Scale up or Scale out: One solution that fits all problem sizes

Sriram Vadlamani

March 25, 2021

In this essay, we are going to go over the term scalability, the various types of scalability and effects of scalability on performance, more specifically, if there exists a universal solution to all problem sizes.

Keywords

Cloud computing, Scale up / Scale down, Scale out / Scale in, Load scalability, space and structural scalability.

1 Background

The term scalability refers to the measure of a computing environment's capability to increase or decrease in the performance and cost per the demands. Scaling up refers to adding more resources into an existing node and scaling out refers to building more nodes in order to catch up with the computing demands.

Scalability for a computing environment is essential today with the growing need of data analytics. Large scale cloud systems and big data analytics are often used practically today, with dynamicity of massive user requests which can result in poor utilization of resources and vulnerable system dependability[2]. An effective methodology to scaling while utilising resources effectively should be introduced.

This essay will look at some terminology and existing research on the specified problem to come to a conclusion on the abstract nature of the problem.

2 Introduction

2.1 Terminology

Scale up and Scale down
The idea of scaling up is to add more resources into an existing node or a computing environment. Scaling down being the contrary i.e., removing resources from an existing node.

Scale out and scale in
Scaling out refers to adding new nodes into an existing environment and 'scaling out' capacity i.e., expanding it and 'scaling in' removes a few nodes from an existing environment.

Cloud computing
Cloud computing is the practice of using remote nodes or a computing environment to store, process and manage data from elsewhere.

Load scalability
We say that a system is load scalable if it has the ability to function gracefully i.e., without undue delay and unproductive resource consumption at light, moderate or heavy loads while making good use of available resources[3].

Space and structural scalability
We say that a system or an application is regarded as having space scalability if it's memory requirements do not go to intolerable levels as the number of items it supports grows higher[3]. And we think of a system as structurally scalable if it's implementation does not impede with the growth of number of objects it encompasses, atleast not within the given time frame[3].

2.2 An ideal system

We can easily see from the above definitions that an ideal scalable system satisfies all the above mentioned characteristics. It is trivial to see that this is not possible. A very good analogy would be ideal gases in

thermodynamics, they are known as ideal as they cannot exist but we model our studies based on them in order to study properties of existing systems and possible make them more efficient (The carnot heat engine for example). The goal would be to eventually model and study existing systems or applications based on the given definitions.

3 Overview and challenges faced with massive scale computing

[A]In an environment it is common to have users with varying requests i.e., with different computation purposes co-exist with diverse resource requirements and patterns. Massive-scale systems are typically composed by hundreds of thousands to millions of alive and interacting components comprised by the resource manager, service framework and computational applications. With increasing scale of a cluster, the probability of hardware failures also arises[2].

[B]Another problem that can be faced is if incoming requests are not handled timely by the resource manager, The requests may aggregate. Making prompt scheduling decisions at such a fast rate means that the resource allocation must realize a mapping of the CPU, memory and other desirable machine resource to all tasks within every decision making[2].

[C]Scheduling tasks and message scalability is another challenge as internal scheduling requests are backed by periodical interactive messages. Keeping the rate of requests constant is a tedious task as having a long period for a message might reduce the load but it can aggregate requests.

4 Effective scalability and adaptive workloads

As a computer science student, managing memory and resources is a vital task. During a college project where an implementation of malloc (memory allocator in C) was asked to be done, it was clear to me that by using some certain memory and building a wrapper around it (the job of malloc) is the most efficient way. The inital allocation of a memory is for a job or a process in the program. In a similar way, by making this analogy more abstract, we can see that scalability of nodes is no different.

If a Job A has to be expanded - we can submit a new job B which depends on A which requests a certain number of nodes N_b to be added to the initial job. Once these nodes are allocated, update the nodes for the job B to zero and cancel the job B and add the previously allocated new nodes to the inital job and update it's size to $N_a + N_b$ [1].

Data driven methodology

Analysis and simulation of cloud tasks and data can prove to be helpful for both end users and providers equally. It can help ensure resource management mechanisms. As an example, we can exploit the heterogeneous nature of the jobs at hand to minimize performance interference of physical equipment like servers, analyze the relation between failures to reduce resource consumption to come to a global solution[2].

5 Observations

We have seen a few challenges that can be faced that reduce efficiency of a computing environment and how scaling up or scaling out can have their own drawbacks. However it is clear that scaling out has an edge on efficiency and has the versatility to develop algorithms to help minimize the risk of misuse of resources in a node or several nodes. Nonetheless, we can categorize problems and study them by using data driven methodologies, i.e., analyzing the workload data and finding relations between the problems.

It is clear to see that there is no universal solution to a problem as scaling up can prove to be quite useful in some cases. For example, when the known end users are fixed i.e., need a specific set of resources and their requests grow up linearly. However, scaling out should be given a preference as it has potential for optimization and there is a limit to buying more powerful hardware, and it can take a toll on the expenditure.

Scaling out doesn't depend on resources of an individual nodes, rather the combined power of existing nodes and by adding homogeneous nodes, the computing power helps deliver more efficiently, but it may result in a more complex architecture of the environment and can get hard to manage.

6 Conclusion

Scaling up is very simple compared to scaling out, easier to maintain whereas scaling out has a more complex architecture, and resource management but can have efficiently working algorithms and keeps the cost low as it doesn't depend on the computing capability of individual nodes,(which can stagnate after buying the most powerful hardware) rather the combined capacity of several nodes which can be cost effective. Choosing a scaling strategy has to be a business decision and a method that can be beneficial to the end user. Although it is recommended to scale out first, as it is cost effective and then implement a few working algorithms to optimize the resource scheduling and consumption, along with some analysis on the data to find recurring relations in the problems / challenges faced with the current environment.

References

[1] Efficient Scalable Computing through Flexible Applications and Adaptive Workloads Sergio Iserte, Rafael Mayo, Enrique S.Quintana - Orti. Universitat Jaume I.

[2] Computing at Massive Scale: Scalability and Dependability Challenges Renyu Yang and Jie Xu, School of computing, Beihang university, Beijing, China.

[3] Characteristics of Scalability and Their Impact on Performance André B. Bondi, AT and T labs, Network Design and Performance Analysis Department.