LAMK
Lahden ammattikorkeakoulu
Lahti University of Applied Sciences

RENDERING SYSTEMS

Reducing render times using distributed systems

LAHTI UNIVERSITY OF APPLIED SCIENCES
Bachelor of Engineering, Information and Communications Technology
Media Technology
Spring 2019
Gerard Amorós Serra

Abstract

Author Amorós Serra, Gerard	Type of publication Bachelor’s thesis	Published Spring 2019
	Number of pages 44	

Title of publication

Rendering systems
Reducing render times using distributed systems

Name of Degree

Bachelor of Engineering, Information and Communications Technology

Abstract

The rendering process is a common bottleneck in businesses like animation studios. There are many ways to get around this problem, among them is a render farm.

In this study the factors affecting rendering are examined, from hardware and lighting effects to parallelization costs. To analyze this, a render farm with Autodesk’s Backburner is set up and multiple benchmarking tests are realized.

The purpose of this study is to better understand the render farm as a solution to long render times. Balancing their costs with its benefits and creating models to predict the speedup in real world scenarios.

How to set up a render farm and what effectiveness we can expect from it at different scale points. How to avoid bottlenecks and other problems and how to get high efficiencies. We found some guidelines on price points for a targeted speedup and when cloud computing options are better suited for a use case.

Specifically, small render farms from one to thirty machines are studied, how the length of rendering times affects the speedup and how higher resolutions impact rendering times. The concept of using the manager machine as a rendering machine for increased throughput is put to the test. It is found that long jobs achieve higher speedups and that small render farms can use the manager machine to render at the same time without a noticeable impact on the network.

The work focused around the parallelization costs and speedups, explaining what factors limit the speedup and discussing ways to minimize this. Finally, a brief introduction to the environmental impact of a render farm expressed in its carbon footprint is discussed.

Keywords

Rendering, Backburner, 3DS Max, distributed systems, render farm, Amdahl’s law, parallelization,

CONTENTS

LIST OF TERMINOLOGY

CPU: The Central Processing Unit is the brain of the computer that processes most of the data.

Frame: A frame is a still image in an animation. Usually animations have a set number of frames per second (fps) from 24 for cinema to 30 for American television.

GPU: The Graphics Processing Unit is a circuit that specializes in the generation of images or frames. It is generally much faster than the CPU for those tasks due to the high level of parallelization they have.

HDD: Hard Drive Disk, refers to the main storage unit of a machine, the technology used is a magnetic drive to store information.

IP address: An Internet Protocol address, used by computers as an identification needed to send and receive data.

RAID: Redundant Array of Independent Disks is a technology used to provide more secure and/or faster storage by aggregating several independent storage units.

RAM: Random Access Memory, it is a volatile memory in between processor cache and storage in the memory hierarchy.

Render: It is the act of generating images with a computer from predefined models.

SSD: Solid State Drive, used for storage like HDD but with faster data access speeds.

SAN: Storage Area Network, is a computer network used to provide access to one or more machines to a remote storage.

Turboboost: Turboboost is a feature in many modern CPUs that allows them to reach higher clock speeds than normal, in some cores, for a short period of time. Turboboost is the name Intel uses, while AMD calls it Turbo Core.

VRAM: Video-RAM, refers to the amount of memory a GPU has aiding its processing.

1 INTRODUCTION

Generating images by computer or rendering is a resource intensive process that often requires the use of multiple machines working together to complete the job in time. This means that the creation of these systems of connected computers is necessary for some businesses.

First, there is a short overview of the rendering process, what it is and why speed is important enough to justify the creation of big networks of machines for rendering. Then there is a review of the different factors affecting the rendering times, from hardware to resolution and parallelization.

After, a network of computers is built and used to test how the factors affect the real-world examples, and to understand the strengths and limitations of a distributed rendering system. The impact of higher resolutions is also tested as well as the possibility of using the manager machine as a rendering machine to increase performance.

Finally, the performance cost of parallelization and the economic cost of a system like the one used is analyzed. Example scenarios are provided of when a business would want to have their own render farms and in what cases its costs are not justified.

For benchmarking the following components are used: Rendering software (Redshift, Arnold, Quicksilver and ART), Autodesk’s 3DS Max and Autodesk’s Backburner as well as 31 machines from LAMK equipped with Intel Core i7-7700 and GeForce GTX 1080.

2 RENDER SYSTEMS

2.1 Rendering

Rendering is the process of generating a digital image from predefined digital models using a computer. Often, many of these images are produced to make an animation. This process is affected by many factors and generally takes considerable resources like time, processing power and energy.

The process is orchestrated by CPUs and is often assisted by GPUs. There are different ways the rendering process can be performed, like rasterization or ray tracing. For example, rasterization calculates a 3D environment using vector math and assigns different color values to the coordinates and generates a 2D image by mathematical calculations that have no base in how real-life light works and in fact casts no shadows or reflections.

More advanced techniques like ray tracing simulates the real world by calculating the path of a photon between the camera and an object and between the objects and light sources with light bouncing of objects according to the information in the materials. This creates a much more realistic image but is a slower process.

Different rendering engines use a variety of methods and algorithms that change resource requirements and quality of the final product. Usually the desired quality of a renderer can be adjusted in the settings before starting to render.

2.2 Importance of speed

The rendering process is of special importance in some work environments like an animation studio where several iterations of the final product will be made as the employees adjust the settings trying to improve the final product as much as possible.

In this and many other cases the time it takes to render those images can be crucial as it can bottleneck the creative process if, for example, the artist needs to wait long times to have a high-quality visualization of his work. For example, the frames in Cars 2 by Pixar seen in Figure 1 took an average of 11.5 hours to render (Terdiman, 2011).

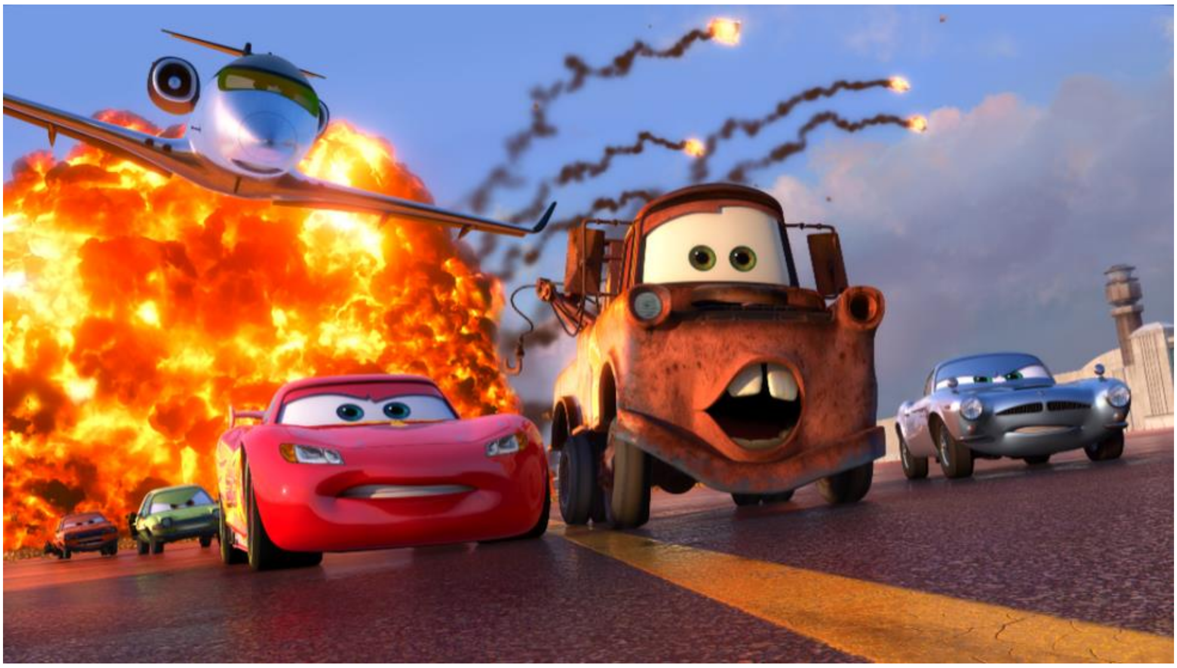


Figure 1 New technology revs up Pixar's 'Cars 2' (Terdiman 2011)

Many advances in rendering technology, both in hardware and software, have increased the speed dramatically. From faster clock speeds and core count to more efficient algorithms they are all crucial to achieve the speeds seen today.

But as higher video quality has been made available to the public, the demand for rendering capabilities has increased to the point that, for many professional usages, a single machine is not capable of rendering fast enough.

2.3 The solution: teamwork

Rendering being too slow with a single machine has been the case for many years, but the solution has remained the same: splitting the work so it can be done in parallel. Each frame can be split into “buckets”, so multi-threaded processors can work on multiple parts at the same time and splitting the work again even further by giving a set of frames to different machines.

By splitting the work, rendering time is cut down significantly. In our previous Cars 2 example a render farm of 12500 CPUs was used since the whole movie (106 minutes) would have taken just over 200 years to render in a single machine.

2.3.1 Render farms

The term render farm originated in 1990 during the production of The Bored Room, one of the first productions made with 3D Studio (later renamed to 3DS Max), where they had to render the frames on different computers and put them back together without an automated system. The person who gathered the frames from these machines, Jamie Clay, usually wore farmer overalls and the name render farm was created as an inside joke since he was “farming” the frames. (Clay 1990)

In the interest of faster rendering times, work can be split among different machines, usually giving each a set of frames and in the end stitching them all together. This work is focused on the usage of Autodesk Backburner, as it is a very relevant option used in professional environments but, there are different ways of splitting the task with other software. There are many proprietary options that require their own license or work in a specific rendering engine.

When networking computers, there is always some loss in efficiency due to the extra task of spreading the workload and slower data transmissions compared to working on a single machine. This is usually compensated by much faster rendering times, but it should be considered, if the higher speed is relevant enough to offset the loss in efficiency.

2.3.2 Cloud computing and outsourcing

Cloud computing is technically a render farm too, but in this case, it is usually owned by a company that takes care of all the details of workload balance, priority management and server maintenance, and offers as a service the use of their servers for rendering projects for others.

Cloud computing is a much simpler option to use than owning a render farm and the cost efficiency of it will be studied compared to having an owned render farm and the possibility of using both an in-house render farm and outsourced cloud computing services for extraordinary spikes in activity. The prices of cloud computing services are usually calculated with a formula involving GHzh, number of nodes and queue priority.

3 FACTORS AFFECTING RENDER TIMES

3.1 Hardware

The hardware components have a huge impact on performance. Over the years the hardware has improved for both consumers and producers, so even if much more computers are available, the new quality requirements still mean that, even using cutting edge technology, parallelization is still required.

3.1.1 CPU

The CPU will do most of the work for some renderers like ART and Arnold. It will also affect, to some extent, any renderer since it will be involved even if most of the work is done with a GPU.

When working with a scene, usually single core performance is the decisive factor, along with Turboboost helping deliver a pleasant experience. But for rendering, the most important factor is having as many cores as possible, followed by high clock speeds. Turboboost speeds are not as important since when all cores are maxed under constant load, it will not affect production times.

Figure 2 AMD Threadripper (proshop)

Due to the current trend of CPU manufacturers increasing processor’s core count even for consumer grade products, there are many good options at multiple price ranges. For example, AMD Ryzen Threadripper 2990WX is 1900€ for 64 processing threads, less than 30€/thread, seen in Figure 2.

3.1.2 GPU

A GPU will do most of the work for some GPU renderers like Redshift, but it does not participate at all in CPU based renderers. Usually the renderers using GPU are faster and nowadays the quality is generally as high as with CPU rendering. Some renderers like V-Ray can use GPU or CPU to render with the exact same result (chaosgroup 2019).

GPUs have the option of connecting more than one in the same machine and use them as a single entity, but this has a performance cost as well. In general, two GPUs render at 1.8 times the speed, while four will do so at 3.5 times the speed, generating a big loss in performance that might still be offset, if compared to the cost of a second machine to house the second graphics card. Although some renderers can use multiple GPUs with minimal loss in performance. (Glawion, A 2019).

3.1.3 Cooling

The cooling system is critical. Rendering can often overwork CPUs and GPUs to the point of overheating. This can lead to thermal throttling, reducing efficiency or even leading to crashes and emergency shutdowns in cases of extremely bad cooling. GPUs might be even more reliant on cooling since many of them nowadays have a built-in overclock system that will go quite high if the power consumption and temperatures are properly controlled.

Cooling will also be a big part of the energy expenses and, in practical scenarios, loud. So, for recording studios and similar, the location and insulation of the server rooms might be a complex topic.

> *”For every watt that is spent running servers," says Dr Brad Karp, of University College London, "the best enterprises most careful about minimising the energy of cooling and maximising efficiency typically find they are spending 40-60% extra energy on just cooling them”.*

(Hancock 2009)

3.1.4 RAM

Although RAM speeds are not generally significant in rendering, having enough RAM is critical, since without it the process can crash or return an incorrect result. For renderers using GPU, the internal VRAM will have a similar significance since using system memory will considerably slow down GPU renderers.

Nowadays it is recommended to have at least 16GB of system RAM for small projects and at least 32GB for general use. For GPU rendering it is recommended to have two times as much RAM as VRAM.

3.1.5 Storage

Storage space is generally not all that important for rendering, especially on the server machines in a distributed rendering network, but it is still a good idea to have enough space so that the drives are not near saturation and using SSDs over HDDs.

Fast storage can be very important for working with the scenes and a good studio should have 1Gb or 10Gb Ethernet if working with large files.

3.1.6 Network

When rendering on a single machine, network speeds are not relevant, although for working with 3D models an option is to have these stored on a network instead of the machine, in which case higher speed will make working easier for the 3D artist.

When working with a render farm, network speed becomes very important. At least 1Gb Ethernet is recommended, but 10Gb is, of course, better. The network speed will be an important factor on the parallelization overhead, although the main problem is that the manager will connect to every server machine to assign their tasks. So, in a star configuration with a switch in the center, the link between manager and switch will be a bottleneck.

For better performance, the rendered images should be stored in a networked drive and never in the manager, since that would also force a lot of data to go to a single link already in use.

In a best-case scenario, the scene we want to render should be stored in a Storage Area Network that the servers can access and to which they will deliver the rendered frames. This way we can have a high-speed link (aggregated links) between switch and storage, possibly with high-speed storage in a fast and secure RAID.

In most cases this is not necessary, and all needed is 1Gb connections since the higher speeds only give small benefits unless the network saturates and bottlenecks the process.

3.2 Software

The software used for rendering is important in terms of speed, although generally less so than hardware. There are many options to choose from and often the licensing works with a subscription model.

3.2.1 Modeling software

The modeling software is generally the one 3D artists and animators work with, for example Blender or 3DS Max. When the project is ready for rendering, this program will send the task data (scene, frames to render, resolution, exposure values, etc.) to the rendering engine that will do the actual rendering work. Creating and sending the task data can have an impact on rendering times, but this impact is generally small, since its completion time is, usually, significantly faster than the rendering.

We will be using 3DS Max since it is one of the most popular options for professional use, and Backburner as the work manager for our render farm. Other popular software in 3D modeling and animation are Maya, Blender and Cinema 4D.

3.2.2 Rendering engine

The rendering engine is the software component that performs the actual rendering process. There are many engines available, some for free, like the built-in options in 3DS Max: Scanline (fast but lackluster), ART (commonly used in architecture) and Arnold (new to the latest versions of 3DS Max). There are also proprietary engines like V-Ray or Octane, which require a license of their own.

Some renderers can give better results at the cost of higher rendering times. Choosing an ideal renderer depends on the specifics of the task at hand and is beyond the scope of the work here, in this document only a quick look to real-world time examples of their differences is taken, to get a sense of scale and to compare their scalability, figure 3.

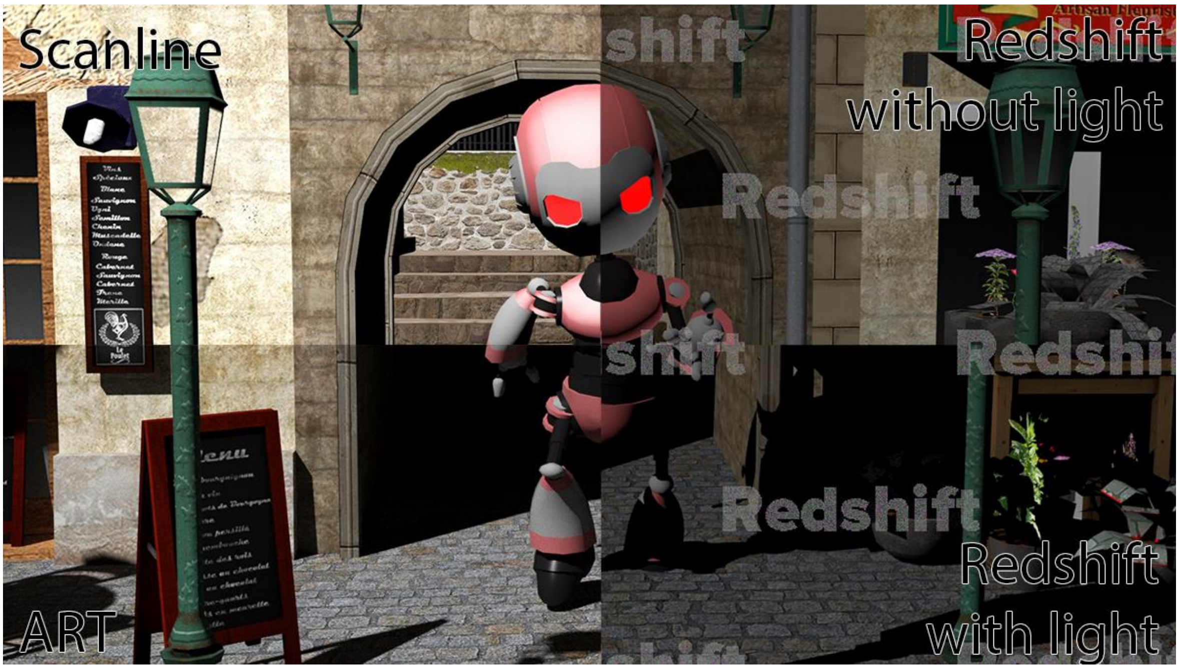


Figure 3 Comparison of diferent renderers

The options built into 3DS Max are mostly CPU renderers and the only GPU renderer included in 3DS Max is Quicksilver, but it is not generally used commercially. Because of that we will also use Redshift for GPU testing, allowing us to compare GPU and CPU better. Other popular renderers include Cycles, V-Ray and Octane.

For testing purposes ART, Arnold, Quicksilver and Redshift will be used. Redshift has a free trial version that adds a watermark to the frames. Also, it is worth noting that while rendering with Arnold is free in iterative work, Arnold adds a watermark when working with Backburner unless a license is paid, as explained in Chapter 5.

Buckets and frames

Rendering engines generate frames by splitting each one in many rectangles of the final image called buckets. This allows each processor thread to work on a different bucket at the same time, speeding up work.

When using Backburner, the manager will split the work in frames and send them to different machines, but it cannot split buckets the same way since one of its limitations is its inability to have more than one machine working on the same frame, because it does not coordinate writing data.

Of course, for rendering a single image, splitting the bucket load is the only way to use multiple machines on one task, so having this option is always worthwhile when building a system that might have to work with single images as well as image sequences. Some renderers like V-Ray have an alternative to Backburner that allows splitting the work in buckets instead of frames for higher parallelization at the cost of more network traffic.

3.3 Internal factors

There are many internal factors of the scene we want to render that will influence our render times. The number of elements to render are clearly a scaling factor since the more models to render the longer it will take to calculate the interactions between them. Similarly, rendering more pixels per frame or more frames will increase our times.

The internal factors often cannot be changed without compromising the original design intention. Thus, they should only be tampered with when unavoidable, with particle and light effects being usual culprits in increasing the render time dramatically.

3.3.1 Resolution

Resolution might be the most important factor since it scales faster than it seems and will compound on all other factors. Usually a certain resolution is needed for certain tasks and cannot be downgraded. Consumers are getting used to high resolutions with full HD being the absolute minimum and 4K a common demand nowadays.

Table 1 Resolution and pixel count

Name	Width (pixels)	Height (pixels)	Pixel count
Full HD (16:9)	1920	1080	2 073 600
4K UHD (16:9)	3840	2160	8 294 400
8K UHD-2 (16:9)	7680	4320	33 177 600
16K (16:9)	15360	8640	132 710 400

Even though 8k displays are not available in most households yet, they will be more relevant as the technology gets cheaper. As for 16k, it is not available to consumers yet but support for it is being developed and might have a role to play in immersive VR soon. As can be seen in Table 1 Resolution and pixel count, the pixel count scales quadratically so even though 4K has twice the dimensions of full HD, the pixel count is four times greater.

3.3.2 Frame rate

For video rendering, the number of frames will increase our render time linearly. Common frame rates go from 24 frames per second in cinema to 120 fps in video games.

For example, a 1-minute video at 8k 240 Hz would have the same number of pixels as 160 minutes of HD video at 24 Hz:

60*33177600*240 = 160*60*2073600*24 = 477757440000 = 4.8 $*10^{11}$ pixels

When using a render farm, the number of frames should not impact significantly parallelization efficiency. This will be tested during benchmarking.

3.3.3 Models

The number and complexity of models will affect the render time. For example, models with many polygons take more time to calculate, and having a high number of objects on screen, like in particle effects used for dust or snow, will heavily impact the rendering as well as the amount of RAM needed. Usually small details like the billboards in the background buildings of Figure 4 are flat for faster renders and have depth added with textures and materials.

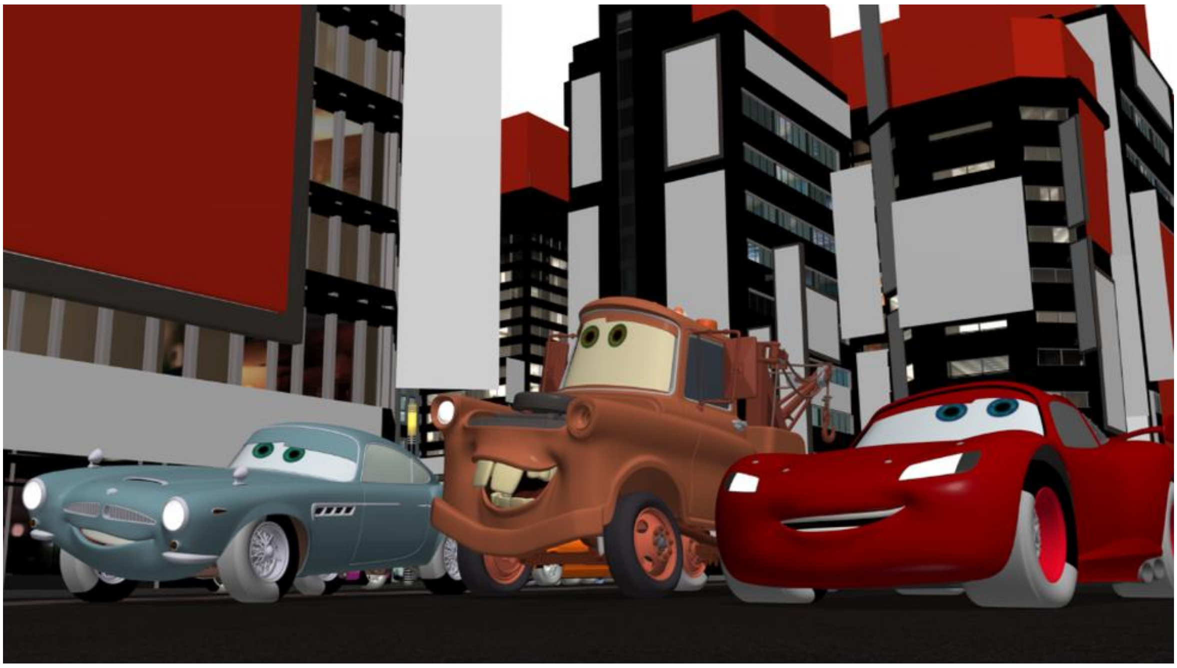


Figure 4 New technology revs up Pixar's 'Cars 2' (Terdiman 2011)

There is also the problem of rendering models that are not visible, as some renderers will calculate covered parts of the objects for their possible impact on reflections.

3.3.4 Textures and materials

Textures define color detail for the models. The texture quality of the models will impact render times. Also, texture maps are notorious for being large images in storage that take a while to load to RAM and take a big chunk of it, being able to crash low RAM systems or destroy their time if the system starts swapping.

Materials are a way to associate properties to the models. From physics behaviors to shading effects, they influence the type and amount of resources dedicated to each model and some materials might have incompatibilities with some renderers. For example, in Figure 5 materials determine how reflective different characters are, the rust effect is colored with a texture map in the diffuse layer, while the lack of shine is handled by the specularity layer of the material.

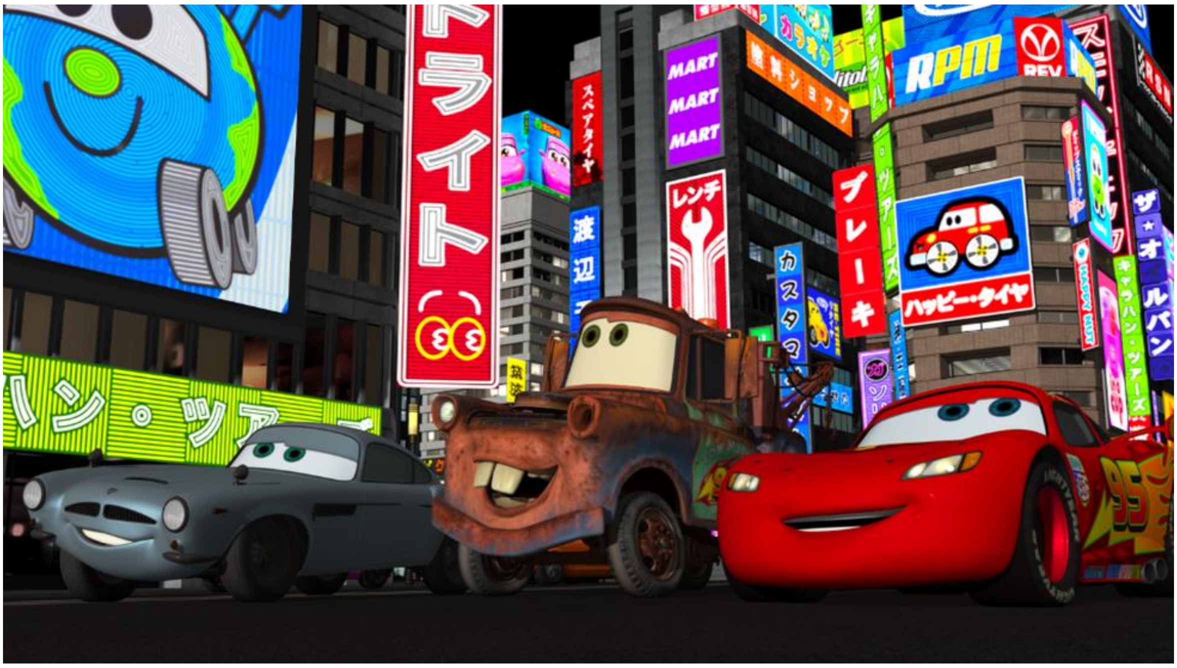


Figure 5 New technology revs up Pixar's 'Cars 2' (Terdiman 2011)

3.3.5 Light

Light is one of the most taxing and complex effects. A myriad of factors influence how it will affect the rendering. There are many ways to calculate light effects and the complexity of the lights, shadows and reflections will severely impact the rendering times. In Figure 6 an example of a high complexity light scene with many reflections can be seen.

Figure 6 New technology revs up Pixar's 'Cars 2' (Terdiman 2011)

Light increases render times a lot because the common rendering methods like ray casting and ray tracing calculate the path of photons through the scene so light and reflective materials increase the complexity a lot. In a similar way having particles like smoke or rain can significantly increase the number of rays and bounces. On the other hand, renderers can generally be customized and limit the number of bounces of light to reduce rendering times.

3.4 Parallelization

When connecting multiple machines to increase our rendering speeds, the first question is how much faster we can go. Two machines (n = 2) will never be twice as fast. There are many reasons for this.

3.4.1 Amdahl's law

Amdahl's law was formulated by Gene Amdahl in 1967. It gives the theoretical speedup for a given improvement. In parallel computing it can be used to calculate the theoretical speedup of adding more computers. This law is formulated in Equation 1 where "$S_{latency}$" is the theoretical speedup, "s" is the speedup of the part we are improving, and "p" is the proportion of execution time that can be parallelized.

$$S_{\text{latency}}(s) = \frac{1}{(1-p) + \frac{p}{s}}$$

Equation 1 Amdahl's law (Wikipedia)

For example, if we have ten machines (s=10) and the parallelizable task takes 95% of the time (p= 0.95), then S will be 6.9, making the render time almost 7 times shorter. In this case the theoretical limit with infinite machines would result in a speedup of S=20.

This law proves that the speedup is limited by how much of the task can be parallelized. For example, no matter how many PCs are added if the network speed is very low it will reach a limit in performance quite soon, or if the manager machine needs to access a slow hard drive to get the information for the servers, that part cannot be improved with parallelization.

However, for rendering, parallelization is quite efficient. Since most of the processing time is used to generate each frame, which is a task independent from the rest, the manager can send several frames to each machine and they can each work at their own speed without bottlenecks.

Of course, everything the manager is doing cannot be parallelized but, during rendering, most of the job can be split with minimal loss in efficiency. That is why server farms have been used for many years and some of them have hundreds of thousands of processing cores.

3.4.2 Parallelization overhead

Parallelization overhead is the difference in the algorithm between the same serialized and parallelized work. In our case for example, splitting the task in frames, assigning frames to servers and generally everything done by the manager program, is an artificial tool that does not exist when rendering on a single machine.

For a good speedup (close to n), a low parallelization overhead is the key factor. Since part of our overhead is almost the same with two computers or two hundred, when doing very lengthy tasks on many computers, the overhead may seem to diminish. On the other hand, some of the overhead increases with the number of computers (n), like the network communications, and can impose a hard limit on how many machines can be connected for a given task.

3.4.3 Latency and network overhead

Because the tasks are split into several machines, sometimes they need information from each other. This might make a machine wait idle for an answer if the communications are slow. This is called latency or lag.

Similarly, the data transfer rates are slower than when a machine needs to access memory or other processing cores in the same motherboard or even processor. On top of that, if the amount of data traffic is too high for the network, it will slow down even further.

In rendering this can be a minor inconvenience or a huge problem depending on the number of computers and the network quality. Since all data must go from one storage to all servers and then back to a storage, this simultaneous transfer of data can easily collapse an unprepared network.

3.4.4 Synchronization, deadlock and errors

When the machines are communicating with each other they need a way to synchronize since they have different clock times and it can cause problems with the execution order. This is a minor problem in rendering since the manager coordinates all servers and servers do not need anything from each other.

A deadlock situation in parallel computing happens when a part of the code “A” can’t be executed because it needs a “key” that can’t be obtained until another part “B” completes first but for B to complete it needs a key obtained by completing A, as seen in Figure 7. This will lock the machines in an endless wait for tasks to be completed. It should not affect rendering unless there is some bug in the program managing the servers since a deadlocked server is just an error.

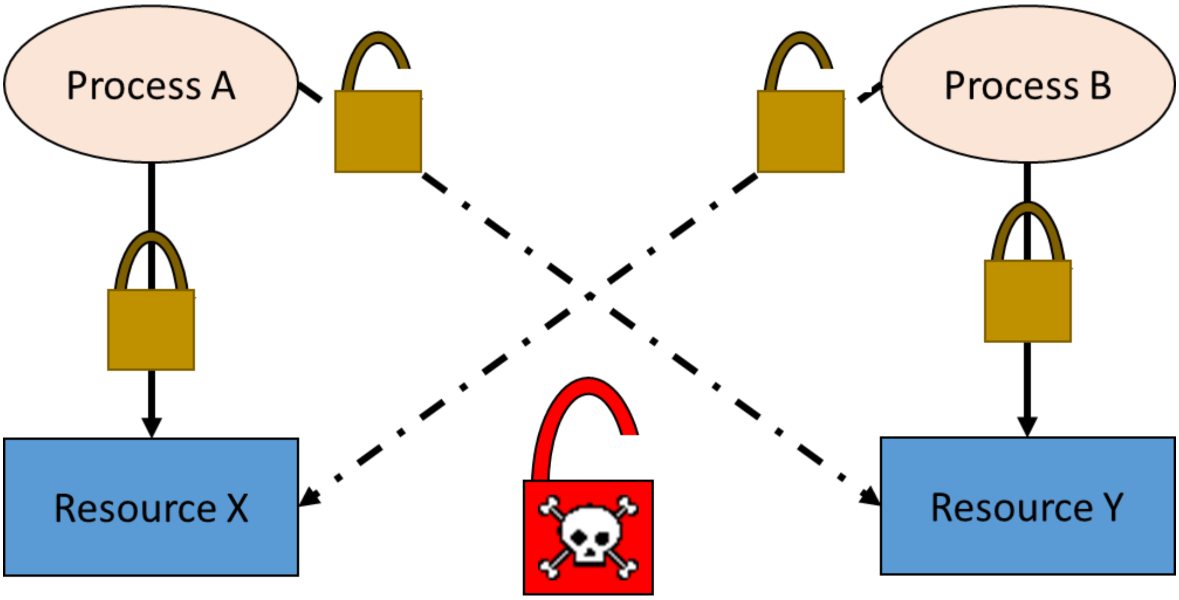


Figure 7 Deadlock (Steynberg 2016)

An error takes place when a machine returns an incorrect result, always or sometimes (making the machine unreliable), or no result at all. In our case a faulty machine could generate bad frames that we only notice after assembly, although this is unlikely. The most common errors are non-responsive machines due to execution or network errors and, in this case, rendering programs are perfectly capable of resending the task to another machine.

In more general terms, with a distributed system errors are difficult to detect and correct since the machine detecting the error could itself be the faulty one and so complex systems need to be made to avoid them. In our case for rendering, if the manager machine works correctly, we should obtain the correct result since any machines not completing their tasks in time will be discarded by the manager and their tasks will be reassigned to other machines.

3.4.5 Expected performance

Due to rendering being a highly parallelizable task, we can expect speedup results to be a bit lower than Amdahl’s law’s maximum theoretical speedup. If we keep adding machines to the system, the parallelization overhead or the network overhead will make the speedup stop growing and go down again.

In every parallel task there is always some factor that cannot keep scaling forever and limits its speedup past a certain point. With current technology, that is not a big problem in rendering since a two or even three order of magnitude increase is achievable and is more than enough for most practical uses.

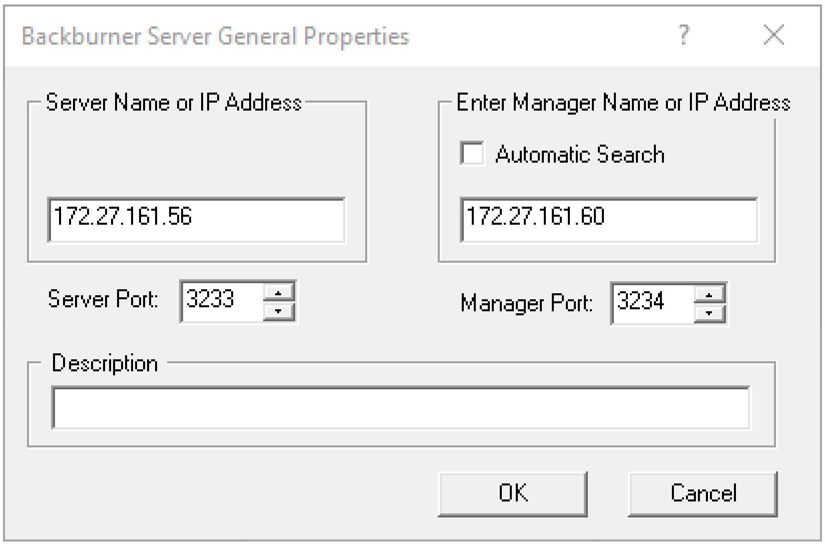
4 EMPIRICAL STUDY

4.1 Setting up Backburner

For setting up Backburner, a set of machines are needed. They can be networked in LAN or through the internet but in this study a LAN network is used with 1Gb connections. Then, a shared folder must be set up so that all machines can save files to the same address. It can be a Windows shared folder or a cloud service like OneDrive. In the benchmarking a OneDrive folder is used. It’s not an ideal solution since it will add some latency and network traffic but, working within LAMKs network, a shared folder was not an option.

Autodesk’s 3DS Max and Backburner need to be installed in all the machines used as well as any specific rendering software like Redshift and any necessary licenses. In the benchmarking section Redshift’s free trial version is used as well as the renderers included in 3DS Max. Windows firewall exceptions might be necessary.

Finally, in the manager computer, the manager program is started, leaving the default settings on. For the servers, the IP address of both the server machine and the manager must be specified as in Figure 8 if the server IP is not specified by using “localhost” the connection will be rejected.



Backburner Server General Properties

Server Name or IP Address: 172.27.161.56

Enter Manager Name or IP Address

□ Automatic Search

172.27.161.60

Server Port: 3233

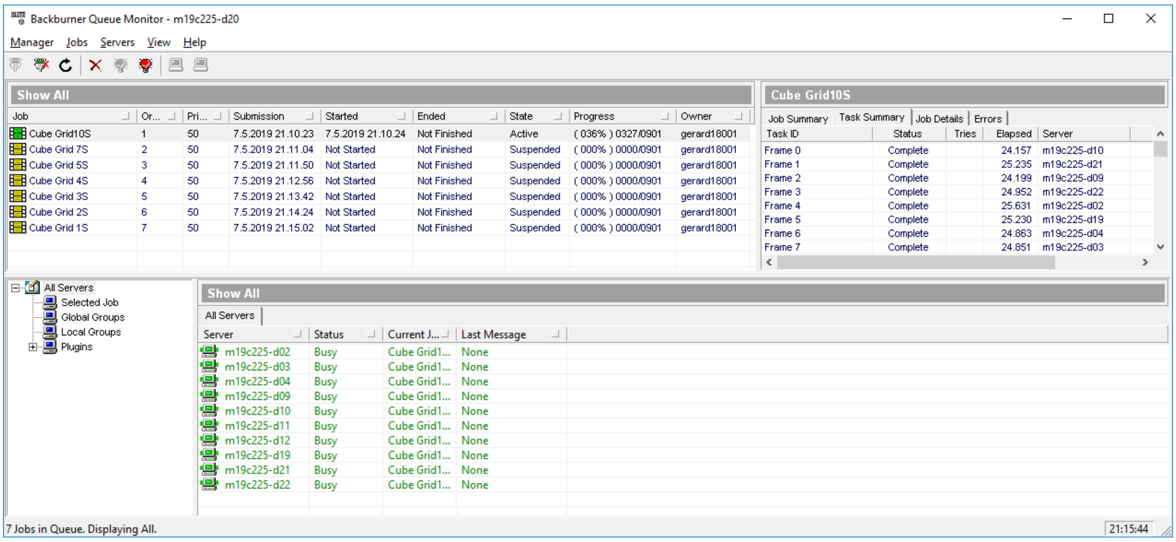
Manager Port: 3234

Description:

Figure 8 Backburner server

We are now ready to start queuing jobs in Backburner from any computer in the same LAN. It’s important to notice that when rendering animations, the frames must be saved as an image sequence .jpg or .png, otherwise Backburner will assign each job to a computer without splitting the frames. For assembling the image sequence, we can use AfterEffects. The destination folder for rendering must be the OneDrive folder and every machine must have access to it. Backburner Monitor (Figure 9) can be used from any machine in the

LAN to see the progress but changes like queue management are restricted to the manager computer. The web monitor can give a certain amount of control over projects started in the local machine but never on other projects.



Job	Or...	Pri...	Submission	Started	Ended	State	Progress	Owner
Cube Grid10S	1	50	7.5.2019 21.10.23	7.5.2019 21.10.24	Not Finished	Active	(036%) 0327/0901	gerard18001
Cube Grid 7S	2	50	7.5.2019 21.11.04	Not Started	Not Finished	Suspended	(000%) 0000/0901	gerard18001
Cube Grid 5S	3	50	7.5.2019 21.11.50	Not Started	Not Finished	Suspended	(000%) 0000/0901	gerard18001
Cube Grid 4S	4	50	7.5.2019 21.12.56	Not Started	Not Finished	Suspended	(000%) 0000/0901	gerard18001
Cube Grid 3S	5	50	7.5.2019 21.13.42	Not Started	Not Finished	Suspended	(000%) 0000/0901	gerard18001
Cube Grid 2S	6	50	7.5.2019 21.14.24	Not Started	Not Finished	Suspended	(000%) 0000/0901	gerard18001
Cube Grid 1S	7	50	7.5.2019 21.15.02	Not Started	Not Finished	Suspended	(000%) 0000/0901	gerard18001

Task ID	Status	Tries	Elapsed	Server
Frame 0	Complete		24.157	m19c225-d10
Frame 1	Complete		25.235	m19c225-d21
Frame 2	Complete		24.199	m19c225-d09
Frame 3	Complete		24.952	m19c225-d22
Frame 4	Complete		25.631	m19c225-d02
Frame 5	Complete		25.230	m19c225-d19
Frame 6	Complete		24.863	m19c225-d04
Frame 7	Complete		24.851	m19c225-d03

Server	Status	Current J...	Last Message
m19c225-d02	Busy	Cube Grid1...	None
m19c225-d03	Busy	Cube Grid1...	None
m19c225-d04	Busy	Cube Grid1...	None
m19c225-d09	Busy	Cube Grid1...	None
m19c225-d10	Busy	Cube Grid1...	None
m19c225-d11	Busy	Cube Grid1...	None
m19c225-d12	Busy	Cube Grid1...	None
m19c225-d19	Busy	Cube Grid1...	None
m19c225-d21	Busy	Cube Grid1...	None
m19c225-d22	Busy	Cube Grid1...	None

7 Jobs in Queue. Displaying All. 21:15:44

Figure 9 Backburner monitor

4.2 Benchmarking

In our testing we will use the render farm to study the factors affecting render times and the cost of parallelization at every level. We will set up so every test is done with a control group without the use of Backburner (C), a single machine using Backburner as queue manager and renderer or server (M=S), (M+XS) where M is a manager computer not rendering and X is the number of server machines (M+1, M+2, M+3, M+4, M+5, M+7 and M+10) and (M=S+XS) where we have X servers and the manager helps them render to compare them to having a dedicated manager to study network impact.

4.2.1 Methodology

The methodology and our raw results are exposed in Appendix 1. The topography of the network is as seen in Figure 10. All our tests have the following characteristics:

- We are using machines with the following hardware: Intel Core i7-7700 (4 cores, 8 threads @ 3.6 GHz), GeForce GTX 1080/PCIe/SSE2, 16GB of RAM with a Cinebench score of 1772 Multi CPU and 417 single CPU

- Each test is recorded in Appendix 1. Some tests are repeated 3 times and the median value is taken for analysis.

- The six scenes used are described in Appendix 2. The scenes one trough six are called respectively: Cube Grid, Simple Physics, Cell Noise, Shuffle Swap, Robot and Building
- The tests are recorded as: Test #, renderer, scene, resolution, number of frames, date. Example: Test 1, Arnold, S1, 4K, 30, 23/04/19
- The tests record the times for the following setups: C (control group without Backburner), M=S (manager and server in one machine), 1M 1S (two machines), 1M 2S, 1M 3S, 1M 4S, 1M 5S, 1M 10S. Some tests use M=S + XS, meaning that it has X servers plus the manager machine also working as a server.

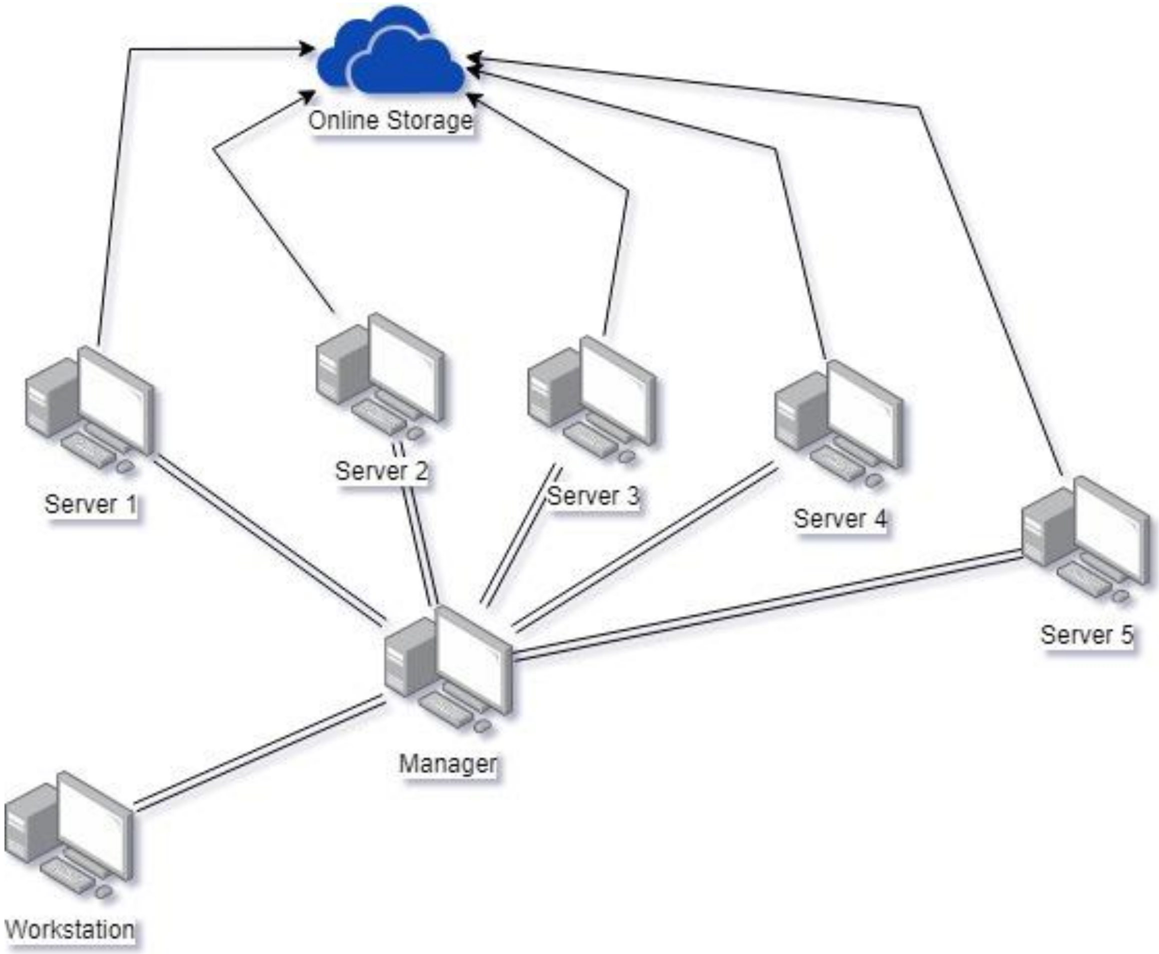


Figure 10 Network topography

4.2.2 Stage 1: machine comparison

First a test of the 11 machines used for most of the testing was done to check that there is no significant difference in their speeds. It can be seen in Table 2 that there is no machine generally faster or slower than the rest except, of course, the manager machine being faster due to not having to connect and transmit data.

Table 2 Stage 1, test 1

	A	B	C	Median	Standard deviation
Machine 1	580	582	581	581	1
Machine 2	581	582	582	581	0,57735
Machine 3	580	581	570	577	6,082763
Machine 4	576	576	588	580	6,928203
Machine 5	580	581	561	574	11,26943
Machine 6	582	581	593	585	6,658328
Machine 7	579	581	579	580	1,154701
Machine 8	581	582	583	582	1
Machine 9	589	589	579	586	5,773503
Machine 10	581	580	581	580	0,57735
Manager	503	511	509	507,6667	4,163332

4.2.3 Stage 2: analysis of parallelization speedup

The parallelization speedup is calculated from using the data of a single server machine, but it must be noted, that rendering on a single machine without using Backburner (result C) provides faster results. The time to render a single frame is the same but it’s much faster to start and save the data resulting in faster task completion time. In Figure 11 the rightmost data point is the speedup of not using backburner compared to using it in only one machine.

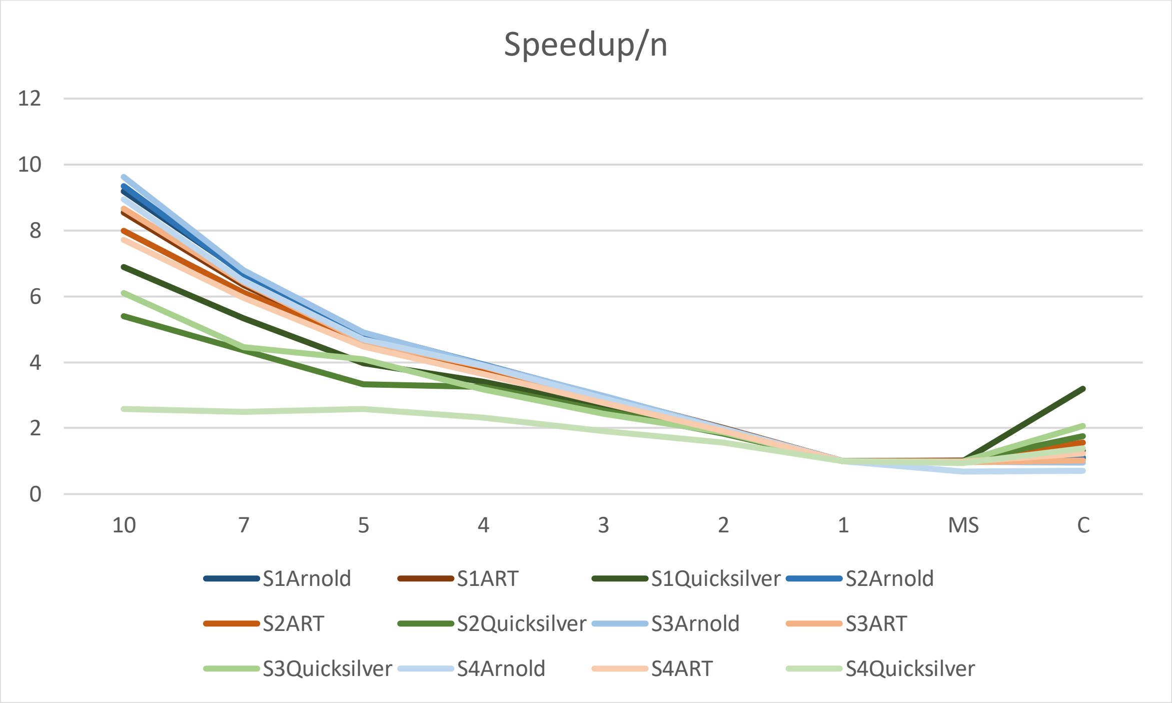


Figure 11 Stage 2, speedup analysis

Testing reveals how longer render times correlate to faster speedup. This is because the software takes between 10 to 20 seconds to start rendering when using Backburner and only connects to up to four servers at a time. So, to connect 30 machines the last one might not get started for about two minutes. Because of this when splitting tasks with few frames and/or short render times the speedup is very low. For example, Test 13 Scene 4 Quicksilver seen in Table 3 The speedup achieved with 10 machines is tiny since the total render time is so low.

Table 3 Test 13 S4 Quicksilver

	A	B	C	Median	Speedup
10S	55	54	55	55	2,581818
7S	57	54	62	57	2,491228
5S	55	54	55	55	2,581818
4S	65	60	61	61	2,327869
3S	74	74	73	74	1,918919
2S	91	91	92	91	1,56044
1S	142	142	143	142	1
M=S	146	152	151	151	0,940397
C	102	74	151	102	1,392157

Organizing the data from shortest completion time to longest, a correlation between longer render times and higher speedups can be easily spotted in Figure 12.

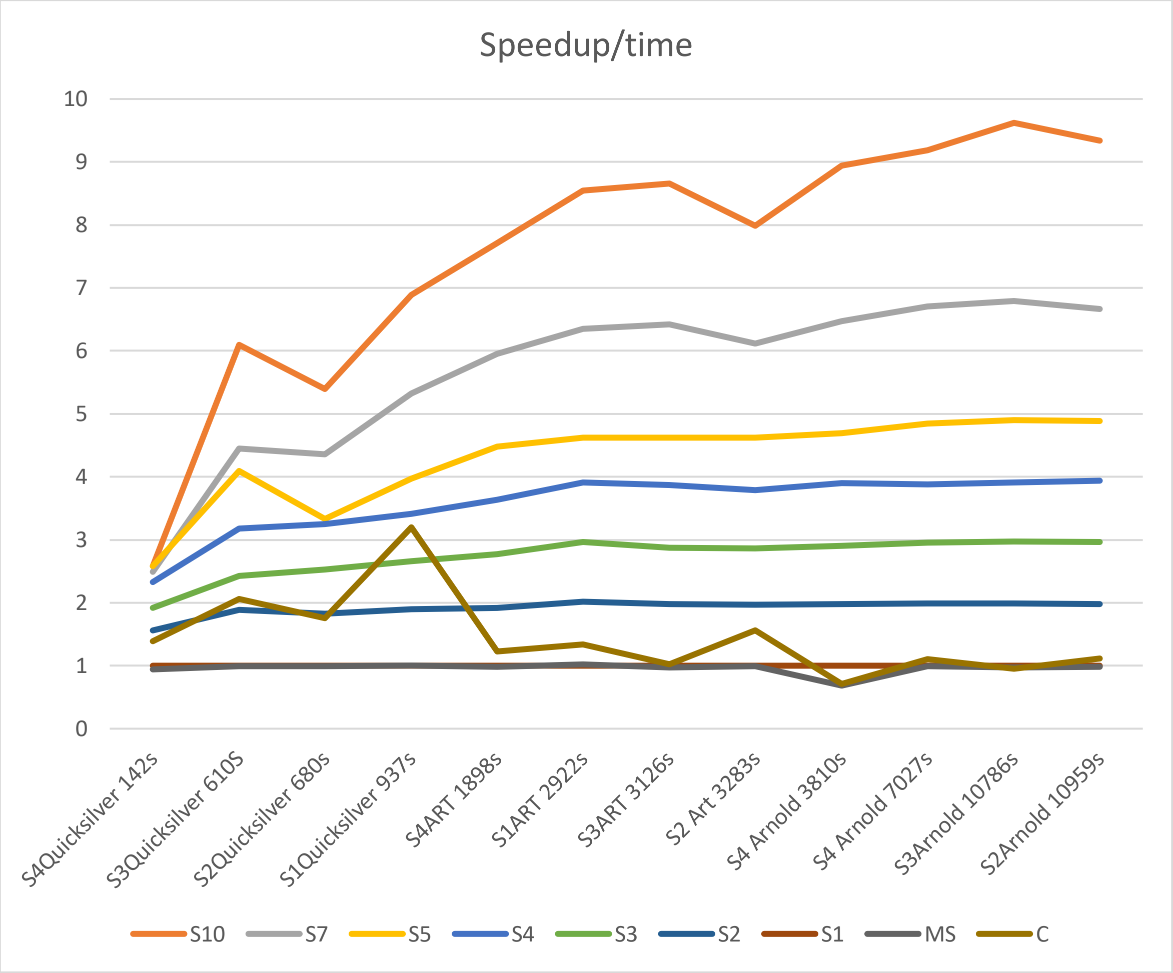


Figure 12 Speedup evolution related to render time

4.2.4 Stage 3: resolution and frame rate

Renders of scenes five (robot) and six (building) are done at resolutions of full HD, 4K and 8K. The render times of HD quality are compared to those of 4K and 8K. Since 4K and 8K have four and sixteen times the number of pixels respectively, a similar speedup is expected in Figure 13, where robot and building refer to scene five and six respectively and “tested” represents the results found empirically, while “theoretical” is calculated by multiplying the found value in the HD test by four or sixteen as a baseline of what would be expected.

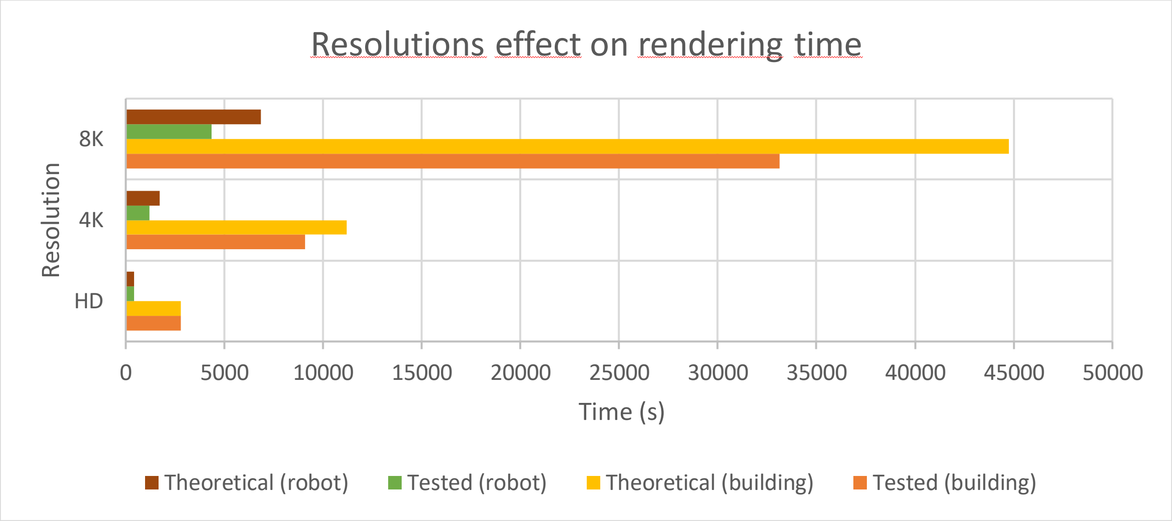


Figure 13 Study on resolution

Instead it is found that the increase between HD and 4K averages around three times increase while the difference between HD and 8K is only about ten times increase compared to the expected sixteen. In Table 4 it can be seen in the last 2 columns where the 4K and 8K times are divided by the HD time.

Table 4 Study on resolution, Test 14 Table

	HD	4K	8K	4K/HD	8K/HD
Tested building (s)	2798	9101	33136	3,25268	11,84274
Theoretical building (s)	2798	11192	44768	4	16
Tested robot (s)	428	1202	4351	2,808411	10,16589
Theoretical robot (s)	428	1712	6848	4	16

These findings are very interesting since the resolution demanded by the market keeps growing. The fact that the increase in render time is lower than expected will be even more relevant if 16K makes it to the consumer’s homes, since this one would have 64 times the number of pixels in HD video. This phenomenon could be a result of higher resolutions having higher parallelization potential since the frames were rendered using ten machines, but it is still a good thing that, in practice, big resolutions are not as time consuming to render as one might think.

As for frame rate, by itself it doesn’t really affect rendering times, but it should be noted that, because of Backburner’s limitation, it’s not possible to split a frame’s buckets to render among many machines, so rendering more frames than machines are available is needed to be able to parallelize the process. In the testing done here as little as 500 frames divided among 30 computers were used without noticing any delay in results.

4.2.5 Stage 4: using manager machine as server

In the guidelines for using Backburner, as well as the general advice provided for setting any render farm, it is stated that the machine running the manager program should be left unused to avoid a bottleneck caused by a delay on new orders from the manager to the servers.

This was challenged in a study using up to 31 computers and comparing the results of using the manager as server against reserving it for running the Backburner manager. The results seen in Figure 14 show that the difference between both uses is not significant when working with small number of server machines.

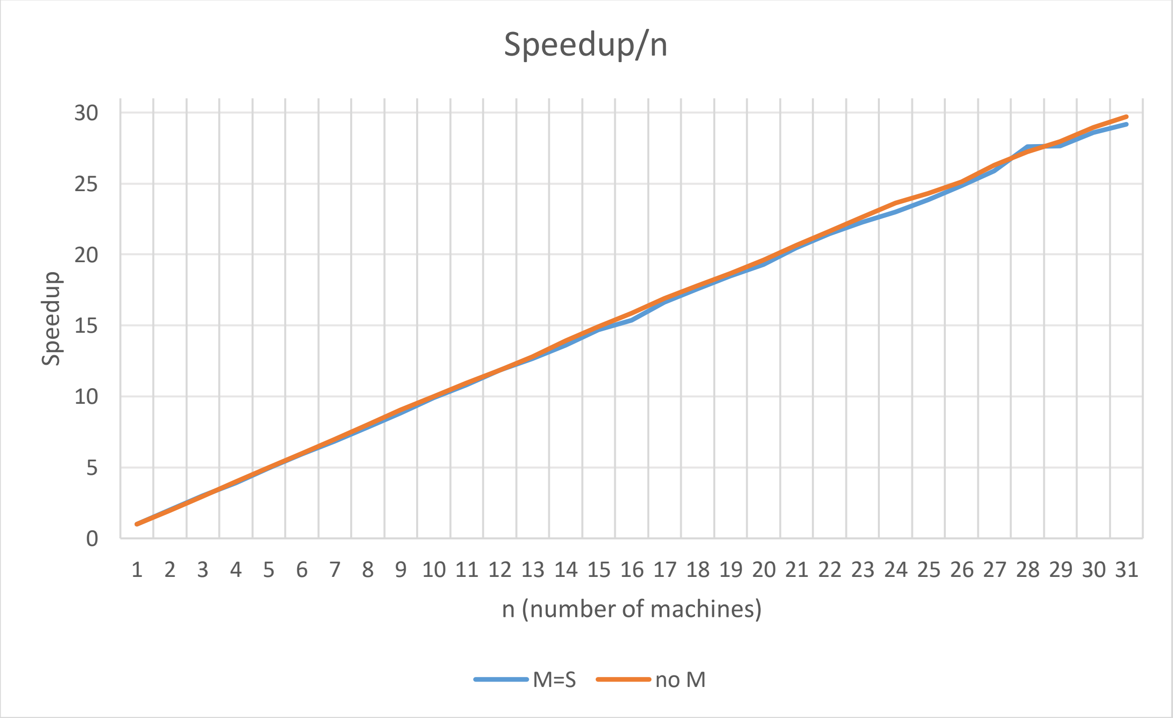


Figure 14 Using manager as server

5 COST ANALYSIS

5.1 Hardware and licensing costs

The cost of licensing 3D Studio Max for commercial use can be as low as 152,43€ a month as seen in Table 5. Additionally, when using Backburner, only the manager computer requires a license so a company only needs to pay one license for each 3D artist and one more for the manager computer of the render farm, resources can be saved by making one of the artist's workstations be the manager computer of the network although this is discouraged for big networks for the same reason is advisable to not run the server program on the manager machine, it can slow the whole network.

Table 5 3DS Max license price list (Autodesk 2019)

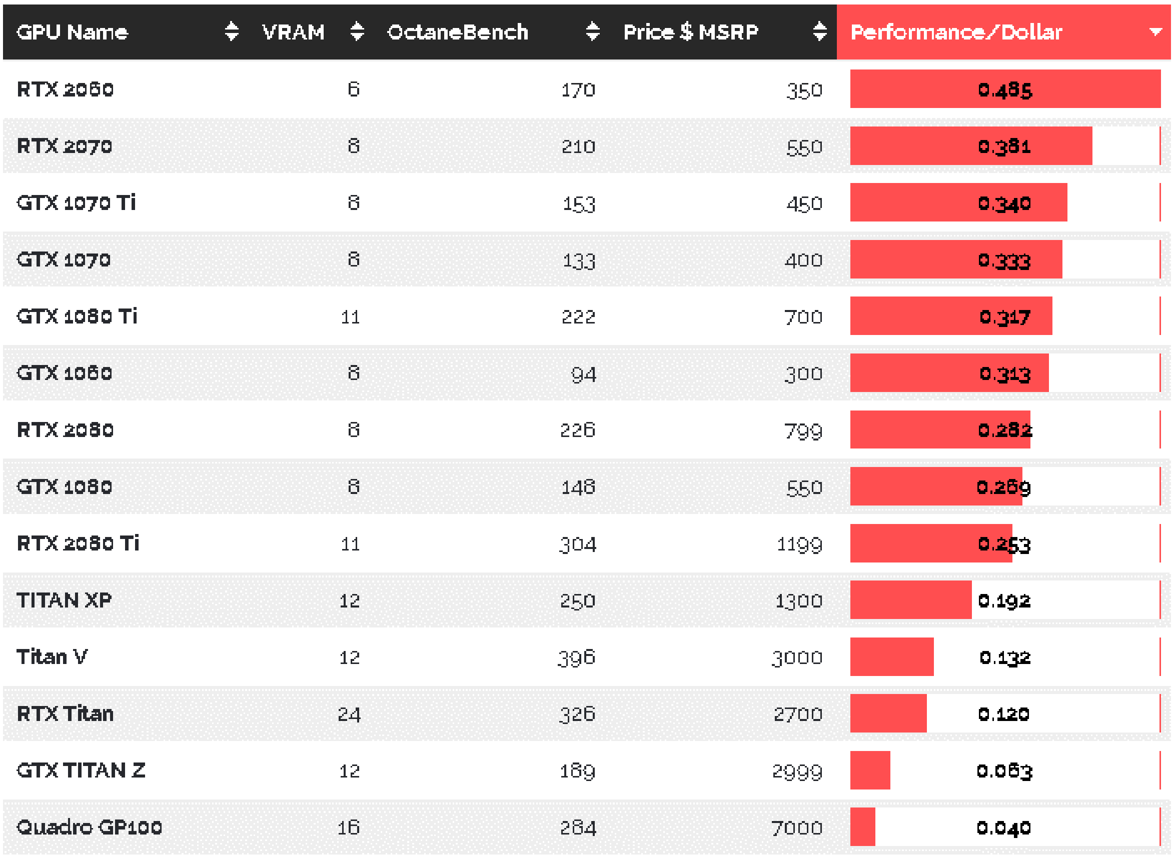
Months	Price	Price/month
1	254,10€	254,10€/month
12	2032,80€	169,40€/month
36	5487,35€	152,43€/month

When rendering in Backburner you also need an Arnold license (Table 6) in every rendering node. Other renderers might have different license prices and rules like Redshift's one payment permanent license of 500€ per machine. Because of license prices being a considerable expense, the use of artist workstation as render servers becomes interesting. Software like Backburner allows to have servers work on specific schedules, so the workstation can be used to render after work hours.

Table 6 Arnold license price list (arnoldrenderer 2019)

	Monthly 1	Yearly 1	Yearly 5	Yearly 25	Yearly 100
Price	70€	685€	2580€	10 260€	27 550€
Price/ year/ machine	840€	685€	516€	410,40€	275,50€

The hardware costs are high as well. In our example, each system costs about 1200€ although when building a render farm from scratch it can be lower price for the same performance buying components that have better value. For example, as seen in Figure 15 the GTX 1070 performs almost as well as the GTX1080 at a much lower price making it more efficient when planning to build rendering machines. Also, when considering GPU renderers, many can use up to four or even eight GPUs at a time, considerably increasing performance per machine.



GPU Name	VRAM	OctaneBench	Price $ MSRP	Performance/Dollar
RTX 2060	6	170	350	0.485
RTX 2070	8	210	550	0.381
GTX 1070 Ti	8	153	450	0.340
GTX 1070	8	133	400	0.333
GTX 1080 Ti	11	222	700	0.317
GTX 1060	8	94	300	0.313
RTX 2080	8	226	799	0.282
GTX 1080	8	148	550	0.269
RTX 2080 Ti	11	304	1199	0.253
TITAN XP	12	250	1300	0.192
Titan V	12	396	3000	0.132
RTX Titan	24	326	2700	0.120
GTX TITAN Z	12	189	2999	0.063
Quadro GP100	16	284	7000	0.040

Figure 14 GPU value (Glawion 2019)

5.2 Energy cost

The amount of energy used to render is linked to the hardware. For example, the hardware we will use in the next chapter is represented in Table 7.

Table 7 Energy costs (Tom's hardware 2016)

Part name	Maximum power (tested or TDP)	Idle power
Intel i7 7700	65 W	(irrelevant)
GeForce GTX 1080	180 W	12 to 20 W
2X 8GB RAM DDR4	2 X 3W = 6 W	2 X 0.5 = 1 W

According to Table 7 the power consumption for CPU tasks will be about 71W while the GPU tasks will be about 251W. But as mentioned in 3.1.3 the cost of cooling in optimized environments can be from about 40% to 60% of the computing energy. Then, in our poorly optimized case we will assume 60%, our CPU and GPU tasks rise to 113,6 W and 401,6 W respectively.

So, 10 machines working constantly doing CPU tasks half the time and GPU half the time averages 2500 W. That’s 1800kWh in a month, at Finland’s average rate of 7 c/kWh that’s 126€/month of energy consumption.

Environmental cost

When using a computer network for rendering we end up using more resources, both in hardware components and energy for powering and cooling the systems. We will attempt to calculate how the carbon footprint grows as we get faster times increasing the number of machines.

In Finland it’s possible to buy electricity from hydropower at the rates discussed earlier. This energy has very low CO_2 emissions between 4 and 14 g/kWh (IPCC 2019, 471). That would put the energy carbon footprint at between 86,4 and 302,4 KgCO_2e/year.

The hardware components themselves are estimated at 1 ton of CO_2e per 1600€ (carbon-footprint 2019). So, for our 10 machines that adds up to 7,5 tons of CO_2e.

6 CONCLUSIONS

6.1 Impact of data transfers

For small server farms like the one tested in (up to 31 computers), with 1Gb links there was no impact from data transfers on render times. With 31 computers getting speedups of 29 to 30 it can be said that for small render farms 1Gb Ethernet is enough and using the manager machine as a rendering machine is viable as well. Especially when using the web manager monitor so settings can be adjusted from the workstation that submitted that task.

Considering we were using OneDrive instead of a SAN the results are quite good. Since OneDrive tries to sync all PCs it constantly uploads the rendered frames of each computer and downloads it in every other computer so they all have a copy of the complete work. This means that network was being used much more than necessary and even then, great speedup results were obtained. It follows that with a properly managed SAN, networks of a few hundred computers shouldn't need more than 1Gb except maybe a 10 Gb connection between switch and storage or even for the manager machine's connection to the switch.

6.2 Economic calculations

In our sample case of 10 computers the capital cost in a 5 years operation is:

12000 + 152.43*5*12 +126*5*12 = 28 705,80€

The environmental cost is:

7500+86,4*5/1000= 7,932 tons of CO_2e.

7500+302,4*5/1000= 9,012 tons of CO_2e.

The average Finn carbon footprint is 8.66 tons (carbonfootprint 2019)

A 10-machine setup, over 5 years, costs about 30 000€ (less than a fifth the price of an employee) and has a carbon footprint of 8 to 9 tons of CO_2e (about as much as the average Finn in 1 year). So, for a 100-machine set the cost of buying and running them for 5 years is around 300 000€ with a carbon footprint of 80 to 90 tons of CO_2e.

6.3 Setting up a long-term render farm in LAMK

The tests show that LAMK has the infrastructure needed for a decent size render farm. With 100 machines connected with 1Gb links and quite powerful machines only a few changes would be needed.

Figure 16 LAMK campus render farm

First setting up so Users folder is in D drive instead of C. Many machines were unusable because the C drive was completely full. A networked drive would also be needed, possibly with quotas of data for each user. This drive would be used as target for the render farm. For the convenience of everyone a manager pc should be designated, and everything else should be used as server machines. Using the extra machine as server can be done if needed, but more testing is advised before doing this.

If possible, setting backburner server to automatically start in all machines and configured to point to the manager would be a good way of making sure the service is running. Also, only classes C224 and C225 have security permissions to run the programs so this policy should be extended to the other classrooms intended to use this way.

Finally, from the backburner monitor, schedules can be added so servers aren’t used during class hours. Alternatively, if they are wanted for running during class, an administrator can oversee policing the system so not everyone can send jobs to the render farm.

Adding an Arnold license to every machine would cost 27 550€ a year to be able to render without watermark although it could also be done so that only some of the machines are licensed for Arnold and use only those for Arnold renders, while using the others on free renderers like ART, Scanline and Quicksilver. Even Scanline can produce some good-looking results if willing to tailor the settings to the needs of the project.

REFERENCES

Angelini, C. 2016. Power consumption results [accessed 8 April 2019]. Available at: https://www.tomshardware.com/reviews/nvidia-geforce-gtx-1080-pascal,4572-10.html

Arnold 2019. Arnold license [accessed 4 May 2019] Available at: https://www.arnoldrenderer.com/arnold/buy/

Ashton, M. 2017. Autodesk's MCG – What's New 2018 + Sample pack [accessed 29 April 2019] Available at: https://area.autodesk.com/tutorials/mcg-whats-new-2018--sample-pack/

Autodesk 2019. Autodesk [accessed 29 April 2019]. Available at: https://www.autodesk.com/

Carbonfootprint 2019. Calculator [accessed 1 May 2019] Available at: https://www.carbonfootprint.com/calculatorfaqs.html

Chaosgroup 2019. VRAY [accessed 29 April 2019] Available at: https://www.chaosgroup.com/

Chukov, S. 2019. Sketch 3D max RedShift [accessed 10 May 2019] Available at: https://www.behance.net/gallery/79498267/Sketch-3D-max-RedShift-Free-dowloand

Clay, J. 1990. Making of Bored Room [accessed 2 April 2019]. Available at: https://vimeo.com/11358064

Glawion, A. 2019. CGDirector [accessed 8 April 2019]. Available at: https://www.cgdirector.com/

Hancock, S. 2009. Iceland looks to serve the world, BBC News [accessed 5 April 2019]. Available at: http://news.bbc.co.uk/2/hi/programmes/click_online/8297237.stm

IPCC 2011. Intergovernmental Panel on Climate Change 2011 SRREN report, 471[accessed 1 May 2019]. Available at: http://www.ipcc-wg3.de/report/IPCC_SRREN_Ch05.pdf

Junaid_shakoor 2016. Lowpoly robot [Accessed 14 May 2019] Available at: https://sketchfab.com/3d-models/lowpoly-robot-d964729422964112b1cd38d2a5d2e6a6

Mathiasdierickx 2016. City Scene – Inspired by Montpellier [Accessed 14 May 2019] Available at: https://sketchfab.com/3d-models/city-scene-inspired-by-montpellier-5b9edb1d448f430ab0006f5ebd94d91a

Proshop 2019. AMD Ryzen Threadripper 2990WX CPU [accessed 29 April 2019]. Available at: https://www.proshop.fi/CPU/AMD-Ryzen-Threadripper-2990WX-CPU-3-GHz-AMD-TR4-32-ydintae-AMD-Boxed-WOF-ilman-jaeaehdytintae/2674837

Steynberg, M. 2016. What is a SQL server deadlock [accessed 30 April 2019] Available at: https://www.sqlshack.com/what-is-a-sql-server-deadlock/

Terdiman, D. 2011. New technology revs up Pixar's 'Cars 2' [accessed 7 May 2019] Available at: https://www.cnet.com/news/new-technology-revs-up-pixars-cars-2/

Wallossek, I. 2017. Intel Core i7-7700: Power Consumption And Temperatures [accessed 8 April 2019]. Available at: https://www.tomshardware.com/reviews/intel-kaby-lake-core-i7-7700k-i7-7700-i5-7600k-i5-7600,4870-9.html

APPENDIX 1

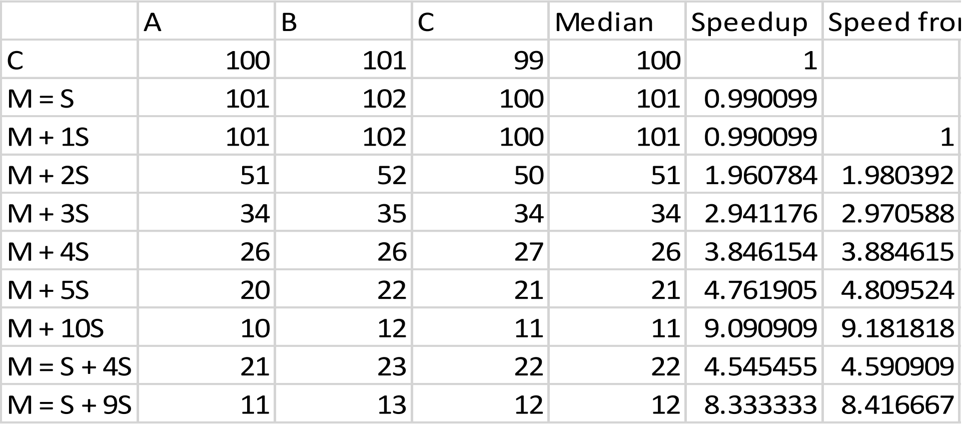
Methodology

The methodology and our raw results are exposed in Appendix 1. The topography of the network is as seen in Figure 10. All our tests have the following characteristics:

- We are using machines with the following hardware: Intel Core i7-7700 (4 cores, 8 threads @ 3.6 GHz), GeForce GTX 1080/PCIe/SSE2, 16GB of RAM with a Cinebench score of 1772 Multi CPU and 417 single CPU
- Each test is recorded in Appendix 1. Some tests are repeated 3 times and the median value is taken for analysis.
- The six scenes used are described in Appendix 2. The scenes one trough six are called respectively: Cube Grid, Simple Physics, Cell Noise, Shuffle Swap, Robot and Building
- The tests are recorded as: Test #, renderer, scene, resolution, number of frames, date. Example: Test 1, Arnold, S1, 4K, 30, 23/04/19
- The tests record the times for the following setups: C (control group without Backburner), M=S (manager and server in one machine), 1M 1S (two machines), 1M 2S, 1M 3S, 1M 4S, 1M 5S, 1M 10S. Some tests use M=S + XS, meaning that it has X servers plus the manager machine also working as a server.

Sample

Table 4 Test 0, Arnold S0, HD, 30 F, 23/04/19



	A	B	C	Median	Speedup	Speed fro
C	100	101	99	100	1	
M = S	101	102	100	101	0.990099	
M + 1S	101	102	100	101	0.990099	1
M + 2S	51	52	50	51	1.960784	1.980392
M + 3S	34	35	34	34	2.941176	2.970588
M + 4S	26	26	27	26	3.846154	3.884615
M + 5S	20	22	21	21	4.761905	4.809524
M + 10S	10	12	11	11	9.090909	9.181818
M = S + 4S	21	23	22	22	4.545455	4.590909
M = S + 9S	11	13	12	12	8.333333	8.416667

Raw results

Table Test 1, Arnold, HD, 101 F, 27/04/19

	A	B	C	Median	Standard deviation
Machine 1	580	582	581	581	1
Machine 2	581	582	582	581	0,57735
Machine 3	580	581	570	577	6,082763
Machine 4	576	576	588	580	6,928203
Machine 5	580	581	561	574	11,26943
Machine 6	582	581	593	585	6,658328
Machine 7	579	581	579	580	1,154701
Machine 8	581	582	583	582	1
Machine 9	589	589	579	586	5,773503
Machine 10	581	580	581	580	0,57735
Machine manager	503	511	509	507,6667	4,163332

Table 5 Test 2, Arnold, S1, HD, 901 F, 27/04/19

	A	B	C	Median	Speedup C	Speedup 1S
10S	763	766	765	765	8,30719	9,185621
7S	1048	1056	1048	1048	6,063931	6,705153
5S	1507	1434	1451	1451	4,379738	4,842867
4S	1812	1806	1809	1809	3,512991	3,884467
3S	2365	2394	2375	2375	2,675789	2,958737
2S	3540	3539	3569	3540	1,795198	1,985028
1S	7022	7027	7100	7027	0,904369	1
M=S	7154	7109	7083	7109	0,893937	0,988465
C	6355	6332	6664	6355	1	1,105744

Table 6 Test 3, ART, S1, HD, 901 F, 27/04/19

	A	B	C	Median	Speedup C	Speedup 1S
10S	342	340	343	342	6,406433	8,54386
7S	459	462	460	460	4,763043	6,352174
5S	632	633	628	632	3,466772	4,623418
4S	748	752	747	748	2,929144	3,906417
3S	978	987	985	985	2,224365	2,966497
2S	1455	1449	1446	1449	1,512077	2,016563
1S	2890	2928	2922	2922	0,749829	1
M=S	2856	2894	2862	2862	0,765549	1,020964
C	2191	2220	2187	2191	1	1,333638

Table 7 Test 4, Quicksilver, S1, HD, 901 F, 27/04/19

	A	B	C	Median	Speedup C	Speedup 1S
10S	140	134	136	136	2,154412	6,889706

7S	160	180	176	176	1,664773	5,323864
5S	234	238	236	236	1,241525	3,970339
4S	281	275	275	275	1,065455	3,407273
3S	352	347	354	352	0,832386	2,661932
2S	493	491	494	493	0,59432	1,900609
1S	956	937	935	937	0,3127	1
M=S	942	930	930	930	0,315054	1,007527
C	298	273	293	293	1	3,197952

Table 8 Test 5, Arnold, S2, HD, 501 F, 27/04/19

	A	B	C	Median	Speedup C	Speedup 1S
10S	1173	1161	1173	1173	8,41347	9,342711
7S	1663	1643	1625	1643	6,006695	6,670116
5S	2241	2245	2235	2241	4,403838	4,890228
4S	2778	2783	2792	2783	3,546173	3,937837
3S	3685	3708	3696	3696	2,670184	2,965097
2S	5521	5541	5598	5541	1,781086	1,977802
1S	10915	11096	10959	10959	0,900538	1
M=S	11134	11064	11131	11131	0,886623	0,984548
C	9882	9842	9869	9869	1	1,110447

Table 10 Test 6, ART, S2, HD, 501 F, 27/04/19

	A	B	C	Median	Speedup C	Speedup 1S
10S	394	419	411	411	5,109489	7,987835
7S	534	537	538	537	3,910615	6,113594
5S	703	711	711	711	2,953586	4,61744
4S	867	858	868	867	2,422145	3,786621
3S	1148	1146	1138	1146	1,832461	2,864747
2S	1671	1665	1672	1671	1,256732	1,964692
1S	3253	3287	3283	3283	0,639659	1
M=S	3298	3305	3295	3298	0,63675	0,995452
C	2100	2121	2071	2100	1	1,563333

Table 11 Test 7, Quicksilver, S2, HD, 501 F, 27/04/19

	A	B	C	Median	Speedup C	Speedup 1S
10S	135	122	126	126	3,079365	5,404762
7S	156	154	159	156	2,487179	4,365385
5S	205	193	204	204	1,901961	3,338235
4S	208	213	209	209	1,856459	3,258373
3S	273	258	269	269	1,442379	2,531599
2S	366	372	375	372	1,043011	1,830645
1S	678	680	681	680	0,570588	1,001471
M=S	687	685	678	685	0,566423	0,994161

C	397	371	388	388	1	1,755155

Table 13 Test 8, Arnold, S3, HD, 501 F, 28/04/19

	A	B	C	Median	Speedup C	Speedup 1S
10S	1124	1120	1121	1121	10,06155	9,621766
7S	1588	1590	1586	1588	7,102645	6,792191
5S	2211	2196	2201	2201	5,124489	4,9005
4S	2729	2761	2802	2761	4,085114	3,906556
3S	3608	3629	3666	3629	3,108019	2,972169
2S	5431	5409	5441	5431	2,076781	1,986006
1S	10760	10802	10786	10786	1,045707	1
M=S	11065	11078	11146	11078	1,018144	0,973641
C	11279	11130	11490	11279	1	0,95629

Table 13 Test 9, ART, S3, HD, 501 F, 28/04/19

	A	B	C	Median	Speedup C	Speedup 1S
10S	363	359	361	361	8,457064	8,65928
7S	487	493	487	487	6,268994	6,418891
5S	677	676	662	676	4,516272	4,62426
4S	827	804	808	808	3,778465	3,868812
3S	1087	1087	1099	1087	2,808648	2,875805
2S	1592	1584	1578	1584	1,927399	1,973485
1S	3126	3150	3115	3126	0,976647	1
M=S	3229	3254	3203	3229	0,945494	0,968102
C	3046	3053	3091	3053	1	1,023911

Table 13 Test 10, Quicksilver, S3, HD, 501 F, 28/04/19

	A	B	C	Median	Speedup C	Speedup 1S
10S	100	100	103	100	2,96	6,1
7S	137	142	135	137	2,160584	4,452555
5S	148	151	149	149	1,986577	4,09396
4S	192	191	194	192	1,541667	3,177083
3S	251	248	251	251	1,179283	2,430279
2S	320	324	326	324	0,91358	1,882716
1S	614	608	610	610	0,485246	1
M=S	616	615	616	616	0,480519	0,99026
C	299	296	294	296	1	2,060811

Table 13 Test 11, Arnold, S4, HD, 501 F, 28/04/19

	A	B	C	Median	Speedup C	Speedup 1S
10S	426	428	426	426	9,103286	8,943662
7S	589	589	588	589	6,584041	6,468591

5S	814	799	811	811	4,781751	4,697904
4S	973	976	1033	976	3,973361	3,903689
3S	1311	1315	1308	1311	2,958047	2,906178
2S	1913	1924	1939	1924	2,015593	1,980249
1S	3784	3823	3810	3810	1,017848	1
M=S	3865	3954	3906	3906	0,992832	0,975422
C	3902	3812	3878	3878	1	0,982465

Table 13 Test 12, ART, S4, HD, 501 F, 29/04/19

	A	B	C	Median	Speedup C	Speedup 1S
10S	256	246	244	246	6,308943	7,715447
7S	307	325	319	319	4,865204	5,949843
5S	424	417	432	424	3,660377	4,476415
4S	525	522	522	522	2,97318	3,636015
3S	688	681	684	684	2,269006	2,774854
2S	990	990	991	990	1,567677	1,917172
1S	1886	1944	1898	1898	0,817703	1
M=S	1935	1926	1934	1934	0,802482	0,981386
C	1535	1564	1552	1552	1	1,222938

Table 13 Test 13, Quicksilver, S4, HD, 501 F, 29/04/19

	A	B	C	Median	Speedup C	Speedup 1S
10S	55	54	55	55	1,854545	2,581818
7S	57	54	62	57	1,789474	2,491228
5S	55	54	55	55	1,854545	2,581818
4S	65	60	61	61	1,672131	2,327869
3S	74	74	73	74	1,378378	1,918919
2S	91	91	92	91	1,120879	1,56044
1S	142	142	143	142	0,71831	1
M=S	146	152	151	151	0,675497	0,940397
C	102	92	151	102	1	1,392157

Table 13 Test 14, Redshift, S5, S6, HD, 4K, 8K, 501 F, 2/05/19 – 5/05/19

10S	A	B	C	median
RSHD	2803	2810	2533	2803
RSHD	2798	2792	2813	2798
RS4k	9074	9101	9112	9101
RS8k	33120	33136	33153	33136
robotHD	428	430	427	428
robot4K	1177	1202	1206	1202
robot8K	4361	4351	4345	4351

Table X Test 15, Arnold, S6, HD, 501F, 6/05/19-07/05/19

	Time	S		Time	S

1S	52274	1	M=S	52968	1
2S	26084	2,004064	M=S+1S	26684	1,98501
3S	17505	2,986233	M=S+2S	17793	2,976901
4S	13360	3,912725	M=S+3S	13265	3,993064
5S	10561	4,949721	M=S+4S	10614	4,99039
6S	8827	5,922057	M=S+5S	8820	6,005442
7S	7636	6,845731	M=S+6S	7579	6,988785
8S	6685	7,819596	M=S+7S	6620	8,001208
9S	5924	8,824105	M=S+8S	5864	9,032742
10S	5282	9,89663	M=S+9S	5293	10,00718
11S	4830	10,82277	M=S+10S	4844	10,93476
12S	4417	11,83473	M=S+11S	4475	11,83642
13S	4133	12,64796	M=S+12S	4133	12,81587
14S	3839	13,61657	M=S+13S	3798	13,94629
15S	3553	14,71264	M=S+14S,	3551	14,91636
16S	3404	15,35664	M=S+15S	3337	15,87294
17S	3146	16,61602	M=S+16S	3131	16,91728
18S	2971	17,59475	M=S+17S	2976	17,79839
19S	2831	18,46485	M=S+18S	2840	18,6507
20S	2708	19,30355	M=S+19S	2702	19,60326
21S	2554	20,4675	M=S+20S	2567	20,6342
22S	2438	21,44135	M=S+21S	2446	21,65495
23S	2345	22,29168	M=S+22S	2340	22,6359
24S	2274	22,98769	M=S+23S	2243	23,6148
25S	2190	23,86941	M=S+24S	2179	24,3084
26S	2103	24,85687	M=S+25S	2108	25,12713
27S	2018	25,90387	M=S+26S	2016	26,27381
28S	1893	27,61437	M=S+27S	1944	27,24691
29S	1890	27,6582	M=S+28S	1896	27,93671
30S	1829	28,58065	M=S+29S	1829	28,96009
31S	1792	29,17076	M=S+30S	1783	29,70723

APPENDIX 2

The scenes used will be taken from Autodesk's MCG – What's New 2018 + Sample pack tutorial (Ashton 2017), sketchfab (Junaid_shakoor 2016, Mathiasdierickx 2016) and behance (Chukov 2019).

From the first we take the Simple Physics, Shuffle Swap, Cell Noise, Spline Twister and Cube Grid scenes. Then we altered a few details to make them demand enough for our testing. From sketchfab the robot and city environment were taken and made into a scene and finally a panoramic animation of the behance scene was prepared.

Scene 1: Cube Grid

Timeline scaled to 901 frames

Scene 2: Simple Physics

For this scene we'll select the splines and activate rendering. Timeline scaled to 501 frames

Scene 3: Cell Noise

Timeline scaled to 501 frames. Material on the object set to physical material with frosted glass preset.

Scene 4: Shuffle Swap

Timeline scaled to 501 frames

Scene 5: Robot

Timeline scaled to 501 frames. Camera does an arc around the robot.

Scene 6: Building

Timeline scaled to 501 frames. Camera animated to do a loop around the building.