Towards a Declarative Framework for Managing Application and Network Adaptations

Palanivel Kodeswaran and Anupam Joshi
Department of Computer Science and Electrical Engineering
University of Maryland, Baltimore County
Baltimore, MD 21250
Email: palanik1@umbc.edu,joshi@cs.umbc.edu

***Abstract*—Cross layer optimizations are increasingly being used in a variety of applications to improve application performance. However most of these implementations are ad hoc and performed on a per application basis. In this paper we propose a declarative framework for managing application and network adaptations. The declarative paradigm provides a much needed clean line of separation between the high level goals and the low level implementations. Our framework exposes the tunable features of both the application and the network across layers of the network stack which can then be jointly optimized. We allow operators to control the adaptation process through operator specified policies. This enables operators to retain control over their networks while the application and the network adapt in response to changing conditions. To support evolution, we pursue an ontological approach and use semantic web languages such as OWL and RDF in our framework for the policy and declarative specifications, thereby also leveraging the inherent reasoning and conflict resolution features of these languages. We then describe our framework developed on top of NS2 to demonstrate the utility of our approach in the easy implementation of cross layer optimizations through sample application scenarios.**

I. Introduction

The existing layered network architecture assumes all application intelligence is present at the end hosts and views the network as a dumb transport medium providing only best effort service. This architecture may have been good enough to support the simple applications of the past such as FTP and email. On the other hand, emerging network applications such as video streaming and VoIP demand harder delay and reliability requirements from the network. These applications exhibit the potential to deliver improved performance in the presence of network support as witnessed in application level overlay networks, where each overlay is optimized for a specific application/metric. There is also growing diversity in the physical layer technologies in terms of channel characteristics such as loss rate, bandwidth, security etc. Clearly, given the requirements of emerging applications and the growing diversity at the physical layer, a one size fits all solution cannot yield optimal performance across the entire gamut of applications. There is a need for a mechanism that can map high level application requirements to underlying network capabilities in a generic and extensible manner.

Cross layer optimizations have often been used as a tool to improve application performance by making the network more application aware. However, most of these adaptations are ad hoc and implemented on a per application basis. Moreover, the network has no understanding of the application's adaptation capabilities. While the underlying principle for a majority of the adaptations is the same, the ad hoc nature of implementation precludes sharing of mechanisms among similar applications. In this paper, we propose a declarative framework for managing cross layer adaptations. In our framework, application and network features are exposed across layers in a seamless manner. This enables joint reasoning over the specifications to choose the optimal adaptation. The declarative nature of the framework emphasizes on the high level goals of the adaptation, rather than focussing on the low level implementation mechanisms. Additionally, our framework can be seen as a tool for implementing and experimenting with new adaptation strategies as well as changing adaptation policies at run time without modifying any application or network code. We would also like to point out that our framework only enables the intelligent management of application and network smarts, and does not provide any performance guarantees beyond those provided by the underlying mechanisms.

The main contributions of this paper are

- Proposing an ontology based declarative framework for describing network features and application capabilities
- Providing a clean interface for implementing cross layer optimizations through policy specifications over exposed mechanisms in the above developed ontology
- Leveraging the language support of semantic web languages for evolution, reasoning and conflict resolution in cross layer adaptations

II. Related Work

There is considerable consensus in the networking community that the current Internet architecture has ossified and we need to explore newer designs to accommodate the requirements of emerging applications. The focus in [1], [2] is on exploring the design of networks from scratch using a clean state approach by considering architectural solutions to issues that are typically excluded in the current Internet architecture such as security, location management, economics, scalable network control and management. Our work can be seen in this context as the enabling technology that allows glueing the various architectural blocks together to compose optimized per application network stacks.

One of the closest related works to our framework is the SILO architecture for composable network services [3], [4]. Their main goal is to design a network architecture that enables cross layer interactions gracefully. The SILO architecture consists of fine grained services implementing well defined functions, a composition agent and a services ontology modelling the constraints among services. Our framework differs from SILO in that we focus on providing a declarative framework for managing cross layer interactions. Being declarative in nature, our framework hides the internal implementation details of services and can be easily extended to include SILO-like services which have well defined data and control interfaces. Unlike SILO, ontologies are used in our framework to model the application and network intelligence that is exposed across layers of the network stack to enable cross-layer adaptation. Furthermore, we pursue an evolutionary approach requiring applications and networks to expose only enough semantics as necessary to enable synergetic collaboration between them as opposed to completely rewriting existing network services to meet the control and data interface specifications of the SILO architecture.

In [5], Loo et al. describe their work in the declarative specification and implementation of overlays using the overlog declarative language. In [6], the authors propose the declarative routing framework, where in various routing protocols are expressed as recursive datalog queries. In [7], Singh et al. propose using recursive queries for debugging and monitoring declarative network systems. Performance studies showed that the overhead of monitoring applications using queries is within acceptable levels. The advantages of the declarative definitions are compactness and the ability to express new protocols as modifications to existing ones without having to rewrite the protocol implementation completely. In [8], the authors formally define the NDLog language for declarative network specifications and propose query optimization techniques for distributed execution. While the focus of these works has been on the declarative specification of routing and network protocols, we concentrate on using the declarative technology for managing application and network adaptations. Also, since we reason over application requirements in our framework, we could potentially leverage declarative networks for setting up appropriate network level mechanisms such as QoS aware routing when application requirements dictate so.

In prior work from our group [9], we proposed the CoCoNET architecture for content based networking in which packets are semantically tagged using an operator specified ontology. The meta data could be used to specify a variety of properties such as type of data, security credentials etc. A reasoner at each node could then reason over the meta data and perform appropriate functions as specified by policies [10]. In [11], we extend the framework to manage BGP export policies. Our current work can be seen as a further extension in that we now focus on reasoning over application requirements and adapting both the application and network layers jointly.

There is a large body of existing literature on cross layer solutions, particularly for the wireless medium, and we do not attempt to provide an exhaustive survey here. In [12], the authors study the performance of using link layer ARQ and FEC for improving the performance of TCP Tahoe and Reno. In [13] and [14], the authors propose cross layer solutions for streaming video over wireless links. Their basic idea is to provide higher reliability to more important packets compared to lesser important packets. In [15], Choi et al. consider jointly optimizing inter and intra layer functions for improving video over wireless performance. In [16], the authors stress the need for applications to respond to external events where as [17] describes an application driven resource management framework for meeting application requirements.

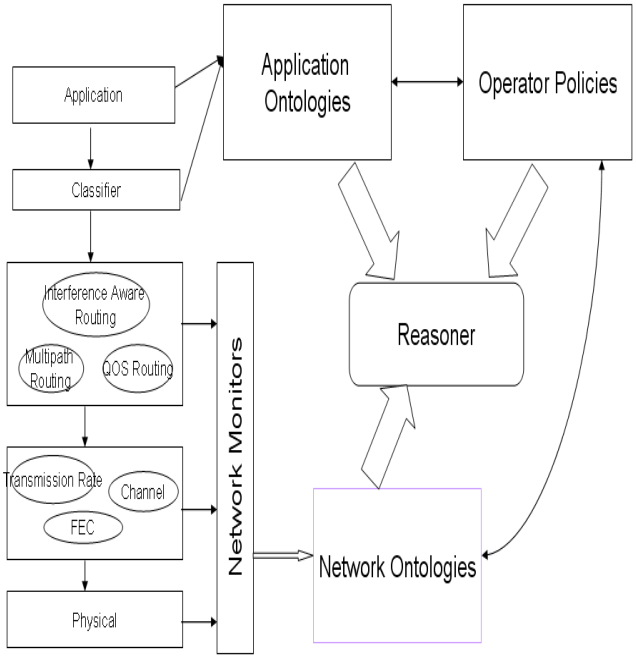

Fig. 1. System Architecture

III. System Architecture

In this section we describe our system architecture illustrated in Figure 1. One of our main goals is to provide a framework that enables users/operators to seamlessly express and implement cross layer optimizations. This enables not only easy implementation but also allows sharing optimization mechanisms among similar applications. Our architecture involves exposing application semantics and network capabilities across layers of the network stack in a declarative fashion. Application requirements and network features are encoded in appropriate application and network ontologies. Cross layer optimizations are then specified through operator policies in the form of rules over the exposed mechanisms. The ontologies and operator policies are input into the reasoner which implements the cross layer optimization by setting appropriate parameter values across layers.

Design Rationale Enabling a generic framework for composing cross layer optimizations requires that layers of the

network stack be aware of each other, and expose semantics above and below each other. We do not intend to completely expose each layer to the other and thereby run the inadvertent risk of coupling functionalities between layers. Rather, we would like to retain the benefits of protocol isolation that the current layered architecture provides. We instead focus on hiding the protocol implementation details within each layer, but merely provide a control interface to manage the behavior of the protocol. Thus, our architecture can be seen as augmenting the traditional data interface of protocols with a control interface. The control interface itself needs to be only simple. A large majority of existing cross layer optimizations involve setting appropriate values of parameters across layers. Thus, the functionalities of each layer can be parameterized with the parameters being the only piece of information that need to be exposed across the control interface. Furthermore, each layer could have multiple mechanisms that provide the same functionality. For example, reliability at the wireless MAC layer could be provisioned through either FEC or retransmission. Again, the strength of the FEC and the maximum retransmission count determine the amount of reliability provided to the upper layers. Thus, to implement cross layer optimizations in our framework, we need to expose only the following pieces of information across layers: i) the Mechanisms available at each layer and their purported functionality and ii) the control parameters that enable managing protocol/mechanism behavior. We would also like to stress that while our architecture facilitates the seamless exposure of application and network semantics, we do not "require" either to be exposed. In the worst case, our architecture will default back to the generic best effort model.

A. Ontology Based Framework

The interface used to expose semantics across layers must be expressive enough to represent network and application intelligence. Furthermore, the interface should be able to support the requirements and capabilities of emerging applications and networks. To this end, we pursue an ontological approach and use OWL encoded descriptions for exposing application and network functionalities across layers. OWL is actually a collection of three sub-languages viz. OWL-Lite, OWL-DL and OWL Full, each varying in expressivity and computational tractability of reasoning. Of these, OWL-DL which is based on description logic provides the right balance of expressivity and computational tractability and is used in our framework. Semantic web languages have a number of advantages such as being firmly grounded in logic, support for evolution, possessing model theoretic semantics as well as enabling easy verification of descriptions. The rationale behind choosing OWL as the specification language in our framework is motivated by its ability to easily model class hierarchies, properties and restrictions typically seen in the networking domain. Furthermore, operator policies (described in later sections) can be easily specified as SWRL (Semantic Web Rule Language) [18] rules over OWL ontologies to express Event-Condition-Action rules that are largely prevalent in networks. We recognise that semantic reasoning is a computationally expensive task and the scalability of reasoning needed in an approach like ours is a concern. However, we also note that there is significant new work in this space (for instance [19]) that creates highly scalable reasoners that could be used in our framework.

We now describe the ontologies used in our framework

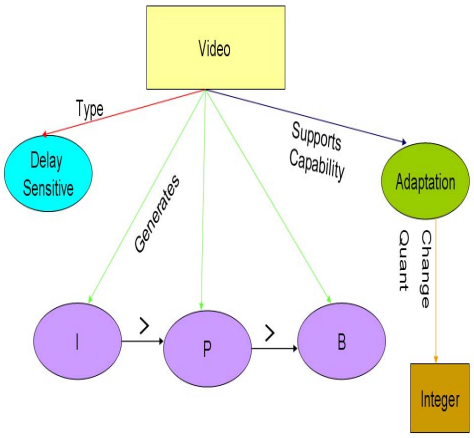

Fig. 2. Video Application Ontology

1) Application Ontology: An application ontology is a formal specification that describes application requirements and capabilities which can then be used in cross layer optimizations. Applications need to expose only features that enable controlling application behavior as well as those that could serve as hints to other layers. Typically, an application ontology would minimally specify the following

Type of application : Specifies whether the application is delay sensitive like Video or throughput sensitive like FTP. The idea is to enable choosing the right adaptation based on the type of the application.

Adaptation Capabilities : Lists the adaptation capabilities of the application in response to changing network conditions and the method to invoke them.

Generated messages and relative priorities : When the available network bandwidth falls below a threshold, the network layer could use the message type and relative priorities to decide which packets to drop. Similarly, we could provide higher reliability to more important messages than lower priority ones.

We are currently developing a video application ontology as shown in Figure 2 that is available online at "http://www.umbc.edu/∼palanik1/adaptation.owl". The ontology in figure 2 describes video as a delay sensitive application generating three types of messages(frames) viz. I, P and B in that order of priority. Also, video supports an adaptation capability of changing the quantization level through the "changeQuant" function in response to varying network conditions.

The intent is that application developers will also provide

the application ontology describing the semantics of the application. In cases where this is not feasible, we may need to look at packet headers and payload data to infer the application semantics. This is achieved through the classifier module in the system architecture.

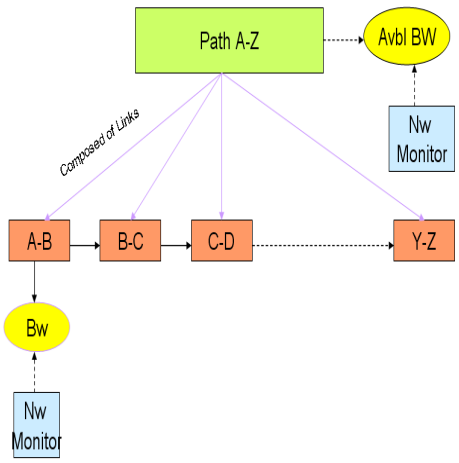

Fig. 3. Network Ontology

2) Network Ontology: Similar to the application ontology, the network ontology is a formal specification that describes the capabilities of the network. We are currently developing a network ontology available online at [20]. As described earlier, the goal is to expose mechanisms available at each layer and their control parameters. Unlike applications, networks have dynamic properties such as available bandwidth that vary with time and need to be modelled in our framework. To achieve this, we propose the use of " network monitors" that measure dynamic properties such as bandwidth and error rate and update the values in appropriate ontologies. Alternatively, the monitors could store the instantaneous measured values in a database, and the reasoner could retrieve the values from the database when needed. Figure 3 shows a sample network ontology in which path A-Z is described as being composed of individual links. Furthermore, each link and path has an associated bandwidth property that is measured and updated by appropriate network monitors.

3) Operator Policies: Now that application and network capabilities are exposed through appropriate ontologies, we focus on the implementation of cross layer optimizations. In our framework, operator policies are used to specify the adaptation process. Operator policies represent an important building block in our architecture as most operators may refuse to deploy a framework over which they have no control. In this context, operator policies are used to represent the requirements and preferences of operators in composing adaptations and ensuring that operators are still in control of their network. Such preferences may arise for a variety of reasons such as system wide policies, availability of hardware accelerators for certain operations etc. In our current framework, operator policies are specified as SWRL rules over the network and application ontologies. The advantage of using SWRL is that the rules and the mechanisms are specified in the same language making them easier to write and understand. Operator policies can be changed at run time by simply modifying the appropriate rules resulting in new adaptations without any change to the application or network code. On the other hand, changing adaptations in SILO like architectures may involve recomposition of services to create a new virtual network stack. Let us now consider a sample operator policy for the Video application defined in [21]. Suppose the policy states that *"when the available bandwidth falls below a threshold of three mbps, downsample the quantization level to 15"*. This policy is expressed in our framework as follows

```
  Application(?a) ∧
appBandwidth(?a,?bw) ∧
swrlb:lesserThanOrEqual(?bw,3) ∧
hasProfile(?a,?b) ∧
supportsAdaptation(?b,?c) ∧
increaseQuant(?c) ∧
  → changeQuantTo(?c,15) ∧
 inferredAction(?a,?c)
```

The above SWRL rule is to be interpreted as follows. If application *a* has bandwidth *bw* which is less than three mbps , and *a* is described by profile *b* that supports adaptation, then create an adaptation instance (representing the action to be taken by the application) *c* with quantization level set to 15, and bind this instance *c* to the profile *b* and return it to the application through the inferred action property. The inferred action property essentially represents the actions returned to the application by the reasoner. In this case, the application downsamples the quantization level to 15 which corresponds to lower quality video.

IV. Evaluation

In this section we describe our simulator and demonstrate the utility of our approach through sample applications. We have developed a simulator using freely available tools on top of the popular NS2 simulator. In our simulator, Protege [22] is used as the policy editor for writing and editing SWRL rules as well as ontology modelling. We use Jess [23] as the reasoning engine motivated by its easy integration with Protege. We also wrote a java process that listens for OWL streams from applications, invokes the reasoning engine, and returns the results back to the application for appropriate policy enforcement.

A. Video Adaptation

The goal of this experiment is to demonstrate how application and network adaptations can be used to improve video performance. We created a 1836 frame video sample consisting of videos from the ASU video trace [24] viz. football followed by highway in CIF format encoded at 25fps. In our experiments, we use PSNR as the evaluation metric.

Figure 4 shows the network topology used in our experiments. The video sequence is transmitted between nodes 0 and 4, traversing the bottleneck link between nodes 2 and 3. Cross traffic is simulated by CBR sources between nodes 1 and 5. By varying the rate of the CBR sources, we can control the amount of bandwidth available to the video application. We will now focus on the adaptation capabilities available at the application and network layers.

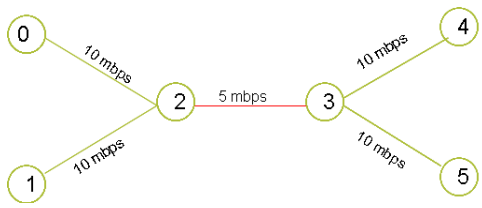

Fig. 4. Topology

1) Application Adaptation Capabilities: The stored CIF files can be encoded at multiple rates resulting in video of differing quality. The encoding rate is specified by the quantization parameter Q that can vary between 2 and 31. Video encoded at quantization level 2 yields the best picture quality video while video encoded at quantization level 31 results in the poorest quality. For our experiments, we modified the Evalvid-RA [25] toolset developed for trace driven simulation of rate adaptive video. Particularly, we exposed the quantization level as an application adaptation parameter which is set by invoking the reasoner. In our experiment, similar to evalvid-ra, the reasoner is invoked at the beginning of a Group of Pictures (12 frames in the sequence IPPPPPPPPPPP) to determine the quantization level. This results in all frames transmitted during a GOP having the same quantization level. Our current policies involve setting the quantization level to either 2 or 15 depending on available bandwidth being greater than or lesser than 3 mbps respectively. We would like to note that the goal of this simple scenario is to illustrate our system architecture and policy specifications rather than showing the efficiency of the adaptation algorithm. Further, as an optimization, we could invoke the reasoner only when there is a significant change in available network bandwidth instead of invoking the reasoner at the start of each GOP.

2) Network Adaptation Capabilities: As an example of a network adaptation capability, we implemented a flow id based packet filter. The filter allows only packets set with a specific flow id to pass through; all other packets are dropped at the filter. The intent here is to ensure the reliable delivery of high priority packets by marking them with the appropriate flowid while saving bandwidth on lower priority packets. Note that the filter is a generic mechanism that can be used with other applications besides video as well.

With no cross traffic, the average PSNR for quantization level 2 was 43.43 while for quantization level 15 was 32.33. Figure 5 shows the PSNR variation in the presence of application and network adaptation when the cross traffic is maintained at 4 mbps. In this experiment, the frame-drop

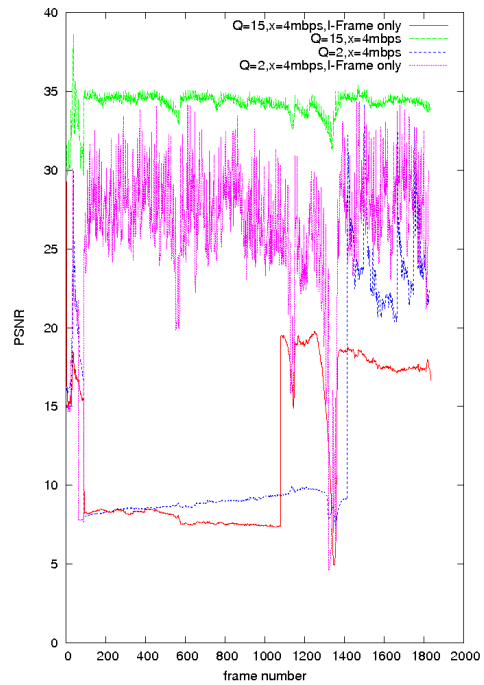

Fig. 5. PSNR Variation with application and network adaptation

filter drops all packets except I-frames. Also, we use a playout buffer of 250 ms at the receiver. The aim of these experiments is to understand the ground truth and design policies appropriately. From figure 5, we can see that the optimal adaptation to perform when the available bandwidth falls below three mbps is to down-sample the quantization level to 15. Also, if a network filter is being employed, we need to ensure that we set the appropriate flow id as well to ensure that no packets are dropped at the filter, else the PSNR would drop drastically. Such a policy would be written in our framework as follows

```
  Application(?a) ∧
appBandwidth(?a,?bw) ∧
swrlb:lesserThanOrEqual(?bw,3) ∧
hasProfile(?a,?b) ∧
supportsAdaptation(?b,?c) ∧
increaseQuant(?c) ∧
  → changeQuantTo(?c,15) ∧
 inferredAction(?a,?c) ∧
 newFlowid(?a,0)
```

In our current filter implementation, packets with flowid 0 are allowed to pass through the filter, while all other packets are dropped. The inferred action object represents the response returned by the reasoner to the application containing actions to be taken by the application. Similarly, when the bandwidth available to the application increases beyond three mbps, the policy should be to up-sample the quantization level to 2 which can be expressed analogous to the above rule.

Figure 6 shows the variation of PSNR with application

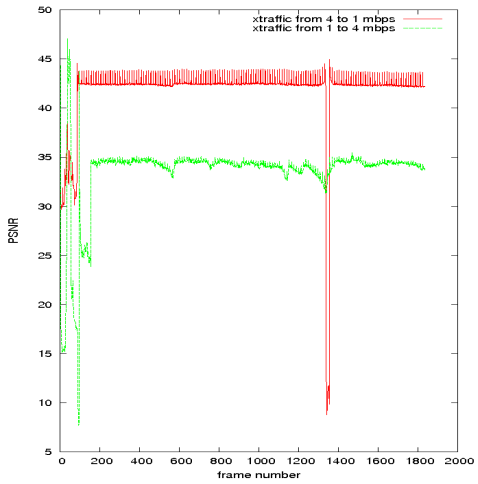

Fig. 6. PSNR Variation with application and network adaptation

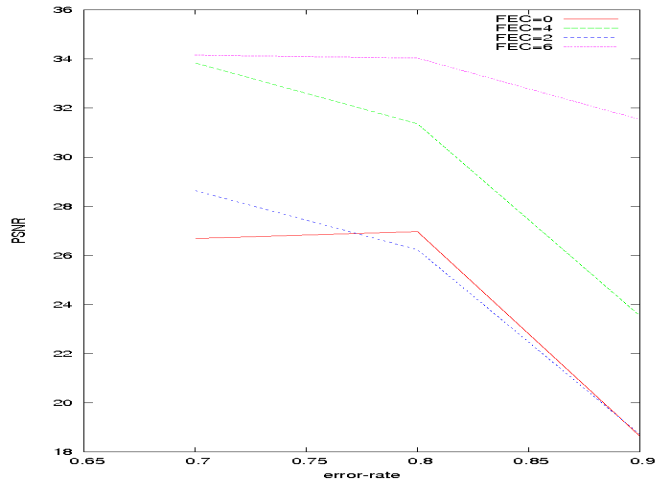

Fig. 7. PSNR Variation with loss rate and FEC strength

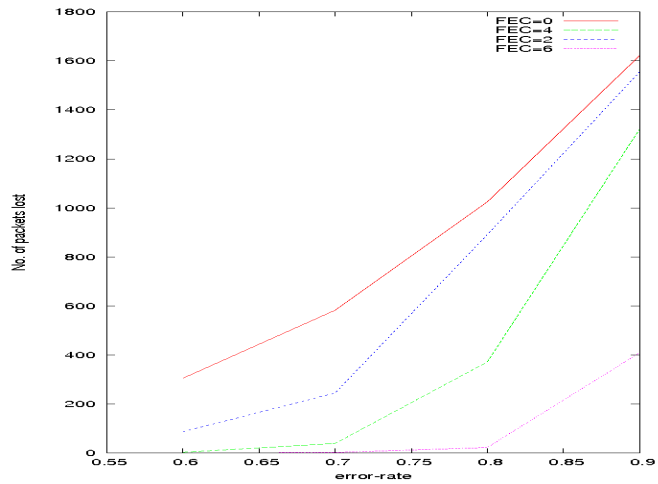

Fig. 8. Loss Rate Variation with error rate and FEC strength

and network adaptation in the presence of varying available bandwidth. In this experiment, the cross traffic is increased (decreased) from 1 to 4 mbps (4 to 1 mbps) two seconds after the video sequence starts. As stated in our policies, when the available bandwidth increases beyond three mbps, the quantization level is to set to 2, resulting in an average psnr of 41.74. This is much higher than the PSNR obtainable at quantization level 15 with no adaptation. Similarly, when the available bandwidth drops below three mbps, the quantization level is set to 15 resulting in an average PSNR of 33.42. These results validate that the quantization level is set in accordance with our policies.

B. FEC Strength over wireless links

In this experiment, we use our framework to set the FEC strength at the wireless MAC layer. We have implemented an FEC algorithm similar to the one described in [26]. Essentially we perform block coding on a group of eight packets. For every block of eight packets, we create additional FEC packets which are assumed to recover any of the above eight packets if lost. If the sum of the number of actual and FEC packets for a block received at the receiver exceeds the block size, all packets in the block are assumed to be received error free. Higher the number of FEC packets, higher the reliability with a corresponding increase in bandwidth usage. In our experiment, we use the same video sequence described earlier encoded at quantization level 15. The video sequence is streamed between two nodes over a 802.11 wireless link with the link bandwidth set to 11 Mbps. In our current implementation, the FEC strength is statically set by the reasoner through an operator specified policy.

Figures 7 and 8 show the variation of PSNR and packet loss with FEC strength and packet error rate. As expected, higher FEC results in higher PSNR and lower packet loss. The exposed FEC mechanism can be used in adaptive applications as follows. A nice example of a policy that involves joint adaptation is one that states that *"when the available bandwidth falls below three mbps, down-sample the video and provide high FEC at the MAC layer"*. The rationale behind such a policy is that during high channel error rates, the packet loss ratio increases with a corresponding decrease in application bandwidth. In such cases, in addition to down-sampling, we need to ensure that the down-sampled video is reliably delivered to the receiver through the error prone wireless channel. The required reliability is achieved through stronger FEC coding. The above policy is stated in our framework as follows

```
  Application(?a) ∧
appBandwidth(?a,?bw) ∧
swrlb:lesserThanOrEqual(?bw,3) ∧
hasProfile(?a,?b) ∧
supportsAdaptation(?b,?c) ∧
increaseQuant(?c) ∧
Queue(?q)∧
  → changeQuantTo(?c,15) ∧
 inferredAction(?a,?c) ∧
```

```
fec(?q,4)
```

V. Conclusion and Future Work

In this paper we have proposed a declarative framework for managing cross layer adaptations in networks. We pursue an ontological approach in which application and network capabilities are encoded in appropriate ontologies and exposed across layers of the network stack. Cross layer optimizations are specified as operator policies representing operator's requirements and preferences. We use semantic web languages in our framework and leverage their inherent capabilities for evolution, reasoning and conflict resolution. In ongoing work, we are exploring complex scenarios that require richer context than bandwidth and can leverage existing ontologies such as SOUPA [27], thereby demonstrating the power of our ontological approach to seamlessly extend the system. For example, in an emergency care scenario, depending on the current medical context we may want to allocate higher bandwidth to vital signs compared to facial imagery and vice versa. Such context specific policies can be easily represented in our framework. Also, with regards to scalability of reasoning, we plan on exploring the use of predefined "adaptation templates" optimized for different applications and networking technologies. The idea is to choose the appropriate template at the beginning of an application and invoke the reasoner only for certain registered network events. We are also working towards extending our system to handle distributed policies which require cooperation among multiple nodes. Such distributed policies are prevalent in wide area routing such as BGP where each node is free to choose its local routing policy. Furthermore, in these scenarios security and trust among nodes is an important issue that needs to be addressed. The adaptation agent at each node could use existing trust relationships to negotiate with neighboring nodes and arrive at a globally acceptable configuration.

Acknowledgment

We are grateful to the reviewers for their useful comments. This work was supported in part by DARPA under contract W31P4Q-06-C-0395.

References

[1] "100x100 clean state project." [Online]. Available: http://www.nets-find.net/index.php

[2] "Nsf future internet design." [Online]. Available: http://www.nets-find.net/index.php

[3] R. Dutta, G. N. Rouskas, I. Baldine, A. Bragg, and D. Stevenson, "The silo architecture for services integration, control, and optimization for the future internet," in *IEEE ICC*, 2007, pp. 24–27.

[4] I. Baldine, M. Vellala, A. Wang, G. Rouskas, R. Dutta, and D. Stevenson, "A unified software architecture to enable cross-layer design in the future internet," Aug. 2007, pp. 26–32.

[5] B. T. Loo, T. Condie, J. M. Hellerstein, P. Maniatis, T. Roscoe, and I. Stoica, "Implementing declarative overlays," *SIGOPS Oper. Syst. Rev.*, vol. 39, no. 5, pp. 75–90, 2005.

[6] B. T. Loo, J. M. Hellerstein, I. Stoica, and R. Ramakrishnan, "Declarative routing: extensible routing with declarative queries," in *SIGCOMM '05: Proceedings of the 2005 conference on Applications, technologies, architectures, and protocols for computer communications*. New York, NY, USA: ACM, 2005, pp. 289–300.

[7] A. Singh, T. Roscoe, P. Maniatis, and P. Druschel, "Using queries for distributed monitoring and forensics." [Online]. Available: citeseer.ist.psu.edu/singh06using.html

[8] B. T. Loo, T. Condie, M. Garofalakis, D. E. Gay, J. M. Hellerstein, P. Maniatis, R. Ramakrishnan, T. Roscoe, and I. Stoica, "Declarative networking: language, execution and optimization," in *SIGMOD '06: Proceedings of the 2006 ACM SIGMOD international conference on Management of data*. New York, NY, USA: ACM, 2006, pp. 97–108.

[9] S. B. Kodeswaran and A. Joshi, "Content and context aware networking using semantic tagging," in *ICDEW '06: Proceedings of the 22nd International Conference on Data Engineering Workshops*. Washington, DC, USA: IEEE Computer Society, 2006, p. 77.

[10] S. B. Kodeswaran, O. Ratsimor, A. Joshi, and F. Perich, "Utilizing Semantic Tags for Policy Based Networking," in *Globecom 2007 (accepted for publication)*, November 2007.

[11] P. A. Kodeswaran, S. B. Kodeswaran, A. Joshi, and F. Perich, "Utilizing semantic policies for managing BGP route dissemination," in *Automated Network Management (INFOCOM workshops)*, April 2008.

[12] A. Chockalingam and M. Zorzi, "Wireless tcp performance with link layer fec/arq." [Online]. Available: citeseer.ist.psu.edu/431529.html

[13] R. Kapoor, M. Cesana, and M. Gerla, "Link layer support for streaming mpeg video over wireless links," 2003. [Online]. Available: citeseer.ist.psu.edu/kapoor03link.html

[14] Y. Shan and A. Zakhar, ""cross layer techniques for adaptive video streaming over wireless networks"," in *ICME Multimedia and Expo.*, 2002, pp. 277–280.

[15] L.-U. Choi, M. T. Ivrlac, E. Steinbach, and J. A. Nossek, "Bottom-up approach to cross-layer design for video transmission over wireless channels," in *62nd IEEE Vehicular Technology Conference (to be published)*, September 2005.

[16] B. D. Noble, M. Satyanarayanan, D. Narayanan, J. E. Tilton, J. Flinn, and K. R. Walker, "Agile application-aware adaptation for mobility," *SIGOPS Oper. Syst. Rev.*, vol. 31, no. 5, pp. 276–287, 1997.

[17] P. Chandra, Y.-H. Chu, A. Fisher, J. Gao, C. Kosak, T. E. Ng, P. Steenkiste, E. Takahashi, and H. Zhang, "Darwin: Customizable resource management for value-added network services," *IEEE Network*, vol. 15, no. 1, January 2001.

[18] I. Horrocks, P. F. Patel-Schneider, H. Boley, S. Tabet, B. Grosof, and M. Dean, "Swrl: A semantic web rule language combining owl and ruleml," W3C Member submission 21 may 2004, Tech. Rep., 2004. [Online]. Available: http://www.w3.org/Submission/SWRL/

[19] "Shared spectrum company. xenoglaux - tiny owl reasoner." [Online]. Available: http://www.sharedspectrum.com/

[20] "Network Ontology." [Online]. Available: http://ebiquity.umbc.edu/resource/html/id/275/NetworkOnto

[21] "A video application ontology." [Online]. Available: http://www.umbc.edu/ palanik1/adaptation.owl

[22] "The protege toolkit." [Online]. Available: http://protege.stanford.edu/

[23] "Jess rule engine,http://herzberg.ca.sandia.gov/links/." [Online]. Available: http://herzberg.ca.sandia.gov/links/

[24] "Video traces research group." [Online]. Available: http://traces.eas.asu.edu/

[25] A. Lie and J. Klaue, "Evalvid-ra: trace driven simulation of rate adaptive mpeg-4 vbr video," *Multimedia Systems*, vol. 14, pp. 33–50, 2007.

[26] C.-H. Lin, C.-H. Ke, C.-K. Shieh, and N. K. Chilamkurti, "An enhanced adaptive fec mechanism for video delivery over wireless networks," *Networking and Services, International conference on*, vol. 0, p. 106, 2006.

[27] H. Chen, T. Finin, and A. Joshi, *The SOUPA Ontology for Pervasive Computing*, ser. Whitestein Series in Software Agent Technologies. Springer, July 2005.