Distributed Scheduling in Large Scale Monitoring Infrastructures

Josep Sanjuàs-Cuxart* Pere Barlet-Ros* Gianluca Iannaccone† Josep Solé-Pareta*

* Universitat Politècnica de Catalunya
Barcelona, Spain

† Intel Research
Berkeley, CA

1. INTRODUCTION

Network monitoring is becoming a necessity for network operators, who usually deploy several monitoring applications that aid in tasks such as traffic engineering, capacity planning and the detection of attacks or other anomalies. There is also an increasing interest in large-scale network monitoring infrastructures that can run multiple applications in several network viewpoints [4].

Network monitoring infrastructures are extremely resource constrained, due to the continuous growth of network link speeds and computational complexity of monitoring applications. It is therefore highly desirable for such systems to be scalable, e.g., to provide their operators with the ability to incrementally add more computing nodes to the system, in order to support more applications and to sustain a higher volume of traffic. However, providing network monitoring applications with the ability to migrate across nodes is not trivial. The solution of simply replicating the incoming traffic to other nodes is prohibitively expensive, since it requires additional bandwidth and may involve traffic replication devices and packet capture hardware.

Furthermore, such infrastructures must efficiently deal with hot spots that monitoring applications naturally create, since the events of interest are usually localized (e.g. intrusion and anomaly detection). Such events can cause load to be distributed unevenly across the monitoring infrastructure. This scenario differs from the ones traditionally explored in distributed systems research [2] in that monitoring applications are continuous and never finish. Therefore, in such an environment, the principal scheduling mechanism is task migration.

In this work, we propose an architecture for network monitoring applications that enables task migration across nodes. We then propose a distributed scheduling scheme that dynamically balances the load across all the nodes of the monitoring infrastructure.

ACM CoNEXT 2008 Student Workshop, December 9, 2008, Madrid, SPAIN

2. ARCHITECTURE

We propose a two-stage architecture for the applications to provide network monitoring infrastructures with migration capabilities. The first stage of each application performs those computations with severe real-time constraints that require access to the raw packet stream. Therefore, the first stage must run in the nodes equipped with the specialized packet capture hardware (i.e., capture nodes). The main goal of the first stage is to perform traffic filtering and short-term aggregation to enable the second stage to be easily migrated to remote nodes. The second stage continuously receives the results from the first stage and performs stateful, potentially more complex and longer-term computations.

Our architecture provides the following migration procedure: 1) the application is paused, 2) it is asked to serialize its state, 3) its code, state and queued input are transferred to the destination node, 4) it is loaded and asked to deserialize its state, and 5) it is resumed.

Applications that do not support migration can still run on the capture node. We have in the past addressed the resource management problem in the capture node by developing a load shedding subsystem [1]. In this work, we explore the complementary problem: deciding how to (re)assign the second stage of each application to the available computing nodes.

3. DISTRIBUTED SCHEDULING

The goal of our scheduling algorithm is to introduce the smallest possible processing delays in the applications, while balancing the load across nodes in order to provide fairness of service (i.e., to allow all applications to experience similar delays). These objectives can be summarized as a single one: *to minimize the maximum processing delay across applications.*

Traditional distributed scheduling approaches are designed for finite tasks and are not appropriate in our

environment, given the continuous nature of monitoring applications. Our thesis is that only a predictive approach can consider the long-term impact of scheduling decisions. We also argue that the migration costs and the associated benefits form the fundamental tradeoff that should guide the scheduling algorithm. However, previous proposals in similar environments have not considered the migration costs explicitly.

Our proposal is to consider estimates of the future workloads of applications to balance the load across nodes, while only paying the migration overheads when they can be amortized in the future. We propose the following distributed algorithm, based on the well-known technique of pairwise load balancing.

Periodically, the nodes of the infrastructure randomly pair with another node. Both nodes predict the delay that applications will experience under different migration plans, including the migration overheads, and select the one that yields the lowest maximum delay.

Our prediction relies on two different models based on Holt-Winters forecasting. The first maintains the cost per item (received from the first stage) for each application, while the second models the future item arrivals. Both parameters exhibit temporal patterns that Holt-Winters can capture. Using these models and the current amount of queued items, each node can estimate the future delays of each application. Other solutions that consider these parameters as constant or slowly changing over time would result in less stable plans.

As nodes keep pairing, the load fluctuates from more loaded to less loaded nodes, continuously adapting to the varying CPU usage of applications.

4. PRELIMINARY RESULTS

We implement the two-stage architecture described in Section 2 on eight representative network monitoring applications. For the sake of reproducibility, we use the CoMo system [3] to record the arrivals from stage 1 and the CPU time required to process them in stage 2 using a 15 minutes long real-world packet trace collected at the Catalan RREN. A description of the applications and the packet trace can be found in [1]. Each application is run with different configuration parameters resulting in 100 application workload traces.

Our current prototype only considers the costs of queued (unprocessed) items to perform the prediction, and does not yet model future arrivals. We also implement a reactive strategy to compare against. In this scheme, when the queue of a node is empty, the application that suffers the highest delay across the infrastructure is moved to the idle node.

We perform several simulations with varying number of nodes and a fixed available bandwidth between nodes of 1000 KB/s. As we increase the number of nodes, we reduce their processor speed to maintain a constant aggregate system capacity.

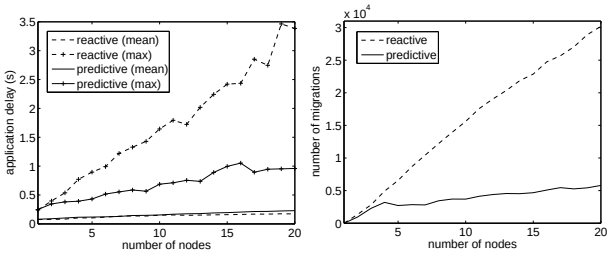

Figure 1: Average and average maximum delay (left) and number of migrations (right).

Figure 1 (left) shows similar average application delays for both strategies, but a significant difference in the average maximum application delay, indicating that our strategy is more fair to the applications. Note that the delays increase with the number of nodes, since it is easier to correctly handle an equal amount of load with a reduced amount of more capable processors.

Figure 1 (right) plots the number of migrations performed by each strategy as a function of the number of nodes. As expected, the number of migrations in the reactive strategy increases linearly with the number of nodes, while our strategy scales better.

5. SUMMARY AND FUTURE WORK

We have presented an early prototype of a distributed network monitoring architecture that can balance the load across nodes, moving tasks across the system as necessary. The proposed scheduling algorithm uses a prediction of the CPU requirements of tasks and the migration costs to govern its decisions.

The most important piece of future work is to incorporate a sense of the long-term consequences of migrations. Our current prototype algorithm only predicts the cost of the unprocessed queued items for each application, but does not yet anticipate future arrivals.

We are also working on conducting a study of both the accuracy of our prediction techniques and the computational overhead of our algorithm.

6. REFERENCES

[1] P. Barlet-Ros et al. Load shedding in network monitoring applications. In *Proc. of USENIX Annual Technical Conf.*, June 2007.

[2] T. Casavant and J. Kuhl. A taxonomy of scheduling in general-purpose distributed computing systems. *IEEE Transactions on Software Engineering*, 14(2):141–154, 1988.

[3] G. Iannaccone. Fast prototyping of network data mining applications. In *Proc. of PAM*, Mar. 2006.

[4] kc claffy et al. Community-oriented network measurement infrastructure (CONMI) workshop report. *SIGCOMM CCR*, 36(2), 2006.