Oct 9, 2024

CS 350 DISCUSSION 5

NETWORK OF QUEUES

You are studying the performance of an Audio/Video Encoder which processes encoding requests coming to the system, on average, every 0.5 seconds. The inter-arrival of requests appears to be exponentially distributed. The A/V Encoder has three resources that it internally uses to process requests, namely a CPU for general processing, a GPU for video encoding, and a DSP (Digital Signal Processor) for audio encoding. Each request is comprised of a sequence of chunks that need to be processed sequentially.

The first chunk of a new request always requires processing at the CPU. After a CPU chunk is processed, there are three possibilities: (1) the request might not require any further processing (and be considered done at the encoder) with probability 0.4. Otherwise, (2) with probability 0.2, the next chunk requires processing at the GPU. Alternatively, (3) the next chunk requires DSP processing.

After processing at the GPU, the request might be done and leave the system with probability 0.6. Similarly, after a chunk is completed at the DSP, the request is completed and it leaves the system with probability 0.7. It is always the case that, if a request did not complete after a GPU or DSP chunk, the next chunk will require processing at the CPU. If this happens, the request goes back to the CPU and continues from there as if it was a new request.

System Resource	**CPU**	**GPU**	**DSP**
Avg. Processing Time (ms)	300	1600	950

Table 1: Average processing time of CPU, GPU, and DSP chunks.

You have measured the average processing time of CPU, GPU, and DSP chunks. They turned out to be exponentially distributed and provided in Table 1.

Problem 12.2 (Textbook)

You are studying the performance of an Audio/Video Encoder which processes encoding requests coming to the system, on average, every 0.5 seconds. The inter-arrival of requests appears to be exponentially distributed. The A/V Encoder has three resources that it internally uses to process requests, namely a CPU for general processing, a GPU for video encoding, and a DSP (Digital Signal Processor) for audio encoding. Each request is comprised of a sequence of chunks that need to be processed sequentially.

The first chunk of a new request always requires processing at the CPU. After a CPU chunk is processed, there are three possibilities: (1) the request might not require any further processing (and be considered done at the encoder) with probability 0.4. Otherwise, (2) with probability 0.2, the next chunk requires processing at the GPU. Alternatively, (3) the next chunk requires DSP processing.

After processing at the GPU, the request might be done and leave the system with probability 0.6. Similarly, after a chunk is completed at the DSP, the request is completed and it leaves the system with probability 0.7. It is always the case that, if a request did not complete after a GPU or DSP chunk, the next chunk will require processing at the CPU. If this happens, the request goes back to the CPU and continues from there as if it was a new request.

System Resource	**CPU**	**GPU**	**DSP**
Avg. Processing Time (ms)	300	1600	950

Table 1: Average processing time of CPU, GPU, and DSP chunks.

You have measured the average processing time of CPU, GPU, and DSP chunks. They turned out to be exponentially distributed and provided in Table 1.

Question 1

Draw the system diagram, with the routes that the requests can follow as they flow through the system labeled with the corresponding routing probability.

State the model you are using to model the resources and the assumptions you are making to approach this problem.

You are studying the performance of an Audio/Video Encoder which processes encoding requests coming to the system, on average, every 0.5 seconds. The inter-arrival of requests appears to be exponentially distributed. The A/V Encoder has three resources that it internally uses to process requests, namely a CPU for general processing, a GPU for video encoding, and a DSP (Digital Signal Processor) for audio encoding. Each request is comprised of a sequence of chunks that need to be processed sequentially.

The first chunk of a new request always requires processing at the CPU. After a CPU chunk is processed, there are three possibilities: (1) the request might not require any further processing (and be considered done at the encoder) with probability 0.4. Otherwise, (2) with probability 0.2, the next chunk requires processing at the GPU. Alternatively, (3) the next chunk requires DSP processing.

After processing at the GPU, the request might be done and leave the system with probability 0.6. Similarly, after a chunk is completed at the DSP, the request is completed and it leaves the system with probability 0.7. It is always the case that, if a request did not complete after a GPU or DSP chunk, the next chunk will require processing at the CPU. If this happens, the request goes back to the CPU and continues from there as if it was a new request.

System Resource	**CPU**	**GPU**	**DSP**
Avg. Processing Time (ms)	300	1600	950

Table 1: Average processing time of CPU, GPU, and DSP chunks.

You have measured the average processing time of CPU, GPU, and DSP chunks. They turned out to be exponentially distributed and provided in Table 1.

Question 1

Draw the system diagram, with the routes that the requests can follow as they flow through the system labeled with the corresponding routing probability.

State the model you are using to model the resources and the assumptions you are making to approach this problem.

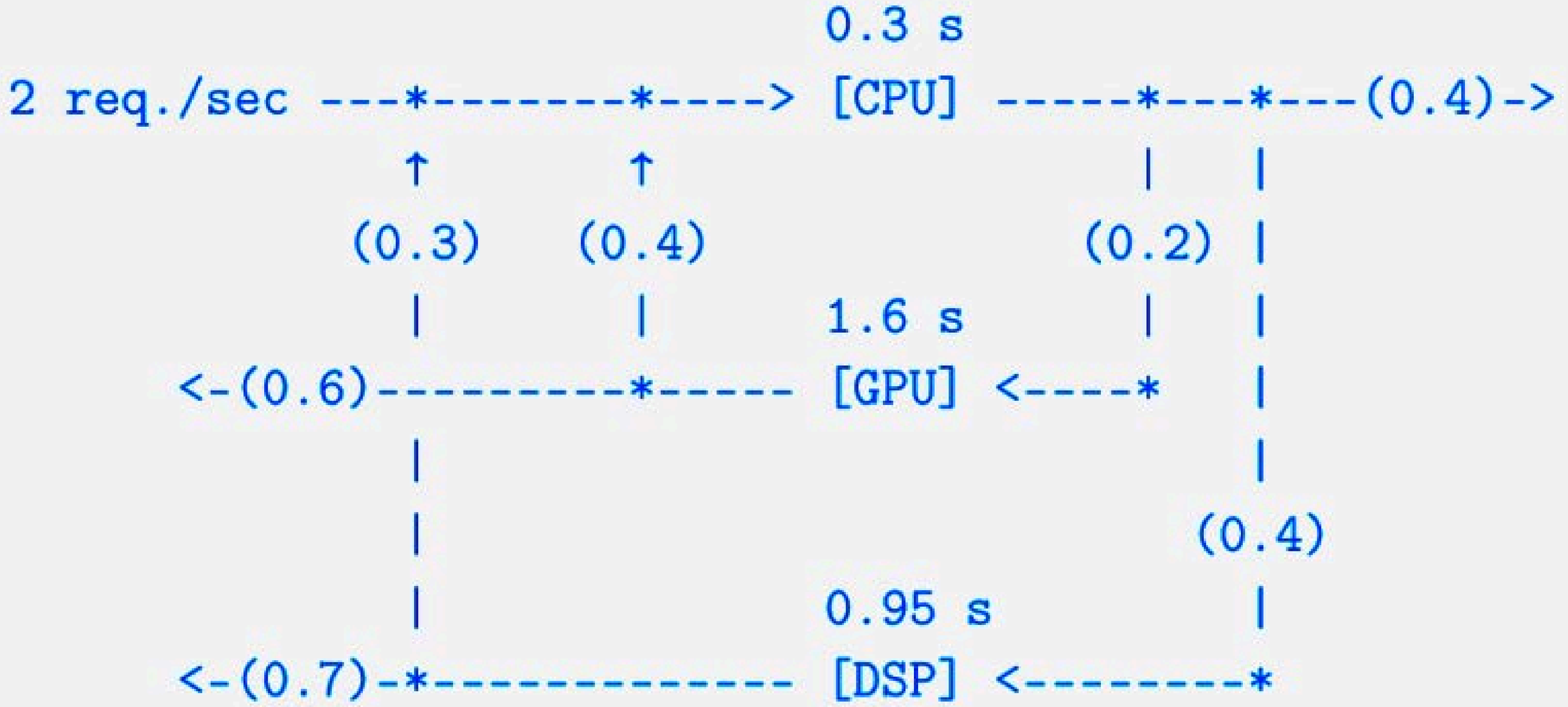
Solution

Assumptions:

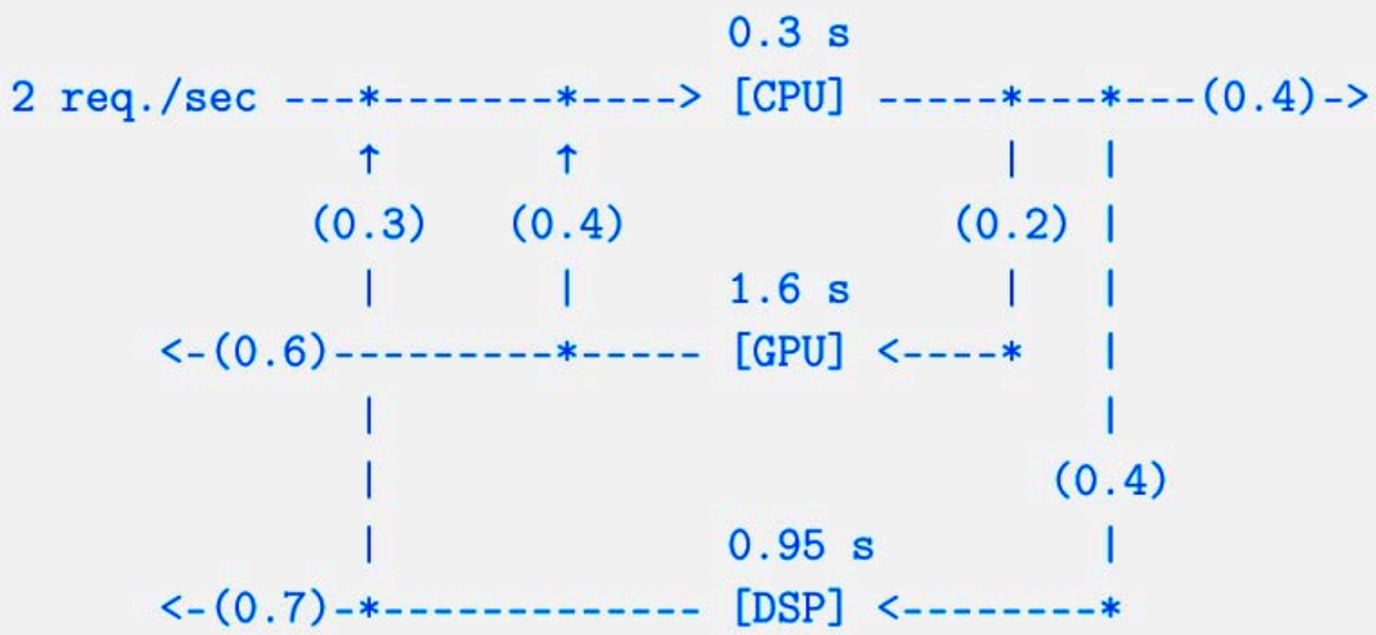
(1) CPU, DSP and GPU are M/M/1's — i.e. they have exponentially distributed service time and infinite queues;

(2) External arrivals are Poisson;

(3) System is at steady-state;



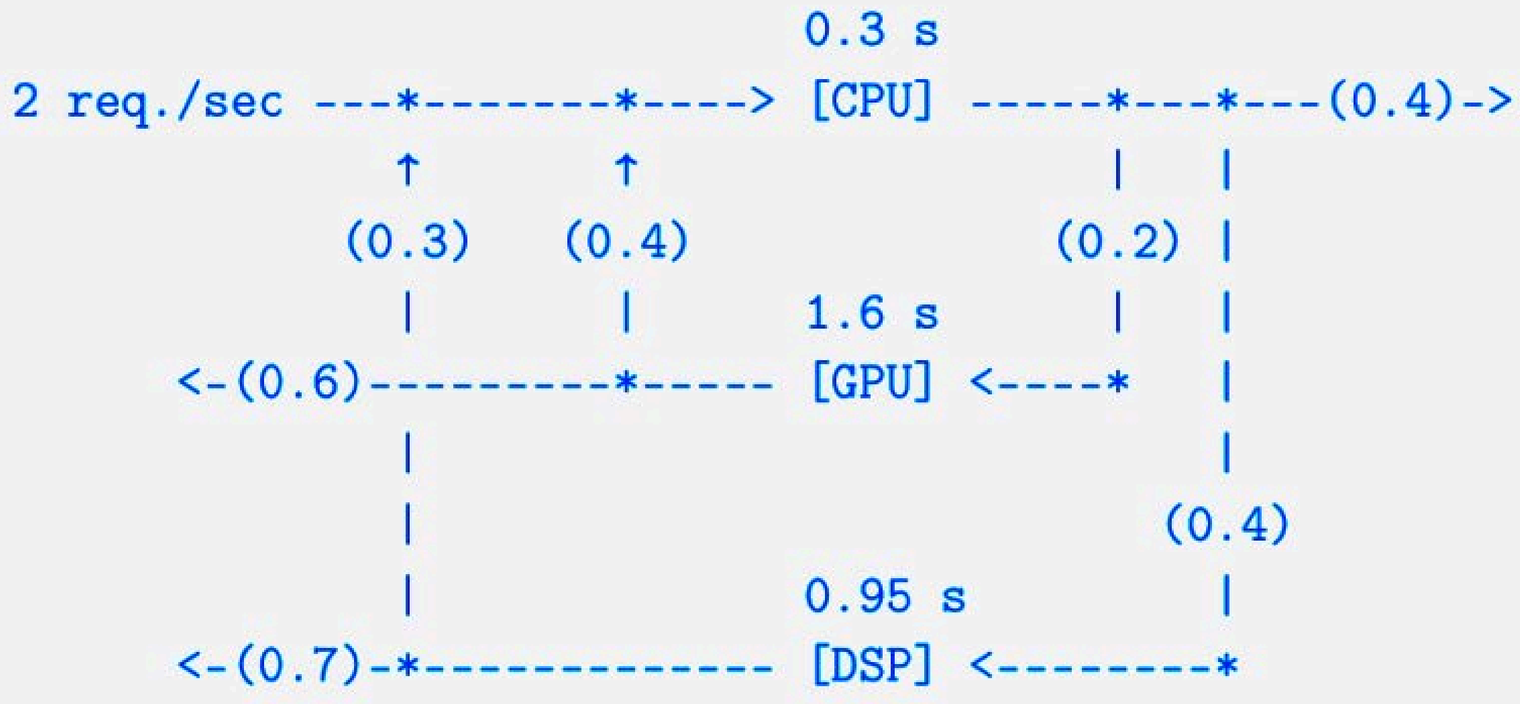
System Diagram



System Diagram

Question 2

When a request arrives the CPU and leaves the CPU, we say the CPU processes a chunk. How many chunks of CPU processing a generic request has on average?



System Diagram

Question 2

When a request arrives the CPU and leaves the CPU, we say the CPU processes a chunk. How many chunks of CPU processing a generic request has on average?

Solution

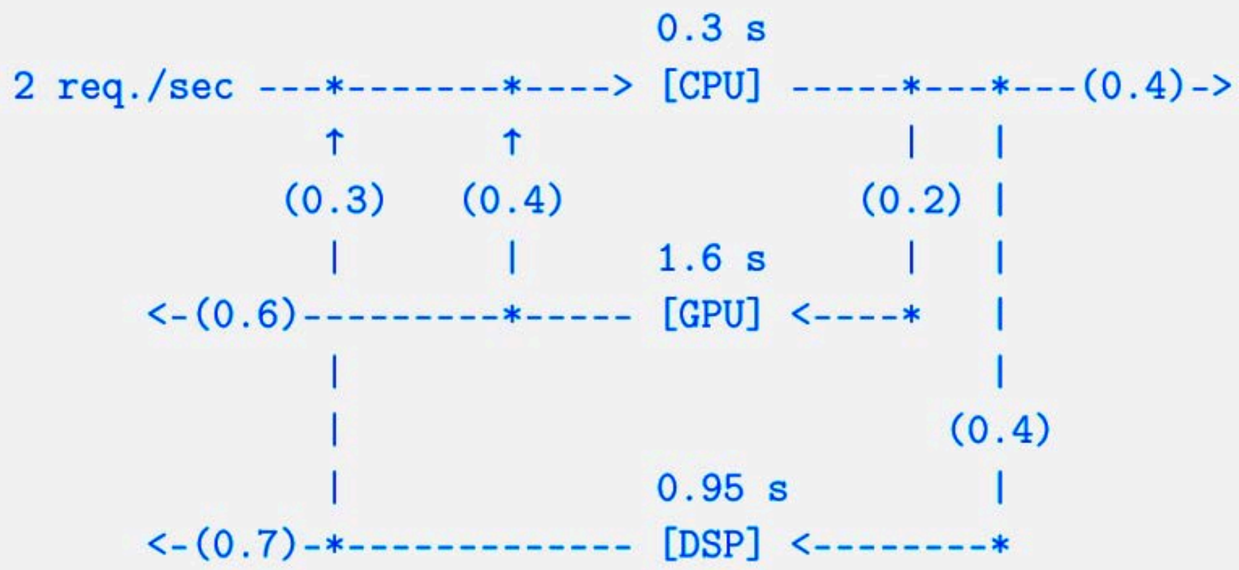
Each round of a request through the system and back to the CPU is essentially a single Bernoulli trial. After the first chunk at the CPU, the request comes back to the CPU ("fails") with probability

$$p = 0.2 \times 0.4 + 0.4 \times 0.3 = 0.2$$

The Average number of failures for a Bernoulli with p probability of failure is

$$p / (1-p) = 0.25$$

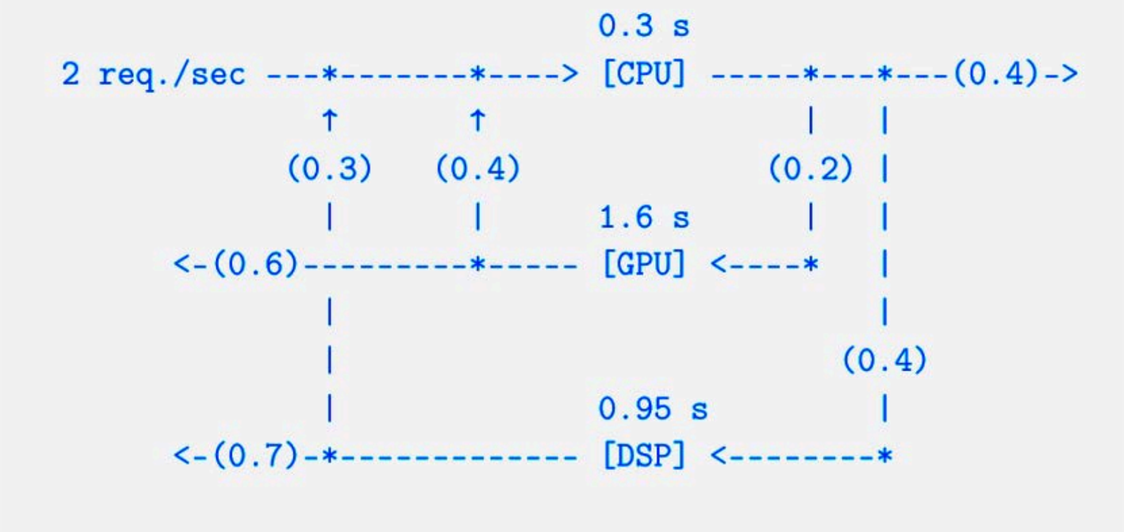
So on average a request will have 1.25 CPU chunks.



System Diagram

Question 3

Which resource represents the bottleneck in the system? Motivate your answer.



System Diagram

Question 3

Which resource represents the bottleneck in the system? Motivate your answer.

Let's first calculate the traffic through all resources. The external arrivals are $\lambda = \frac{1}{0.5} = 2$ reqs/sec

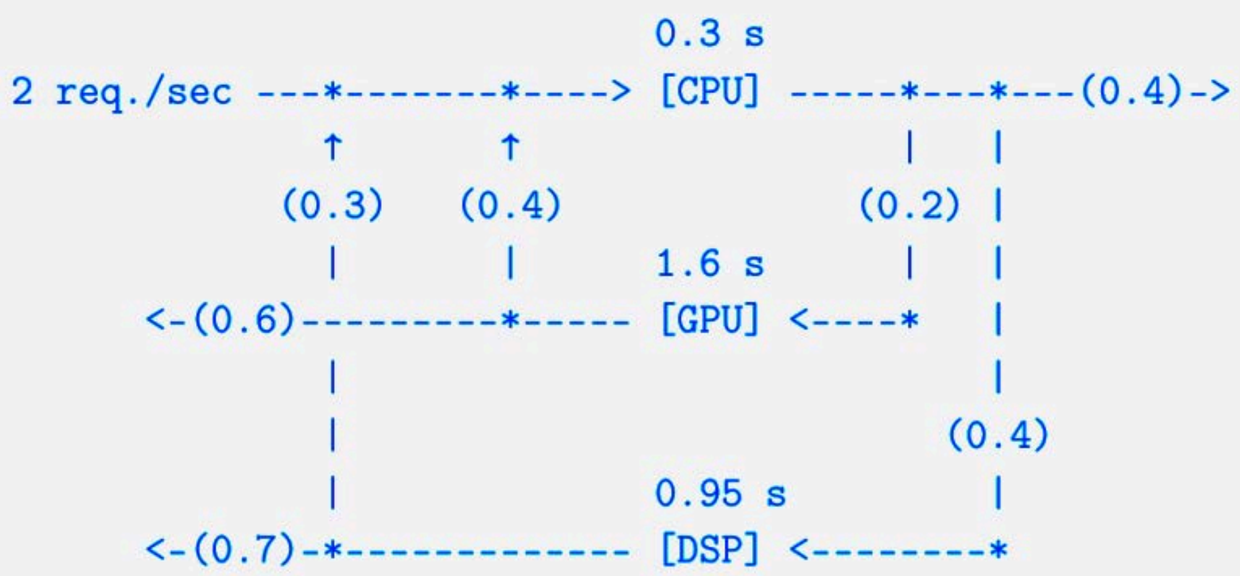
To find the bottleneck, we now need the utilization of all processes. To find that, we need to find the arrival rate into the CPU as it depends on the arrival rate of other processes:

$$\lambda_{cpu} = 2 + (\lambda_{cpu} * 0.2 * 0.4) + (\lambda_{cpu} * 0.4 * 0.3) = 2.5 \text{ reqs/sec}$$

Next, $\lambda_{gpu} = 0.2\lambda_{cpu} = 0.5$. Finally, $\lambda_{dsp} = 0.4\lambda_{cpu} = 1$.

Now we can derive the utilization for all the resources. These are $\rho_{cpu} = 2.5T_{cpu} = 0.75$, $\rho_{gpu} = 0.5T_{gpu} = 0.8$, and $\rho_{dsp} = 1T_{dsp} = 0.95$.

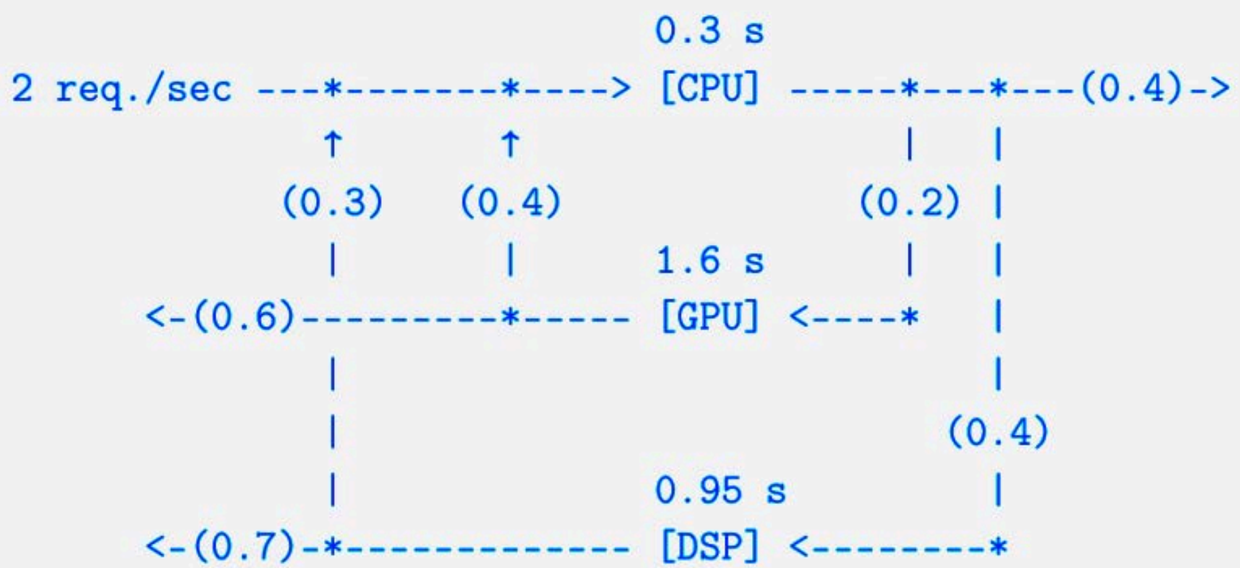
So the bottleneck is the DSP because it is the resource with the highest utilization.



System Diagram

Question 4

How many requests on average will be inside the system at steady state? That includes any request that is currently receiving service or being queued at any of the resources of the encoder.



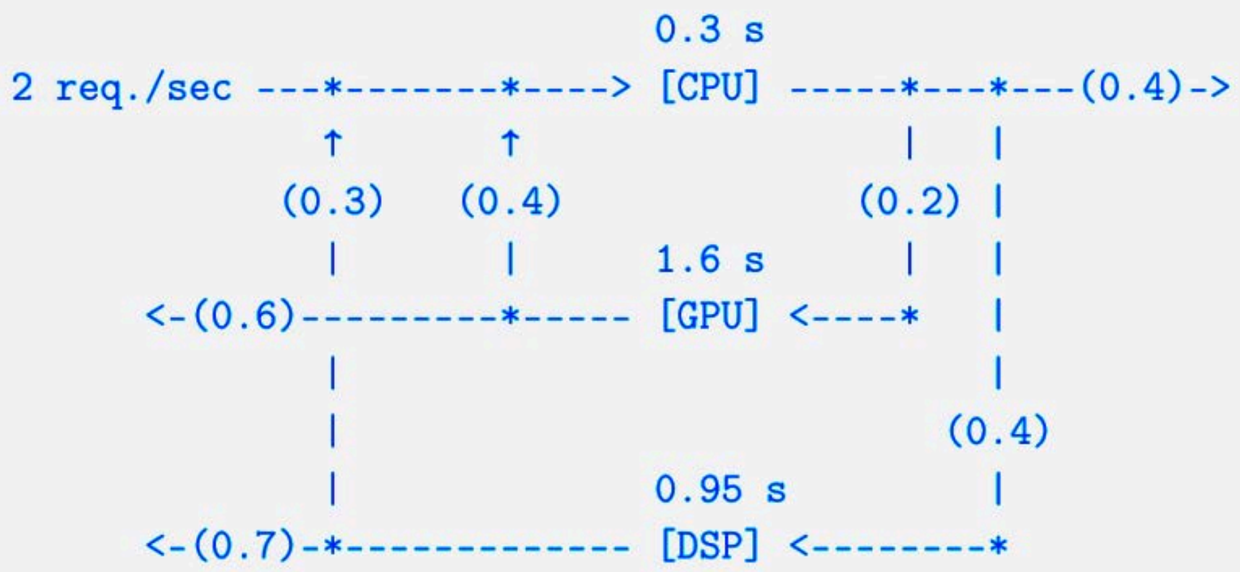
System Diagram

Question 4

How many requests on average will be inside the system at steady state? That includes any request that is currently receiving service or being queued at any of the resources of the encoder.

Solution:

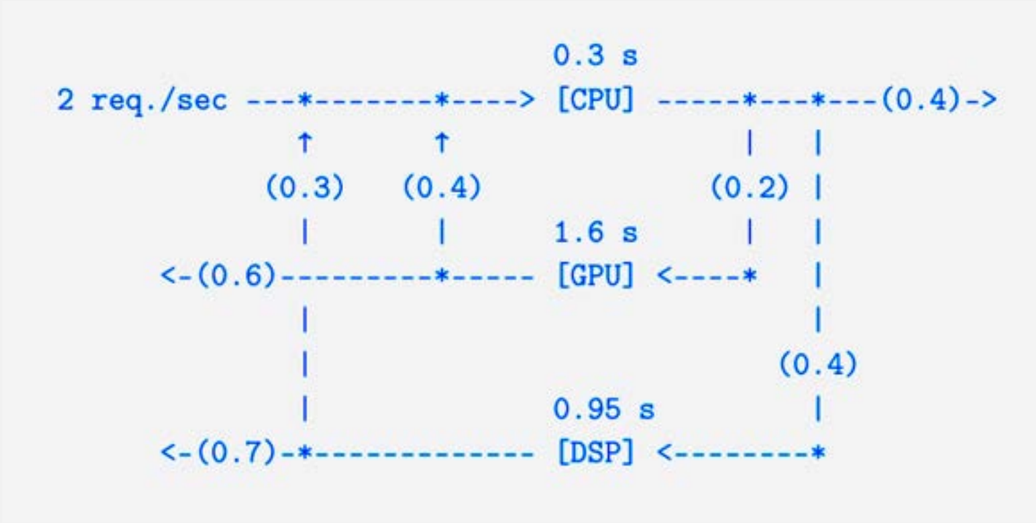
For this part, we can analyze the various M/M/1 systems in isolation leveraging the Jackson's Theorem. We derive $q_{cpu} = \frac{\rho_{cpu}}{1-\rho_{cpu}} = 3$, $q_{gpu} = 4$, and $q_{dsp} = 19$. So the total average number of requests in the system is $q_{tot} = 26$.



System Diagram

Question 5

What is the capacity of the encoder?



System Diagram

Question 5

What is the capacity of the encoder?

Solution:

The capacity is the input rate λ^* that makes the bottleneck explode. Because the bottleneck is the DSP, we set the inequality $\lambda^*_{dsp} T_{dsp} < 1$. It follows that $\lambda^*_{dsp} < 1.0526$.

To solve this, let us generalize from Q3,

$$\lambda^*_{cpu} = \lambda^* + (\lambda^*_{cpu} * 0.2 * 0.4) + (\lambda^*_{cpu} * 0.4 * 0.3)$$

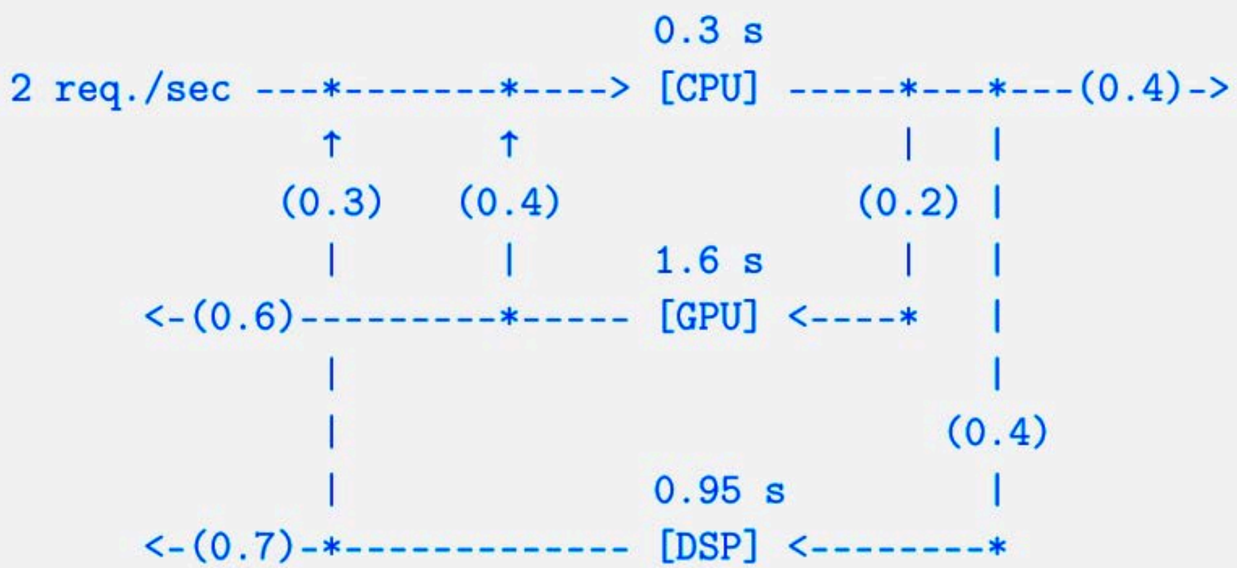
$$= \boxed{\lambda^*_{cpu} = \frac{\lambda^*}{0.8}}$$

Substitute the capacity in the following : $\lambda^*_{dsp} = 0.4 * \lambda^*_{cpu}$

$$= \frac{0.4 * \lambda^*}{0.8}$$

$$\lambda^*_{cpu} = \frac{\lambda^*}{0.8}$$

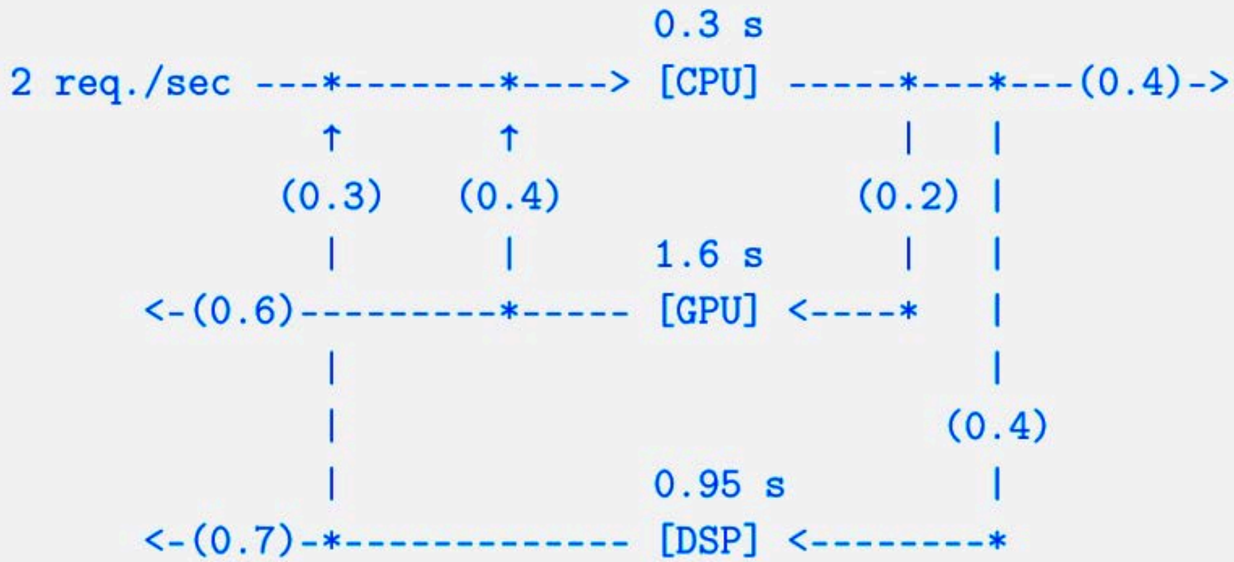
Finally, we know that $\lambda^*_{dsp} = \frac{0.4\lambda^*}{0.8}$. So the maximum traffic (i.e. capacity) that the whole system can sustain is $lambda^* = 2\lambda^*_{dsp} = 2.1052$ req./sec.



System Diagram

Question 6

Can we upgrade the GPU to be twice as fast to improve the capacity of the system?



System Diagram

Question 6

Can we upgrade the GPU to be twice as fast to improve the capacity of the system?

Solution:

Nope, because the bottleneck is the DSP, so improving the speed of the GPU will not impact the capacity of the system.

Good luck on the midterm :)