Exploring Instance Heterogeneity in Public Cloud Providers for HPC Applications

Eduardo Roloff[1], Matthias Diener[2], Luciano P. Gaspary[1] and Philippe O. A. Navaux[1]
[1]*Informatics Institute, Federal University of Rio Grande do Sul, Porto Alegre, Brazil*
[2]*University of Illinois at Urbana-Champaign, U.S.A.*

Keywords: Cloud Computing, Cost Efficiency, Heterogeneity, Microsoft, Amazon, Google.

Abstract: Public cloud providers offer a wide range of instance types with different speeds, configurations, and prices, which allows users to choose the most appropriate configurations for their applications. When executing parallel applications that require multiple instances to execute, such as large scientific applications, most users pick an instance type that fits their overall needs best, and then create a cluster of interconnected instances of the same type. However, the tasks of a parallel application often have different demands in terms of performance and memory usage. This difference in demands can be exploited by selecting multiple instance types that are adapted to the demands of the application. This way, the combination of public cloud heterogeneity and application heterogeneity can be exploited in order to reduce the execution cost without significant performance loss. In this paper we conduct an evaluation of three major public cloud providers: Microsoft, Amazon, and Google, comparing their suitability for heterogeneous execution. Results show that Azure is the most suitable of the three providers, with cost efficiency gains of up to 50% compared to homogeneous execution, while maintaining the same performance.

1 INTRODUCTION

Executing large parallel applications (such as those from the High Performance Computing (HPC) domain) in the cloud has become an important aspect of cloud computing in recent years (Foster et al., 2008; Buyya et al., 2009; Armbrust et al., 2009). By running these applications in the cloud, users can benefit from lower up-front costs, higher flexibility, and speedier hardware upgrades compared to traditional clusters. However, raw performance as well as cost efficiency for long-term usage can be a disadvantage (Roloff et al., 2012). In recent years, several approaches were introduced to use the cloud efficiently when executing HPC applications (Abdennadher and Belgacem, 2015; d. R. Righi et al., 2016; Zhang et al., 2015).

Many public cloud providers offer different types of instances to users, which contain different CPU types and memory sizes, in contrast to traditional clusters, which in most cases consist of homogeneous machines. Furthermore, many HPC applications consist of tasks that have different computational demands, due to load imbalance or uneven work distribution. By matching the demands of the application to the characteristics of the instance types offered by a cloud provider, creating a cluster of heterogeneous instances, and mapping each task to its most appropriate instance, we can create a more cost-efficient execution of the application in the cloud (Roloff et al., 2018).

In this paper, we use a parallel application, *ImbBench*, to analyze three different cloud providers (Microsoft Azure, Amazon AWS, and Google Cloud) in terms of their suitability for such a heterogeneous execution of large parallel applications. ImbBench is an MPI-based application that can create different imbalance patterns in CPU and memory usage that mimics the behavior of real-world HPC applications (Roloff et al., 2018). We evaluate the cost efficiency as well as performance when running ImbBench in homogeneous and heterogeneous instances with different imbalance patterns and computational demands.

Our results show that the suitability for heterogeneous execution is very different between the three providers. Azure benefited the most from heterogeneous execution, improving cost efficiency by up to 50% for the memory-bound 8Level pattern, while maintaining the same performance. On average, Azure obtains gains of 30% in cost-efficiency by us-

Roloff, E., Diener, M., Gaspary, L. and Navaux, P.
Exploring Instance Heterogeneity in Public Cloud Providers for HPC Applications.
DOI: 10.5220/0007799302100222
In *Proceedings of the 9th International Conference on Cloud Computing and Services Science (CLOSER 2019)*, pages 210-222
ISBN: 978-989-758-365-0

ing heterogeneous configurations. AWS benefits less from heterogeneous execution, as in our tests the cheapest instance was also the fastest. Google Cloud only offers instances that differ in memory size, but not in CPU or memory speed, and therefore does not show potential for heterogeneous execution currently. These results show that heterogeneous execution can be beneficial in situations where the cloud provider offers a sufficient variety of instance types. It also motivates the future analysis of other aspects, such as I/O and network performance.

2 BACKGROUND: CLOUD INSTANCE TYPES

There are many providers in the public cloud, with a large set of instance type offered to the users. The goal of this paper is to evaluate whether it is possible for a cloud user to benefit from this level of heterogeneity to create adequate executions environments for his applications.

In this work, we will evaluate the suitability for heterogeneity of three major IaaS providers: Amazon, Microsoft, and Google. These three providers represent the state of the art of the public cloud providers. Each provider has its own classification for the available services and instance configurations.

Amazon divides their instances into five groups: General purpose, Compute Optimized, Memory Optimized, Accelerated Computing and Storage Optimized. General purpose instances includes balanced machines, in terms of memory and CPU amounts, this group includes machines with ARM processors as well. Compute, Memory and Storage Optimized instances focus on one of the three components. The Accelerated Computing group contains the instances equipped with GPUs and FPGAs. In terms of number of CPUs Amazon ranges from instances with a shared core up to instances with 128. The available memory ranges from 500MB up to almost 4TB.

Microsoft classifies instances into six groups: General purpose, Compute optimized, Memory optimized, Storage optimized, GPU and High performance compute. The General purpose group aims to provide a balanced memory to CPU ratio and was designed for small workloads. Similar to Amazon, the Compute, Memory and Storage optimized instances focus in one of the three instance components. The GPU group provides instances with single or multiple NVIDIA GPUs. The High performance compute group provides instances with substantial CPU power and several memory configurations, low latency InfiniBand network is available as well. The CPU range of Microsoft ranges from 1 to 72 CPUs and the memory starts from 1GB up to near 4TB.

Google has five groups of pre-configured instances: Standard, High-memory, High-CPU, Shared-core and Memory-optimized. The Standard group has a fixed ratio of 3.75 GB of memory per CPU and are instances for balanced workloads. The High-memory and High-CPU instances have, respectively, 6.50GB and 0.90 GB of memory per CPU. The Shared-core group provides instances that use less than a single core, sharing the same core with other instances. The Memory-optimized group presents instances with large amounts of memory, suitable for workloads that perform in-memory compute.

Google has the capability to attach GPUs to the available instances, with some exceptions, however it does not provide a group of GPU pre-configured instances. The pre-defined instances start from 0.2 CPUs up to 96 and from 600MB up to 4TB of memory. Moreover, Google has the unique capability to customize the instances according to the user's needs. In this case, the user can configure instances starting from 1 up to 96 cores and the memory starts from 1 GB up to 1.4TB. This feature seems very interesting, because the user could configure instances with the amount of memory that the application needs.

In terms of price, there is a significant variation as well. The price of the same instance type presents variation inside the same provider. This variation is due to the datacenter location where the instance is allocated. For example, a c5.2xlarge Amazon instance costs US$ 0.34 in the Oregon datacenter and US$ 0.524 in the Sao Paulo datacenter, a difference of 54%. Because of the variation within the same provider, we will not present the instance price range of the three providers.

Table 1 shows the instance prices as well as the performance result of the High-Performance Linpack (HPL) (Dongarra et al., 2003) benchmark, which is widely used to measure computing performance. Table 2 shows the results of the memory performance measured using the STREAM (McCalpin et al., 1995) benchmark. Due to this heterogeneity in the public cloud, we explore the variety of instances in order to benefit the user to reduce the cost without losing performance.

3 RESULTS

This section presents the results of our evaluation of homogeneous and heterogeneous cloud cluster with ImbBench.

Table 1: Characteristics of the Cloud Instances. Linpack results are given in GFlops/sec.

Instance type	Price/hour	Memory (GB)	Linpack
Azure			
A8	US$ 0.975	56	149.81
A8v2	US$ 0.40	16	74.25
D4	US$ 0.559	28	261.53
D13	US$ 0.741	56	232.91
E8	US$ 0.593	64	127.62
F8	US$ 0.498	16	218.50
G3	US$ 2.44	112	291.76
H8	US$ 0.971	56	332.76
H8m	US$ 1.301	112	326.25
L8	US$ 0.688	64	259.52
Google			
n1-standard-8	US$ 0.38	30	136.96
n1-highmem-8	US$ 0.4736	52	137.41
n1-highcpu-8	US$ 0.2836	7.2	135.03
Amazon			
c4.2xlarge	US$ 0.398	15	182.31
c5.2xlarge	US$ 0.34	16	194.25
m4.2xlarge	US$ 0.40	32	154.16
m5.2xlarge	US$ 0.384	32	159.52
r4.2xlarge	US$ 0.532	61	154.17
r5.2xlarge	US$ 0.504	64	161.80
t2.2xlarge	US$ 0.3712	32	298.18
t3.2xlarge	US$ 0.3328	32	159.50

Table 2: Results of the STREAM benchmark for the Cloud Instances (in GB/sec).

Instance type	Copy	Scale	Add	Triad
Azure				
A8	11.85	13.11	13.79	13.62
A8v2	5.86	6.39	6.68	6.69
D4	11.03	11.00	12.42	12.26
D13	10.09	9.85	11.23	11.15
E8	8.99	9.46	10.07	10.15
F8	10.50	10.49	12.00	11.70
G3	11.53	11.85	13.39	13.16
H8	13.06	12.78	14.70	14.39
H8m	13.08	12.88	14.71	14.48
L8	9.96	10.10	11.33	11.20
Google				
n1-standard-8	12.18	9.37	9.98	9.88
n1-highmem-8	10.01	9.76	10.57	10.50
n1-highcpu-8	12.18	9.49	10.25	10.20
Amazon				
c4.2xlarge	19.72	11.98	13.39	13.22
c5.2xlarge	10.56	12.50	13.25	13.26
m4.2xlarge	16.75	9.03	9.88	9.85
m5.2xlarge	10.25	11.99	12.71	12.68
r4.2xlarge	18.40	9.82	10.79	10.77
r5.2xlarge	10.45	12.12	13.10	13.05
t2.2xlarge	18.94	11.06	12.29	12.35
t3.2xlarge	9.95	10.83	11.59	11.51

3.1 Methodology

3.1.1 Hardware and Software Configuration

For all experiments, we configure a cluster of 32 cores, consisting of four instances with eight cores each. All instances run Ubuntu 18.04 LTS. To provide a fair comparison between the providers and their instances, we chose datacenter locations as close as possible. For Amazon we chose the ”Oregon” region, for Microsoft the ”West US” region and for Google the ”Oregon” region.

For each experiment, we run 30 executions and show average values as well as the standard deviation. For each instance type presented in Section 2, we run a homogeneous cluster, but discard the types that did not perform well or cost too much. The following instance types were used in this section:

- **Azure:** H8, F8, A8v2
- **AWS:** c5.2xlarge, t2.2xlarge
- **Google:** n1-highcpu-8, n1-highmem-8, n1-standard-8

3.1.2 ImbBench

We used *ImbBench* as the workload to perform the tests of the cloud configurations. ImbBench is a proto-benchmark that implements several load patterns for MPI processes (Roloff et al., 2017). ImbBench can produce several imbalance patterns. In this paper, we use the following four patterns: 8Level, Amdahl, Linear, and Balanced. In the 8Level pattern, there are 8 distinct load levels for the MPI ranks, and there is an equal number of ranks for each level. In the Amdahl pattern, the first rank has more work than all other ranks. In the Linear pattern, the load increases linearly among all ranks. The Balanced pattern has the same load across all ranks. It is used to verify the behavior of a homogeneous application.

It is possible to perform CPU performance tests, where the workload is a random number generator, that was profiled with the PERF tool (de Melo, 2010) as almost 100% CPU-bound computation. The memory performance could be measured as well, in this case the workload of ImbBench is a BST manipulation, in this case the PERF profiling results in an application that is 50% memory-bound.

3.1.3 Cost Efficiency Metric

To measure the cost of the configuration, we use the Cost Efficiency methodology (Roloff et al., 2012; Roloff et al., 2017). In a summarized way, this methodology helps the user to determine which

provider and configuration offers the higher performance for the price paid. We define cost efficiency as the number of executions that could be made in an hour with a given application, divided by the price per hour of the cloud. Equation 1 formalizes this definition, where *execution time [hours]* represents the execution time of an application in hours, and *price per hour* represents the price per hour of the cloud where the application was executed on.

$$CostEfficiency = \frac{\frac{1\ hour}{execution\ time\ [hours]}}{price\ per\ hour} \quad (1)$$

The result values are just to compare two environments for a given application, where higher values of indicate higher efficiency. The value itself does not have another meaning.

3.2 Microsoft Azure

Microsoft Azure has the largest number of instances with 8 cores among the three providers of this study. We selected 10 instance types for our evaluation, covering the whole spectrum of available configurations. In terms of processing power, we observed a range of the HPL results starting from 74 GigaFlops up to 332 GigaFlops, as shown in Table 1. Otherwise, in terms of memory performance the results presented less variation than the processing power, but it is still possible to observe different performance groups, as seen in Table 2. Those results lead us to state that Microsoft Azure presents an interesting heterogeneity among its available instances. We used the homogeneous cluster that presented better performance, the H8 cluster, as the baseline for all the tests and create heterogeneous clusters to improve the cost efficiency of the execution.

3.2.1 CPU

To evaluate the CPU performance of Microsoft Azure, we performed tests with different instances types. In order to identify if the combination of different instance types could be used to reduce the execu-

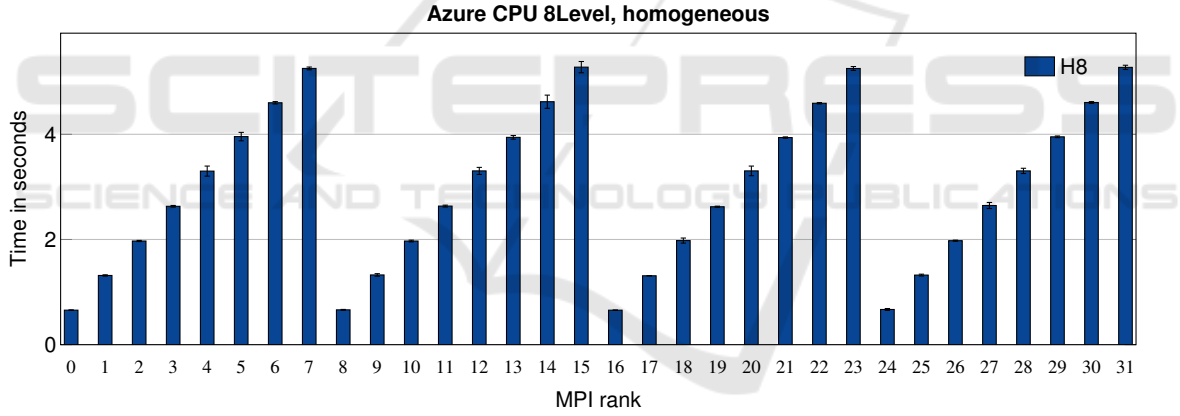

Figure 1: Results for the 8Level pattern and CPU performance for the homogeneous H8 cluster.

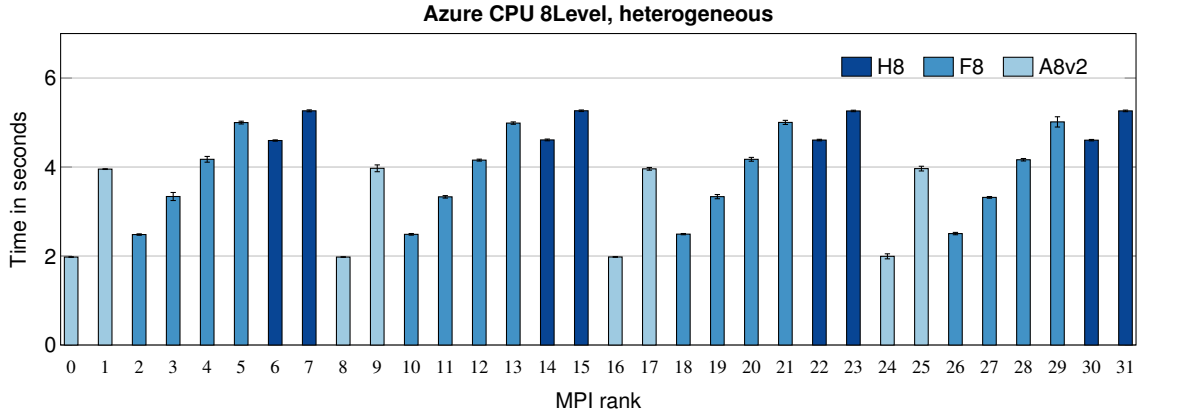

Figure 2: Results for the 8Level pattern and CPU performance for the Azure heterogeneous cluster.

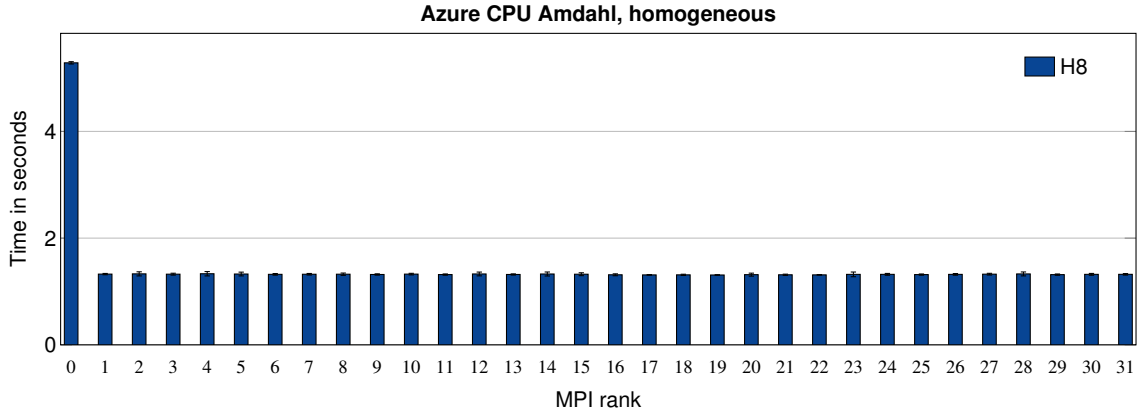

Figure 3: Results for the Amdahl pattern and CPU performance for the homogeneous H8 cluster.

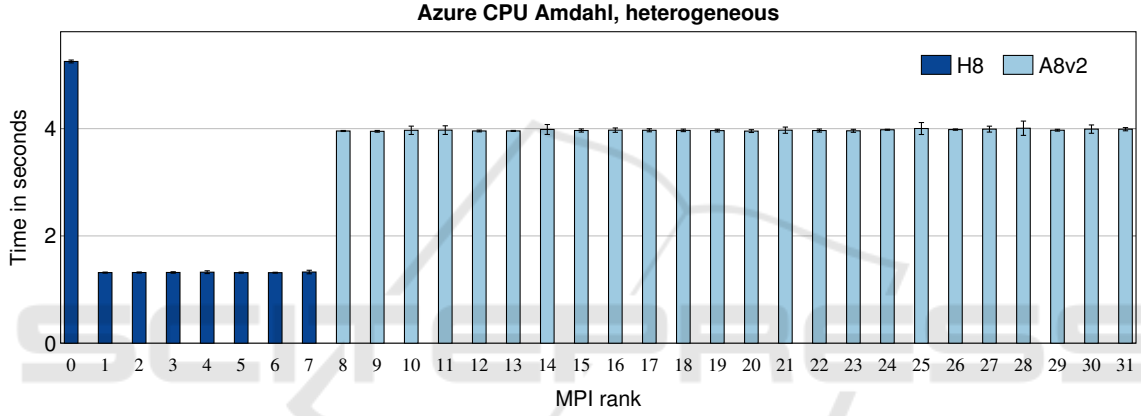

Figure 4: Results for the Amdahl pattern and CPU performance for the Azure heterogeneous cluster.

tion cost, we build several combinations of instances. We show and discuss three different patterns here: 8Level, Amdahl and Linear.

The 8Level pattern simulates an application whose processes present eight different load levels. It is quite common that applications present different levels of imbalance, and this pattern of ImbBench simulates that kind of applications. Figure 1 shows the execution of the 8Level pattern using the homogeneous H8 cluster. We can note the load distribution levels in the eight groups. MPI ranks 7, 15, 23 and 31 were the ones that took most time to execute, around 5 seconds. This means that all the other processes have certain level of idle time during the application execution.

In order to reduce the execution cost without losses in the execution time, we performed experiments with combinations of different instance types. In the Figure 2 we show the combination that presented the best balance between price and performance. This configuration is a combination of one H8 instance, two F8 instances and one A8v2 instance. The H8 instance was used to execute the processes with more load, the F8 instances were used to execute the processes with medium load and the A8v2 instance the processes with low load level. As seen in the figure, we were able to maintain the same total execution time. However, due to the combination with cheaper instances, the cost of the execution was reduced.

In terms of cost efficiency, the homogeneous cluster has a factor of 175 and the heterogeneous cluster has a results of 288. This means that the heterogeneous configuration is more efficient. In terms of cost per hour, the heterogeneous configuration present a reduction of US$ 1.51, or 40%, without performance loss.

The Amdahl pattern represents an application that has one process that performs much more computation than all the others. Normally this situation occurs in applications that were implemented to centralize the work-flow in a master process and the other processes receive and send data to it. In Figure 3 we can observe the Amdahl execution time, using a ho-

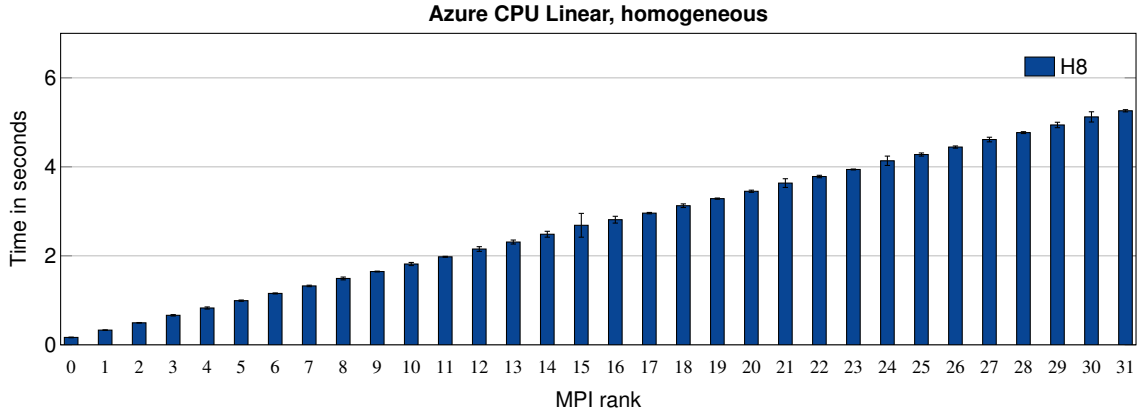

Figure 5: Results for the Linear pattern and CPU performance for the homogeneous H8 cluster.

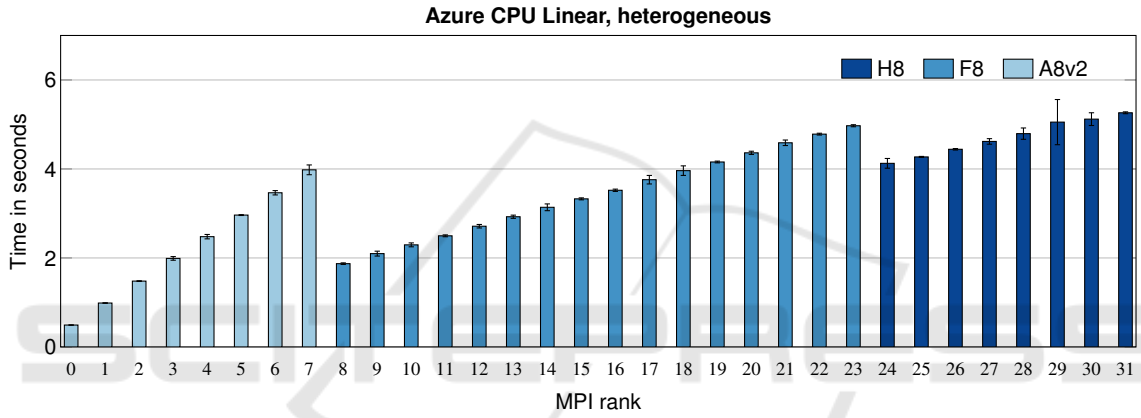

Figure 6: Results for the Linear pattern and CPU performance for the Azure heterogeneous cluster.

mogeneous cluster built of H8 Azure instances. We can observe that process 0 presents an execution time of around 5 seconds and the other 31 processes execute in less than 2 second. This means that we have a large amount of idle time during the application execution. Moreover, in a Cloud Computing environment the user is charged as well for this idle time.

Building a cluster with different instances types could help the user to reduce his execution cost, without significant losses in execution time. Figure 4 shows the execution of the same application by using an heterogeneous cluster. The heterogeneous cluster was configured with one H8 instance and three A8v2 instances. As we can note in the figure, the process 0 still was the critical-path of the application, taking around 5 seconds. Processes 8 to 31 took around 4 seconds, because they where executed in the instances with less performance. However, the total execution time remain the same as on the homogeneous cluster.

In terms of cost efficiency, the homogeneous cluster for the Amdahl pattern results in a factor of 175 and the result for the heterogeneous cluster was 315. This means, again, that the heterogeneous configuration is more efficient. In terms of cost per hour, the heterogeneous configuration present a reduction of US$ 1.71, or 45%, without performance loss.

The Linear pattern presents a situation where each process of the application executes a different load. The process 0 has the lowest load and the process n, in our case process 31, has the highest load of the application. This patter was designed to explore the scalability capacity of a given configuration. Figure 5 shows the results of the Linear pattern in the H8 cluster. We can note that the load increases according with the processes number increases as well. The process with the highest rank, in our case 31, executes the biggest load, meaning that it is the one determining the total execution time. All the other processes execute in less time, so we have a situation of idle time again.

In order to reduce the overall idle time, we propose the execution of the application using a combination of different instances. We could achieve this by using instances with less performance and price. Figure 6 shows the execution of the application in a heterogeneous cluster. In this case, we configured a

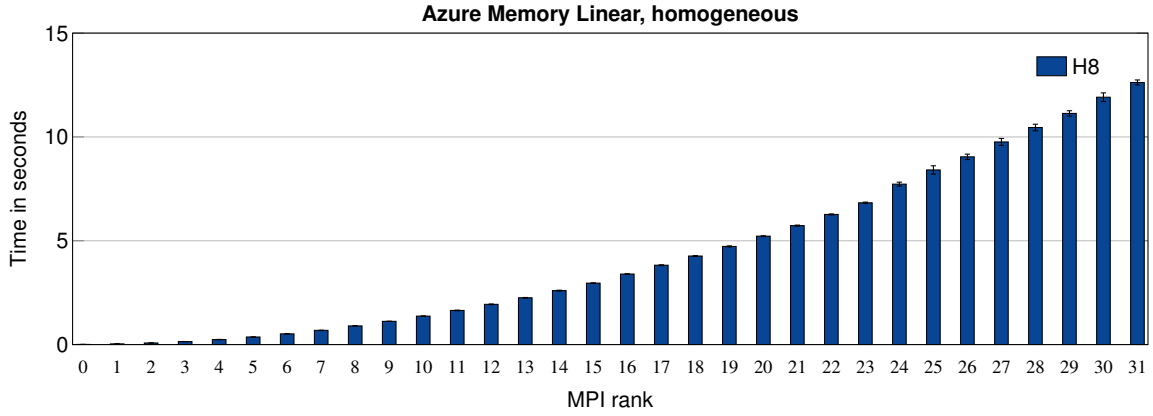

Figure 7: Results for the Linear pattern and Memory performance for the homogeneous H8 cluster.

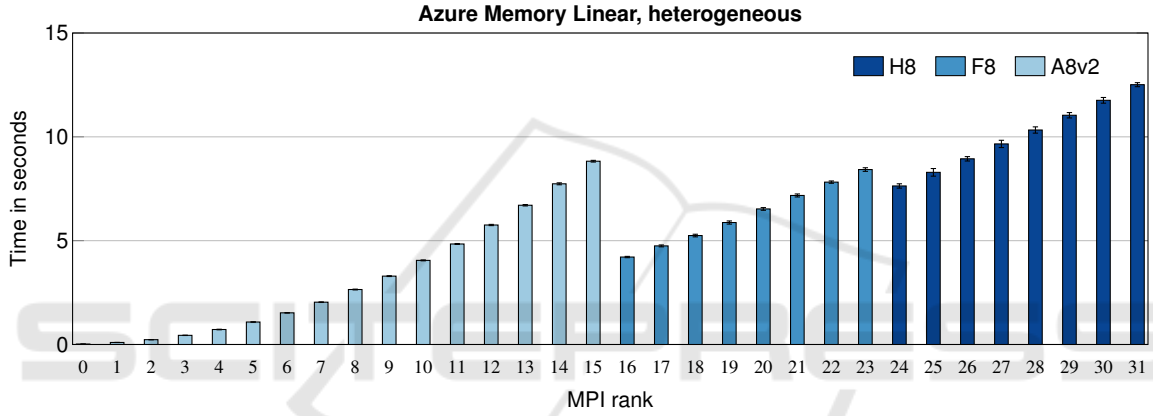

Figure 8: Results for the Linear pattern and Memory performance for the Azure heterogeneous cluster.

cluster with one A8v2 instance, two F8 instances and on H8 instance. The execution time remains the same as the homogeneous cluster. However the idle time was reduced, by using the cheaper instances.

The cost efficiency repeats the results of the 8Level pattern, where the homogeneous cluster was of 175 and for the heterogeneous cluster was 288. The cost per hour reduction was adherent with the 8Level pattern as well, with a reduction of US$ 1.51, or 40%.

3.2.2 Memory

In terms of memory, there is less heterogeneity among the Microsoft Azure instances. Only the A8v2 instance presented a result that is far different from the other instances. The rest of the instances are close, but we still can group them into two groups. We could separate the instances that performed 13 and 14 GBytes/sec from the instances that performed 10,11 and 12 GBytes/sec. The first group is composed by the A8, G3, H8 and H8m instances; and the second contains the D4, D13, E8, F8 and L8 instances. These groups have close results but, after some experimentation, we found some heterogeneity that could be exploited.

Figure 7 shows the results of ImbBench Linear pattern for testing the memory in a homogeneous H8 cluster. As observed, the growth of execution time is linear, according with the increase of the process number. Likewise the CPU testing, the process with the biggest rank, in our experiments the process 31, is the critical one for the execution time.

After some experiments with different instance combinations, we found a configuration that does not increase the execution time while reduces the cost. Figure 8 shows the results of a cluster composed of two A8v2 instances, one F8 instance and one H8 instance. As shown, the total execution time was not affected, because the processes with higher load, 24-31) were stile executed in the most powerful instance. And the processes with less load, 0-23, were executed by using less powerful instances. This configuration presented the best cost-efficiency without increasing the execution time.

The cost efficiency for the memory Linear pattern was 73 for the homogeneous cluster and for the het-

erogeneous was 126. The cost per hour was reduced as well, presenting a reduction of US$ 1.61, or 42%.

3.3 Google Cloud Platform

The Google instances presented similar results in terms of processing power, as observed in Table 1, where the three instances performed around 135 Gigaflops in the HPL benchmark. In terms of memory performance, the instances presented almost the same results, with low variation. For a practical comparison, we used the TRIAD test which represents a mix of all the STREAM tests. The n1-standard-8, n1-highmem-8 and n1-highcpu-8 memory results are shown in Table 2.

We executed the whole suite of ImbBench in the three Google instances. We observed that the instances presented the about same performance for both CPU and memory.

3.3.1 CPU

In terms of processing power measured with the ImbBench, the Google instances presented a homogeneous behavior, which was expected due to the HPL results. We present the ImbBench results only for the Balanced and Linear patterns, because these two patterns contain two situations that can help to identify the homogeneity of instances.

First, the Balanced pattern means a full load of all the available cores of the instance for the entire execution time. The Figure 9 presents the Balanced pattern results. We could observe that there is a low variation with all the 32 processes executing their loads in around 10 seconds for the three Google instances. We do not observe variation between the four instances of the cluster as well. The Standard Deviation presented, also, a low rate meaning that the execution times were constant among the 30 executions of our experiments.

Secondly, the Linear pattern presents a variable load inside the instances processes and between instances as well. Each one of the 32 processes executed a different load. The Figure 10 presents the Linear pattern, and the homogeneity could be observed as well. We could observe that there is a homogeneous behavior in all of the 32 levels of the ImbBench. This means, as in the Balanced pattern, that the three instances types of Google have a homogeneous behavior. Like in the Balanced pattern, the Standard Deviation presented a low rate showing the constance in the execution time.

3.3.2 Memory

In terms of memory performance that we measured with ImbBench, the Google instances presented a homogeneous-like behavior. This situation was expected because of the STREAM results, where the three instances presented convergent results. As in the CPU evaluation, we just present the ImbBench results for the Balanced and Linear patterns.

The Balanced pattern means that each process executes a full load of memory operations, during the application execution. The Figure 11 shows the Balanced pattern results for the Memory evaluation. As we can observe, the behavior is slightly different than the CPU results. We have different performance results in all the four instances os the cluster. It is possible to observe that in the first group of processes, 0 to 7, the standard instance performed worst than the highmem and highcpu instances. The second group of processes presented slightly better results for the standard instance, but still worst than the other ones, and the highcpu instance performed worst as well. The third group, 16 to 23, presented similar results for all the three instance types. Finally in the fourth group, 24 to 31, the standard machine presented better performance than the highmem and highcpu instances. Otherwise, the Standard Deviation rates were low, meaning that the executions presented a constant behavior. This could be explained with the concurrency of the instances over the same hardware, which means that the instances are not 100% isolated.

The Linear pattern results are shown in Figure 12. Different from the Balanced pattern, we observed a homogeneous behavior in the three first groups of processes, from process 0 up to 23. These results means that when the memory pressure is lower, all the three instances respond with good performance. However, in the fourth group of processes, 24 to 31, there were different results among the three instances. In this group, the highmem and highcpu instances performed worse than the standard instance type. This situation could also be explained due to the concurrency level in the underline hardware. However, this is just regarding to our experience in virtualization, because we do not have access to this level of information from the provider.

The discussion of cost efficiency is not necessary for the Google instances. The performance of n1-standard-8, n1-highmem-8, and n1-highcpu-8 are practically the same, both in terms of memory and processing power. The price difference among them is relative to the amount of resources, where the machines with less resources are cheaper than the ones with more resources. The major difference is in terms

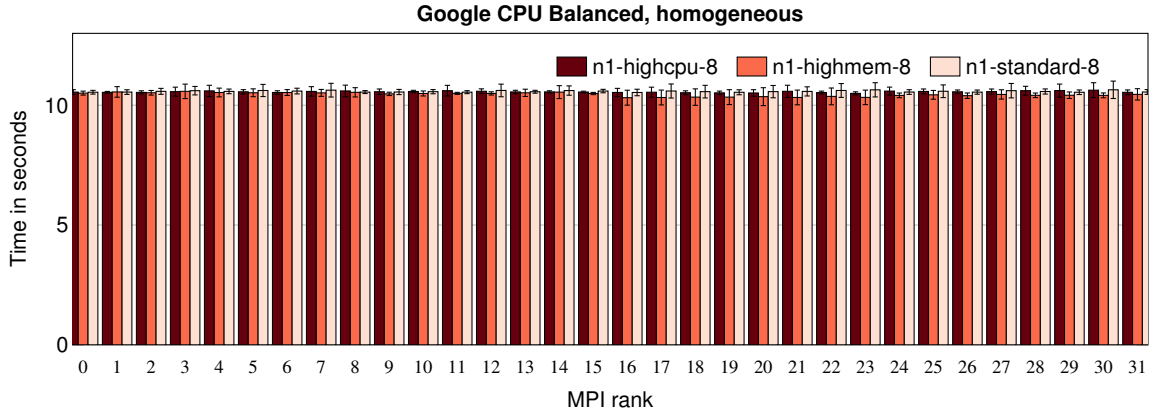

Figure 9: Results for the Balanced pattern and CPU performance for the homogeneous Google instances cluster.

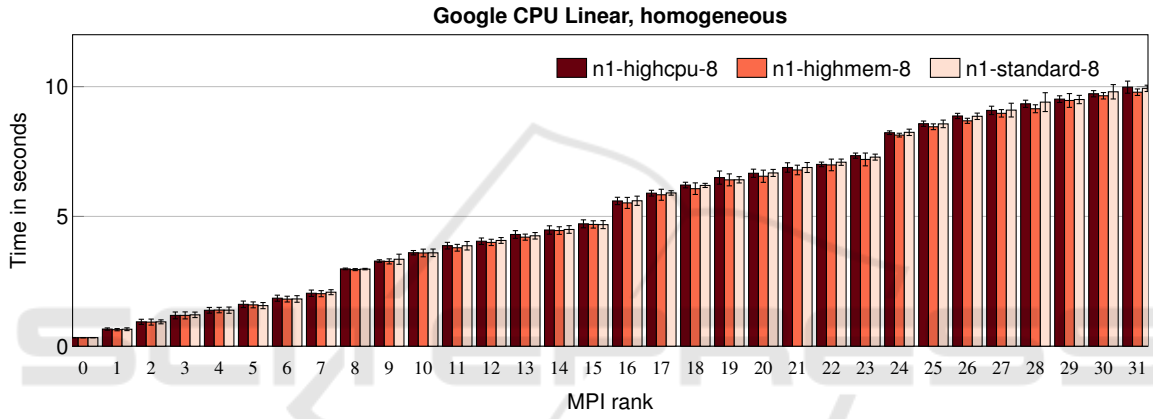

Figure 10: Results for the Linear pattern and CPU performance for the homogeneous Google instances cluster.

of memory amount, and not memory performance. The ImbBench memory test is related to the memory performance and does not use the total size of the memory.

3.4 Amazon Web Services

We selected eight different instances from the Amazon Web Services provider. The AWS instances do not present a high level of heterogeneity in terms of processing power. In Table 1 we can observe that only the t2.2xlarge instance presented an HPL result that was far from the other instances. The other seven instances presented similar performance results. In terms of memory, all the instances presented similar results, with exception of the m4.2xlarge instance, that performed lower than the others.

3.4.1 CPU

In terms of CPU evaluation, we performed several tests with the AWS instances. However, only the c5.2xlarge and t2.2xlarge instances were interesting to conduct an evaluation. Due to the price-performance relation these two instances outperformed all the others, both in terms of performance and cost-efficiency.

Figure 14 shows the result of the Balanced pattern of ImbBench for the two instances. We could note that the performance of the t2.2xlarge was better than the c5.2xlarge, with the first presenting an execution time of around 6.5 seconds and the second one around 8 seconds. However, these results are not as good as expected, because in the HPL profile the t2.2xlarge presented results 50% better than the c5.2xlarge. These results could be explained due to the underline hardware and the configuration of them. During the HPL tests, we note that the eight cores of the c5.2xlarge instance were 4 cores with hyperthreading and the t2.2xlarge delivered eight cores. Moreover, the c5.2xlarge uses an Intel Xeon Platinum 8000 series as processor and was designed for computing performance and the t2.2xlarge does not specify the model of the processor, just mentions that it is an Intel Xeon. These were our findings to explain the results of the Balanced pattern.

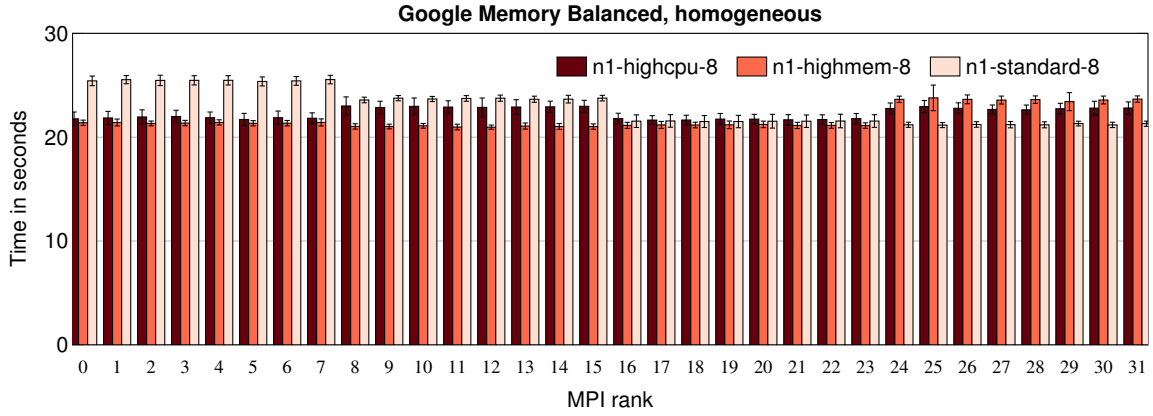

Figure 11: Results for the Balanced pattern and Memory performance for the homogeneous Google instances cluster.

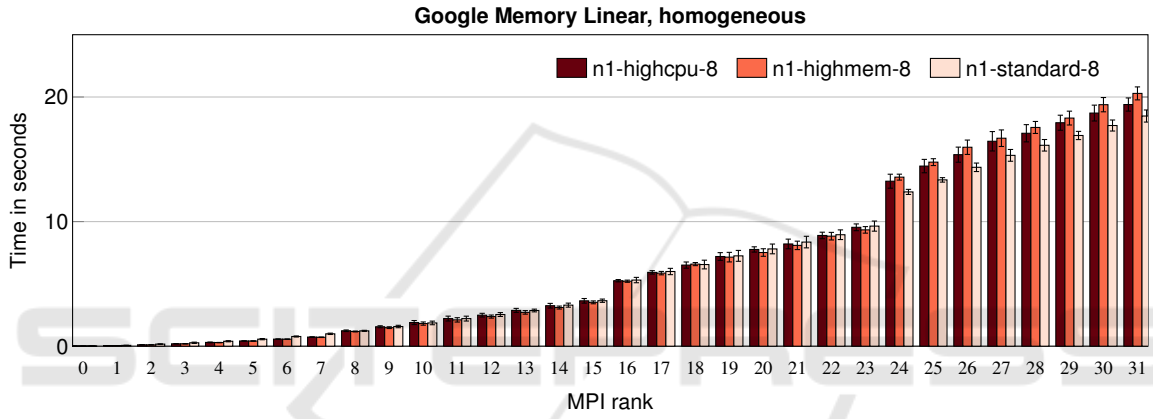

Figure 12: Results for the Linear pattern and Memory performance for the homogeneous Google instances cluster.

When we used another pattern of ImbBench, the 8Level pattern, another situation occurs. Figure 13 shows the results of this execution. The performance of all the eight levels were similar in both instances. But, with a closer look, we could observe that the first four processes, the processes with less load, of the application presented worst performance in the c5.2xlarge instance. And the last four processes, the ones with highest load, performed better in c5.2xlarge than in the t2.2xlarge.

3.4.2 Memory

Related to memory performance, the AWS instances were similar to the CPU tests. Again the c5.2xlarge and t2.2xlarge instances were chosen to perform the evaluation. Figure 15 shows the results of the memory performance test using the Linear pattern from ImbBench. As we can observe, the memory performance of both instance types is identical, meaning that there is no heterogeneity to be exploited.

The AWS instances are well balanced, as the Google instances, and there is not opportunities to exploit the heterogeneity. The instance c5.2xlarge is one of the cheapest of the provider and it presents the best performance as well. According to our tests, this instance performed better in all patterns, with exception of the Balanced pattern, where the t2.2xlarge instance performed better. Due to this situation, the cost efficiency of the AWS will be better by using homogeneous clusters made of c5.2xlarge.

3.5 Discussion

Table 3 shows the overall performance and cost efficiency results for all experiments. In terms of the performance across all the providers, Microsoft Azure presented the instance with best performance, the H8 instance, that performed better both in terms of processing power and in memory performance. The AWS c5.2xlarge instance presented slightly less performance than the H8 one. Google got the third place with its three instances. However in terms of price, Google's n1-highcpu-8 instance has the best price,

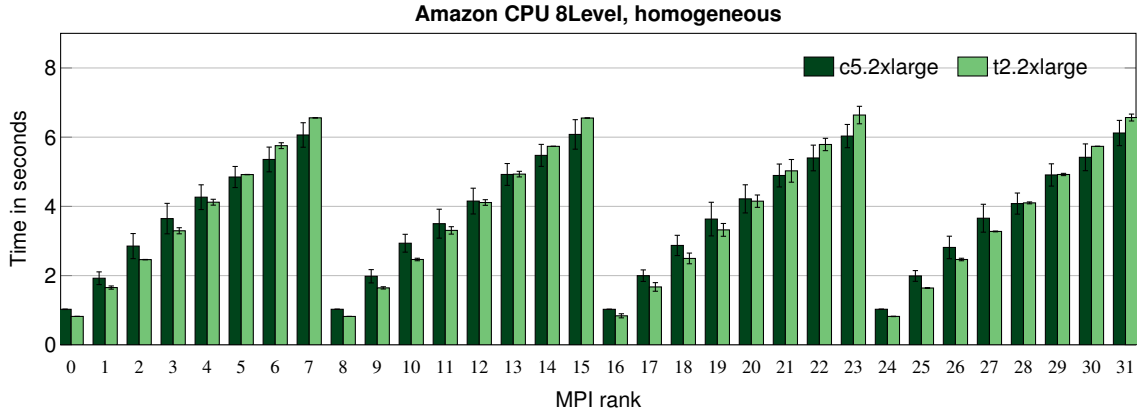

Figure 13: Results for the 8Level pattern and CPU performance for the homogeneous Amazon instances cluster.

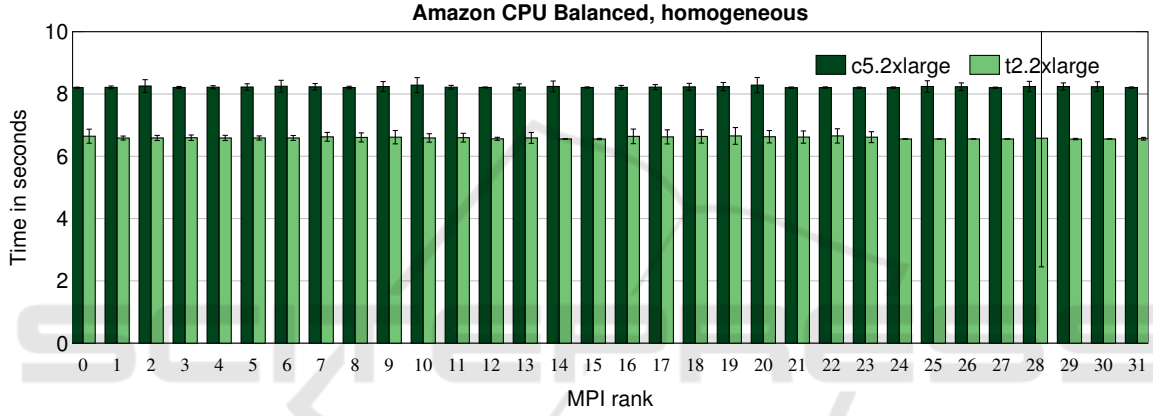

Figure 14: Results for the Balanced pattern and CPU performance for the homogeneous Amazon instances cluster.

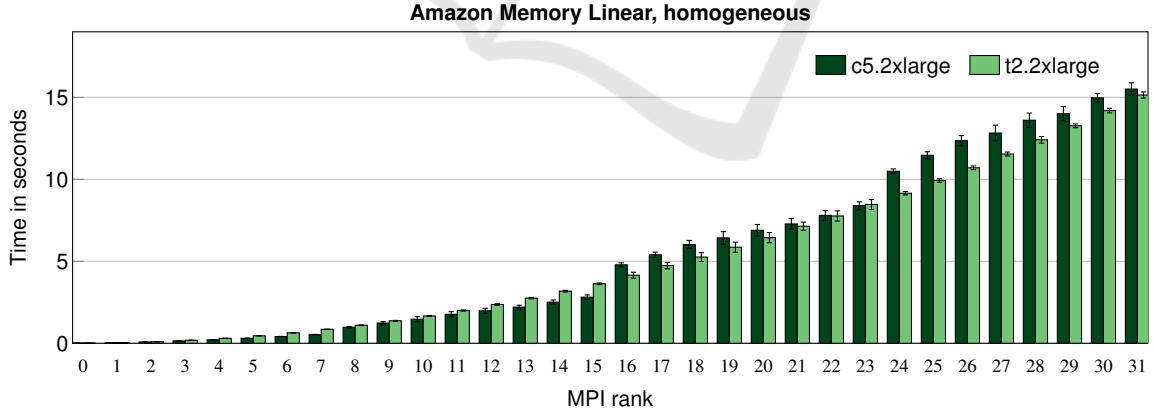

Figure 15: Results for the Linear pattern and Memory performance for the homogeneous Amazon instances cluster.

followed by AWS and Azure instance has the higher price.

Several interesting conclusions can be drawn from the results presented in this section. First, the Azure cloud presents a beneficial case for heterogeneous clouds, since cost efficiency improves dramatically over a homogeneous allocation with an imbalanced application. On the other hand, Google's cloud instances consists of machines of the same CPU and memory speed, varying only in memory size, and can therefore benefit less from application heterogeneity. Amazon, in contrast to the other two providers, shows the interesting situation where the cheapest machine is also the fastest, which limits its suitability for het-

Table 3: Cost efficiency characteristics of the cloud instances. Performance is shown in seconds. Higher values for cost efficiency are better.

Configuration	CPU		MEM	
	Cost eff.	Perf.	Cost eff.	Perf.
8Level				
Azure H8	175	5.27	75	12.22
Azure Het.	288	5.26	129	12.22
Google n1-standard-8	290	8.15	121	19.53
Google n1-highmem-8	230	8.25	114	16.62
Google n1-highcpu-8	386	8.21	193	16.43
Amazon c5.2xlarge	432	6.12	194	13.58
Amazon t2.2xlarge	365	6.63	158	15.32
Amdahl				
Azure H8	175	5.28	75	12.62
Azure Het.	315	5.25	138	12.14
Google n1-standard-8	301	7.85	139	16.97
Google n1-highmem-8	245	7.73	128	13.68
Google n1-highcpu-8	408	7.76	228	13.84
Amazon c5.2xlarge	455	5.81	236	11.17
Amazon t2.2xlarge	369	6.55	159	15.22
Linear				
Azure H8	176	5.26	73	12.14
Azure Het.	288	5.26	126	12.51
Google n1-standard-8	238	9.93	128	18.47
Google n1-highmem-8	194	9.79	93	20.29
Google n1-highcpu-8	317	9.98	163	19.40
Amazon c5.2xlarge	343	7.71	170	15.50
Amazon t2.2xlarge	368	6.58	160	15.14

erogeneous execution. However, both Google and Amazon might benefit more from heterogeneity in the future when the instance types they offer change. Furthermore, the suitability of heterogeneous execution is similar between the CPU and memory-bound versions of ImbBench. The results shown in the paper can therefore also be seen as a motivation for providers to offer instance types with more diverse characteristics, as these can be used to improve cost efficiency of executing applications with uneven computational demands in the cloud.

4 RELATED WORK

Prabhakaran and Lakshmi (Prabhakaran and Lakshmi, 2018) evaluate the cost-benefit of using Amazon, Google and Microsoft cloud instances instead of a supercomputing. They found that the cost-benefit of the supercomputer was better than the cloud. However they consider a 7x24 hours usage, and do not consider the idle time, which affects the cost calculations. Kotas et al. (Kotas et al., 2018) compared AWS and Azure as a platform for HPC. The authors executed the HPCC and HPCG benchmarks suite. However they evaluate one instance type of each provider and build cluster configurations from 1 to 32 nodes.

Aljamal et al. (Aljamal et al., 2018) compare Azure, AWS, Google and Oracle Cloud as a platform for HPC. They made a comparative analysis of the providers configurations and offers, but did not perform any performance or cost efficiency experiments. (Li et al., 2018a) performed an analysis of how to benefit from the cloud heterogeneity to perform video trans-coding, in order to maximize the efficiency for video streaming services. (Li et al., 2018b) created a cost and energy aware scheduler to reduce the cost and energy consumption of executing scientific work flows in the cloud. They evaluated their proposal by using the CloudSim simulator.

Costa el al. (G. S. Costa et al., 2018) performed a performance and cost analysis comparing the on-demand versus preemptive instances of the AWS and Google cloud providers. However they used only profiling of the instances and do not evaluate benchmarking or applications. In our previous works (Roloff et al., 2017; Roloff et al., 2018), we exploited the heterogeneity of public cloud and application demand. This work is an improvement of the previous work, by performing a deep analysis of the instances performing memory and CPU characterizations and a comprehensive analysis of different public clouds.

5 CONCLUSIONS AND FUTURE WORK

Executing HPC applications in cloud environments has become an important research topic in recent years. In contrast to traditional clusters, which consist of mostly homogeneous machines, executing HPC applications in the cloud allows them to use heterogeneous resources offered by the cloud provider. This way, the cloud instances can be adapted to an imbalanced behavior of the HPC application, reducing the price of executing it in the cloud.

In this paper, we performed a comprehensive set of experiments evaluating the potential of using heterogeneous instance types on three public cloud providers, Microsoft Azure, Amazon AWS, and Google Cloud. Using ImbBench, a benchmark that can create different imbalance patterns of CPU and memory usage, we compared performance and cost efficiency on homogeneous and heterogeneous cloud instances. Our results show that heterogeneous execution is most beneficial on Azure, improving cost efficiency by up to 50% compared to execution on homogeneous instances, while maintaining the same performance. The other two providers are less suitable,

since the cheapest instance type is also the fastest (AWS), or the provider offers only instances that vary in memory size, but not in performance (Google).

As future work, we intend to extend our ImbBench benchmark with support for I/O operations and network communication to evaluate these aspects in terms of heterogeneity. We also intend to add support for combining different types of operations to be more representative of real-world applications. Furthermore, we will provide an automatic way to suggest a mix of cloud instances for a particular application behavior.

ACKNOWLEDGEMENTS

This work has been partially supported by the project "GREEN-CLOUD: Computacao em Cloud com Computacao Sustentavel" (#16/2551-0000 488-9), from FAPERGS and CNPq Brazil. This research received partial funding from CYTED for the RICAP Project.

REFERENCES

Abdennadher, N. and Belgacem, M. B. (2015). A high level framework to develop and run e-science applications on cloud infrastructures. In *High Performance Computing and Communications (HPCC)*.

Aljamal, R., El-Mousa, A., and Jubair, F. (2018). A comparative review of high-performance computing major cloud service providers. *2018 9th International Conference on Information and Communication Systems, ICICS 2018*, 2018-Janua:181–186.

Armbrust, M., Fox, A., Griffith, R., Joseph, A. D., Katz, R. H., Konwinski, A., Lee, G., Patterson, D. A., Rabkin, A., and Zaharia, M. (2009). Above the clouds: A berkeley view of cloud computing. Technical report.

Buyya, R., Yeo, C. S., Venugopal, S., Broberg, J., and Brandic, I. (2009). Cloud computing and emerging it platforms: Vision, hype, and reality for delivering computing as the 5th utility. *Future Generation Computer Systems*, 25(6):599 – 616.

d. R. Righi, R., Rodrigues, V. F., da Costa, C. A., Galante, G., de Bona, L. C. E., and Ferreto, T. (2016). Autoelastic: Automatic resource elasticity for high performance applications in the cloud. *IEEE Transactions on Cloud Computing*, 4(1):6–19.

de Melo, A. C. (2010). The New Linux 'perf' Tools. In *Linux Kongress*.

Dongarra, J. J., Luszczek, P., and Petitet, A. (2003). The linpack benchmark: past, present and future. *Concurrency and Computation: practice and experience*, 15(9):803–820.

Foster, I., Zhao, Y., Raicu, I., and Lu, S. (2008). Cloud computing and grid computing 360-degree compared. In *2008 Grid Computing Environments Workshop*, pages 1–10.

G. S. Costa, B., Reis, M. A. S., P. F. Araújo, A., and Solis, P. (2018). Performance and cost analysis between on-demand and preemptive virtual machines. In *Proceedings of the 8th International Conference on Cloud Computing and Services Science - Volume 1: CLOSER,*, pages 169–178. INSTICC, SciTePress.

Kotas, C., Naughton, T., and Imam, N. (2018). A comparison of Amazon Web Services and Microsoft Azure cloud platforms for high performance computing. *2018 IEEE International Conference on Consumer Electronics, ICCE 2018*, 2018-Janua:1–4.

Li, X., Amini Salehi, M., Joshi, Y., Darwich, M., Bayoumi, M., and Landreneau, B. (2018a). Performance analysis and modeling of video transcoding using heterogeneous cloud services. *IEEE Transactions on Parallel and Distributed Systems*, pages 1–1.

Li, Z., Ge, J., Hu, H., Song, W., Hu, H., and Luo, B. (2018b). Cost and energy aware scheduling algorithm for scientific workflows with deadline constraint in clouds. *IEEE Transactions on Services Computing*, 11(4):713–726.

McCalpin, J. D. et al. (1995). Memory bandwidth and machine balance in current high performance computers. *IEEE computer society technical committee on computer architecture (TCCA) newsletter*, 1995:19–25.

Prabhakaran, A. and Lakshmi, J. (2018). Cost-benefit Analysis of Public Clouds for offloading in-house HPC Jobs. *2018 IEEE 11th International Conference on Cloud Computing (CLOUD)*, pages 57–64.

Roloff, E., Diener, M., Carissimi, A., and Navaux, P. O. A. (2012). High performance computing in the cloud: Deployment, performance and cost efficiency. In *4th IEEE International Conference on Cloud Computing Technology and Science Proceedings*, pages 371–378.

Roloff, E., Diener, M., Carreño, E. D., Gaspary, L. P., and Navaux, P. O. A. (2017). Leveraging cloud heterogeneity for cost-efficient execution of parallel applications. In *Euro-Par 2017: Parallel Processing - 23rd International Conference on Parallel and Distributed Computing, Santiago de Compostela, Spain, August 28 - September 1, 2017, Proceedings*, pages 399–411.

Roloff, E., Diener, M., Gaspary, L. P., and Navaux, P. O. A. (2018). Exploiting load imbalance patterns for heterogeneous cloud computing platforms. In *Proceedings of the 8th International Conference on Cloud Computing and Services Science - Volume 1: CLOSER,*, pages 248–259. INSTICC, SciTePress.

Zhang, J., Lu, X., Arnold, M., and Panda, D. K. (2015). Mvapich2 over openstack with sr-iov: An efficient approach to build hpc clouds. In *Cluster, Cloud and Grid Computing (CCGrid), 2015 15th IEEE/ACM International Symposium on*, pages 71–80.