This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE Globecom 2010 proceedings.

A Cognitive Approach for Cross-Layer Performance Management

Neumar Malheiros[†], Dzmitry Kliazovich[‡], Fabrizio Granelli[*], Edmundo Madeira[†], and Nelson L. S. da Fonseca[†]
[†]Institute of Computing – University of Campinas, Brazil
Email: {ncm, edmundo, nfonseca}@ic.unicamp.br
[‡]University of Luxembourg, Luxembourg
Email: dzmitry.kliazovich@uni.lu
[*]DISI – University of Trento, Italy
Email: granelli@disi.unitn.it

***Abstract*—The evolution of network technologies brought increasing management complexity of networking infrastructure and protocols. Cognitive networking was introduced to deal with such complexity. This work presents a cognitive algorithm for cross-layer performance management which is the core of a decentralized framework for self-configuration of communication protocols. We illustrate the proposed solution for the joint reconfiguration of protocol parameters at different layers. The cognitive joint adaptation of TCP congestion window and MAC layer data rate is carried out as a proof of concept. Simulation results show performance improvement given by the proposed approach under changing network conditions.**

I. Introduction

With the increasing level of complexity, performance management of current communication networks represents a challenge to be overcome. Several performance drawbacks in current network environments are inherited from the limitations of the Internet protocols designed for wired networks. Despite its success, the TCP/IP protocol suite still presents fundamental open issues [1]. It neither includes adaptation mechanisms for dynamic network environments, nor it allows communication and coordination between layers. Moreover, the traditional centralized network management solutions, based on a client/server approach, are not suitable to deal with the complexity of heterogeneous networks with stringent QoS requirements [2], [3].

The dynamic adjustment of network protocols using cognitive processes represents a promising approach to deal with network management complexity. In particular, the concept of cognitive networks has emerged as a major technology for designing adaptive communication systems. Cognitive Network [4], [5] is a networking paradigm that combines cognitive algorithms and cross-layer design in order to provide real-time optimization of complex communication networks focusing on system-wide goals. Recent research efforts have demonstrated how cross-layer design can provide performance optimization by allowing information sharing among non-adjacent layers mainly in the case of wireless networks [6].

In a previous work [7], we proposed a cognitive framework for dynamic reconfiguration of communication protocols. Such framework, called CogProt, is based on the main concepts of the cognitive network paradigm. In CogProt, performance optimization is obtained by periodically reconfiguring protocol parameters through a quality feedback loop. In this work, we formalize the procedures involved in such framework and present an algorithm for implementation of the cognitive loop. Such cognitive algorithm provides dynamic reconfiguration of protocol parameters even at different layers. It is scalable and does not require any change on protocol operation.

In addition, we analyze different deployment scenarios. The proposed solution is designed to operate not only between different protocol layers of a single node, but between different nodes of the network in a decentralized way. In previous work, we have illustrated the application of CogProt in the intra-layer scenario [7], [8]. In this work, we present a case study to demonstrate the application of the proposed cognitive algorithm in the inter-layer scenario. As a proof of concept, we have implemented the proposed solution as an extension to the Network Simulator (ns-2). We illustrate the dynamic reconfiguration of protocol parameters at different layers. Specifically, TCP congestion window and MAC layer data rate are dynamically adapted in a TCP-over-WiFi scenario. Simulation results demonstrate performance improvement resulting from the ability to timely react to network dynamics.

This paper is structured as follows: Section II describes the core of the proposed approach and provides insights into architectural and deployment scenarios; Section III presents a case study to illustrate the application of the proposed approach in inter-layer optimization; Section IV concludes the paper outlining directions for future research on the topic.

II. Cognitive Self-Configuration

The cognitive framework provides dynamic reconfiguration of protocol parameters, through learning and reasoning, in order to optimize system-wide performance. To this aim, a cross-layer cognitive plane is introduced as illustrated in Fig. 1. This plane provides self-configuration functionalities to support runtime reconfiguration of protocol stack parameters at different layers. This makes a network node implementing the cognitive framework capable of adapting its protocol parameters to fit current network conditions.

* The authors would like to thank CNPq and FAPESP (process number 06/50512-4) for supporting this work.

978-1-4244-5637-6/10/$26.00 ©2010 IEEE

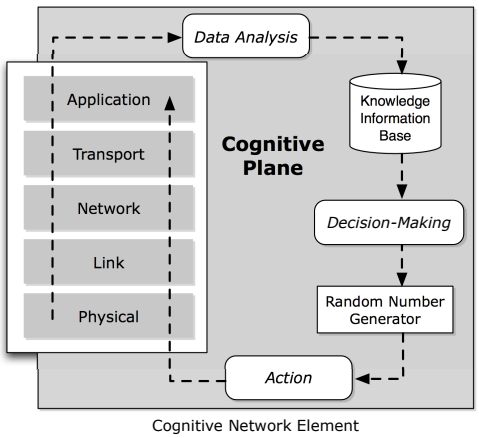


Fig. 1. Cross-Layer Framework for Cognitive Self-Configuration.

A. The Cognitive Loop

The self-configuration process consists in continuously monitoring the performance obtained in current setup and then enforcing reconfiguration actions to adjust protocols parameters. Those cross-layer self-configuration capabilities are supported by the quality feedback loop illustrated in Fig. 1. Such cognitive loop consists of three phases: *Data Analysis*, *Decision Making*, and *Action*. Consider the adjustment of a parameter of interest P. In order to define such parameter, we need to provide the attributes described in Table I. Performance information is analyzed at the end of each iteration interval τ according to a pre-defined quality metric associated with the parameter being adjusted. The performance monitoring may include different quality metrics, for example, frame error rate for MAC Layer and end-to-end delay for network layer. Performance information about each value $P_i \in \{P_{min}, \cdots, P_{max}\}$ is maintained in the knowledge base.

Let P_c be the current value for the parameter of interest P. In the *Data Analysis* phase, the mechanism updates the average performance $E[P_c]$ for P_c (the current value of P). The obtained performance information is averaged with an exponentially weighted moving average (EWMA) as follows:

$$E[P_c]^t = E[P_c]^{t-1} * (1-s) + m(P_c) * (s) \qquad (1)$$

where E^t represents the knowledge base with performance information for each value of P at iteration t; $E[P_c]$ is the average performance information for the value P_c; s (the smoothing factor) is the weight assigned to current performance obtained from P_c; and $m(x)$ is the measured performance obtained with a given value x. That is, the measured performance for the current value of P is used to calculate the average performance for that value. As the process iterates, the node learns the average performance of each $P_i \in \{P_{min}, \cdots, P_{max}\}$.

In the *Decision Making* phase, the mechanism decides on the best value for P. The local knowledge base reflects the performance history for each possible value. The algorithm analyses that base to find the value of P that provides the best performance. The corresponding value is assigned to the mean of the normal distribution employed for the selection of the next value of P. In this way, the mean of the random number generator (RNG) is assigned to the current best value for P. As a consequence, the cognitive process explores alternative values "around" the best value of P, in order to track eventual changes in the operating environment. The cognitive algorithm continuously adjusts the mean of the distribution, with the aim to converge to the best value under current conditions. In the *Action* phase, a new random value is randomly generated according to a normal distribution (in the range $\{P_{min}, \cdots, P_{max}\}$), and assigned to parameter P.

There are three control variables for the cognitive mechanism as summarized in Table II. The iteration interval (τ) of the cognitive loop depends on the application requirements and level of operation (for instance, flow or frame level). The standard deviation (σ) of the normal distribution defines the aggressiveness of the mechanism. The lower the σ, the more conservative is the behavior of the algorithm in trying new values for P. Therefore, this parameter directly affects the convergence time and system stability. The smoothing factor (s) can be used to control the relevance of recent performance information and also to balance between stability and reactiveness to changing network conditions. High values of s increase the weight of immediate collected performance information on the average performance, which enable the algorithm to quickly react in the face of changes.

B. Algorithm Description

In the following, we present a high level algorithm for implementation of the quality feedback loop for the parameter of interest P. As the algorithm iterates, the knowledge base is filled with average performance for each value of P. In the algorithm, such base is referred to as B, that is, $B[i]$ is the average performance obtained from $i \in \{P_{min}, \cdots, P_{max}\}$.

For the duration of the optimization process, the cognitive node monitors the performance of current protocol parameters setup according to pre-defined target quality metrics. With the proposed cognitive algorithm, the quality metric can be chosen

TABLE I
PARAMETER DEFINITION.

P	Parameter of interest
$P_{default}$	Default value for P
$\{P_{min}, \cdots, P_{max}\}$	Range for possible values for P
Q	The corresponding performance quality metric which could be a simple or compound metric, or even a utility function.

TABLE II
QUALITY FEEDBACK LOOP CONTROL VARIABLES.

τ	Interval of operation of the quality feedback loop.
s	The weight (smooth factor) associated with the performance obtained for the current value of P.
σ	The standard deviation for the normal distribution function.

Algorithm 1 Coginitive Self-Configuration of P

Require: $P_{default} \in [P_{min}, \cdots, P_{max}]; \tau, \sigma, s \in (0, 1]$
Ensure: Continuously reconfigure parameter P
  $P \Leftarrow P_{default}$
  $\mu \Leftarrow P_{default}$
  repeat
    $wait(\tau)$
    $x \Leftarrow measurePerformance(P)$
    $B[P] \Leftarrow B[P] \times (1 - s) + x \times s$
    $\mu \Leftarrow i | max(B[i], i \in [P_{min}, \cdots, P_{max}])$
    $P \Leftarrow normal(\sigma, \mu)$
  until forever

according to the Quality of Service requirements of applications. Furthermore, the quality metric can be changed during network operation depending on priority among applications. In this way, a network node can choose the proper quality metric in order to optimize different kinds of applications as real-time or data applications.

Furthermore, the quality metric is not limited to a single metric. It should be designed according to the application scenario. The quality metric could be a compound metric which reflects priorities on different variables or even a more complex function, as a utility function that enforces fairness. In order to enable system-wide performance optimization the quality metric should be designed to reflect end-to-end goals.

C. Signaling and Deployment

The proposed cognitive mechanism adds only a limited degree of complexity bringing the advantage of improved performance and reactiveness to changing operating scenarios. This way it can be implemented even in network elements with limited resources and incrementally deployed in the network. In fact, the proposed mechanism does not change protocol messages and operation.

The cognitive protocol optimization can be performed either within a single-layer or involving several layers of the protocol stack. Fig. 2 illustrates cognitive optimization loop for single-layer and inter-layer scenarios. In the inter-layer scenario, the cognitive mechanism is responsible for simultaneous reconfiguration of protocol parameters at different layers. In this case, the mechanism is able to provide cross-layer performance management. Performance information from different layers is stored in the knowledge information base. This architecture characterizes a cross-layer solution based on a shared database according to the taxonomy described in [6].

The decision making process is intrinsically decentralized since it can be performed by each node independently without any communication with other nodes. This is an important point, since the proposed framework aims at supporting incremental deployment in current networks as well as providing a scalable solution (i.e. reducing communication and information exchange to the minimum). However, besides local performance information, the decision-making procedure may also take into account remote performance information and

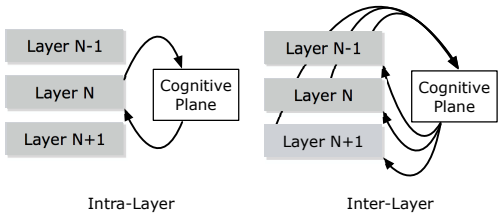
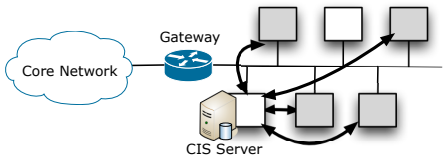
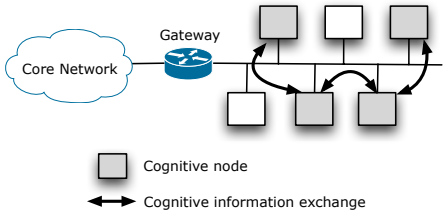


Fig. 2. Intra- and inter-layer scenarios.



(a) Centralized (service-based).



(b) Distributed.

Fig. 3. Signaling architectures.

common configuration policies. According to the scope it operates, the cognitive mechanism can be classified into: *local* (learning and decision procedures are based only on locally available information, that is, the knowledge information base built by the node itself); or *remote* (learning and decision processes take into account additional information available in the network remotely).

In the latter case, the cognitive algorithm may utilize information available at the neighboring nodes or provided by a Cognitive Information Service (CIS). Two signaling architectures are envisioned: *centralized* (this is a service-based architecture and decision-making uses reconfiguration policies provided by CIS as in Fig. 3(a)); and *distributed* (decision-making considers additional network information received from neighboring cognitive nodes as in Fig. 3(b)).

The centralized approach may fit well in several scenarios, for instance, in wireless access networks with powerful base stations providing coordinated management actions on behalf of the mobile nodes. In this case, the nodes can use the CIS to share knowledge about the local network to support the cognitive optimization process. The CIS fosters cognitive signaling in the network segment (in a similar way a DHCP service provide dynamic configuration for IP addressing). When a node joins the network, it can request configuration information to the CIS. That information accumulates the

knowledge already obtained by neighboring nodes. The CIS builds its knowledge base by gathering information from nodes in the local network. It can also be used to enforce high-level administrative policies. On the other hand, the decentralized approach provides a higher degree of scalability with the cost of requiring the definition of a proper signaling protocol. In this case, neighboring nodes may share their performance information to help each other on reconfiguring the protocol parameters according to current conditions.

III. Case Study

As a proof of concept, we have implemented the proposed cognitive mechanism into the Network Simulator (ns-2). In this work, we consider an inter-layer scenario, that is, the joint optimization of multiple parameters at different layers. In this section, we describe the parameters of interest and the simulated scenario, and discuss simulation results.

A. Parameters of Interest and Topology

TCP window increase factor (α) and MAC data rate were selected as primary parameters for cognitive optimization. The congestion window increase factor accounts the amount of bytes the TCP window is increased each RTT period with no losses detected. Controlling the TCP window evolution allows improving the flow performance. Higher α values bring performance increase and they are desirable in high bandwidth-delay network with low or moderate congestion levels, and should be avoided otherwise. However, there is no way for a network node to determine available network bandwidth and the level of congestion at the end-to-end path in advance without performing explicit measurements. Therefore, it is not possible to define the best default value for the α.

The second parameter of interest is the MAC data rate. In wireless networks, higher data rates are commonly achieved by using modulation schemes that efficiently take advantage of good channel conditions. However, these schemes are more sensitive to medium quality degradation and do not perform well for long range transmissions. On the other hand, the use of robust modulation schemes leads to more resilient connections, but it results in lower data rates due to redundancy and control overhead. Here, the challenging issue is how to dynamically select an appropriate modulation scheme (data rate) to current conditions in order to optimize network performance.

Fig. 4 illustrates the simulated topology. TCP flows are initiated between the mobile source node S and the fixed destination node D. We have performed experiments considering three scenarios: when the node is fixed near the base station (BS); when the node is fixed far away from the BS; and when the node moves. In the simulations performed, the mobile node is either placed stationary close (5m) or distant (50m) from the base station, or moves between these two locations. In addition, a cross-trafc flow organized between the nodes C1 and C2 shares the common bottleneck link (R1-R2).

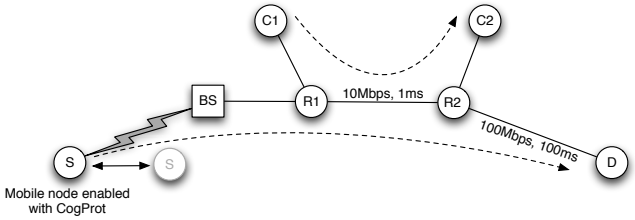
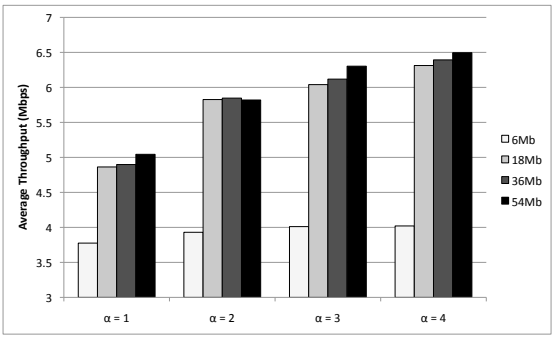


Fig. 4. Simulated topology.

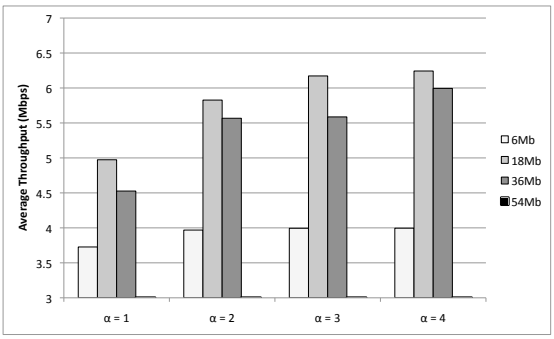
B. Simulation Results

We have performed simulations with runtime adjustment of both parameters. We consider $\alpha \in \{1, \cdots, 4\}$ and the value 1 is the default value. With respect to the second parameter, we consider four data rates {6, 18, 36, and 54Mbps} (which are available in IEEE 802.11g interfaces) and the value 6Mbps is the default value. The quality metric for both parameters is the average flow throughput. A two-dimensional array is used as the data structure for performance information base. The average performance for each combination of an increase factor and data rate value is stored into such knowledge base.

In a first experiment, there is no cross-traffic and the source node does not move. Fig. 5 presents the performance for fixed values of α and data rate. Fig. 5(a) presents the performance when the node is fixed near the Base Station (BS) (5m). As expected, the higher data rate the better the performance. Fig. 5(b) presents the performance when the node is fixed farther from the Base Station (BS) (50m). For the highest data rate the frame error rate is 100%. In both cases, we can see that the performance improves if we increase the value of α.



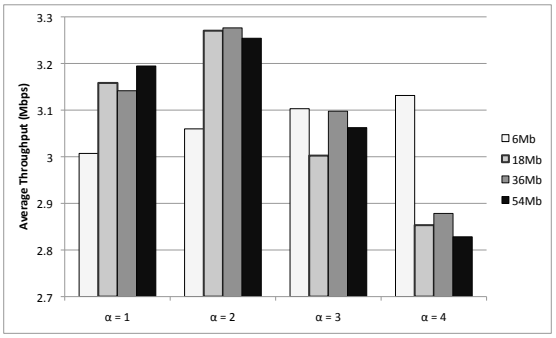
(a) Near



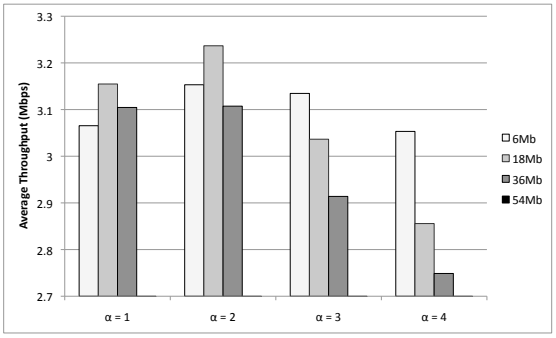
(b) Far

Fig. 5. Average TCP throughput without cross-traffic.

In a second experiment, we consider CBR over UDP cross-traffic. The source node does not move. Fig. 6 presents the performance for fixed values for α and data rate. In this case, no matter if the node is fixed near (6(a)) or far away (6(b)) from the BS, higher values for α do not provide the best performance, as opposed to the previous result. We also can see that there is no a global optimal value for the data rate.



(a) Near



(b) Far

Fig. 6. Average TCP throughput with cross-traffic.

From the two previous experiments, it is clear that it is not possible a priori to set the best values for α or data rate, as they depend on the position of the node and the traffic interference. Then, for any static configuration the node will experience performance degradation in face of changing conditions. In a third experiment, we consider changing conditions. There is ON/OFF UDP cross-traffic and the source node moves farther away from the BS. In this case, the proposed cognitive mechanism provides the highest average performance as we can see in Figure 7. The dynamic reconfiguration enabled by the cognitive algorithm outperform any combination of values for increase factor (α) and data rate.

IV. Conclusion

The proposed cognitive mechanism allows dynamic reconfiguration of main protocol parameters at different layers for achieving performance goals driven by target quality metrics. The algorithm does not add a high degree of complexity, and, therefore, it can be implemented even in network elements with limited resources. In addition, the proposed approach

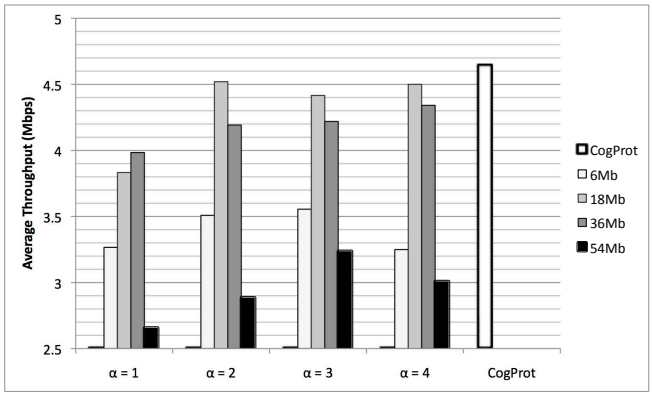


Fig. 7. Average TCP throughput with changing conditions.

does not raise compatibility issues. Cognitive nodes can interoperate with ordinary nodes since the mechanism does not change protocol messages and operation. The proposed approach was illustrated for the joint adaptation of TCP window and MAC data rate. Both parameters are adjusted during runtime based on the average flow throughput experience achieved in the immediate past. The mechanism is able to adjust the parameters to avoid performance degradation, therefore improving average performance on dynamic environments.

Future work will be focused on implementation of the proposed algorithm in a real testbed for dynamic reconfiguration of the TCP protocol. In addition, we will work on further development of architecture for the Cognitive Information Service, as well as on signaling mechanisms for cognitive information exchange between network nodes and the service, and among the nodes themselves.

References

[1] F. Foukalas, V. Gazis, and N. Alonistioti, "Cross-layer design proposals for wireless mobile networks: a survey and taxonomy," *Communications Surveys & Tutorials, IEEE*, vol. 10, no. 1, pp. 70–85, 2008.

[2] B. Jennings, S. van der Meer, S. Balasubramaniam, D. Botvich, M. Foghlu, W. Donnelly, and J. Strassner, "Towards autonomic management of communications networks," *Communications Magazine*, vol. 45, no. 10, pp. 112–121, 2007.

[3] N. Samaan and A. Karmouch, "Towards autonomic network management: an analysis of current and future research directions," *Communications Surveys Tutorials, IEEE*, vol. 11, no. 3, pp. 22–36, 2009.

[4] R. W. Thomas, D. H. Friend, L. A. Dasilva, and A. B. Mackenzie, "Cognitive networks: adaptation and learning to achieve end-to-end performance objectives," *Communications Magazine*, vol. 44, no. 12, pp. 51–57, 2006.

[5] C. Fortuna and M. Mohorcic, "Trends in the Development of Communication Networks: Cognitive Networks," *Computer Networks*, vol. 53, no. 9, pp. 1354–1376, 2009.

[6] V. Srivastava and M. Motani, "Cross-layer design: a survey and the road ahead," *Communications Magazine*, vol. 43, no. 12, pp. 112–119, 2005.

[7] D. Kliazovich, N. Malheiros, N. L. S. da Fonseca, F. Granelli, and E. Madeira, "CogProt: A Framework for Cognitive Configuration and Optimization of Communication Protocols," in *The 2nd International Conference on Mobile Lightweight Wireless Systems*, 2010.

[8] L. Chaves, N. Malheiros, E. Madeira, I. Garcia, and D. Kliazovich, "A Cognitive Mechanism for Rate Adaptation in Wireless Networks," in *Modelling Autonomic Communications Environments (MACE)*, ser. Lecture Notes in Computer Science, vol. 5844. Springer, 2009, pp. 58–71.