Experience with Automated Planning for Panda

Alexey Rudenko and Peter Reiher
{arudenko, reiher}@cs.ucla.edu

Computer Science Department
University of California, Los Angeles
Technical Report CSD-TR 010041

Abstract - *Open network architectures (ONA) allow dynamic deployment of services in the networks. The Internet would benefit from quick deployment of protocols that provide customized services to handle user data streams. Complexities and dynamism of network conditions make it unreasonable to expect users and application writers to foresee and handle all possible problems. Having network systems automatically adjust to those problems would be a great improvement, but doing so clearly indicates a need for automated planning of services in ONA. Panda is an ONA system that is based on the Active Network paradigm and is capable of automated planning for peer-to-peer UDP connections.*

This paper uses experimental results describing the overheads related to automated planning in Panda and the benefits Panda can achieve for real user multimedia applications.

1. Introduction

Open network architectures allow dynamic deployment of services in routers or special servers located in the networks. While traditional networks passively transport bits from one end system to another, ONA technology allows networks to deploy adaptations dynamically. However, mapping ONA services into the connection routers is a difficult problem. User applications might not be aware of current conditions of highly variable networks; the connection routers are aware of network conditions, but they are not aware of the application requirements to the connection protocol. Since users cannot always predict the best way to use the open architecture for their data streams under the current conditions, ONA require full-scale automated planning to efficiently map adaptations into the ONA-enabled nodes used by a connection. Planning should guarantee that adaptations are compatible and operate effectively together. Planning should also use adaptations efficiently because their execution uses limited computing resources at intermediate nodes and increases the latency of data delivery. Unnecessarily repeating a particular adaptation or improperly locating it is highly undesirable.

The planning protocol implemented in Panda consists of three consequent phases

- planning data gathering
- calculation of a plan
- deployment of the plan.

There are two major planning protocols implemented in Panda: incremental planning and centralized planning.

Panda source node intercepts user application data packets, converts them into Panda packets and further forwards them from node to node and adapts them according to a plan. Panda destination node converts the packets back to their original state and forwards them to the receiving site of the user application using the correspondent port.

Incremental planning occurs on all nodes of a connection. When a source node intercepts a user's application data packets it locks the data stream so that data packets are buffered on the source Panda node. The source node calculates and deploys the local plan for the link between itself and second node. Then the second node calculates and deploys a plan for the link between itself and the third node of the connection and so on, until a local plan for the link leading into the destination node itself is calculated and deployed. The destination node informs the source node that local plans are ready; the source node unlocks the data stream and data transfer starts.

In centralized planning the plan calculation occurs on the source node of a connection. When a source Panda node intercepts user application data packets that are initiated on the same node, it locks the stream and accumulates its packets in a buffer. A message from the source to the destination collects the planning data on all nodes of the connection and returns this data to the source node. Then centralized planning runs on the source node. The adapters chosen by the plan are deployed on the proper nodes of the connection. The source node sends messages to all nodes that should run particular adaptations. If a node does not have a particular adaptation it asks the source node to deliver it. The source node sends the adapter to the node and the node sends the acknowledgement back to the source node. When the source node obtains acknowledgements from all nodes of the connection it unlocks the stream and data packets are sent to the destination. The protocol is presented on Figure 1.

After the stream is unlocked the packets pass through all adapters that are deployed on the nodes of the connection. The adaptations process the packets and forward them further along the connection.

If the conditions of the connection change Panda re-runs the planning process again to adjust to the new conditions. This process is called *re-planning*.

Incremental planning is quick and inefficient; centralized planning is slow and more efficient. In Panda both planning algorithms run concurrently. When incremental planning is done data packets start to be forwarded to the destination. When the central plan is calculated and deployed the data stream switches to the improved plan.

The first point of the interest is the overhead of the planning protocol:

- Latency of incremental planning
- Latency of centralized planning data gathering
- Latency of centralized plan calculation
- Latency of centralized plan deployment.
- Latency of re-planning
- Number of packets sent under the incremental plan before the centralized plan is calculated and deployed

Other points of interest are Panda overheads not related to planning:

- Latency of a packet delivery from the source node to the destination node
- Latency of adaptation.

The ultimate measure of Panda and planning is the quality of service that was achieved using adaptations compared to an Internet connection that is unable to adapt user data.

The rest of the paper is organized as following. Section 2 presents the description of the testbed used, section 3 presents the latency of user packet delivery, section 4 presents adaptation latency, section 5 presents the overhead of the planning protocol, section 6 presents the QoS test results, section 7 discusses the results, and section 8 offers conclusions.

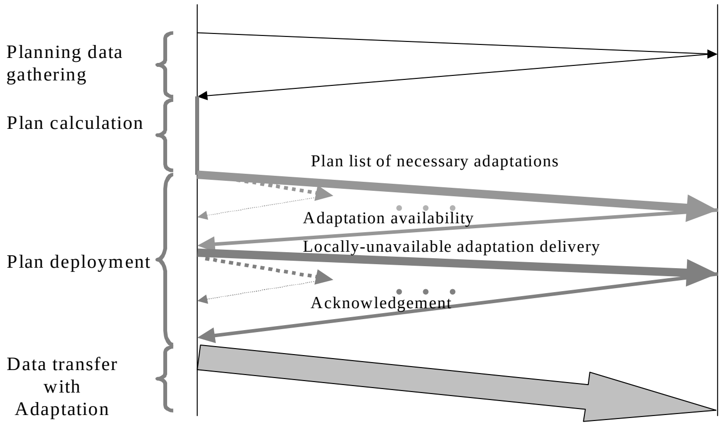

Figure 1: Planning protocol and data transfer on the source Panda node

2. The Testbed

2.1 Computers and Networks

The connection was tested with twisted pair sequential connections of up to four computers as shown on Figure 2. Dell Inspiron Omnibook 4150 laptops with 333 MHz processors were used for one set of tests and Hewlett Packard laptops with 500 MHz processors for another set of tests; all machines used Linux Red Hat 7.0 with the 2.2.16 kernel. Xircom RealPort2 Ethernet 10/100 pcmcia cards were used for the network connection between the machines. The source and destination machines run a user application and the Panda node concurrently. The priority of the user application was set lower on the source machine and higher on the destination machine to ensure proper allocations of resources.

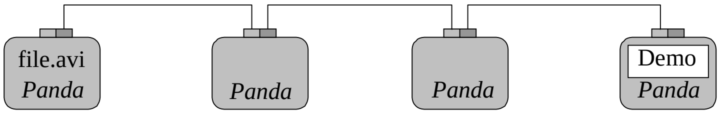

Figure 2: Panda peer-to-peer connection

Throughput of the network links is varied among 150 Kbps, 800 Kbps, 2000 Kbps, and 5000 Kbps using CBQ.

2.2 Adapters

We used two kinds of adaptations: *null adaptations* and real adaptations. Null adaptations do not perform any data processing; they are used to measure the overhead of just to having an adapter in a connection. Filters and encryption were used as real adaptations. The filters drop particular packets with color or quality data and computationally are very economic; the encryption adapter performs heavyweight processing of the data.

2.3 The Problem of Synchronization

The following method was applied to measure one-way packet delivery. The packets were stamped with the local time on the source machine. Upon the arrival at the destination machine the stamped time was subtracted from the destination local time to obtain *measured time delivery*. The synchronization of the source and destination machines' clocks was done with NTP. The NTP server was located on the destination node. The source node synchronized itself to the destination local time before the first packet was sent to the destination. Then 20,000 packets were sent the destination. After the last packet was delivered, the source machine measured the *skewing value*. It was presumed that skewing grows uniformly by time. The *actual time delivery* was calculated with formula for each data packet *n*:

$$ActualTimeDelivery(n) = measuredTimeDelivery(n) - \frac{skewingValue}{20{,}000} \bullet n$$

2.4 Applications

Three different applications using the UDP protocol were used for the performance tests. The latency of the packet delivery and null-adaptations were tested on a special application called *Connector* that was designed in Java for this purpose. Connector is able to generate data packets of different size.

The overhead of the planning protocol and real life adaptations was tested with the WaveVideo application [Frankhauser99], which generated a video stream using .avi files.

As an alternative to this video stream application, an audio stream generating application RAT was used, which generated audio stream using .au files.

The quality of service was tested with the WaveVideo measurement package that compares the initial data stream with the one that was actually delivered. The result is presented in PSNR units, which are the ratio of the initial stream to the error that occurred during the transmission.

3. Packet Delivery and Adaptation Latency

Figure 3 presents packet delivery latency for different packet sizes. Panda without adaptations extends normal Internet latency 3-4 times, being a relatively slow Java application. Null adapters added to the connection make Panda overhead even heavier for packet delivery. The packet delivery latency also contains the adaptation latency. Error bars on this figure and all further figures show the value of standard error, unless otherwise indicated.

Figure 3 shows that adding Panda to a data stream increases its latency 50-150%, with longer packets seeing less effect. Adding more Panda-enabled nodes or more adapters modestly increases the delay for each addition.

Figure 4 presents the latency of null adaptations. All adaptations were deployed on one of the nodes of the connection. Of course, without Panda no adapters can be deployed, so the extra latency for that case is defined to be zero. Every Panda node always runs at least one *forward* adapter, whose only task is to forward a packet to a next node after all other adapters are executed. The number of forward adapters equal to the number of connection nodes is always present in a Panda connection but it is not counted on our graphs.

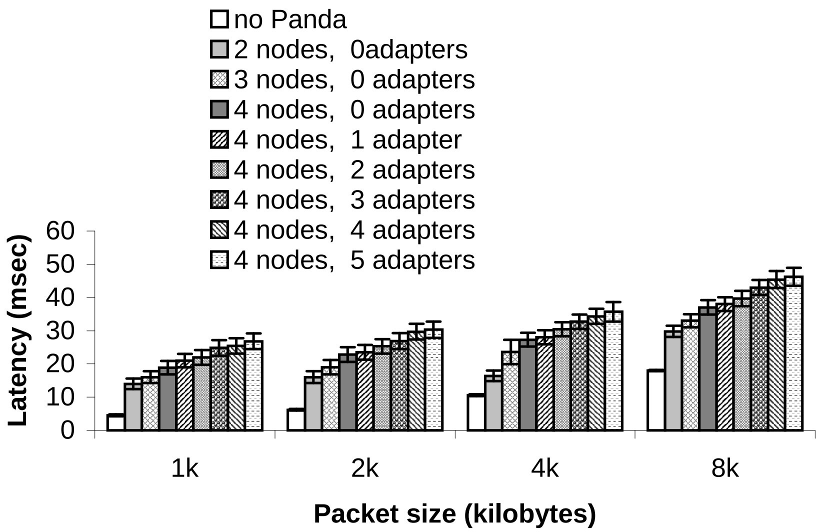

Figure 3: Packet delivery latency

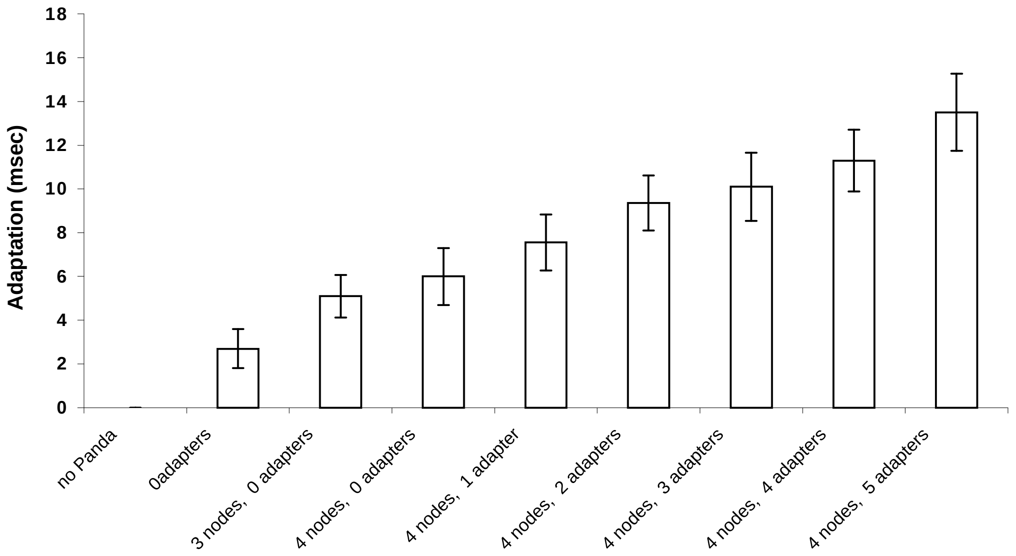

Figure 4: Adaptation latency

Figure 5 presents the packet loss that occurred in the data stream of 20,000 packets tried for different packet size. The data stream without Panda had no packet loss. No packet for 2k-packet data stream was lost either. Packet loss increases with packet size because of extra memory allocation by the Panda Java code. Figure 6 shows that Panda throughput grows with the packet size. At the same time Panda packet loss grows with the packet size, but it never reaches more than 0.5%. The packet size of multimedia applications varies anyway because some applications apply their own compressing protocols to the data packets. Error bars represent 95% confidence intervals.

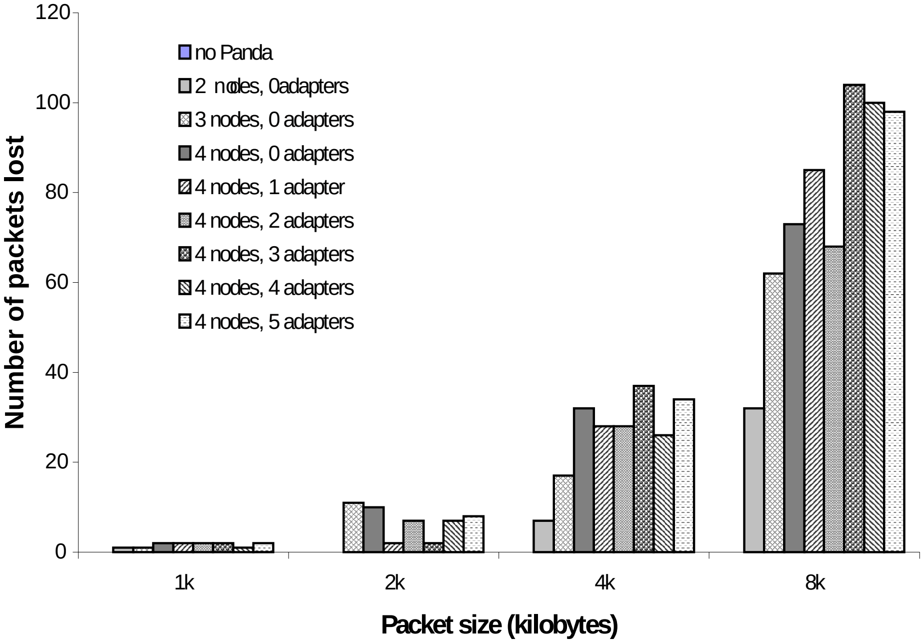

Figure 5: Packet loss

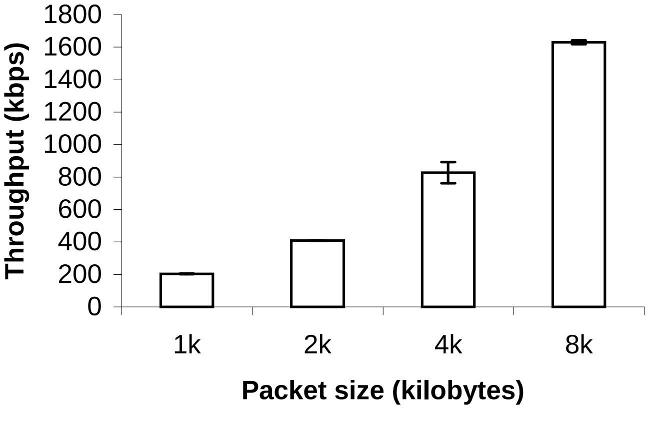

Figure 6: Throughput of Panda associated with packet size

Figure 7 presents the sample of 1000-packet latency distribution. The stream occurred on the connection of 4 Panda nodes without adapters for 1k packets.

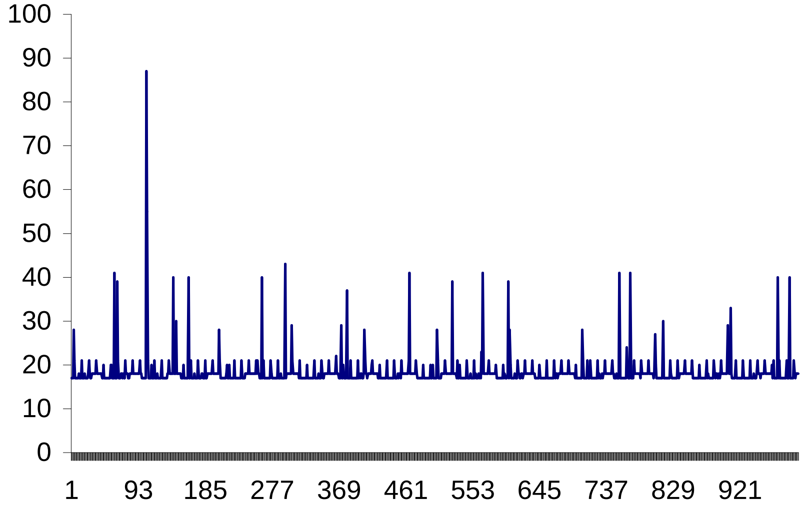

Figure 7: Sample of the distribution of packet delivery latency on packet numbers.

All Figures 3-7 are obtained running the Connector application and null adapters.

4. Planning Procedure Latency with the Connector Application and Null Adapters

The planning procedure latency consists of planning data gathering latency, plan calculation latency, and plan deployment latency. Planning data gathering takes one round trip; the source node forwards the data gathering message to the end node and waits for its return. Planning data gathering throughout four Panda nodes takes 108 +/- 2.85 milliseconds.

For centralized planning, the time required to deploy the plan depends on whether the adaptations are pre-loaded on nodes. Obviously, if adaptations are pre-loaded the deployment latency is much shorter. Figure 8 presents the deployment latency for the case when adapters are not preloaded. The bars represent the deployment latency of 1-5 null-adapters that were deployed on each of the connection nodes. The deployment on Node #1 is always fast because it is the source node, the storage site of all adaptations. In centralized planning, the more adaptations that must be transmitted to remote nodes, the longer the deployment process takes.

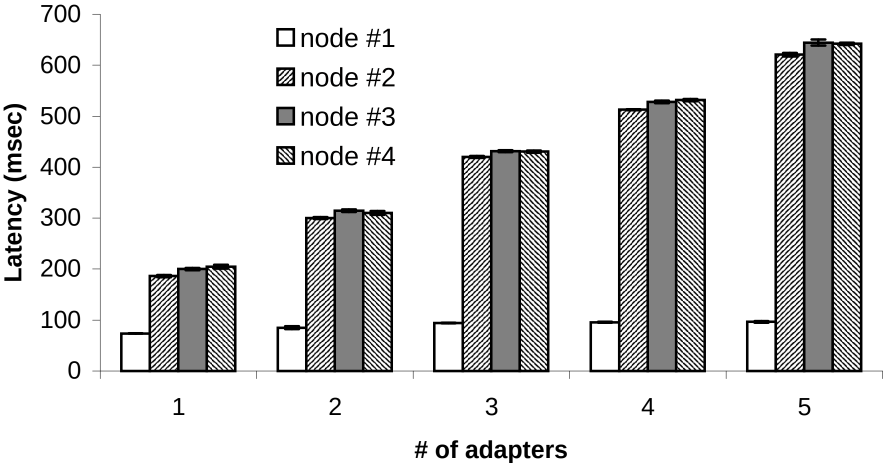

Figure 8: Deployment without pre-loaded adapters

Figure 9 presents the deployment in case of pre-loaded adapters. The latency of deployment is much shorter in this case because adapters need not to be transmitted to remote nodes. However, the deployment protocol without adapter transmission still must be performed completely, and that is why the deployment of more adapters takes longer per node.

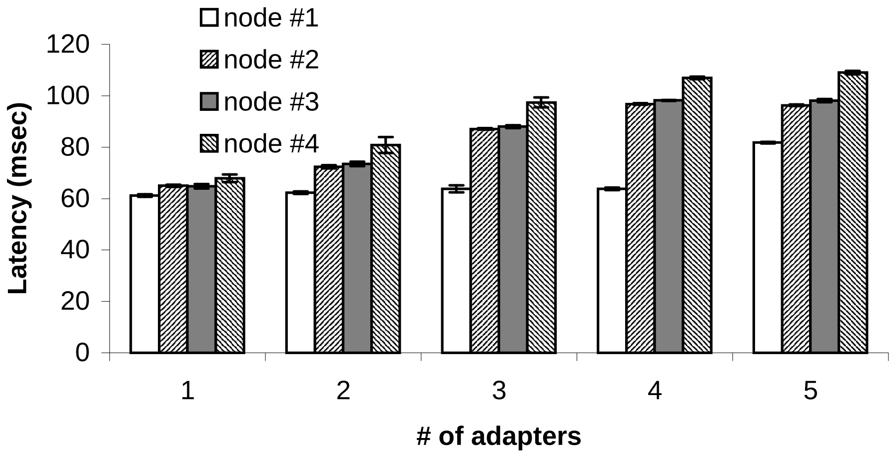

Figure 9: Deployment with pre-loaded adapters

Figure 10 presents the latency of the deployment protocol when no adapters are selected. In this case the deployment protocol consists of the querying messages sent by the source node to the intermediate nodes asking them if they are ready to receive user data and their acknowledgements.

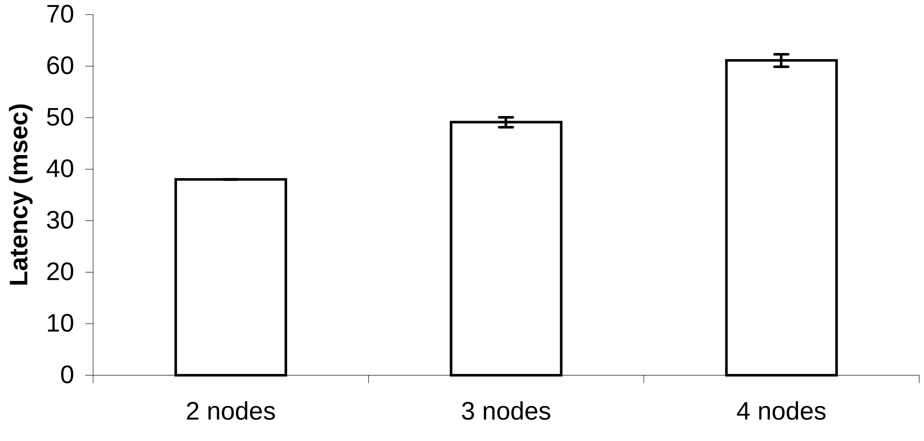

Figure 10: Latency of the deployment protocol without adaptations

Figure 11 presents the latency of the incremental planning process. The bars for deployment on planning machine refers to adaptations that must be deployed on the machine that calculates the local plan. The bars for deployment on the next machine refers to the adapters that must be deployed on the machine downstream from the planning machine. The bars for deployment on both machines refer to cases where the incremental plan requires adapters on both of these machines. Incremental planning does not include any adaptation transmission. All nodes are presumed to be storing all adaptations that can be chosen by their local planners.

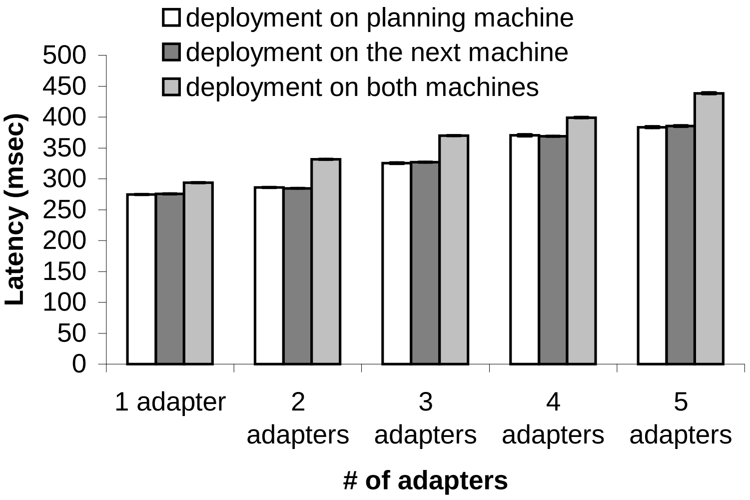

Figure 11: Incremental planning and deployment latency

Figure 12 presents the latency of performing both centralized and incremental planning. The bars marked "Incremental" show the latency of the initial incremental plan. The bars marked "Central" show the latency of the planning procedure if no incremental planning occurs. The bars marked "Central plan with incremental plan in the background" show how incremental planning slows the centralized planning. Once the incremental plan is established at all nodes, data packets start to flow. These packets compete with the business of centralized planning, slowing that procedure down.

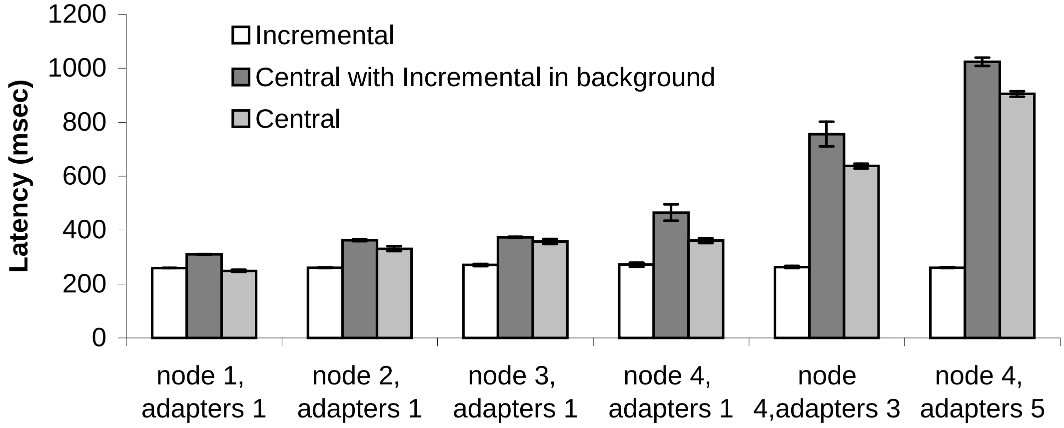

Figure 12: Incremental versus centralized planning latency

Figure 13 shows with the number of packets that are forwarded under the incremental plan before the central plan is calculated and deployed. The bars marked "node 1, 2, 3 or 4 adapters 1" demonstrate the cases when centralized plan requires the deployment of one adapter on 1st-4th nodes respectively. The further the adapter must be deployed from the source node, the longer process of the deployment, and, thus, more packets are sent under the incremental plan. The bars marked "node 4, adapters 1, 3, 5" demonstrate the cases when the centralized plan requires the deployment of 1, 3 or 5 adapters respectively on node 4. The more adapters must be deployed, the longer the deployment process lasts, and, thus, the more packets are sent under the incremental plan. The graph suggests that very short data streams, for example NTP, may not require central planning, as all of their messages will be delivered before the centralized plan is calculated and deployed.

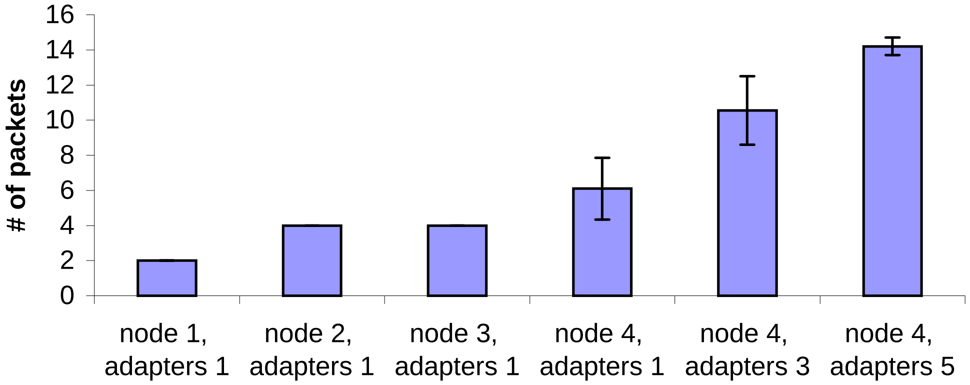

Figure 13: The number of packets sent under incremental plan before central plan is calculated and deployed

Sometimes the conditions of networks change when a connection is already established. If the changes are dramatic enough, the plan is no longer effective, and the system must replan. The process of re-planning runs concurrently with the data packet stream and therefore takes longer than the initial planning. Figure 14 presents the latency of re-planning. The transparent bars show the latency of the initial planning process, where one adapter is deployed on the third machine. The gray bars show the latency of re-planning, where one adapter is deployed on the first, the second, the third, and the forth machines respectively. Replanning takes at least 50% longer than the initial centralized planning. Replanning process where an adapter is deployed on the source machine still takes longer than the initial planning. It happens because packet storing and forwarding on the source machine takes longer than the packet storing only in the initial centrlized planning, and the traffic of the data packets still delays the packet exchange of the planning protocol. The graph also shows that the further an adaptation must be transmitted from the source node, the longer it takes to complete re-planning.

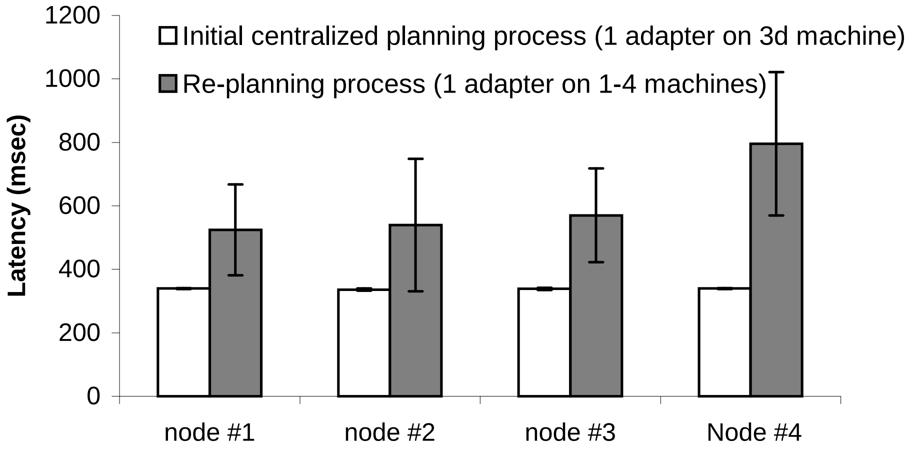

Figure 14: The latency of re-planning process

5. Planning Procedure Latency with Real Life Applications and Adaptations

The results in the previous section used the artificial Connector application and null adapters, and were conducted on Dell Inspiron machines. The following test results used the WaveVideo application and real adaptations, *ResolutionDrop* and *Encryption*, on Dell Inspiron and Hewlett Packard machines.

Figure 15 presents the centrlized planning procedure latency for both the Connector and the WaveVideo applications. Of cource the Resolution Drop adapter was not meaningful for the Connector data packets, but it was not an obstacle to use it for planning procedure measurements. The WaveVideo application generates data packets ten times as fast as the Connector application. This intensity puts extra burden on the CPU of the source node and suppresses Panda activity. Thus, the resource requirements of the user application influence the performance of the Panda. Figures 15, 16 and 17 demonstrate the planning procedure latency, plan calculation latency, and deployment latency respectively. Figure 16 shows that the plan calculation is strongly influenced by

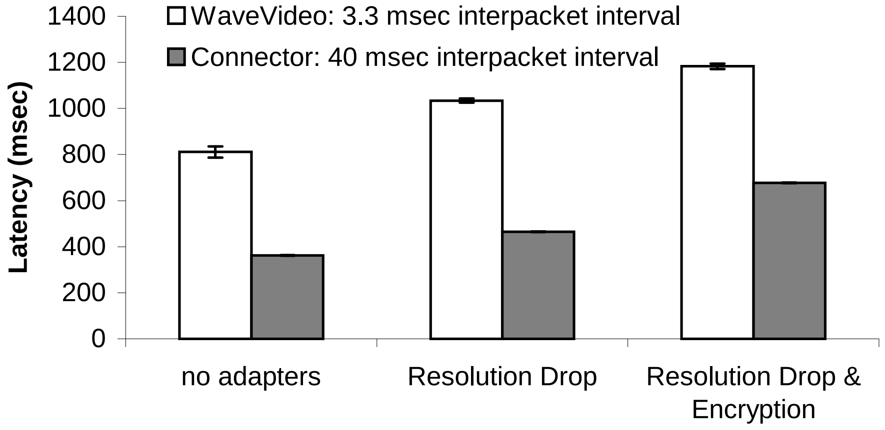

Figure 15: Planning procedure latency for the Connector and the WaveVideo applications

the requirements of the user application. Figure 17 shows that the deployment latency is almost unaffected.

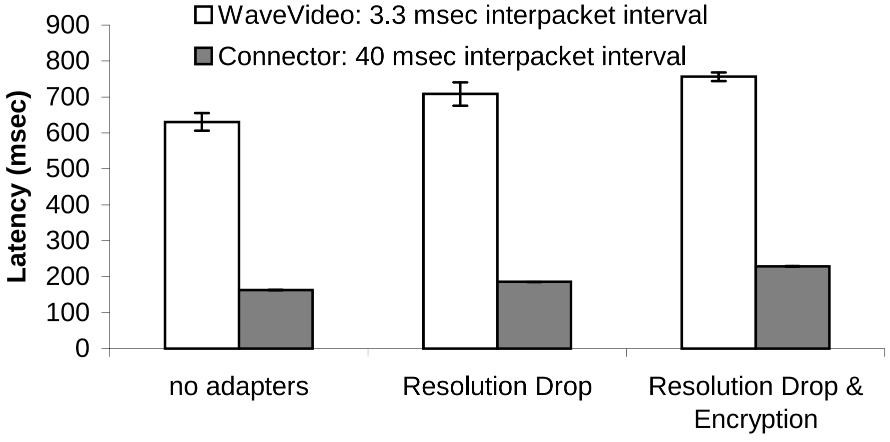

Figure 16: Plan calculation latency for the Connector and the WaveVideo applications

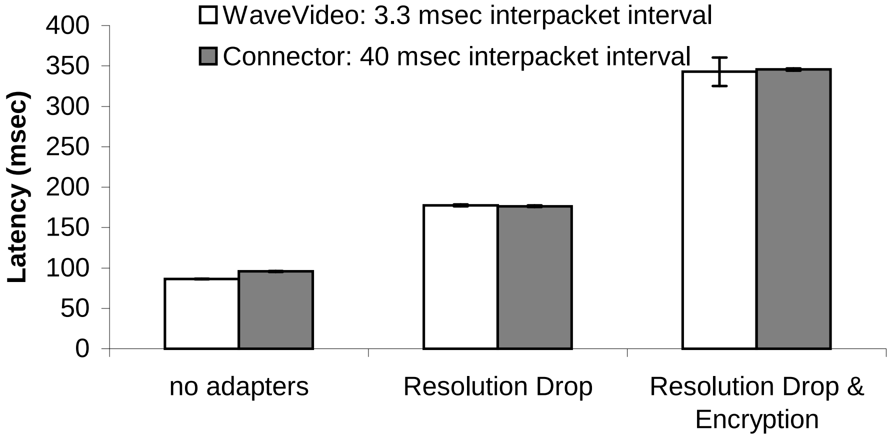

Figure 17: Deployment latency for the Connector and the WaveVideo applications

Figures 18-20 present the latencies for the planning procedure, plan calculation, and deployment respectively for the WaveVideo application for different network bandwidth that varies with different CBQ settings.

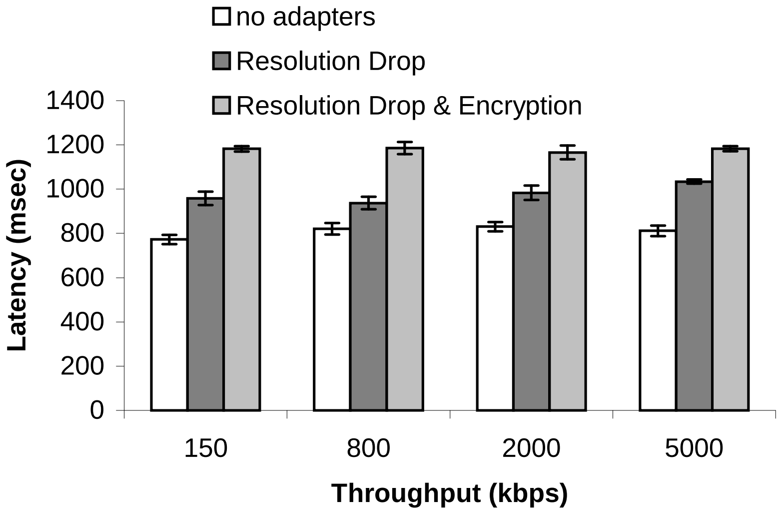

Figure 18: Planning procedure latency with Dell Inspirons

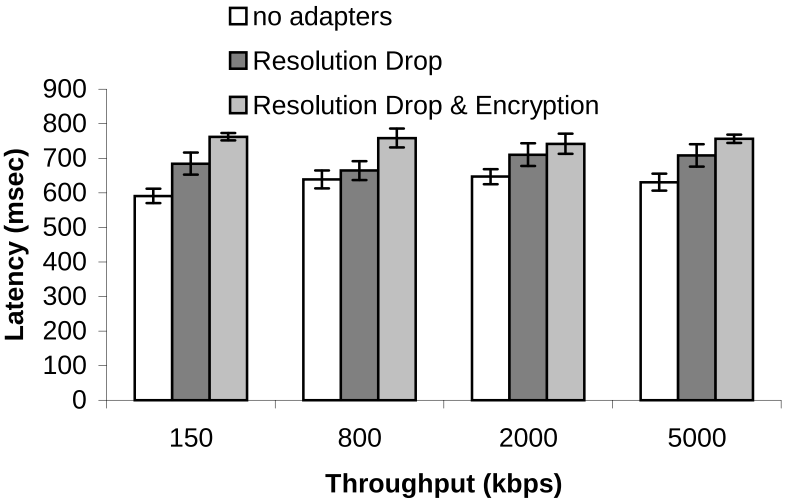

Figure 19: Plan calculation latency on Dell Inspirons

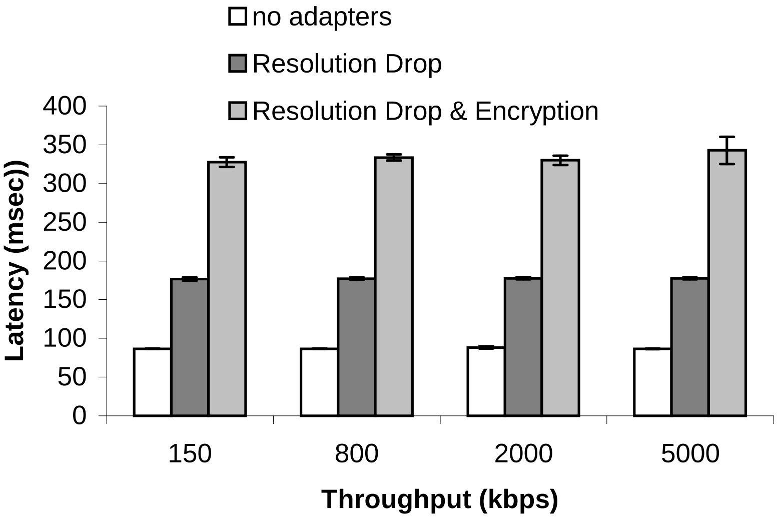

Figure 20: Deployment latency on Dell Inspirons

The graphs show little dependency of latencies for planning procedure on the network bandwidth, but strong dependance on the number of adaptations.

Figure 21 presents the incremental planning latency for the WaveVideo application.

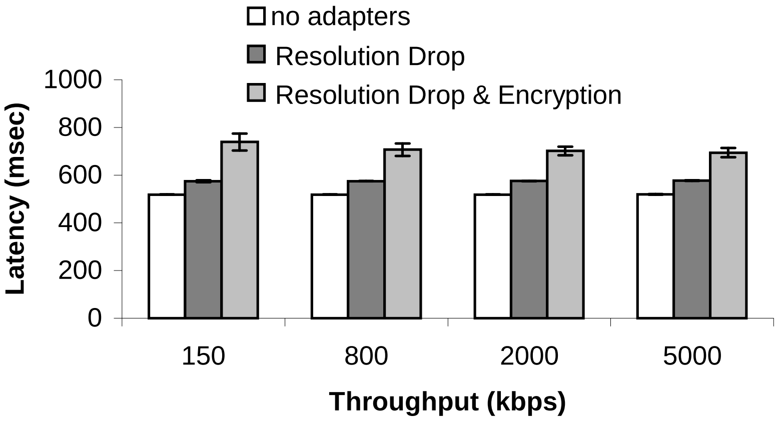

Figure 21: Incremental planning latency for Dell Inspirons

Figure 22 presents the latency of re-planning incremental plan with central plan.

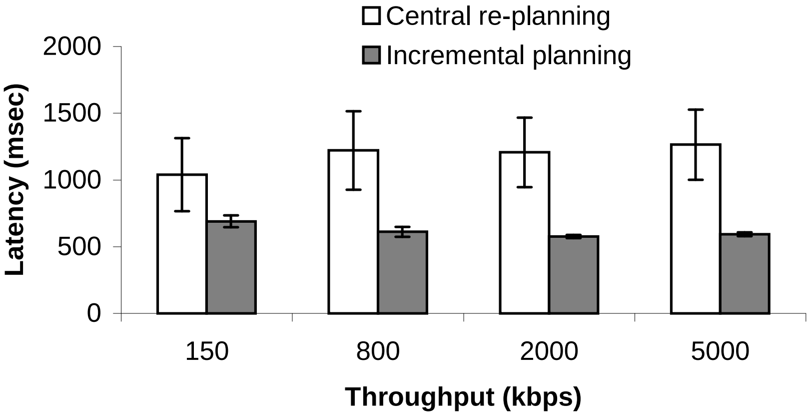

Figure 22: Incremental and central planning procedures (Resolution Drop only) on Dell Inspiron connection

The gray bars on Figure 22 are the same as the Resolution Drop bars on the Figure 21. The throughput has little effect on the latency because the Resolution Drop adapter is very small, and can be quickly deployed regardless of the throughput.

Figure 23 presents the number of packets that were sent under the incremental plan before the central plan was calculated and deployed. The latency of the central planning procedure increases the number of the packets. The number of the packets shows that very short sessions that transmit a small number of packets require incremental planning only.

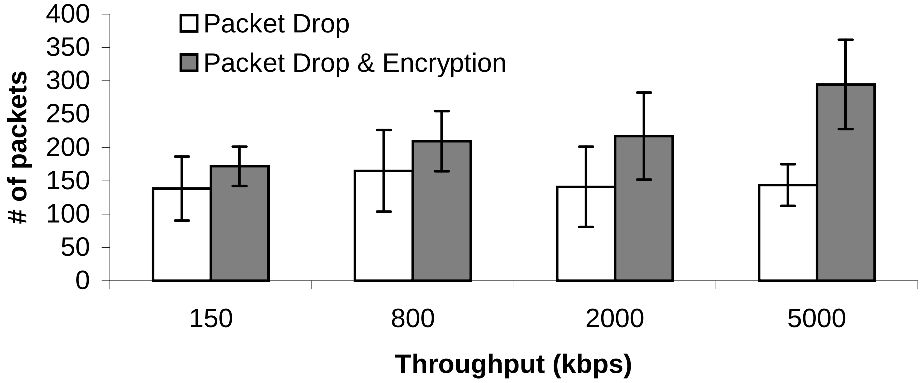

Figure 23: The number of packets sent under the incremental plan before the central plan is calculated and deployed

The error bars on the graph are 95% confidence intervals.

Figure 24 presents the re-planning that occurs during the session.

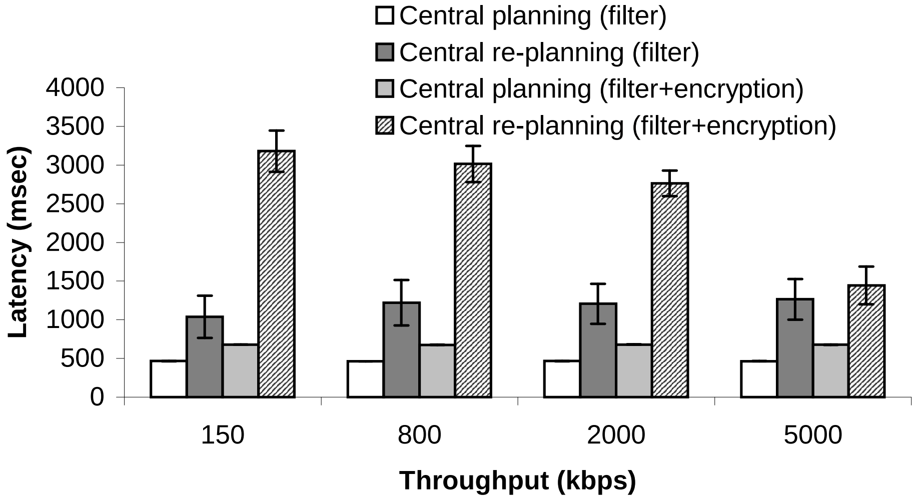

Figure 24: Central re-planning process on Dell Inspirons

Initial central planning represents the planning that occurred before the first packet has sent without performing incremental planning. Central re-planning occurs in the middle of the session concurrently with data packets. The encryption adapter is a relatively large piece of code and its deployment is seriously affected by the limited throughput of the connection; the re-planning procedure lasts from 1.5 seconds for 5000 Kbps to 3 seconds for 150 Kbps.

The next series of experiments were run with more powerful HP Omnibooks (500 GHz of HP versus 333 HGz of Dell Inspirons) to determine the effects of processor power on planning and adaptation.

Figure 25 presents the latency of the adaptation with real adapters on both the Dell and HP machines. Inspiron (null-adapters) bars represent the adaptation latency with 0, 1 and 2 null adapters on Inspiron, which is compared with realistic cases. This figure shows that processing power has a major effect on the cost of running realistic adaptations as would be expected.

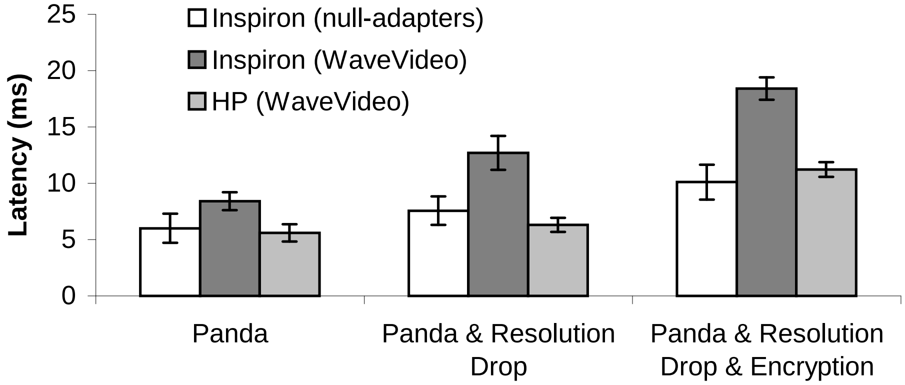

Figure 25: The comparison of adaptation latencies on Dell Inspiron and Hewlett Packard machines connections

Figures 26-28 compares the planning procedure, plan calculation, and the plan deployment latencies for Dell Inspiron and Hewlett Packard machines.

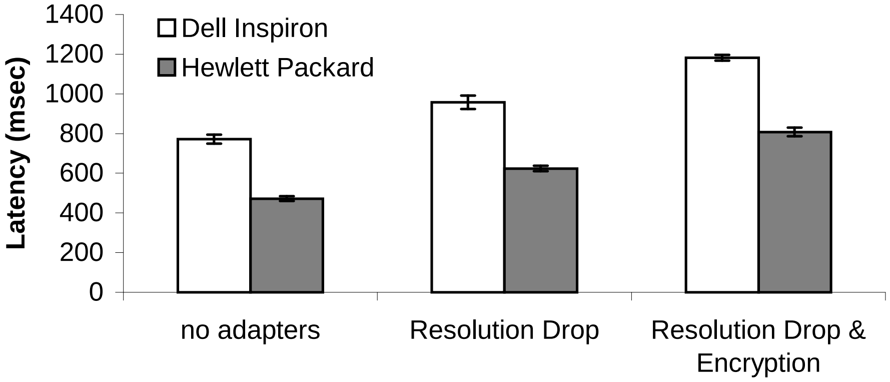

Figure 26: The planning procedure latency on Dell Inspirons and HPs

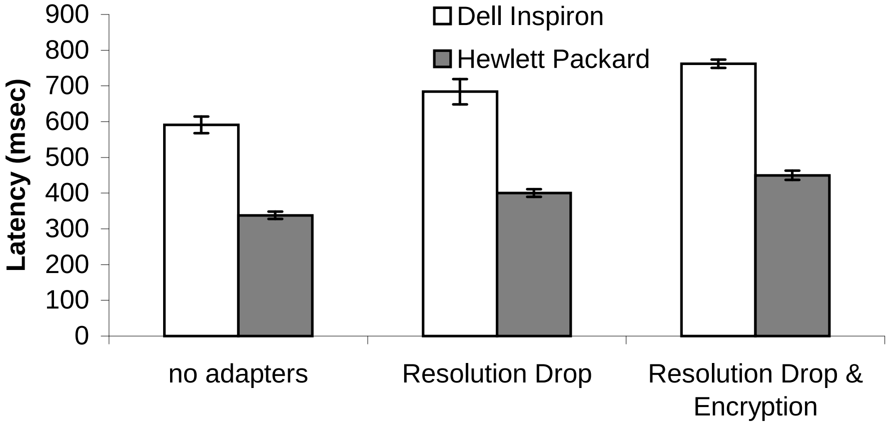

Figure 27: The plan calculation latency on Dell Inspirons and HPs

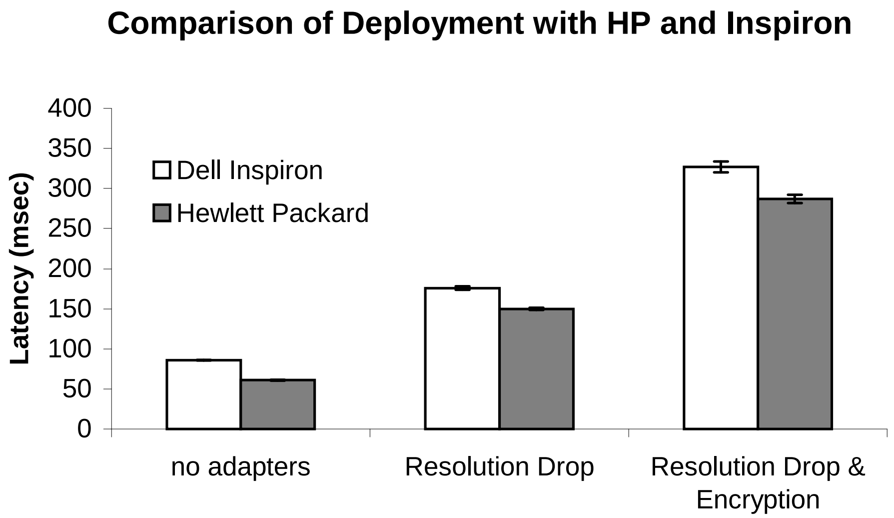

Figure 28: Deployment latency on Dell Inspirons and HPs

These figures show that planning is a CPU intensive activity that can be assisted by more powerful machines. Much of the costs of deployment, however, are more dependant on the network than on CPU, so increasing CPU power provides less benefit in this stage.

The rest of tests were run only on Hewlett Packard machines. The planning data gathering procedure took 72 +/- 6 milliseconds for all situations. Figures 29-31 present the planning procedure, plan calculation, and plan deployment latencies.

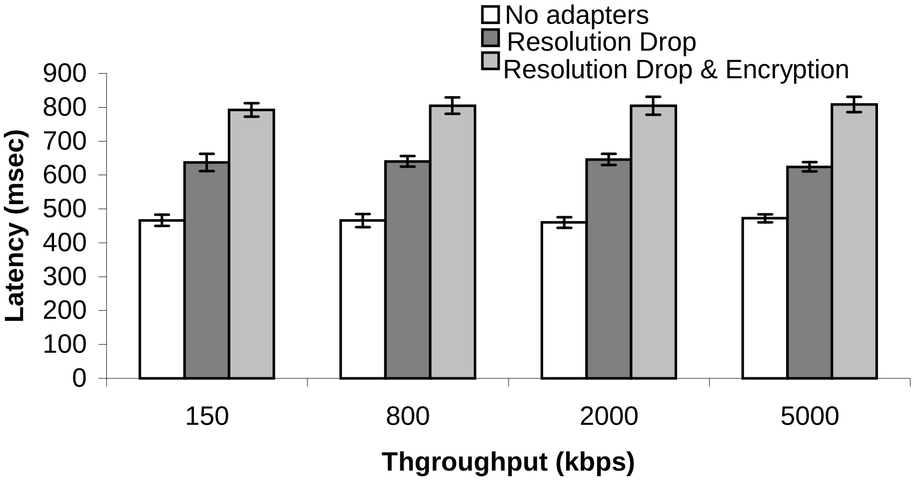

Figure 29: The planning procedure latency on HPs

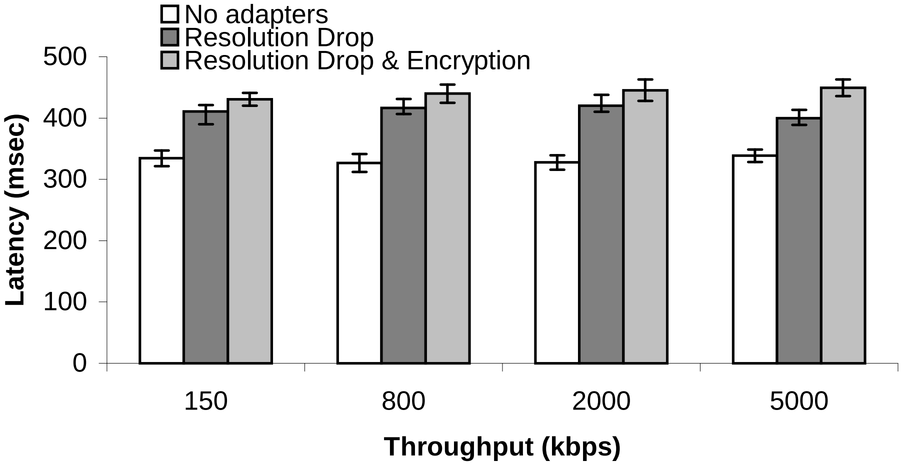

Figure 30: Plan calculation latency on HPs

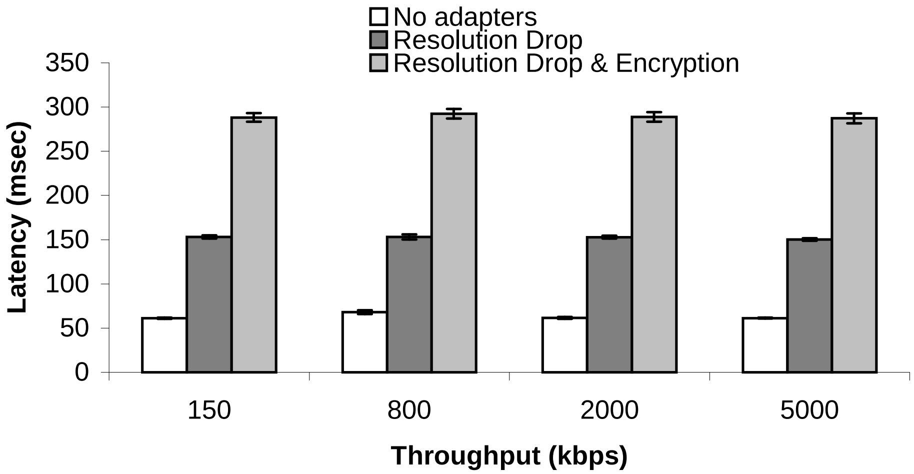

Figure 31: Deployment latency on HPs

Comparing these Figures with Figures 17-18 obtained for Dell Inpirons we can conclude that more powerful machines reduce the overhead of running the planning protocol and adaptations on Panda nodes. In both cases the planning procedure latency depends more on the number of adapters and less on the network bandwidth.

Figures 32 and 33 present the incremental planning, central planning, and replanning latencies for Resolution Drop and Encryption adaptations respectively. The graphs show that incremental planning is faster than central planning, and central planning is faster than central re-planning. The difference between initial central planning and central re-planning is bigger for bigger adapters because the transmission of the adapters depends on the traffic between the connection nodes, and the re-planning process competes with the data packet transmission. The bars for 150 kbps on Figure 33 show that the influence of the data packet traffic on that difference is even more significant if the available network bandwidth is smaller as expected.

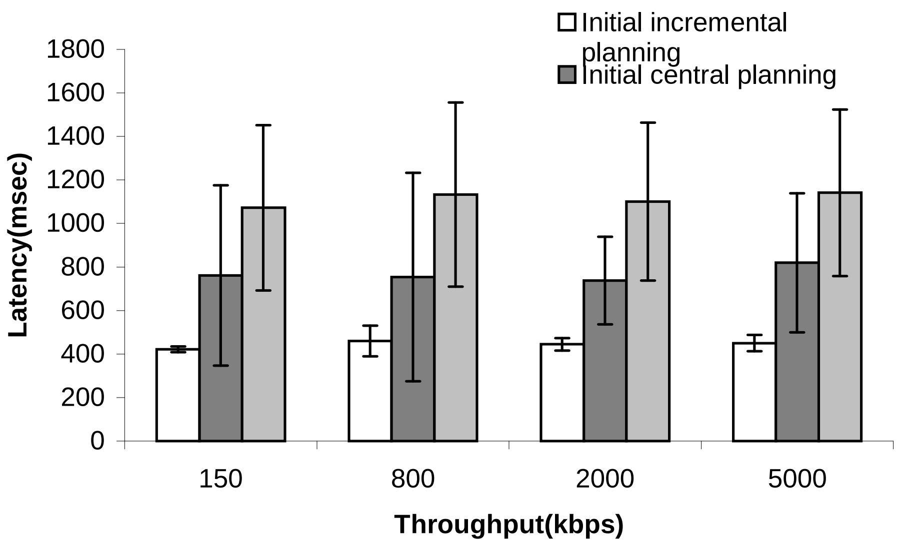

Figure 32: Incremental planning latency for Resolution Drop for HPs

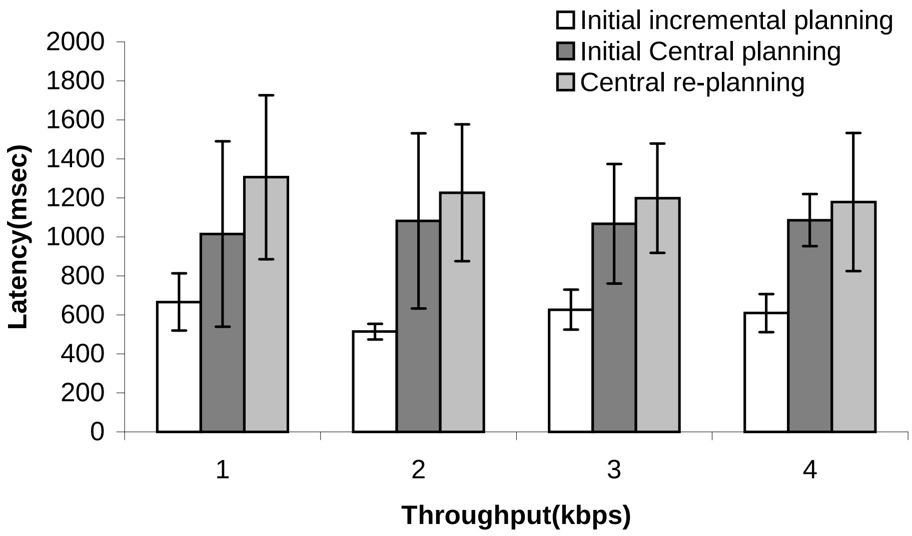

Figure 33: Incremental planning latency for Resolution Drop and Encryption for HPs

Figure 34 presents the number of packets sent under the incremental plan before the central plan was calculated and deployed on the HPs.

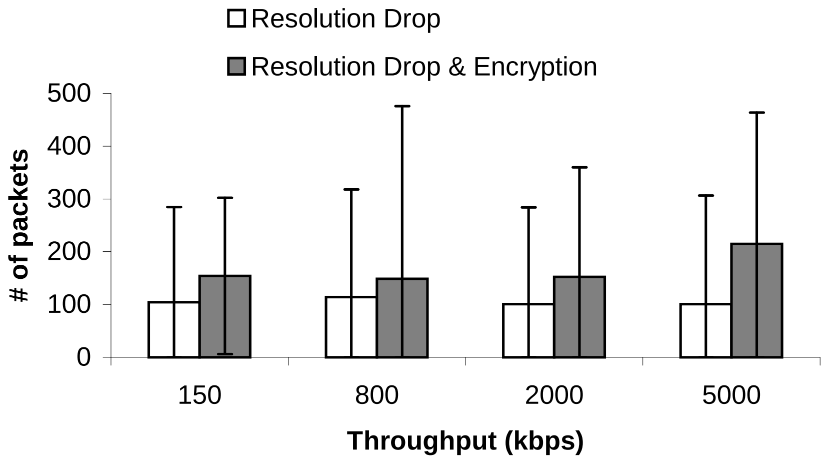

Figure 34: The number of packets sent under the incremental plan before the central plan was calculated and deployed

The number of the packets sent under incremental plan before the central plan is calculated and deployed varies very widely from 0 to 475. It makes the confidence intervals wide, allowing us to draw few conclusions about the effects of varying throughputs of different numbers of adapters.

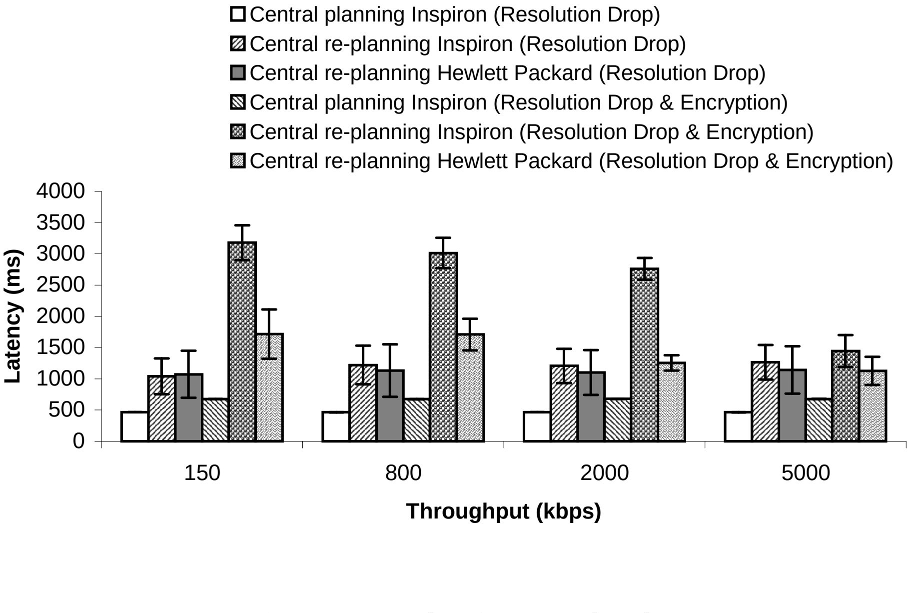

Figure 35: Re-planning procedure latency on HPs

Figure 35 presents the re-planning procedure latency compared to the correspondent initial central planning latency on Dell Inspirons and HPs. The graph shows that higher CPU power reduce the latency of the planning and re-planning proceudures. Slower machines also demonstrate stronger dependancy on the available network bandwidth. Bigger adapters make this dependancy even stronger.

The transfer of the Resolution Drop adaptation is not affected much by the throughput because it is a small adaptation. Encryption is a very large adaptation whose deployment takes much longer, and is more affected by competing data transfer traffic, thus varying from 1.5 seconds with 5000 Kbps throughput to more than 3 seconds with 150 Kbps throughput. Recall that the latency of the deployment that does not compete with data transfer traffic is presented on Figure 28.

More powerful computers, as Hewlett Packard machines with comparison to Dell Inspiron are, reduces the latency planning protocol and adaptation.

Another realistic application, RAT, was run to compare to WaveVideo application. Figure 36 presents the latencies of the plan calculation for WaveVideo and RAT. Both applications received plans calling for only the same encryption adapters.

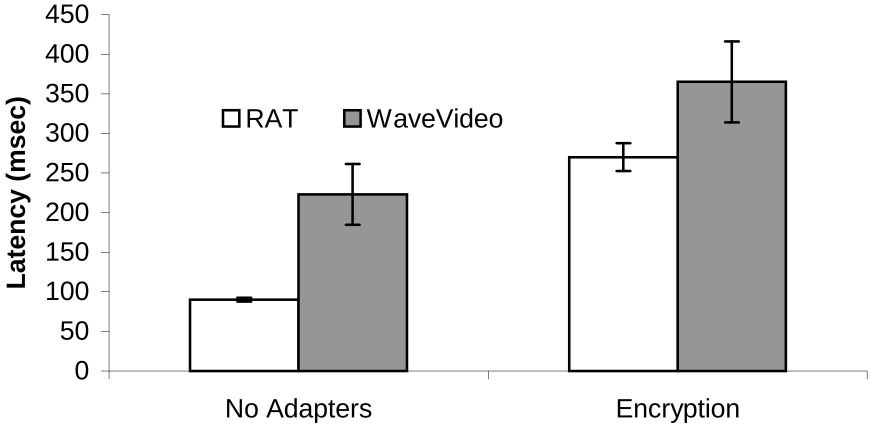

Figure 36: The plan calculation latency for Rat and WaveVideo applications

The RAT application transfers audio data, which is less intensive than video data. Since more resources of the source node can be used for the planning procedure, RAT reveives its plan faster than WaveVideo.

6. Quality of Service Improvement

The Panda overheads described in the previous section are acceptable if Panda's adapattions improve applicatrion-meaningful quantities. Here we present evidence of such improvements. As we mentioned in Section 2.4 QoS is measured in dB of PSNR as conventional units. PSNR expresses the difference between sent and delivered signal.

Figure 37 and 38 present PSNR luminance and Cb values respectively for the WaveVideo application discussed earlier on Dell Inspiron machines with a link limited to 150 Kbps.

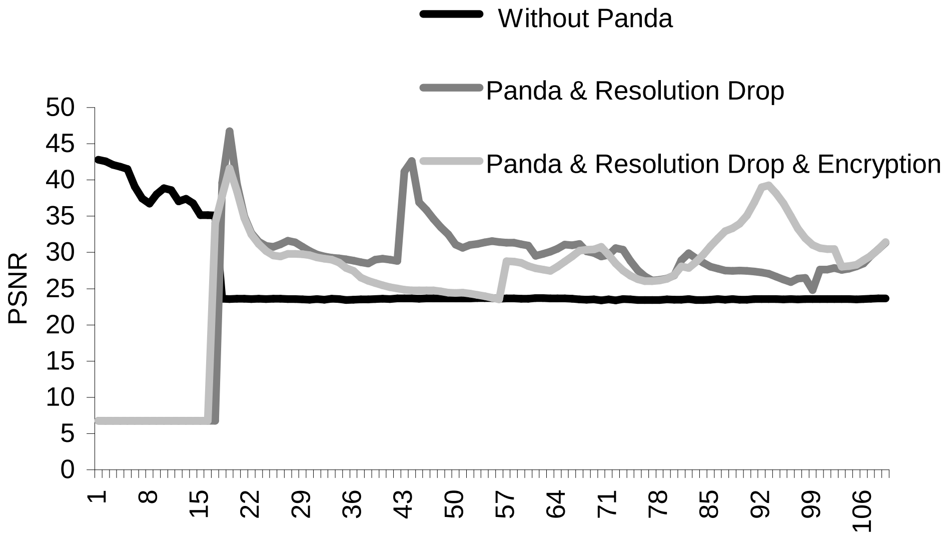

Figure 37: PSNR (luminance) on Dell Inspirons for 150 Kbps links for different numbers of video frames

Without Panda, at first the curve falls once the channel is saturated; Panda's curve improves after its planning protocol is completed, providing better PSNR after around 20 frames. Panda achieves this improvement by dropping unimportant packets thus allowing more important packets to arrive on time. The PSNR performance of the Panda with Resolution Drop and Encryption adaptation in some points can be even better than the Panda with Resolution Drop only. One possible reason can be the Panda extra buffering that slows the data stream but reduces the undesired packet loss.

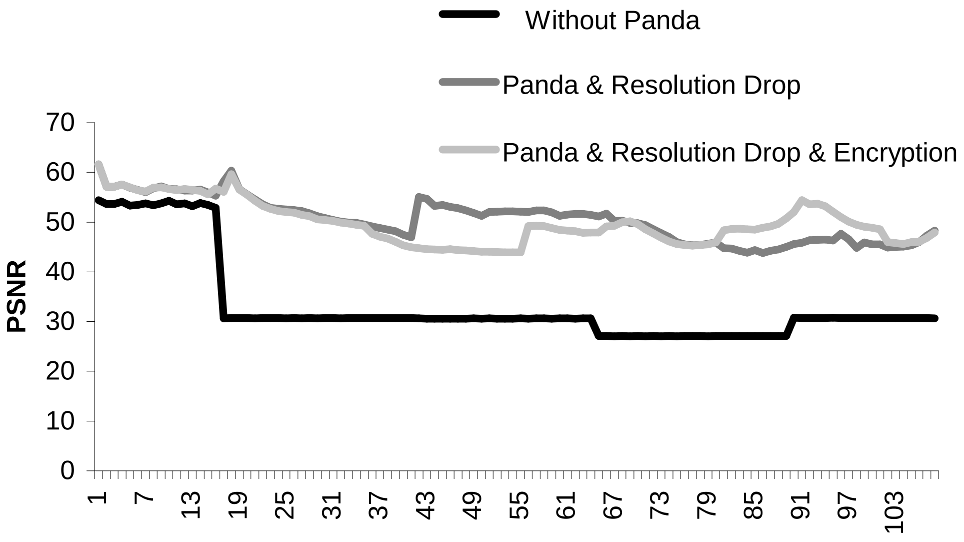

Figure 38: PSNR (Cb) on Dell Inspiron machines connection for 150 Kbps for different numbers of video frames

Figure 39 and 40 present PSNR luminance and Cb values respectively on Dell Inspiron machines with 5000 Kbps links. In this case the Panda service is not necessary because the network is powerful enough to deliver packets on time. These figures demonstrate the importance of a network-aware planning process. If the Resolution Drop adapter were blindly applied or not applied without considering the network conditions, poorer PSNR would result for some cases.

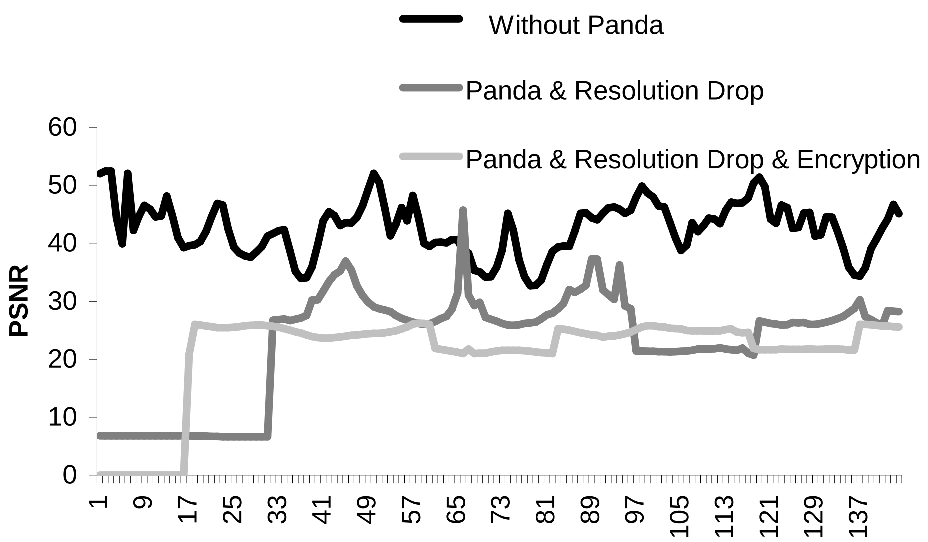

Figure 39: PSNR (luminance) on Dell Inspirons for 5000 kbps for different numbers of video frames

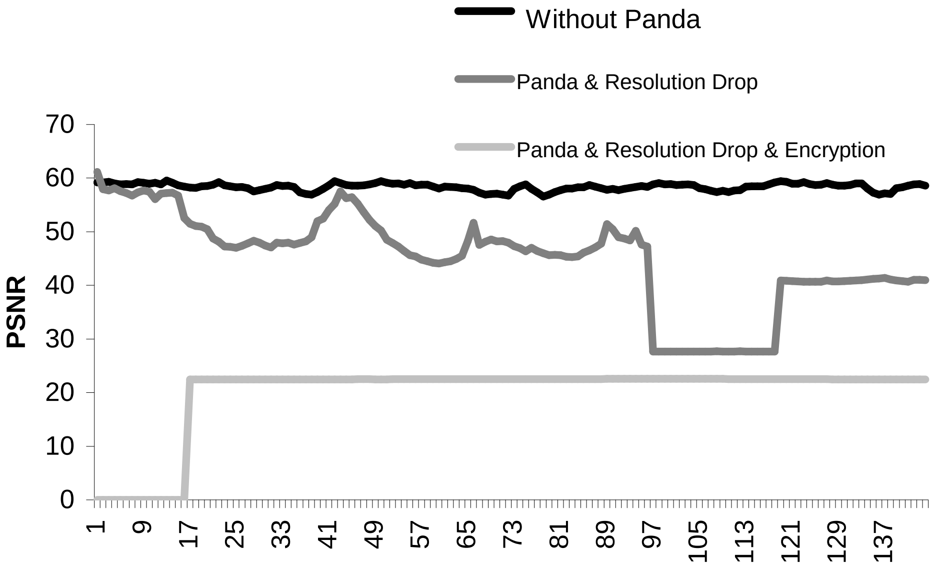

Figure 40: PSNR (Cb) on Dell Inspirons for 5000 kbps for different numbers of video frames

More powerful machines can process more data packets and reduce packet loss in poor-condition networks, and thus increase PSNR. Figures 41 and 42 present PSNR values on Hewlett Packard machines.

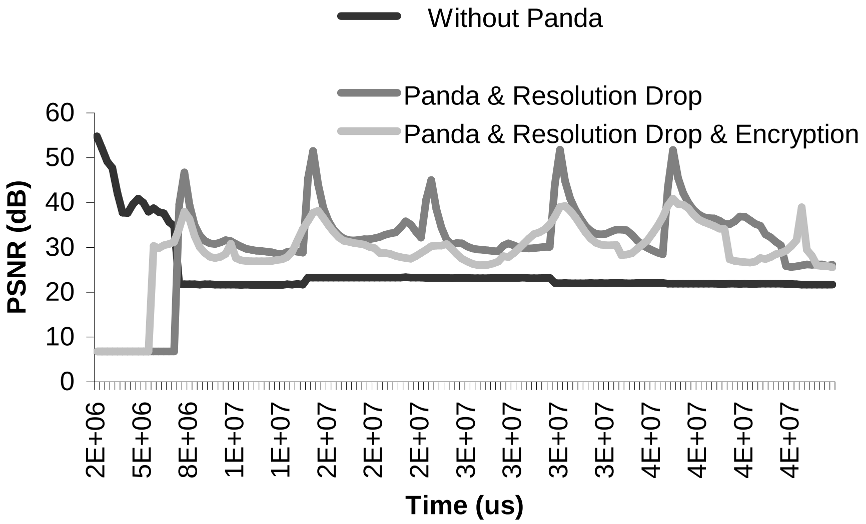

Figure 41: PSNR (luminance) on HPs for 150 kbps for different numbers of video frames

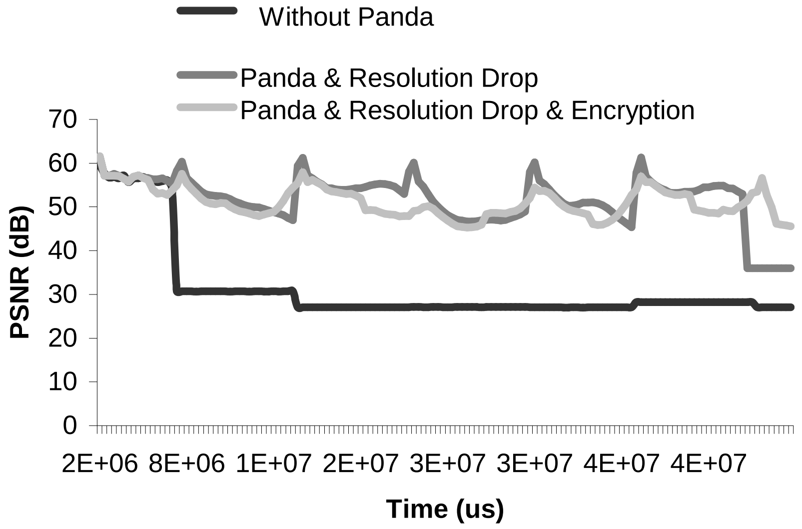

Figure 42: PSNR (Cb) on HPs for 150 kbps for different numbers of video frames

Panda provides greater improvement with the more powerful Hewlett Packard machines.

Figures 43 and 44 present PSNR luminance and Cb respectively on Hewlett Packard machines with 5000 kbps. Even with this more capable network, in a few cases Panda provides better PSNR.

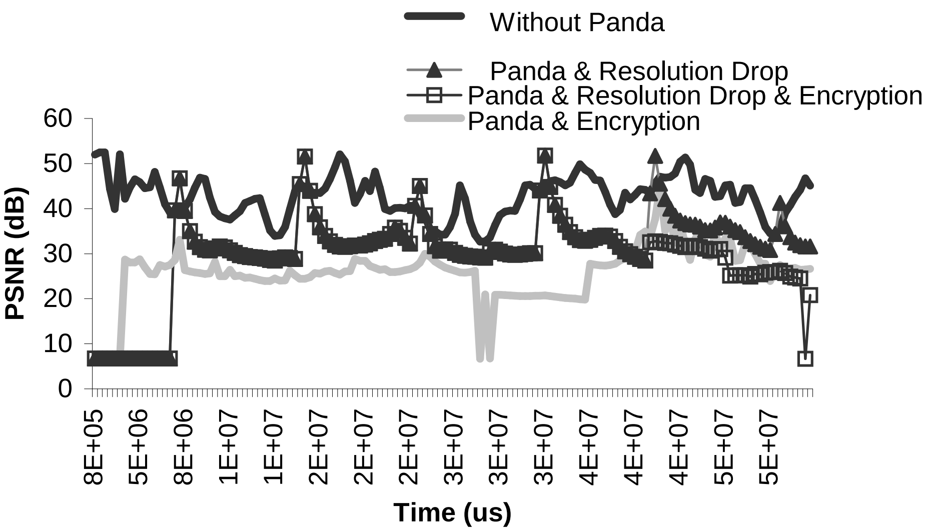

Figure 43: PSNR (luminance) on HPs for 5000 kbps links for different numbers of video frames

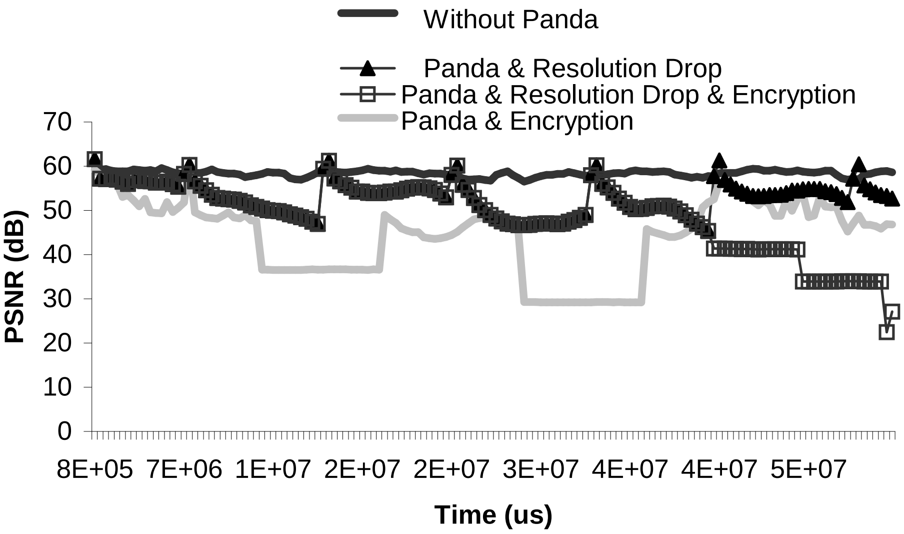

Figure 44: PSNR (Cb) on HPs for 5000 kbps links for different numbers of video frames

In figures 37-44, we include data for applying encryption along with packet dropping using Panda. For this data, Panda is providing a benefit beyond PSNR improvements by keeping the video secret. Without also dropping frames, though, much greater degradation in PSNR would accompony the improved security, as shown in Figure 46.

This data demonstrate the importance of considering all network conditions and possible remedies as a whole.

If all links have enough bandwidth but not all of them secure the deployment of Encryption can be necessary. It can be necessary to apply data compression just to compensate the effects of the Panda and its security remedies.

Figure 45 presents PSNR values in various network conditions.

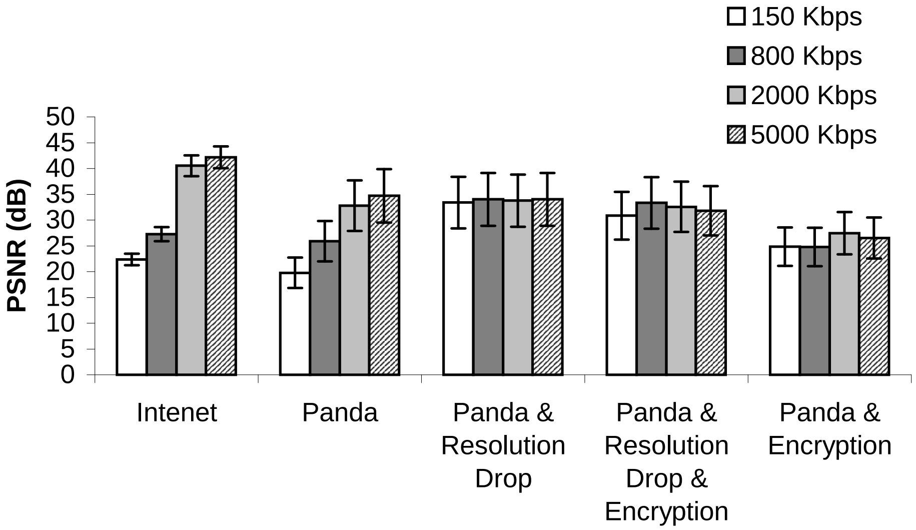

Figure 45: PSNR (luminance) on HPs

The Figure 45 clearly shows that Panda provides more benefit for more capable networks.

PSNR measurements can be used also for the quantifying the quality of the calculated plans for video data streams. Consider the following example. Figure 46a shows an example of a connection. One link in this connection has poor bandwidth, which is insufficient to carry all the data. Another is defined to be insecure. If the link adjacent to the source requires encryption and the next link requires filtering, then the incremental plan will contain an encryptor on the source node and a decryptor and a filter on the next node (Figure 46b). It is clear that this plan is less optimal than the optimized plan that will put the filter and encryptor on the source node and a decryptor on the next node (Figure 46c). In the latter case, encryption and decryption will be applied to fewer data packets. Figure 47 demonstrates better PSNR for a filtered and then encrypted and decrypted data stream (the dark gray line) than with an encrypted, decrypted, and then filtered data stream (the light gray line). The black line shows the PSNR without using Panda. This example shows that a naive planner that allocates remedies next to links where problems occur can produce plans that are not only theoretically suboptimal, but that give poorer application-meaningful performance.

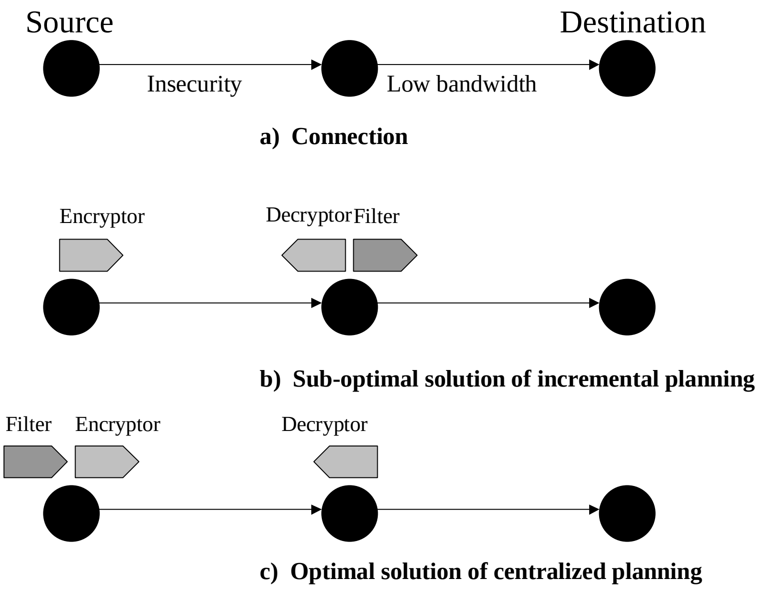

Figure 46: The advantage of centralized planning over incremental planning

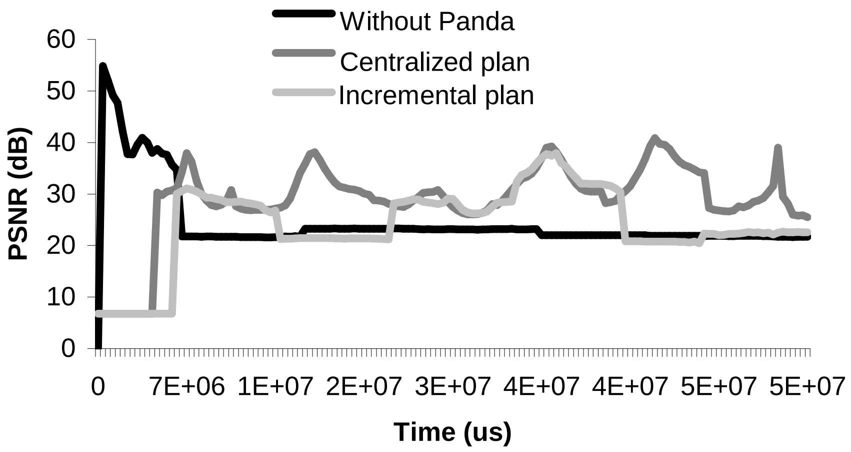

Figure 47: PSNR (luminance) for incremental and centralized plans

7. Discussion

The tests show that the overhead of using Panda to adapt data streams can be compensated with higher quality of service. The overheads are reasonable, particularly for relatively long-lived data streams. The latency added by the planning protocol is in the magnitude of 1 second. Panda also slows down the latency of data packets 4-10 times. The QoS however, can be improved up to 100%.

More computationally expensive user applications can increase the latency of plan calculation because application and planning processing compete on the source node. Plan calculation for a 10 times more intensive user application takes 4 time longer.

More powerful computers reduce the overhead of Panda and increase the delivered QoS. 1.6 times more powerful computers reduce 1.6 times the latency of planning procedure and increase QoS by 30%.

In low-bandwidth networks even the presence of Panda as an extra buffer for a bursty traffic can improve the QoS.

Incremental planning can produce and deploy plans 50% faster than the corresponding central plan, but QoS for the incremental plan can be 45% worse in some cases as shown on Figure 47. The number of packets that are sent under an incremental plan before a central plan is calculated and deployed varies from zero to some hundreds depending on the variance of the latency of the central planning procedure. Therefore, brief sessions should use incremental planning only.

Re-planning can take a number of times longer than initial planning because it runs concurrently with data traffic.

8. Conclusion

Because Active Networks technologies are complex, many applications will not be coded to take advantage of their capabilities. The data streams sent by such applications can obtain the benefits of Active Networks technologies, provided an automated system can determine the proper choice and placement of Active Networks adaptations. This paper has demonstrated that it is possible to build an automated planning system that is quick and effective. The overhead and benefits of Panda were measured in various situations. Two real multimedia applications were tested for video and audio data streams. Their performance was improved by Panda: WaveVideo application data packets were compressed and encrypted, RAT application packets were encrypted.

The measurements presented in this paper were made in three dimensions: applications with different data generation intensity, computers with different CPU power running Panda nodes, and network links with different bandwidth and security levels. Various observations were made based on the results of the testing.

The results presented in this report can be used for the design of a new generation of Active Network technologies.

9. References

[Fankhauser99]	G. Fankhauser, M. Dasen, N. Weiler, B. Plattner, and B. Stiller, "WaveVideo-an integrated approach to adaptive wireless video," *Mobile Networks and Applications*, vol.4, ACM Press, 1999, p.255-71.