Performance Modelling

Budapest University of Technology and Economics
Fault Tolerant Systems Research Group

A Well Known Example…

A Well Known Example…

Kiszolgálóhiba történt az alkalmazásban: „/hallgato”.

Futásidejű hiba

Leírás: Alkalmazáshiba történt a kiszolgálón. Az alkalmazás jelenlegi egyéni hibakezelési beállításai (biztonsági okok ... es adatainak távoli megjelenítését. A ... böngészőben azonban megtekinthetők.

Részletek: Ha távoli gépeken is meg szeretné jeleníteni a hibaüzenet részletes adatait, hozzon lé ... árában elhelyezett „web.config” ... címke „mode" attribútumát állítsa „Off" értékre.

```
<!-- Web.Config konfigurációs fájl -->

<configuration>
    <system.web>
        <customErrors mode="Off"/>
    </system.web>
</configuration>
```

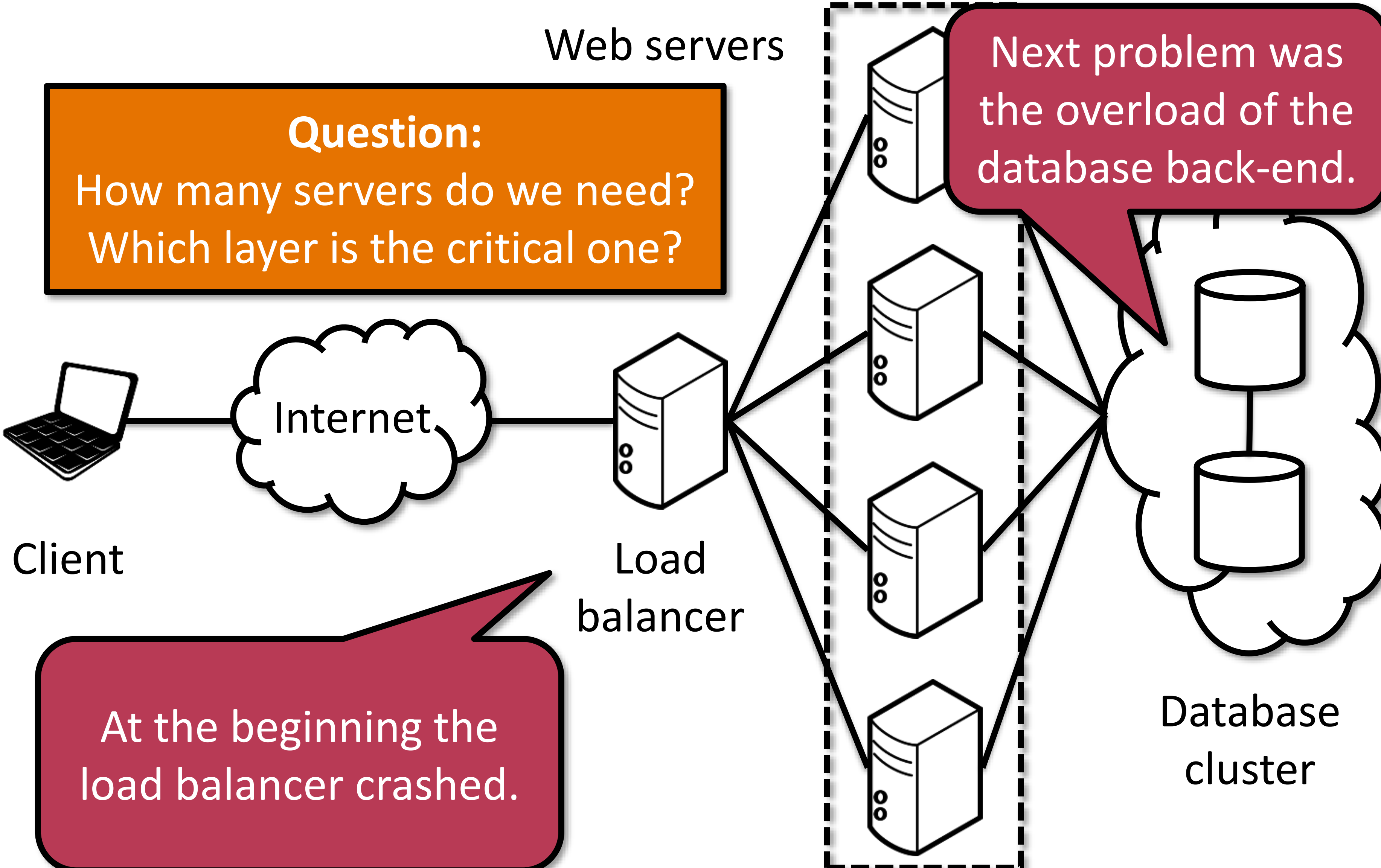
Megjegyzések:

```
<!-- Web.Confi

<configuration>
    <system.web>
        <customErr
    </system.web>
</configuration>
```

Motivation:
Prepare for performance issues in design time!

Scematic Architecture of Neptun System



Performance Modelling

- Review: non-functional requirements
  - Performance, throughput, dependability, etc.
  - How can such requirements be verified?
    (without building the actual system)
- **Performance modelling:**
  - Extending the discussed models with timing, resources, capacity constraints, …
  - Aim of this activity:
    - Evaluating performance of the system during design phase
    - Identifying bottlenecks
    - Scaling, capacity planning, dimensioning

Content

Foundations

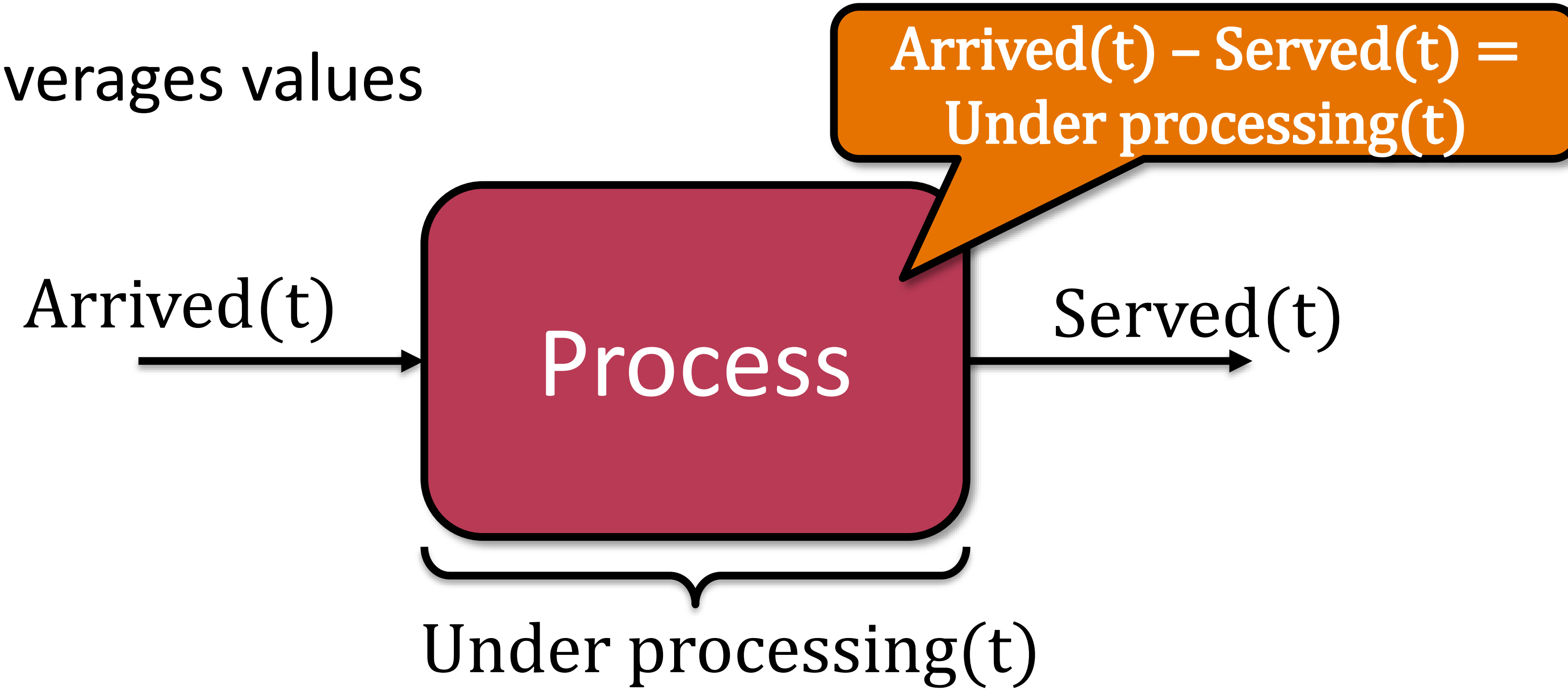
Load Diagram

Resource Modelling

FOUNDATIONS

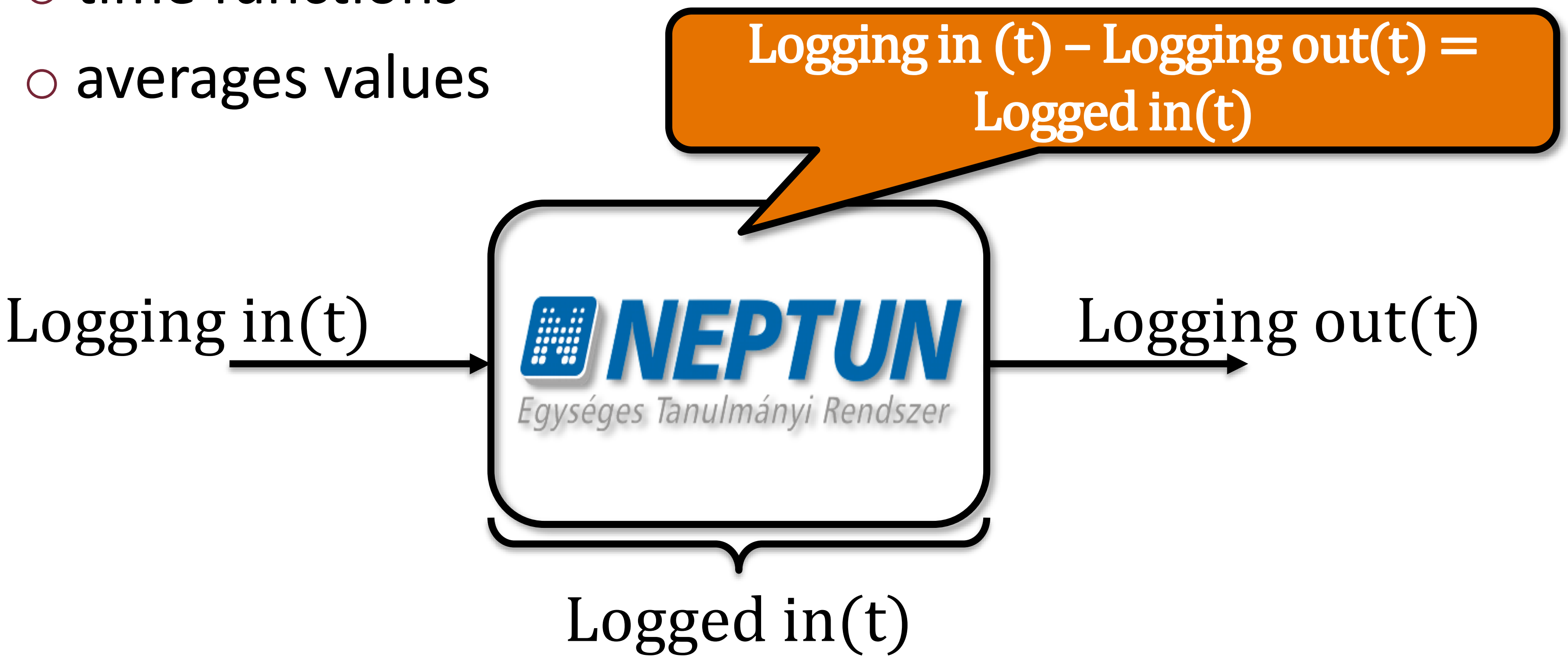
Basic Model

- Execution of a process in case of multiple requests
  - Subject of the analysis: time dependent behaviour
- Description of the behaviour:
  - time functions
  - averages values



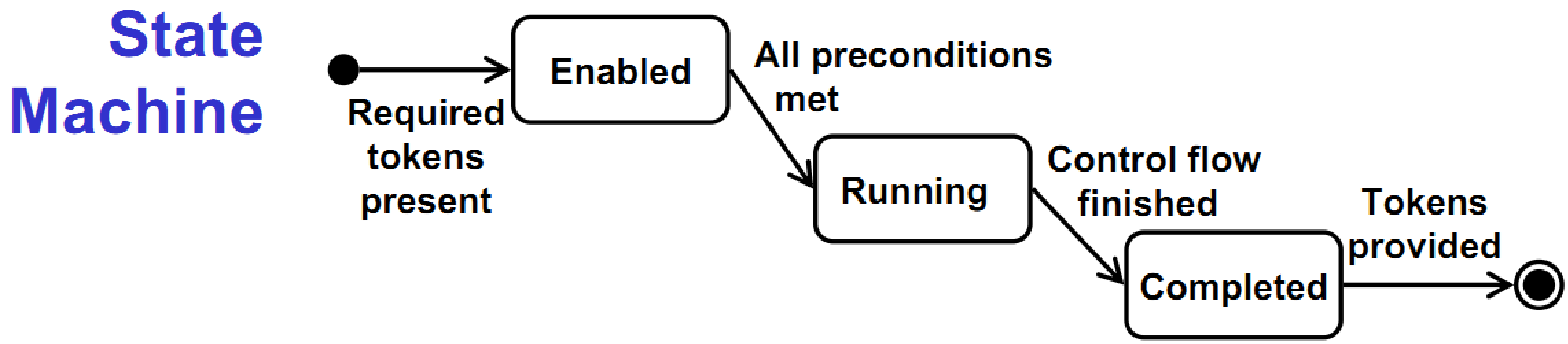
Basic Model

- Execution of a process in case of multiple requests
  - Subject of the analysis: time dependent behaviour
- Description of the behaviour:
  - time functions
  - averages values



Review: Execution States of Activities

- State of the process/activity execution:

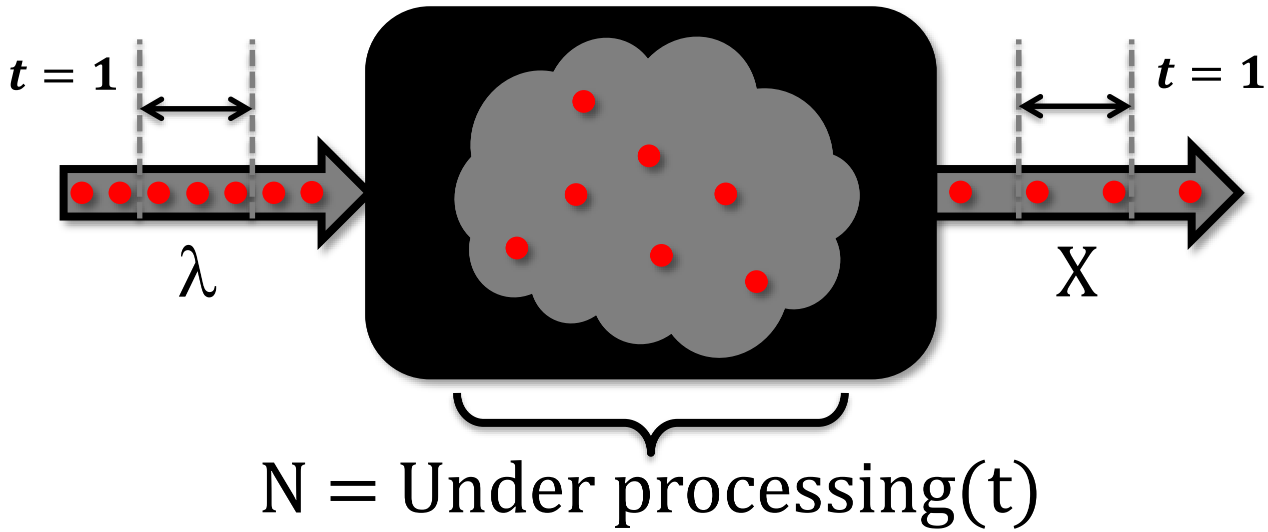


- Arrived(t): number of tokens in „*Enabled*” state
- Under processing(t): # of tokens in „*Running*” state
- Served(t): number of tokens in „*Completed*” state

Definition: Arrival Rate, Throughput

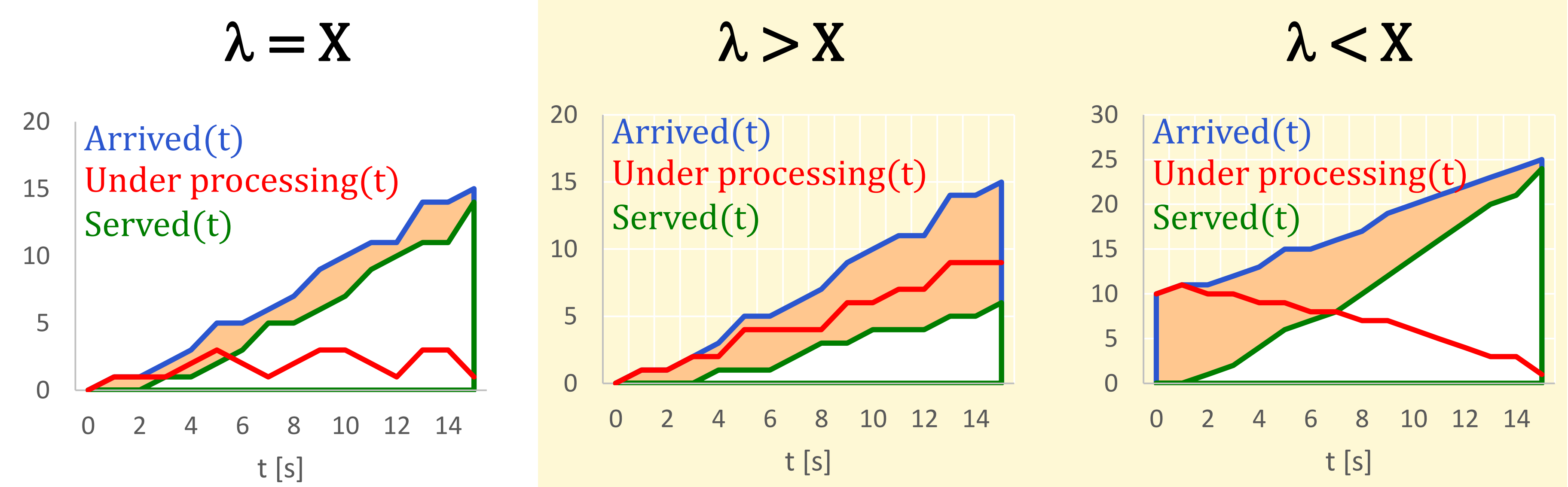
Arrival rate: number of **arriving** requests during a specific unit of time. $\lambda = \frac{Arrived(t)}{t}$ $[\lambda] = \frac{1}{s}$

Throughput: number of requests **processed** during a specific unit of time. $X = \frac{Served(t)}{t}$ $[X] = \frac{1}{s}$



Definition: Stable State

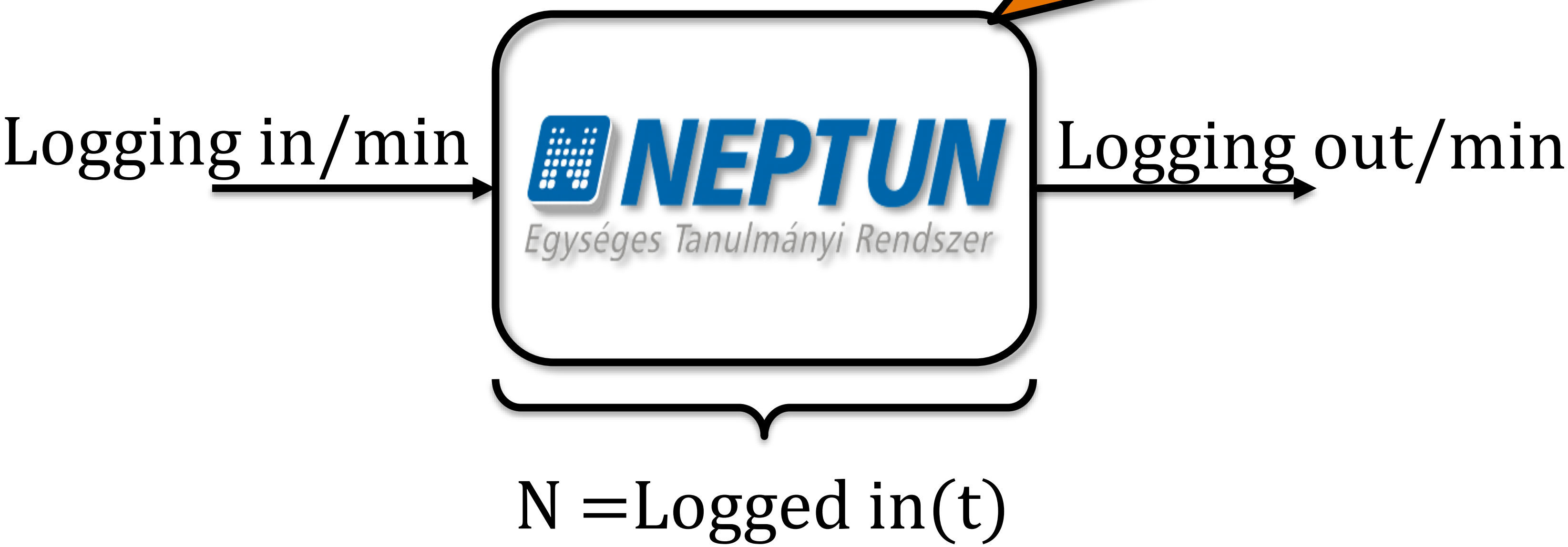
- **Stable state:** *Under processing(t)* is approximately constant
  - Average values can be applied in such state!
  - A system is in balance, if: $\lambda = X$



Definition: Stable State

- **Stable state:** *Under processing(t)* is approximately constant
  - Average values can be applied in such state!
  - A system is in balance, if:

In stable state:
Same number of logins and logouts per minute



Limited Capacity – DoS

- N is not infinite in real life
- So, what is then?

Denial of Service Attack

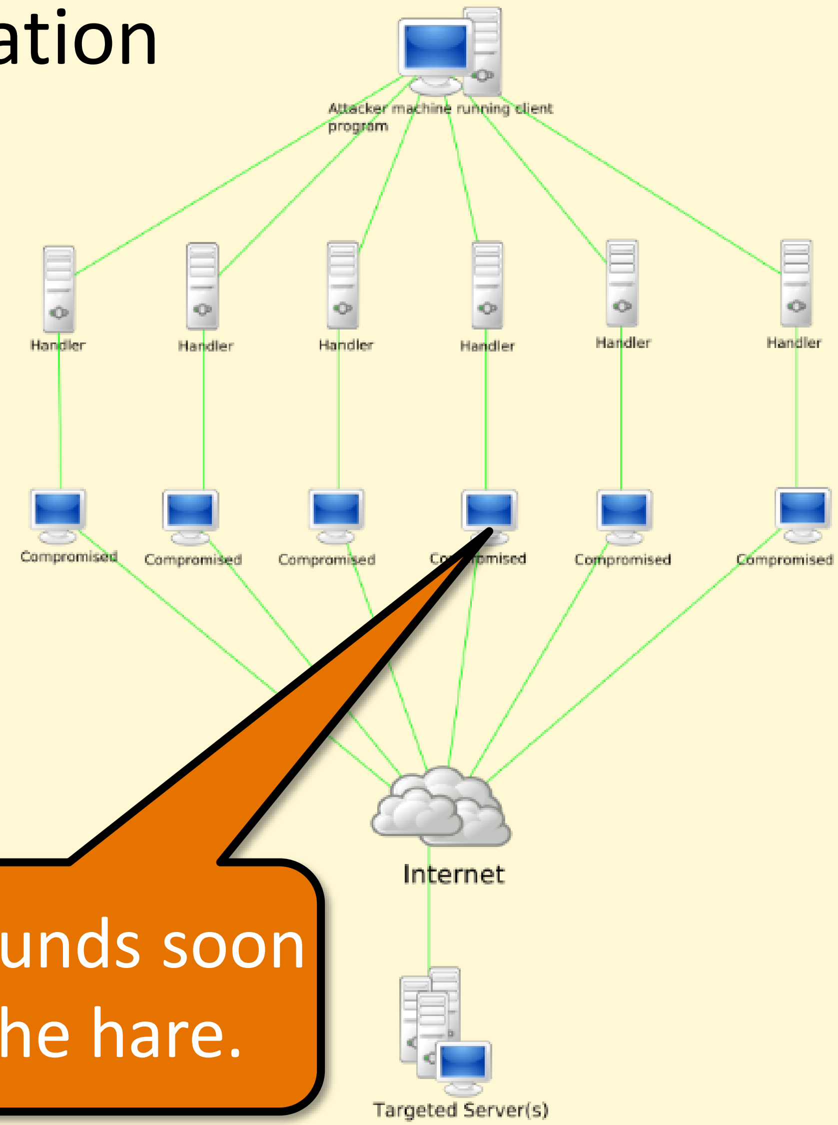
Thursday, August 6, 2009 | By Biz Stone (@biz) 08/06/2009 - 15:00

Tweet

On this otherwise happy Thursday morning, Twitter is the target of a denial of service attack. Attacks such as this are malicious efforts orchestrated to disrupt and make unavailable services such as online banks, credit card payment gateways, and in this case, Twitter for intended customers or users. We are defending against this attack now and will continue to update our status blog as we continue to defend and later investigate.

(Distributed) Denial of Service – (D)DoS

- Mass scaled request generation
  → overload of a system
- An overloaded system is fragile (can be cracked)
- Complete services can be knocked out
- Favourite method of the Anonymous group



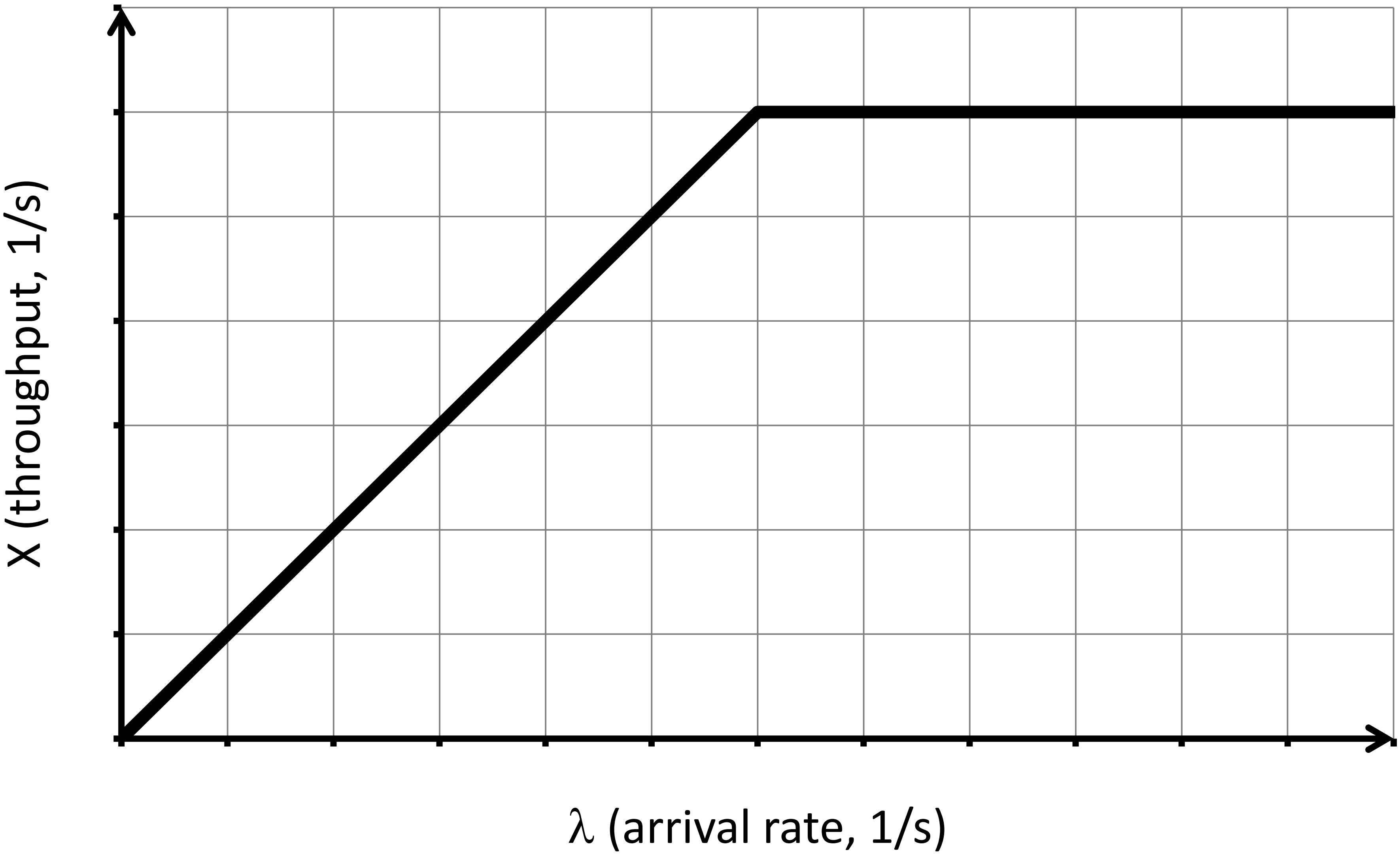
Many hounds soon catch the hare.

Foundations → **Load Diagram** → Resource Modelling

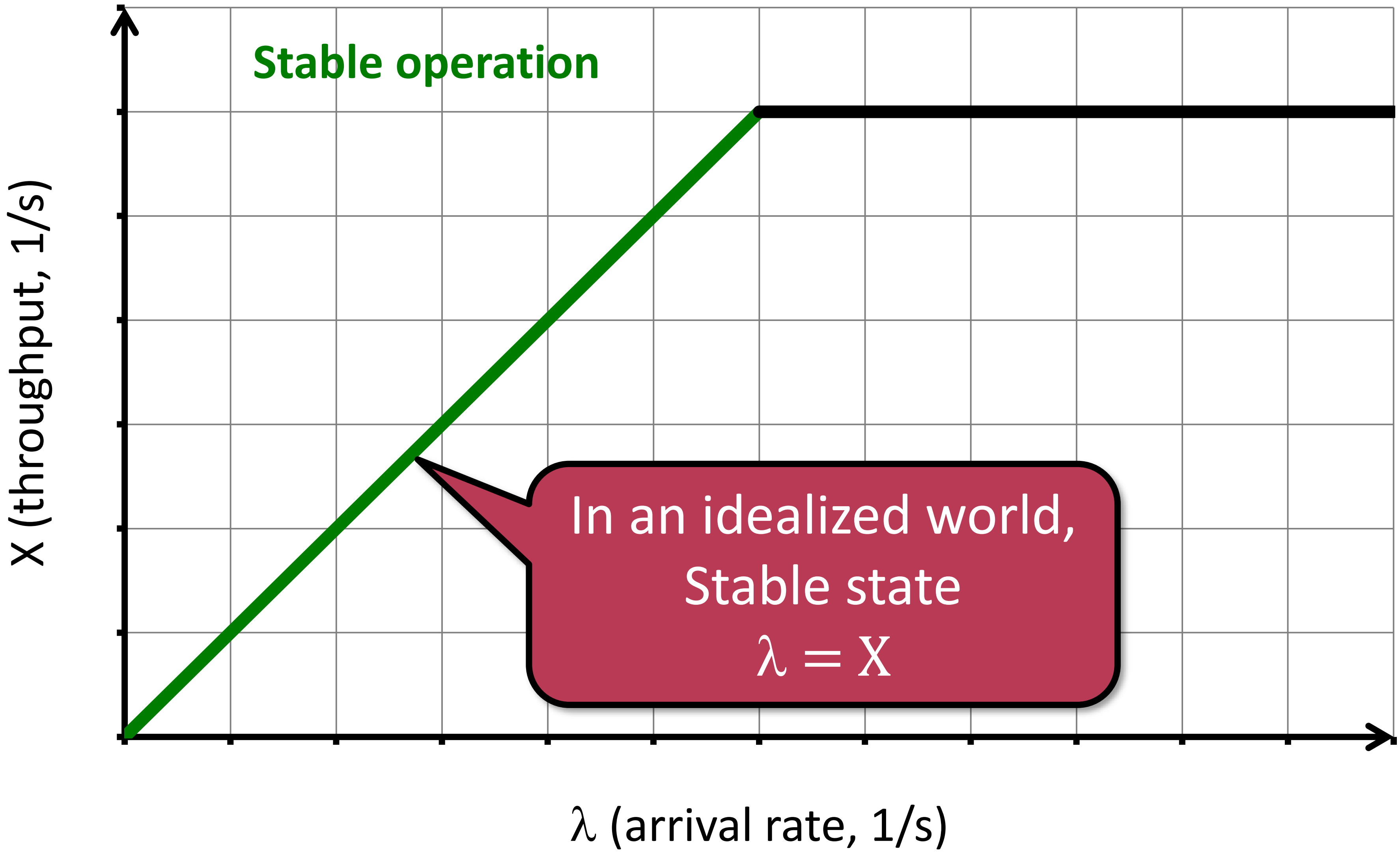
LOAD DIAGRAM

- Relation between arrival rate and throughput
- Maximum throughput

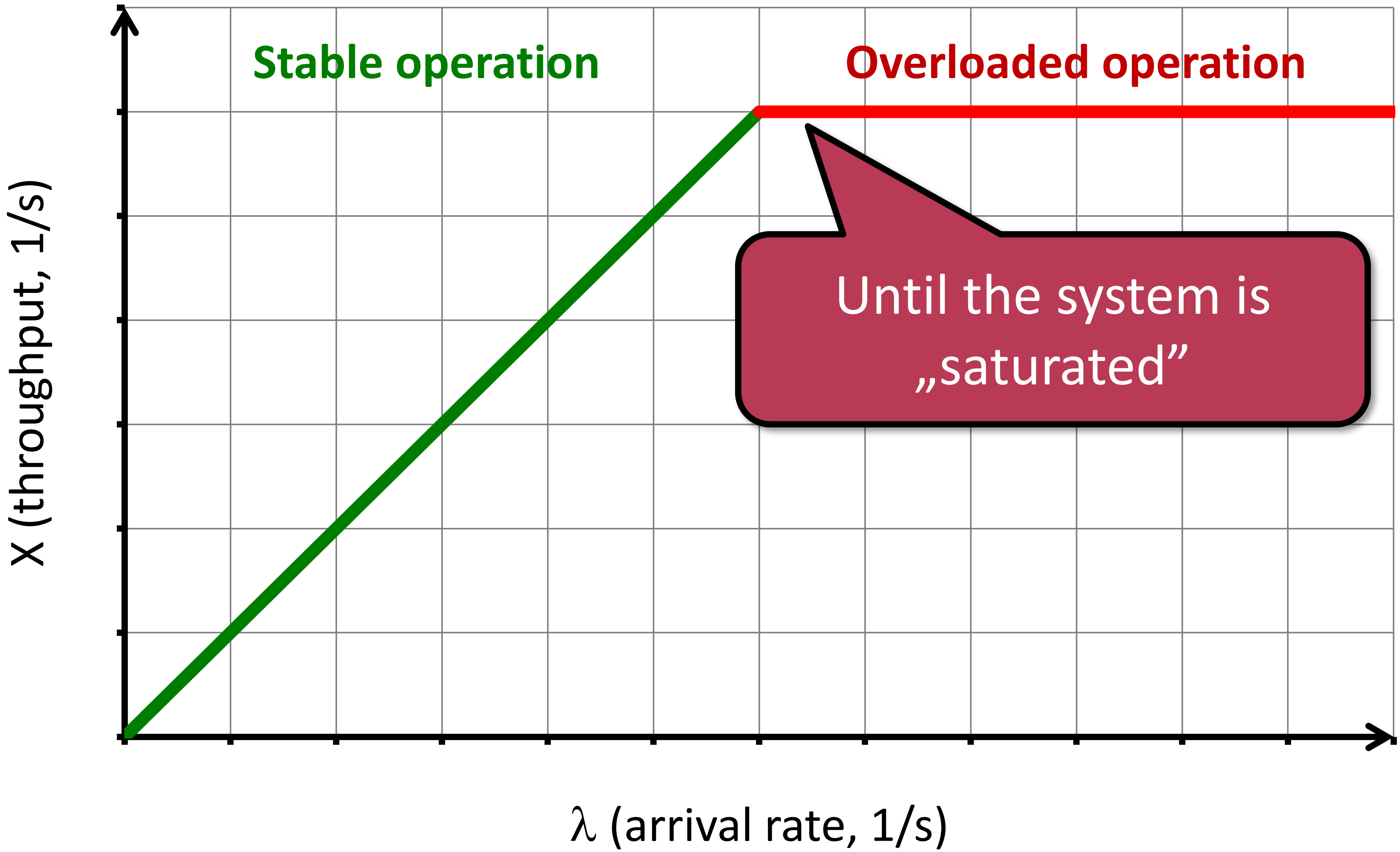
Load Diagram



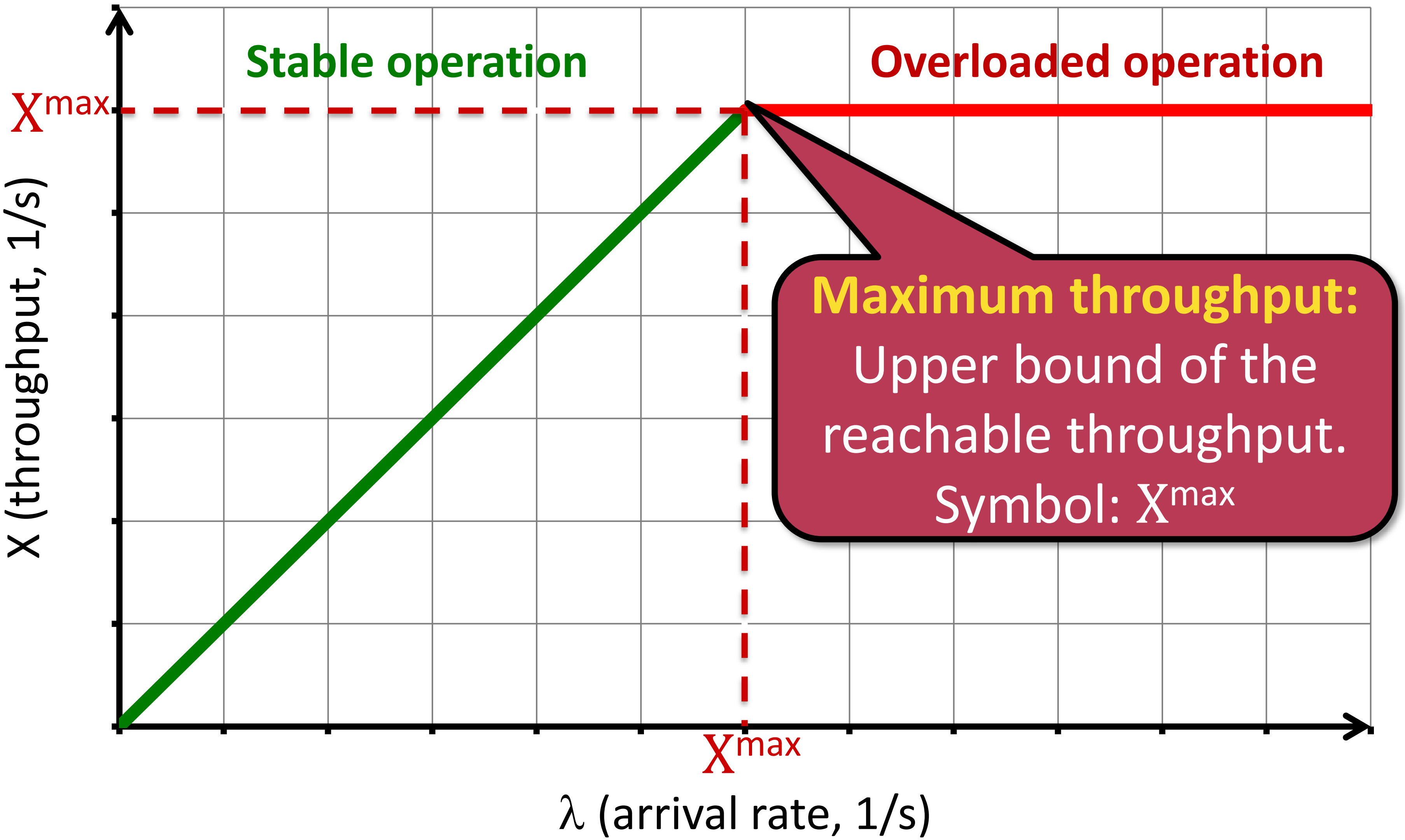
Load Diagram



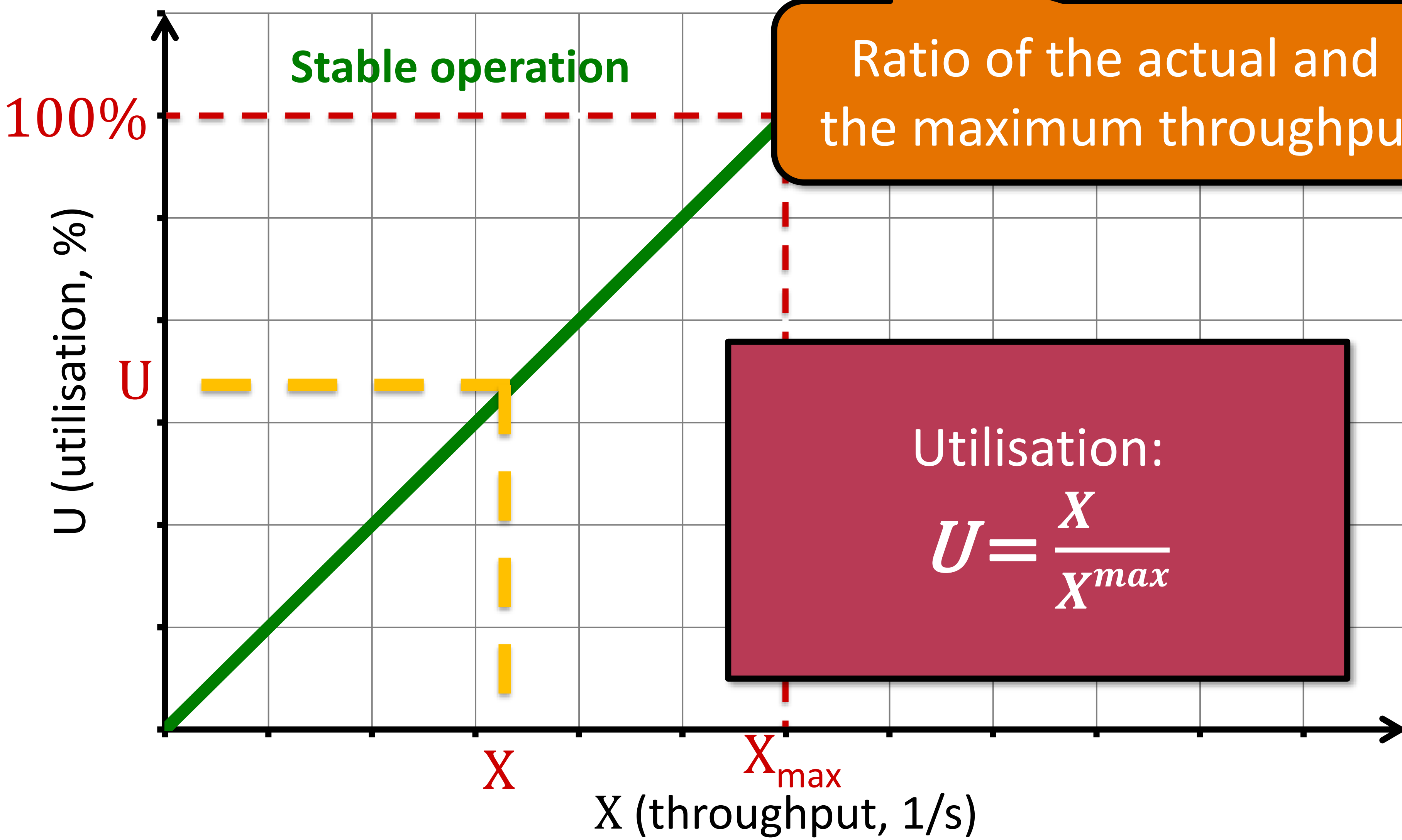
Load Diagram



Maximum Throughput



Utilisation

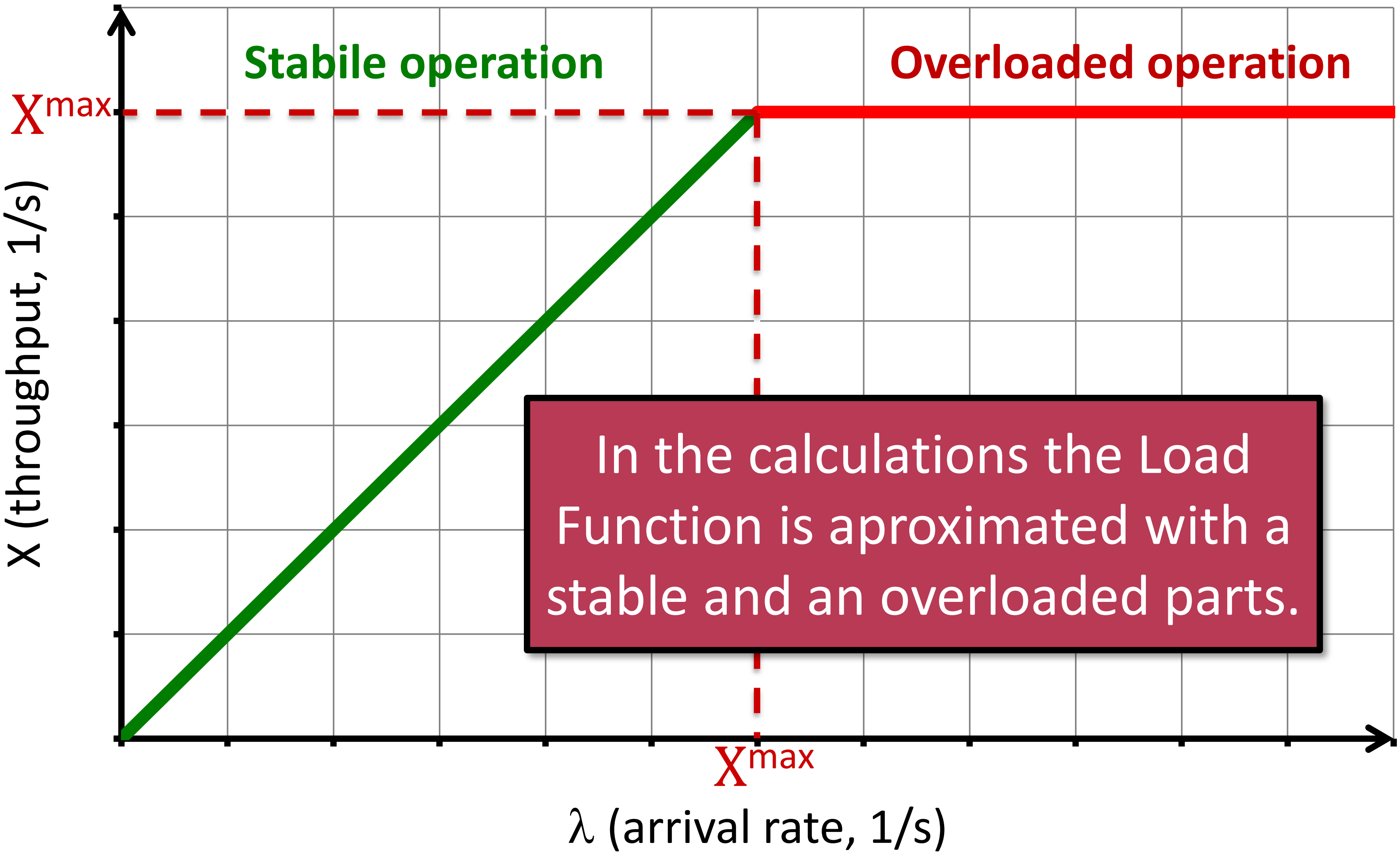


Ratio of the actual and the maximum throughput

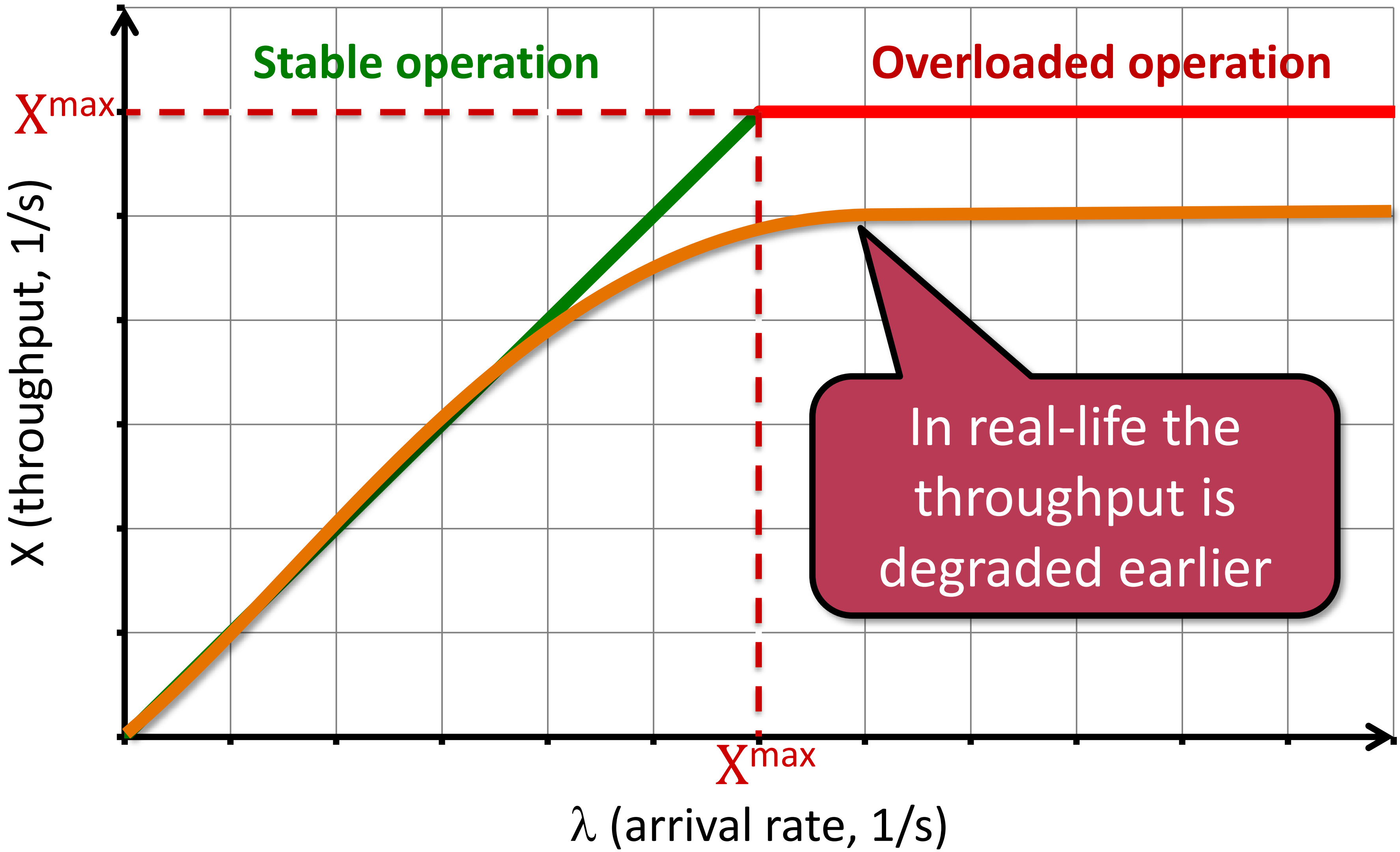
Utilisation:

$$U = \frac{X}{X^{max}}$$

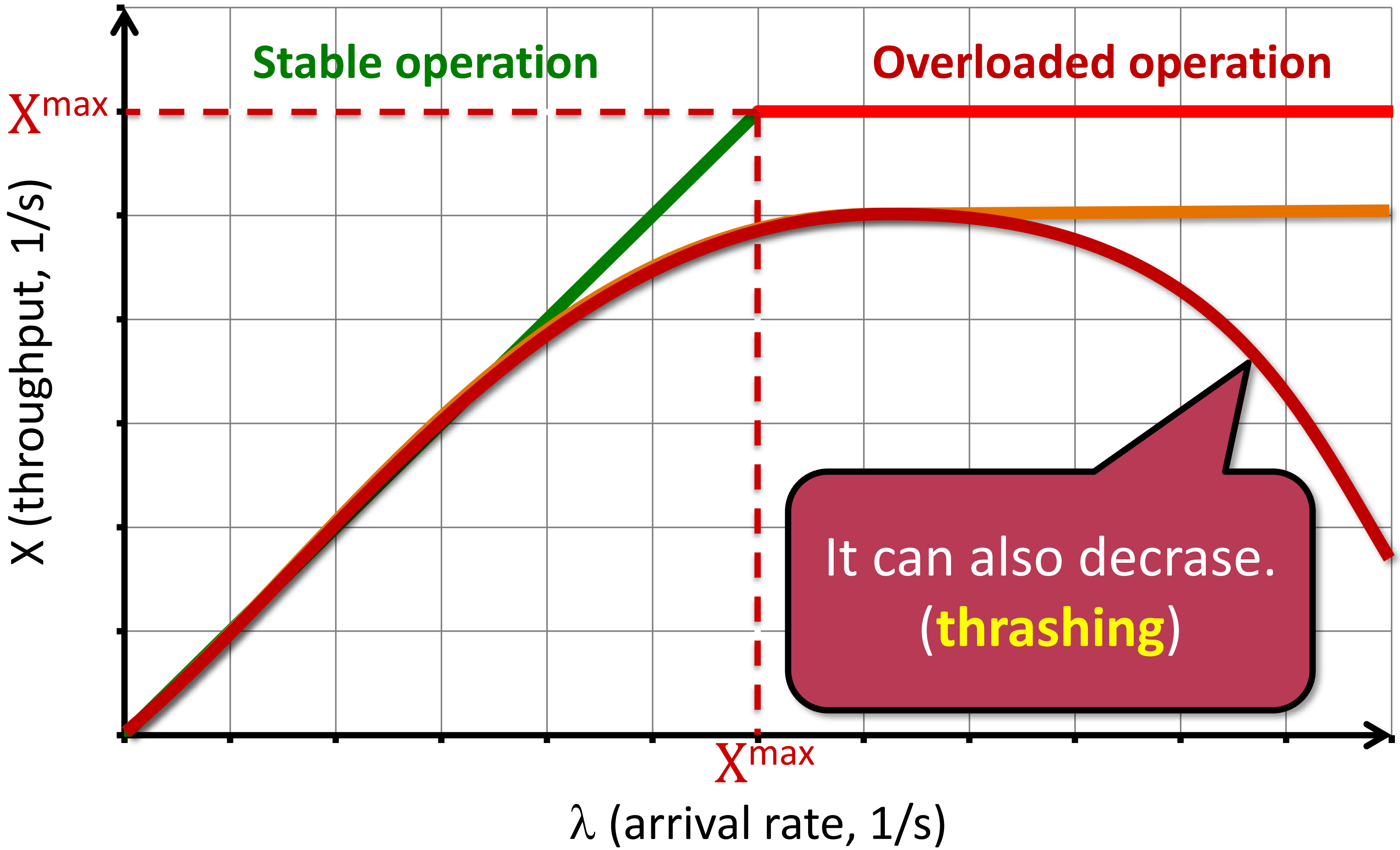
Approximative Load Function



Real Load Diagram



Real Load Diagram



Definitions

Maximum throughput: **maximal reachable** throughput

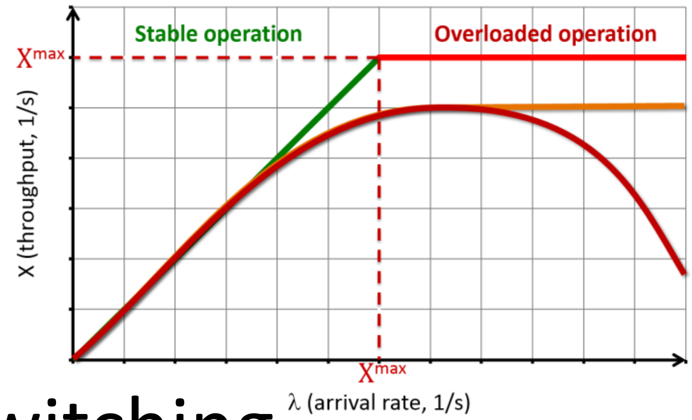
- Symbol: X^{max}

Utilisation: **ratio** of the actual and the maximum throughput

- Symbol: $U = \frac{X}{X^{max}}$

Thrashing: throughput decreases during overloaded operation

Effects Ignored by the Model



- Cost of task switching
  - Cleanup after processing
  - Preparation for the next process
- Computation cost of resource switching
- Multiple overload
  - Multiple resources (e.g. servers) can be overloaded
  - E.g. if there is an accident on the M7, there is also a traffic jam on the country road 7

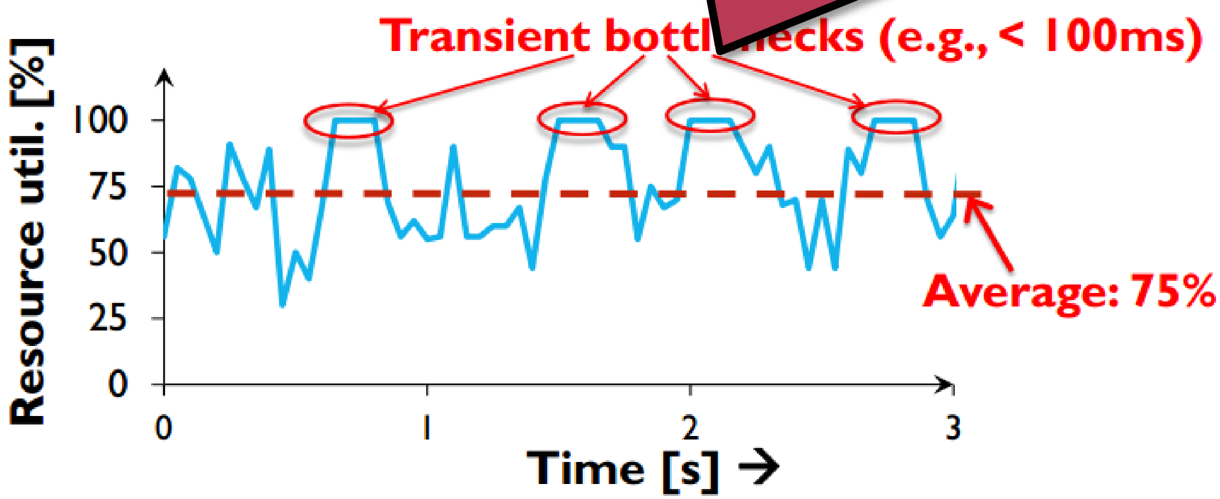
Scared Off Requests

- Number of requests are only independent of waiting time in case of fanatic customers

Effects of Load Fluctuation

- Average values vs. **real load**

In case of an overloaded system the waiting time can be **2-3 magnitued higher**

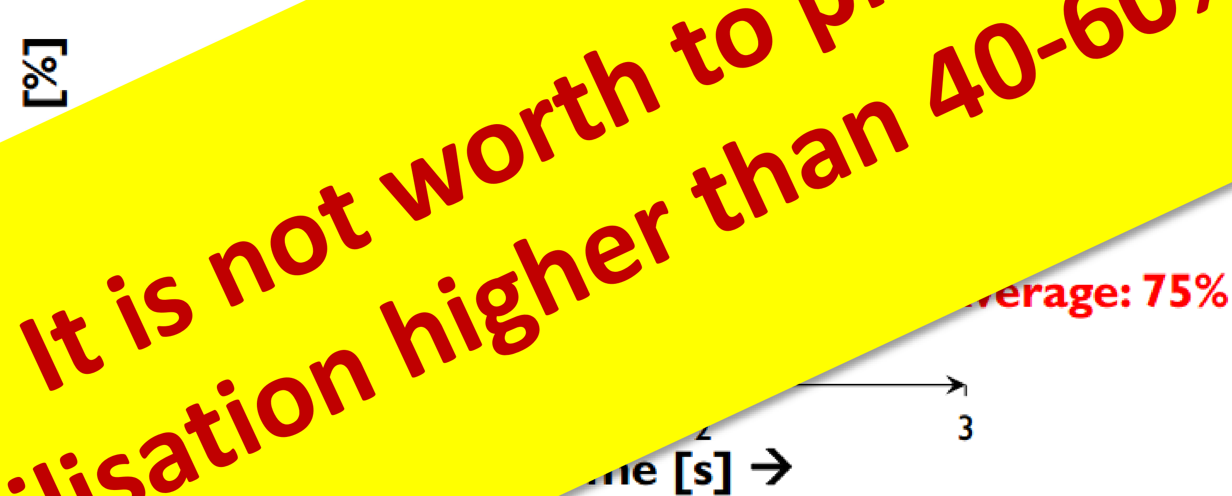


Qingyang Wang, Yasuhiko Kanemasa, Jack Li, Deepal Jayasinghe, Toshihiro Shimizu, Masazumi Matsubara, Motoyuki Kawaba, Calton Pu, "Detecting Transient Bottlenecks in n-Tier Applications through Fine-Grained Analysis", In Proc. of the 33rd International Conference on Distributed Computing Systems (**ICDCS'13**), Philadelphia, Pennsylvania, July 2013.

Effects of Load Fluctuation

- Average values vs. **real load**

In case … loaded
sy… me can
…gher



It is not worth to plan a utilisation higher than 40-60%!

Qingyang Wang, Yasu… …sa, Jack Li, Deepal Jayasinghe, Toshihiro Shimizu, Masazumi Matsubara, Motoyuki Kawaba, Calton Pu, "Detecting Transient Bottlenecks in n-Tier Applications through F… Grained Analysis", In Proc. of the 33rd International Conference on Distributed Computing Systems (**ICDCS'13**), Philadelphia, Pennsylvania, July 2013.

Neptun Course Sign-ups

NEPTUN

Max. # of concurrent users

Max. # of concurrent users at optimal operation

Az elmúlt időkben előfo… …has…

Dátum	Jelleg	Max. felhasználó	Op. felhasználó
2010.11.29 18:00	vizsga	7303	4623
2010.12.22 06:00	tárgy (EO)	831	831
2011.01.10 18:00	tárgy	12062	4837
2011.01.12 18:00	tárgy (EP)	1765	1765
2011.01.31 16:00	tárgy	1519	1519
2011.05.02 18:00	vizsga	2761	2761
2011.06.07 18:00	tárgy	6095	6095
2011.11.28 18:00	vizsga	4897	4897
2012.01.16 18:00	tárgy	8120	5328
2012.01.30 16:00	tárgy	1703	1703
2012.05.02 18:00	vizsga	2603	2603

Neptun Course Sign-ups

≈ Arrival rate (λ)

≈ Max. Throughput (X^{max})

Az elmúlt időkben előf … ha…

Dátum	Jelleg	Max. felhasználó	Op. felhasználó
2010.11.29 18:00	vizsga	7303	4623
2010.12.22 06:00	tárgy (EO)	831	831
2011.01.10 18:00	tárgy	12062	4837
2011.01.12 18:00	tárgy (EP)	1765	1765
2011.01.31 16:00	tárgy	1519	1519
2011.05.02 18:00	vizsga	2761	2761
2011.06.07 18:00	tárgy	6095	6095
2011.11.28 18:00	vizsga	4897	4897
2012.01.16 18:00	tárgy	8120	5328
2012.01.30 16:00	tárgy	1703	1703
2012.05.02 18:00	vizsga	2603	2603

When was Neptun overloaded?

Neptun Course Sign-ups

≈ Arrival rate (λ)

≈ M… (X^{max})

Az elmúlt időkben elő… ha…

Dátum	Jelleg		
2010.11.29 18:00	vizsga		
2010.12.22 06:00	tárgy (…		
2011.01.10 18:00			
2011.01.12 18:00			
2011.01.31 1…			
2011…			2761
2011.0…			6095
2011.11…			4897
2012.01.1…			5328
2012.01.30…		1703	1703
2012.05.02 1…		2603	2603

The performance of an overloaded system can be degraded drastically, it can even crash!

…n was Neptun overloaded?

Neptun Course Sign-ups

≈ Arrival rate (λ)

≈ Max. Throughput (X^{max})

Az elmúlt időkben el[illegible] has[illegible]

Dátum	Jelleg	Max. felhasználó	Op. felhasználó
2010.11.29 18:00	vizsga	7303	4623
2010.12.22 06:00	tárgy (EO)	831	831
2011.01.10 18:00	tárgy	12062	4837
2011.01.12 18:00	tárgy (EP)	1765	1765
2011.01.31 16:00	tárgy	1519	1519
2011.05.02 18:00	vizsga	2761	2761
2011.06.07 18:00	tárgy	6095	6095
2011.11.28 18:00	vizsga	4897	4897
2012.01.16 18:00	tárgy	8120	5328
2012.01.30 16:00	tárgy	17[illegible]	[illegible]
2012.05.02 18:00	vizsga	26[illegible]	[illegible]

Correctly configured servers, appropriate capacity planning!

Neptun Course Sign-ups

Protection against (D)DoS attacks is based on the same principles

…ate (λ)

≈ M… …ut (X^{max})

			…áló
2010.11.29 16:00			
2010.12.22 06:00	tárgy…		
2011.01.10 18:00			
2011.01.12 18:00			
2011.01.31 1…			
2011.05…			2761
2011.06.0…			6095
2011.11.28			4897
2012.01.16 1…			5328
2012.01.30 16…		17…	
2012.05.02 18:0…		26…	

Limiting the number of users can keep the system in stable state

Correctly configured servers, appropriate capacity planning!

RESOURCE MODELLING

Why is there a *maximum* of the throughput?

Example: Mid-Term Exam Grading

- Grading a single exam takes **15 mins**
- How many exams can be graded by a single lecturer in an hour?

$$X_{(1)}^{max} = \frac{1\ exam}{15\ min.} = 4\ \frac{exams}{h}$$

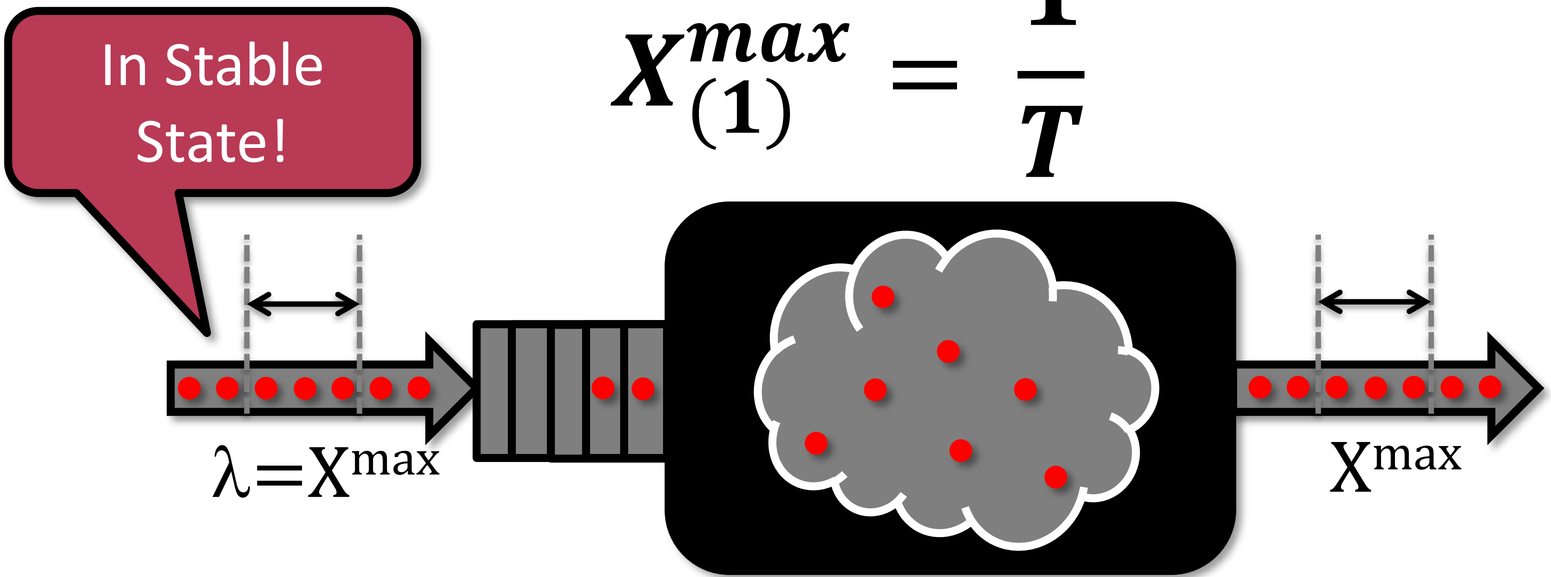
- How many exams can be graded by **eight** lecturers?

$$X_{(8)}^{max} = 8 \times X_{(1)}^{max} = 8 \times \frac{1\ exam}{15\ min.} = 32\ \frac{exams}{h}$$

Max. Throughput of a Single Server System

- If **only a single** request can be processed
  - e.g. served by a single server, or by working with the same (shared) variables
  - the remaining requests are queued
- Then with **the average execution time T**:

$$X_{(1)}^{max} = \frac{1}{T}$$



Max. Throughput of a Single Server System

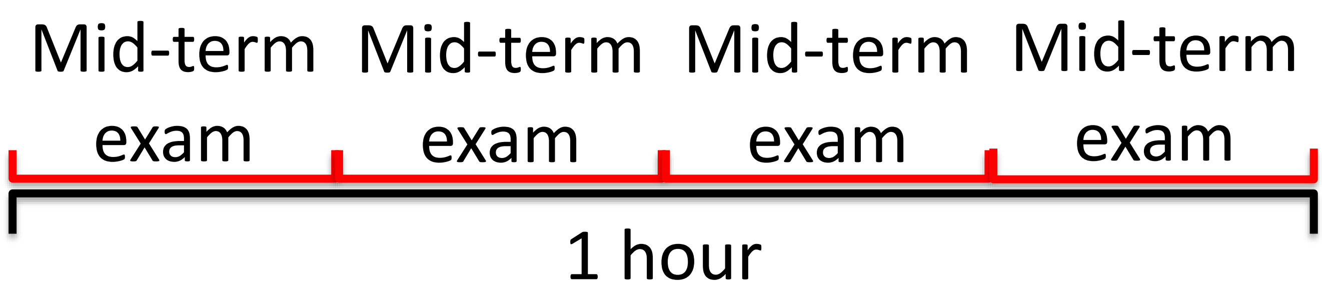
- If **only a single** request can be processed
  - e.g. served by a single server, or by working with the same (shared) variables
  - the remaining requests are queued
- Then with **the average execution time T**:

$$X_{(1)}^{max} = \frac{1}{T}$$

„How many requests can be processed in a unit of time?”

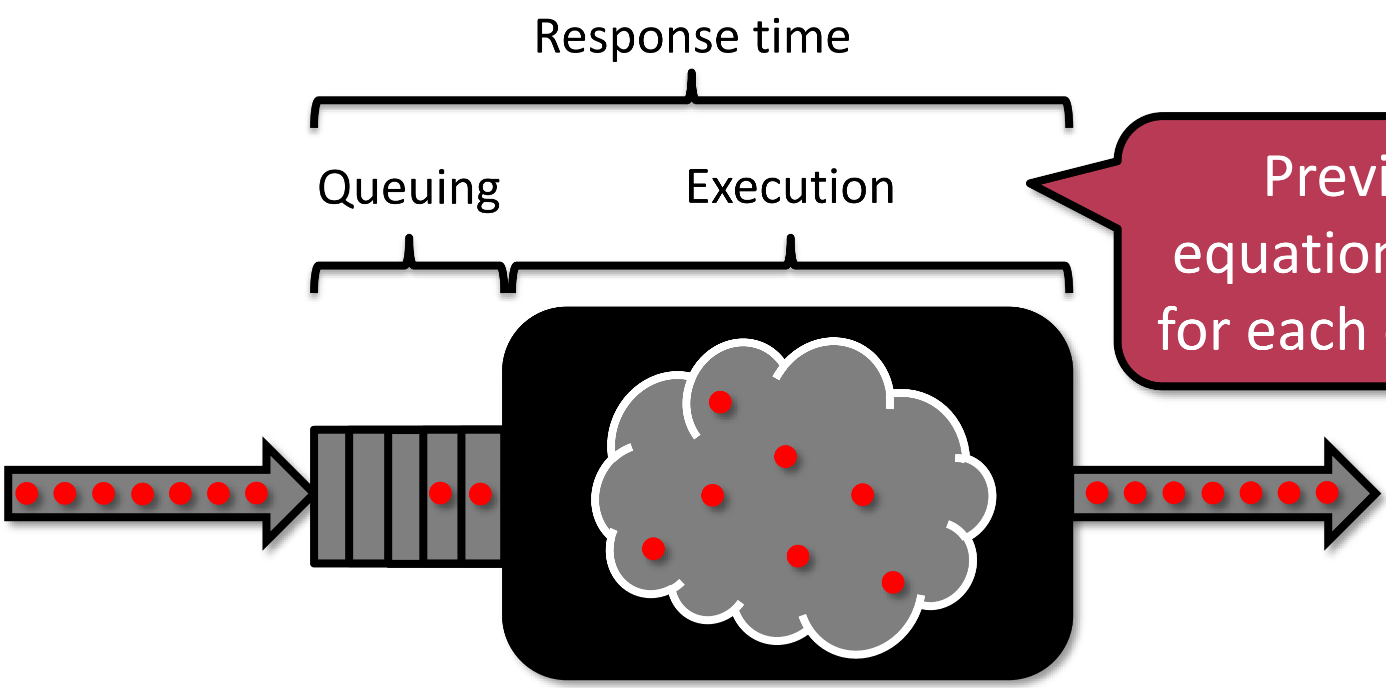
$T_{exam} = 15 \text{ min.}$

$X_{max} = 4 \frac{1}{h}$



Different Times to Measure

- **Queuing time:** request waiting for a resource
- **Execution time:** request under processing
- **Response time:** Queuing + Execution times



Previous equations work for each of them

Utilisation of a Single Server System

- Utilisation: „Ratio of the maximum and the actual throughputs”

$$X_{(1)}^{max} = \frac{1}{T} \quad \Rightarrow \quad X_{(1)}^{max} \times \mathrm{T} = 1$$

Utilisation of a Single Server System

- Utilisation: „Ratio of the maximum and the actual throughputs"

$$X_{(1)}^{max} = \frac{1}{T} \quad \Rightarrow \quad X_{(1)}^{max} \times \mathrm{T} = 1 = \mathrm{U}$$

- The equation of the utilisation:

$$\mathbf{U} = \mathbf{X} \times \mathbf{T}$$

- Intuition:

„If X requests arrive in a unit of time,

each of which takes T time to process,

what percentage of time is spent busy?"

Utilisation of a Single Server System

- Utilisation: „Ratio of the maximum and the actual throughputs"

$$X_{(1)}^{max} = \frac{1}{T} \quad \Rightarrow \quad X_{(1)}^{max} \times \mathrm{T} = 1 = \mathrm{U}$$

- The equation of the utilisation:

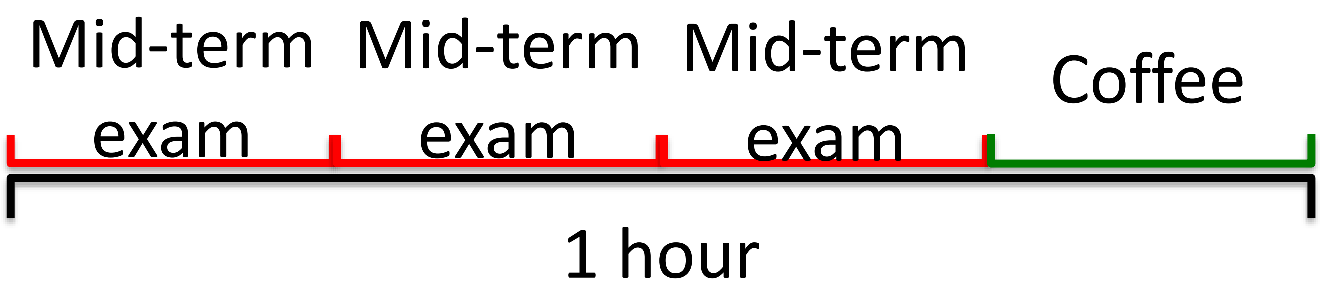
$$\mathrm{U} = \mathrm{X} \times \mathrm{T}$$

- Intuition:

$$\mathrm{X} = 3\,\frac{1}{h}$$

$$\mathrm{T}_{\mathrm{exam}} = 15\ \mathrm{min}$$

$$\mathrm{U} = 75\%$$

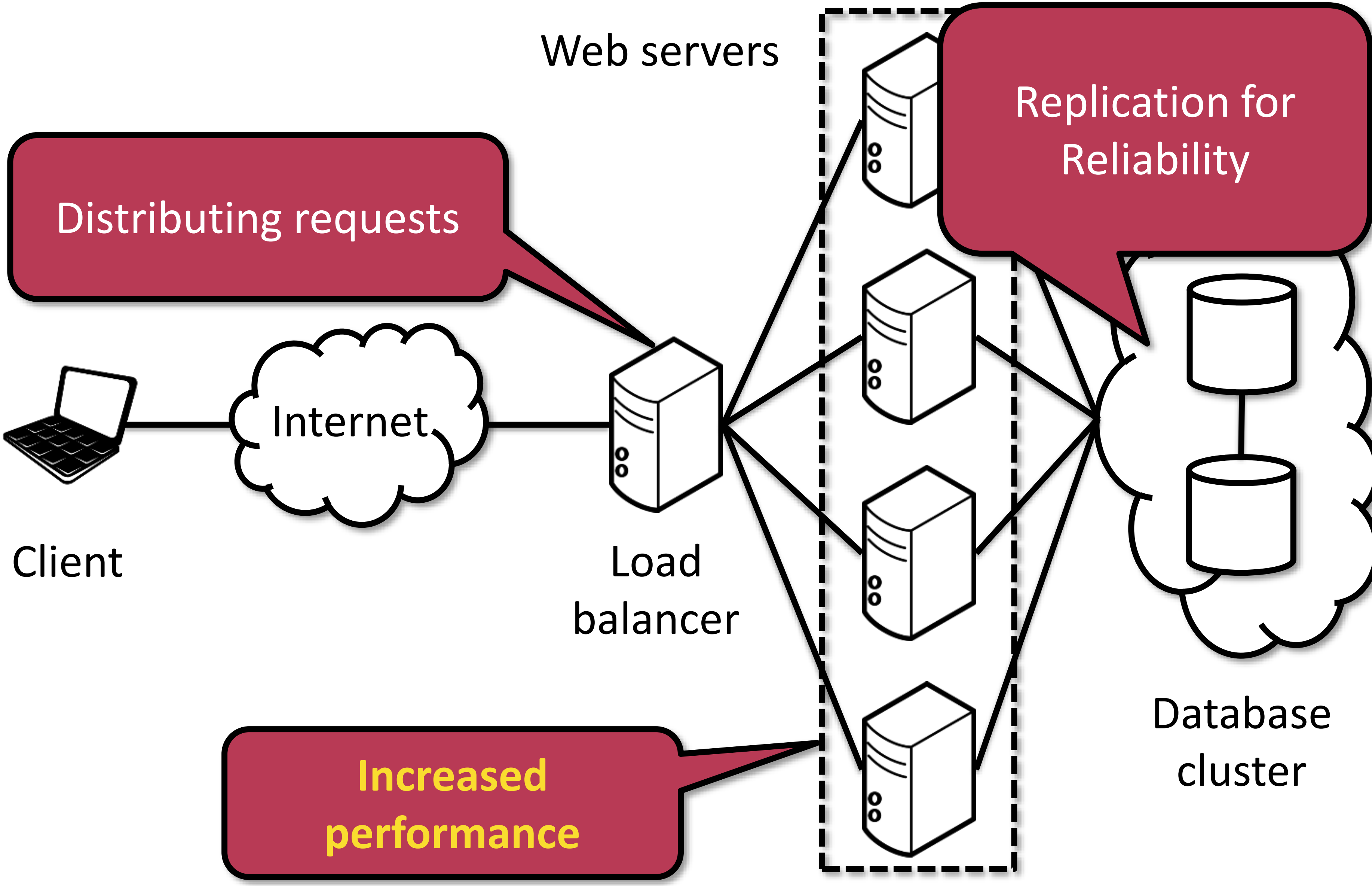


Outlook: Scalability

- **Vertical scaling** (Scale-up):
  - The **power** of the processing units is **increased**
  - E.g. stronger CPU, more RAM
  - Simple and great ☺
  - Technological constraints ☹
- **Horizontal scaling** (Scale-out):
  - The **number** of the processing units is **increased**
  - E.g. Multiple (core) CPU, multiple servers
  - Theoretically no limitations ☺
  - Extra complexity ☹

Slace-out in Everyday Life

Scale-out of Neptun



Max. Throughput of a Multiple Server System

- If **maximum** K requests are allowed to be served simultaneously
  - e.g. K clustered servers can process them
  - the remaining requests are queued
- Then with **the average execution time T**:

$$X_{(K)}^{max} = K \times X_{(1)}^{max} = \frac{K}{T}$$

Max. Throughput of a Multiple Server System

- If **maximum** K requests are allowed to be served simultaneously
  - e.g. K clustered servers can process them
  - the remaining re
- Then with **the av**

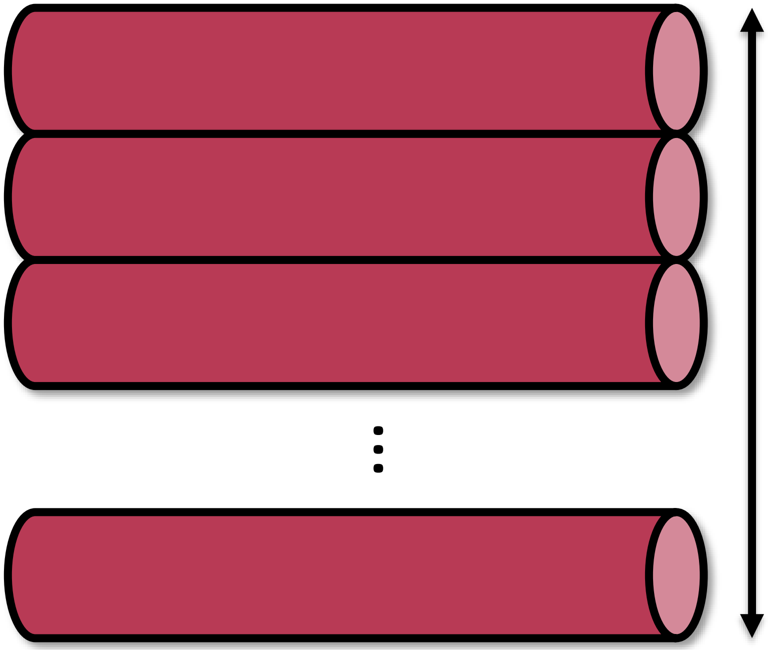
With more resources and parallelisation the system is **scalable**.

$$X_{(K)}^{max} = K \times X_{(1)}^{max} = \frac{K}{T}$$

Pipe Analogy

Single Resource Instance

K Resource Instances (with free choice)



Throughput:

$$X_{max}^{(1)}$$

Throughput :

$$X_{max}^{(K)} = \mathrm{K} \times X_{max}^{(1)}$$

Utilisation of a Multiple Server System

- Analogous with the previous calculation:

$$X_{(K)}^{max} = \frac{K}{T} \quad \Rightarrow \quad X_{(K)}^{max} \times T = K \; ?$$

Utilisation of a Multiple Server System

- Analogous with the previous calculation:

$$X_{(K)}^{max} = \frac{K}{T} \quad \Rightarrow \quad X_{(K)}^{max} \times T = K = K \times U$$

- The equation of utilisation in this case:

$$U = \frac{X}{K} \times T$$

- Intuition:

„What is the utilisation of a single instance by the average throughput of one instance?”

„What percentage of K units of time is the processing time of a single instance?”

Utilisation of a Multiple Server System

- Analogous with the previous calculation:

$$X^{max}_{(K)} = \frac{K}{T} \quad \Rightarrow \quad X^{max}_{(K)} \times T = K$$

- The equation of utilisation in this case:

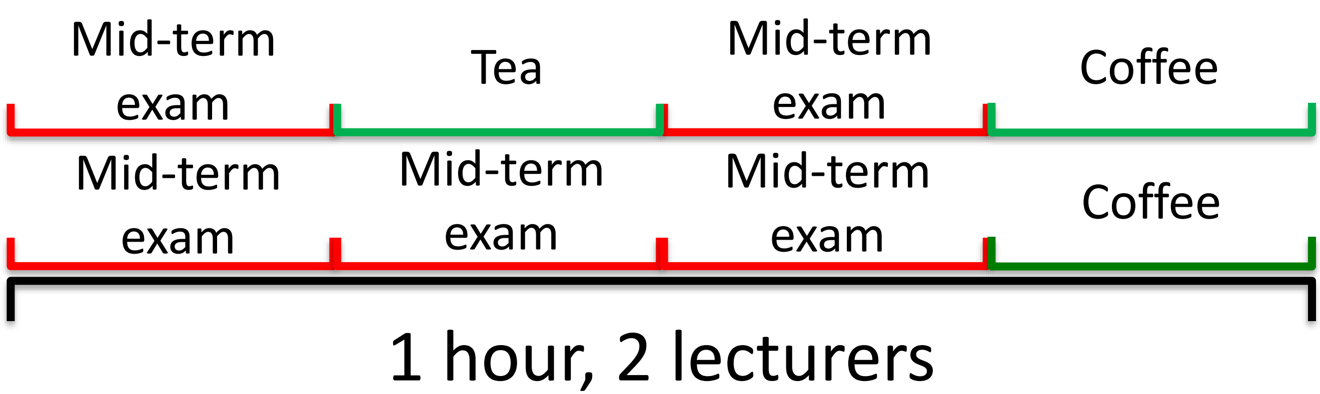
$$U = \frac{X}{K} \times T$$

- Intuition:

$X = 5\,\frac{1}{h}$

$T_{exam} = 15\text{ min.}$

$U = 62{,}5\%$



Summary

- **Stable State:**
  - Calculating with average values
  - $\lambda = X$ (arrival rate = throughput)
- **Maximum throughput:**
  - Upper bound of the reachable throughput
  - $X^{max} = \frac{K}{T}$ (in case of K resource instances)
- **Utilization:**
  - Ratio of the actual and the maximum throughputs
  - $U = \frac{X}{K} \times T$ (in case of K resource instances)