VaaS: Video as a Service on the Cloud

Kishore Ramachandran, College of Computing, Georgia Tech
Liviu Iftode, Dept. of Computer Science, Rutgers University
Ajay Mohindra, Smarter Cloud Platform, IBM T. J. Watson Research
E-mail: rama@cc.gatech.edu, iftode@cs.rutgers.edu, ajaym@us.ibm.com

Topics:

Programming Models for the Cloud
Clouds Architectures and Systems

Current Research: Sensors of various modalities and capabilities, especially cameras, have become ubiquitous in our environment. Their intended use is wide ranging and encompasses surveillance, transportation, healthcare, emergency response, disaster recovery, and the like. Such applications, often referred to as *situation awareness* applications, are continuous in nature and are characterized by a *feedback control loop* involving sensing, processing, actuation, and possible retargeting of the sensors. Technological advances and the low cost of such sensors enable deployment of large-scale camera networks in metropolises such as London and New York. Multimedia algorithms for analyzing and drawing inferences from video and audio have also matured tremendously in recent times. Processing, archiving, indexing, and retrieving distributed video feeds 24×7 requires a massive computational infrastructure, which if naively architected would scale up in proportion to the number of cameras. The data storage and analysis on such a large-scale needs a very big data center. An enterprise may choose to have its own data center. However, with the emergence of cloud computing, a more sensible alternative is for an enterprise to leverage this platform, and let the IT companies (such as Amazon, Yahoo, Google, and Microsoft) deal with the generational changes needed to spruce up the computational infrastructure.

Our current research is focused on understanding the *pressure points* on the infrastructure, i.e., the processing, memory, network, and storage bandwidth needed for a large-scale computationally demanding streaming application, using video-based surveillance as a canonical example. We are well-positioned to carry out this study since we have already built a cluster-based distributed system called *ASAP* (OPODIS 2007) for situation awareness that incorporates multi-modal sensors (cameras, RFID, and audio). In ASAP, the sensor-level front-end processing (such as filtering and threshold of motion in a camera image) happens in a software entity called sensor agent (SA). The computationally intensive part of the application (such as face detection and high level inferencing) happens in a software entity called ASAP agent (AA). The distributed programming system, *Persistent Temporal Streams (PTS)* (ACM Middleware 2009), provides location-independent time-indexed channels for communication of stream data among these entities. It also manages transparent migration of stream data between memory and backend storage. The accrued flexibility of this architecture allows the SAs and AAs to be run anywhere in the distributed system. Thus, ASAP serves as a good testbed for determining the pressure points as we scale up the number of sensors as well as the sophistication of the processing that happen in the SAs and AAs. Preliminary results for moving the computation into the cloud with real-time guarantees are promising. For example, it is possible to sustain a frame rate of 15 frames/sec for streaming low resolution (151×100 pixels) images from a camera into the cloud (Amazon EC2) and to execute a face detection algorithm on every frame. We have also conducted preliminary experiments with partitioned files and empirically determined the optimal size of the partitions to store continuous video streams into the Amazon S3 store taking into account the communication latency between the camera sensors and the cloud, and the internal latency for storing a file chunk from Amazon EC2 VM to the S3 store. In our current work, we are undertaking more elaborate performance studies to understand how best to architect VaaS from the point of view of eliminating any bottlenecks that will inhibit real-time situation awareness.

Future Research: We wish to provide *Video as a Service (VaaS)* for demanding real-time streaming applications to exploit the utility computing framework offered by the cloud. This raises several challenges, the solutions to which form the expected intellectual contribution of our future research. Firstly, simply transporting all video feeds to the cloud is neither necessary (if there is no change in the sensed environment) nor practical considering the network pressure of ingesting all these feeds and the storage pressure of archiving all the video. Secondly, running video analytics on the streams is computationally demanding and would make sense to run on the cloud. However, there is a tension between real-time response for alert generation based on video analytics and response time from the cloud. Thirdly, while IP cameras are becoming commonplace, other sensors (e.g., RFID and legacy cameras) that may be used in the sensing infrastructure for increasing the fidelity of high-level situation awareness may not be IP-ready and thus not easily stream-able into the cloud. Fourthly, we believe the interfaces available today for cloud computing are not ideal for such large-scale demanding real-time streaming applications.

Having the right level of abstraction is important for facilitating application development and resource management. However, current cloud offerings are not the correct level of abstraction for video analytics applications. Infrastructure as a Service (IaaS) clouds, such as Amazon EC2, are too low-level, making development difficult and resource management unintelligent and inefficient. Platform as a Service (PaaS) clouds like Microsoft Azure are an appropriate level, but they do not provide the right kinds of abstractions for video analysis applications. Therefore, VaaS is a cloud platform that offers the right abstractions for developing and executing video analytics applications.

We have preliminary ideas on appropriate high-level abstractions for video-based surveillance. *Mobile Virtual Sensor (MVS)* is a unit of resource acquisition and scheduling that has a one-to-one association with a *target* being tracked by a surveillance application. MVS shields the application programmer from details such as camera streams to get input from for tracking as the target moves in an environment. Similarly, the execution engine supporting the MVS abstraction orchestrates management of the cloud computing resources needed for executing the tracking algorithm and storing the streams.

Currently, we have implemented a programming system based on MVS in a cluster. In the proposed research, we will extend these ideas to realize VaaS on a cloud. In particular, we will investigate the mechanisms that will allow an application to decide dynamically the split of data and computation between the sites (where the sensor are located) and the cloud. Such a split may be needed for a variety of reasons including sensitivity of the data and computation for the enterprise. Our measurements to understand the pressure points (our current research) will help us discover the right abstractions to allow improved resource management, and our development experience with measurement applications will guide us to abstractions that ease development efforts.

There are a number of existing systems that efficiently handle streaming computation, such as IBM System S (InfoSphere Streams), Aurora / Borealis (Brown/MIT), STREAM (Stanford), and StreamIt (MIT). However, these are general purpose systems that do not handle streaming video well. In our proposed research, we will develop a runtime system that performs efficient resource management for computational streaming video on the cloud.