COMP 3430

Operating Systems

May 8th, 2019

Goals

By the end of today's lecture, you should be able to:

- Justify whether or not an OS is necessary for a problem or hardware.
- Compare and contrast basic scheduling algorithms.

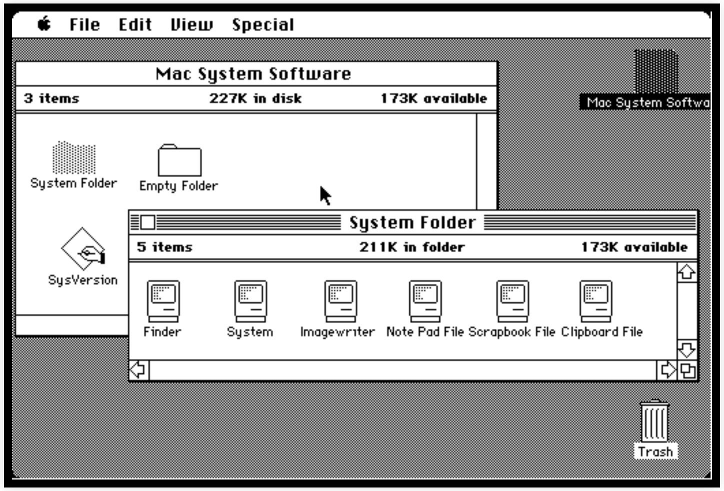
Pixabay License

Quiz

Did you read the text?

Operating systems…?

- Operating systems are **everywhere**.
  - They live in our pockets!
- …*do* they live **everwhere**?



Fair Use

Where do they live?

By yourself: Write down a list of 3-5 **electronic devices** that you own or have owned.

© ICCNS CC BY 2.0

My device

Considering the responsibilities of an OS, *does* this device have an OS?

- Some things to think about:
  - How many **programs** are running at the same time?
  - How does a new program get **loaded onto** this device?
  - How many **users** does this device have?
  - How much does the program **know** about the hardware?

Public Domain

Does an OS live here?

With the person beside you:

1. Share the list of devices you came up with.
2. Take turns: *does* an OS live on this device? Why? Why not?

Does an OS live here?

- Generally, the answer is: “It depends”.
  - How much **control** does a program need over hardware?
  - Do programs need to run at the same time?
  - Related: Does this device have more than one user?

Basic scheduling

- The OS needs to manage access to machine resources.
    - Managing access to the CPU is *scheduling*.
- Historically, we've seen three ways to schedule programs:
    1. Manually, one at a time,
    2. Automatically, one at a time,
    3. Automatically, *many* at a time.
- Let's think about scheduling *one at a time*.

Pixabay License

Scheduling *algorithms*

We'll consider *three* algorithms (there are **many**)

1. First come, first served.
2. Priority scheduling.
3. Shortest remaining time first.

First come, first served

- The simplest possible solution for scheduling programs.
- First program started is the first to be run.
- Second program started is the second to be run.
- Third program started is the third to be run.
- Fourth program started is the fourth to be run.
- Fifth program started is the fifth to be run.
- Sixth program started is the sixth to be run.
- Seventh program started is the seventh to be run.
- Eighth program started is the eighth to be run…

Pixabay License

First come, first served

Some things to ponder:

1. What kind of a **data structure** would be best suited for this algorithm?
2. How do we evaluate the performance of this algorithm? What can we **measure**?
3. Is this algorithm **good**? Can you even answer this question?

Priority Scheduling

- A *straightforward* algorithm for scheduling programs.
- Programs are started in some order.
  - Programs have a *priority* (some integer value)
  - Programs with higher priority can run *before* programs with lower priority – *regardless* of submission order.

Pixabay License

Priority Scheduling

Similar questions:

1. What kind of a **data structure** would be best suited for this algorithm?
2. *Who* is responsible for assigning priority? How is the decision to choose a priority made?
3. Is this algorithm **good**? Can you even answer this question?

Shortest remaining time first

- A *straightforward* algorithm for scheduling programs.
- Programs are started in some order.
  - Programs have a *known remaining time* (some integer value)
  - Programs with lower remaining time can run *before* programs with higher remaining time – *regardless* of submission order.

Pixabay License

Shortest remaining time first

Similar questions:

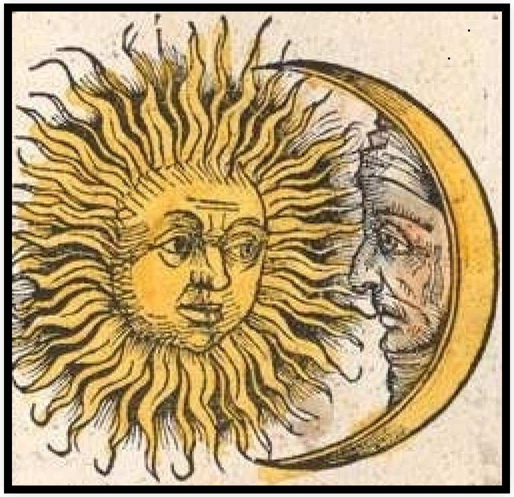
1. What kind of a **data structure** would be best suited for this algorithm?
2. How do you measure *remaining time* of a program? *Can* you measure remaining time generally?
3. What kind of problems might this algorithm have in practice?
4. Is this algorithm **good**? Can you even answer this question?

Basic scheduling

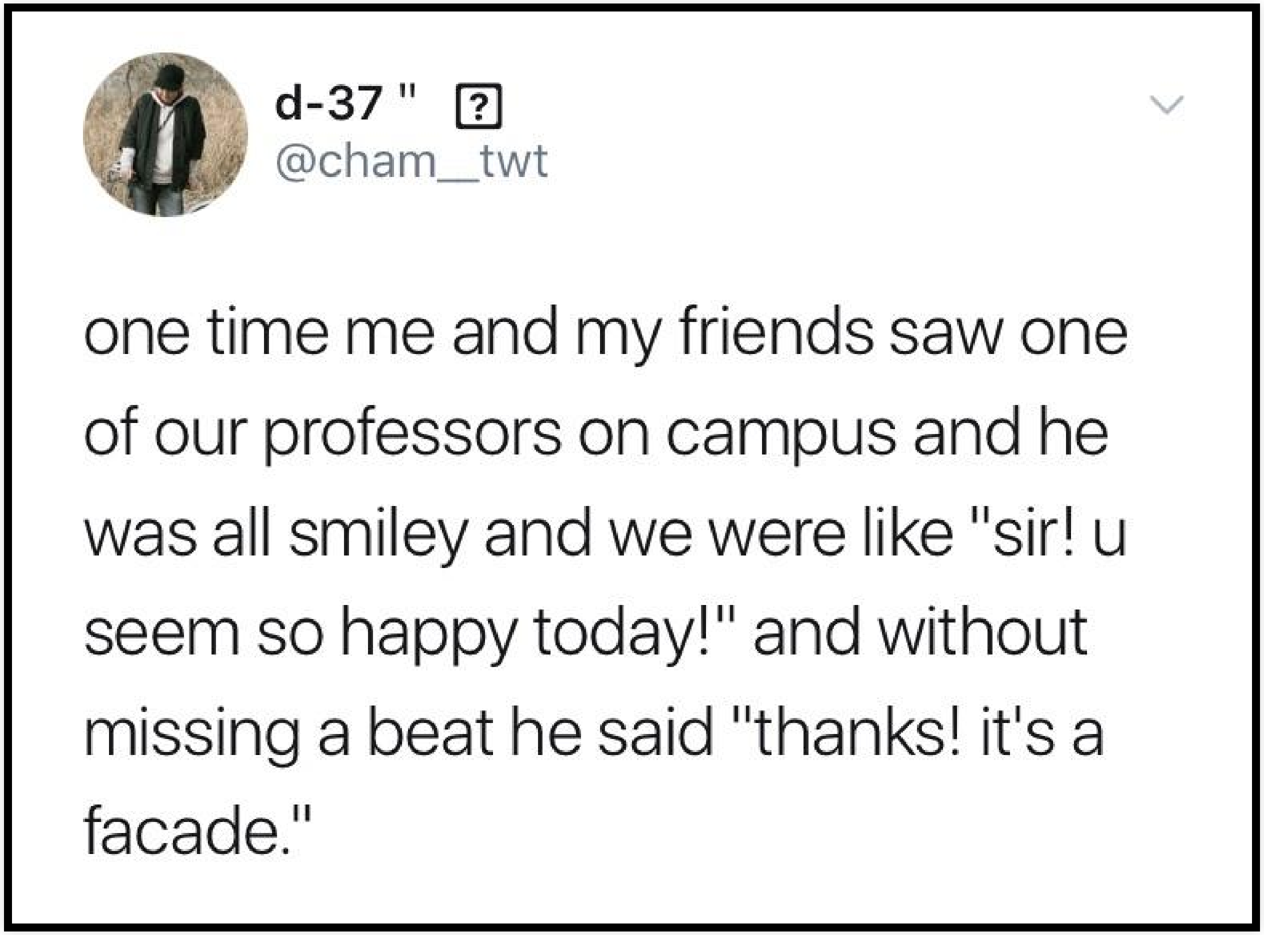
- Scheduling algorithms *can* use similar data structures.
- Performance is measured and can be compared on one or more attributes related to time.
- *All* scheduling algorithms are flawed.
  - No **best** scheduling algorithm exists.
  - *A* best for *a* workload exists.

Next week

Let’s take a look at the schedule.



Public Domain



d-37 " ⍰
@cham__twt

one time me and my friends saw one of our professors on campus and he was all smiley and we were like "sir! u seem so happy today!" and without missing a beat he said "thanks! it's a facade."