COMP9334
Capacity Planning of Computer Systems and Networks

Week 1B:

Operational analysis

Last lecture

- Solve capacity planning by solving a number of performance analysis problems

- Performance metrics
  - Response time, waiting time
  - Throughput

- Single server FIFO queue
  - A server = A processing unit

This lecture

- Queueing networks
- Operational analysis
  - Fundamental laws relating the basic performance metrics

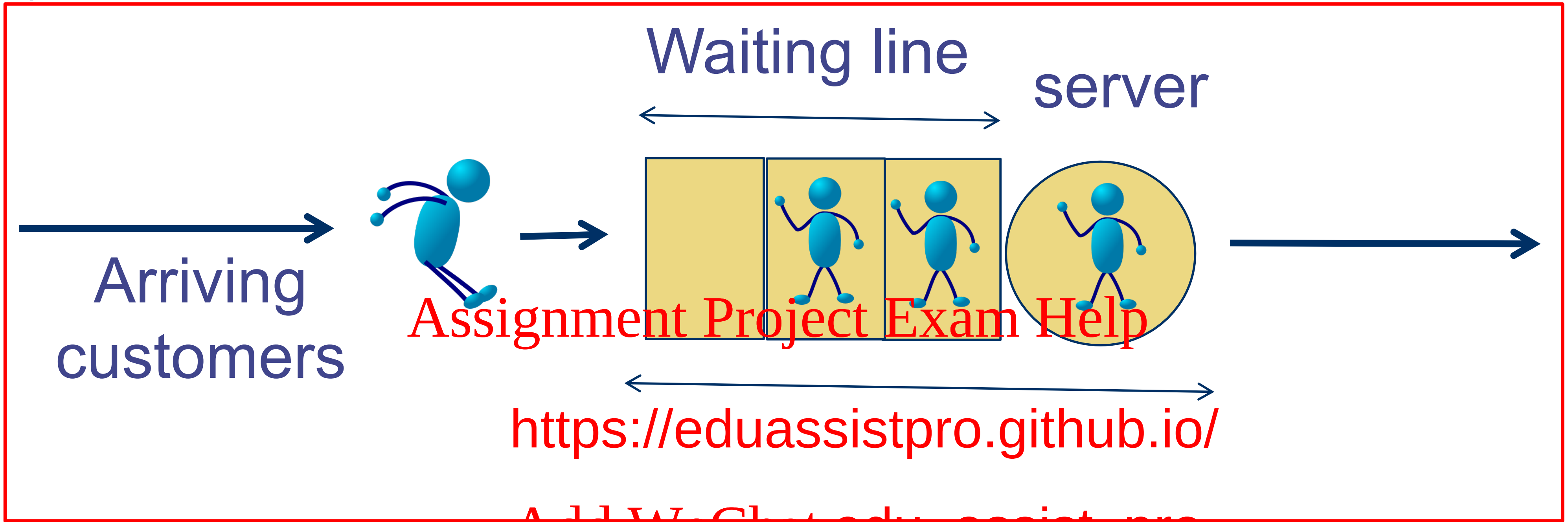
Modelling computer systems

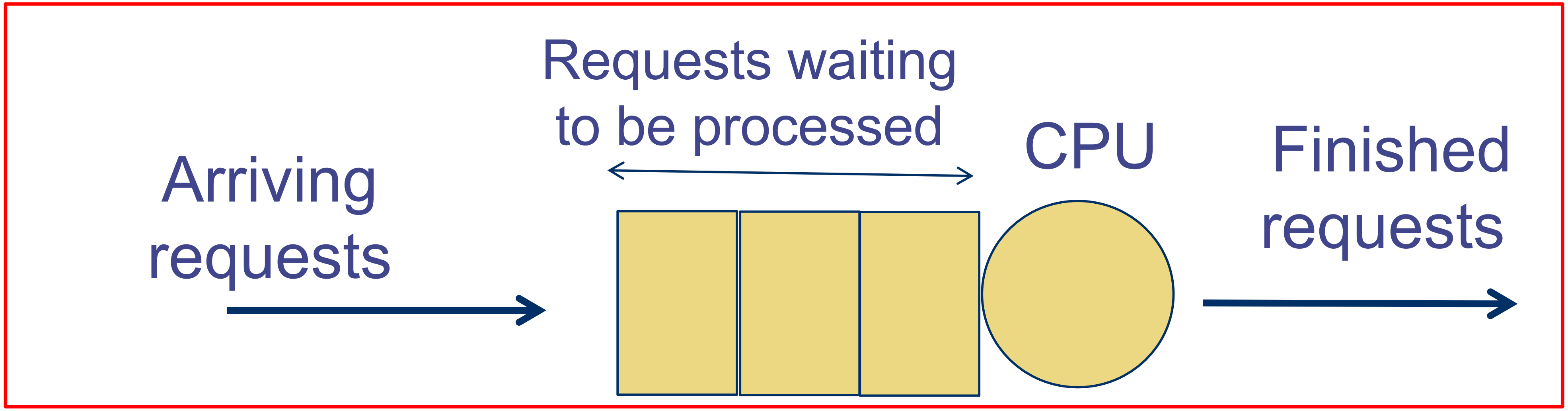
- Single server queue considers only a component within a computer system
  - A component can be a CPU, a disk, a transmission channel

- A request may require multiple resources
  - E.g. CPU, disk,

- We model a computer systems le resources by a Queueing Networks (QNs)

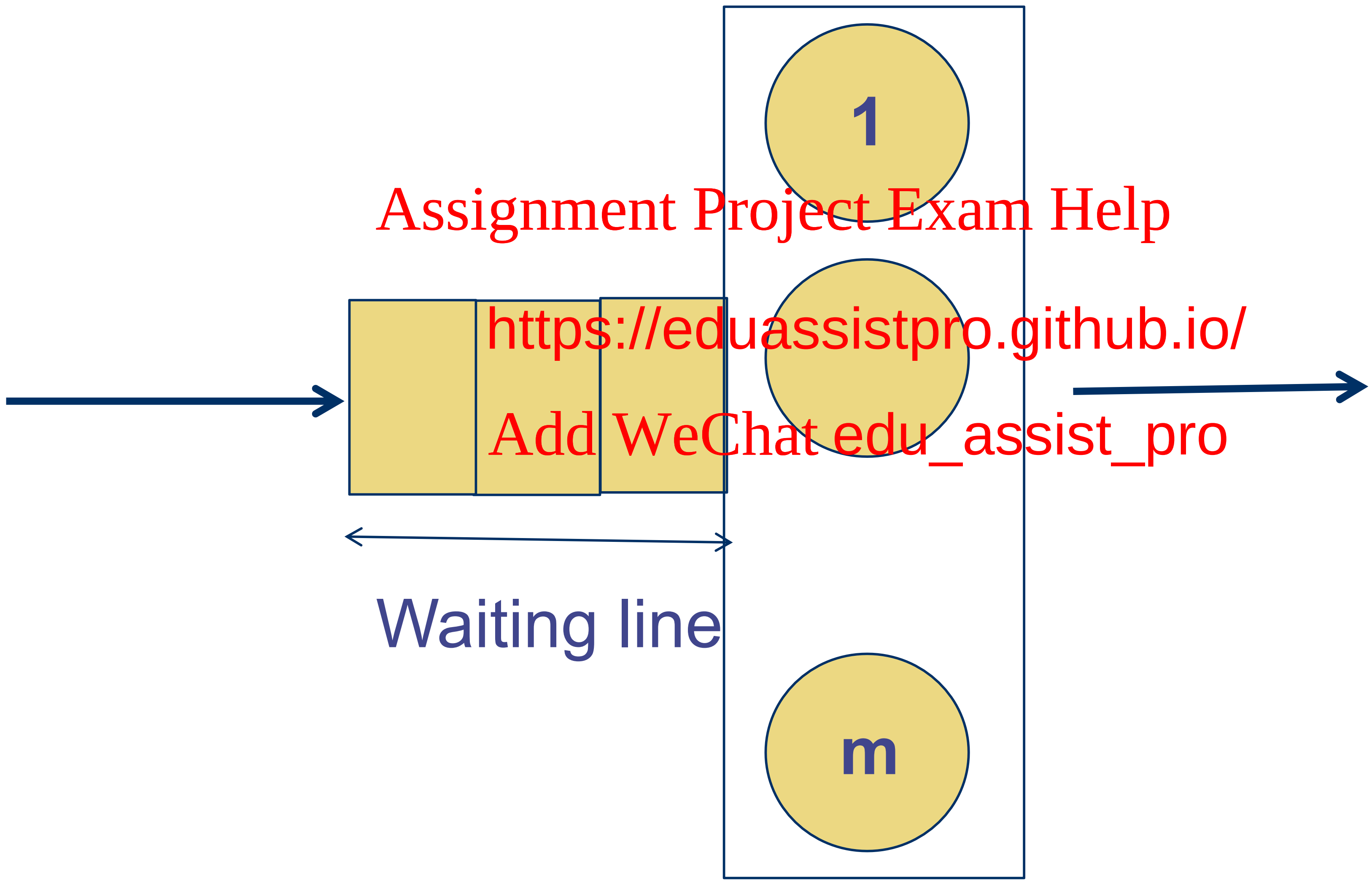
Pictorial representation of single server queues





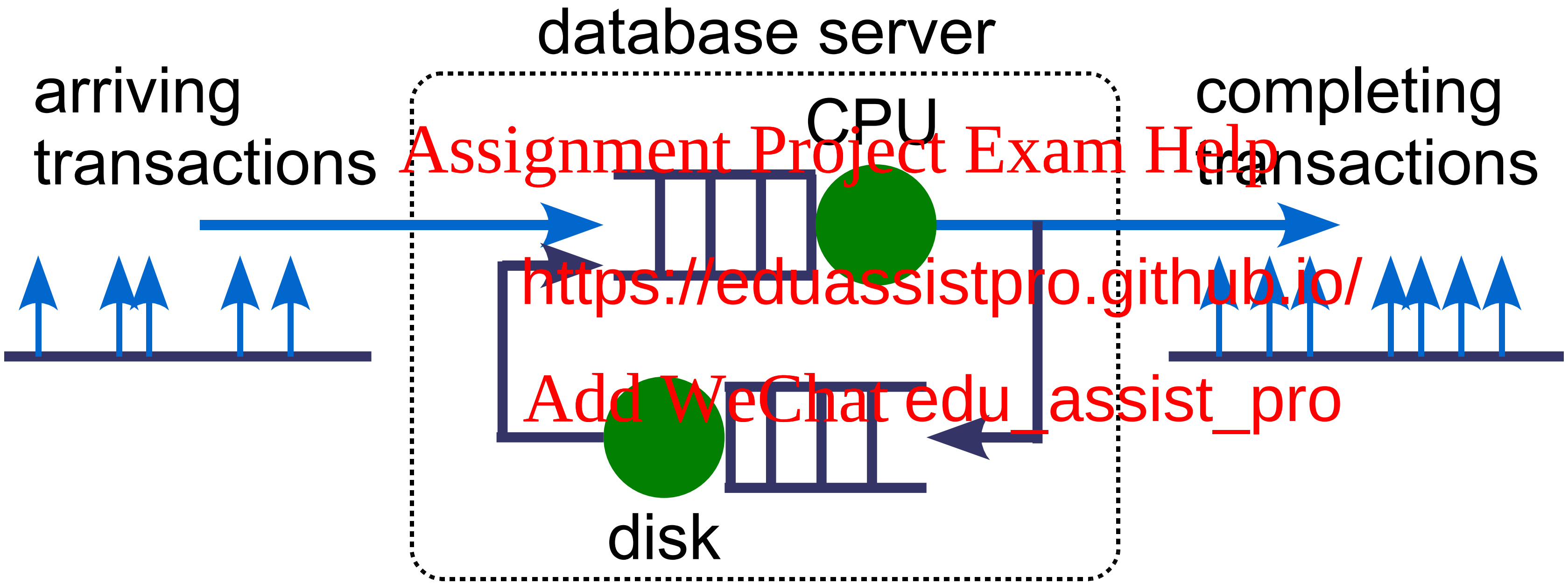
Pictorial representation of queues

Systems with *m* servers



A simple database server

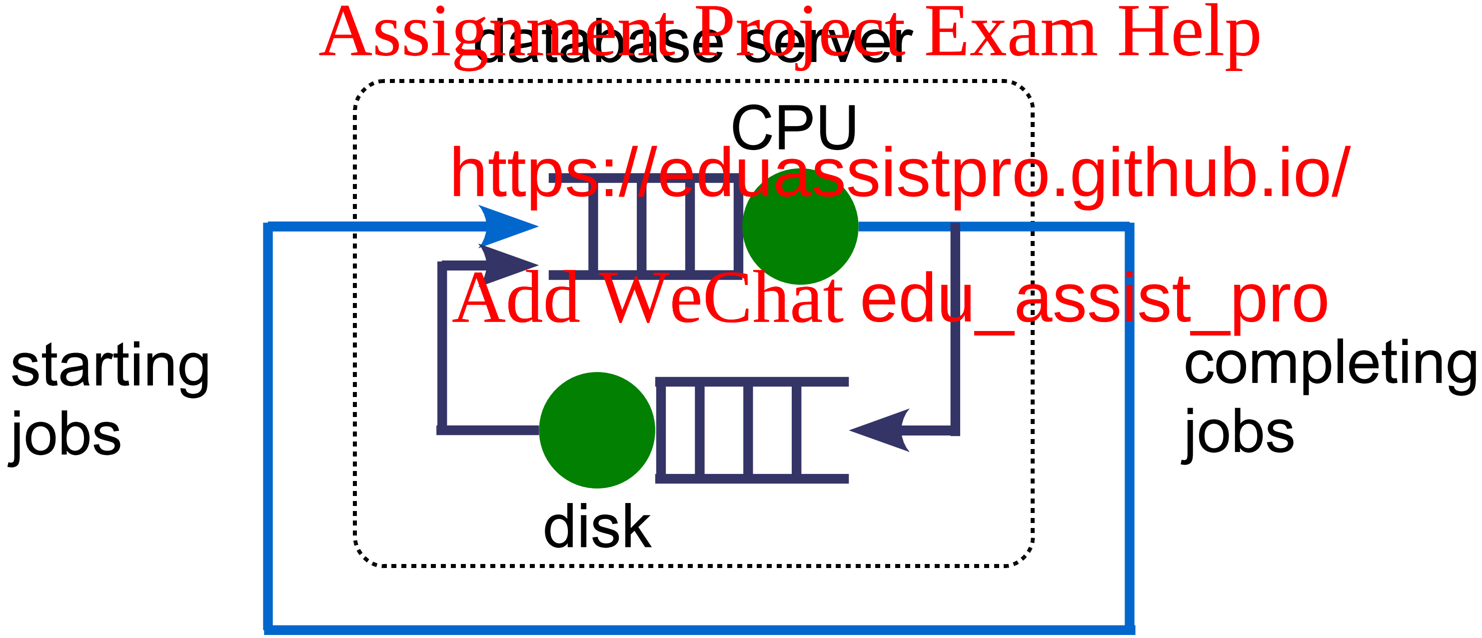
The server has a CPU and a disk.



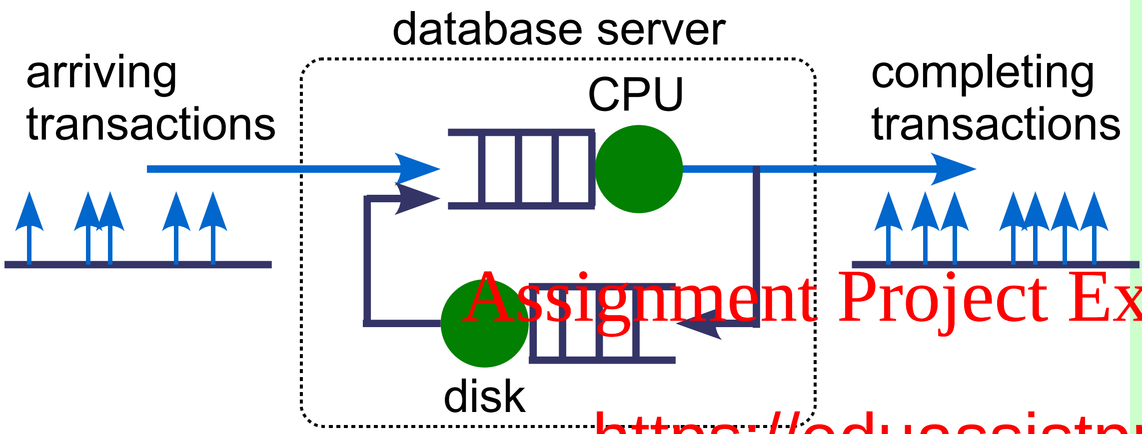
A transaction may visit the CPU and disk multiple times.

Database servers for batch jobs

- Example: Batch processing system
  - E.g. For summarization data from databases
  - No on-line transactions

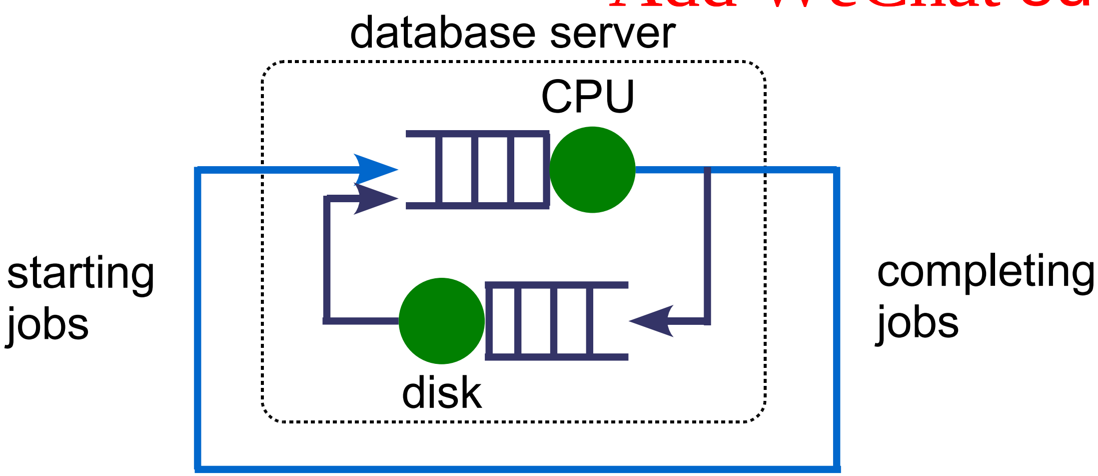


Open vs. closed queueing networks (1)



Open queueing network

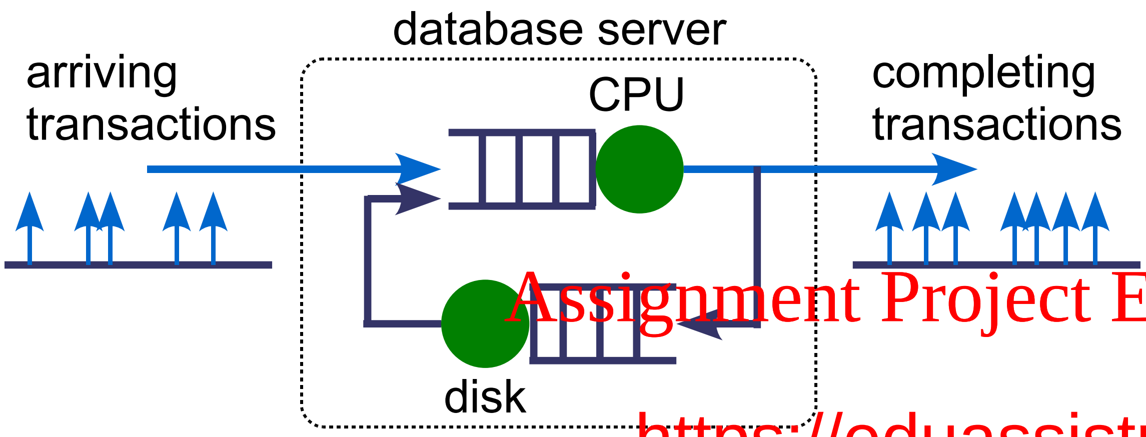
- External arrivals
- Workload intensity specified by arrival te



Closed queueing network

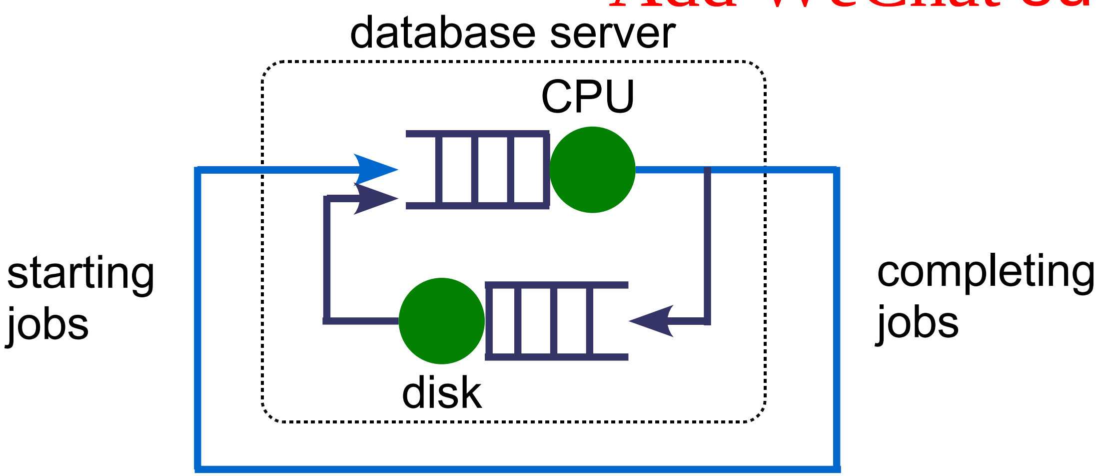
- No external arrivals
- Workload intensity specified by customer population

Open vs. closed queueing networks (2)



Open queueing network

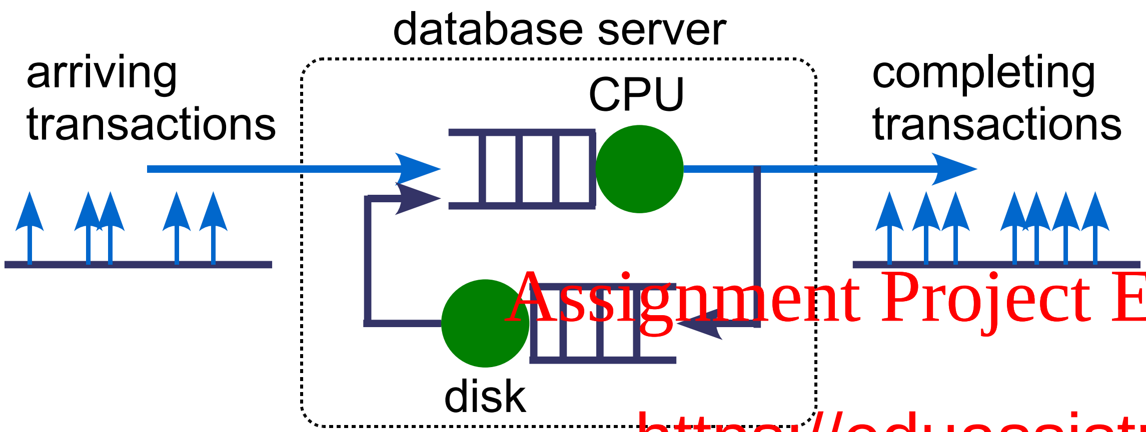
- Possibly unbouned #customers
- For stable equilibrium
  Throughput = arrival rate



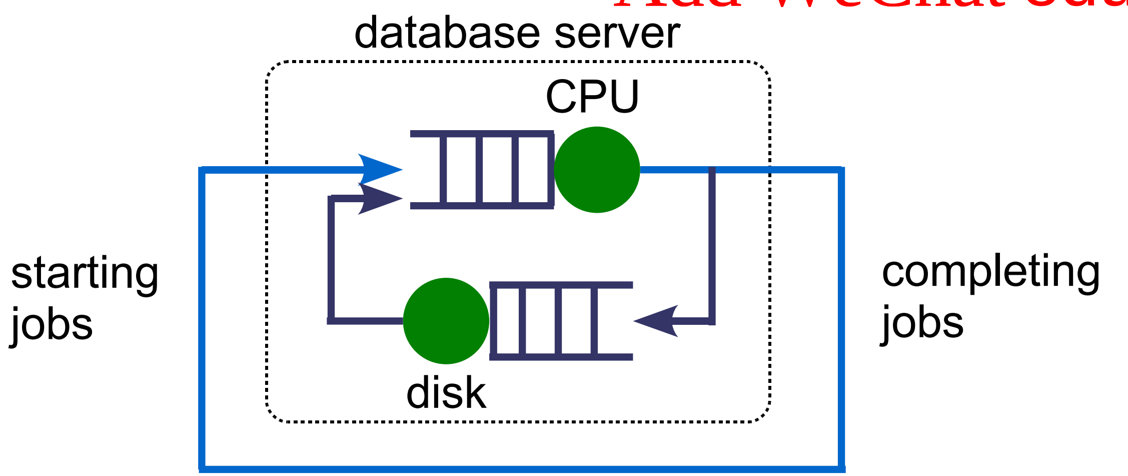
Closed queueing network

- Known #customers
- Throughput depends on #customers etc.

Open vs. closed queueing networks - Terminology



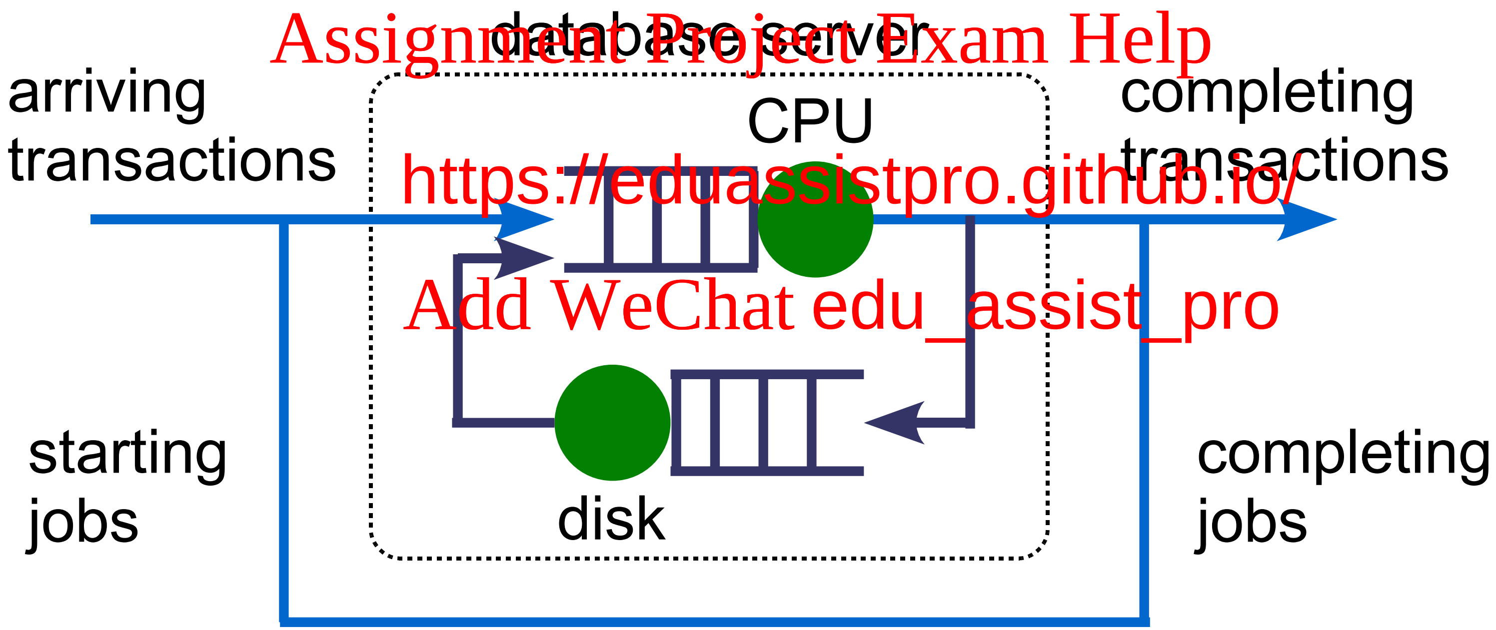
Work in an open queueing network is called transaction



Work in a closed queueing network is called jobs

DB server - mixed model

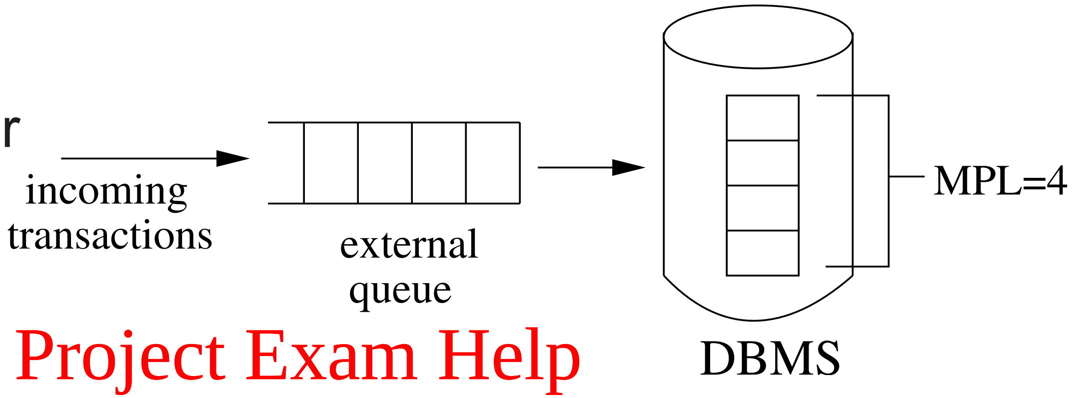
- The server has both
  - External transactions
  - Batch jobs



Different techniques are needed to analyse open and closed queueing networks

DB server – Multi-programming level

- Some database server management systems (DBMS) set an upper limit on the number of active transactions within the system
- This upper limit is called multi-programming level (M



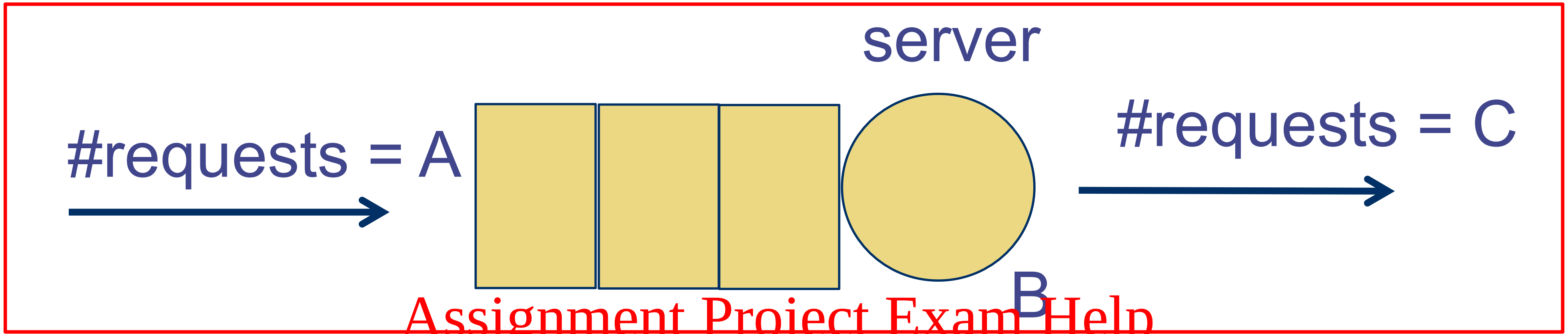
view of the mechanism used in ... limited number of trans-actions (... ed into the DBMS simul-taneousl... ansactions are held back in an external queue. Response time is the time from when a transaction arrives until it completes, including time spent queueing externally to the DBMS.

- A help page from SAP explaining MPL
- http://dcx.sap.com/1200/en/dbadmin_en12/running-s-3713576.html
- Picture from Schroder et al. "How to determine a good multi-programming level for external scheduling"

Operational analysis (OA)

- "Operational"
  - Collect performance data during day-to-day operation
- Operation laws
- Applications:
  - Use the data for building queueing network models
  - Perform bottle
  - Perform modifi

Single-queue example (1)



In an observational y for time B
A requests arrived,

A, B and C are basic measurements

Deductions: Arrival rate $\lambda = A/T$
Output rate $X = C/T$
Utilisation $U = B/T$
Mean service time per completed request $= B/C$

Motivating example

- Given
  - Observation period = 1 minute
  - CPU
    - Busy for 36s.
    - 1790 requests arrived
    - 1800 request
  - Find
    - Mean service time per complet
    - Utilisation =
    - Arrival rate =
    - Output rate =

Utilisation law

- The operational quantities are inter-related
- Consider
  - Utilisation U = B / T
  - Mean service time per completion S = B / C
  - Output rate X = C / T

- Utilisation law – d X?

- Utilisation law is an example of operational law.

Application of OA

- Don't have to measure every operational quantities
  - Measure B to deduce U - don't have to measure U
- Consistency checks
  - If $U \neq S X$, something is wrong
- Operational laws can be used for performance analysis
  - Bottleneck anal
  - Mean value analysis (Later in th

Equilibrium assumption

- OA makes the assumption that
  - C = A
  - Or at least C ≈ A
- This means that
  - The devices and system are in equilibrium
    - Arrival rate put rate of requests for that device = Thr
    - The above statement also app stem, i.e. replace the word "device" by "system"

OA for Queueing Networks (QNs)

The computer system has K devices, labelled as 1,…,K.

The convention is to add an additional device 0 to represent the outside world.

OA for QNs (cont’d)

- We measure the basic operational quantities for each device (or other equivalent quantities) over a time of T
  - A(j) = Number of request arriving at device j
  - B(j) = Busy time for device j
  - C(j) = Number of completed requests for device j

- In addition, we h

  - A(0) = Number of
  - C(0) = Number of completions fo

- Question: What is the relationship between A(0) and C(0) for a closed QNs?

Visit ratios

- A job arriving at the system may require multiple visits to a device in the system
  - Example: If every job (or transaction) arriving at the system will require 3 visits to the disk (= device j), what is the ratio of C(j) to C(0)?

    - We expect

  - V(j) = Visit ratio of device j

    = Number of times a job (transaction) visits device j

    - We have V(j) = C(j) / C(0)

Forced Flow Law

Since $$V(j) = \frac{C(j)}{C(0)}$$

The forced flow law is

$$V(j) = \frac{X(j)}{X(0)}$$

Service time versus service demand

- Ex: A job requires two disk accesses to be completed. One disk access takes 20ms and the other takes 30ms.

- Service time = the amount of processing time required *per visit* to the device

  - The quantities “2 individual service times.

- D(j) = Service demand of a job is the total service time required by that job
  - The service demand for this job = 20ms + 30 ms = 50ms

Service demand

- Service demand can be expressed in two different ways
  - Ex: A job requires three disk accesses to be completed. One disk access takes 20ms and the others take 30ms and 28ms.
    - What is D(j)?

    - What are V(j)
      - Recall that

e j

  - Service demand D(j) = V(j) S(j)

Service demand law (1)

Given D(j) = V(j) S(j)

Since $$V(j) = \frac{X(j)}{X(0)}$$

t is X(j) S(j)?

Service demand law $$D(j) = \frac{U(j)}{X(0)}$$

Service demand law (2)

- Service demand law D(j) = U(j) / X(0)
  - You can determine service demand without knowing the visit ratio
  - Over measurement period T, if you find
    - B(j) = Busy time of device j
    - C(0) = Number of requests completed
  - You've enough information to find D(j)

- The importance of service de
  - You will see that service dema amental quantity you need to determine the performance of a queueing network
  - You will use service demand to determine system bottleneck in Lecture 2A

Server example exercise

Measurement time = 1 hr		
	# I/O per second	Utilisation
Disk 1	32	0.30
	36	0.41
	50	0.54
C		0.35
Total # jobs=13680		

What is the service time of Disk 2?
What is the service demand of Disk 2?
What is its visit ratio?

Server example solution

Measurement time = 1 hr		
	# I/O per second	Utilisation
Disk 1	32	0.30
	36	0.41
	50	0.54
C		0.35
Total # jobs=13680		

Service time
System throughput
Service demand
Visit ratio

Little's law (1)

- Due to J.C. Little in 1961
  - A few different forms
    - The original form is based on stochastic models
  - An important result which is non-trivial
    - All the other operational laws are easy to derive, but Little's Law's derivation is more elaborate.

- Consider a single-
  - Navg = Average number of reque vice
    - When we count the number of requests in a device, we include the one being served and those in the queue waiting for service

Little's Law (2)

- X = Throughput of the device
- Ravg = Average response time of the requests
- Navg = Average number of requests in the device
- Little's Law (for OA) says that

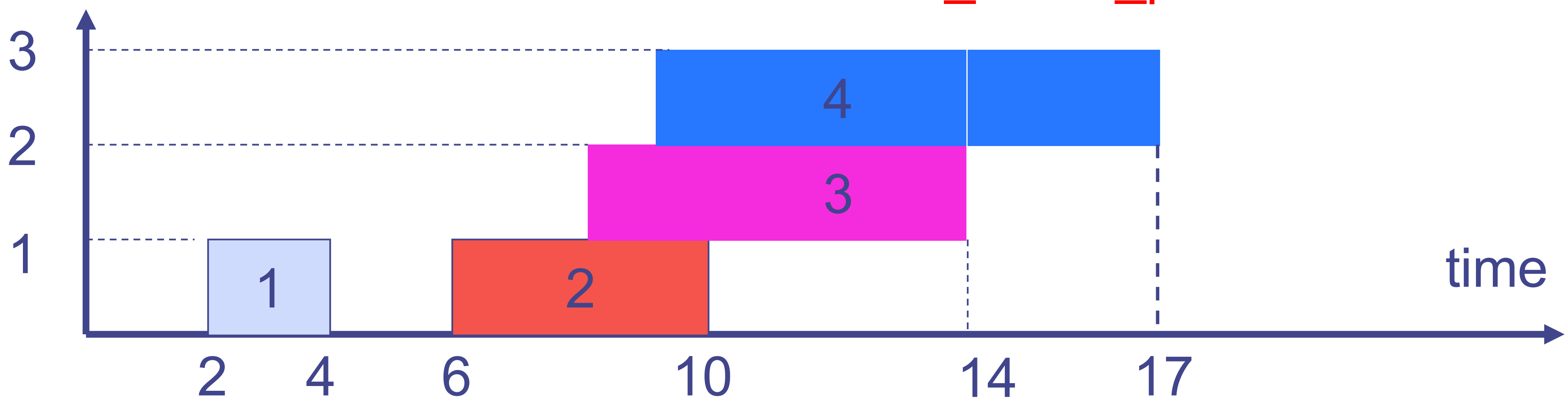
Navg

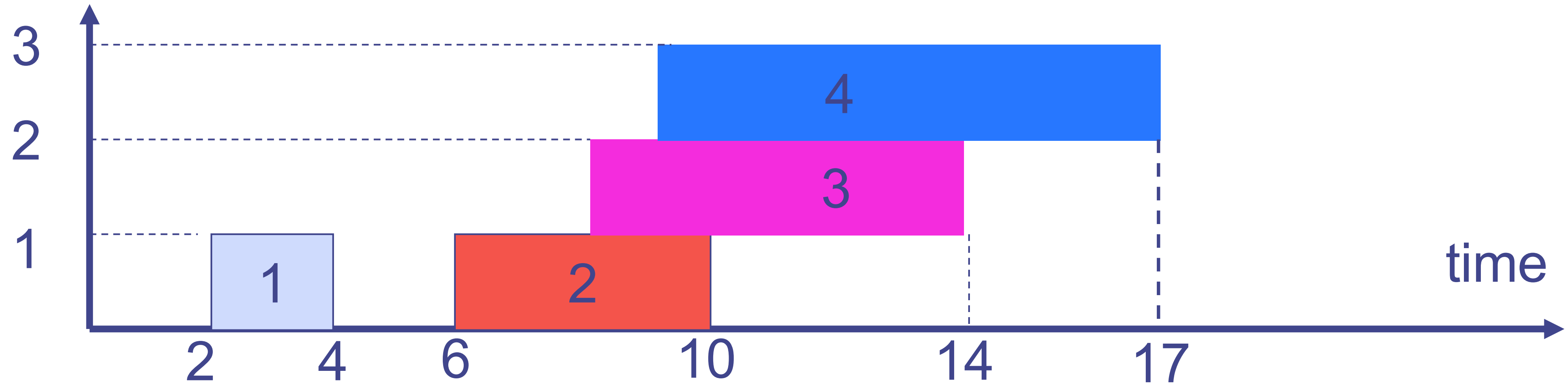
We will argue the validity of Little ing a simple example.

Consider the single sever queue example from Week 1

Request index	Arrival time	Service time	Departure time
1	2	2	4
2	6	4	10
3	8	4	14
4	9	3	17

Let us use blocks of the span of the requests, i.e. width of each block time of the request

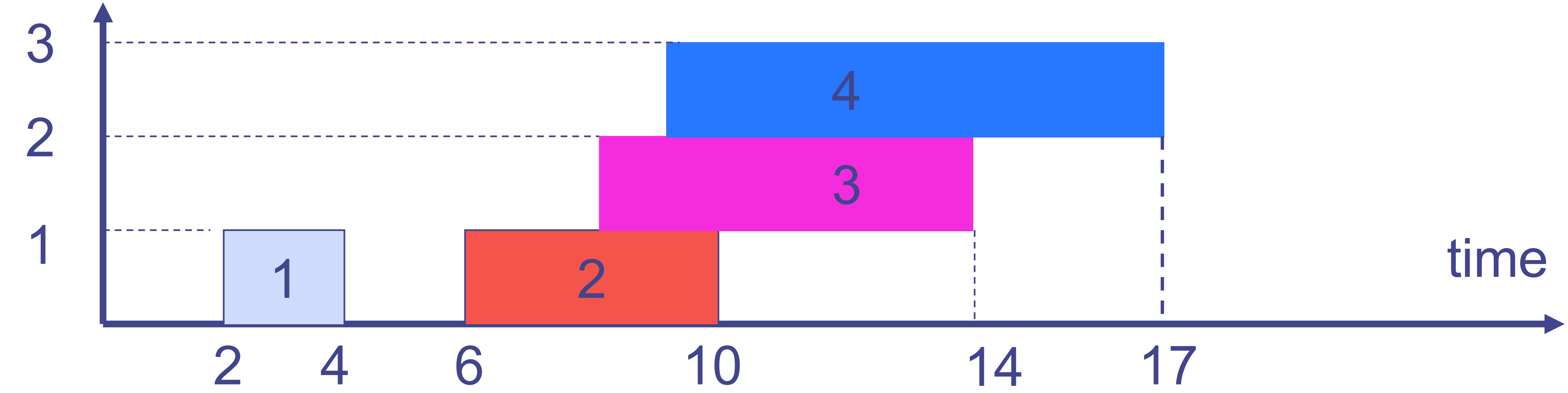




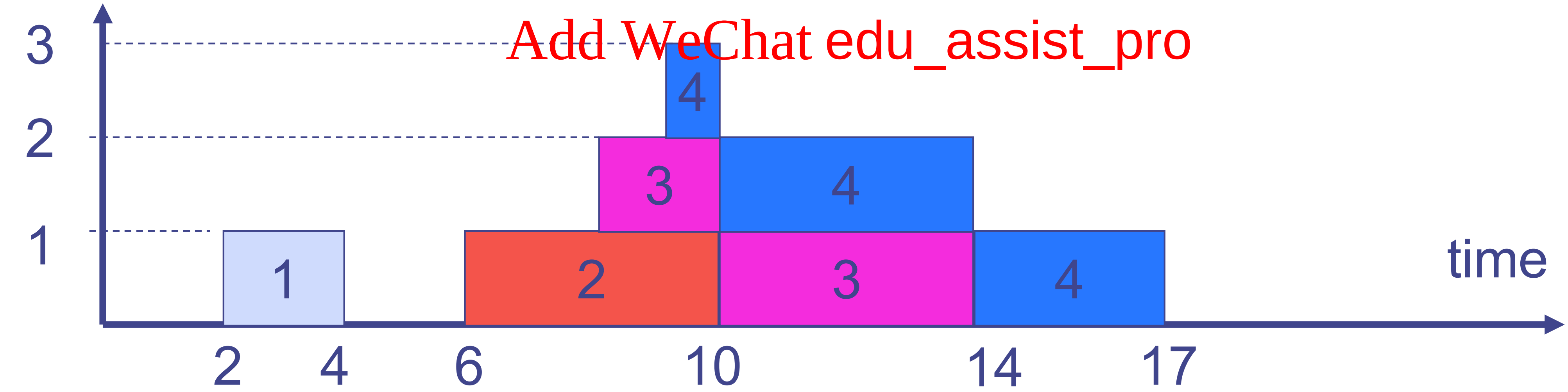
Assuming that in the measurement time interval [0,20] these 4 requests arrive and depart from this device, i.e. the device is in equilibrium.

Total area of the blocks
= Response time of request 1 + Response time of request 2 + Response time of request 3 + Response time of request 4
= Average response time over the measurement interval * Number of requests completed over the measurement interval

This is one interpretation. Let us look at another.

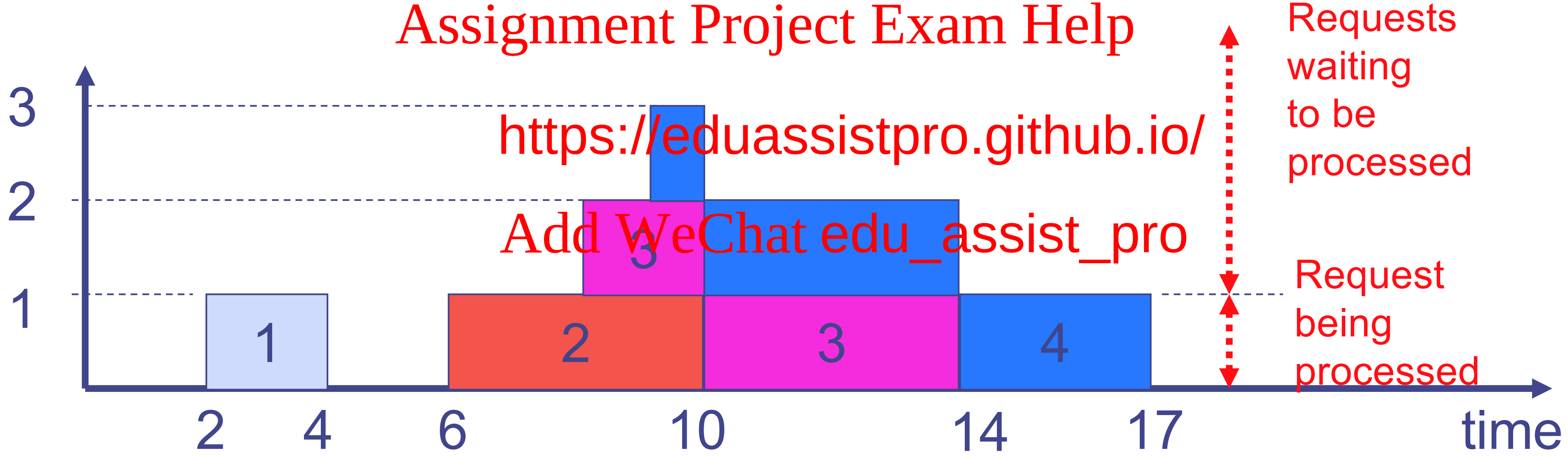


Let us assume these blocks are "plasticine" and let them fall to the ground. Like t



There is an interpretation of the height of the graph.

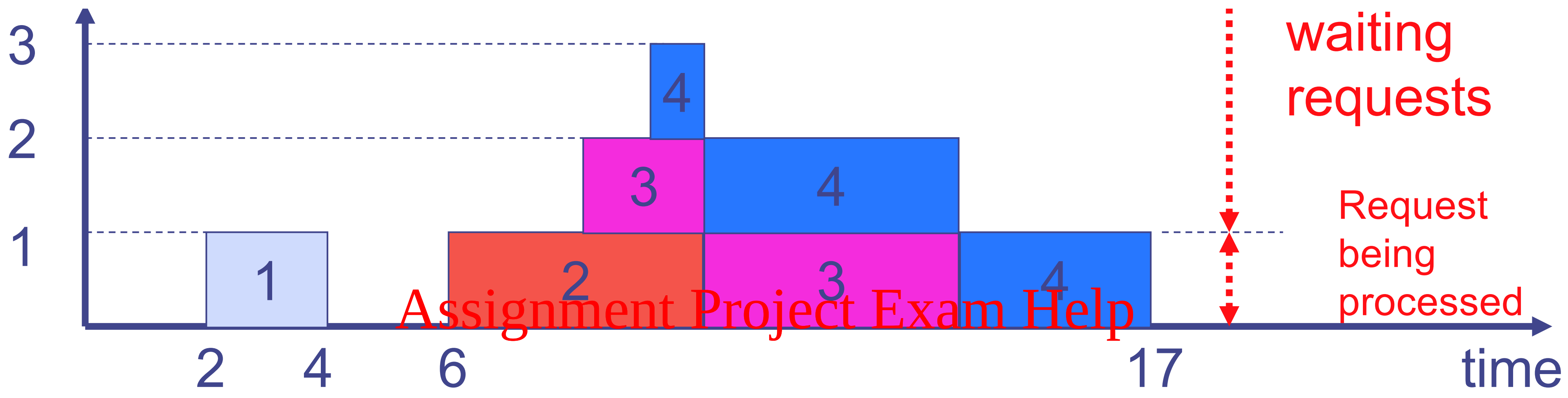
Request index	Arrival time	Service time
1	2	2
2	6	4
3	8	4
4	9	3



Interpretation: Height of the graph = #requests in the device

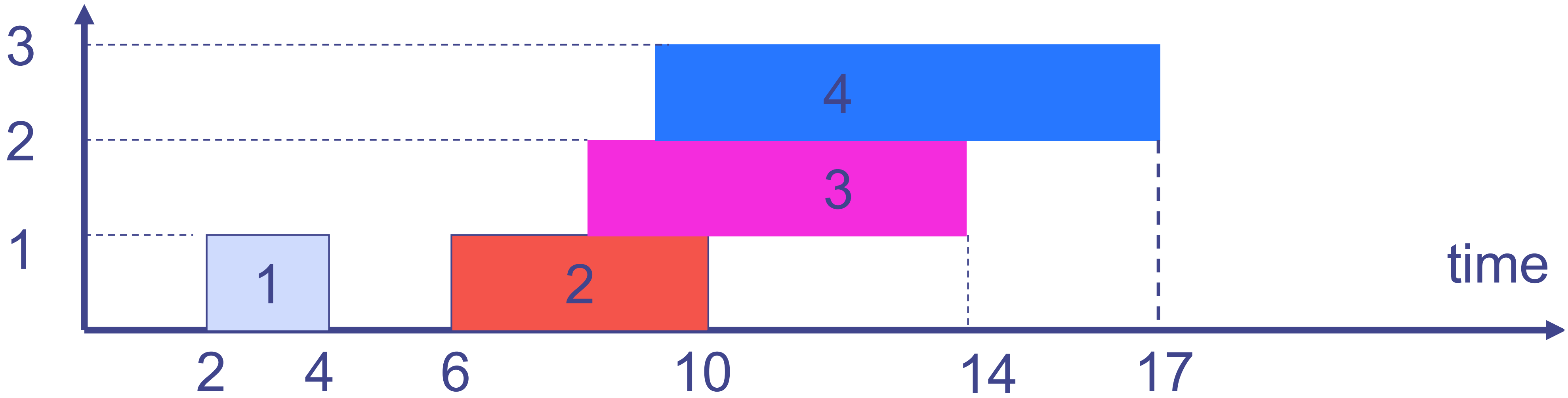
E.g. Number of requests in [9,10] = 3

E.g. Number of requests in [11,12] = 2 etc.

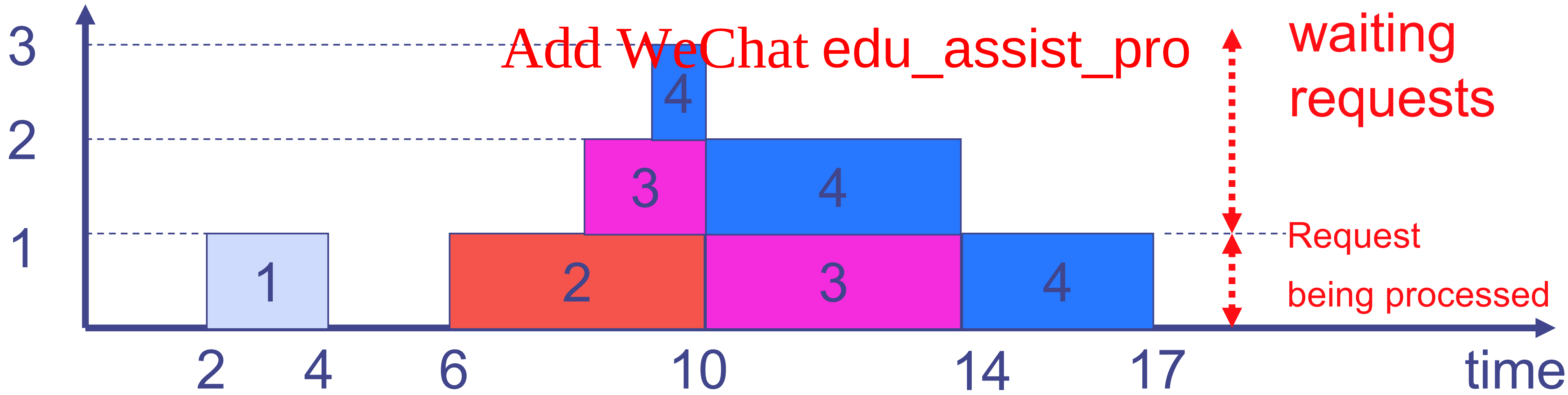


Again, consider the measurement interval of [0,20].

Area under the graph in [0,20]
= Height of the graph in [0,1] + Height of the graph in [1,2] + …
Height of the graph in [19,20]
= #reqs in [0,1] + #reqs in [1,2] + … + #reqs in [19,20]
= Average number of requests in [0,20] in the device * 20



Area = Average response time over [0,T] *
Number in [0,T]



Area = Average number of requests in [0,T] * T

Deriving Little's Law

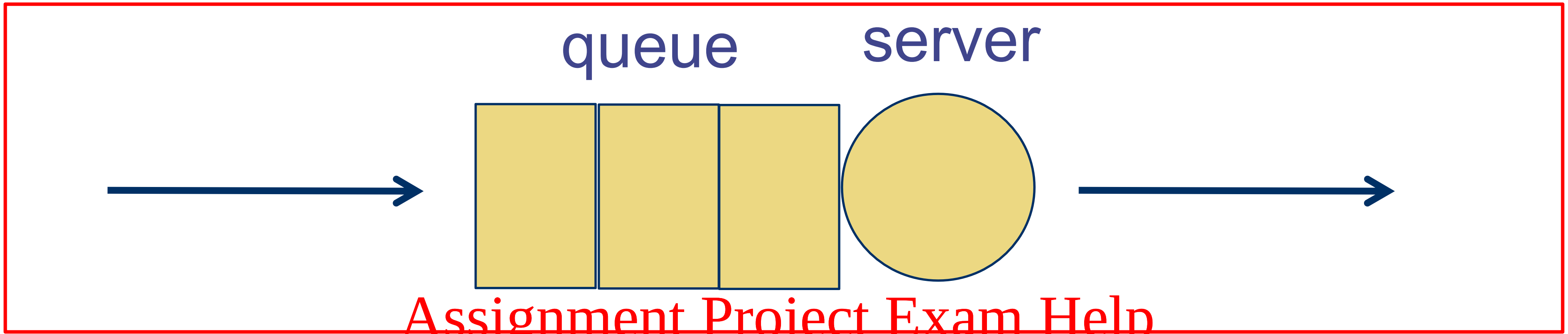
Area = Average response time of all jobs *
Number of requests completed in [0,T] (Interpretation #1)
= Average #requests in [0,T] * T (Interpretation #2)

Since Number of req T] / T
= Device throughput in [0,T]

We have Little's Law.

Average number of requests in [0,T]
= Average response time of all reqs * Device throughput in [0,T]

Using Little's Law (1)



- A device consists ue
- The device comp uests per second
- On average, there are 3.2 requ device
- What is the response time of the device?

Intuition of Little's Law

- Little's Law
  - Mean #requests = Mean response time * Mean throughput
- If #requests in the device ⬆️ , then response time ⬆️
  - And vice versa

Applicability of Little's Law

- Little's Law can be applied at many different levels
- Little's law can be applied to a device
  - Navg(j) = Ravg(j) * X(j)

- A system with K devices
  - Navg(j) = #requ

  - Average numb m Navg = Navg(1) + …. + Navg(K)

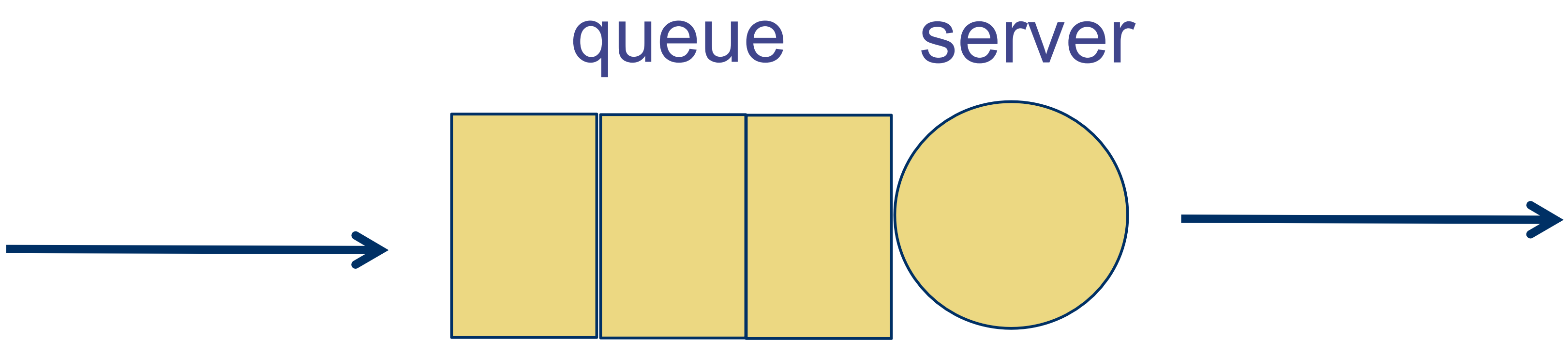
  - Average response time of the s g
- We can also apply it to an entire system
  - Navg = Ravg * X(0)

Assignment Project Exam Help

https://eduassistpro.github.io/

Add WeChat edu_assist_pro

Using Little's Law (2)

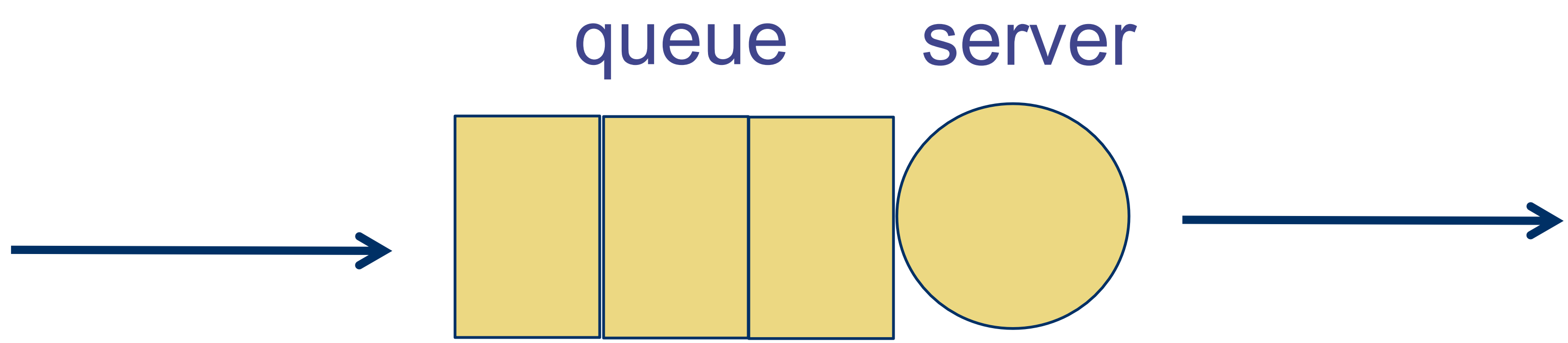


- The device compl ests per second
- On average, there

  - 3.2 requests in the device

  - 2.4 requests in the queue
  - 0.8 requests in the server
- What is the mean waiting time and mean service time?
- Hint: You need to draw "boxes" around certain parts of the device and interpret the meaning of response time for that box.

Using Little's Law (2)



- The device compl ests per second
- On average, there

  - 3.2 requests in the device

  - 2.4 requests in the queue
  - 0.8 requests in the server
- What is the mean waiting time and mean service time?

References

- Operational analysis
  - Lazowska et al, Quantitative System Performance, Prentice Hall, 1984. (Classic text on performance analysis. Now out of print but can be download from http://www.cs.washington.edu/homes/lazowska/qsp/
    - Chapters 3 and 5 (For Chapter 5, up to Section 5.3 only)
  - Alternative 1: You can read Menasce et al, “Performance by design”, Chapter 3. From beginning of Chapter 3 to Section 3.2.4.
  - Alternative 2: You c r 6. The treatment is more rigorous. You can g ioning ergodicity.

- Little’s Law (Optional)
  - I presented an intuitive “proof”. A more formal proof of this well known Law is in Bertsekas and Gallager, “Data Networks”, Section 3.2

- Revision questions based on this week’s lecture are available from course web site