How should we mix scale-up and scale-out strategies

Kazuho Fujii

May. 16, 2015

1 Background

For enhancing computing power we have two approach: scale-up strategy and scale-out strategy. Under the scale-up strategy we buy a larger size computer when we want more performance. On the other hand, under the scale-out strategy we buy more number of computers in the case.

Today the scale-out strategy is successful, e.g. Hadoop: a distributed computing framework, Ceph: a distributed file system. This is because scale-out strategy is more cost-effective than scale-up strategy. When scaling up machine the performance does not grow linearly to the cost because of the law of diminishing returns. But, we can increase the computing performance linearly with scale-out architecture system.

However, scale-out strategy has a limit. This is well known as Amdahl's law: the performance of parallel computing system can not exceed a value however large number of machine added.

Therefore we should mix scale-up and scale-out strategy. The question here is how should we mix them? What size of machine we should buy? In this note, I think about it.

2 Scale-up strategy

For the first step, I model scale-up and scale-out system arithmetically and consider about nature of them. In this section I think about scale-up strategy.

I assume that computing performance of machine in scale-up strategy is obeyed by Cobb-Douglas production function:

$$Y = K^{\alpha}$$

. Where, Y is computing performance of a unit machine; K is price of a unit machine; α is output elasticity. Y and K are normalized by the values of a standard machine.

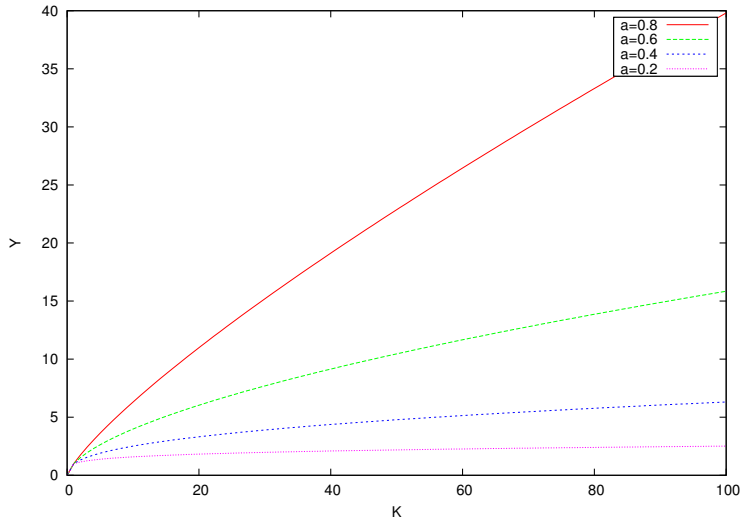

Fig. 1: Relationship between machine price and performance.

Y includes effects of total components of a computer: CPU, memory, network-device, disk, and so on. In the other words, I assume that performance of a machine is determined uniquely by its price. Or the configuration of a machine is already optimized.

Elasticity α, which is an economics term, is an external variable. It is not configurable. It is in the range of $0 < \alpha < 1$.

As seen in Fig.1, performance does not linearly grow with price. This nature is known as the law of diminishing returns. Performance increment by cost is higher in a small machines, but it is smaller in a large machine. If we want high computing performance with one machine we need a large amount of money. Therefore the other strategy: scale-out is gathering attention.

3 Scale-out strategy

The scale-out strategy is to parallelize many machines rather than increase the size of the machine. It is also called as distributed computing, grid computing or cluster computing.

I assume that performance of a computer cluster is obeyed by a modified Amdahl's law:

$$S = \frac{1 + C}{(1 - P) + PN^{-1} + CN}$$

. Where, S is computing performance of a cluster consisted of standard machines. S is

normalized by performance of a single machine environment. N is the number of nodes the cluster has; P is the proportion of a program that can be made parallel; C is penalty of clusterization per a machine.

When $C = 0$ this is the standard Amdahl's law. As seen in Fig.2, however large number of nodes added, the performance can not exceed the value $1/(1-P)$.

Penalty of clusterization means cost by having many machines: e.g. network traffic, loss by operation. It is not ignorable when number of machines is increased. When having this term, the performance of cluster does not saturate and gets limit at $N = \sqrt{P/C}$.

P and C depend on the system architecture. In the proper scale-out architecture, $C = 0$ and $P = 1$. In this case the cluster performance grows linearly. Therefore, the architecture is important. Great hackers have been developing good architectures, that can be used as open source software.

4 The best mix of scale-up and scale-out

Now I discuss about the case of mixing scale-up and scale-out. I think about creating a cluster, which has N nodes of K price machines. The total cost is

$$B = NK$$

.

Assuming the performance of a cluster is in proportion to the performance of machines, it is

$$\begin{aligned} Z &= YS \\ &= \frac{(1+C)B^{\alpha}}{(1-P)N^{\alpha} + PN^{\alpha-1} + CN^{\alpha+1}} \end{aligned}$$

. Z is the performance of a cluster. It gets the maximum value when

$$N = \frac{-a(1-P) + \sqrt{a^2(1-P)^2 + 4(1-a^2)CP}}{2(1+a)C} \ (=: N_e)$$

. When $C = 0$,

$$N_e = \frac{(1-a)P}{a(1-P)}$$

. This is the most effective number of nodes. It tell us how should we decide the size of machines for a cluster.

In non-scalable architecture: $P = 0$ or $C \to \infty$, the best number of nodes $N_e \to 0$ as seen in Fig.3. This means that only scale-up strategy has effect. We should use a higher performance machine in this case.

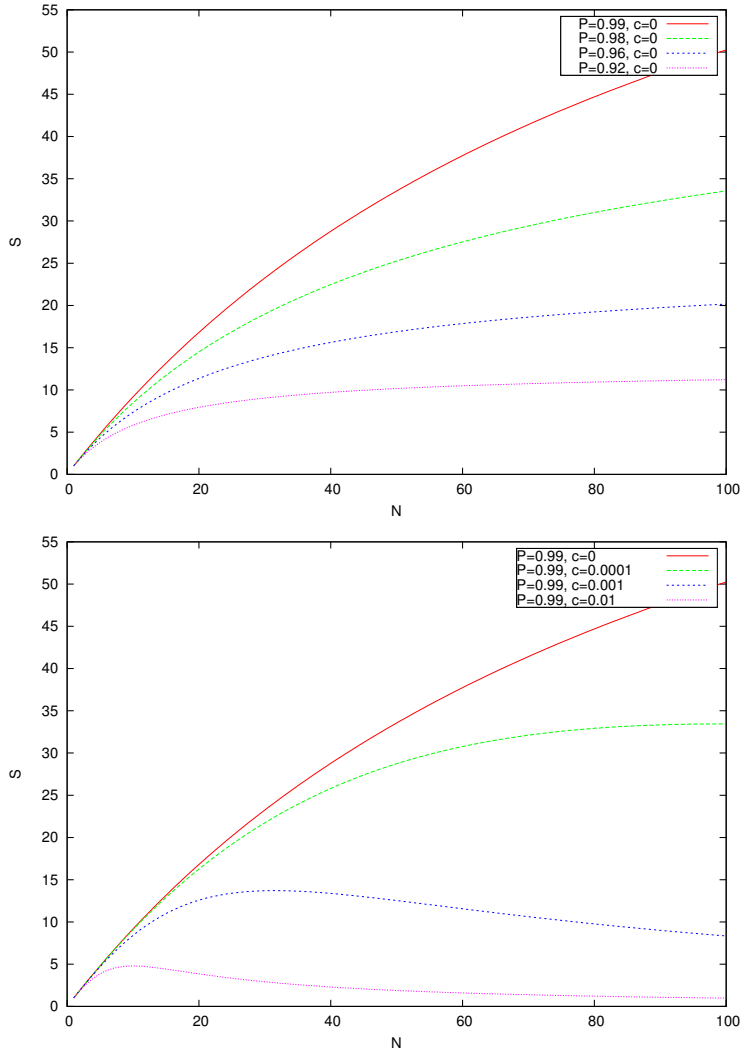

Fig. 2: Relationship between machine number and performance.

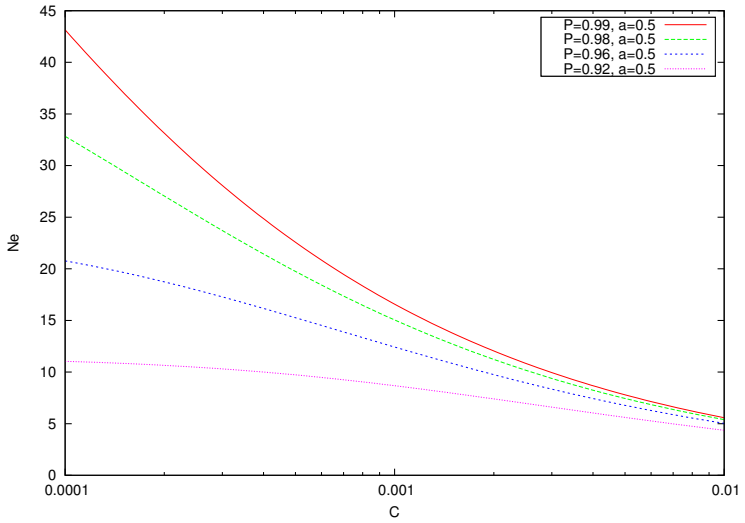

Fig. 3: The most effective number of nodes in cluster.

Otherwise in the proper scalable architecture: $P = 1$ and $C = 0$, $N_e \to \infty$. This means that only scale-out strategy has effect. We should use more and cheaper machines rather than less and richer machines.

When we get budget what machines should we buy? N_e does not depend on the budget B. It is determined just by system architecture and elasticity of machine performance. That is, we should not change the number of node with the budget, but we should change the machine size to buy. We should select machine size that we can buy N_e. However, with scale-out architecture, in almost case excepting the case we have ample budget, we can not get N_e machines even though we select the cheapest machine size. So we should create a cluster with the cheapest machine size. After we have gotten N_e nodes environment, we should scale up.

5 Conclusion

In this note, I derive the most effective number of nodes in a scale out cluster. The best way to mix scale-out and scale-up strategies is that we should select the size of machine which we can buy the most effective number.