Design a Cloud Expert system for performance, Cost, and Energy Management of GPU render Farm Based on Cloud User Preferences.

1st Karam Turki
Gilgamesh Studio and Lab
Baghdad, Iraq
karam.turki@gilgameshstudio.com

2nd Auday Aldulamy
name of organization (of Aff.)
City, Country
email address or ORCID

***Abstract*—This paper introduces a novel method for optimizing the selection of suitable GPU in render farms based on user preferences. The main goal of the paper is to reduce the cost and energy consumed in cloud data centers, especially those related to the rendering of images and movies. The method is based on designing an expert system residing in the cloud to take the user preferences for performance, cost, and energy consumption and read the scene rendering parameters to choose the most suitable GPUs that fit with user requirements and can handle the scene rendering with suitable resource utilization. The paper introduces a new mechanism for determining the scene complexity based on scene parameters like the number of frames and the number of polycones, then find a value that we call the complicity value, which maps the most suitable GPU that should be used with the scene. the user preferences for the render farm are also processed through the user normalization rule engine then both of these values are aggregated to obtain the most prefered available GPU to render the scene at the lowest cost, lowest energy consumption, and highest performance (Speed).**

***Index Terms*—Expert system, GPU rendering, Cloud rendering , cloud performance , energy consumption**

I. Introduction

3d animation, computer graphics (CG), and virtual/augmented reality works have many complex and time-consuming stages, but the most headache one is rendering which comes with both complexity and cost that lead studios to suffer especially when the platform owner and broadcasters go up in requiring higher resolutions. Higher resolutions like 8K , involve increasing the ongoing hardware investments which typically impact the profitability of these studios. the only solution that the studios find themselves is to switch to cloud-based rendering which is not require in advance hardware investment, nor day-by-day maintenance and support costs. But cloud rendering is expensive, time-consuming, and consumes a lot of energy, which is not acceptable for many. either to financial reasons, deadline requirements by clients , or affecting the environmental sustainability rank of the organization. The render cloud provider methodology, which say one fits all is no longer acceptable and needs to be changed to more user-friendly criteria that satisfy the cloud user, and maintain the profitability of the cloud provider. In the proposed system, the user will have the ability to specify his preferences (performance, cost, energy), and the cloud provider can automatically check the scene complexity and decide which GPU should fit with this scene, both user input and cloud provider program output are aggregated in rule engine mechanism of the proposed expert system to give the optimal available fit of GPU for each of user scenes. this will lead to a win-win relationship between users and rendering cloud providers.

II. Background

In This section, we will introduce some of the principles background and explain some paper-related subjects.

A. GPU Rendering

unlike the CPU(central processing unit), the GPU(Graphical Processing Unit) accomplishes all the processes in a parallel way, thanks to too many cores and threads that they have. For 3D animation especially the CPU rendering is not sufficient and we have to go to a more efficient way of rendering which is GPU rendering.

In GPU rendering, each GPU can handle one frame and these frame parts can also be divided to cover many cores and each thread can take care of one property. In cloud-based GPU rendering, some factors make it preferred for studios over in-house workstations, these factors are the scalability, accessibility, reduced hardware requirements, and collaboration/ sharing. The cloud GU render farm consists of GPU nodes and each node contains many different GPUs all sharing one huge pool of working memory that is named cloud memory. these nodes are connected with management consul and storage via a high-speed switch that is connected to the web through a firewall as shown in Fig (1). In this paper, we introduce 10 different GPUs that are hypothetically named, but they reflect actual GPUs that are in the market today. with each GPU we attached the rank of performance, core count, memory, power consumption, cost of use in US dollars per hour, and number of identical GPUs in the node.

B. Expert systems

Expert systems are one branch of artificial intelligence that uses the knowledge of an expert in a certain field to build an

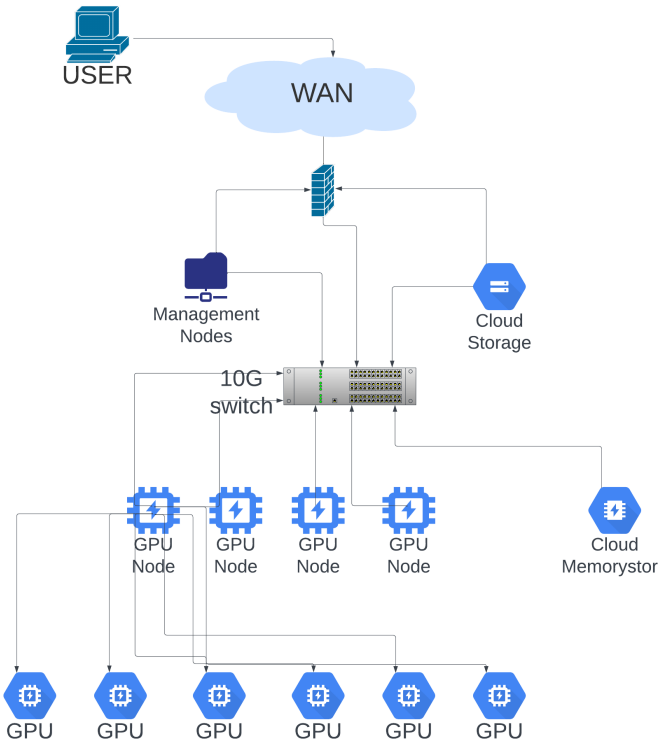

Fig. 1. Render Farm Architecture based on GPU

automatic real-time inference and decision-making system. To build an expert system, we need a knowledge base that can be obtained from experts, and inference rules that are used to make decisions based on the inputs from the users and the knowledge base, the inference rules are designed mutually between the expert system designer and experts in the field that intend to build an expert system for it. usually, expert systems work well with system that have no ambiguous parts and the decision process is known even if there is many stages in making these decisions[] [] [] [] [] []. The expert system generally consists of the following parts, but each expert system doesn't need to have them all:

1-Knowledge Base,
2- Rules Inference Engine,
3-User Interface,
4-Working Memory,
5-Explanation Facility,
6-Knowledge Acquisition and Learning Module.

In this paper, we will concentrate on the first main three parts because this is all that we will need in this work.

Cloud-Based Expert System (CBES) model for decision-making is studied extensively in the last few years [][][][][][]. But there is no work introducing the use of the expert system for GPU cloud rendering, which is essential in the coming years to reduce the cost and environmental impact of rendering in the cloud and all cloud applications in general.

C. Scene Complexity

There are many factors affecting the scene complexity, and that is related to the efforts and resources needed for rendering these scenes.the importance of these parameters in the underhand work is that it is affect the behavior and results of the rendering more than the effects of user preferences for rendering. these parameters can be summarized as in the following:

1- Number of Polygons (Poly)
2- Vertex Count (Ver)
3- Object Count (Obj)
4- Texture Resolution (Res)
5- Number of Light Sources (Lit)
6- Number of Materials (Mat)
7- Animation Complexity (Ani)
8- Environmental Effects (Eff)
9- Number of Frames (Fra)

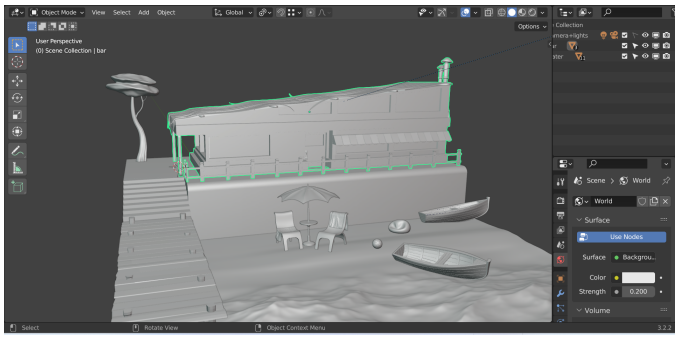

Fig. 2. Scene in blender environment

In Blender and other software like Maya, we can read these parameters manually, but there is also the ability to read these parameters automatically by Python program. When the system API or routine uses Python to read these parameters, then attention must be taken to using pyaaimp library because it makes the procedure easier to program. Fig(2) shows a simple scene in Blender and most of the scene parameters can be read in this software

D. GPU Types used in the renderfarm

many different kinds of advanced GPU are used nowadays in render farms, and all of these work properly. we take specifications from different brands and models and give them hypothetical names. Throughout this paper when we mention the performance of GPU, we mean the speed performance or simply the speed of GPU which is related to the operating frequency of the processor, no. of cores in the processor, and the working memory specified with this GPU. Also, the cost or price of the GPU is the cost of operating and utilizing this resource for one hour. Finally, the Energy consumption of the GPU is the energy required for utilization of the GPU and it is measured in Kwh.

III. Proposed System

The proposed system in this paper is focused on developing a cloud exert system that aggregates the user preferences for performance, cost, and energy consumption and the system requirements for rendering a complex or simple scene. So the user is not 100 percent free to choose the metric when his choice does not fit with the actual requirements for the scene. The proposed criteria will be based on obtaining the best GUP fit for the scene parameters and the best fit for the user preference, then aggregate to obtain the best GPU fit based on the knowledge base that contains all GPUs information and specifications. Fig (3) shows flow-diagram for the proposed system that consist of the working blocks that will be explained below.

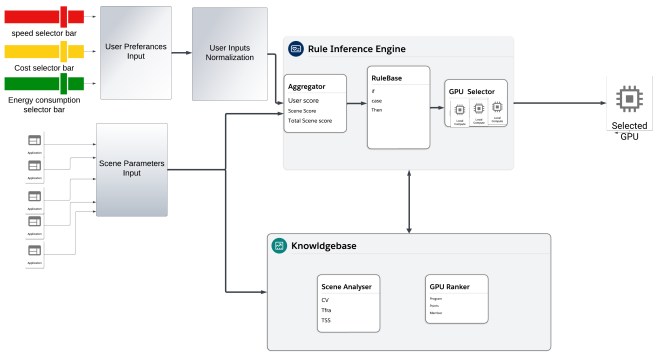

Fig. 3. Proposed Expert system flow diagram

A. User Preferences Inputs

the user can optimize three tuning parameters as follows:
1- the user can adjust the desired level of performance in the range 1 to 10.
2- the user can adjust the desired level of cost depending on his budget in the range 1-10.
3- the user can adjust the desired level of energy consumption if his application is not urgent in the range of 1-10.

These Values have to be aggregated with the values of scene complexity, therefore these values have to be normalized and weighted to reflect their desired effects on the result of choosing GPU.

B. Normalization and weighting of User inputs

We introduce the normalization of user parameters to ensure that all user inputs can be compared and combined to a uniform scale. We will divide each input by 10 to transform the range from 0.1 to 1. So the speed performance (SP), Cost (C), and Energy (E) will be normalized as follows:

$$\text{Normalized Speed Performance} = SP^N = \frac{SP}{10} \tag{1}$$

$$\text{Normalized Cost} = C^N = \frac{C}{10} \tag{2}$$

$$\text{Normalized Energy Consumption} = E^N = \frac{E}{10} \tag{3}$$

Now we will specify a weighting factor for each parameter to reflect its importance. For the sake of this paper, we will assume these parameters have the same importance, but practically the cloud owner can change these weights or delegate to the user this privilege.

$$\text{Weight for Speed Performance} = wSP = \frac{1}{3} \tag{4}$$

$$\text{Weight for Cost} = wC = \frac{1}{3} \tag{5}$$

$$\text{Weight for Energy Consumption} = wE = \frac{1}{3} \tag{6}$$

$$\text{Score for weighted Speed Performance} = SwSP = SP^N \times wSP \tag{7}$$

$$\text{Score for weighted Cost} = SwC = C^N \times wC \tag{8}$$

$$\text{Score for weighted Energy Consumption} = SwE = E^N \times wE \tag{9}$$

Now we reach the stage of aggregation for the scores. note these values can be transferred separately to the next stage, and aggregated separately with scene parameters. for simplicity, we will aggregate user preference at this stage and transfer only one value for the user, that we will call it, Aggregated User Score (AUS) which can be obtained as follows:

$$AUS = SwSP + SwC + SwE \tag{10}$$

C. proposed system Knowledge base

1) Scene Complexity Analyser: Scene complexity is a hard area to handle because of the many parameters affecting it and the different ranges these parameters fill in. So we will introduce a way to quantitative this complexity and we call the final value obtained from this method the complexity value. the method for obtaining the complexity Value is summarized by aggregating each mapped weighted value from the mapping scales in Fig(4) for each scene , so the complexity Value (CV) for the Scene (i) is

$$CV = (wPoly + wVer + wObj + wRes + wLit + wMat + wAni + wEff)/8 \tag{11}$$

the total number of all scenes frames is:

$$Tfra = \sum_{i=1}^{n} \text{Fra}_i \tag{12}$$

The aggregated Scene score (ASS) for the scene (i) is the sum of the complex value of scene i (CV) and the frame value of scene i (FV):

$$ASS_i = AC_i + Fra_i/Tfra \tag{13}$$

Now we should find the speed rank of each scene as follow:

$$\text{scene\_speed\_rank}_i = \lfloor |\text{ASS}_i \times 10| \rfloor \tag{14}$$

and the total speed rank of all scenes can be found as follow:

$$1 < \frac{\sum_{i=1}^{n} \text{scene\_speed\_rank}_i}{n} < 10 \tag{15}$$

where n : is number of scenes.

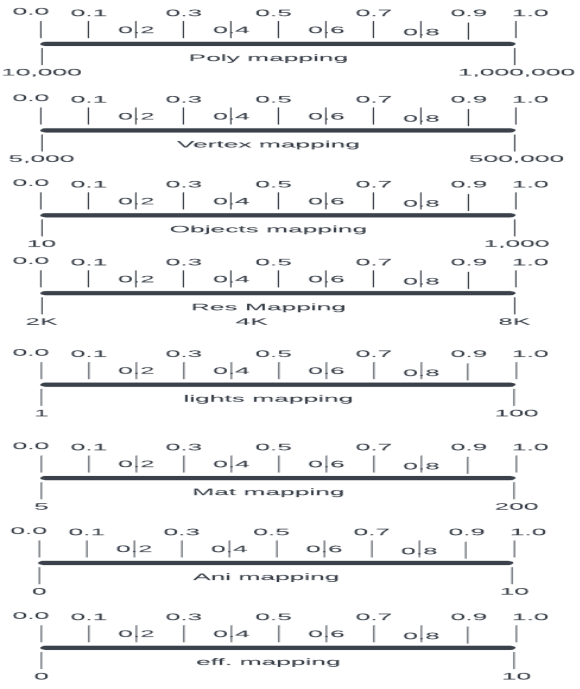

Fig. 4. Mapping of scene parameters to weighted values

2) GPU Ranker: As mentioned in the background, we will use the GPUs that appear in the following table with hypothetical names but practical specifications. the most high performance shall be given the high-rank GPU. The firing (Selecting) of the GPU from this table shall be depend on availability, user preferences, and Scene complexity.

Rank	GPU Model	Core Count	Memory	Power Consumption	price($)	avialable number	Performance Notes
10	spark 9050	10400	24GB	350W	1	10	high speed
9	thinder 8020	8700	20GB	320W	0.9	50	high speed
8	View 5000	8200	16GB	300W	0.8	40	high speed
7	skylight 800	7600	16GB	290W	0.7	22	high speed/low cost
6	Hell 20	6800	12GB	280W	0.6	77	low cost
5	lightpower 2020	6400	11GB	250W	0.5	8	low cost
4	steam 666	5200	8GB	225W	0.4	30	low cost
3	fastcal 30	4500	6GB	160W	0.3	11	low cost/low consumption
2	TUT 8000	1200	4GM	130W	0.2	17	low consumption
1	TUT4000	870	4GM	110W	0.1	20	low consumption

Fig. 5. GPU Specifications and ranker table

article

D. Rule Inference Engine

INFERENCE ENGINE RULES:

Rule 1: Ultimate Rank Greater Than 7
If the ultimate rank is greater than 7:

- Rank Type: Speed Rank
- Rank Value: Maximum of ultimate rank and user speed rank.

Rule 2: AUS Less Than 5
If AUS is less than 5:

- Rank Type: Speed Rank
- Rank Value: Ultimate rank.

Rule 3: AUS Greater Than 5 and Ultimate Rank Less Than 7
If AUS is greater than 5 and ultimate rank is less than 7:

- Rank Type: Maximum of Speed Rank, Cost Rank, and Energy Rank.
- Rank Value: Value associated with the determined rank type.

INFERENCE ENGINE LOGIC:

Determine Rank Type:
Check Rule 1, if true, set Rank Type to Speed Rank.
Check Rule 2, if true, set Rank Type to Speed Rank.
Check Rule 3, if true, determine the rank type based on the maximum value among Speed Rank, Cost Rank, and Energy Rank.

Determine Rank Value:
If Rank Type is Speed Rank:

- If Rule 1 is true, set Rank Value to the maximum of ultimate rank and user speed rank.
- If Rule 2 is true, set Rank Value to ultimate rank.
- If Rule 3 is true, set Rank Value to the value associated with Speed Rank.

If the Rank Type is Cost Rank (from Rule 3):

- Set Rank Value to the value associated with Cost Rank.

If Rank Type is Energy Rank (from Rule 3):

- Set Rank Value to the value associated with Energy Rank.

IV. SYSTEM IMPLEMENTATION, RESULTS AND DISCUSSIONS

A. System Implementation

the core proposed system has been coded by Python using Streamlit library for web user front-end design and implementation. all project documents and code are also hosted in our project GitHub repository. so the app can be accessed by clicking the app link in the Streamlit platform which we connected with our code in Git Hub. the user interface/experience (UI/UX) for the proposed system is simple and easy to follow and consists of the following parts:

- User inputs
  1- The user uploads the scenes parameters file and GPU specifications file.
  2- User selects the desired values for user preferences (speed, cost, and energy).

3- The user has the option to give normal preferences and system analysis to optimize the selected GPU or go with the ultimate choice, which radically obeys the user's ultimate approach to a certain desire.

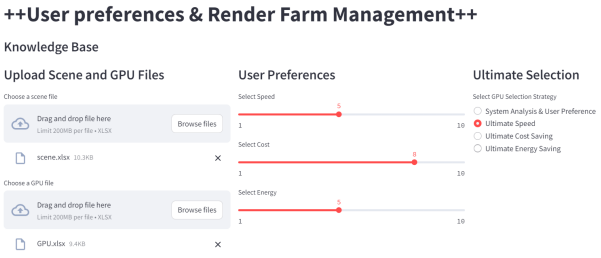

Fig. 6. Proposed system web UI

- system analysis
  Based on Scene parameters

Processed Scene Data

	scene	polygons	vertex	objects	light	materials	frames	Wpolygons	Wvertex	Wobject	Wlight	Wmaterials	CV	FV	ASS	scene_speed_rank
0	1	715,296	394,033	765	66	106	56	0.8	0.8	0.8	0.7	0.6	0.74	0.0132	0.7532	7
1	2	413,540	36,881	939	62	60	171	0.5	0.1	1	0.7	0.3	0.52	0.0403	0.5603	5
2	3	133,829	41,483	927	24	183	103	0.2	0.1	1	0.3	1	0.52	0.0243	0.5443	5
3	4	789,513	442,030	604	68	144	345	0.8	0.9	0.6	0.7	0.8	0.76	0.0814	0.8414	8
4	5	210,041	240,895	675	39	38	218	0.3	0.5	0.7	0.4	0.2	0.42	0.0514	0.4714	4
5	6	854,302	164,220	176	80	89	335	0.9	0.4	0.2	0.8	0.5	0.56	0.079	0.639	6
6	7	914,699	69,118	168	96	99	81	1	0.2	0.2	1	0.5	0.58	0.0191	0.5991	5
7	8	348,537	405,149	269	65	182	413	0.4	0.9	0.3	0.7	1	0.66	0.0974	0.7574	7
8	9	971,024	190,478	576	66	68	193	1	0.4	0.6	0.7	0.4	0.62	0.0455	0.6655	6
9	10	642,986	276,801	644	82	59	140	0.7	0.6	0.7	0.9	0.3	0.64	0.033	0.673	6

Fig. 7. Proposed system analysis for scene file

B. Results:

The above software (platform) and its documentation can be accessed through the GitHub link in [].after running the software you can try it with different Scenario and test the methodology for balancing between the user preferences and scene parameters for specifieying rank type and rank value in order to choose the most suitable GPU matching. Below is one run figure and table showing different scenarios.

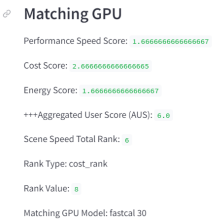

Fig. 8. GPU Matching results

C. Discussion and Conclusion:

When the User Aggregated score (AUS) is under five that means the users have no interest in optimizing for the speed, cost, or energy. and leave the choice to the system to optimize for him. But when the aggregated Scenes score is above 8, then the system shall optimize for speed rank, no matter the choice of the user preferences because the scenes are heavy and need computation resources.

Also if the user choice is to go for the ultimate method then the chosen user preference will be the rank type with a maximum rank value equal to 10. This case has to be limited by the render farm operator if the computation resources are limited because this will affect all the render farm operations and prevent it from profitability. when the above cases are not chosen then the analysis and GPU matching shall be dependent on the computation of rank type and rank value of the user preferences as described in the proposed section. As a conclusion for the results presented, we give the tools for the user to cover all his interests, and also give the render farm operator the tools to ensure profitability without pressing the user preferences whether it is performance speed, Cost, or reducing energy consumption.

user preferences			Ultimate User preferences			Scene speed	Rank Type	Rank Value	GPU Matching
Speed	Cost	Energy	Speed	Cost	Energy	total rank			
8	4	5	✓	✗	✗	6	speed	10	Spark 9050
7	9	7	✗	✓	✗	6	Cost	10	TUT 4000
5	4	6	✗	✗	✓	6	Energy	10	TUT 4000
8	4	5	✗	✗	✗	6	Speed	8	View 5000
7	9	7	✗	✗	✗	6	Cost	9	TUT 8000
5	4	6	✗	✗	✗	6	Energy	6	lightpower 2020
4	5	2	✗	✗	✗	6	Speed	6	Hell 20
8	6	6	✗	✗	✗	9	Speed	9	thinder 8020

Fig. 9. Results table for different cases

D. Future scopes

The future scope for the proposed method can be summarized in the following directions:

1- discuss and test the availability of many users with limited computation capabilities, because, in the current work, we assume the availability of the resources.

2- introduce further Scene parameters that are not numerically quantified, like the animation difficulty, and whether there are/no visual effects.

3- Introduce a more deep and professional pricing strategy.

4- introduce electricity peak load in pricing and cost efficiency, and how much footprint can be reduced by this factor.

5- Use deep learning large language model to discuss with the user the best GPU that fits his requirements.

REFERENCES

This section...

REFERENCES

[1] G. Eason, B. Noble, and I. N. Sneddon, "On certain integrals of Lipschitz-Hankel type involving products of Bessel functions," Phil. Trans. Roy. Soc. London, vol. A247, pp. 529–551, April 1955.

[2] J. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73.