technische universität
dortmund

Präsentation Bachelor Arbeit:
Simulation von Online Scheduling-Algorithmen für parallele Systeme

Hendrik Rassmann

November 02, 2020

TU Dortmund University - Department of Computer Science

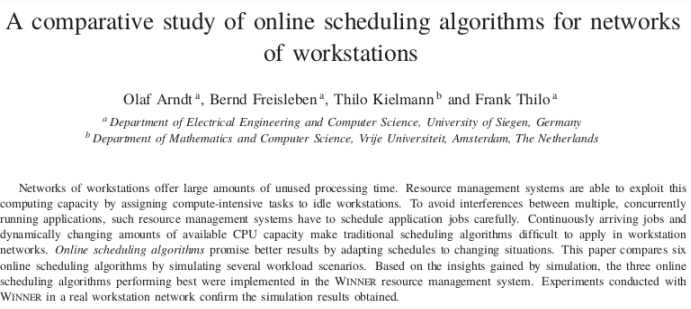
Übersicht

Problemstellung

Simulationsaufbau

Outlook

Startpunkt



A comparative study of online scheduling algorithms for networks of workstations

Olaf Arndt [a], Bernd Freisleben [a], Thilo Kielmann [b] and Frank Thilo [a]

[a] *Department of Electrical Engineering and Computer Science, University of Siegen, Germany*
[b] *Department of Mathematics and Computer Science, Vrije Universiteit, Amsterdam, The Netherlands*

Networks of workstations offer large amounts of unused processing time. Resource management systems are able to exploit this computing capacity by assigning compute-intensive tasks to idle workstations. To avoid interferences between multiple, concurrently running applications, such resource management systems have to schedule application jobs carefully. Continuously arriving jobs and dynamically changing amounts of available CPU capacity make traditional scheduling algorithms difficult to apply in workstation networks. *Online scheduling algorithms* promise better results by adapting schedules to changing situations. This paper compares six online scheduling algorithms by simulating several workload scenarios. Based on the insights gained by simulation, the three online scheduling algorithms performing best were implemented in the WINNER resource management system. Experiments conducted with WINNER in a real workstation network confirm the simulation results obtained.

- Network
- Online
- Experiments ... confirm

Problemstellung

Problemstellung

Ausgehend von:

- Rechner/*Nodes* unterschiedlicher *Geschwindigkeiten* bilden zusammen ein (Rechen)*Cluster*

Problemstellung

Ausgehend von:

- Rechner/*Nodes* unterschiedlicher *Geschwindigkeiten* bilden zusammen ein (Rechen)*Cluster*
- Aufträge/*Jobs* kommen im laufenden Betrieb rein (*online*)

Problemstellung

Ausgehend von:

- Rechner/*Nodes* unterschiedlicher *Geschwindigkeiten* bilden zusammen ein (Rechen)*Cluster*
- Aufträge/*Jobs* kommen im laufenden Betrieb rein (*online*)
- Aufträge unterscheiden sich bez. *Bearbeitungszeit*, *Anzahl* an benötigten Knoten und *Ankunftszeitpunkt*

Problemstellung

Ausgehend von:

- Rechner/*Nodes* unterschiedlicher *Geschwindigkeiten* bilden zusammen ein (Rechen)*Cluster*
- Aufträge/*Jobs* kommen im laufenden Betrieb rein (*online*)
- Aufträge unterscheiden sich bez. *Bearbeitungszeit*, *Anzahl* an benötigten Knoten und *Ankunftszeitpunkt*
- Aufträge sollen auf eine 'geeignete' Art und Weise bearbeitet werden

Zielfunktionen [1]

- *makespan* "Zeit bis Feierabend"

[1] Kriterien von Arndt. et al.

Zielfunktionen [1]

- *makespan* "Zeit bis Feierabend"
- *average flow time* "Supermarkt - Nur der Apfel? Gehen Sie doch gerne vor"

[1] Kriterien von Arndt. et al.

Zielfunktionen [1]

- *makespan* "Zeit bis Feierabend"
- *average flow time* "Supermarkt - Nur der Apfel? Gehen Sie doch gerne vor"
- *average waiting time*

[1] Kriterien von Arndt. et al.

Zielfunktionen [1]

- *makespan* "Zeit bis Feierabend"
- *average flow time* "Supermarkt - Nur der Apfel? Gehen Sie doch gerne vor"
- *average waiting time*
- *maximum waiting time* "Restaurant - Wenn ein Tisch nicht bedient wird, kommen Gäste nicht mehr wieder"

[1] Kriterien von Arndt. et al.

Scheduling-Algorithmen (Referentiell Transparent)[2]

Scheduling-Algorithmus : *Warteliste* $\rightarrow$ *Auftrag*

[2] Auswahl von Arndt. et al.

Scheduling-Algorithmen (Referentiell Transparent)[2]

Scheduling-Algorithmus : *Warteliste* $\rightarrow$ *Auftrag*

- First in First out (*FiFo*)

[2] Auswahl von Arndt. et al.

Scheduling-Algorithmen (Referentiell Transparent)[2]

Scheduling-Algorithmus : *Warteliste* $\rightarrow$ *Auftrag*

- First in First out (*FiFo*)
- Shortest Processing Time first (*SPT*)

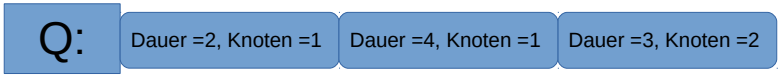
[2] Auswahl von Arndt. et al.

Scheduling-Algorithmen (Referentiell Transparent)[2]

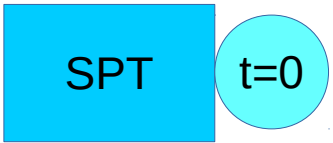
Scheduling-Algorithmus : *Warteliste* $\rightarrow$ *Auftrag*

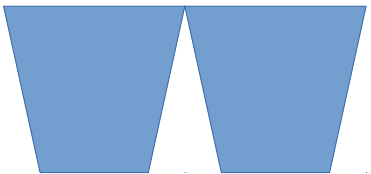
- First in First out (*FiFo*)
- Shortest Processing Time first (*SPT*)
- Greates Processing Time first (*GPT*)

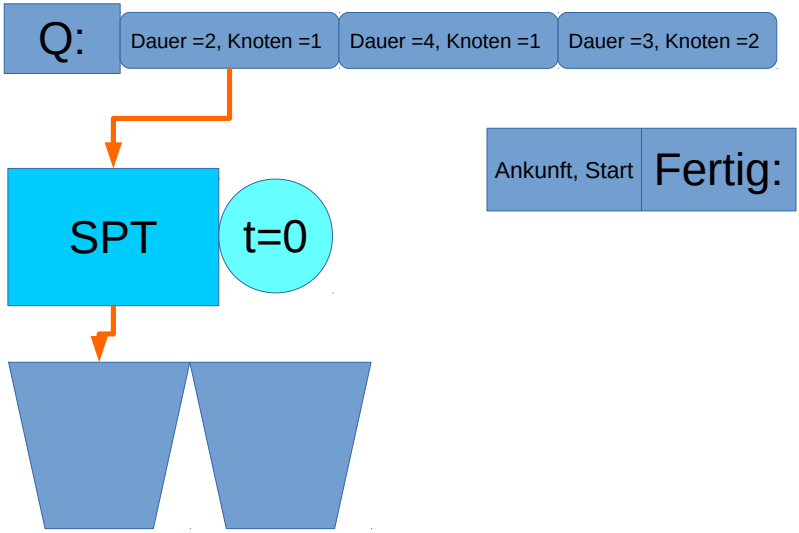
[2] Auswahl von Arndt. et al.

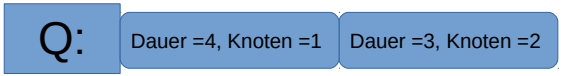

Q:
Dauer =2, Knoten =1
Dauer =4, Knoten =1
Dauer =3, Knoten =2


Ankunft, Start
Fertig:

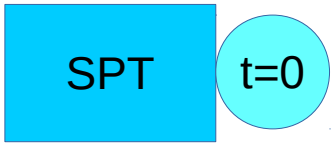

SPT
t=0



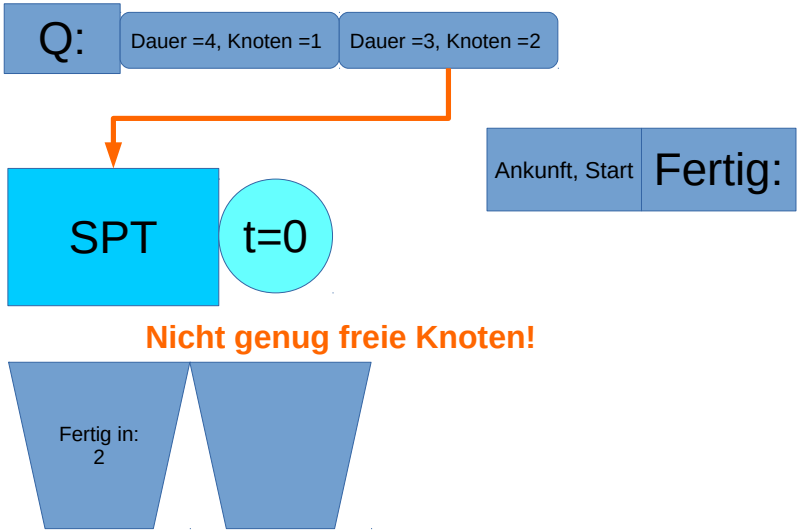

Q:
Dauer =2, Knoten =1
Dauer =4, Knoten =1
Dauer =3, Knoten =2
Ankunft, Start
Fertig:
SPT
t=0

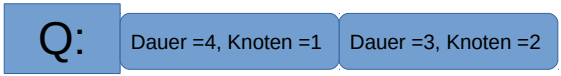

Q:
Dauer =4, Knoten =1
Dauer =3, Knoten =2


Ankunft, Start
Fertig:

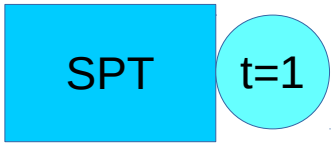

SPT
t=0


Fertig in:
2

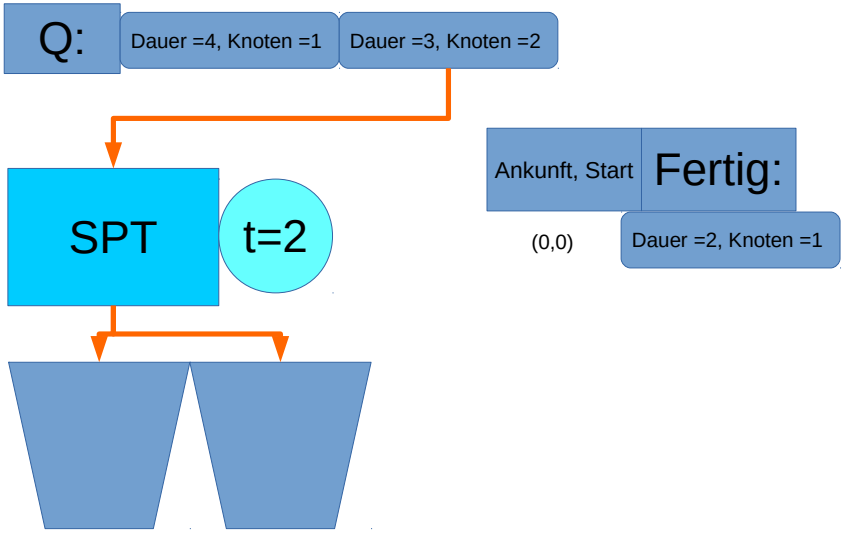

Q:
Dauer =4, Knoten =1
Dauer =3, Knoten =2
Ankunft, Start
Fertig:
SPT
t=0
Nicht genug freie Knoten!
Fertig in:
2

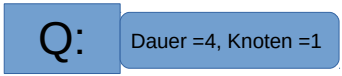

Q:
Dauer =4, Knoten =1
Dauer =3, Knoten =2

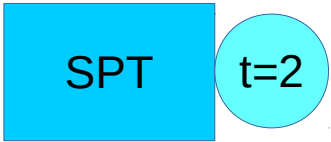

Ankunft, Start
Fertig:

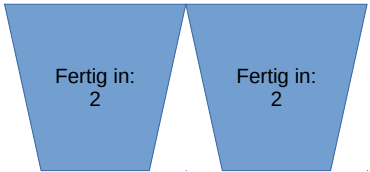

SPT
t=1


Fertig in:
1

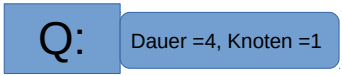

Q:
Dauer =4, Knoten =1
Dauer =3, Knoten =2
SPT
t=2
Ankunft, Start
Fertig:
(0,0)
Dauer =2, Knoten =1

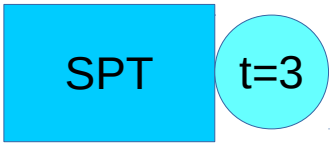

Q:
Dauer =4, Knoten =1

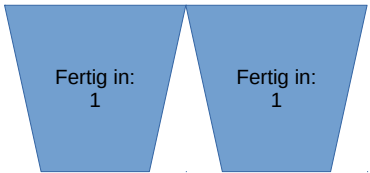

SPT
t=2


Fertig in:
2
Fertig in:
2

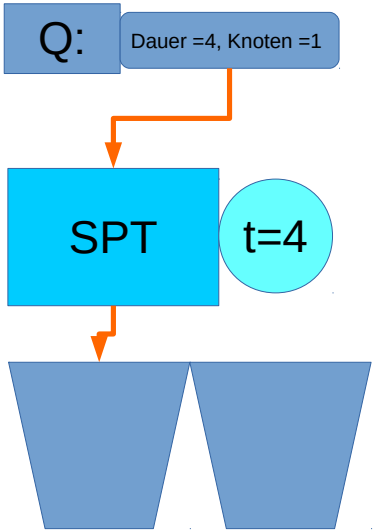

Ankunft, Start
Fertig:
(0,0)
Dauer =2, Knoten =1


Q:
Dauer =4, Knoten =1


SPT
t=3


Fertig in:
1
Fertig in:
1


Ankunft, Start
Fertig:
(0,0)
Dauer =2, Knoten =1



Ankunft, Start	Fertig:
(0,0)	Dauer =2, Knoten =1
(0,2)	Dauer =3, Knoten =2

Q:

SPT

t=4

Fertig in:
4

Ankunft, Start	Fertig:
(0,0)	Dauer =2, Knoten =1
(0,2)	Dauer =3, Knoten =2

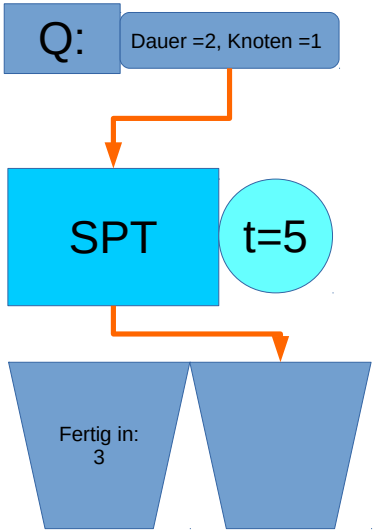
Q:

SPT

t=5

Fertig in:
3

Ankunft, Start	Fertig:
(0,0)	Dauer =2, Knoten =1
(0,2)	Dauer =3, Knoten =2



Ankunft, Start	Fertig:
(0,0)	Dauer =2, Knoten =1
(0,2)	Dauer =3, Knoten =2

Q:

SPT

t=5

Ankunft, Start	Fertig:
(0,0)	Dauer =2, Knoten =1
(0,2)	Dauer =3, Knoten =2

Fertig in:
3

Fertig in:
2

Q:

SPT

t=6

Ankunft, Start	Fertig:
(0,0)	Dauer =2, Knoten =1
(0,2)	Dauer =3, Knoten =2

Fertig in:
2

Fertig in:
1

Q:

SPT

t=7

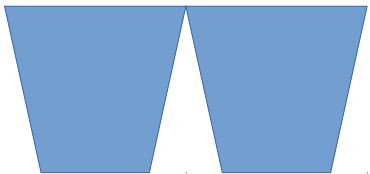
Fertig in:
1

Ankunft, Start	Fertig:
(0,0)	Dauer =2, Knoten =1
(0,2)	Dauer =3, Knoten =2
(5,5)	Dauer =2, Knoten =1

Q:

SPT

t=8



Ankunft, Start	Fertig:
(0,0)	Dauer =2, Knoten =1
(0,2)	Dauer =3, Knoten =2
(5,5)	Dauer =2, Knoten =1
(0,4)	Dauer =4, Knoten =1

Scheduling-Algorithmen (Referentiell Intransparent)

- Random

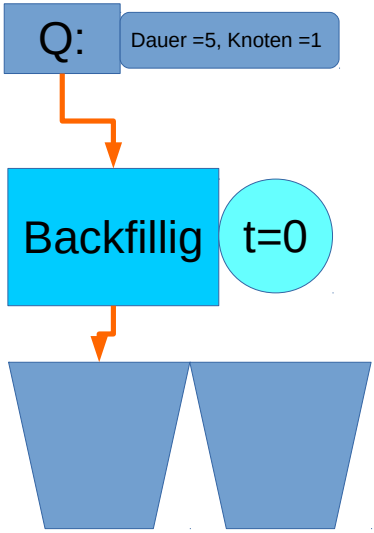
Scheduling-Algorithmen (Referentiell Intransparent)

- Random
- First Fit (Wähle den am längsten wartende Auftrag, der sofort gestartet werden kann)

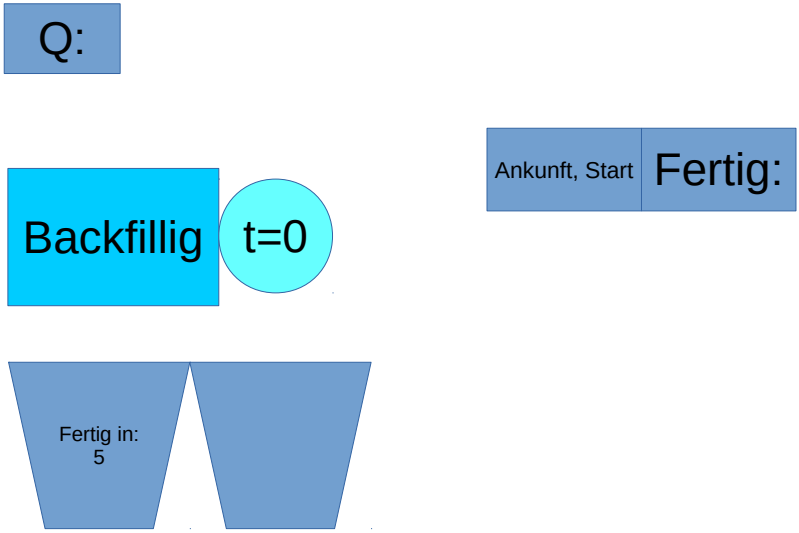
Scheduling-Algorithmen (Referentiell Intransparent)

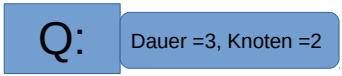
- Random
- First Fit (Wähle den am längsten wartende Auftrag, der sofort gestartet werden kann)
- Backfilling (Wähle einen startbaren Auftrag, der wenn er jetzt gestartet wird, terminiert bevor der am längsten wartende Auftrag starten wird)

Beispiel Backfilling



Ankunft, Start | Fertig:

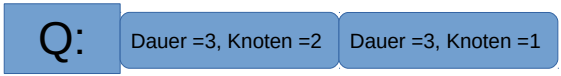

Q:
Ankunft, Start
Fertig:
Backfillig
t=0
Fertig in:
5


Q:
Dauer =3, Knoten =2


Ankunft, Start
Fertig:

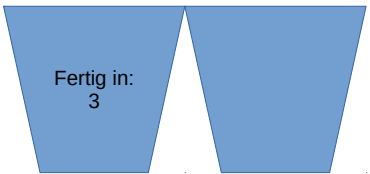

Backfillig
t=1

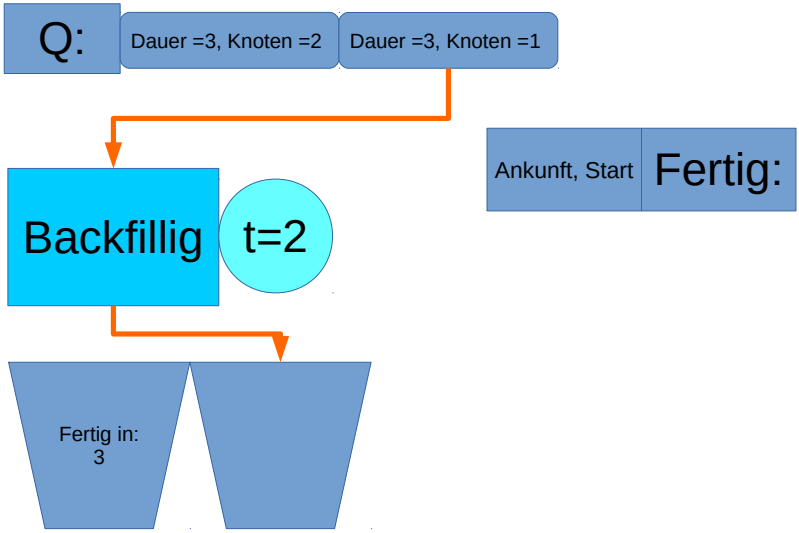

Fertig in:
4


Q:
Dauer =3, Knoten =2
Dauer =3, Knoten =1


Ankunft, Start
Fertig:


Backfillig
t=2


Fertig in:
3


Q:
Dauer =3, Knoten =2
Dauer =3, Knoten =1
Ankunft, Start
Fertig:
Backfillig
t=2
Fertig in:
3

Simulationsaufbau

Simulations Paradigma[3]

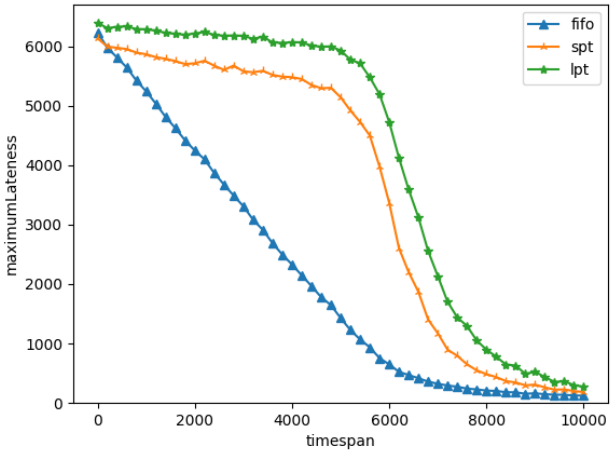
- Aktivitätsorientiert (Wettersystem, Physik Simulation in Videospielen)
- Ereignisorientiert (Fahrplan)
- Prozessorientiert (Parallele Programmierung)

[3] Matloff, Norm: Introduction to discrete-event simulati- on and the simpy language. In: Davis, CA. Dept of Com- puter Science. University of California at Davis. Retrieved on August 2 (2008), Nr. 2009, S. 133

Simulation und Auswertung

- Arndt et al. designen Experimente, um unterschiedliches Verhalten der Algorithmen zu demonstrieren
- Variieren einen Parameter und vergleichen gewählte Zielfunktionen

Beispiel



- timespan - Spätester Ankunftszeitpunkt wird variiert
- Latenz am besten bei FiFo
- An beiden Extremen ähnliches Verhalten

Outlook

Outlook

The font size should be as large as possible

- *Never smaller than the age of the oldest audience member*
- Don't vary sizes to much, never within a slide
- Stick to one font
- Don't overuse formatting like bold, italics, alert, etc.

But you knew all that already...

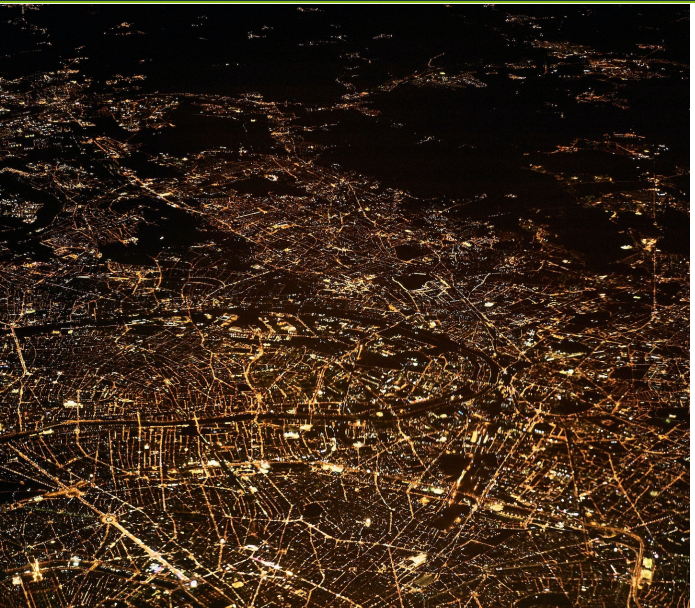
Querying Data used to be simple...

- Tutorial for Beamer https://www.overleaf.com/learn/latex/Beamer_Presentations:_A_Tutorial_for_Beginners_(Part_1)%E2%80%94Getting_Started
- Quelle Template: https://github.com/matze/mtheme
- Quelle Bild: https://unsplash.com/photos/52gEprMkp7M

How about a very large image?

- Itemize Environments are important
- We can highlight words

How about a very large image?



- Itemize Environments are important
- We can highlight words
- We can add animations. (please don't over-do it)

Clever Last Words to Stimulate Discussion