3-6-2014 12:00 AM

A Vector-Based Approach to Virtual Machine Arrangement

Nicholas Cerilli
The University of Western Ontario

Supervisor
Dr. Hanan Lutfiyya
The University of Western Ontario

Graduate Program in Computer Science
A thesis submitted in partial fulfillment of the requirements for the degree in Master of Science

A VECTOR-BASED APPROACH TO VIRTUAL MACHINE ARRANGEMENT

(Thesis format: Monograph)

by

Nicholas Cerilli

Graduate Program in Computer Science

A thesis submitted in partial fulfillment
of the requirements for the degree of
Master of Science

The School of Graduate and Postdoctoral Studies
The University of Western Ontario
London, Ontario, Canada

Abstract

Cloud based data centres benefit from minimizing operating costs and service level agreement violations. Vector-based data centre management policies have been shown to assist with these goals. Vector-based data centre management policies arrange virtual machines in a data centre to minimize the number of hosts being used which translates to greater power efficiency and reduced costs for the data centre overall. I propose an improved vector-based virtual machine arrangement algorithm with two novel additions, namely a technique that changes what it means for a host to be balanced and a concept that excludes undesirable target hosts, thereby improving the arrangement process. Experiments conducted with a simulated data centre demonstrate the effectiveness of this algorithm and compares it to existing algorithms.

Keywords

Virtualization, Vector, Virtual Machine, Arrangement, Data Centre Simulation, Policy, Multiple Resource Management, Power Efficiency, Placement

Acknowledgments

I would like to acknowledge the assistance of Dr. Hanan Lutfiyya for her supervisorial role in the creation of this thesis. Additionally, I would like to acknowledge and thank Gastón Keller and Michael Tighe for allowing the use of their simulator to complete the experiments contained within this thesis.

Table of Contents

List of Tables

List of Figures

List of Appendices

Chapter 1

1 Introduction and Problem Identification

Cloud infrastructures may consist of one or more data centres. These data centres consist of large amounts of computing resources e.g., storage space, memory, and processing power. A possible business model for a cloud infrastructure provider is to rent computing resources to clients that wish to have their applications executed without having to incur the costs associated with buying and maintaining the hardware needed to execute the applications [1]. Such a business model is referred to as *infrastructure as a service*. Infrastructure as a service has many benefits that make it an appealing option for client businesses. IT systems are able to be obviated and outsourced when one opts to utilize the infrastructure as a service industry. With this streamlining of IT, clients will no longer have to incur costs associated with hardware acquisition, testing, maintenance, and staffing devoted solely to the aforementioned processes. Instead, a client pays a subscription fee associated with utilizing the infrastructure present within the cloud [1]. Security, testing, maintenance, uptime, and other requirements now become the responsibility of the cloud provider. Furthermore, the diverse range of hardware present within the cloud makes it possible to run different types of applications. From web servers and email servers to databases, a client company may request the execution of a variety of applications without having to worry about wildly disparate hardware requirements as, once again, the responsibility of hardware procurement falls to the cloud provider.

One of the challenges that a cloud provider has is using the resources within the cloud as efficiently as possible. This is accomplished in part by hosting a client's application in a *virtual machine*. Virtual machines consist of software that encapsulates a client's application and provides all of the operating system and hardware requirements that would normally be provided by a physical machine [1]. A virtual machine allows computing resources to be allocated to it that are a fraction of the computing resources available on the physical machine. This allows multiple applications to run on the same physical machine. A physical machine that is underutilized can have additional virtual machines placed on it. If the demand for an application increases then additional resources can be allocated. If a physical machine is overloaded it is possible to suspend a virtual machine, move it to another physical machine and restart from the state that the virtual machine was suspended in. This is referred to as *migration*. Effectively using cloud resources using virtual machines has been investigated e.g., [2] [3] [4] [5].

1.1 Data Centre Managers

Data centre managers provide the mechanism through which decisions can be made and virtual machines can be reorganized within a data centre to utilize resources more efficiently. The effectiveness of the data centre manager is directly tied to the effectiveness of the policies it implements to determine arrangements of virtual machines on the physical hosts [6]. It should be noted that in much of the literature, arrangements of virtual machines are simply referred to as placements. However, to avoid ambiguity with the initial placement of virtual machines, the locations of the virtual machines within a data centre at any point will be referred to as an arrangement.

1.2 Resource Management

Effective utilization of cloud resources requires the allocation of resources to virtual machines that satisfies the run-time requirements of the application running in the virtual machine. The static approach to resource allocation for an application assigns the maximum amount of resources needed by the application. This approach ensures that an application’s resource demands are met as long as the resource requirements are accurately calculated. This represents an overcommittment of resources [2] [3] [7] [8]. This strategy may result in an underutilization of resources [9] [5] [10] [4]. For example, consider a particular application that had a lifetime of 100 hours and for 90 of those hours only requires 10 units of resource A. For the remaining 10 hours, the application requires 500 units of resource A. With static allocation, 500 units of resource A would be allotted to the application for the entire 100 hours of the application’s execution time. Clearly the majority of resource A could be better used in the execution of some other application.

Static allocation works best when the resource requirements of an application are not highly variable. Another approach allows for the oversubscription of a host's resources [11]. This assumes that applications have highly variable demands and that the set of applications (and hence VMs) varies over time. Dynamic resource management takes advantage of migrating a VM from a physical machine to a host machine that is not overloaded. Migration results in a new arrangement.

1.3 Virtual Machine Arrangement and Bin Packing

The data centre manager selects the physical host to place each virtual machine. This is a non-trivial task since the resource requirements of each virtual machine and the availability of resources of host machines must be considered [12] [13] [14]. Determining an arrangement when only one resource is considered is analogous to the one dimensional bin packing problem, which has been shown to be NP-hard [15]. However determining an arrangement when multiple resources are to be considered is not the same as the multidimensional bin packing problem, and thus existing methods for the multidimensional bin packing problem do not apply [16] [17].

Essentially virtual machine arrangement is a problem that with one resource is analogous to the one dimensional bin packing problem, but with multiple resources it is more complex than the multidimensional bin packing problem. The rest of this section discusses this in more detail.

Virtual machine arrangement design is analogous to the bin packing problem where only one resource is considered. In the bin packing problem, items of varying weight are to be placed in bins. Each of the bins has a maximum weight. The objective is to achieve an arrangement where the minimum number of bins is used without exceeding the maximum weight of any one bin. Virtual machine arrangement design mirrors this task as the objectives are quite similar and approaches to solving the bin packing problem can be utilized to great effectiveness [18]. For example, a data centre may take only one resource under consideration, CPU usage. In this situation, the virtual machines represent the items. The hosts are akin to bins. The CPU requirements of each virtual machine are the weight and the maximum CPU capabilities on a given host represent the maximum weight each bin may hold. The objective would be to satisfy all of the CPU needs for each of the applications while using the least number of physical hosts and maximizing the satisfaction of service level agreements. By minimizing the number of hosts, the data centre avoids underutilization and operates at increased efficiency [18]. Thus, power efficiency is maximized and the operating costs associated with powering a data centre are minimized [4] [19]. A variety of algorithms exist for approximating solutions to the bin packing problem. These algorithms can also be used by a data centre manager.

Although the basic bin packing problem and its approaches are suitable for utilization in a data centre management policy where only one type of resource is considered, the similarities do not extend into higher dimensions and multiple resources [20] [16]. Multidimensional bin packing algorithms consider each dimension to be like an edge on an n-dimensional object. For example, in two dimensions length and width are considered in calculations. If items are thought to be represented by rectangles, a valid arrangement would be one where the rectangles are placed beside each other. This would result in reducing the “amount of width” remaining in a bin but the “amount of length” taken up would be the same as if only one rectangle were placed in a bin. In a virtual machine arrangement, virtual machines cannot be placed “beside” each other. Every additional virtual machine placed in a host must subtract some of the available resources from the host’s total across every dimension. Figure 1 demonstrates the only valid arrangement for two virtual machines in a host where two resources, RAM and CPU, are considered. The axes represent the amount of each respective resource that the host may allocate to the virtual machines. The lightly coloured rectangles represent, through the lengths and widths, the amount of each respective resource the virtual machines require. The diagonal black lines represent areas that cannot be occupied by virtual machines because that would imply the arrangement of a virtual machine with insufficient resources being deducted from the host’s total.

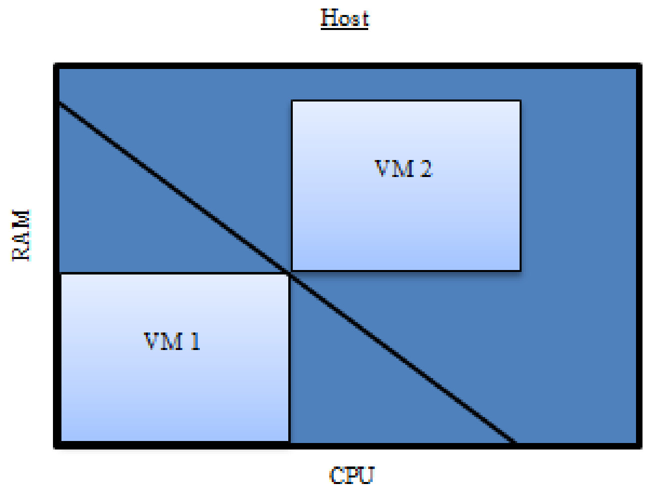

Figure 1 – Valid VM Arrangement

Figure 2 illustrates a virtual machine arrangement that may be furnished by a solution that appropriates techniques from a two dimensional bin packing methodology. Nevertheless, such an arrangement is invalid for use in the virtual machine arrangement problem. This diagrammatic explanation was independently developed by this researcher however similar explanations, including similar diagrams, can be found in the literature, in particular [21]. It is because of the fact that a valid bin packing arrangement does not necessarily correspond to a valid virtual machine arrangement that the multiple resource virtual machine arrangement problem is not analogous to multidimensional bin packing. Consequently, the algorithms and solutions developed for that problem domain cannot be utilized for a data centre's management policy, at least not without heavy modifications and additions.

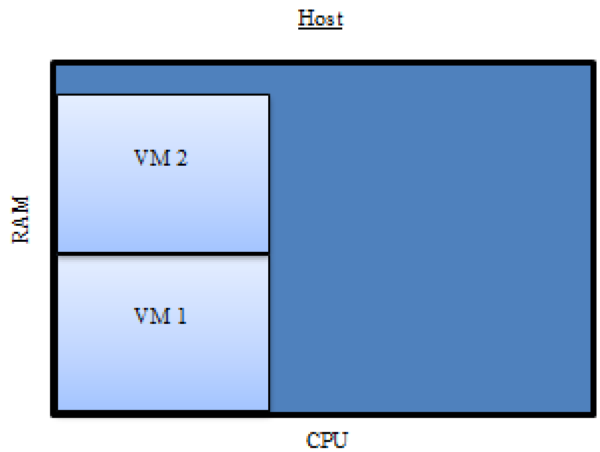

Figure 2 – Invalid VM Arrangement

1.4 Thesis Focus

The focus of this thesis is to examine a novel technique for virtual machine arrangement that considers two resources. A valid technique will be one that allows all of the resource requirements of a series of applications to be met while minimizing service level agreements violations [2]. First, in chapter 2, a series of existing techniques for virtual machine arrangement will be enumerated. Next, in chapter 3, a novel virtual machine arrangement technique will be presented, with a special focus on the concepts and techniques that make the process as a whole novel. In chapter 4, a series of experiments will be presented where the results support the claims that the novel aspects of the new technique are effective when compared to existing, similar techniques and that the new technique overall may perform similarly when compared to said existing techniques. Lastly, in chapter 5 further improvements upon the novel algorithm and suggestions for future work will be discussed as well as any conclusions that can be drawn from the aforementioned experiments.

Chapter 2

2 Related Works

When solving the problem of creating suitable virtual machine arrangements within a data centre, one may approach the problem in different ways and with different techniques. First, the data centre manager must be designed to create either static or dynamic virtual machine arrangements within a data centre. Static arrangements are those that seek to place each virtual machine in the data centre once for the duration of its execution. This is known as performing an initial placement of a virtual machine. Typically, information pertaining to the application's resource requirements is known beforehand and arrangements are constructed with an application's peak resource requirements in mind. This is demonstrated when one utilizes the practice of overcommitting resources [22]. In contrast, dynamic arrangements are those that have the additional ability to migrate virtual machines to other physical hosts should the need arise for a virtual machine to be given more resources than are available on its current physical host [7]. These migrations can occur in the form of virtual machine relocation and consolidation. The method by which a suitable arrangement is calculated can utilize a variety of techniques, e.g. forecasting, genetic algorithms, greedy algorithms, in order to construct valid static and dynamic virtual machine arrangements within a data centre [18] [10] [23].

2.1 Static and Dynamic Resource Management

One of the key considerations for any data centre manager is whether the resource requirements of the applications on the virtual machines are to be considered static or dynamic. Resource requirements for applications generally fluctuate [7] [24]. If one wishes to treat the resource requirements of the applications as static regardless, certain assumptions must be made. Generally this consists of placing virtual machines such that their peak resource requirements are met. This can be very wasteful as a virtual machine's peak resource requirements may only be necessary for a relatively short period of time given the overall execution time of the application [7] [8]. Consequently, dynamic resource management techniques are seen as an overall improvement in the way data centre managers construct their arrangements [25] [24]. When constructing an arrangement to be used by the data centre with dynamic resource management, three operations are available. First, the data centre manager performs a placement operation when the virtual machine is first created. This placement operation is necessary for data centre managers that handle both static and dynamic resources. Metrics for resource requirements are used to place the newly created virtual machine based on information pertaining to each virtual machine's resource usage. This information could be the virtual machine's peak resource usage (as is the case with static arrangements), average resource usage, or typical resource usage. This information is then used as input for an arrangement technique, such as forecasting, integer linear programming, genetic algorithms, or a greedy technique. The output is a valid virtual machine arrangement within the data centre. The second and third operations available to a data centre manager used for dynamic resource management are relocation and consolidation. Virtual machine relocation occurs when the data centre manager has determined that a more efficient virtual machine arrangement can be attained by moving existing virtual machines from one host to another [3]. This is a result of the resource requirements of one or more virtual machines on a host increasing above and beyond the point at which it was when said machines were placed on the hosts, thereby causing the host to enter a stressed state. Typically, a poll of the hosts within a data centre or event driven programming is used to discern when a host enters such a state and relies on existing definitions of what it means for a host to be stressed, partially utilized, and underutilized. Such definitions are assumed to have been supplied to the data centre manager a priori. Finally, virtual machine consolidation occurs when the data centre manager has determined that a more efficient virtual machine arrangement can be attained by vacating all of the virtual machines from an underutilized host, moving said virtual machines to new hosts, and powering off the original host [3]. This is a result of the resource requirements of one or more virtual machines on a host decreasing below the point at which it was when said machines were placed on the hosts (including the possibility of one or more hosts completing their execution), thereby causing the host to enter an underutilized state. Once again, a poll of the hosts within a data centre or event driven programming is used to discern when a host enters such a state and this process relies on existing definitions of what it means for a host to be stressed, partially utilized, and underutilized. The data centre manager then compensates by once again computing a valid data centre arrangement, often by using the same technique that was used for placement, and migrating the relevant virtual machines in order to implement the new arrangement. The ability to dynamically compensate for the fluctuations in the resource requirements of the virtual machines residing on the many hosts within the data centre allows for more efficient use of the data centre's resource complements and as such has been found to be a superior approach when compared to static arrangements [7] [8] [25]. In summary, dynamic relocation and consolidation of virtual machines in response to dynamic resource requirements allows for a more efficient usage of the data centre as a whole.

2.2 Virtual Machine Arrangement Techniques

There exist several broadly defined techniques for virtual machine arrangement. Each technique uses a certain unique concept to decide on which hosts each virtual machine should be placed, relocated, or consolidated within a data centre. Each of these techniques may be used in either a static or dynamic policy. To implement a technique for a static policy, only the initial placement of a virtual machine need be considered. In order to implement a dynamic policy, relocation and consolidation of virtual machines must be considered as well. One such method is called forecasting and it refers to a broad range of techniques where previous arrangements are examined and trends are used to predict appropriate arrangements in the future [2] [26]. Also, techniques that involve the use of genetic algorithms may also be used to define valid virtual machine arrangements [10]. Furthermore, integer linear programming may be incorporated into data centre management policies [26]. Lastly, greedy algorithms have been utilized to provide valid virtual machine arrangements for data centre management policies [18]. Forecasting, genetic algorithms, integer linear programming, and greedy algorithms may all be used to construct valid virtual machine arrangements within a given data centre.

2.2.1 Techniques Involving Forecasting

Forecasting is a method for establishing virtual machine arrangement. Forecasting, as the name suggests, attempts to predict which suitable virtual machine arrangement will be most effective in the future [27] [28] [29]. This is accomplished by examining past and current arrangements and attempting to discern which initial conditions resulted in said arrangements [28]. Once the initial conditions are identified, they are catalogued and stored. From then on, software that is specifically designed to examine the state of the data centre records statistics pertaining to the state. If at any time the state of the data centre matches, to some degree, one of the recorded states that the data centre has already encountered, the data centre manager is notified. The data centre manager will then place, relocate, and consolidate virtual machines as necessary to either match previously successful arrangements or avoid unsuccessful arrangements [23]. As mentioned previously, the success or failure of a given arrangement can be measured via metrics that take into account the number of hosts needed, the overall power consumption, and the ability of the data centre to adhere to service level agreements. Forecasting can most definitely be an effective tool in solving the virtual machine arrangement problem [27] [28].

Forecasting based data centre management policies have varying success depending on the workload. The ideal workload would be one in which the resource usages are periodic or at least have some element of repetition to their traces. The reason behind this being that these repetitions basically train the forecasting software such that it is better able to recognize trends in the workload [23]. Additionally, repetitive workloads give the data centre manager the opportunity to compare slightly different arrangements stemming from the same initial conditions [23]. The logic being that the more candidate arrangements a data centre manager has to choose from, the better chance the manager will have of selecting a successful arrangement. Periodic workloads can occur as a result of external periodic factors. For example, consider an application that runs 24 hours a day, but experiences its heaviest workload during the workday. A forecasting based data centre manager might take this into account by storing one arrangement for the hours during the workday and a second arrangement for any other time. Conversely, workload traces that are random, have difficult to discern patterns, or are simply too short for the software to properly examine will surely cause problems for a forecasting based data centre manager. For example, data centres that execute a variety of applications ranging from CPU intensive HTTP servers to RAM intensive database management systems might not ever encounter similar states twice. Consequently, there will be no opportunity to utilize previously implemented arrangements and the efficiency of the data centre could possibly degrade [28]. All in all a forecast based data centre management strategy works best when the workload traces are such that similar states are often repeated.

2.2.2 Techniques Involving Genetic Algorithms

Another technique for creating virtual machine arrangements is through the use of genetic algorithms [30]. Genetic algorithms replicate the natural phenomenon of survival of the fittest and apply it to complex problems [30]. Genetic algorithms have been shown to be successful in solving said complex problems by trying many solutions, combining those that were successful to make new combinations, and discarding those that underperformed [10] [31]. One such area of success for genetic algorithms is path finding algorithms. It is not incomprehensible that one might expect genetic algorithms to produce desirable results for the virtual machine arrangement algorithm. The task is relatively straightforward. A pool of candidate arrangements is generated. The effectiveness of these arrangements are then rated using some metric [31]. For example, the number of hosts any given arrangement required would be said to be inversely related to its effectiveness. Then, a subset of the arrangements would be selected to move on to the next generation and the rest would be discarded. Finally, those arrangements that made it to the next generation would have their arrangements divided in some way, and crossed over with other candidate solutions [30]. The hope is that through enough generations, the arrangements will only pass on qualities that were successful, thereby resulting in a near optimal arrangement at the end of the evolutionary process. Additionally, random mutations could be included into every generation to account for arrangements that were not present in the original candidates. In theory, it is reasonable to expect genetic algorithms to be able to provide suitable solutions to the virtual machine arrangement problem.

In practice, the use of genetic algorithms did not provide solutions to the virtual machine arrangement problem that sufficiently outperformed other techniques. There are many variables that can be altered when constructing a genetic algorithm. The size of the initial pool of candidate solutions, the point of crossover, the rate of mutation, and the number of generations can all be altered to provide different results and, consequently, different levels of effectiveness when implementing a genetic algorithm. Nevertheless, the particular genetic algorithms that have been constructed to solve the virtual machine arrangement problem have not outperformed other methods [10]. Although the use of genetic algorithms has proven to be successful in the arenas of complex combinatorial problems, the virtual machine arrangement problem has so far left them performing poorer when compared to other load balancing techniques and forecasting techniques [10]. It should be noted that genetic algorithms have been shown to outperform simple greedy algorithms [10]. This may be because an ideal arrangement changes as virtual machines enter and leave the data centre. As mentioned before, data centres can host virtual machines that have dynamic resource requirements and perhaps this concept is difficult to integrate into a genetic algorithm. Furthermore, unlike a path finding algorithm, there is no logical or intuitive was to establish a crossover point for an arrangement. The success of any given arrangement of a virtual machine is inherently dependent on the arrangements that came before it. Contrast this with path finding algorithms where a movement towards the end goal is always considered improvement, regardless of other movements that occurred in the grand scheme of the path. In summary, despite the proclivity for genetic algorithms to solve complex combinatorial problems, they have shown to be suboptimal with respect to the virtual machine arrangement problem.

2.2.3 Techniques Involving Integer Linear Programming

Yet another method to solving the virtual machine arrangement problem encompasses techniques that utilize integer linear programming. Integer linear programming refers to solving a problem where some or all of the variables are restricted to integers [15] [30] [31]. Additionally, the constraints on the variables are linear. That is, there are no restrictions on the functions that are equal to or of higher order than a quadratic. In the context of the virtual machine arrangement problem, the number of hosts must obviously be an integer value. Furthermore, the resource values used when placing the virtual machines can also be expressed as integers [30]. The problem of placing virtual machines suggests utilizing integer linear programming [31]. In conjunction with limiting the values of the problem domain to integers, said techniques incorporate a brute force component where every combination of virtual machine arrangements is explored, to a certain depth [31]. Such an approach can be best described in the context of a decision tree with every node representing a different combination of arrangements [15]. The logic behind this technique is that if the data centre manager tries every possible combination for the current arrangement as well as a certain number of anticipated future arrangements, the data centre manager can make the decision as to which course of actions would be the best to follow [31]. The technique is not unlike those implemented by chess playing computers. That is, the objective is to reach a state with a certain optimal value but there are multiple ways in which to proceed. Every possible path is then computed and ranked by some metric. Then the first step in the best performing path is taken. The process is repeated at each step.

Integer linear programming techniques are prohibitively expensive when applied to the virtual machine arrangement problem. As mentioned before, there is a brute force element to this approach where every combination of a subset of the hosts and virtual machines are explored. In even a small data centre this can lead to the problem of state space explosion. That is, even if the number of hosts is on the order of 100, the total number of possible orderings of these hosts would take too much time to explore. As a result, this technique is restricted in its applications [15]. However, this technique can be shown to provide better orderings as every single possible combination is explored [31]. Additionally, efforts have been made to reduce the time complexity of such techniques by utilizing branch and bound mechanisms. That is to say, paths are checked quite early in their traversal to find if they will result in an optimal ordering and if they are determined to result in a less than optimal ordering, they are excluded from further investigation [15]. This is known as pruning. Nevertheless, the time complexity of implementing such techniques is exponential and is not suitable for all applications. In summary, integer linear programming based approaches allow for the discovery of optimal arrangements, but are useless in some applications due to their prohibitive time complexity.

2.2.4 Techniques Involving Greedy Algorithms

In addition to the previously discussed methods, the virtual machine arrangement problem can be effectively solved using greedy algorithms. In contrast to forecasting or integer linear programming which both try to anticipate the future needs of the data centre, greedy algorithms choose the best possible choice given the immediate situation. It is the hope that by repeatedly choosing the local optimal solution, a global optimal solution will be the end result [18] [21]. Furthermore, the task of constructing virtual machine arrangements is difficult when one considers a single resource, let alone multiple resources. Greedy algorithms provide a way to combine multiple resource values into a single criterion so that they may factor into the creation of a virtual machine arrangement. The key differentiator between greedy algorithms then becomes the criteria with which the local solution is chosen. In the virtual machine arrangement problem there exist multiple ways to rank the hosts and virtual machines if only one resource is considered. A ranking system with such stipulations would simply consist of two parts. The first part would be a metric with which to rank the hosts, usually the resource value under scrutiny and the second part would simply be whether the ordering was increasing or decreasing. One such ranking system is “first fit decreasing” where the hosts are associated with a scalar value based on some metric and then organized from highest to lowest. This ordering has been shown to be effective however alternate orderings exist such as “first fit increasing” where the order of the hosts is reversed from the aforementioned method, as well as methods that divide the target hosts into subsets based on their utilization [18]. When only one resource is under consideration, the scalar value is simply the raw value for whatever resource was chosen. For example, if CPU usage is the only resource to be factored into the arrangement, each host may be ranked according to its percentage of CPU resource currently being used. There have been studies on the effectiveness of ranking hosts in such a way, namely [18]; however the purpose of this thesis is to examine more than one resource requirement and integrate that information into a data centre management policy so such methods will be mentioned only briefly.

Consequently, the matter of how to determine a scalar metric when more than one resource is to be considered becomes an issue. That is to say, there a several ways to combine multiple resource values. Some examples include, summing the values, computing the product, calculating the dot product, and finding a ratio between values [15]. Of particular interest to this thesis are specific summation, product, and ratio methods.

2.2.4.1 Greedy Metric Type 1 – Summation and Product Methods

One way to combine resource values is to simply sum them together. CPU utilization, RAM requirements, and bandwidth usage are common data centre attributes used when deciding how to place virtual machines [30]. For example, if the resources one wishes to consider are the CPU utilization, the RAM requirements, and the bandwidth usage, these three raw values for a given host may simply be summed together. It is up to the data centre management policy designer to decide whether the resource values should be raw values, percentages, or weighted values [15]. A possible equation for use in a summation based management strategy might be as follows: $value(host) = \sum_{r \in Resource} \alpha r$. This equation was inspired by the one discussed in [15]. The resource values of every resource under considerations are simply summed together. The value of alpha can be altered to reflect a weight if one resource should be considered more heavily in the ordering process. A similar method to the summation method is one where the resource values are multiplied together rather than having their sum calculated. Once again, it is up to the data centre management policy designer to decide whether the resource values should be raw values, percentages, or weighted values [15]. A possible equation for use in a product based management strategy might be as follows: $value(host) = \prod_{r \in Resource} \alpha r$. Once again, this equation was inspired by the one discussed in [15]. The resource values of every resource under considerations are simply multiplied together. The value of alpha can be altered to reflect a weight if one resource should be considered more heavily in the ordering process. All in all, summation and product methods have been used effectively to order hosts when more than one resource is under consideration.

2.2.4.2 Greedy Metric Type 2 – Ratio Method

Yet another method of combining multiple resources is one that takes the ratio of the resources under consideration. Now, multiple resources could theoretically be used in this manner however research into this method has been limited to two resources only. Specifically the CPU utilization resource and the RAM requirements resource were considered and the ratio of CPU to RAM was the only combination considered. The equation used to reach the scalar value when only CPU and RAM resource levels are considered is as follows: $value(host) = \frac{amountOfCPU}{amountOfRAM}$ [15]. It should be noted that the assignment of numerator and denominator to their respective resources could be altered and indeed could provide alternate results. Nevertheless, this was the assignment described in the related work, and the assignment used in experiments mentioned in this thesis.

Greedy algorithms are effective when it comes to the virtual machine arrangement problem for several reasons. First, any algorithm that uses a “first fit” methodology has been shown to use no more than twice the number of hosts that the optimal solution would use [15]. The proof of this is trivial and as such is omitted. In addition to this, it has been shown that the number of hosts needed is actually no more than 11/9 times the number of bins that the optimal ordering would use, plus one more bin when one uses a “first fit decreasing” methodology [31]. Additionally, the time complexity for such algorithms is quite favourable. The dominant operation in these types of orderings is the sort used to reach the final ordered state. Due to the fact that the ordering involves comparing pairs of values, the time complexity of the sort can be found to be n*log(n). This is much more desirable than say the integer linear programming technique which experiences time complexities on the order of exponentials [15]. The drawbacks to using such greedy methods are evident when considering multiple resources. They occur when the raw values of the resources are on different orders or a wildly disparate. For example, if the CPU utilization is on the order of thousands, but the RAM utilization is on the order of millions, any summation or product would be dominated by the RAM component. Expressing the resource values as percentages would be a necessity in this instance.

Additionally, using the aforementioned greedy techniques is somewhat naïve as one should be able to place virtual machines such that a host's individual resource limits are used most effectively. When combining multiple resource value into one metric, this information is lost. For example, consider a data centre that considers two resources, CPU and RAM, and utilizes the summation strategy. Imagine a host that is then assigned a scalar value of 10. This value of 10 could be the result of several combinations of CPU and RAM values. 9 CPU units and 1 RAM unit, or 5 CPU units and 5 RAM units both satisfy the equation. Thus it is in the combining of the resources that one loses information pertaining to individual resource needs. One can no longer place virtual machines in such a way to compensate for individual resource disparity. All things considered, greedy algorithms are effective due to their simple equations and their low time complexity, but may not be ideal due to their ability to obfuscate individual resource requirements.

Chapter 3

3 A Novel Algorithm for VM Arrangement

This chapter describes a new algorithm that builds upon the vector-based approach. The first novel contribution to existing vector-based approaches is the use of a balance vector that does not assume resources should be used in equal proportions. The second novel contribution is one of excluding some viable hosts from the list of potential target hosts to make a more intelligent selection.

3.1 Existing Vector-based Techniques

The basis of a vector-based technique is the use of a vector where each element represents a resource usage. Vectors can be used to represent the resource utilization of hosts and virtual machines [21]. Resource utilization is expressed as a percentage of the host's total complement for that resource. Vector-based approaches consider all utilizations as percentages. This is to ensure that the algorithm is extensible to environments where resources can differ by orders of magnitude. Figure 3 illustrates the concept of a resource utilization vector that has a dimension of two. Resource A is 90% utilized and resource B is 30% utilized.

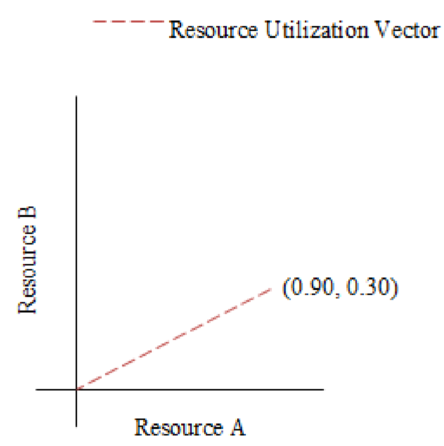

Figure 3 – A Host's Resource Utilization Vector

When a virtual machine is placed on a host, the vector addition of the virtual machine's resource utilization vector and host machine's resource utilization vector results in a new host resource utilization vector. Placing a virtual machine on an existing host is illustrated in figure 4. Vector addition is performed on the virtual machine and host vectors. Consequently, the host's resource utilization vector changes from (0.90, 0.30) to (0.95, 0.45). Placing the virtual machine on the host machine resulted in a 5% point increase in the utilization of 'resource A' and a 15% point increase in the utilization of 'resource B'. The 'Updated Host Resource Utilization Vector' now represents the utilization levels of the host machine.

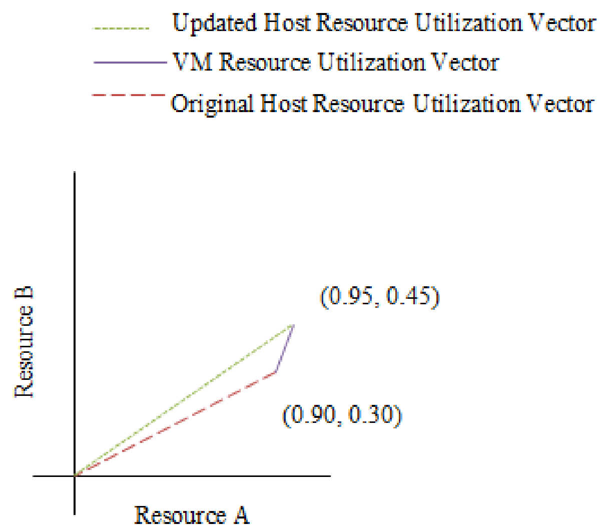

Figure 4 – A VM Resource Utilization Vector added to a Host Resource Utilization Vector

The resource utilization vector of the virtual machine may result in the updated host's resource vector's slope changing. With a vector-based approach, a virtual machine is not necessarily assigned to the first host that can accommodate it. Placement makes use of a balance vector, which represents the ideal utilization of a host machine. A virtual machine is assigned to a host that has the smallest magnitude of the updated host's rejection vector on the balance vector. The vector rejection of a vector v_i on v_j is a vector v_k which is either null or orthogonal to v_j. In this work the rejection vector measures the shortest distance between the end of one vector to another vector. Figure 5 illustrates the concept of the rejection and balance vectors. The rejection vector has one end at the terminus of the host resource utilization vector and the other end meeting the balance vector at a right angle. In Figure 5, the balance vector used represents equal utilization of each resource. Basically, virtual machines are placed on hosts in order to equalize resource usage across all resources and bring host resource utilization closer to what an ideal host's resource usage should be as represented by the balance vector. In contrast to other techniques that simply place a virtual machine onto the first host onto which it will fit, vector-based techniques utilize a best fit concept. A virtual machine's theoretical arrangement is considered on all possible target hosts, and the host that produces the best arrangement, that is the one resulting in the smallest rejection vector, is the one that is selected to house the virtual machine [21].

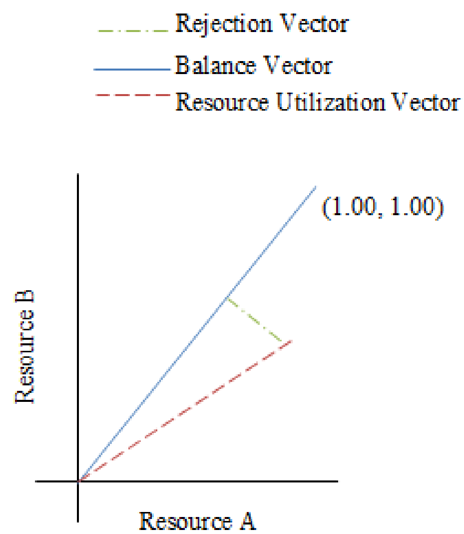

Figure 5 – Balance and Rejection Resource Utilization Vectors

For example, if a host is overutilized with respect to 'resource A' but underutilized with respect to 'resource B', a virtual machine with the opposite resource characteristics, that is one that is underutilized with respect to 'resource A' and overutilized with respect to 'resource B', could be placed on the host so that the host resource utilization is closer to the ideal resource utilization as represented by the balance vector. This is thought to allow for more virtual machine arrangements on the same host as the ability to single out particular resources should allow for arrangements that would otherwise be overlooked.

The idea behind this approach is that it should result in fewer hosts used overall [21]. This concept is illustrated using Figure 6 which considers four available hosts each with different levels of their RAM and CPU complements utilized. Now consider five additional virtual machines to be placed with varying levels of RAM and CPU requirements. Different arrangements are possible that result in a different number of hosts being needed. Note that the arrangement that utilized the fewest hosts was one that placed virtual machines with the intent of using equal amounts of each resource, percentage wise.

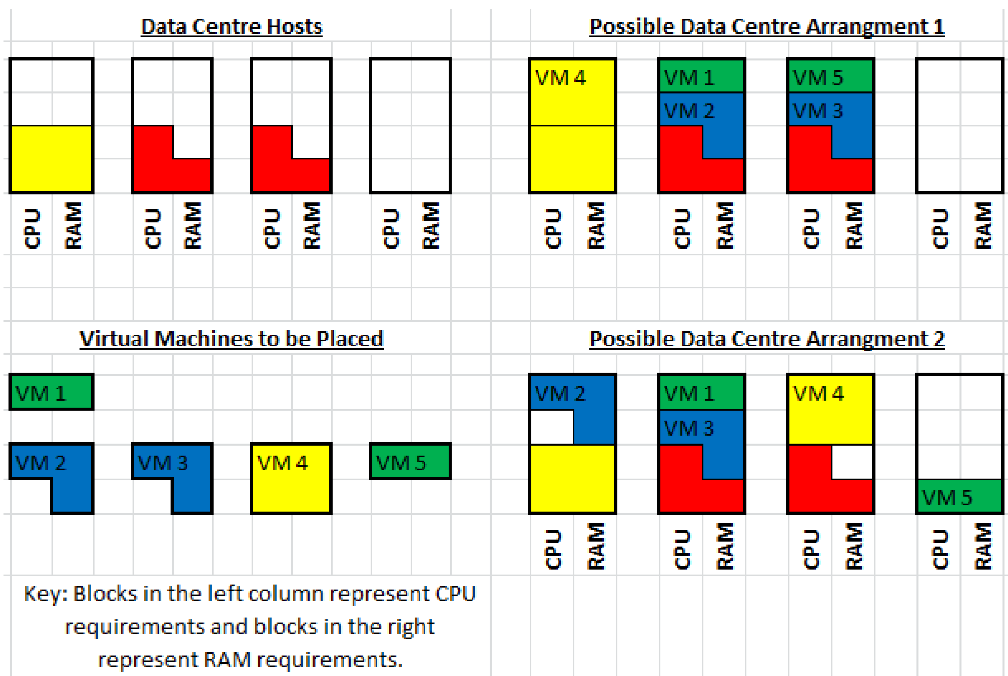

Figure 6 – Different Arrangement Choices Result in Different Number of Hosts Required

One drawback to this approach is that it is a best fit algorithm rather than a first fit. The vector-based approach must compare the VM resource utilization vector with all target host resource utilization vectors. Although an additional series of comparisons is needed, the dominating operation remains the sorting of the hosts based on the rejection vector.

The sorting can use any algorithm that sorts based on a comparison of pairs of values such as the quick sort or merge sort. The possible target hosts are sorted in increasing order of how far away the host resource utilization vector is from the balance vector. The total time complexity is on the order of V*n*log(n) where *V* is the number of virtual machines to be moved and *n* is the number of target hosts. In summary, vector-based approaches preserve individual resource requirements of virtual machines and hosts in an attempt to balance resource utilizations across all resources in a single host.

3.2 Changing the Balance Vector

Current work that uses vector-based approaches uses a balance vector with a slope of one. This assumes that the ideal host utilization is one with equal usage of the host's resources [21] i.e., for each resource the percentage of resource utilization is the same. This assumption may not always result in the best utilization of the data center resources. It may be the case that there are virtual machines that have applications that have a disparately higher need for CPU resources when compared to RAM resources. In this case it is not feasible to assume that all hosts should strive to use both resources equally. The slope of the balance vector could reflect an ideal host resource utilization that does not assume that both resources are used equally. Figure 7 illustrates this point.

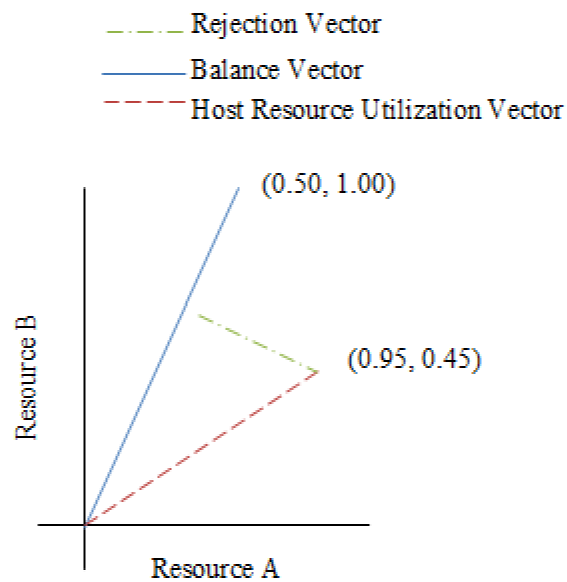

Figure 7 – Altered Balance Vector

We will represent the slope by the ratio of ideal memory usage to the CPU usage. By choosing a balance vector with a slope of 0.50:1.00 rather than 1.00:1.00, the data centre manager will strive to arrange virtual machines such that ‘resource B’ is used twice as much as ‘resource A’. This should counterbalance the resource utilization vector of the host shown which clearly uses a much higher complement of ‘resource A’ than it does ‘resource B’ at the current state of execution. The data centre manager will be inclined to place only those virtual machines that have a high ‘resource B’ requirement onto the host in question. The task then becomes one of finding the ideal slope to use in a given environment. Currently, it is not known what an ideal slope for an environment would be. We acknowledge that a single slope implemented on every host might not be ideal either. It could be the case that it might benefit a large data centre to have several racks or clusters of hosts each with their own respective slopes. Virtual machines could then be assigned to appropriate racks or clusters based on resource requirements. However, whether one considers a single rack, a cluster, or an entire data centre, we believe that a slope that counteracts the average resource requirements for virtual machines to be placed will be most effective. For example, if the resources required by the virtual machines are such that the amount of ‘resource A’ within the data centre is routinely exhausted, then an ideal slope favours arrangements that use ‘resource B’ as much as possible, while using as few hosts as possible. Once found, a more intelligent selection of target hosts can be made by the data centre manager. The hope is that this will allow for more efficient virtual machine arrangements and fewer hosts used overall. In summary, the first novel concept is the introduction of an unequal balance vector to counteract a data centre’s natural tendency to use resources unequally.

3.3 Theta Regions

The second novel component to the algorithm presented in this thesis is the inclusion of a method to exclude possible target hosts. Vector-based approaches use a best fit method to derive the target host on which a given virtual machine is to be placed [21]. However, there is often only a subset of the possible targets that would benefit from the addition of another virtual machine. Consequently, if the set of possible targets was narrowed down then it would speed up the assignment of a VM to a host machine. Existing vector-based approaches employ some method to accomplish this. This usually involves graphing vectors on a plane that represents the utilization of each resource and then selecting hosts that are in a region that is diametrically opposed to the region in which the virtual machine’s vector resides [21]. This technique does not take into account the fact that some suitable hosts are in a severely underutilized state and would be better off being powered down and having their virtual machines consolidated on another host. As such, better opportunities to balance the resource utilizations of certain hosts may be omitted. For example, consider a data centre with two identical hosts. The first host is severely off balance with 90% of its CPU complement in use and 5% of its RAM complement in use. The second host is slightly off balance with 10% of its CPU complement in use and 5% of its RAM complement in use. A virtual machine that would use 10% of a host’s CPU complement and 95% of a host’s RAM complement now needs to be placed. The virtual machine should be placed on the first host to use all of the host’s available resources. If the virtual machine were to be placed on the second host, only the second host’s RAM complement would be exhausted. Additionally, 80% of the second host's CPU complement would be rendered unusable. (A virtual machine cannot utilize only CPU resources.) Consequently, preference as a target should be given to off balance hosts.

The algorithmic construct that has been designed to achieve this is referred to as the "theta region". It is so named because the region takes on the shape of an isosceles triangle (when utilized in two dimensions) and the size of the triangle can be uniquely identified by the angle that the balance vector makes with one of the equal sides of the triangle. By overlaying this triangle, or theta region, on a Cartesian plane populated by vectors representing the resource utilizations of hosts, it has the effect of partitioning the set of available target hosts into those that are balanced and those that are off balanced. This can be seen in figure 8. The physical machines can be sorted by their degree of imbalance and achieve the same effect by considering a subset of the resulting list.

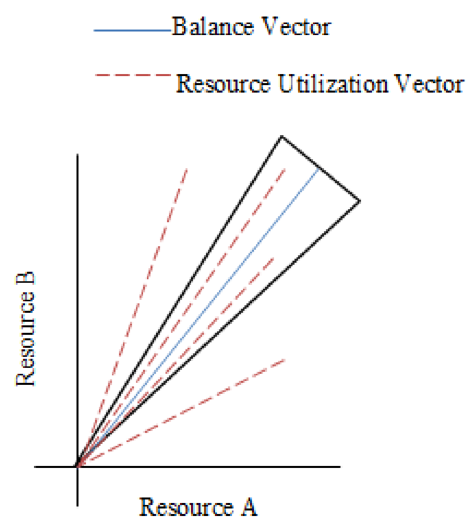

Figure 8 – The Theta Region

The challenge is determining the size of the theta region to reach a subset of target hosts that is small enough to include only the pertinent hosts but large enough to account for the possibility of a host not being able to accommodate an incoming virtual machine. In summation, the second novel contribution is the concept of a region that acts to partition the set of possible target hosts such that a more intelligent host can be selected to receive a given virtual machine.

3.4 The New Vector-Based Algorithm

The new vector-based algorithm outlined in this thesis builds on existing algorithms [21]. It contains the novel additions outlined above that should provide a more economic assignment of virtual machines to data centre hosts. First, it should be mentioned that in order to implement this algorithm in a data centre, it must be replicated in three different instances. The algorithm must be implemented with respect to virtual machine placement, virtual machine relocation, and virtual machine consolidation operations.

Virtual machine placement refers to placing an incoming virtual machine on a host machine [3]. When placement occurs, virtual machines are considered individually, one at a time, as they are created from the pool of incoming client applications. Virtual machine relocation occurs when the data centre manager has determined that a virtual machine arrangement that can execute more virtual machines without powering on additional hosts can be attained by moving existing virtual machines from one host to another [3]. Periodically the data centre manager classifies all hosts as stressed, partially utilized, and underutilized. Virtual machines may be migrated from a stressed host to a host that is not stressed. The definition of stressed, partially utilized and underutilized has been made a priori and is outside of the scope of the algorithm described in this section. The periodicity with which these classifications are made, as well as the resource utilization levels associated with stressed, partially utilized and underutilized states are experimental parameters that are listed in chapter 4. Finally, virtual machine consolidation occurs when the data centre manager has determined that a virtual machine arrangement that can execute the same number of virtual machines on fewer hosts can be attained by vacating all of the virtual machines from an underutilized host, moving said virtual machines to new hosts, and powering off the original host [3]. Relocation and consolidation may also be triggered by other factors, such as SLA violations. In the end, the decision as to whether or not a physical host should be part of a relocation or consolidation depends on the objectives one wishes to achieve within the data centre.

However, proper utilization levels and minimization of SLA violations are most definitely examples of such goals. It should also be noted that a key difference between placement and relocation or consolidation is observed. Whereas placement only involves a single virtual machine, relocation and consolidation handle a set of virtual machines for which new target hosts must be found. The policies used by the three individual operations combine to form a single data centre policy [3]. It should be noted that the algorithm assumes that a suitable balance vector slope and a theta region have been determined a priori. Once the decisions regarding the periodicity of the aforementioned processes and the definitions of stressed, partially utilized, and underutilized have been assigned, the algorithm is then run to create a suitable virtual machine arrangement. The algorithm is run each time there is a need for the data centre manager to perform a placement, relocation, or consolidation operation.

```
Input: VMs, theta, slope
Output: -
targetFound = false
z, pBig, pSmall, uBig, uSmall, e = classHosts(hosts, theta, slope)
targetCategories.add(pBig, uBig, pSmall, uSmall, e)
for vm in VMs do
    for category in targetCategories do
         targets = sortCategoryByRej(category, vm, slope)
         for host in targets do
           if host.hasCapacity(vm) then
              host.deploy(vm)
              targetFound = true
              break
           end if
         end for
         if targetFound then
           break
     end for
end for
```

Figure 9 – The New Vector-Based Algorithm

First, existing hosts must be categorized. There are a total of five categories into which a host may fall. The first category is for hosts that are partially utilized and whose resource vector's terminus is outside the theta region. This category is represented by the variable 'pBig'. The second category is for hosts that are underutilized and whose resource vector's terminus is outside the theta region. This category is represented by the variable 'uBig'. The third category is for partially utilized hosts whose resource vector's terminus is inside the theta region. This category is represented by the variable ‘pSmall’. The fourth category is for hosts that are underutilized and whose resource vector’s terminus is inside the theta region. This category is represented by the variable ‘uSmall’. The fifth and final category is that of empty hosts. This category is represented by the variable ‘e’. For the sake of completion, it should be noted that stressed hosts are never considered valid target hosts for any virtual machine movement operation. This category is represented by the variable ‘z’. This can be seen in lines 4 and 5 of the algorithm. Next, for every virtual machine that is to undergo some sort of movement, a suitable target must be found. This is seen in the loop that starts at line 6 in the algorithm. This is accomplished by inspecting the categories one by one in the order that they were described above. This order was chosen as it best replicated the order and successes found in [18]. This order was successful in that it allowed the data centre to achieve the best utilization levels while incurring few SLA violations when compared to other permutations of the above categories. This success stemmed from the fact that a new host was turned on only after all other hosts were checked and deemed unfit to house an incoming virtual machine. That is, turning on another host was only done so as a last resort. The loop that accomplishes this occurs at line 7. Within each category of potential target hosts, each target host is checked to see if placing a virtual machine on it will make the host’s resource utilization vector trend more towards the balance vector. The terminus of the newly created vector is calculated and its rejection from the balance line is computed. The category of potential target hosts is then sorted by the magnitude of said rejection in increasing order. This occurs at line 8 in the algorithm through the calling of the *sortCategoryByRej* function. This gives the effect of placing the virtual machine onto the host whose state will then be closest to being balanced as a result of hosting the virtual machine undergoing the movement. Next, each host in the category is inspected in the order described above and the first host onto which the virtual machine will fit is selected as the target. The sort combined with the fact that the targets are inspected in order results in a best fit heuristic. This process begins at line 9 of the algorithm. If the target can accommodate the incoming virtual machine with respect to its resource needs, the movement is recorded, set to be executed at the end of the derivation of the arrangement, and the algorithm moves on to the next virtual machine to be moved. If the selected host cannot accommodate the virtual machine, the next target host in the category is inspected. This is accomplished in lines 10-17 of the algorithm. This process repeats until the subset of target hosts is exhausted or until an appropriate target can be found. If the category of hosts is exhausted, the algorithm sorts and inspects the next category of hosts in the manner described above. This process continues until all categories have been exhausted and either results in a new host being turned on or the data centre simply cannot hold another virtual machine. It should be noted that as is the case with other data centre manager policies, conditions are in place to ensure that the source host for a given machine cannot be the same as its destination during relocation and consolidation operations, for obvious reasons. This was seen in the experiments conducted in [2]. In summary, the new vector-based algorithm organizes possible hosts into 5 categories, considers said categories one by one, and sorts potential target hosts in increasing order of the magnitude of the rejection of the vector made by the sum of the host's resource vector and the virtual machine's resource vector with respect to the balance vector.

Chapter 4

4 Experimental Design and Results

This chapter investigates the effectiveness of the new vector-based approach described in Chapter 3. Section 4.1 describes the simulator used in the experiments. Section 4.2 describes the workload traces. Section 4.3 presents the utilization levels used to define underutilized, partially-utilized and stressed hosts. Section 4.4 describes the metrics used to evaluate the different virtual machine arrangements. In sections 4.5 to 4.7, three experiments are presented.

4.1 Simulator

The simulator used in the experiments is DCSim [3] [2]. The simulated data centre configuration used in the experiment, DCSim, consisted of 200 host machines, of which there were an equal number of two types of hosts. The first type of host was modeled after the HP ProLiant SL380G5, with 2 dual-core 3Ghz CPUs and 8GB of RAM. The other type of host was modeled after the HP ProLiant SL160G5 with 2 quad-core 2.5GHz CPUs and 16 GB of RAM. The power consumption of both hosts is calculated using the SPECPower benchmark. The power efficiency of the first type of host was 46.51cpu/watt. The power efficiency of the second type of host was 85.84cpu/watt.

4.2 Traces

The five workload traces used consisted of traces from Clarknet, EPA, SDSC, and two Google cluster data traces. These traces consist of HTTP server requests. The traces were sampled over a fixed time interval and the number of requests during the interval spurred the creation of virtual machines. A greater number of requests resulted in a virtual machine being created with higher resource requirements and a smaller number of requests caused virtual machines to be spawned with lower resource requirements [3]. Five random seeds were chosen as the five starting points in each of the workload traces for the simulations. This was to control the possibility of one point in the traces being more favourable to any given virtual machine placement policy. The random seeds remained constant across all of the experiments and across all changes in policy.

4.3 Virtual Machine Arrival, Departure, and SLAs

Virtual machine arrival and departure was based on several workload traces. These workload traces contained counts of how often server requests were made over a period of time. Higher rates of server requests spur the creation of virtual machines with larger CPU resource requirements and lower rates of server requests spur the creation of virtual machines with smaller CPU resource requirements [3]. These simulated hosts specifications and methods of virtual machine arrival and departure conformed to the specifications outlined in other implementations of experiments that also used DCSim, namely [2]. In DCSim, an SLA violation occurs when resources required by a VM are not available to it and thus performance is impacted. DCSim reports the percentage of time that that amount of required resources was not provided to the VM. For migration, DCSim applies a penalty which corresponds to the percentage of time that sufficient resources are not available to a VM while it is being migrated.

4.4 Utilization Levels

The CPU baseline requirements were set at the creation of the virtual machine. During execution, the CPU requirements of the virtual machines were allowed to fluctuate through a range of 200 CPU resource units. The RAM values were static. For the placement operation, the values used for CPU underutilization and stressed hosts were 60% and 85% with partial utilization being the range between those two values. For the relocation operation, the values used for CPU underutilization and stressed hosts were 60% and 85% with partial utilization being the range between those two values. For the consolidation operation, the values used for CPU underutilization and stressed hosts were 60% and 95% with partial utilization being the range between those two values. The relocation and consolidation operations were run periodically at 10 minutes and one hour of simulation time, respectively. These values also form the criteria by which the hosts are categorized as outlined in the algorithm in chapter 3.

4.5 Evaluation Metrics

There are two metrics used for evaluation. The first metric is the maximum number of hosts used during a given simulation. The second metric is the number of SLA violations that occur during a given simulation. The first metric is an indicator of energy usage in that fewer hosts used typically implies less power consumption. Our goal is to minimize the number of hosts used while committing the least number of SLA violations as possible.

4.6 Experiment 1

This experiment is used to determine the effect of balance vectors where the slope is not one and the effect of the theta region.

4.6.1 Virtual Machines Used

Three types of virtual machines were used as follows:

- Virtual machine type one's resource requirements included 500 shares of CPU resource, 1024MB of RAM, one CPU core and 1GB of storage.

- Virtual machine type two's resource requirements included 500 shares of CPU resource, 1024MB of RAM, one CPU core and 1GB of storage.

- Virtual machine type three's resource requirements included 2500 shares of CPU resource, 1024MB of RAM, two CPU cores and 1GB of storage.

These configurations allow hosts to become unbalanced.

4.6.2 Policies

Different policies choose target hosts in a different manner. The first policy randomly selects virtual machines for migration when a host becomes overloaded. This policy was used as a baseline for comparison with other policies. The second policy uses the vector-based approach described in chapter 3. The slopes used for the balance vector are described in section 4.6.3. Theta regions were not used. The third policy uses the vector-based approach described in chapter 3 but also used the theta region.

4.6.3 Slope and Theta Region Values

The value of the angle identifying the theta region was set to ten degrees for the third policy. The value of the theta region remained the same for the duration of the experiment since the purpose of this experiment was to determine if the theta region has an effect. Ten degrees was seen as a reasonable value but was arbitrarily chosen. Slope values are expressed in the form of a CPU:RAM ratio. The final set of slope values for the vector-based algorithms used in the second and third policies is as follows: 2.00:1.00, 1.50:1.00, 1.10:1.00, 1.00:1.00, 1.00:1.10, 1.00:1.50, 1.00:2.00.

4.6.4 Experiment 1 Results

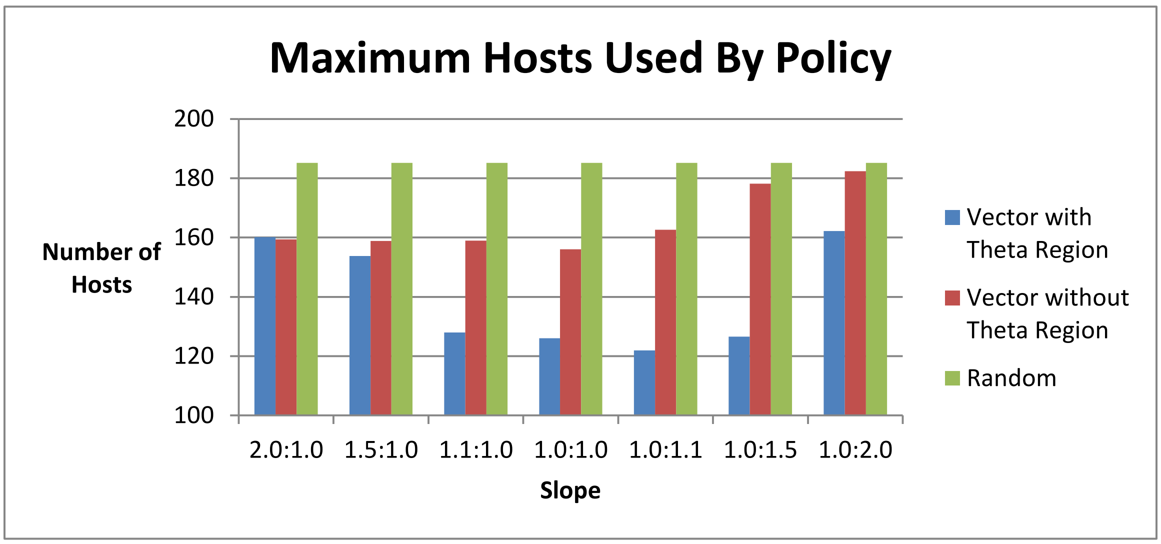

Figure 10 – Experiment 1 Results

A detailed table of the results of experiment one may be found at the end of section 4.6.4. This experiment shows for the second policy that the number of hosts utilized was different for different slopes of the balance vector. This is seen in Figure 10 when comparing the second policy’s worst performance, which occurred when using a ratio of 1.00:2.00 and resulted on average using 182.4 hosts, to the second policy’s best performance, which occurred when using a ratio of 1.50:1.00 and resulted on average using only 158.8 hosts. Using a balance vector where the slope is not one in vector-based approaches and not using a theta region can have an impact on the produced arrangements.

The third policy uses the vector-based algorithm with a theta region. As with the second policy the number of hosts utilized was different for different slopes of the balance vector. Using a ratio of 1.00:2.00 resulted on average using 162.2 hosts while using a ratio of 1.00:1.10 and resulted in using only 122 hosts.

Table 1 shows that at no time did any simulation produce service level agreement violations in excess of 0.101%. This experiment used an experimental environment as described in [2] where three policies that only considered CPU were designed to maximize the utilization of a data centre incurred SLA violations of 0.228%, 0.223%, and 0.220%. The SLA violation values incurred with this experiment are less than half of any of SLA violation values when only CPU was considered. Furthermore the SLA violations incurred in this experiment are similar to that of the Foster et al. Hybrid policy used in [2] that incurred a penalty of 0.092%. In terms of SLA violations, the policies used in this experiment are comparable to those found in the literature.

This experiment showed that the inclusion of a theta region to exclude possible target hosts provided statistically insignificant better results compared to not using the theta region. This effect is seen when comparing the number of hosts used by the vector-based approach without a theta region to the number used by the vector-based approach with a theta region. The vector-based approach with a theta region used fewer hosts for a variety of ratio used. Results show that the vector-based approach with a theta region performs at least as well as the vector-based approach without a theta region on every ratio except 2.00:1.00 where the vector-based with a theta region performed worse by a fraction of a host. Furthermore, the disparity in performance is evident when considering results obtained at ratios between 1.50:1.00 and 1.00:1.50. In every case, the vector-based approach with a theta region used fewer hosts. The effectiveness of the theta region is most evident after examining the results finding that the least number of hosts used by the vector-based approach without a theta region is 156 hosts whereas the least number used by the vector-based approach with a theta region is fewer at 122 hosts. Once again, it should be noted that at no time did any simulation produce service level agreement violations in excess of 0.101%. The vector with theta policy performed statistically insignificantly better than the vector without theta policy. Also, regarding the vector with theta policy, the best slope/theta combination performed statistically insignificantly better than the worst slope/theta combination. The statistical analysis can be found in appendix A.

Table 1 – Experiment 1 Tabulated Results

Experiment 1 Averaged Over 5 Heats						
Policy	**Slope**	**Max Active Hosts**	**SLA Violation**	**Power Consumption (kWh)**	**Avg DC Util**	**Avg Host Util**
Vector Approach with Theta Region of 10 Degrees	2.00 : 1.00	160.2	0.101%	3768.5992	35.65%	60.69%
	1.50 : 1.00	153.8	0.099%	3662.1114	35.53%	62.37%
	1.10 : 1.00	128	0.045%	3392.8988	35.36%	69.17%
	1.00 : 1.00	126	0.027%	3408.3618	35.47%	69.70%
	1.00 : 1.10	122	0.016%	3422.5262	35.46%	69.21%
	1.00 : 1.50	126.6	0.016%	3587.6228	35.71%	66.88%
	1.00 : 2.00	162.2	0.055%	4061.2726	35.78%	59.82%
Vector Approach without a Theta Region	2.00 : 1.00	159.4	0.091%	3843.8032	35.48%	58.72%
	1.50 : 1.00	158.8	0.088%	3878.4224	35.72%	59.04%
	1.10 : 1.00	159	0.083%	3846.355	35.61%	59.07%
	1.00 : 1.00	159.6	0.081%	3861.7744	35.46%	58.56%
	1.00 : 1.10	162.6	0.087%	3933.4162	35.72%	58.09%
	1.00 : 1.50	178.2	0.099%	4194.9368	36.42%	57.17%
	1.00 : 2.00	182.4	0.080%	4389.8494	37.28%	57.16%
Random	N/A	185.2	0.068%	4975.0458	36.10%	52.62%

4.6.5 Experiment 1 Discussion

This section discusses why the two novel contributions may have an effect. Using a balance vector with a slope of one does not consider the resource requirements of the individual virtual machines. The resource requirements of these virtual machines are not necessarily equally balanced so there is no reason to believe that the optimal arrangement of the virtual machines should result in a perfectly balanced host. This suggests focusing on having hosts use their resources to reflect virtual machine resource requirements. This can be achieved by allowing for balance vectors to have a slope other than one. In experiment one the ideal balance vector ratio, that is to say the one that resulted in the fewest number of hosts being needed, was not 1.00:1.00 but rather 1.00:1.10. This 10% preference of one resource over the other implies that the virtual machines themselves fit best when the data centre manager did not try to use both resources equally but rather gave a slight preference or handicap to one. This reflects the unbalanced nature of the virtual machines that can be seen when inspecting the resource requirements of the individual machines set forth in the environment's setup. Upon inspection, it is easy to see that not one of the virtual machines utilizes resources equally. It is the view of this researcher that the value of 1.00:1.10 was the most successful slope in experiment one because it best counteracted the tendencies of the combined efforts of all three virtual machines to slightly favour the CPU resource over the RAM resource. This researcher acknowledges that this particular slope's success does not necessarily generalize to all environments. In summary, an altered balance vector most likely achieves its success because it works to counteract virtual machine arrangements that would otherwise unbalance the data centre overall, yet does not force the resource allotments of a single host to be precisely equal.

Without a theta region, virtual machines are placed on the host that will become the closest to becoming balanced as a result of the newly added virtual machine. This is a logical way to proceed as it simply places virtual machines where at least some benefit is reaped. However, experiment one illustrates that not all arrangements that trend toward improvement should be treated equally. In fact, experiment one seems to imply that it is more important to prioritize hosts that are at risk of becoming unbalanced before tending to those that may become perfectly balanced as the result of the next virtual machine movement operation. There does appear to be some logic behind these results. Unbalanced hosts are generally more difficult to move virtual machines onto. This is because it only takes maxing out of one of the host's resource complements for the movement to fail. Often, the only recourse to a data centre with sufficiently off balance hosts is to power on additional empty hosts. Consequently, it makes sense that a data centre will be more successful if it strives to keep as few unbalanced hosts as possible. To achieve this, every opportunity must be made to transition a host from an unbalanced state and this is accomplished by moving to it virtual machines whose resource requirements naturally counterbalance the current resource utilization levels of the unbalanced host. It is akin to placing more weight on an unbalanced scale. The theta region ensures that rather than wasting virtual machine arrangements on hosts that are already nearly balanced, unbalanced hosts have an opportunity to stabilize. The question now becomes one of finding the ideal size of the theta region, if such a size exists. The theta region should not be too small lest the algorithm fail to exclude any hosts nor too big lest the algorithm exclude all of the hosts. Ideally, the theta region would exclude all but the most off balance suitable host. However, the fact that a given virtual machine might not fit in the most off balance host requires the set of target hosts to strive for some cardinality that allows for off balance hosts to be tended to while still allowing for the possibility that a portion of said off balance hosts might not be suitable. In summation, the concept of the theta region succeeds by allowing data centre managers to identify target hosts that would result in a lesser need to power on additional hosts.

4.7 Experiment 2

The purpose of experiment two was to examine the interactions, if any, between the novel constructs of the altered balance vector and the theta region when they were varied simultaneously. In other words we used experiment two to determine the effect of different combinations of theta values and slope ratios.

4.7.1 Virtual Machines Used

Three types of virtual machines were used as follows:

- Virtual machine type one’s resource requirements included 1500 shares of CPU resource, 512MB of RAM, one CPU core and 1GB of storage.

- Virtual machine type two’s resource requirements included 2500 shares of CPU resource, 512MB of RAM, one CPU core and 1GB of storage.

- Virtual machine type three’s resource requirements included 2500 shares of CPU resource, 1024MB of RAM, two CPU cores and 1GB of storage.

These figures were chosen as they allowed resource usage on hosts to become unbalanced.

4.7.2 Policies

All simulations were run in the data centre with two different policies. The first policy was one in which virtual machines were randomly selected for migration when a host became overloaded. This was used as a control to compare with the other policy. The second policy is the same as the third policy defined for Experiment 1.

4.7.3 Slope and Theta Region Values

The set of ratio values for the new vector-based algorithm expressed in the format of CPU:RAM were as follows: 0.5:1.0, 1.0:1.0, 1.5:1.0, 2.0:1.0, 2.5:1.0. The theta values were as follows: 5 degrees, 10 degrees, 20 degrees, 30 degrees. .

4.7.4 Experiment 2 Results

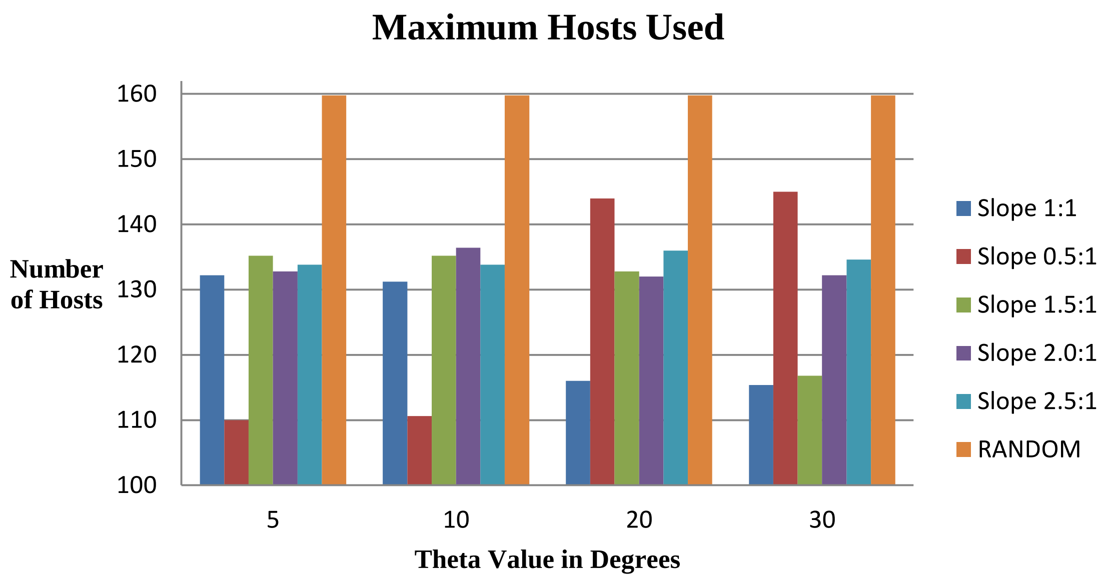

Figure 11 – Experiment 2 Results

A detailed table of the results of experiment two may be found at the end of section 4.7.4. Figure 11 shows the number of hosts used for different pairings of slope and theta values and Table 2 presents results on power consumption and SLA violations for different pairings of slope and theta values. Experiment 2 shows that there are several pairings of various values from the set of ratios and set of thetas that provide approximately the same number of minimal hosts used in their respective virtual machine arrangements. However, it was at first surprising that the pairings that were the most successful did not have any commonalities. That is to say that they shared neither a slope ratio nor a theta value. For example, the pairing that performed the best with approximately 110 hosts used in the arrangement occurred with a slope ratio of 0.5:1.0 and a theta value of 5 degrees. A nearly as successful pairing was that of the ratio of 1.5:10 with a theta value of 30 degrees. Not only that, but the pairing of the ratio of 1.5:1.0 with a theta of 5 degrees performed relatively poorly, utilizing approximately 135 hosts, as did the pairing of 0.5:1.0 degrees with the theta value of 30 degrees as it used approximately 145 hosts. It seems that not only do suitable pairings not have any factors in common, pairings that contain factors that performed well in other instances actually performed abysmally. In summary, the results of experiment 2 imply that there are multiple suitable pairings for the new vector-based approach but pairings that consist of factors that were found in successful pairings perform poorly when paired together.

Table 2 – Experiment 2 Tabulated Results

Experiment 2 Averaged Over 5 Heats							
Policy	Slope	Theta	Max Active Hosts	SLA Violation	Power Consumption (kWh)	Avg DC Util	Avg Host Util
New Vector	0.50:1.00	5	110	0.007%	2652.6522	19.97%	42.79%
		10	110.6	0.004%	2666.301	19.86%	43.22%
		20	144	0.004%	2976.2954	20.71%	37.98%
		30	145	0.004%	3012.6758	20.76%	37.69%
	1.00:1.00	5	132.2	0.027%	2992.7274	20.80%	37.54%
		10	131.2	0.023%	2944.6028	20.80%	38.22%
		20	116	0.008%	2717.0464	20.07%	41.27%
		30	115.4	0.004%	2778.8012	19.99%	43.56%
	1.50:1.00	5	135.2	0.028%	3059.0454	20.27%	37.12%
		10	135.2	0.028%	3050.4632	20.25%	37.14%
		20	132.8	0.024%	2998.401	20.23%	37.79%
		30	116.8	0.009%	2746.4024	19.87%	40.63%
	2.00:1.00	5	132.8	0.028%	3003.2698	20.01%	37.36%
		10	136.4	0.028%	3010.283	20.06%	37.36%
		20	132	0.028%	2999.3992	20.03%	37.41%
		30	132.2	0.021%	2927.449	20.09%	38.47%
	2.50:1.00	5	133.8	0.028%	2995.7836	20.01%	37.40%
		10	133.8	0.028%	2989.3718	20.02%	37.48%
		20	136	0.028%	2997.407	20.08%	37.48%
		30	134.6	0.026%	2980.046	20.10%	37.78%
Random	N/A	N/A	159.8	0.026%	3725.8962	21.59%	41.75%

4.7.5 Experiment 2 Discussion

On the surface, the results of experiment two are not intuitive. However, upon further inspection, the results obtained in experiment two do indeed make sense when considering the objective of the new vector-based approach. The objective of the new vector-based approach is to find a suitable cardinality for the set of target hosts. The new vector-based approach strives to consider only those hosts that are off balanced and suitable for the incoming virtual machine as a target host. Adjusting the balance vector, and the size of the theta region (which is always positioned relative to the balance vector) one directly affects the cardinality of the set of target hosts. By choosing a balance vector that is in the middle of the host resource vectors and a small theta region, it stands to reason that a small number of hosts will be excluded. However, by choosing a balance vector that is far away from all of the host resource vectors and choosing a large theta region also means that a small number of hosts will be excluded. This is because the theta region will ever so slightly encroach on the host resource vector population. This phenomenon is akin to fishing with a net where the balance vector represents the position of the boat in the water and the theta region represents the size of the net. The number of fish caught represents the cardinality of a set. There are two ways to catch the same number of fish. First you may use a small net if your boat is positioned right on top of a school of fish. The fish you catch will most likely be from the centre of the mass of fish. Second, you may use a large net if your boat is far from a school of fish. The fish you catch will most likely be from the periphery of the mass of fish. Nevertheless, the number of fish caught will be similar. This is similar to the new vector-based approach. There are multiple ways to exclude an appropriate amount of hosts. First, you may position the balance vector in the middle of the host resource vectors and use a small theta region. Or, you may position the balance vector away from all of the host resource vectors and use a larger theta region. The cardinality of the set of possible target hosts may be similar and thus similar results are logically obtained. The implication is that there is flexibility in choosing slope values and theta values. A value can be chosen for a construct and the value for the other construct can then be assigned a value. This makes the new vector-based approach easier to deploy.

4.8 Experiment 3

The purpose of experiment three was to examine the performance of the new vector-based algorithm when compared with other virtual machine arrangement algorithms.

4.8.1 Virtual Machines Used

Three types of virtual machines were used as follows:

- Virtual machine type one's resource requirements included 1500 shares of CPU resource, 512MB of RAM, one CPU core and 1GB of storage.
- Virtual machine type two's resource requirements included 2500 shares of CPU resource, 512MB of RAM, one CPU core and 1GB of storage.
- Virtual machine type three's resource requirements included 2500 shares of CPU resource, 1024MB of RAM, two CPU cores and 1GB of storage.

4.8.2 Policies

All simulations were run in the data centre with six different policies. The first policy was one in which virtual machines were randomly selected for migration when a host became overloaded. This was used as a control to compare with the other policies. The second policy was an implementation of the new vector-based virtual machine arrangement policy that had the ability to have its balance vector's slope and theta region value altered prior to the running of the simulation. This is the policy described in chapter 3. The third policy was an implementation of a policy that focused on one resource rather than two. It is referred to as the Foster et al. Hybrid policy and can be found in [2]. The fourth policy was an implementation of the summation policy mentioned in section 2.2.4.1 of this thesis. The fifth policy was an implementation of the product policy mentioned in section 2.2.4.1 of this thesis. The sixth policy was an implementation of the ratio policy mentioned in section 2.2.4.2 of this thesis. The last three policies are all greedy techniques. The purpose of experiment three was designed to compare the new vector-based approach to the other approaches described in the literature

4.8.3 Slope and Theta Region Values

The following sets of slope values and theta values for the new vector-based algorithm were tested and the best performing pair was represented in the final results: CPU:RAM: 0.5:1.0, 1.0:1.0, 1.5:1.0, 2.0:1.0, 2.5:1.0. The theta values were as follows: 5 degrees, 10 degrees, 20 degrees, 30 degrees.

4.8.4 Experiment 3 Results

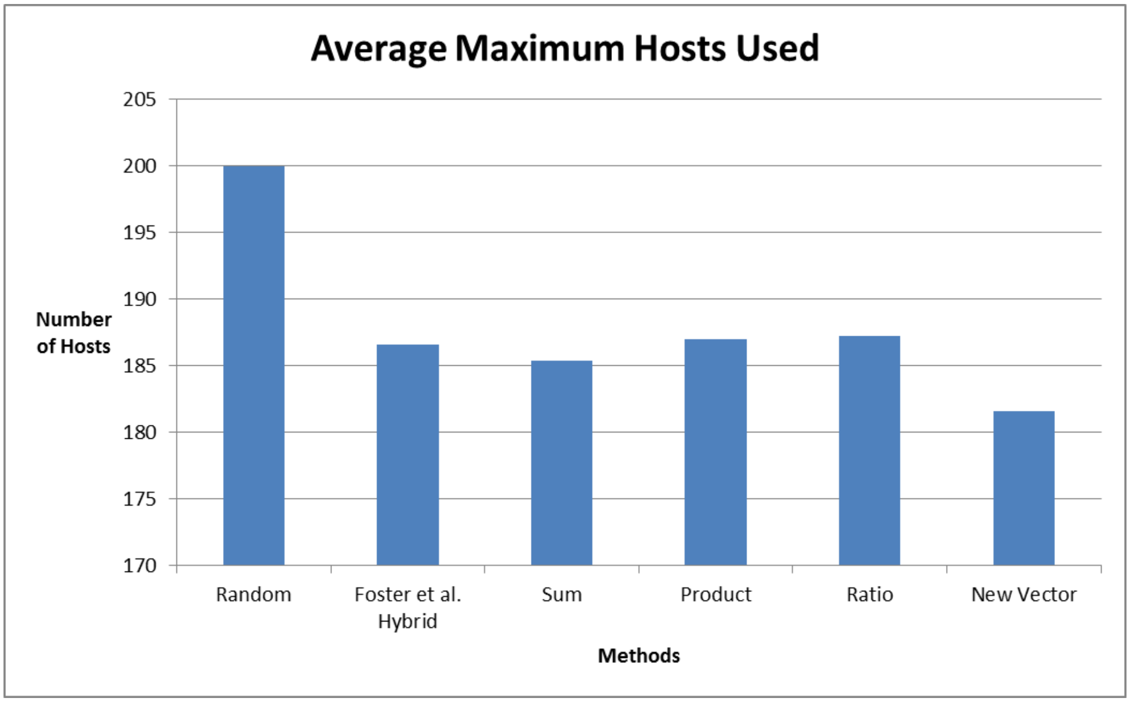

Figure 12 – Experiment 3 Results

A detailed table of the results of experiment three may be found at the end of section 4.8.4. The results of experiment three showed that the new vector-based approach performed statistically insignificantly better than all of the other strategies. The random strategy performed worst. It utilized all 200 of the available hosts in the data centre. The ratio strategy, that is the one that ordered the hosts based on the ratio of a hosts CPU to RAM utilization, utilized, on average, 187.2 hosts. The product strategy performed ever so slightly better utilizing 187 hosts on average. The Foster et al. Hybrid policy managed to utilize only 186.6 hosts on average. The summation strategy performed better still utilizing only 185.4 hosts on average. Lastly, the new vector-based strategy performed the best with 181.6 hosts used on average. It should also be noted that the new vector-based strategy performed well with respect to service level agreement violations as well. The new vector-based approach's best simulation encountered service level agreement violations of 0.077% which is definitely comparable to the violations of its nearest competitor, the summation method, which had service level agreement violations of

0.046%. Furthermore, the new vector-based algorithm performed worse than the Foster et al. Hybrid policy with respect to service level agreement violations in that the Foster et al. Hybrid policy had a service level agreement violation value of 0.041%. However, with such small values, this defeat is not taken too heavily. The new vector-based policy performed statistically significantly better than the random policy with respect to the number of hosts required. However, the new vector-based policy did not perform statistically significantly better than any other policy with respect to the number of hosts required. In conclusion, although the new vector-based approach outperformed all of the other strategies as it used on average, less hosts than all of the other strategies, it did not perform statistically significantly better. The statistical analysis may be found in appendix A.

Table 3 – Experiment 3 Tabulated Results

Experiment 3 Averaged Over 5 Heats							
Policy	**Slope**	**Theta**	**Max Active Hosts**	**SLA Violation**	**Power Consumption (kWh)**	**Avg DC Util**	**Avg Host Util**
New Vector	0.50:1.00	5	199.8	0.241%	5565.8662	68.63%	52.01%
		10	195	0.201%	5412.6312	69.95%	52.13%
		20	181.6	0.091%	4827.2022	77.18%	52.20%
		30	185.4	0.045%	4865.0514	76.65%	52.56%
	1.00:1.00	5	181.6	0.077%	4802.6412	77.65%	52.26%
		10	185.6	0.051%	4843.1434	77.09%	52.57%
		20	184.6	0.055%	5144.3492	77.54%	52.76%
		30	183.2	0.080%	5223.5822	77.83%	52.62%
	1.50:1.00	5	185	0.068%	4897.7146	77.33%	52.52%
		10	185.4	0.060%	4929.7976	76.79%	52.31%
		20	184.6	0.044%	4903.3554	77.15%	52.53%
		30	184.8	0.041%	4870.78	77.23%	52.61%
	2.00:1.00	5	181.8	0.096%	5029.9838	77.47%	51.92%
		10	183.6	0.097%	5098.3732	77.34%	52.44%
		20	185.4	0.056%	4959.209	76.90%	52.42%
		30	184	0.039%	4948.3446	76.98%	52.57%
	2.50:1.00	5	186	0.104%	5010.9974	76.71%	51.61%
		10	182.6	0.106%	5163.3588	77.40%	52.01%
		20	185	0.083%	5085.3868	76.94%	52.34%
		30	185.6	0.046%	5012.8762	76.90%	52.56%
Foster et al. Hybrid	N/A	N/A	186.6	0.041%	4984.0286	76.61%	51.78%
Sum	N/A	N/A	185.4	0.046%	5304.9078	77.24%	51.80%
Product	N/A	N/A	187	0.048%	5361.673	77.27%	51.68%
Ratio	N/A	N/A	187.2	0.045%	4840.7176	76.48%	52.08%
Random	N/A	N/A	200	0.076%	6671.914	60.09%	51.46%

4.8.5 Experiment 3 Discussion

The results of experiment three show the statistically insignificant success of the new vector-based approach. The results of experiment three also support existing literature that suggests that approaches that do not obfuscate information pertaining to individual resource levels should perform more admirably [21]. This is ostensibly a result of the data centre manager being able to make more intelligent arrangement decisions with respect to virtual machine movements. In turn, this naturally lends itself to individual hosts being utilized to a higher level and consequently results in better data centre utilization overall. Both the sum and product strategies suffer from their respective shortcomings mentioned in section 2.2.4.1 of this thesis. The same could be said for the ratio strategy. The Foster et al. Hybrid policy was at an inherent disadvantage because it only considered one of the resources in its arrangement strategy. In order to compensate for this, an environment that would most benefit the Foster et al. Hybrid policy was chosen. Nevertheless, the new vector-based approach outperformed the aforementioned strategies. In conclusion, the new vector-based approach performed best most likely because it utilized all of the information available to it in a logical manner.

Chapter 5

5 Future Work and Conclusions

Although several conclusions were reached through the experiments detailed in this thesis, there is still room for future research. First, the algorithm naturally lends itself to being extended beyond two resources. Also, the process of finding suitable values for the theta region and balance vectors could be automated. Additionally, there are additional methods of ranking the hosts that still involve the magnitude of the rejection; these should be explored. These research areas could add to the conclusions that have already been made through this thesis. It was concluded that different values for the balance vector most definitely affect the success of the algorithm. Furthermore, the usefulness of the theta region construct was also validated. Additionally, an interesting interaction between the novel contributions was observed leading to another incentive to use the new vector-based approach. Lastly, the new vector-based algorithm was shown to outperform other methods. This chapter details possible avenues for future study such as extending the new vector-based algorithm into the third dimension, automating some aspects, and altering the ordering criteria, as well as codifies the conclusions reached through the experiments.

5.1 Future Work

The algorithm set forth in this thesis has the ability to be extended beyond two resources. Although all of the experiments conducted in this thesis only take into account CPU utilization and RAM utilization, there is no inherent reason as to why the new vector-based approach should be limited to only two resources. The main component of the algorithm, that is utilizing the magnitude of the rejection to order possible target hosts, can definitely be extended to three dimensions and beyond [15]. The linear algebra concept of finding projections and rejections is not limited in any way to two dimensions and can be visually represented in three dimensions by placing resource vectors on a 3 dimensional coordinate system. Furthermore, n-dimensional vectors may be used, although it might be difficult to visualize the concept in higher dimensions [15]. Not only can any aspect of the algorithm that deals with vectors be extended to higher dimensions, but the entire theta region construct may be extended as well. In two dimensions, the theta region takes on the appearance of an isosceles triangle however all that is needed to translate the construct into three dimensions is to represent the theta region as a conic section. Figure 12 demonstrates what the approximate visual representation of the new vector-based approach would look like if the algorithm was extended into three dimensions. The conic section has been truncated and hollowed out so that one may see the host vectors (here represented by red spheres) that would be excluded as possible target hosts. If one were to imagine if the maw of the conic section were to be extended to the extent to the coordinate system, it would more accurately demonstrate the theta region concept; however the need to show some spheres being engulfed by the conic section was thought to be paramount to explaining the concept. Clearly, three resources could be managed quite adequately if one were to implement a three dimensional version of the new vector-based algorithm.

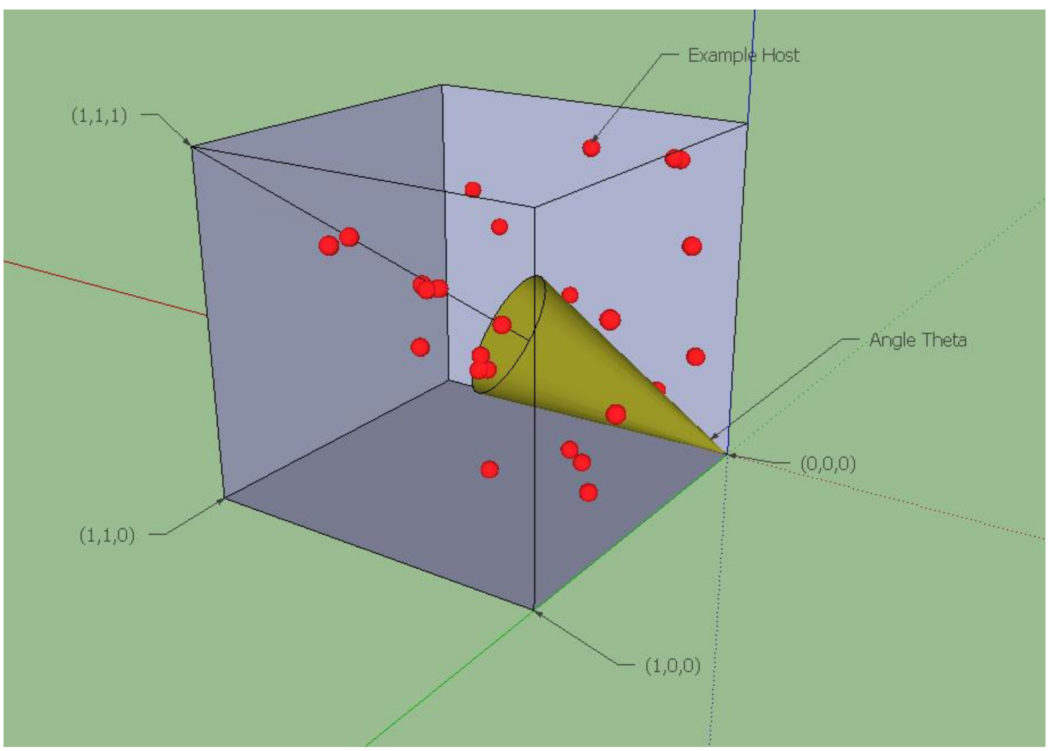

Figure 13 – The New Vector-Based Algorithm in 3D

The process through which appropriate theta values and ratio values are found could be automated in the future. Strictly speaking, the theta values and slope values for the balance line are parameters passed to the new vector-based algorithm. However, if one were to actually implement the algorithm, it would befit the owners of the data centre to automate the process through which those values are found. The experiments conducted in this thesis utilized values that were found experimentally through manual testing. Additionally, different subsections of a data centre could be preprogrammed to have static values for their theta regions and slope values. Furthermore, if applications on virtual machines could come with some sort of resource utilization statistic, this information could be used to further fine tune the data centre management policy automatically. This could obviate the need for finding those values every time a new set of applications was set to run on a data centre. In summary, a logical addition to the new vector-based approach would be a more efficient way of finding suitable values for the theta region and the balance vector slope.

Lastly, it has been suggested that the ordering criteria for the new vector-based approach could be improved upon. The criteria, as mentioned before, are simply a measure of the magnitude of the rejection from the host's resource vector to the balance vector. Hosts are chosen based on how small the rejection would be after the proposed arrangement of the virtual machine in question was calculated. This method was chosen because it most accurately reflected the way vector-based approaches were implemented in the past especially as seen in [21]. However, it may prove useful to use the target host that shows the greatest improvement through receiving the virtual machine in question. That is, the most suitable host is no longer the one that becomes least off balance but rather the one that showed the most improvement. It is not immediately clear if this change in the algorithm would result in significant gains in performance however it is an interesting addendum to the algorithm as it should have the tendency to select more off balance hosts more often. The downside is that it may not produce more balanced hosts at the end of an arrangement. The fact that this proposed change in the algorithm presents a non-trivial area on which to improve upon the algorithm suggests that it is an ideal area for future research. All in all, changing the ordering criteria of the new vector-based approach is most definitely an area for future research.

5.2 Conclusions

A series of conclusions could be drawn from the set of experiments conducted in this thesis. The first conclusion being that the balance vector's slope does indeed impact the successfulness of the algorithm as a whole. It was shown in experiment one that varying the slope ratio parameter's value can have dramatic effects on the number of hosts needed to satisfy all of the virtual machines in the simulation. The number of hosts required varied by a substantial amount for both the vector-based approach without a theta region, and the vector-based approach with a theta region. Minimizing the number of hosts required to run the simulation greatly decreases power consumption and therefore operating costs, so long as service level agreements are not violated, and therefore any method that achieves this should be considered when designing a real world data centre [5] [4] [19]. It was also shown that a slope that represents equal utilization across all resources does not necessarily produce the best virtual machine arrangement. This was best demonstrated when a ratio of 1.00:1.10 was shown to perform the best. This is important as literature surrounding vector-based approaches exclusively uses balance vectors that promote precise equality among all resources [15]. In summation, the varying slope values used proved important as they demonstrated that they can severely impact the overall success of the algorithm, and it should not be taken as fact that equal resource utilization is desirable.

Additionally, it was shown in experiment one that the concept of a theta region can favourably impact the efficiency of vector-based approaches although experiment one did not produce statistically significant results. When compared to vector-based approaches without a theta region, the vector-based algorithm with a theta region equipped often used less hosts to complete its virtual machine arrangement. It may be concluded that vector-based approaches may, in the future, benefit from the use of some such construct to minimize the cardinality of the set of target hosts. This in turn will lead to less hosts used overall and result in lower power consumption and therefore operating costs, so long as service level agreements are not violated, and therefore any method that achieves this should be considered when designing a real world data centre [5] [4] [19]. All in all, it is clear to see that the novel contribution of the theta region does indeed improve vector-based approaches to the virtual machine arrangement problem however this improvement is not statistically significant.

Moreover, it was seen in experiment two that using the novel contributions in tandem will provide more opportunities for a successful implementation. The new vector-based approach relies on two parameters, namely the balance vector slope and the theta region value. It was noted in experiment two that there are a wide range of possible, suitable values such that this component to the process should not be looked at as a hindrance to implementing the new vector-based approach. It was observed that different combinations of theta values and slope values produced approximately the same benefits. This should assuage any thoughts of shying away from using the new vector-based approach for fear of having to devote effort to finding such suitable values. The ease with which suitable parameters are found for the new vector-based algorithm should be taken as an impetus to utilize the algorithm in real world data centres.

Lastly, in experiment three, the new vector-based approach was shown to outperform other data centre management policies but the results were not all statistically significant. The other data centre management policies ranged in complexity from one that only took into account one of the resources, to random arrangement of virtual machines, to policies that ordered hosts based on an arbitrary binary operation performed on the hosts' resource utilization levels. It should be noted that all of the policies that were compared to the new vector-based approach were ones that were used in the literature, and some are in fact routinely used to benchmark the success or failure of other data centre management policies [21]. Therefore, by using fewer hosts in a data centre to run the same simulation, the new vector-based approach was shown to be more efficient than those policies that came before it however these results were not statistically significant. Yes, the new vector-based approach utilized fewer hosts on average, while still adhering within reason to service level agreements, thereby proving it to be an effective data centre management policy given the environment. In summation, the new vector-based approach outlined in this thesis is most definitely a policy that can rival the performance of its real world counterparts.

The new vector-based approach is a data centre management policy that has been proven to be effective through its use of varying the balance vector’s slope, the inclusion of the theta region, the ease with which said parameters could be found, and direct comparisons to other policies. However the results of the various experiments were not statistically significant. By varying the balance vector’s slope it was concluded that said variable plays an important role in vector-based management policies and that an equal ratio is not necessarily the optimal one. Additionally, the inclusion of the theta region to reduce the size of the target host set proved to have a profound impact by allowing the vector-based approach with a theta region to outperform the vector-based approach without a theta region with respect to the number of hosts used. Furthermore, the ease with which slope values and theta values could be combined to produce suitable pairings spoke to the new vector-based approach’s ease of use. Lastly, when faced with direct competition from various virtual machine arrangement policies, the new vector-based approach performed best with respect to the number of hosts used, all while adhering to service level agreements within a reasonable margin. It should be noted that this result was not statistically significant. In conclusion, the new vector-based approach and its novel components have been proven effective and should be considered for future research as well as real world implementation.

References

[1] X. Zhu, D. Young, B. J. Watson, Z. Wang, J. Rolia, S. Singhal, B. McKee, C. Hyser, D. Gmach, R. Gardner, T. Christian and L. Cherkasova, "1000 islands: an integrated approach to resource management for virtualized data centers," *Cluster Computing,* vol. 12, no. 1, pp. 45-57, 2009.

[2] G. Foster, G. Keller, M. Tighe, H. Lutfiyya and M. Bauer, "The Right Tool for the Job: Switching Data Centre Management Strategies at Runtime," in *IFIP/IEEE International Symposium on Integrated Network Management*, 2013.

[3] M. Tighe, G. Keller, M. Bauer and H. Lutfiyya, "DCSim: A Data Centre Simulation Tool for Evaluating Dynamic Virtualized Resource Management," in *6th International DMTF Academic Alliance Workshop on Systems and Virtualization Management*, 2012.

[4] A. Beloglazov and R. Buyya, "Optimal Online Deterministic Algorithms and Adaptive Heuristics for Energy and Performance Efficient Dynamic Consolidation of Virtual Machines in Cloud Data Centers," *Concurrency and Computation: Practice and Experience,* vol. 24, no. 13, pp. 1397-1420, 2012.

[5] A. Beloglazov, J. Abawajy and R. Buyya, "Energy-aware resource allocation heuristics for efficient management of data centers for Cloud computing," *Future Generation Computer Systems,* vol. 28, no. 5, pp. 755-768, 2012.

[6] C. Tang, M. Steinder, M. Spreitzer and G. Pacifici, "A Scalable Application Placement Controller for Enterprise Data Centers," in *Proceedings of the 16th internatnional conference on World Wide Web*, 2007.

[7] M. Andreolini, S. Casolari, M. Colajanni and M. Messori, "Dynamic load management of virtual machines in a cloud architectures," *Cloud Computing,* pp. 201-214, 2010.

[8] S. A. Baset, L. Wang and C. Tang, "Towards an understanding of oversubscription in cloud," in *2nd USENIX conference on Hot Topics in Management of Internet, Cloud, and Enterprise Networks and Services*, 2012.

[9] E. Arzuaga and D. R. Kaeli, "Quantifying Load Imbalance on Virtualized Enterprise Servers," in *Proceedings of the first joint WOSP/SIPEW international conference on Performance engineering*, 2010.

[10] H. Hlavacs and T. Treutner, "Genetic Algorithms for Energy Efficient Virtualized Data Centers," in *International Conference on Network and Service Management (CNSM)*, 2012.

[11] L. Wang, R. A. Hosn and C. Tang, "Remediating Overload in Over-subscribed Computing Environments," in *IEEE 5th International Conference on Cloud Computing (CLOUD)*, 2012.

[12] B. Simmons and H. Lutfiyya, "Strategy-trees: A feedback based approach to policy management," *Modelling Autonomic Communications Environments*, pp. 26-37, 2008.

[13] D. Bonde, *Techniques for Virtual Machine Placement in Clouds*, 2010.

[14] M. Stillwell, D. Schanzenbach, F. Vivien and H. Casanova, "Resource Allocation Algorithms for Virtualized Service Hosting Platforms," *Parallel and Distributed Computing*, vol. 70, no. 9, pp. 962-974, 2010.

[15] R. Panigrahy, K. Talwar, L. Uyeda and U. Wieder, "Heuristics for Vector Bin Packing," research.microsoft.com, 2011.

[16] A. Lodi, S. Martello and M. Monaci, "Two-dimensional packing problems: A survey," *European Journal of Operational Research*, pp. 241-252, 2002.

[17] N. Bansal, A. Lodi and M. Sviridenko, "A Tale of Two Dimensional Bin Packing".

[18] G. Keller, M. Tighe, H. Lutfiyya and M. Bauer, "An Analysis of First Fit Heuristics for the Virtual Machine Relocation Problem," in *6th International DMTF Academic Alliance Workshop on Systems and Virtualization Management (SVM)*, 2012.

[19] A. Beloglazov and R. Buyya, "Energy Efficient Resource Management in Virtualized Cloud Data Centers," in *Proceedings of the 2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing*, 2010.

[20] A. Lodi, "Algorithms for Two-Dimensional Bin Packing and Assignment Problems".

[21] M. Mishra and A. Sahoo, "On Theory of VM Placement: Anomalies in Existing Methodologies and Their Mitigation Using a Novel Vector Based Approach," in *IEEE 4th International Conference on Cloud Computing*, 2011.

[22] X. Zhang, Z.-Y. Shae, S. Zheng and H. Jamjoom, "Virtual Machine Migration in an Over-committed Cloud," in *IEEE Network Operations and Management Symposium (NOMS)*, 2012.

[23] N. Bobroff, A. Kochut and K. Beaty, "Dynamic Placement of Virtual Machines for Managing SLA Violations," in *International Symposium on Integrated Network Management*, 2007.

[24] A. Pokluda, G. Keller and H. Lutfiyya, "Managing Dynamic Memory Allocations in a Cloud through Golondrina," in *4th International DMTF Academic Alliance Workshop on Systems and Virtualization Management (SVM)*, 2010.

[25] L. Xu, W. Chen and Z. Y. S. Wang, "Smart-DRS: A Strategy of Dynamic Resource Scheduling in Cloud Data Center," in *IEEE International Conference on Cluster Computing Workshops*, 2012.

[26] S. Upendra, P. Shenoy, S. Sahu and A. Shaikh, "Kingfisher: A System for Elastic Cost-aware Provisioning in the Cloud," Dept. of CS, UMASS, Tech. Rep. UM-CS-2010-005, 2010.

[27] S. Chaisiri, B.-S. Lee and D. Niyato, "Optimal virtual machine placement across multiple cloud providers," in *Services Computing Conference, 2009. APSCC 2009. IEEE Asia-Pacific*, 2009.

[28] N. Roy, A. Dubey and A. Gokhale, "Efficient autoscaling in the cloud using predictive models for workload forecasting," in *Cloud Computing (CLOUD), 2011 IEEE International Conference on*, 2011.

[29] D. Breitgand and A. Epstein, "Improving Consolidation of Virtual Machines with Risk-aware Bandwidth Oversubscription in Compute Clouds," in *INFOCOM*, 2012.

[30] H. Nakada, T. Hirofuchi, H. Ogawa and S. Itoh, "Toward virtual machine packing optimization based on genetic algorithm," *Distributed Computing, Artificial Intelligence, Bioinformatics, Soft Computing, and Ambient Assisted Living,* pp. 651-654, 2009.

[31] J. Hu, G. Jianhua, G. Sun and Z. Tianhai, "A scheduling strategy on load balancing of virtual machine resources in cloud computing environment," *Parallel Architectures, Algorithms and Programming (PAAP), 2010 Third International Symposium on,* pp. 89-96, 2010.

[32] J. Strebel and A. Stage, "An economic decision model for business software application deployment on hybrid Cloud environments," *Multikonferenz Wirtschaftsinformatik,* p. 47, 2010.

[33] S. A. Wolfman and D. S. Weld, "Combining linear programming and satisfiability solving for resource planning," *The Knowledge Engineering Review,* vol. 16, no. 01,

pp. 85-99, 2001.

[34] W. Meng, X. Meng and L. Zhang, "Consolidating Virtual Machines with Dynamic Bandwidth Demand in Data Centers," in *INFOCOM*, 2011.

[35] Y. Minyi, "A simple proof of the inequality FFD (L) ≤ 11/9 OPT (L) + 1, ∀ L for the FFD bin-packing algorithm," *Acta Mathematicae Applicatae Sinica (English Series),* vol. 7, pp. 321-331, 1991.

Appendices

Appendix A - Statistical Analysis

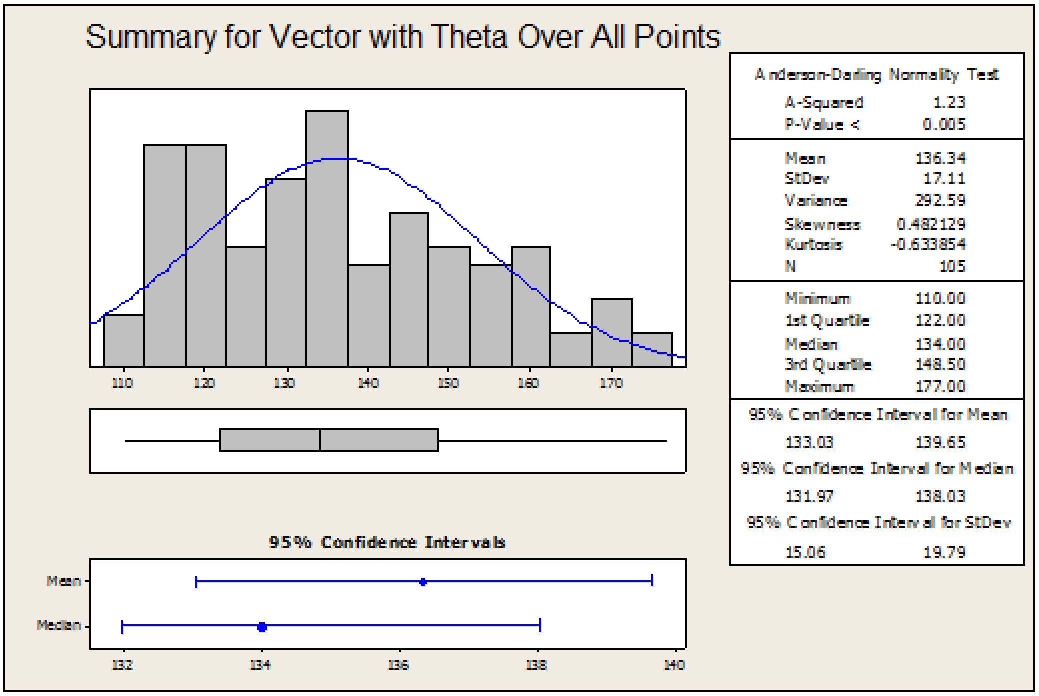

Figure A1 – Experiment 1 Results, Vector with Theta Policy

Figure A1 illustrates the data points and statistical analysis for the vector with theta region policy found in experiment 1. The mean value of the number of hosts utilized was found to be 136.34 hosts. The median value was found to be 134 hosts. The standard deviation was found to be 17.11 hosts. The mean and standard deviation were used to check for statistical significance as shown in figures A3 and A4.

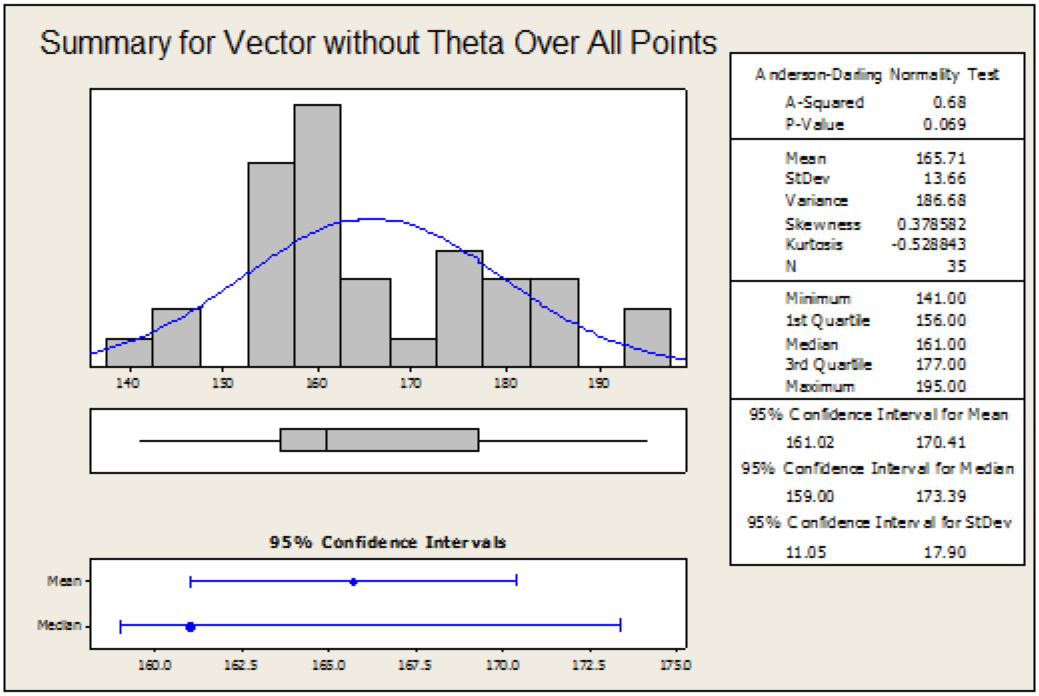

Figure A2 – Experiment 1 Results, Vector without Theta Policy

Figure A2 illustrates the data points and statistical analysis for the vector without theta region policy found in experiment 1. The mean value of the number of hosts utilized was found to be 165.71 hosts. The median value was found to be approximately 161 hosts. The standard deviation was found to be 13.66 hosts. The mean and standard deviation were used to check for statistical significance as shown in figures A3 and A4.

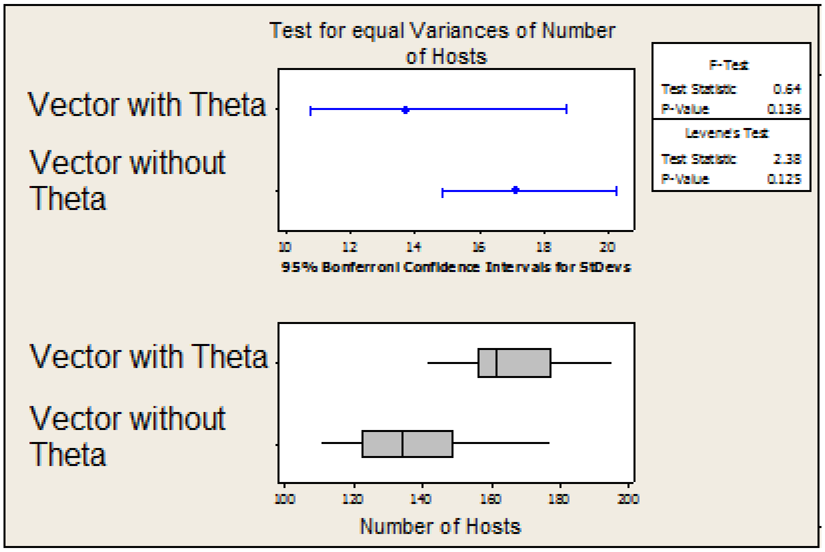

Figure A3 – Experiment 1 Results, Policy Variance Comparison

Figure A3 illustrates the statistical insignificance between the performances of the vector with theta region policy and the vector without theta region policy found in experiment 1. The purpose of this test was to determine if there was a statistically significant improvement due to the use of the theta region construct. Although the vector with theta region policy utilized a fewer number of hosts on average when compared to the vector without theta region policy, the fact that both policies' means are within one standard deviation of each other proves that this is a statistically insignificant improvement. Furthermore, statistical tests yielded values of $p > 0.05$ which is traditionally taken to mean that an experiment's results are statistically insignificant.

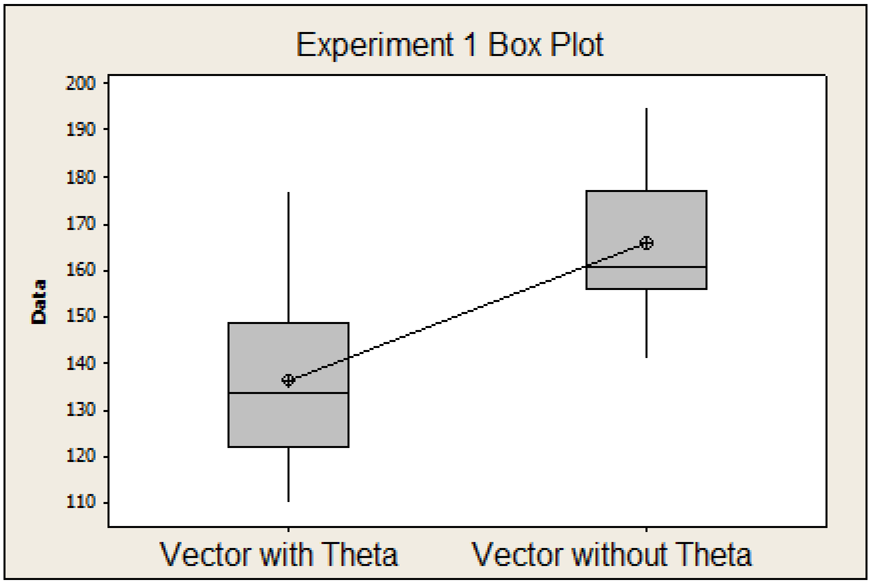

Figure A4 – Experiment 1 Results, Box Plot

Figure A4 illustrates the statistical insignificance between the performances of the vector with theta region policy and the vector without theta region policy in experiment 1. Once again, the purpose of this test was to determine if there was a statistically significant improvement due to the use of the theta region construct. Although the vector with theta region policy utilized a fewer number of hosts on average, when compared to the vector without theta region policy, the fact that both policies' means are within one standard deviation of each other proves that this is a statistically insignificant improvement. This is evident when one inspects the whiskers in the above box and whisker plot. Due to the fact that the whiskers of each policy overlap, the results from experiment 1 can be concluded to be statistically insignificant.

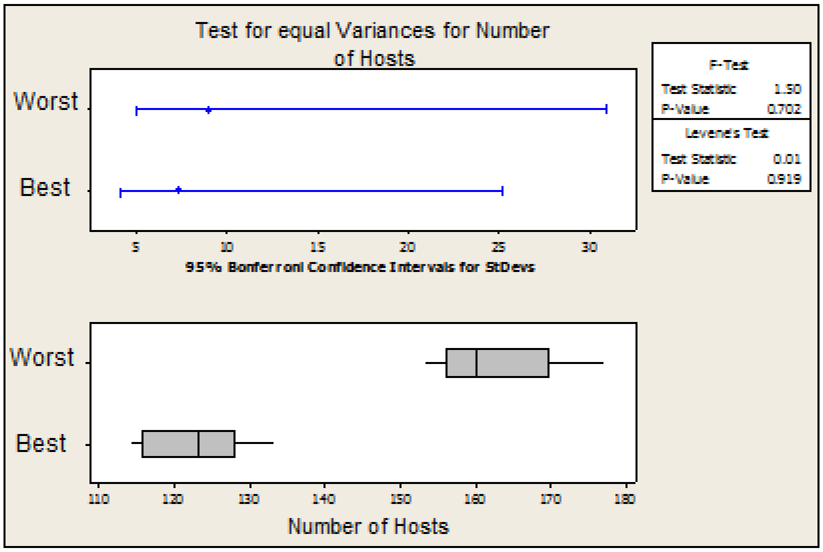

Figure A5 – Experiment 1 Results, Best Performing Vector with Theta and Worst Performing Vector with Theta Test for Equal Variance

Figure A5 illustrates the statistical insignificance between the performances of the vector with theta region policy's best performing run and the vector with theta region policy's worst performing run in experiment 1. The purpose of this test was to determine if there was a statistically significant improvement when varying the value of the balance vector's slope. Specifically, the purpose was to see if there was a statistically significant improvement over the worst performing slope value when compared to the best performing slope value. Although the vector with theta region policy's best performing slope value utilized fewer hosts on average when compared to the vector with theta region policy's worst performing slope value, statistical tests yielded values of $p > 0.05$ which is traditionally taken to mean that an experiment's results are statistically insignificant.

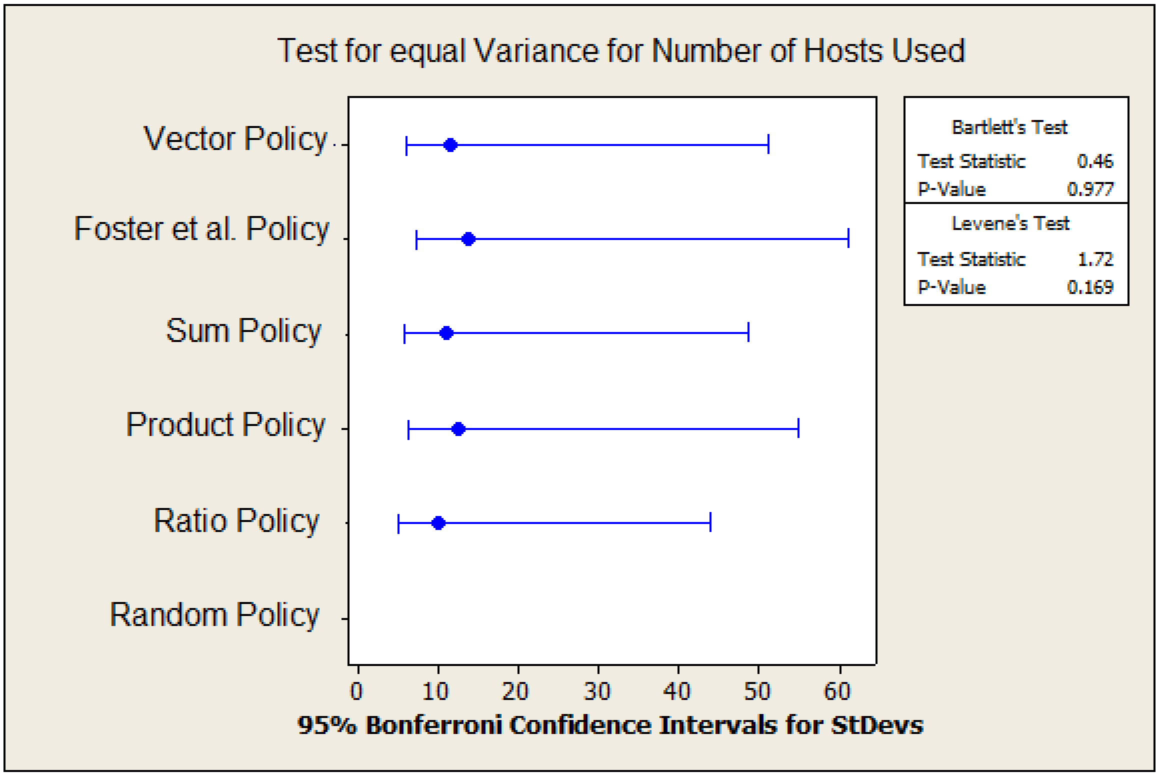

Figure A6 – Experiment 3 Results, Test for equal Variances for Number of Hosts Used

Figure A6 illustrates the statistical insignificance between the performances of the new vector-based policy, the Foster et al. hybrid policy, the sum policy, the product policy and the ratio policy. The purpose of this test was to determine if there was a statistically significant improvement when using the new vector-based policy to construct virtual machine arrangements when compared to the other, aforementioned policies. Although the new vector-based policy utilized fewer hosts on average when compared to the other policies, statistical tests yielded values of $p > 0.05$ which is traditionally taken to mean that an experiment's results are statistically insignificant. Use of the new vector-based policy to construct virtual machine arrangements does not result in statistically significant improvements with respect to the number of hosts used when compared to the other polices mentioned.

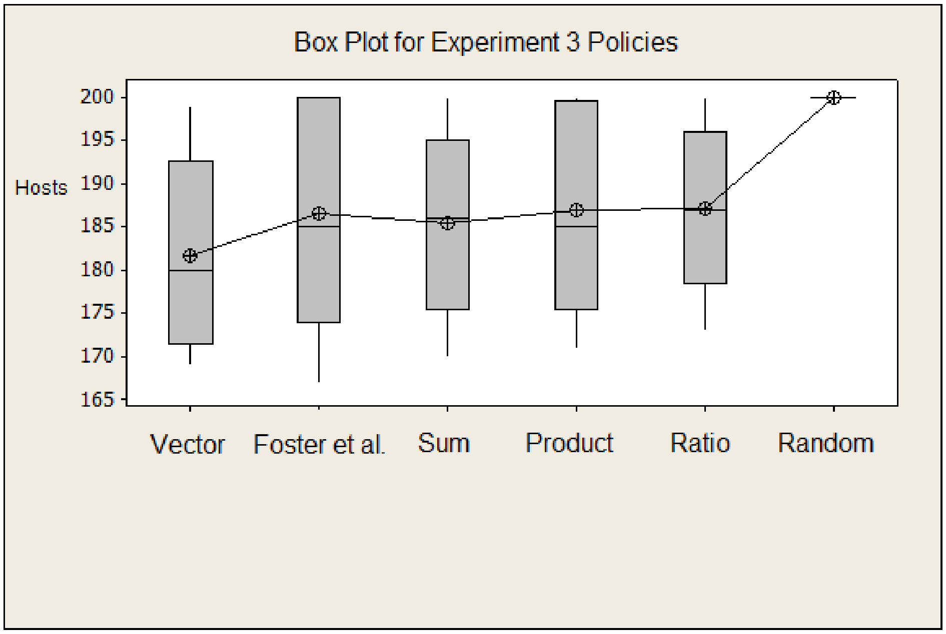

Figure A7 – Experiment 3 Results, Box Plot

Figure A7 illustrates the statistical insignificance between the performances of the new vector-based policy, the Foster et al. hybrid policy, the sum policy, the product policy and the ratio policy. The purpose of this test was to determine if there was a statistically significant improvement when using the new vector-based policy to construct virtual machine arrangements when compared to the other, aforementioned policies. Although the new vector-based policy utilized fewer hosts on average when compared to the policies, the fact that the aforementioned policies' means are within one standard deviation of each other proves that this is a statistically insignificant improvement. This is evident when one inspects the whiskers in the above box and whisker plot. Due to the fact that the whiskers of each policy overlap, the results from experiment 3 can be concluded to be statistically insignificant. Use of the new vector-based policy to construct virtual machine arrangements does not result in statistically significant improvements with respect to the number of hosts used. However, it should be noted that the above statistical analysis did not discount the new vector-based policy from being a statistically significant improvement over random placement. This was further analyzed in figure A8.

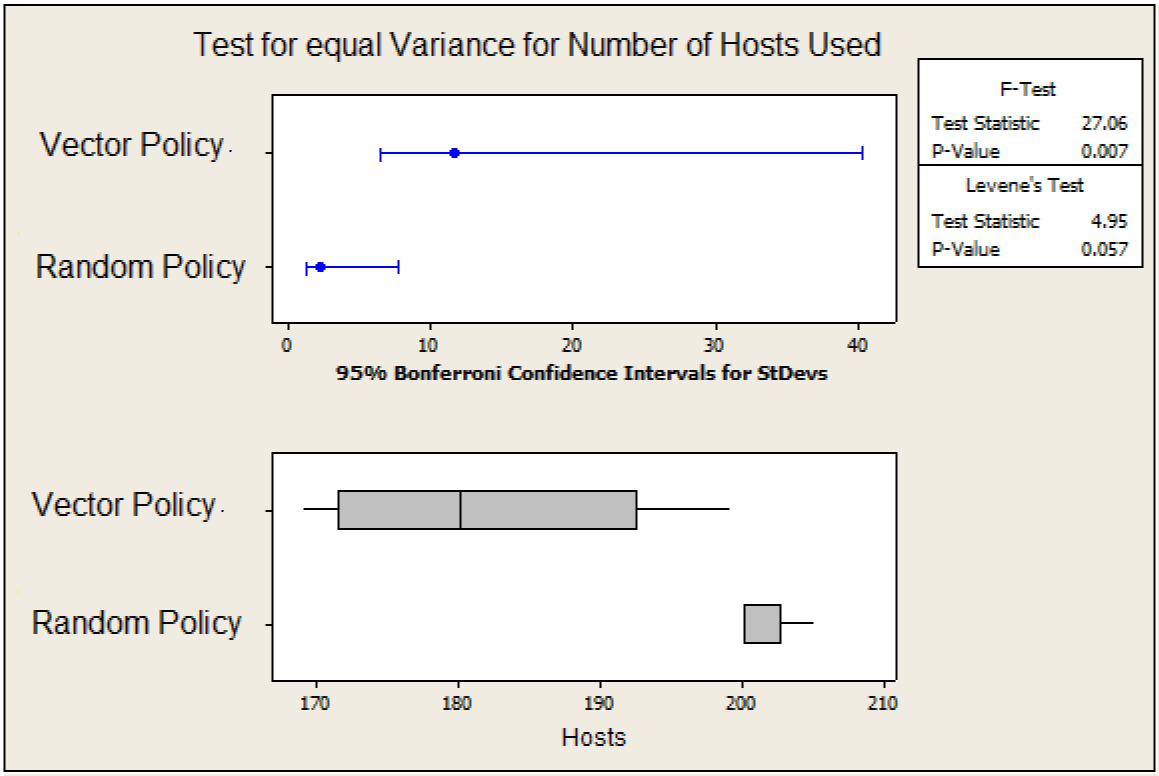

Figure A8 – Experiment 3 Results, Test for equal Variances between New Vector Policy and Random Policy Only

Figure A8 illustrates the statistical significance between the performances of the new vector-based policy, and random placement. The purpose of this test was to determine if there was a statistically significant improvement when using the new vector-based policy to construct virtual machine arrangements when compared to random placement. The new vector-based policy utilized fewer hosts on average when compared to randomly generated virtual machine arrangements. Statistical tests yielded values of $p < 0.05$ which is traditionally taken to mean that an experiment's results are statistically significant. Use of the new vector-based policy to construct virtual machine arrangements results in statistically significant improvements with respect to the number of hosts used when compared to random virtual machine arrangements.

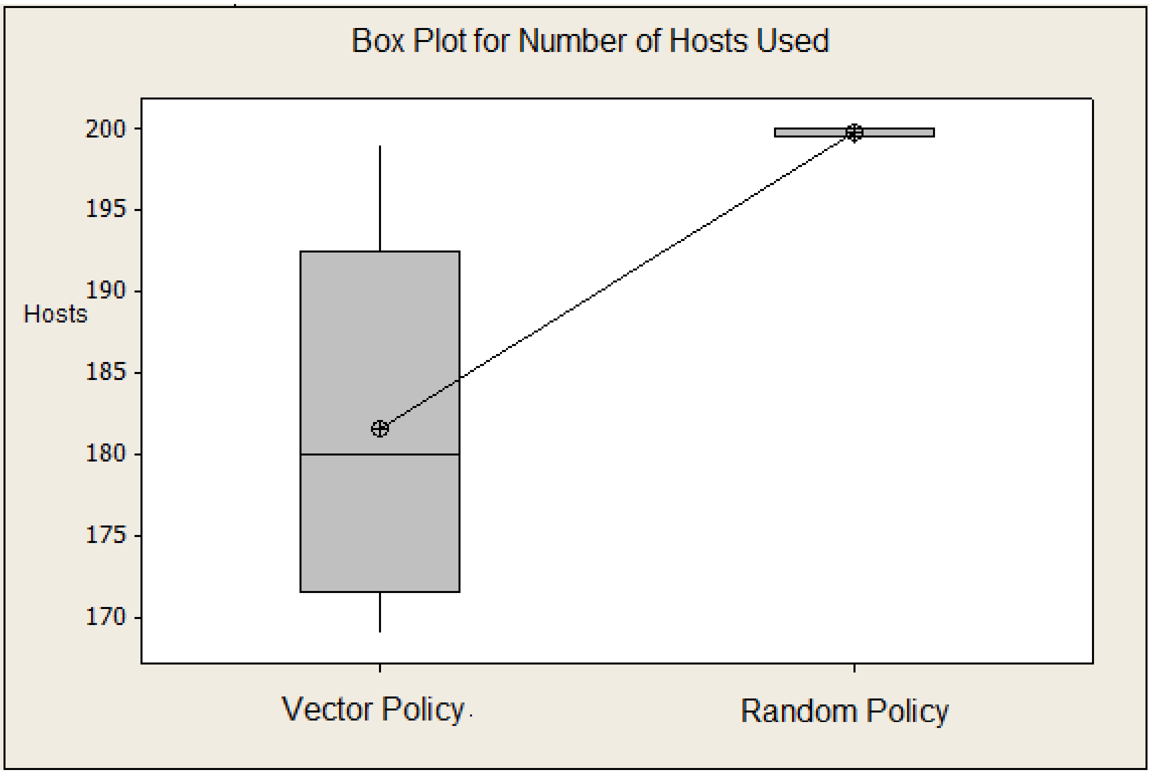

Figure A9 – Experiment 3 Results, Box Plot for New Vector Policy and Random Policy Only

Figure A9 illustrates the statistical significance between the performances of the new vector-based policy, and random placement. The purpose of this test was to determine if there was a statistically significant improvement when using the new vector-based policy to construct virtual machine arrangements when compared to random placement. The new vector-based policy utilized fewer hosts on average when compared to randomly generated virtual machine arrangements. This is evident when one inspects the whiskers in the above box and whisker plot. Due to the fact that the whiskers of each policy do not overlap, the results from this portion of experiment 3 can be concluded to be statistically significant. Use of the new vector-based policy to construct virtual machine arrangements results in statistically significant improvements with respect to the number of hosts used when compared to random virtual machine arrangements.

Curriculum Vitae

Name:	Nicholas Cerilli
Post-secondary Education and Degrees:	Laurentian University Sudbury, Ontario, Canada 2008-2012 B.Sc. Hons
	The University of Western Ontario London, Ontario, Canada 2012-2014 M.Sc.
Honours and Awards:	Governor General's Medal for Academic Excellence 2012
Related Work Experience	Teaching Assistant The University of Western Ontario 2012-2014