example, Fig. 9 represents the evolution of the mean allocated bandwidth of the SLA_2 profile when the initial value of K_p is set to 2.7. Moreover, in blue and referred to the axis on the right, the variation of the K_p values is represented. As it can be observed in the graph, for this initial random value of K_p the tuning process last over 10000 s, since this value is far from the range of optimal K_p values. Obviously, if the initial random value is near that range, the tuning process ends more quickly.

 ISBN: 978-1-61208-349-0

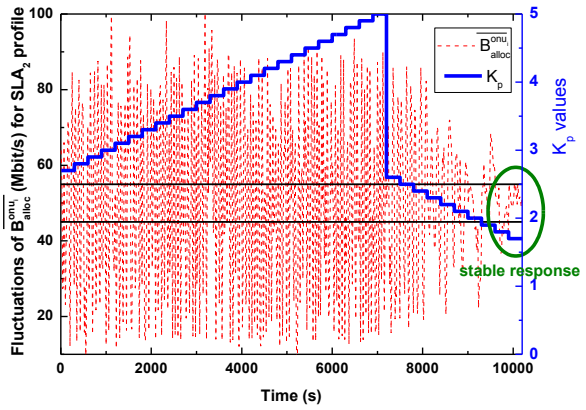

Figure 9. Tuning time of the Ziegler-Nichols method for the SLA_2 profile for a initial K_p of 2.7

On the contrary, the tuning time in GA-SPID is lower. Indeed, the tuning time in this algorithm is given by (3), where N is the population size of the genetic algorithm, m is the number of iterations of the PID in which each individual is tested to obtain its fitness, T_{sample} is the sample time used by the PID controller to obtain each error and N_{Gen} is the number of generations of the stop criterion.

$$T_{tuning} = N \cdot m \cdot T_{sample} \cdot N_{Gen} \quad (3)$$

Therefore, considering the values of Table II, and with a T_{sample} equal to 1 s (which is a suitable value for GA-SPID), the maximum tuning time for GA-SPID is equal to 400 s. Hence, the great difference between both algorithms as regards the time to tune the PID can be noticed. Consequently, the main advantage of GA-SPID is related to the automation of the process, which involves a high reduction of the processing time. Moreover, thanks to the use of a genetic algorithm a more complete evaluation of the solution space is carried out, which leads to a more accurate tuning.

C. Simulation study of GA-DaSPID

The main purpose of GA-DaSPID, as in DaSPID [6], is to control the mean packet delay of delay-sensitive applications taking into account client differentiation. This control is especially critical for high and medium network loads, when it is indispensable to efficiently assign the available resources so that all users comply with their network requirements. However, whereas in DaSPID the tuning process is made following the frequency response method of Ziegler-Nichols, GA-DaSPID uses a genetic algorithm. Therefore, in order to show the performance of GA-DaSPID, Fig. 10 and Fig. 11 represent the mean packet delay of P_1 for SLA_1 and SLA_2, respectively, for the highest network load, that is, ONUs transmitting at 100 Mbit/s. Only the performance of this class of service and these two user profiles is represented due to the lack of space. However, the performance of P_0 service for every profile and P_1 service for the SLA_0 profile is similar.

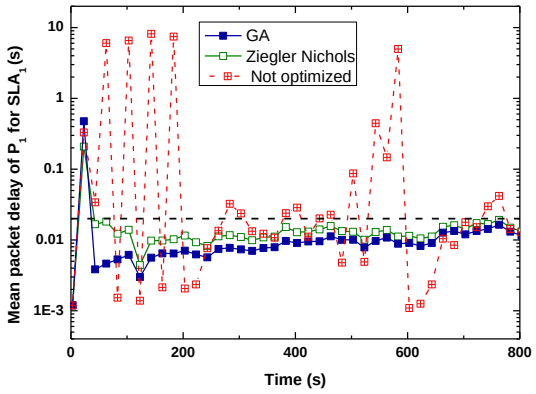

Figure 10. Real time evolution of the mean packet delay of P_1 for the SLA_1 profile considering different values for the tuning parameters

As it can be observed in Fig. 10 and Fig. 11, an optimum selection of the tuning parameters is essential to ensure a quick evolution of the mean packet delay under the limits specified for each profile. In fact, it can be appreciated that the not optimized solution is not able to keep the mean packet delay under the delay limits even in 800 s. In contrast, both Ziegler-Nichols and GA-DaSPID achieve this objective in less than 50 s. In this case, the differences between Ziegler-Nichols and GA-DaSPID are quite small, since the genetic algorithm has proposed a very similar solution to the Ziegler-Nichols method to tune the P controller.

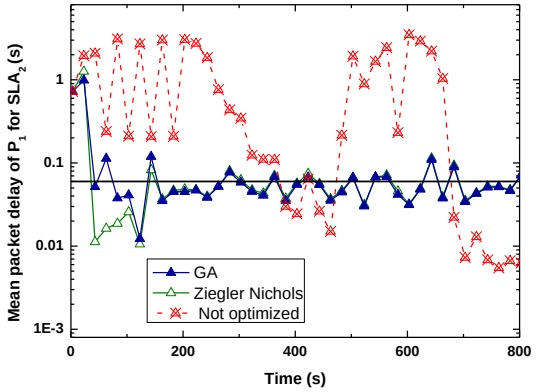

Figure 11. Real time evolution of the mean packet delay of P_1 for the SLA_2 profile considering different values for the tuning parameters

Regarding the comparison of the tuning time in both algorithms, Fig. 12 shows the results for the automated Ziegler-Nichols method to ensure delay limited bounds for SLA_2 profile when the initial K_p value is equal to 2.1. In this case, the allowed range of fluctuations is a 30% above and below of the delay bound (60 ms). As it can be noticed, for this initial random value, the tuning time is higher than 11000 s. Obviously, if the initial K_p value is near the optimal K_p values, the tuning time will be lower.

 ISBN: 978-1-61208-349-0

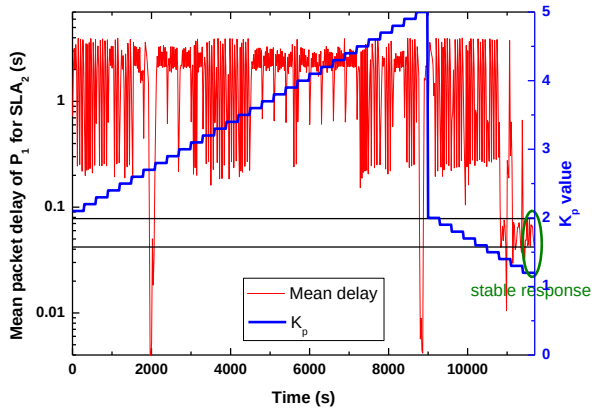

Figure 12. Tuning processing time of the Ziegler-Nichols method for the SLA_2 profile for a initial K_p of 2.1

In contrast, for the GA-DaSPID algorithm, the tuning time is also given by (3). Thus, according to the parameters of Table II and with a T_{sample} time of 10 s (which is the optimal value to control the delay [6]), the tuning time in GA-DaSPID is 7500 s. Therefore, the main advantage of GA-DaSPID is the efficient automation of the tuning process, which leads to a lower tuning time, as it happened in GA-SPID.

IV. CONCLUSIONS

In this paper, the development of a genetic algorithm to tune PID controllers that have been designed to efficiently provide Quality of Service (QoS) in PON networks has been presented. In particular, one PID controller focuses on guaranteeing minimum bandwidth levels to different priority profiles, whereas the other one aims to provide delay requirements to different priority classes of service. In contrast to manual tuning techniques, such as the Ziegler-Nichols frequency response method, the genetic algorithm speeds up and automatically adapts the tuning process according to the stipulated objectives.

In order to demonstrate the benefits of this proposal over manual techniques, we have compared its performance with the Ziegler-Nichols frequency response tuning method. Simulation results have shown that the genetic algorithm efficiently automates the tuning process. Indeed, for the PID controller with bandwidth guarantees, the genetic algorithm allows a more stable response than Ziegler-Nichols for the mean allocated bandwidth and the maximum permitted bandwidth to every SLA. Regarding the P controller, which provides delay guarantees, the genetic algorithm achieves more stability of the mean packet delay of the high priority traffic (P_0, P_1). Furthermore, another important advantage of the genetic algorithm is a significant reduction of the tuning time, since, as it has been demonstrated, the Ziegler-Nichols method could become extremely time-consuming and quite tedious when calculating the tuning parameters. Consequently, the implementation of genetic algorithms to tune PID controllers provides a more accurate, efficient and fast performance than manual techniques, such as the Ziegler-Nichols frequency response method.

REFERENCES

[1] H. Song, B. W. Kim, and B. Mukherjee, "Long-Reach optical access networks: a survey of research challenges, demonstrations and bandwidth assignment mechanisms," IEEE Communications Surveys & Tutorials, vol. 12, no. 1, pp. 112-122, 1st Quarter 2010.

[2] K. Ahl, FTTH Council Europe, "Creating a brighter future. A sustainable future enabled by fibre to the home," [Online]. Available from: www.scotland.gov.uk/Resource/0042/00425685.ppt 05.2014

[3] T. Jiménez et al., "Self-adapted algorithm to provide multi-profile bandwidth guarantees in PONs with symmetric and asymmetric traffic load," Photonic Network Communication, vol. 24, no. 1, pp. 58-70, January 2012.

[4] N. Merayo et al., "EPON bandiwdth allocation algorithm based on automatic weight adaptation to provide client and service differentiation," Photonic Network Communication, vol.17, no. 2, pp. 119-128, April 2009.

[5] T. Jiménez et al., "Implementation of a PID controller for the bandwidth assignment in Long-Reach PONs," Journal of Optical Communications and Networking, vol. 4, no. 5, pp. 392-401, May 2012.

[6] T. Jiménez et al., "A PID-based algorithm to guarantee QoS delay requirements in LR-PONs," Optical Switching and Networking, in press. D.O.I:http://dx.doi.org/10.1016/j.osn.2014.01.005

[7] C. H. Chang, N. M. Alvarez, P. Kourtessis, R. M. Lorenzo, and J. M. Senior, "Full-service MAC protocol for metro-reach GPONs," Journal of Lightwave Technology vol. 28, no. 7, pp. 1016 – 1022, April 2010.

[8] N. Merayo et al., "Adaptive polling algorithm to provide subscriber differentiation in a long-reach EPON," Photonic Network Communications, vol. 19, no.3, 257-264, June 2010.

[9] K. H. Ang, G. Chong, and Y. Li, "PID control system analysis, design and technology," IEEE Transactions on Control Systems Technology, vol. 13, no. 4, pp. 559-576, July 2005.

[10] P. Cominos and N. Munro, "PID controllers: recent tuning methods and design to specification," IEEE Control Theory and Applications, vol. 149, no. 1, pp. 46-53, Enero 2002.

[11] K. J. Aström and T. Hägglund, "Advanced PID control". Research Triangle Park, NC: ISA-The Instrumentation, Systems, and Automation Society, 2006.

[12] D. A Wati and R. Hidayat, "Genetic algorithm-based PID parameters optimization for air heater temperature control," 2013 International Conference on Robotics, Biomimetic, Intelligent Computational Systems (ROBIONETICS), Nov. 2013, pp. 30 – 34,

[13] M. J. Neath, A. K. Swain, U. K Madawala, and D. J. Thrimawithana, "An optimal PID controller for a bidirectional inductive power transfer system using multiobjective genetic algorithm," IEEE Transactions on Power Electronics, vol. 29, no. 3, pp. 1523-1530, March 2014.

[14] C. K. Chen, H. H. Kuo, J. J Yan, and T. L. Liao, "GA-based PID active queue management control design for a class of TCP communication networks," Expert Systems with Applications, vol. 36, no. 2 Part 1, pp. 1903-1913, March 2009.

[15] Opnet Modeler. [Online]. Available from: http://www.opnet.com 04.2014.

[16] IEEE 802.3ah Ethernet in the First File Task Force [Online]. Available from: http://www.ieee802.org/3/efm/public 04.2014.

 ISBN: 978-1-61208-349-0