2019

Cloud Resource Optimization for Processing Multiple Streams of Visual Data

Zohar Kapach
Purdue University

Andrew Ulmer
Purdue University

Daniel Merrick
Purdue University

Arshad Alikhan
Purdue University

Yung-Hsiang Lu
Purdue University

See next page for additional authors

Cloud Resource Optimization for Processing Multiple Streams of Visual Data

Zohar Kapach, Andrew Ulmer, Daniel Merrick, Arshad Alikhan, Yung-Hsiang Lu,
Purdue University, West Lafayette, IN, USA
{zkapach, ulmera, dmerrick, aalikhan, yunglu}@purdue.edu
Anup Mohan
Intel, Santa Clara, CA, USA
anup.mohan@intel.com
Ahmed S. Kaseb
Cairo University, Giza, Egypt
akaseb@eng.cu.edu.eg
George K. Thiruvathukal
Loyola University Chicago, Chicago, IL, USA
Argonne National Laboratory, Argonne, IL, USA
gkt@cs.luc.edu

Abstract

Hundreds of millions of network cameras have been installed throughout the world. Each is capable of providing a vast amount of real-time data. Analyzing the massive data generated by these cameras requires significant computational resources and the demands may vary over time. Cloud computing shows the most promise to provide the needed resources on demand. In this article, we investigate how to allocate cloud resources when analyzing real-time data streams from network cameras. A resource manager considers many factors that affect its decisions, including the types of analysis, the number of data streams, and the locations of the cameras. The manager then selects the most cost-efficient types of cloud instances (e.g. CPU vs. GPGPU) to meet the computational demands for analyzing streams. We evaluate the effectiveness of our approach using Amazon Web Services. Experiments demonstrate more than 50% cost reduction for real workloads.

Introduction

Visual data is projected to grow exponentially in the coming years. Video is expected to grow 26% annually and account for 82% of consumer Internet traffic; live video on the Internet is expected to grow 1,500% from 2016 to 2021 [1]. Surveillance cameras are expected to see similar growth. Today, more than 240 million surveillance cameras have been installed globally [2] and Stratistics MRC predicts that the market will continue to grow 18.3% annually. These video surveillance cameras can produce vast amounts of real-time data. With the rapid progress of machine learning and computer vision, it is possible to analyze these real-time streams. Two key technologies are essential to harness the potential of these real-time data steams: (1) analyzing the data using computer vision, and (2) employing scalable resources to meet the demands of computation.

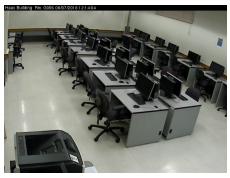
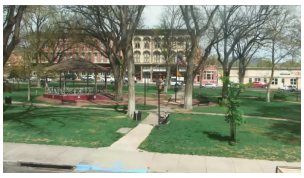
This paper focuses on solving the second problem. This research group adopts both supercomputing and cloud computing to address the problem from different perspectives. Supercomputing

systems are optimized for speed and I/O throughput but are limited when it comes to meeting fluctuating demands because of job scheduling requirements. Cloud computing offers the potential for high-end computing resources and also allows for on-demand operation. Proponents of cloud computing are willing to make trade-offs when it comes to exchanging CPU and I/O speed in favor of high availability and flexible scheduling. This paper describes our experiences with cloud computing to analyze many video streams from network cameras.

Many factors affect the resource requirements for analyzing visual data streams, including the complexity of the analysis programs, the content being analyzed, the size (number of pixels) of each image or video frame, the frame rates, etc. The requirements may vary over time. For example, a program that analyzes video streams from traffic cameras to detect congestion may run during rush hours only. Cloud computing is the best option to meet these varying needs. Cloud computing allows users to dynamically allocate virtual machines (called *instances*) on demand. Cloud vendors, such as Amazon EC2 (Elastic Cloud Computing), Microsoft Azure, and IBM Cloud, provide many types of cloud instances with different amounts of memory, number of cores, and number of GPUs (graphics processing units) at different prices (US dollars per hour). These instances reside in data centers distributed in North and South America, Europe, Asia, and Australia. The challenge is minimizing the cost of cloud instances without sacrificing performance.

To drive the research in cloud resource optimization for processing multiple visual data streams, a software infrastructure named *Continuous Analysis of Many Cameras* (CAM^2) has been established at Purdue University [3]. CAM^2 uses network cameras that provide real-time visual data publicly available on the Internet. These cameras observe traffic intersections, metropolitan areas, university campuses, tourist attractions, etc. Some cameras provide videos and others show snapshots. The analysis of real-time data streams can be used in many applications, such as urban planning and emergency response.

CAM^2 is designed as a computing platform for analyzing real-time network camera data. Network cameras capture all sorts of visual data. The examples below show a night club, a computer classroom, a park, and a city street. CAM^2 has the following major components: (1) a camera database storing the information about the cameras' geographical locations, frame rates, etc, (2) a run-time system retrieving data from the network cameras and sending the data for analysis, and (3) a resource manager allocating and releasing computing resources to meet application demands. CAM^2's application programming interface (API) allows the same analysis programs to retrieve and analyze data from a diverse set of network cameras [4]. CAM^2 uses only public data available on the Internet. Readers interested in using CAM^2 can register at `https://www.cam2project.net`. The Terms of Use and Privacy Policy are available on the website.


RESOURCE MANAGEMENT OF CLOUD INSTANCES

As mentioned earlier, cloud instances would be ideal solutions for meeting the varying demands of video analytics. Cloud vendors provide many options; some instances have GPUs while the others do not. Amazon EC2 has optimizers for processes, memory, and networks. IBM cloud gives users the option of selecting virtual machines or physical machines (called "bare metal servers"); within each category there are dozens of options with different types of processors and amounts of memory. Microsoft Azure also has many configurations to choose from. Table I shows the prices of several types of cloud instances at different locations.

Vendor	Instance	Cores	Memory (GiB)	GPU	Price Per Hour (US$)		
					Virginia	London	Singapore
EC2	c4.2xlarge	8	15	0	0.398	0.476	0.462
	c4.8xlarge	36	60	0	1.591	1.902	1.848
	g3.8xlarge	32	244	2	2.280	N/A	3.340
					US East	West Europe	East Asia
Azure	D8 v3	8	32	0	0.384	0.480	0.625
	NC24r	24	224	4	3.960	5.132	N/A

TABLE I. Prices of different Amazon EC2 and Microsoft Azure cloud instances at different locations.

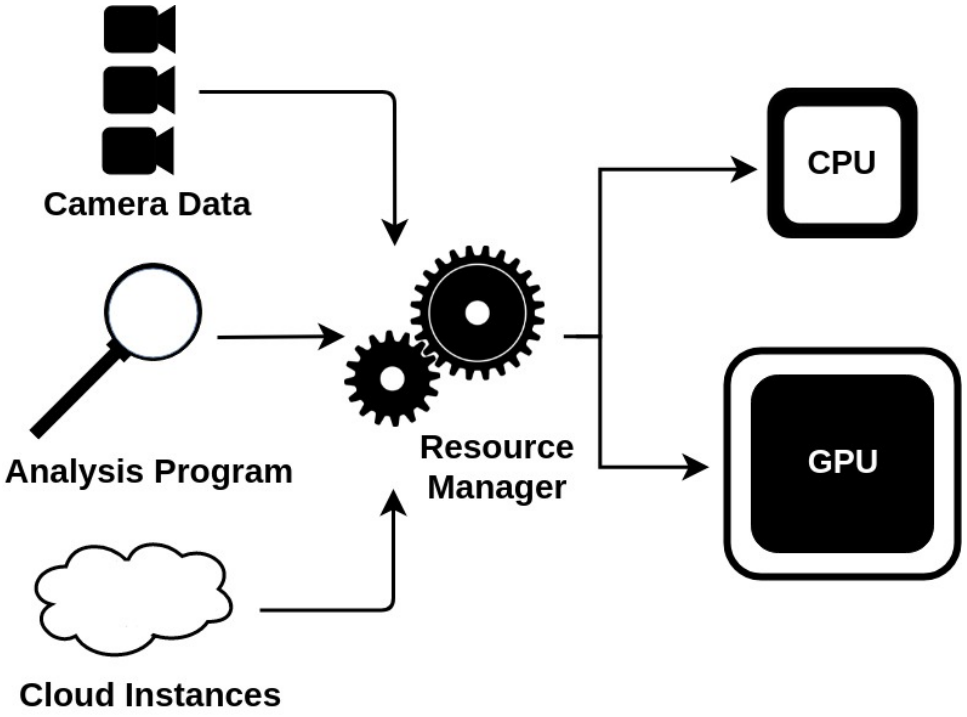


Fig. 1. The resource manager considers many factors, including the data content, the analysis programs, and the types and costs of cloud instances. The manager selects the most cost-efficient instances (these may have CPUs only or they may include GPUs as well).

Because of the wide range of prices, instance types, and locations, a resource manager is essential when optimizing large scale cloud usage. Figure 1 illustrates the role of a resource

manager. The resource manager has to consider the following factors when selecting the most appropriate cloud instances:

- **Characteristics of analysis programs.** Different programs have different resource requirements: some programs benefit from more cores, some need more memory, and some need GPUs.
- **Desired frame rates.** Some analysis programs (such as tracking moving objects) need high frame rates. For some other programs (such as observing air quality or traffic congestion), low frame rates are sufficient.
- **Image or frame sizes, in terms of pixels.** If an image has more pixels, more computation is needed.
- **Content of the data.** The execution time and resource requirements depend on the complexity of the content. Complex content (e.g., many moving objects in a video stream) may require more computing resources than simple content.
- **Locations of cameras and cloud instances.** When analyzing video streams, the distances between cloud instances and cameras (measured by the round-trip time) can affect the frame rates [5]. Therefore, there may be restrictions on the location of an instance.

Based on these inputs, the resource manager selects the cloud instances. An instance's configuration (also called the type) corresponds to the number of cores, the amount of memory, the presence of GPUs, and the geographical location. The choice is made to meet the resource requirements at the lowest possible cost. This resource manager is dynamic and its decisions may change over time because the demands may vary. This paper presents two optimization strategies: one manages CPU and GPU usage, and the other manages the locations of instances.

Cloud Resource Optimization Problem

The problem: selecting the cloud instances (types and locations) for meeting the resource requirements needed to analyze real-time streaming data at the lowest costs.

Adaptive Resource Management for Video Analysis in the Cloud

A solution to handling cloud resource management, proposed by Mohan et al. [6], is Adaptive Resource Management for Video Analysis in the Cloud (ARMVAC) . This method does the following: (1) reads inputs necessary for modeling the problem as a Vector Bin Packing Problem, (2) selects the locations of cloud instances to be considered for the given analysis, (3) determines the types and number of cloud instances needed for the analysis, and (4) employs an adaptive resource management solution to adjust resource requirements during runtime. Kaseb et al. [7] improve ARMVAC by considering cloud instances with both CPU and GPU. Mohan et al. [8] extend ARMVAC by considering instances' locations. The following sections explain these improvements.

CPU and GPU Management in the Cloud

Differences between CPUs and GPUs: Central processing units (CPUs) and graphics processing units (GPUs, also called general-purpose graphics processing units GPGPUs) have different characteristics and capabilities. CPUs, with several to dozens of processing cores, are the brains of computers. CPU cores can handle complex program flows with many control statements. In contrast, GPUs have thousands of smaller and simpler cores. GPUs can be significantly faster

when performing similar tasks on many pieces of data, such as videos and images. GPUs adopt the SIMD (single instruction, multiple data) style parallelism: the same computation runs on different elements of arrays.

Another key difference between CPUs and GPUs is price. For example, Amazon EC2's `c5d.9xlarge` CPU instance has 36 virtual CPUs with 72 GB of memory and costs $1.728 per hour. GPUs, however, tend to be much more expensive. The `p3.2xlarge` GPU instance has 8 virtual CPUs with 61 GB of memory and costs $3.06 per hour. Another GPU instance, `p3.8xlarge`, has 32 virtual CPUs and 244 GB memory and costs $12.24 per hour.

Multi-Dimensional Multi-Choice Packing Problem

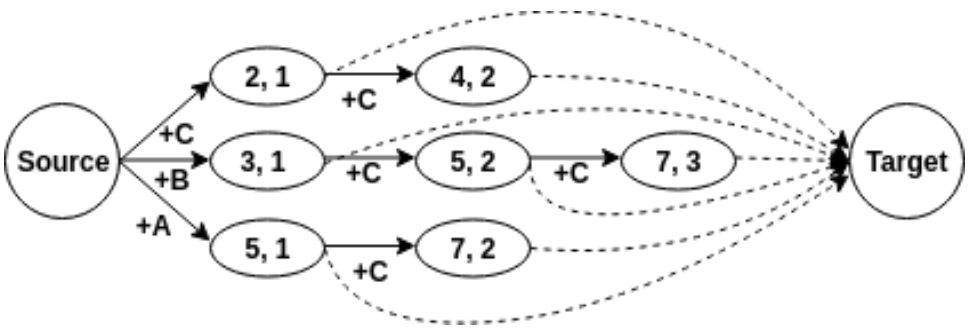
Multi-dimensional multi-choice packing problems occur in everyday life. Consider renting a truck (or trucks) for moving boxes. The trucks come in different sizes at different costs and the boxes have different dimensions (length, width, height) as well. The objective is to find the cheapest truck (or trucks) that can accommodate these boxes. Some boxes, however, have more requirements than others. For example, it is possible that a box may include frozen food and would need a truck that offers refrigeration.

This problem is analogous to the problem of finding the most cost-effective cloud instances for analyzing the data streams. Each type of instance can be compared to a type of truck, in that it has several dimensions: the number of CPUs, the amount of memory, and the presence of GPUs. Each analysis program, running on one data stream, is a box with a particular size. Some analysis programs (e.g., tracking) need GPUs to process the data at the desired frame rates, similar to the boxes that require refrigerated trucks. Some other programs can use low frame rates and can run on CPUs only. Finding the most cost-effective cloud instance (or instances) to accommodate the analysis programs for all data streams is comparable to finding the most cost-effective truck (or trucks) to transport all boxes.

In order to explain the multiple choice vector bin packing problem, we created an arc-flow graph like the one presented by Brandao and Pedroso [9]. In the arc-flow graph illustrated below, the nodes weight represents the box dimensions (width, height), and the demand represents the number of boxes. Any path from the source node to the target node represents a viable set of boxes to pack into a truck. In this example, a truck with dimensions (7, 3) is filled with 3 types of boxes with the following weights and demands:

A Weight: (5, 1) Demand: 1
B Weight: (3, 1) Demand: 1
C Weight: (2, 1) Demand: 2

First, box A is added as many times as the demand requires without over filling the truck. Then, box B goes through this process. And finally, box C. Once this graph is built, a second step is required to compress the graph. In cases where there can be hundreds of boxes and hundreds of trucks, a compressed graph is required to reduce the number of paths using the same set of boxes. This in turn will result in time saved when solving the graph. After the compression is complete a Gurobi 5.0.0 branch-and-cut solver is used to find the best path to get the maximum number of boxes into a truck. This method, however, only solves for one type of truck. In order to solve for multiple choices of trucks, we implemented the multiple choice method used by Brandao and Pedroso [10]. In this implementation, a graph is constructed for each truck type, and then solved using the Gurobi solver.



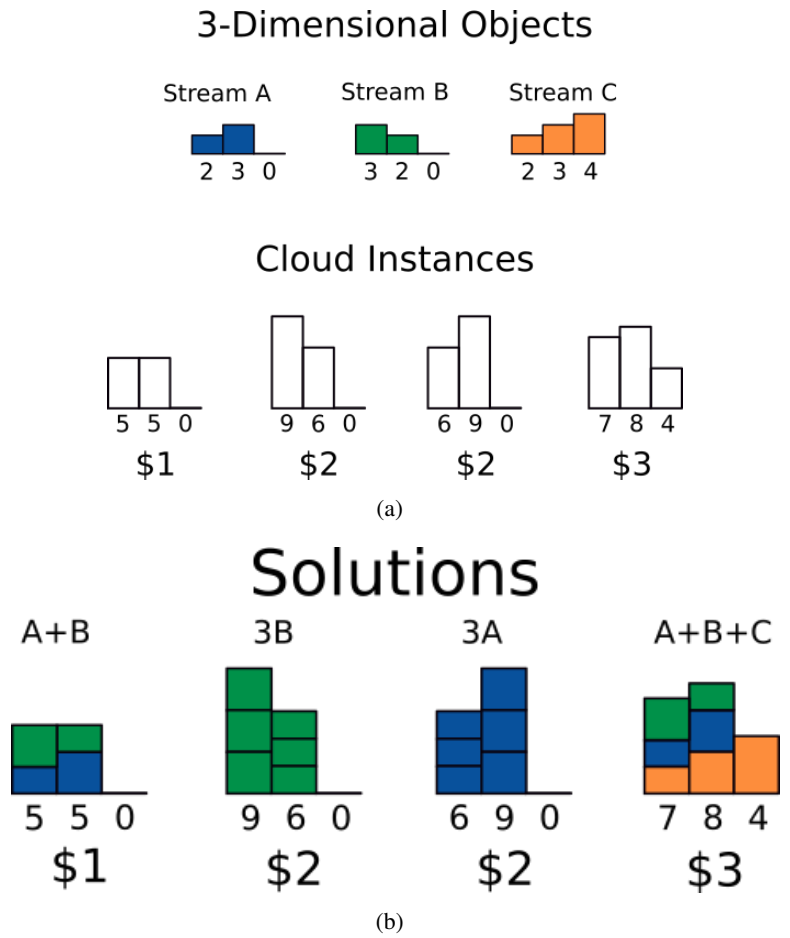


Fig. 2. (a) A three-dimensional multiple-choice packing problem. A - C represent three types of data streams whose resource requirements are expressed in three dimensions (CPU, Memory, and GPU). The heights represent the resource requirements, comparable to the width, height, and lengths of boxes. There are four choices of cloud instances each offering different sizes and costs. They are comparable to different types of trucks. The heights represent the sizes of these dimensions, comparable to the sizes of trucks. The goal is to fit the data streams (boxes) into the cloud instances (trucks). (b) Four possible solutions to accommodate different combinations of data streams.

Select Instances with CPU and GPU: The significant differences in price and performance of CPU-only and GPU-equipped instances makes it critical to select the most cost-effective instances when analyzing many real-time data streams. To effectively select CPU and GPU instances for this task, Kaseb et al. [7] formulated the problem as multi-dimensional multi-choice packing problem (please see the sidebar for explanation). This is illustrated in Figure 2 (a). In this figure, there are three types of data streams and four types of cloud instances. The goal is to fit the streams so that they completely fit inside the cloud instance and as little space is wasted as possible. Figure 2 (b) shows possible solutions to effectively pack different combinations of the data streams.

Kaseb's solution organizes the resource requirements into four dimensions: CPU, memory size, GPU, and GPU memory size. The method considers the frame sizes and frame rates of the video streams for determining the resource requirements needed to run analysis on different data streams. Due to the fluctuations in executing the analysis programs, the study discovers that when any dimension is more than 90% utilized, the performance starts to degrade. Thus, the

method keeps the utilization of each dimension below 90%. This multi-dimension, multi-choice optimization solution demonstrates considerable cost savings in different experimental settings.

Evaluation results from [7] are shown in Figure 3. The experiments use ten network cameras from CAM^2's database, with frame rates varying from 0.2 frames per second to 8 frames per second. Two object detection programs are used to analyze the data: VGG16 [11] and ZF [12]. At the highest frame rates, GPUs can accelerate these two analysis programs up to 16 times. At the lowest frame rates, the improvement falls below 5%. In other words, the benefits of GPUs are apparent only when the frame rates are high. At low frames rates, CPUs are preferred because of the lower costs. This solution can reduce the costs by as much as 61% by matching the resource requirements of the analysis programs and the cloud instances' capabilities.

Scenario	Analysis Programs	Frame Rate	Cameras
1	VGG-16	0.25	1
	ZF	0.55	3
2	VGG-16	0.20	1
	ZF	0.50	1
3	VGG-16	0.20	2
	ZF	8.00	10

Strategy	Number of Selected Instances		Hourly Cost (US$)	Cost Savings (%)
	Non-GPU	GPU		
ST1	4	-	$1.676	0%
ST2	-	1	$0.650	61%
ST3	-	1	$0.650	61%
ST1	1	-	$0.419	36%
ST2	-	1	$0.650	0%
ST3	1	-	$0.419	36%
ST1	Fail	Fail	Fail	Fail
ST2	-	11	$7.150	0%
ST3	1	10	$6.919	3%

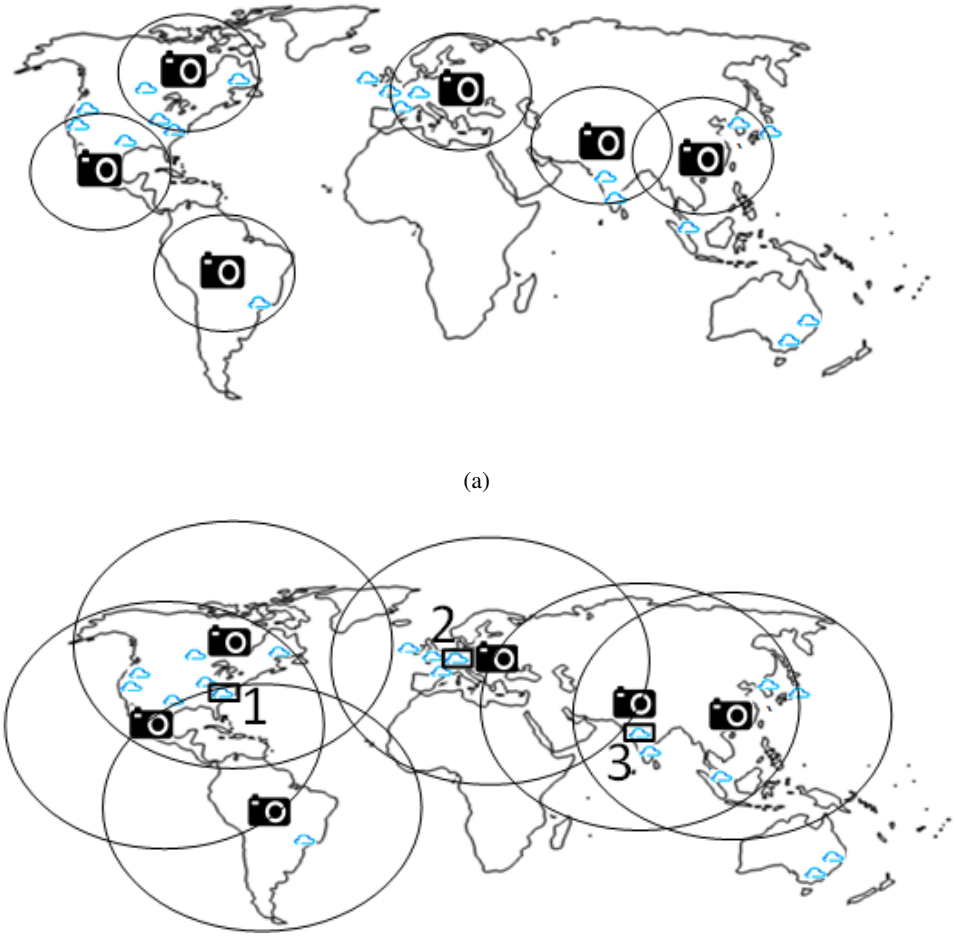
Fig. 3. Consider three scenarios and three unique instance selection strategies. Each scenario runs two analysis programs, VGG16 and ZF, at a different combination of frame rates and number of cameras. Strategy 1 (ST1) uses instances with only CPUs, strategy 2 (ST2) uses instances with only GPUs, and strategy 3 (ST3), Kasab's method, selects between GPU and CPU instances.

This study provides deep insights on how to offer computer vision services at lower costs as they become widely available in the cloud. This study, however, does not consider the geographical locations of network cameras. The next section explains how cameras' locations impact network distances and the cloud resource management of different types of instances.

Optimizing Instance Type and Location

In order to determine the most effective configuration of resources, the resource manager considers the cost of an instance in the context of its location. To make these considerations in practice, the location and type are first evaluated independently as follows.

Local Optimization: Table I shows that the same type of cloud instance at different locations can have different costs. Sometimes this cost disparity can exceed 60%. For example, the Azure `D8 v3` instance costs 63% more in Singapore than in Virginia USA ($\frac{0.625}{0.384} = 1.63$). A natural question is whether the data from network cameras should be sent to the cloud instances with the lowest prices. A prior study [5] shows that the observed frame rate is reduced when the distance (measured by network round-trip time) between a network camera and a cloud instance increases. Thus, when analyzing data streams from worldwide network cameras, the locations of the cameras and cloud instances must be considered.



(a)

(b)

Fig. 4. Consider six network cameras geographically distributed in America, Europe, and Asia. Amazon EC2 offers cloud instances in many locations (shows as the cloud icons). In (a), if high frames are required, the data must be analyzed by instances that are closer to the cameras. The circles mark the maximum distances the data can travel. As a result, six instances are needed. In (b), the required frame rates are lower and the circles are larger. One cloud instance can analyze multiple data streams and only three instances are needed. The three selected instances, marked with black boxes, are one potential solution.

Existing video cameras are designed for human viewing. For this purpose, 30 frames per second provide seamless experience. When video streams are analyzed by computers, the needed frame rates depend on the purposes. Though high frame rates are needed for tracking fast moving objects, low frame rates are sufficient for observing phenomena such as weather. Mohan et al. [13] study the necessary frame rates to track objects such as people walking, jogging, cycling etc. The study discovers that for cameras watching pedestrians walking, the frame rates can be reduced to as low as six frames per second. For objects that are far away from the cameras, even lower frame rates suffice.

Figure 4 illustrates the relationships between frame rates and geographical locations. In this figure, a small circle indicates a high frame rate. When a high frame rate is desired, the data stream can be sent only a short distance - measured by the round-trip time (RTT). This requires the resource manager to analyze the data stream at a cloud instance near the network camera. In Figure 4 (a), six separate cloud instances are needed because the circles do not overlap. If

a lower frame rate is acceptable, the acceptable RTT is higher and the circles can be larger, as shown in Figure 4 (b). One cloud instance is capable of analyzing multiple data streams. As a result, only the three boxed instances are needed and the cost can be reduced.

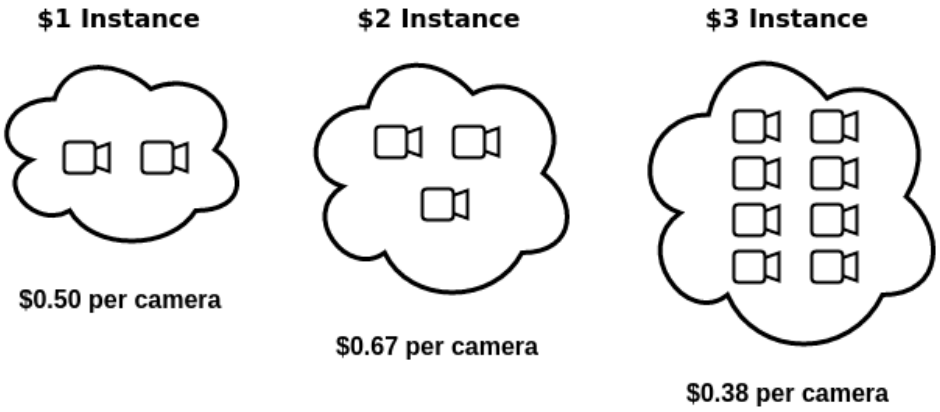


Fig. 5. There are 3 possible cloud instances, each with a different size and cost.

Instance Type Optimization: Typically, when multiple data streams are analyzed at one instance, additional cores, memory, or the presence of GPUs are required. This results in higher costs. Thus, to effectively optimize cloud instance usage, the resource manager has to consider the number of instances as well as their capabilities. Consider the example in Figure 5. Here, data from eight network cameras is analyzed, and the cloud manager must decide which instances to use. The cloud manager has the options to choose three types of instances at $1, $2, and $3 per hour. The first instance has the fewest cores and the least amount of memory; consequently, it can analyze only two data streams. The third type of instance, despite the higher cost, can analyze eight data streams at the lowest cost per stream.

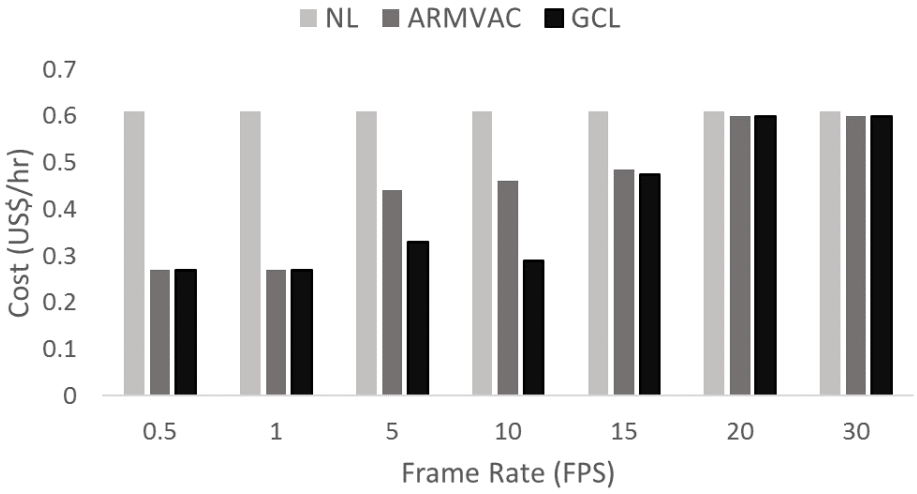


Fig. 6. Consider three different cloud resource managers: Nearest Location (NL), Adaptive Resource Management for Video Analysis in the Cloud (ARMVAC), and Globally Cheapest Location (GCL). A cost comparison between each resource manager is shown at various target frame rates.

Considering instances' types and locations simultaneously makes cloud resource management a complex optimization problem. Mohan et al. [6], [8] propose to first eliminate instance locations that are outside the acceptable RTT range. This method, named ARMVAC, then selects the lowest-cost instances from the remaining pool, and sends as many data streams to this instance while meeting the desired frame rates. This strategy performs well for high and low frame rates; streams with higher than 20 frames per second perform well since few instances can meet the processing requirements. Analyzing data streams with lower than one frame per second also performs well since there are few restrictions on instance requirements. The method does not perform well, however, when the desired frame rates are between one and twenty frames per second. In this range there are too many instance selections that can analyze the data. Mohan et al. [8] resolve this issue by formulating it as the multi-dimensional, multi-choice packing problem that accounts for the camera to cloud instance price ratio. This method, named Globally Cheapest Location (GCL), can reduce cost by as much as 56% compared with a resource manager that always selects the Nearest Location (NL) instances, and 31% compared with the ARMVAC method. An evaluation of the relationship between cost and frame rates is shown in Figure 6 which compares ARMVAC, GCL, and NL solutions. As explained earlier, the analysis programs' resource demands may vary due to a wide range of reasons. These methods can make resource decisions quickly and be applied during runtime. An experiment shows the adaptive solutions implemented in Amazon EC2 responding to the changing needs is presented in [14].

SUMMARY

With the rise of the "Internet of Video Things" [15], comes the possibility to make use of the massive amount of visual data. Analyzing the data requires a large amount of cloud computing resources. With the variety of cloud services available, it is important to optimize cloud-instance utilization to save money. This work proposes a cloud resource manager to make cost-effective use of both the real-time video data available on the Internet and the wide variety of cloud services available. The resource manager determines cost-effective ways to analyze video streams using cloud instances. It considers the geographic location of an instance relative to a camera, as well as the resources available in particular instances. By taking these factors into consideration, more than 50% cost can be saved when using a commercial cloud vendor.

ACKNOWLEDGEMENTS

This research project is supported by the National Science Foundation OAC-1535108, IIP-1530914, OISE-1427808, and CNS-0958487. We also acknowledge the Lynn CSE Fellowship at Purdue University, Amazon Web Services, Microsoft Azure, Google, Facebook, and Intel for their financial or technical supports. Thiruvathukal has a director's discretionary allocation from the Argonne National Laboratory to support the supercomputing aspects of this research. We thank the owners of the data for the permission to conduct the experiments. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the sponsors.

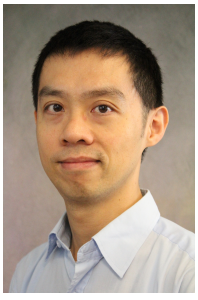
About The Authors

Zohar Kapach is pursuing his B.S degree in computer engineering from Purdue University. He is leading the transfer learning research team in the CAM^2 project. His research interests include cloud resource optimization, deep learning, and computer vision.
Email: zkapach@purdue.edu

Andrew Ulmer is a senior undergraduate student studying computer engineering and statistics at Purdue University. His interests include deep learning, computer vision, and entrepreneurship.
Email: ulmera@purdue.edu

Daniel Merrick is pursuing his B.S degree in electrical engineering from Purdue University, West Lafayette with an expected graduation date of Fall 2019. His research interests include machine learning and computer vision.
Email: dmerrick@purdue.edu

Arshad Alikhan is currently a BS student in Computer Science at Purdue University with a concentration in Machine Learning and a Mathematics minor. He is currently doing research with CAM2 in the area of transfer learning. His research interests include machine learning and cloud computing.
Email: aalikhan@purdue.edu



Yung-Hsiang Lu is a professor in the School of Electrical and Computer Engineering and (by courtesy) the Department of Computer Science of Purdue University. He is an ACM distinguished scientist and ACM distinguished speaker. Dr. Lu is a co-founder and the scientific adviser of a technology company using video analytics to improve shoppers' experience in physical stores.
Email: yunglu@purdue.edu

Anup Mohan obtained his Ph.D. from the School of Electrical and Computer Engineering at Purdue University in 2017. His research interests include large-scale video analysis, cloud computing, and big data analysis. Anup Mohan is currently working at Intel Corporation, Santa Clara, U.S.A.
Email: anup.mohan@intel.com

Ahmed S. Kaseb is an assistant professor of computer engineering in the Faculty of Engineering at Cairo University. He obtained the Ph.D. in computer engineering from Purdue University in 2016. He obtained the M.S. and B.E. in computer engineering from Cairo University in 2013 and 2010 respectively.
Email: akaseb@eng.cu.edu.eg

George K. Thiruvathukal is a Professor of Computer Science at Loyola University Chicago and visiting faculty at Argonne National Laboratory in the Argonne Leadership Computing Facility.
Email: gkt@cs.luc.edu

REFERENCES

[1] Cisco, *Cisco visual networking index: Forecast and methodology, 2016-2021*, Document ID:1465272001663118, Sep. 2017.

[2] N. Jenkins, *245 million video surveillance cameras installed globally in 2014*, IHS Markit Insight, Jun. 2015.

[3] A. S. Kaseb, Y. Koh, E. Berry, K. McNulty, Y. H. Lu, and E. J. Delp, "Multimedia content creation using global network cameras: The making of cam2," in *2015 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, Dec. 2015, pp. 15–18.

[4] A. S. Kaseb, E. Berry, Y. Koh, *et al.*, "A system for large-scale analysis of distributed cameras," in *2014 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, Dec. 2014, pp. 340–344.

[5] W. Chen, A. Mohan, Y. H. Lu, T. Hacker, W. T. Ooi, and E. J. Delp, "Analysis of large-scale distributed cameras using the cloud," *IEEE Cloud Computing*, vol. 2, no. 5, pp. 54–62, Sep. 2015.

[6] A. Mohan, A. S. Kaseb, Y.-H. Lu, and T. Hacker, "Adaptive resource management for analyzing video streams from globally distributed network cameras," *IEEE Transactions on Cloud Computing*, 2018.

[7] A. S. Kaseb, B. Fu, A. Mohan, Y.-H. Lu, A. Reibman, and G. K. Thiruvathukal, "Analyzing real-time multimedia content from network cameras: Using cpus and gpus in the cloud," *ArXiv preprint arXiv:1802.08176*, 2018.

[8] A. Mohan, A. S. Kaseb, Y.-H. Lu, and T. J. Hacker, "Location based cloud resource management for analyzing real-time videos from globally distributed network cameras," in *Cloud Computing Technology and Science (CloudCom), 2016 IEEE International Conference on*, IEEE, 2016, pp. 176–183.

[9] F. Brandao and J. P. Pedroso, "Bin packing and related problems: General arc-flow formulation with graph compression," *Computers & Operations Research*, vol. 69, pp. 56–67, 2016.

[10] ——, "Multiple-choice vector bin packing: Arc-flow formulation with graph compression," *ArXiv preprint arXiv:1312.3836*, 2013.

[11] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *ArXiv preprint arXiv:1409.1556*, 2014.

[12] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *European conference on computer vision*, Springer, 2014, pp. 818–833.

[13] A. Mohan, A. S. Kaseb, K. W. Gauen, Y. Lu, A. R. Reibman, and T. J. Hacker, "Determining the necessary frame rate of video data for object tracking under accuracy constraints," in *2018 IEEE Conference on Multimedia Information Processing and Retrieval (MIPR)*, Apr. 2018, pp. 368–371.

[14] A. S. Kaseb, A. Mohan, and Y. Lu, "Cloud resource management for image and video analysis of big data from network cameras," in *2015 International Conference on Cloud Computing and Big Data (CCBD)*, Nov. 2015, pp. 287–294.

[15] Y.-K. Chen and S.-Y. Chien, "Perpetual video camera for internet-of-things," in *2012 Visual Communications and Image Processing*, Nov. 2012, pp. 1–7.