Space Informatics

Week 12: Safety and Reliability of Space System

Computer Science and Communications, University of Luxembourg

3 December 2019

1. NASA's VIPER Moon rover: requirements and design specifications
2. Case study: Cubesat constellation

VIPER Moon rover

https://www.universetoday.com/143036/why-is-the-moons-south-pole-so-important-its-all-about-water/

https://www.nasa.gov/feature/new-viper-lunar-rover-to-map-water-ice-on-the-moon

https://phys.org/news/2019-10-viper-lunar-rover-ice-moon.html

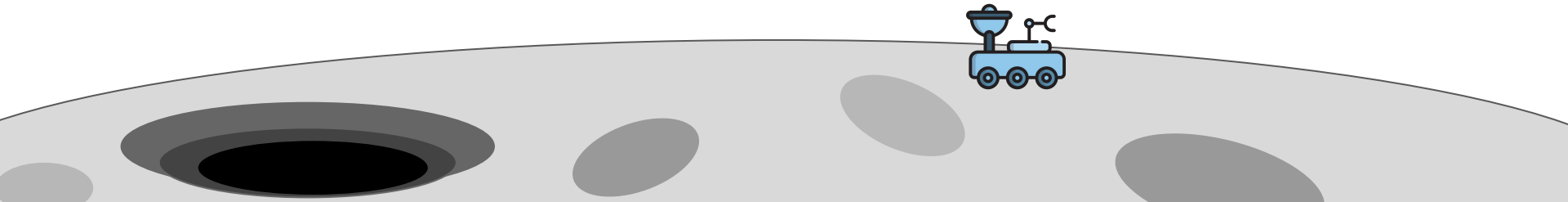
https://spacenews.com/nasa-confirms-plans-to-send-prospecting-rover-to-the-moon/

https://www.skyandtelescope.com/astronomy-news/nasa-announces-viper-lunar-rover/

https://www.theguardian.com/science/2019/dec/03/indias-crashed-vikram-moon-lander-spotted-on-lunar-surface

ISO Standards: https://www.iso.org/standards.html

VIPER Moon rover



Requirements specification

- provide requirements of a mission, a system, a component

  → asks “*what*”

Requirements specification

- provide requirements of a mission, a system, a component

  → asks "*what*"

```
Requirements spec. for VIPER

preserve the payload → land gently
avoid craters
charge batteries with sunlight
sample soil environments → carry a drill
analyse data on board
send analysed data back to Earth
```

Design specification

- provide details and answers to specifications of a mission, a system, a component
- asks “*how*”

e.g. fault trees

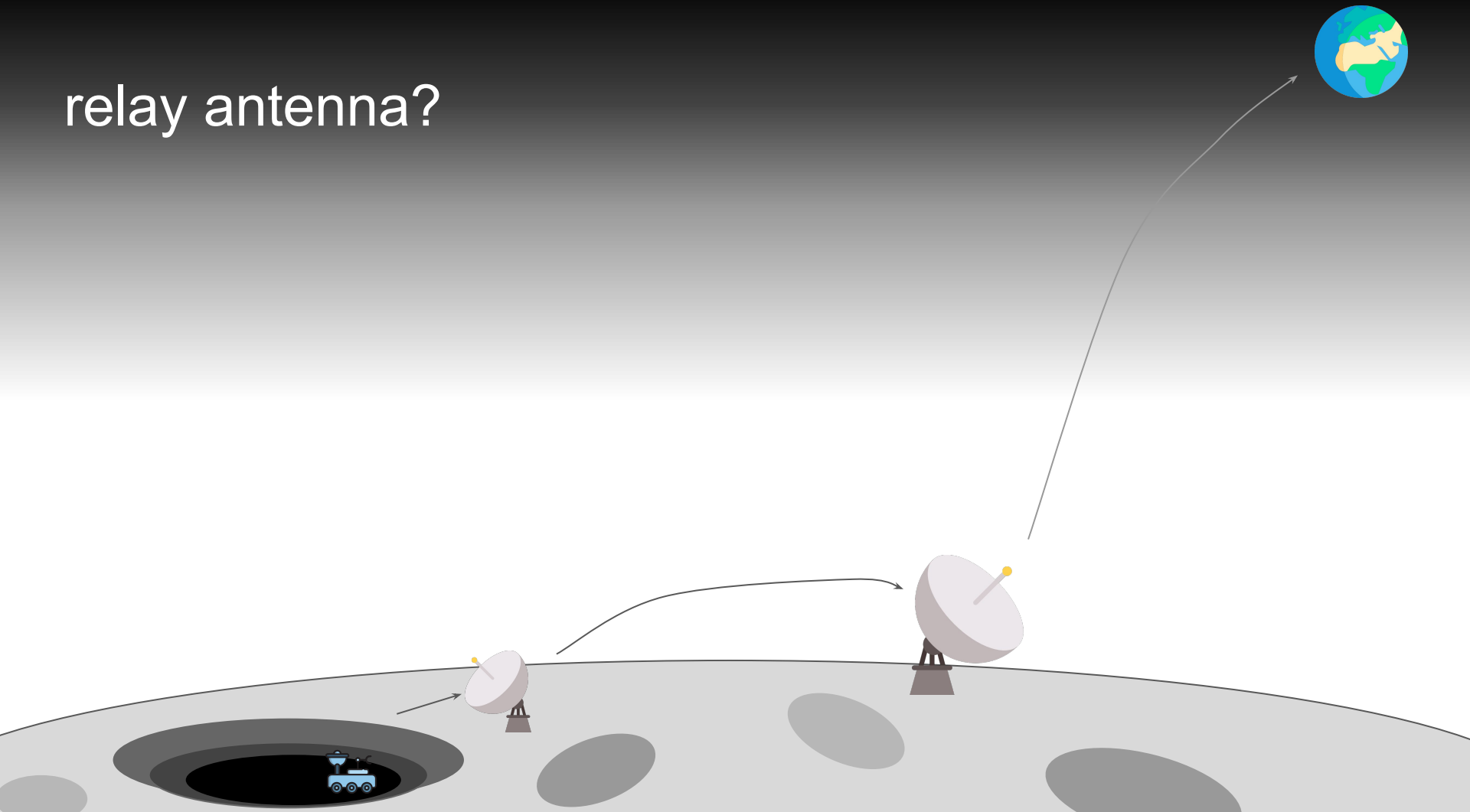
Design specification

Requirement: analyze data on board	Embedded computers - Constellation Single Board Computer (cSBC) - 28V-70V power supply - processor 152 DMIPS with up to 32MB of Radiation Hardened SRAM and 4MB EEPROM - global positioning system (GPS) - Independant batteries from the rover's controller system

Design specification

Requirement: send analysed data back to Earth	- Wifi emitter with large bandwidth 2,53 Gbps - relay antenna - wired communication channels?

relay antenna?



wired communication channel?

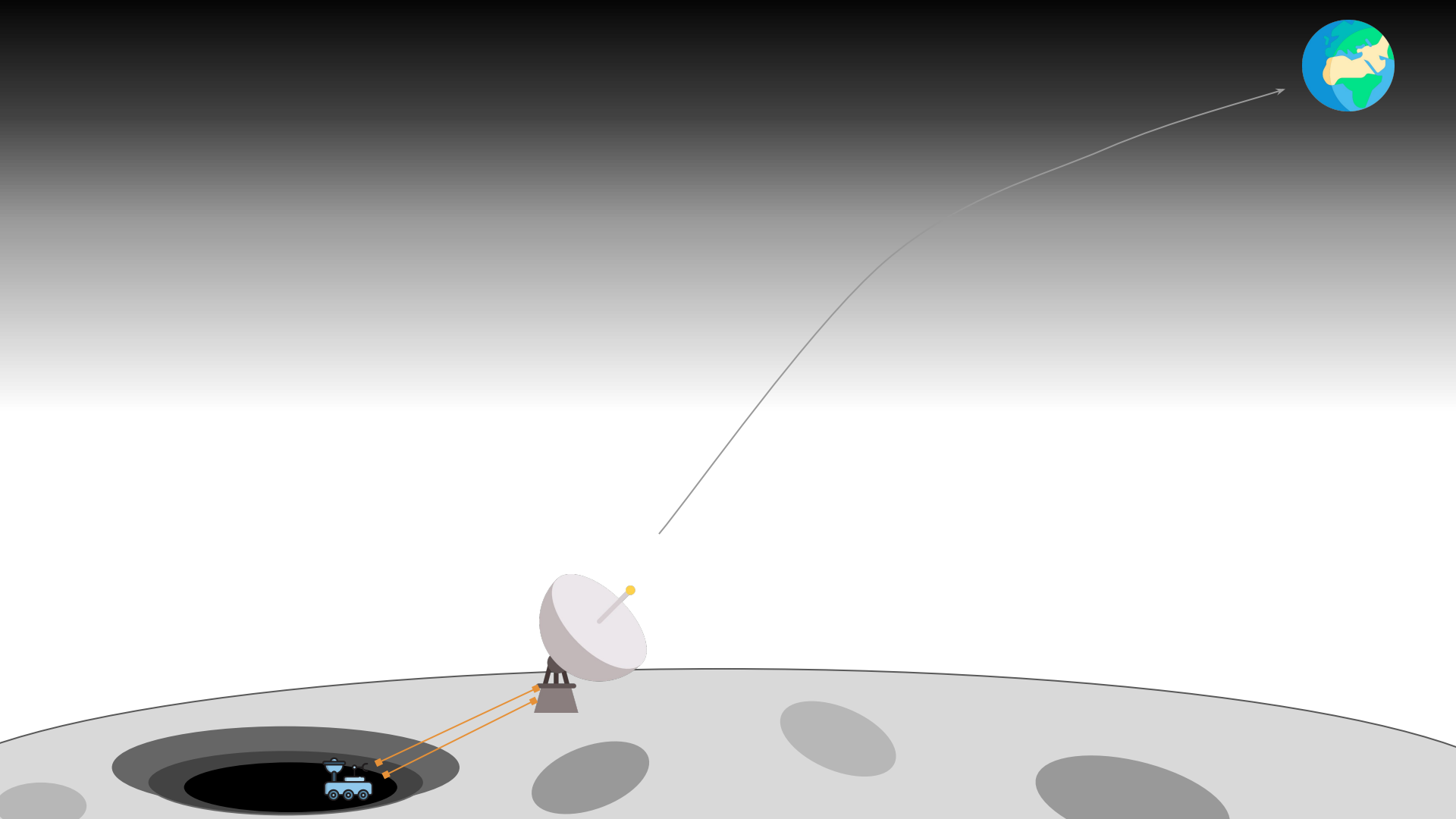
as promising spots to find water
could provide oxygen to breathe
s and rockets. The Moon's tilt
water ice from comet and meteor
he lunar soil, can collect without
ed a rocket into a large crater near
nce of water ice. Data from
that the Moon has reservoirs of
ons. Now, we need to understand
potentially accessible resources to

the new and unique environment
can harvest that water," said
VIPER will tell us which locations
below the surface to go to get

Pole, the rover will collect data on
light and temperature – those in
ct sunlight. By collecting data on
h, NASA can map out where else

e the Neutron Spectrometer
below the surface for further
a drill, The Regolith and Ice Drill for
with Honeybee Robotics, to dig

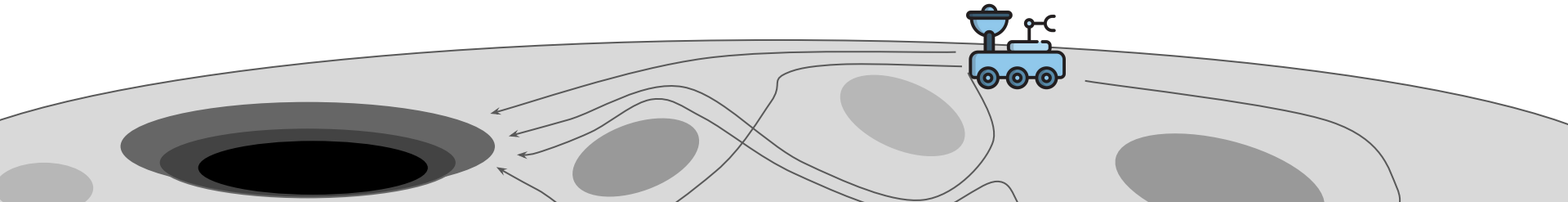
Pictured here is a VIPER mobility testbed, an engineering model created to evaluate the rover's mobility system. The testbed includes mobility units, computing and motor controllers. Testing involves evaluating performance of the rover as it drives over various slopes, textures and soils that simulate the lunar environment.
Credits: NASA/Johnson Space Center



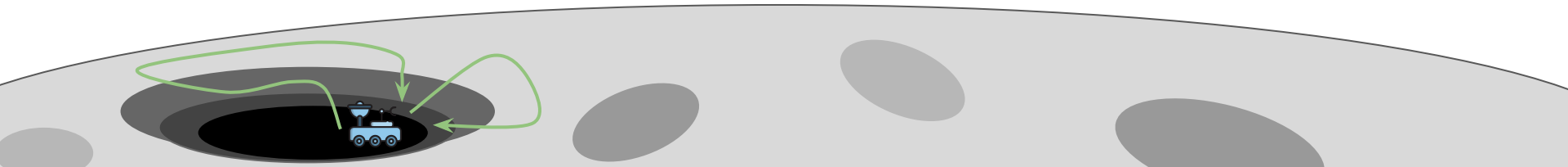
Design specification

Requirement: avoid craters and charge batteries with sunlight	- mathematical model for the trajectories? - optimization procedures for the charging/discharging cycles?

VIPER Moon rover trajectories



VIPER Moon rover trajectories



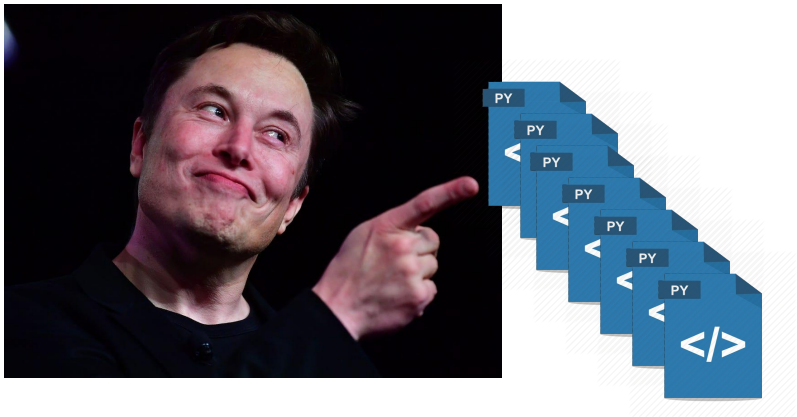
Design specification

mathematical model for the rover's trajectories
→ NASA

optimization procedures for the charging/discharging cycles?
→ batteries and solar panel manufacturer

e.g. "The spacecraft lander and launch vehicle that will deliver VIPER to the surface of the Moon, will be provided through NASA's Commercial Lunar Payload Services (CLPS) contract"

How reliable is a complex software, written by multiple programmers

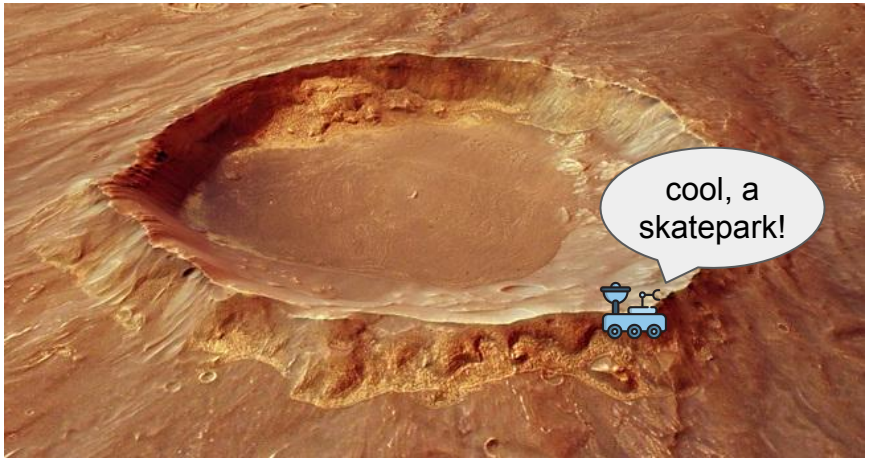



```
def split_prefix(leaf, start_pos):
    line, column = start_pos
    start = 0
    int16 a = 12
    value = spacing = ''
    bom = False
    int64 b = 0
    while start != len(leaf.prefix):
        match =_regex.match(leaf.prefix, start)
        spacing = match.group(1)
        value = match.group(2)
        if not value:
            break
        type_ = _types[value[0]]
        yield PrefixPart(
            leaf, type_, value, spacing,
            start_pos=(line, column + start - int(bom) + len(spacing))
        )
        if type_ == 'bom':
            bom = True

        a = b
        start = match.end(0)
        if value.endswith('\n'):
            line += 1
            column = -start

    if value:
        spacing = ''
    yield PrefixPart(
        leaf, 'spacing', spacing,
```

especially for critical systems

What is “critical”?

Robotic vacuum cleaner ≠ rover



software testing

- perform a set of tests to ensure the stability, absence of bugs in a system
- often needs a large test set in order to cover all possible behaviours
- machine learning can help

software testing vs. formal verification

- Testing is insufficient to prove the absence of bugs!
- bug detection is difficult for complex systems as there is usually **an infinite number of possible behaviours to test**

formal verification

- prove or disprove the correctness of a program/algorithm/system **before** the testing phase

For simple programs, *static code analysis* ✔️

For more complex mathematical reasoning, *proof assistants* ✔️

For complex critical embedded systems, *model-checking*

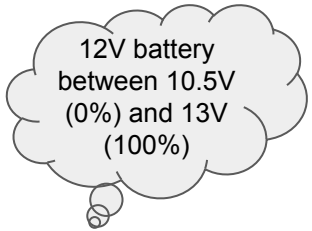
Batteries and solar panels

a battery, powered by two solar panels.

additional requirements specifications:

- charging and discharging are time dependant
    → it can be dependant on which component is running

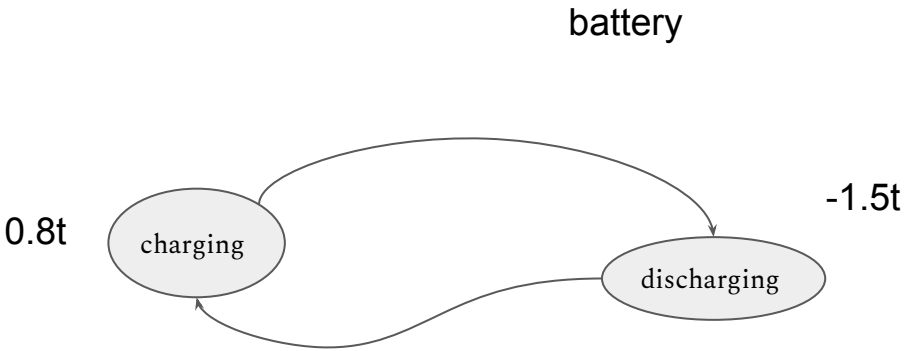
Batteries and solar panels



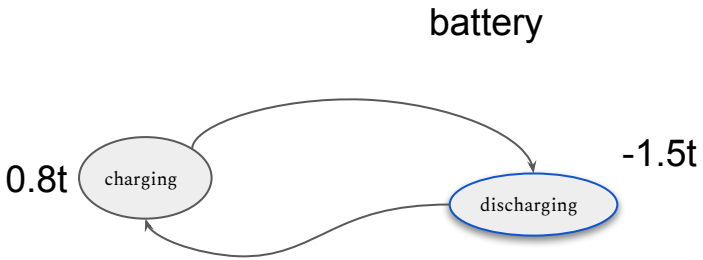
design specifications:
- charging according to a function f(t)=0.8t
    → South Pole of the Moon, so brightness is not optimal

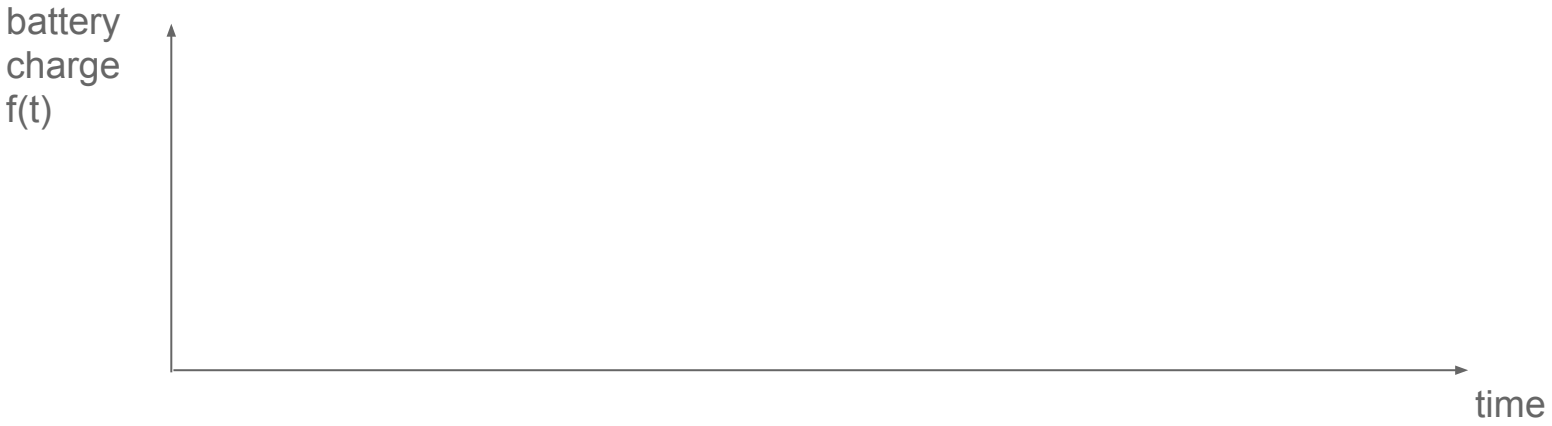
- discharging according to a function f(t)=-1.5t
    → the Regolith and Ice Drill for Exploring New Terrain, or TRIDENT
    → the Mass Spectrometer Observing Lunar Operations, or MSolo
    → the Near InfraRed Volatiles Spectrometer System, known as NIRVSS

modeling with automata

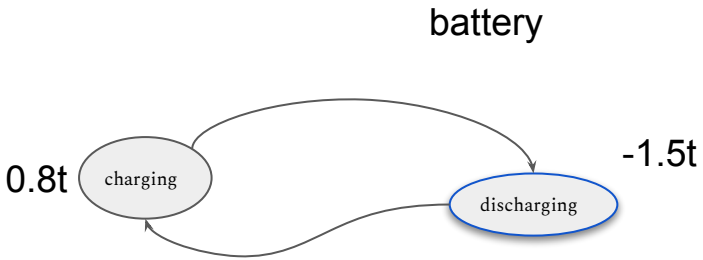


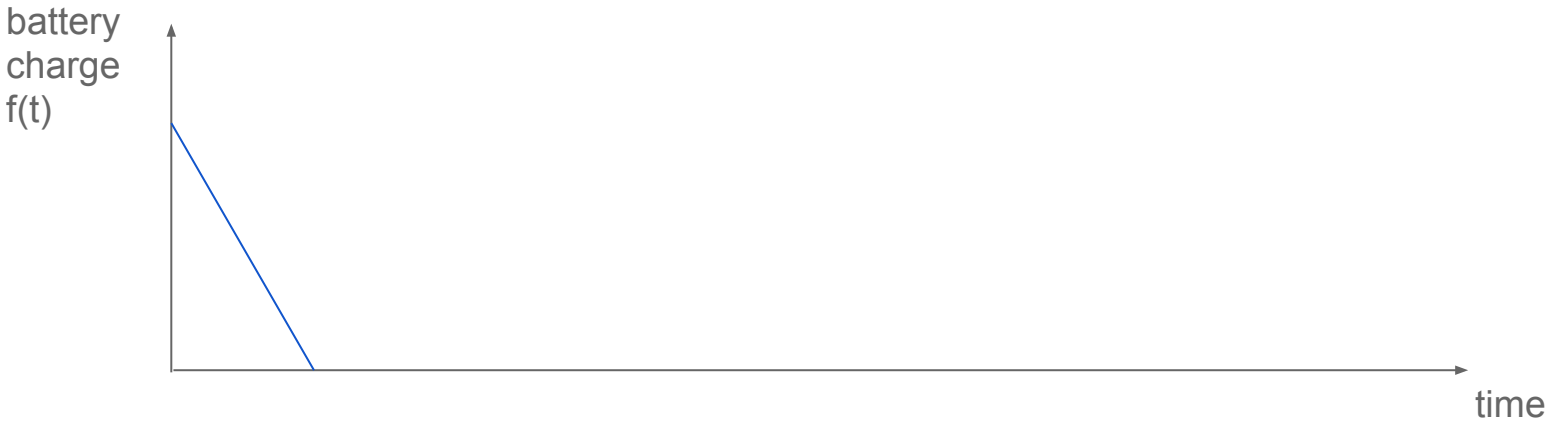
modeling with automata



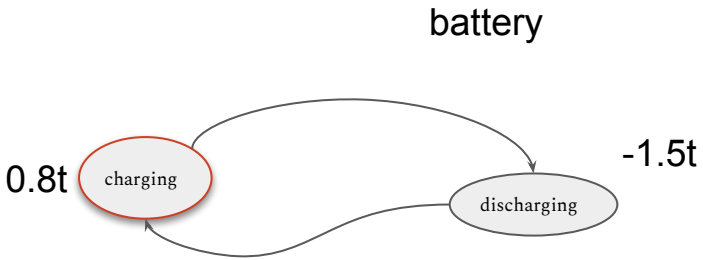


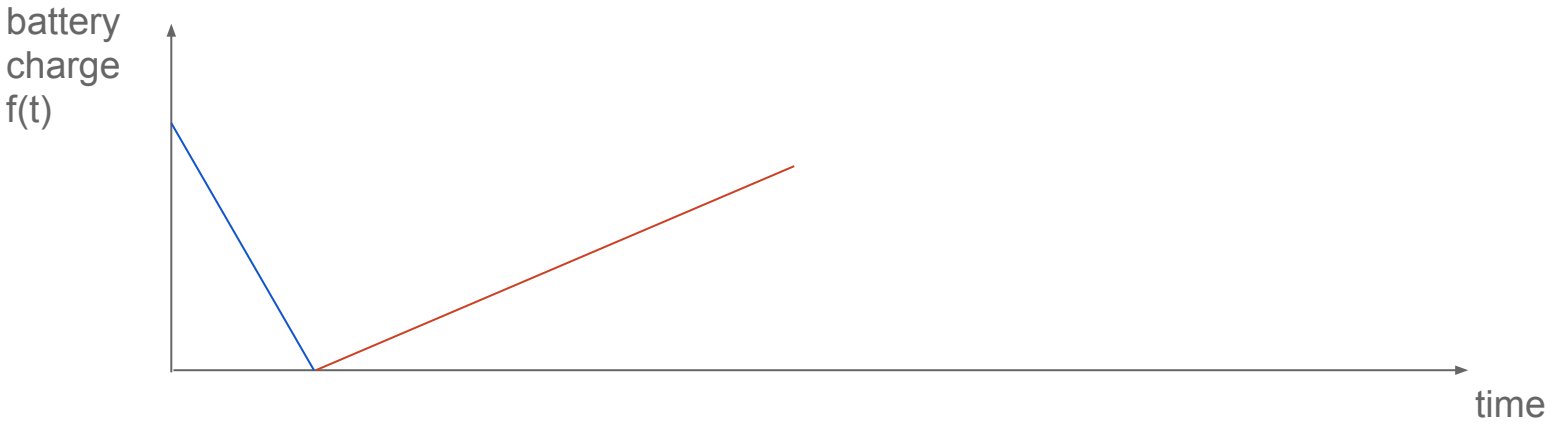
modeling with automata



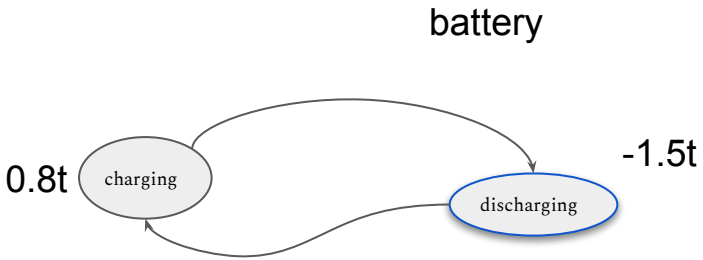


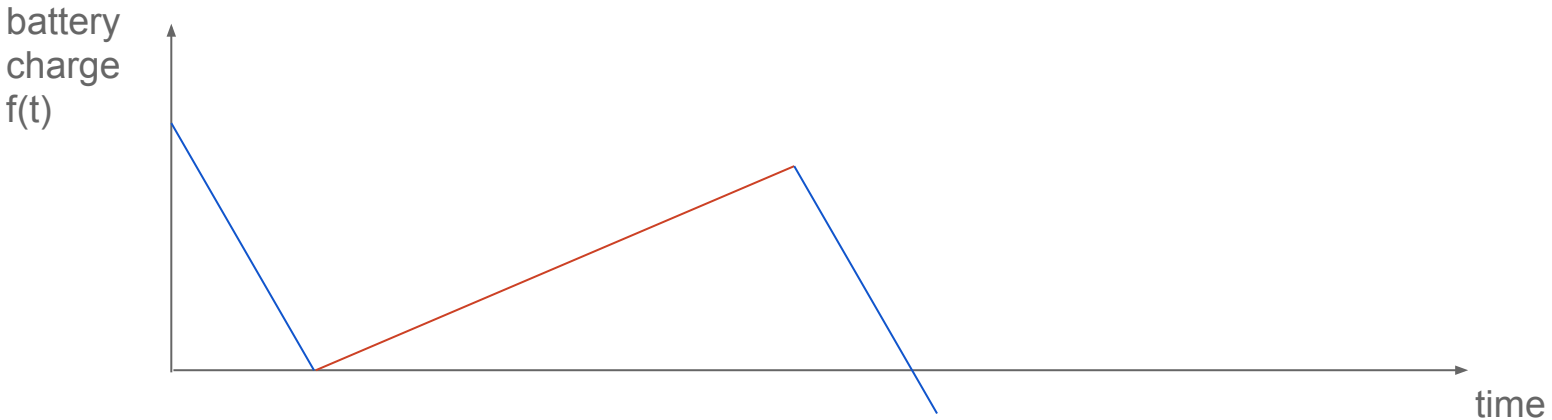
modeling with automata



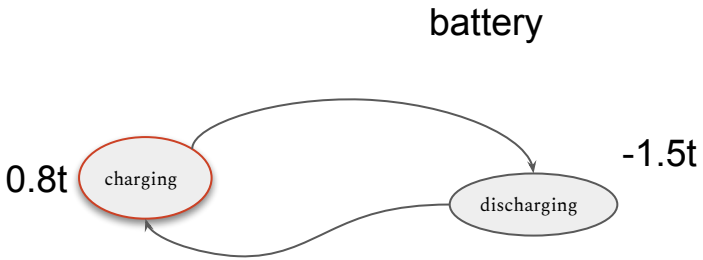


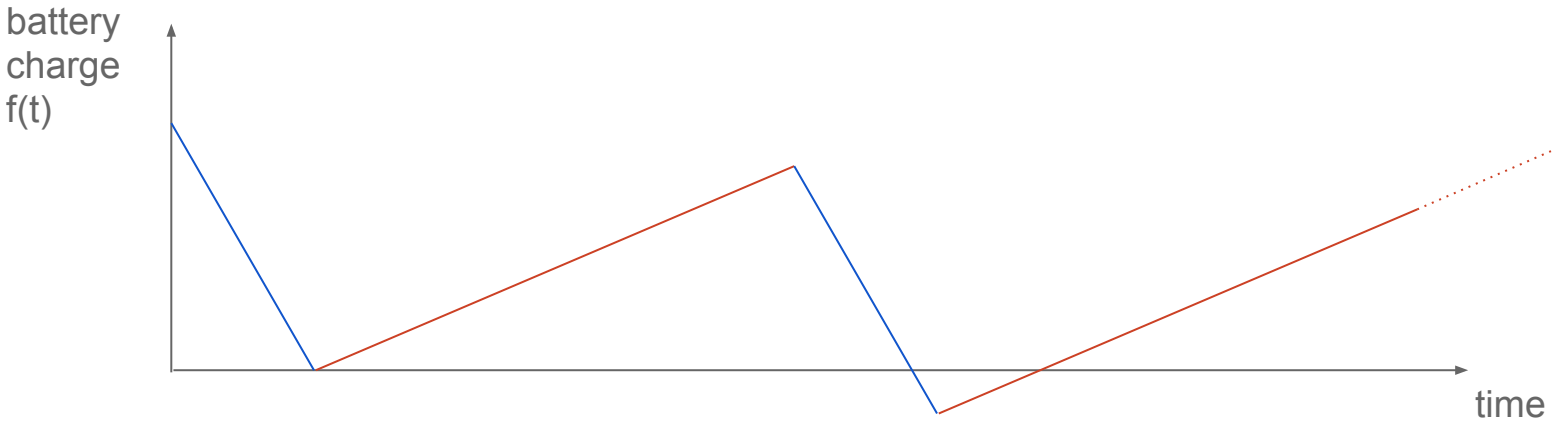
modeling with automata



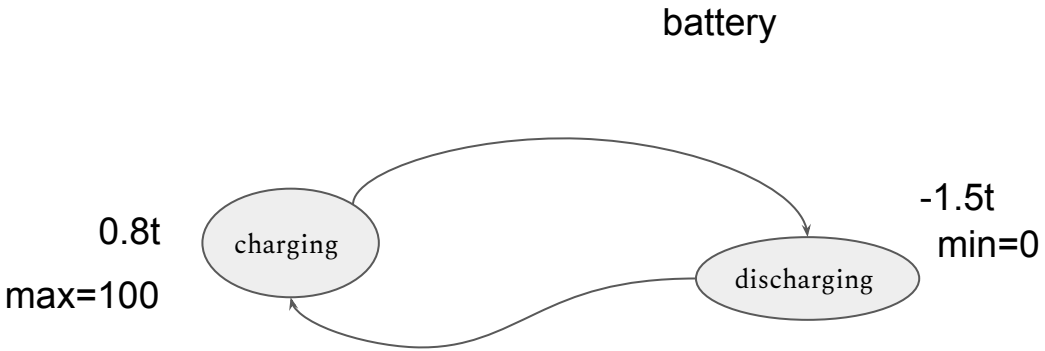


modeling with automata

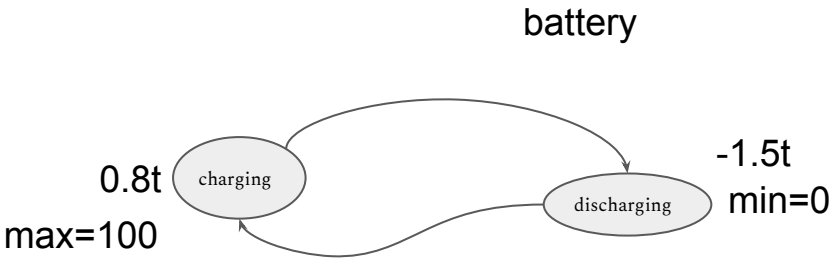


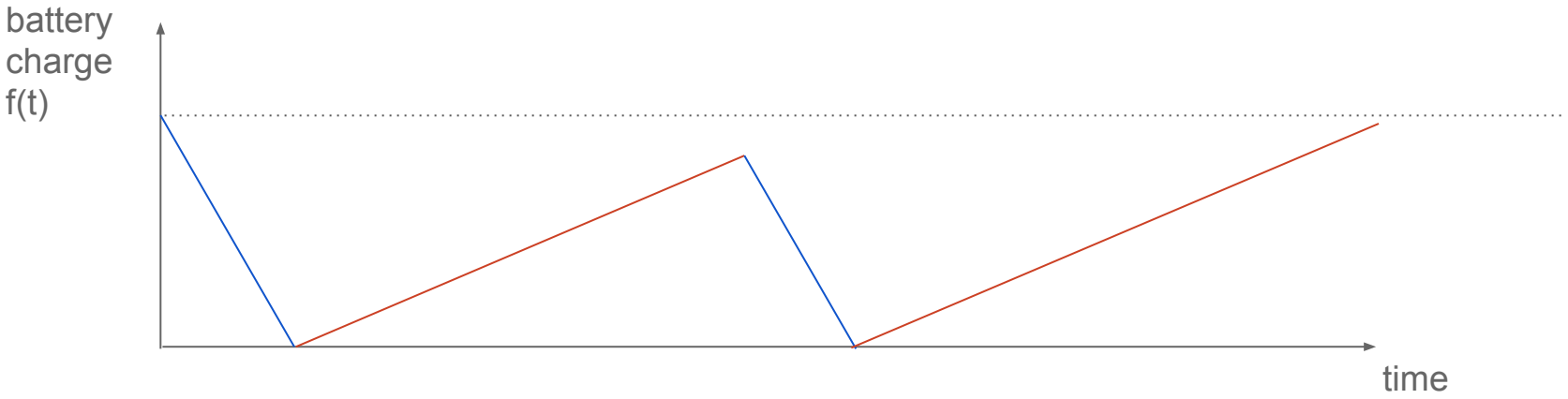


modeling with automata



modeling with automata





Batteries and solar panels

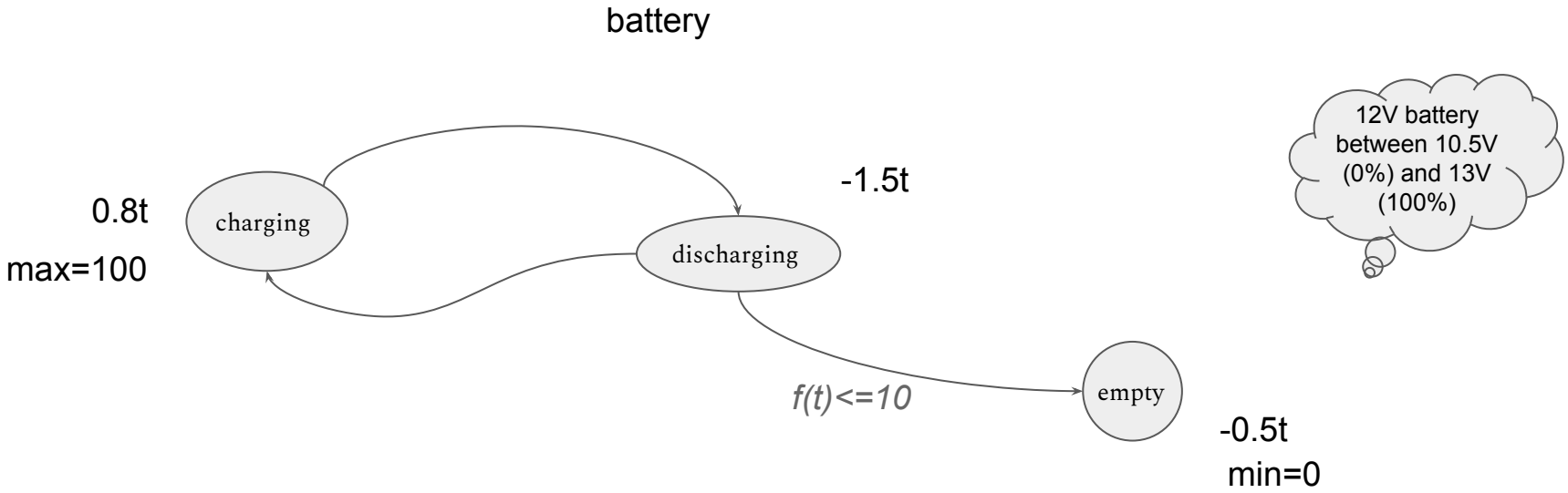
a battery, powered by two solar panels.

additional requirements specifications:

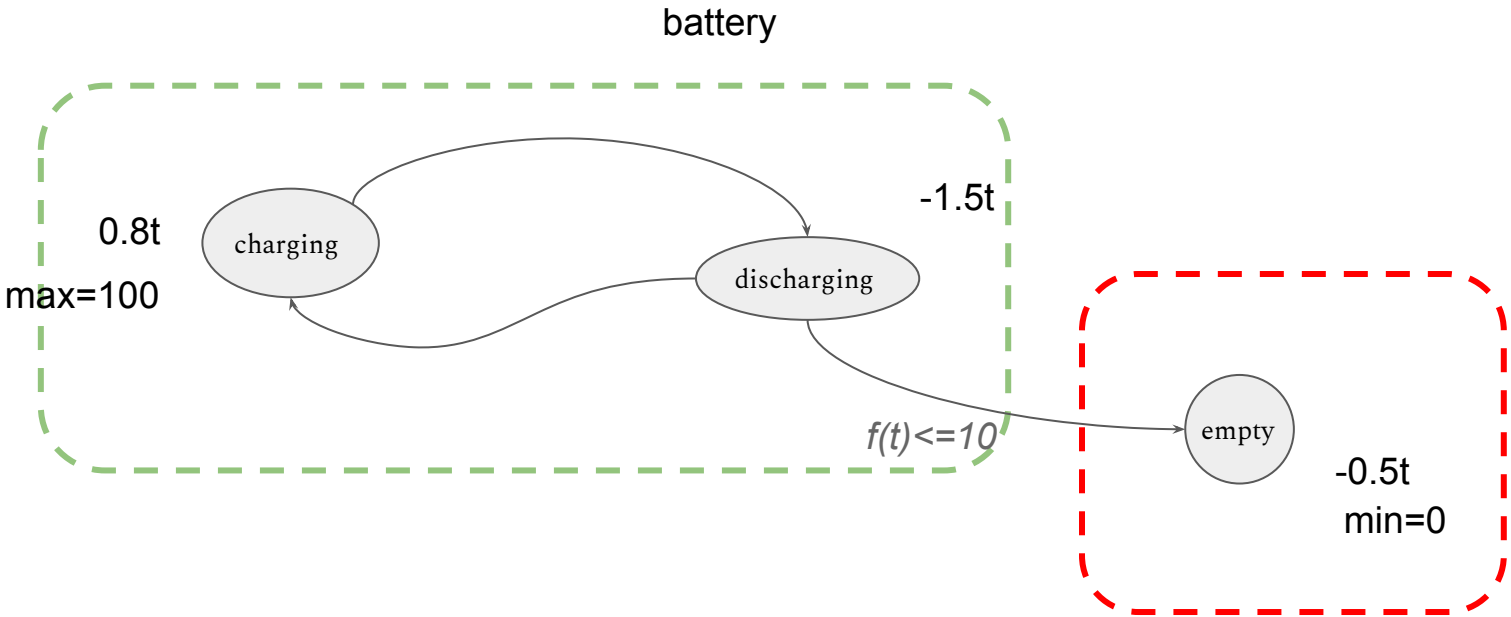
- charging and discharging are time dependant
    → it can be dependant on which component is running

- power cannot decrease below a threshold
    → when power is running low, sufficient power is needed to return to a zone with sunlight.

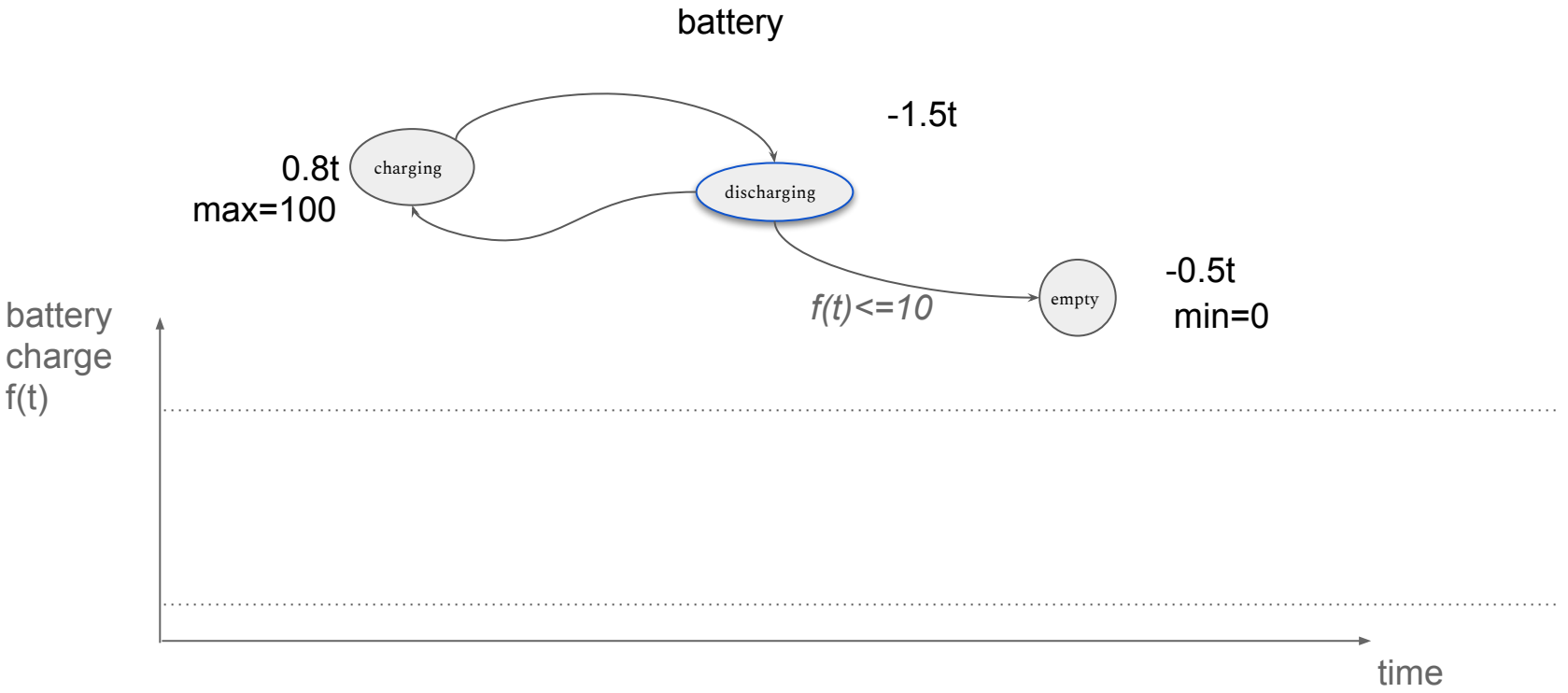
modeling with automata



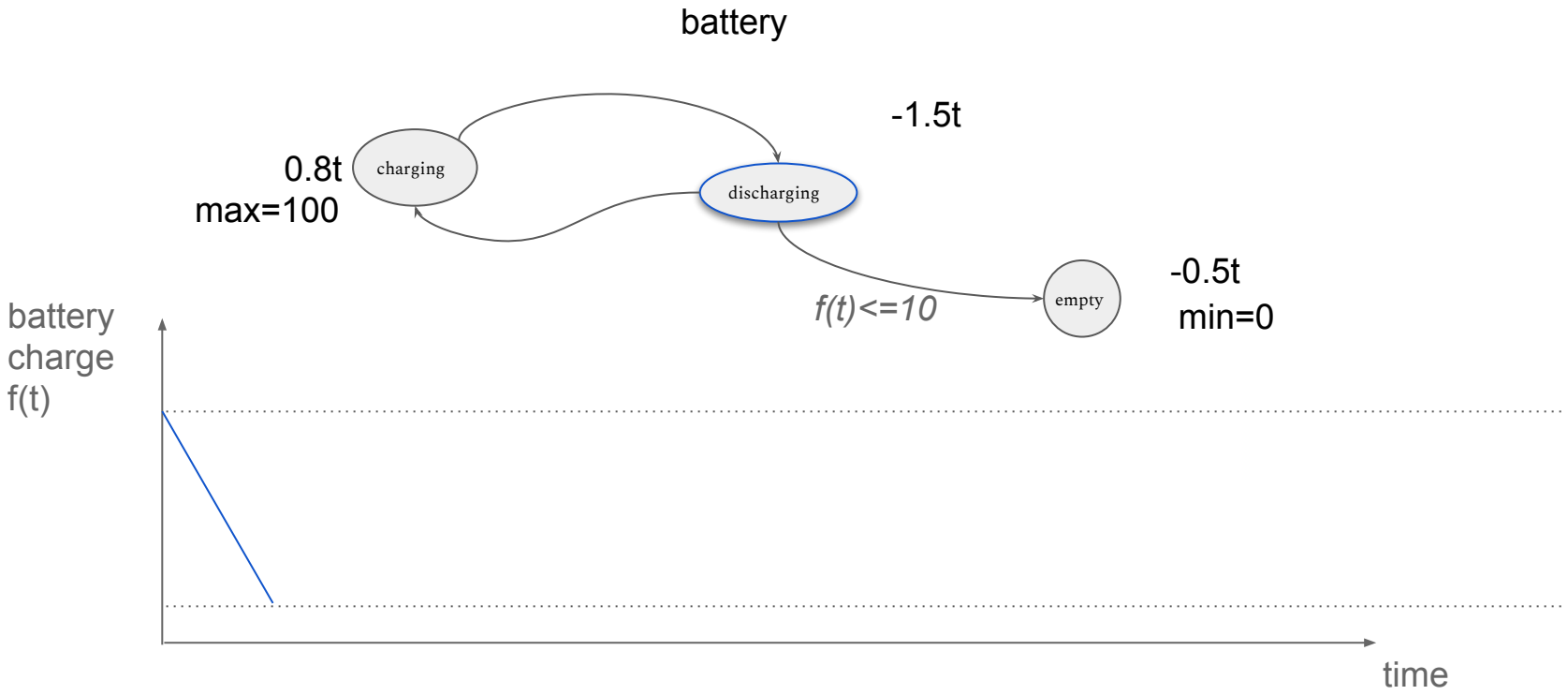
modeling with automata



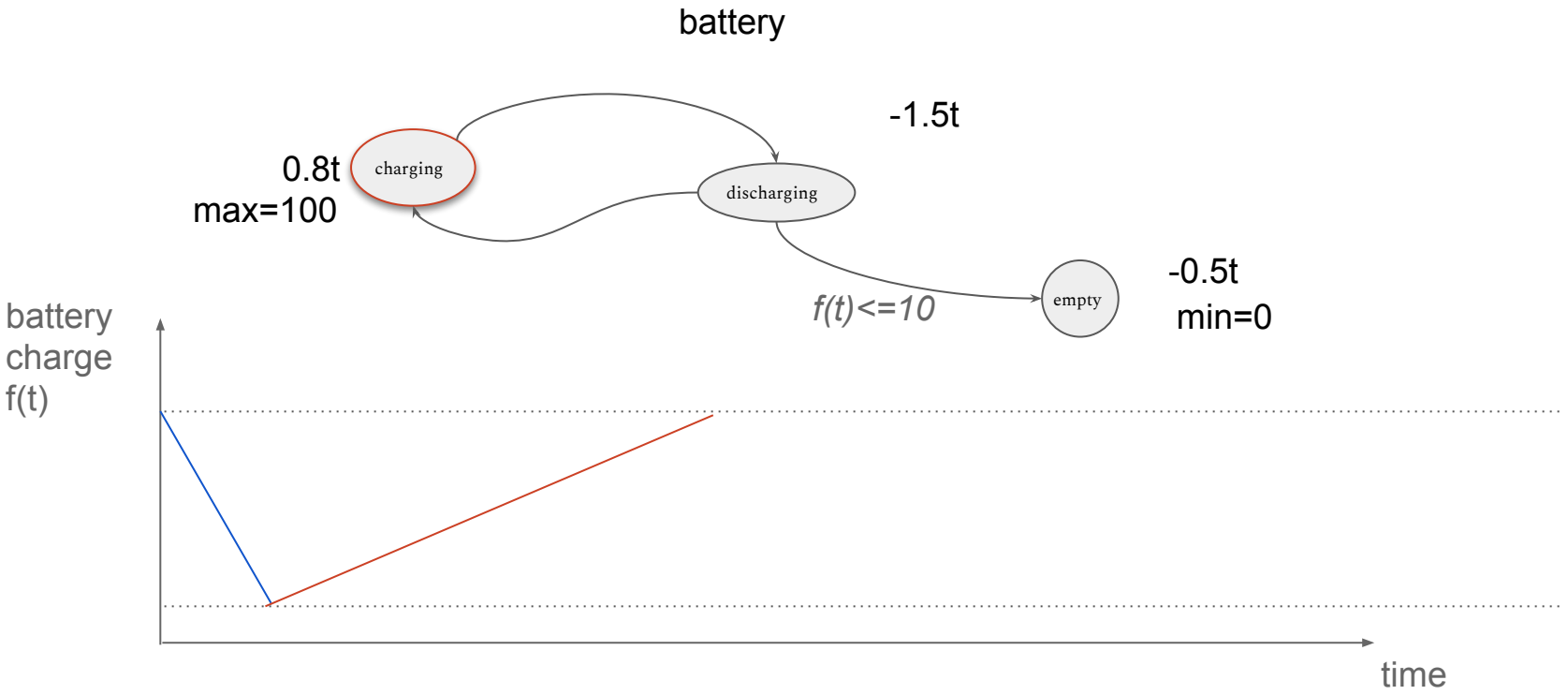
modeling with automata



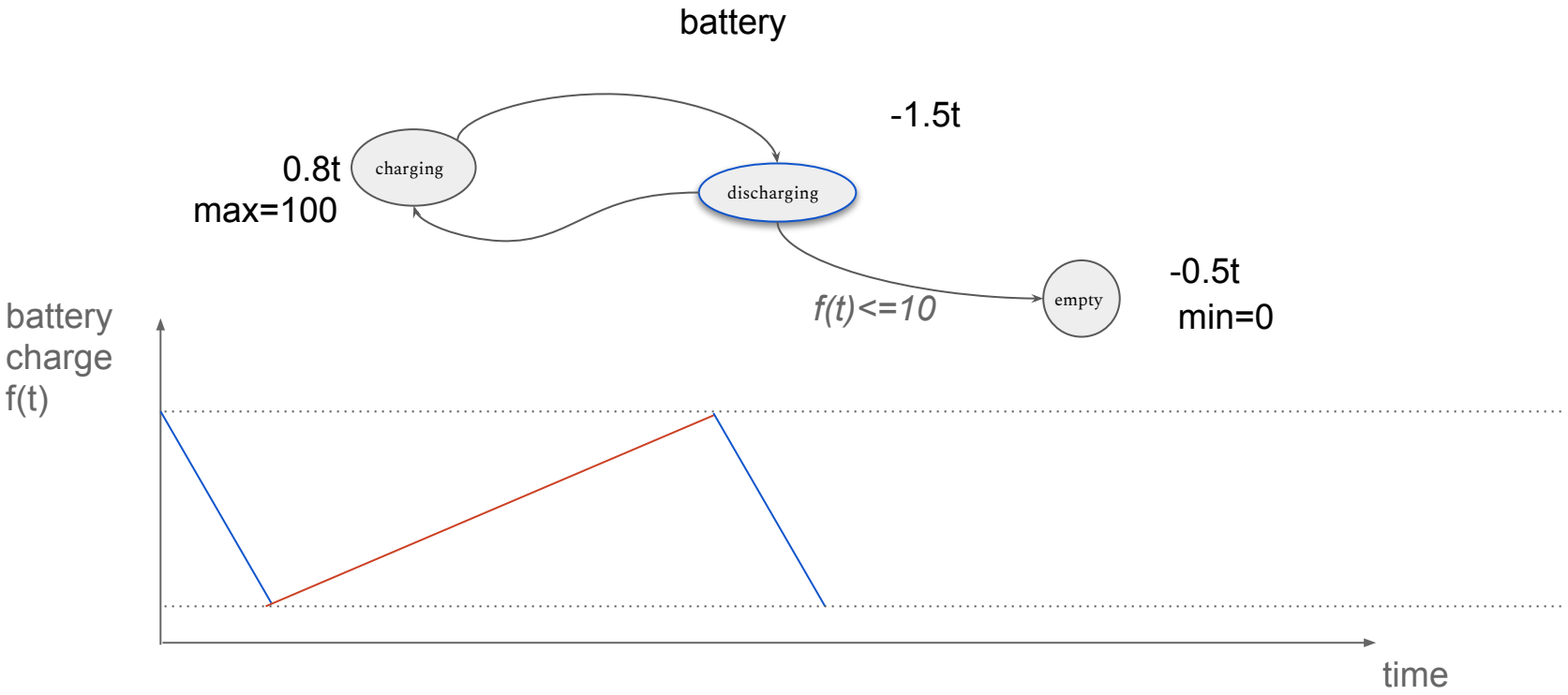
modeling with automata



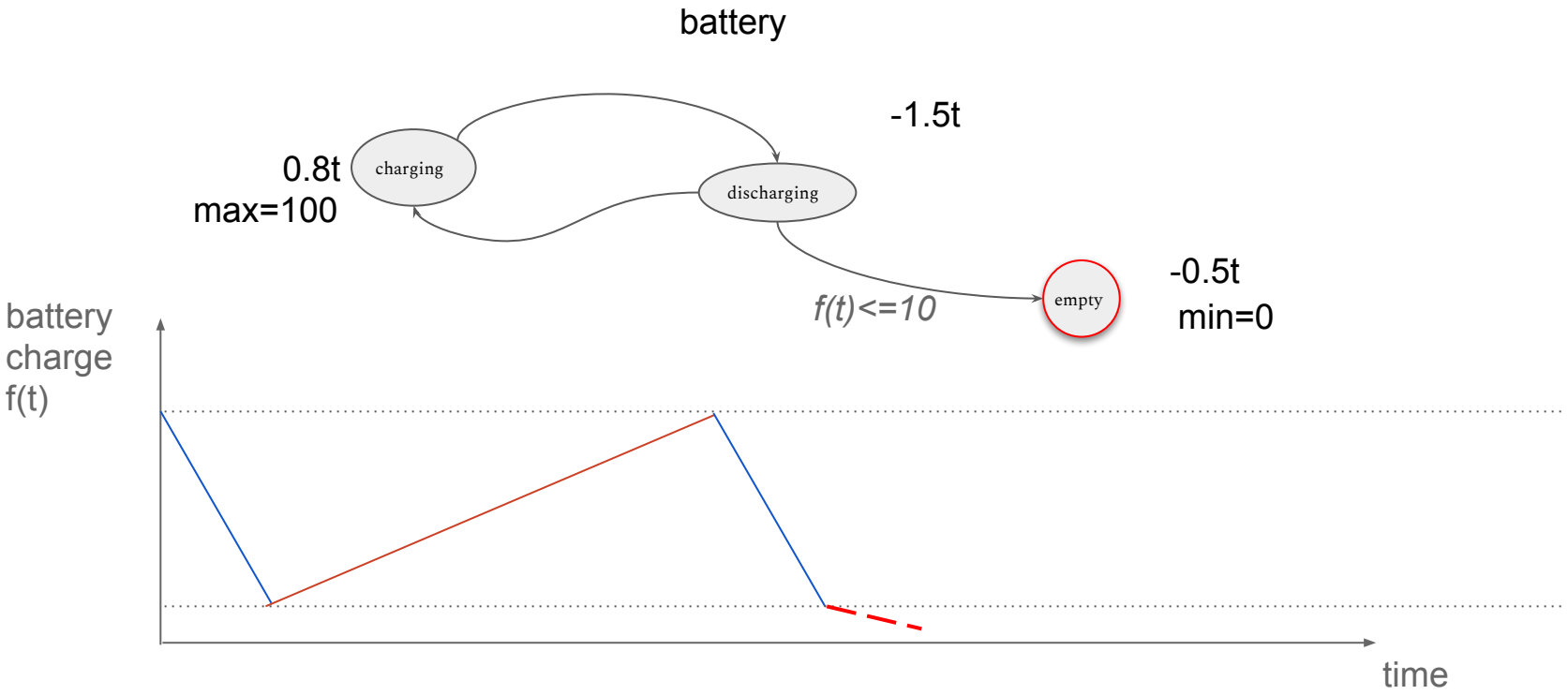
modeling with automata



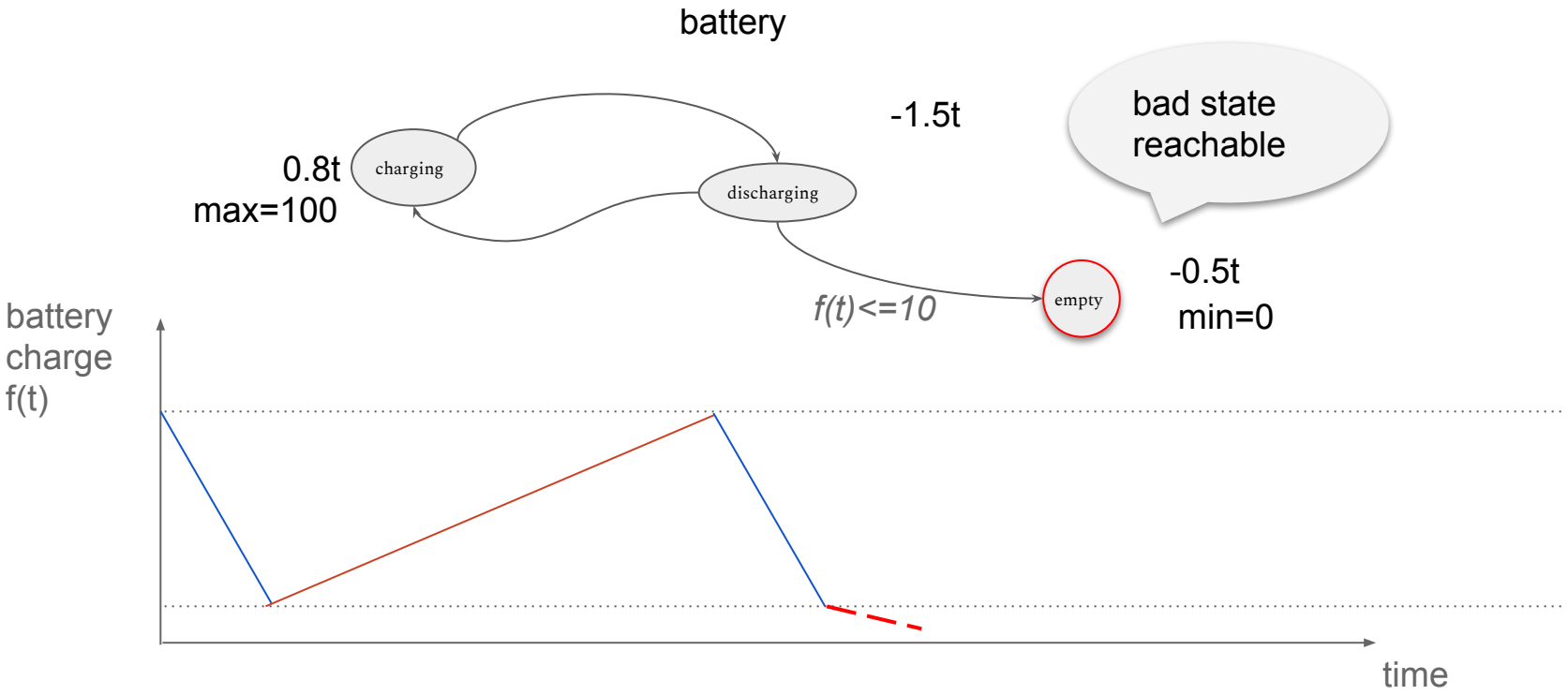
modeling with automata



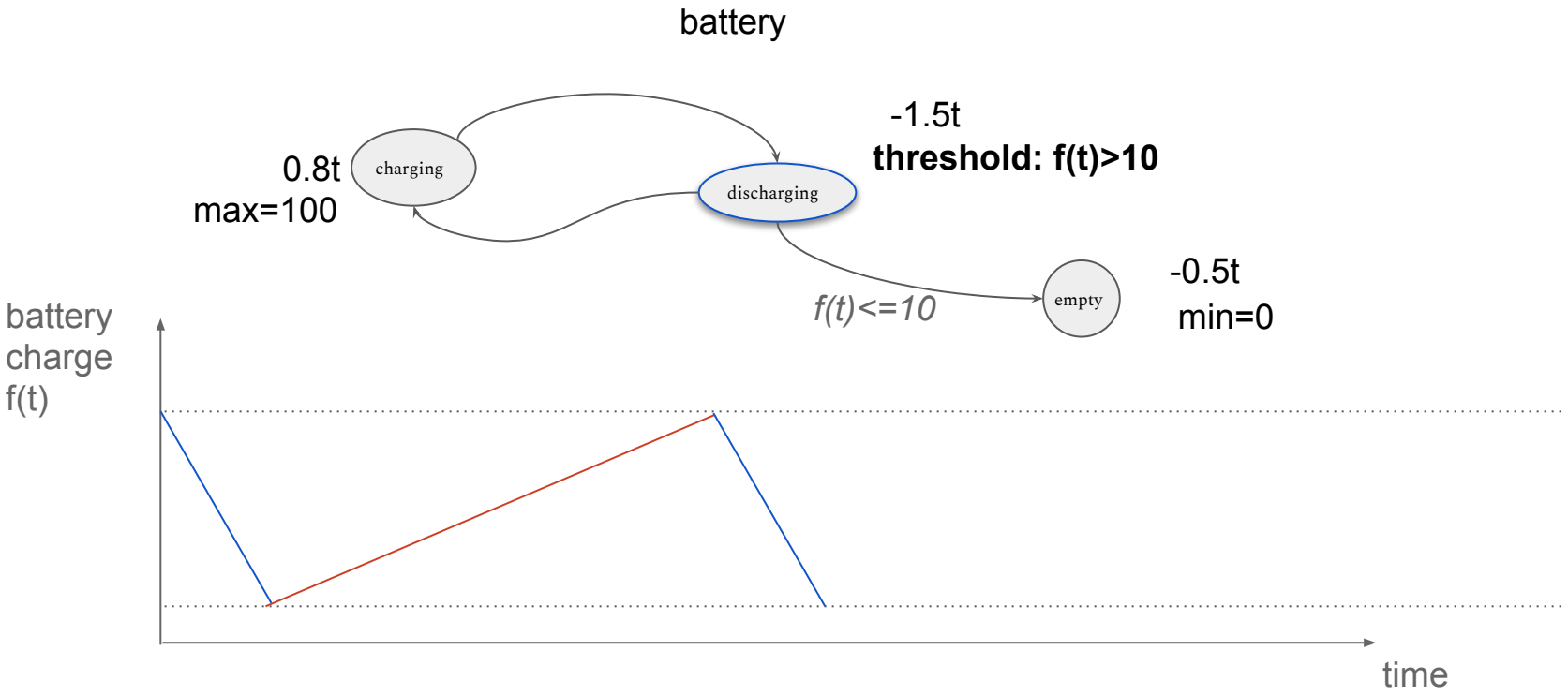
modeling with automata



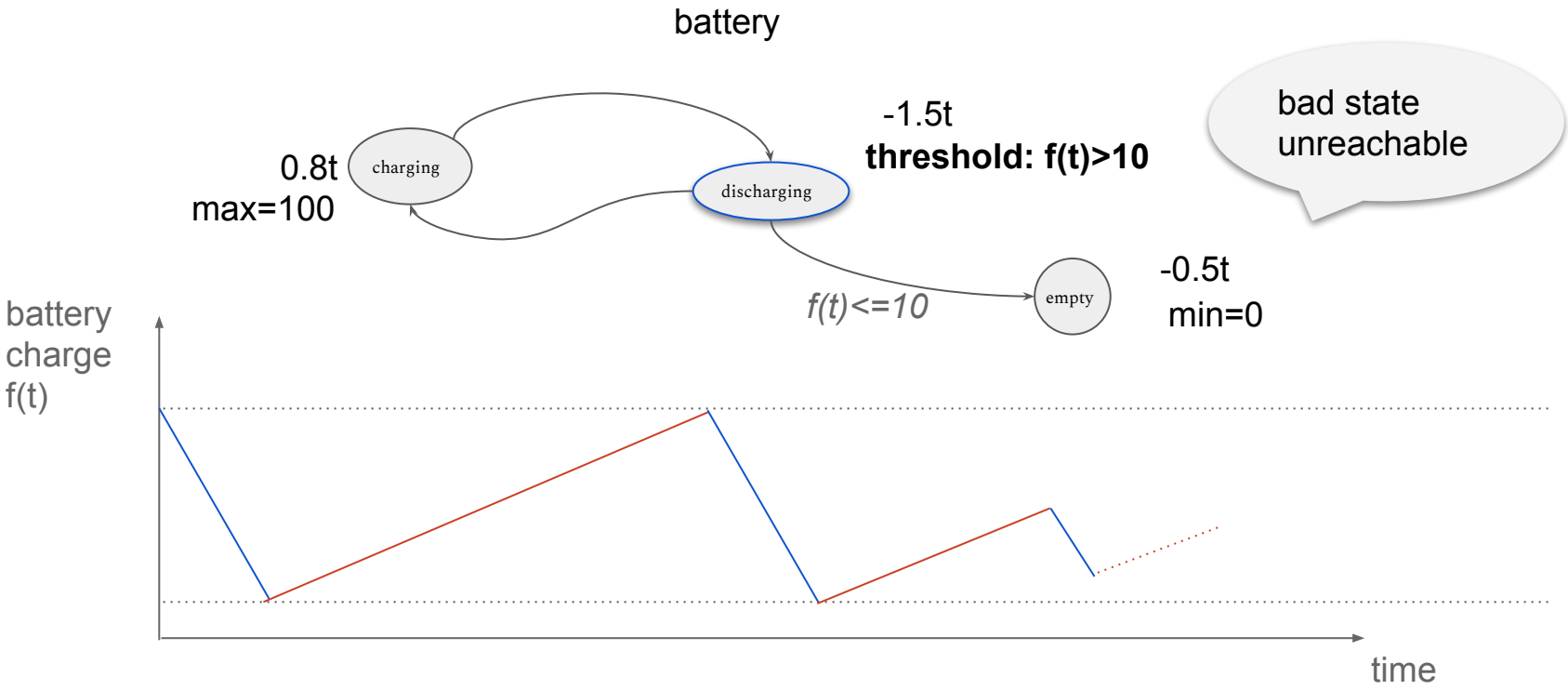
modeling with automata



modeling with automata

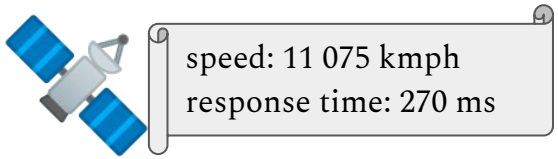


modeling with automata



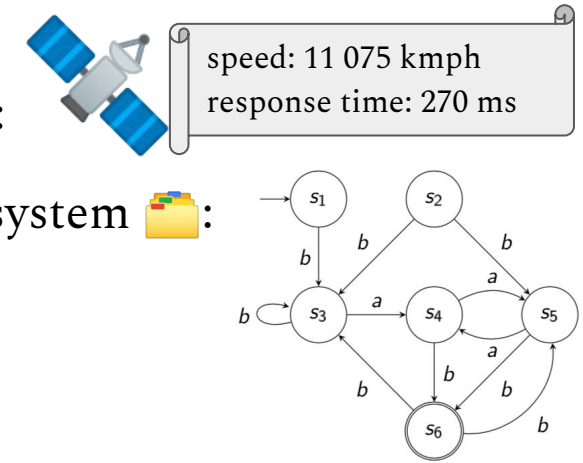
model-checking

- a system or a subcomponent of a system:



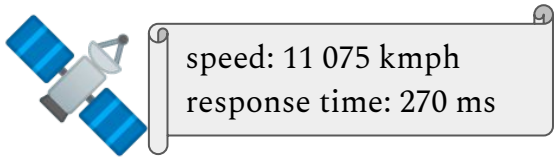
model-checking

- a system or a subcomponent of a system:
- an abstract/mathematical model of this system 🗂️:

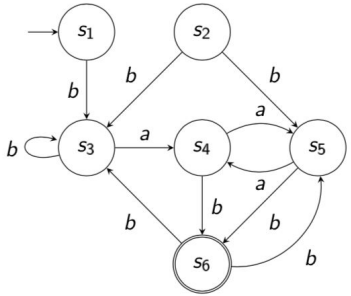


model-checking

- a system or a subcomponent of a system:



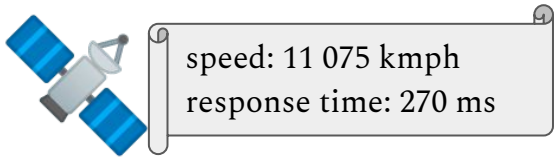
- an abstract/mathematical model of this system 🗂️:



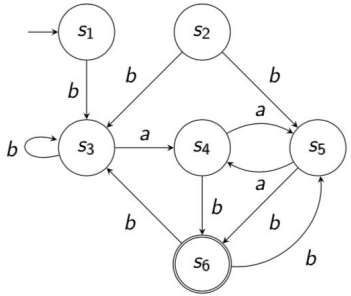
- a property 🅿️ e.g., *“given the speed and response time, can I eventually lose the communication channel to my satellite”*

model-checking

- a system or a subcomponent of a system:



- an abstract/mathematical model of this system 🗂️:



- a property 🅿️ e.g., *"given the speed and response time, can I eventually lose the communication channel to my satellite"*

  Check whether the model 🗂️ satisfies the property 🅿️: ❌ or ✅ ?

modeling with automata

Purpose?

provide design specifications for developers

developers write *correct* code according to specifications.

model-checking

- proves a model of a system is reliable 👍
- used in industry (COMPASS, Uppaal) 👍

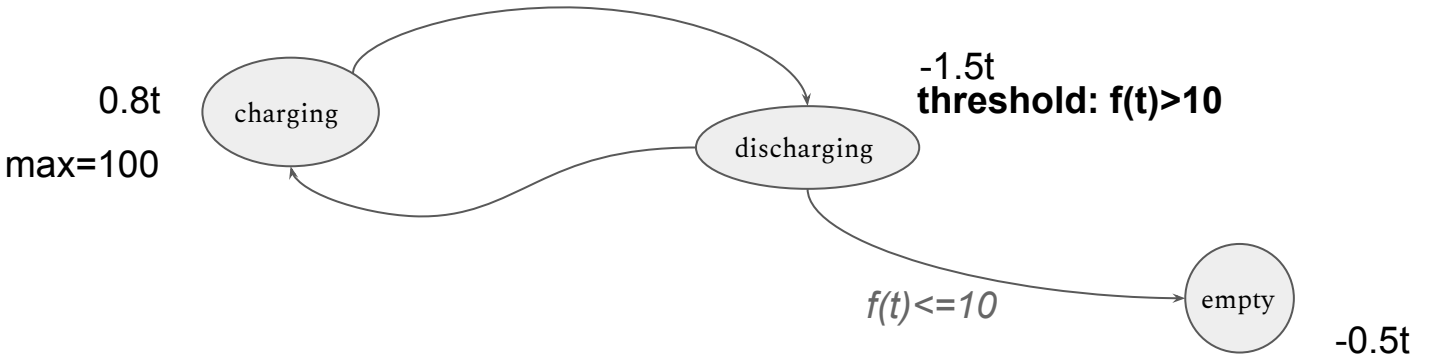
model-checking

- proves a model of a system is reliable 👍
- used in industry (COMPASS, Uppaal) 👍

- very costly (computational complexity, workforce) 👎
- *sometimes not possible* 👎

modeling with automata

the battery model 🗂️:

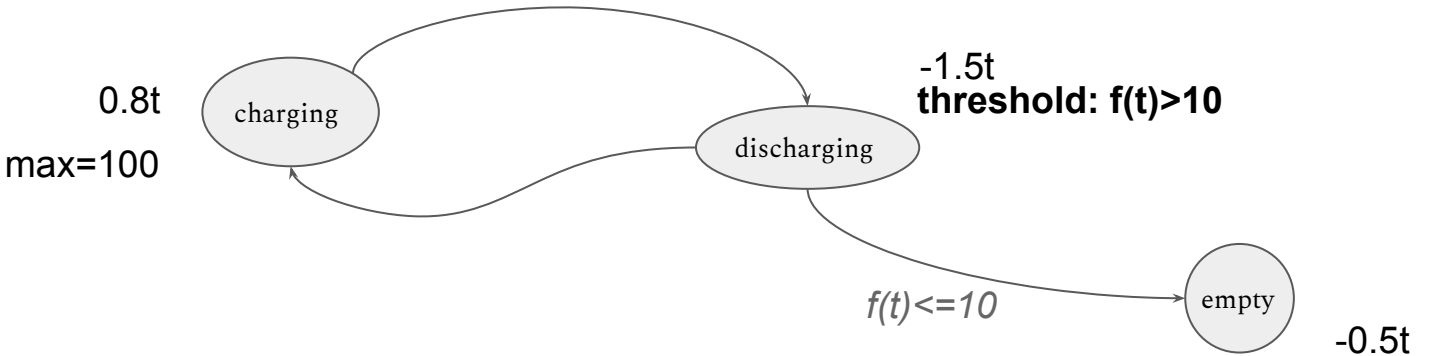


a property 🅿️: is (empty) unreachable?

Check whether the model 🗂️ satisfies the property 🅿️: ❌ or ✔️ ?

modeling with automata

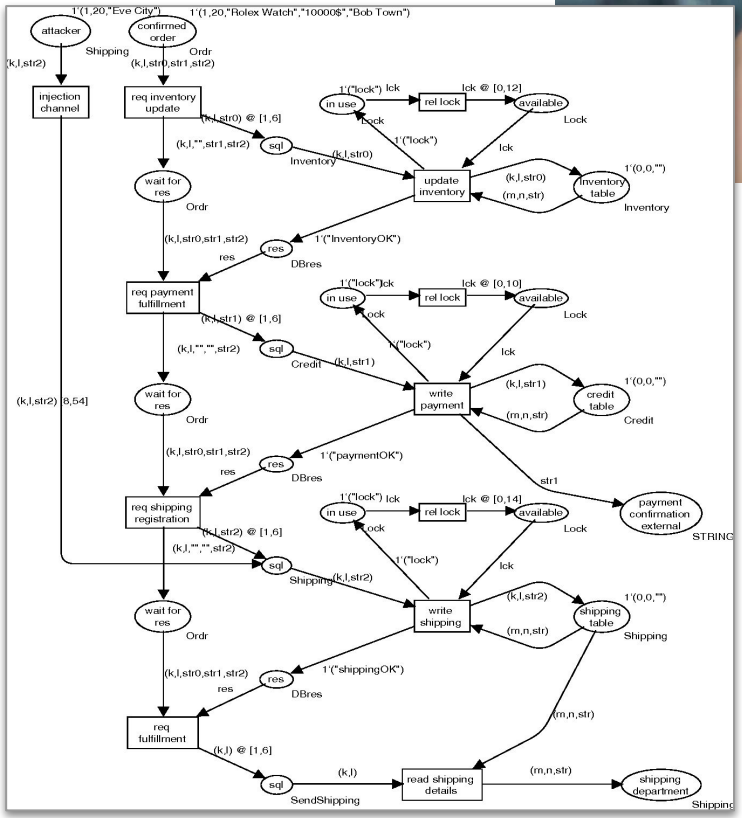
the battery model 🗂️:



a property 🅿️: is (empty) unreachable?

Check whether the model 🗂️ satisfies the property 🅿️: ✅

model-checking is automatic



model-checking

If there is an algorithm that takes as inputs

- the model 🗂️
- the property 🅿️

and outputs ❌ or ✅ to the question *the model* 🗂️ *satisfies the property* 🅿️ ?

then we say 🅿️ is **decidable** for 🗂️

model-checking

If there is an algorithm that takes as inputs

- the model 🗂️
- the property 🅿️

and outputs ❌ or ✅ to the question *the model* 🗂️ *satisfies the property* 🅿️ ?

then we say 🅿️ is **decidable** for 🗂️

PROBLEM: the algorithm does not always terminates.

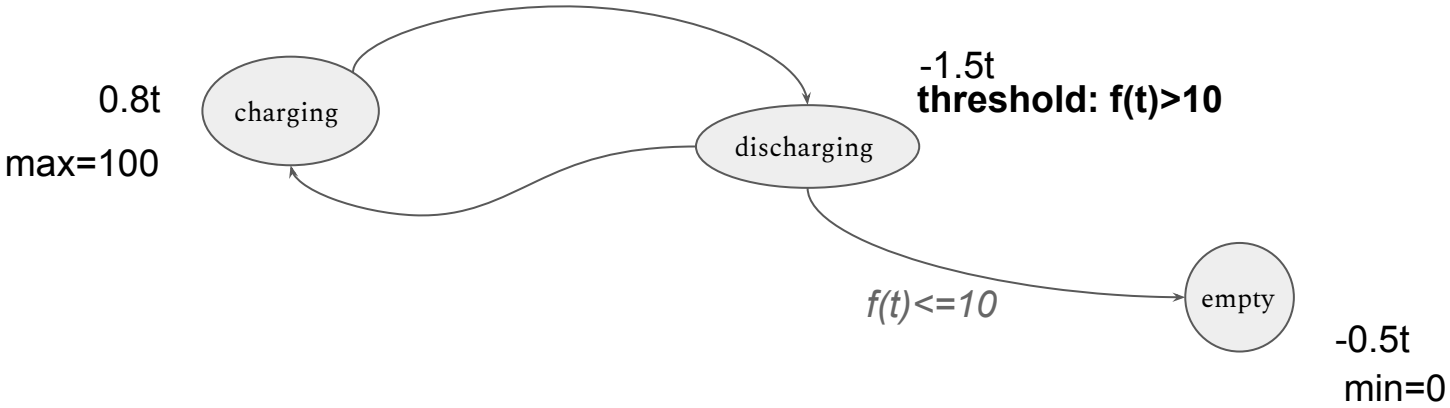
decidability vs undecidability

PROBLEM: the algorithm does not always terminates. *Why?*

decidability vs undecidability

PROBLEM: the algorithm does not always terminates. *Why?*

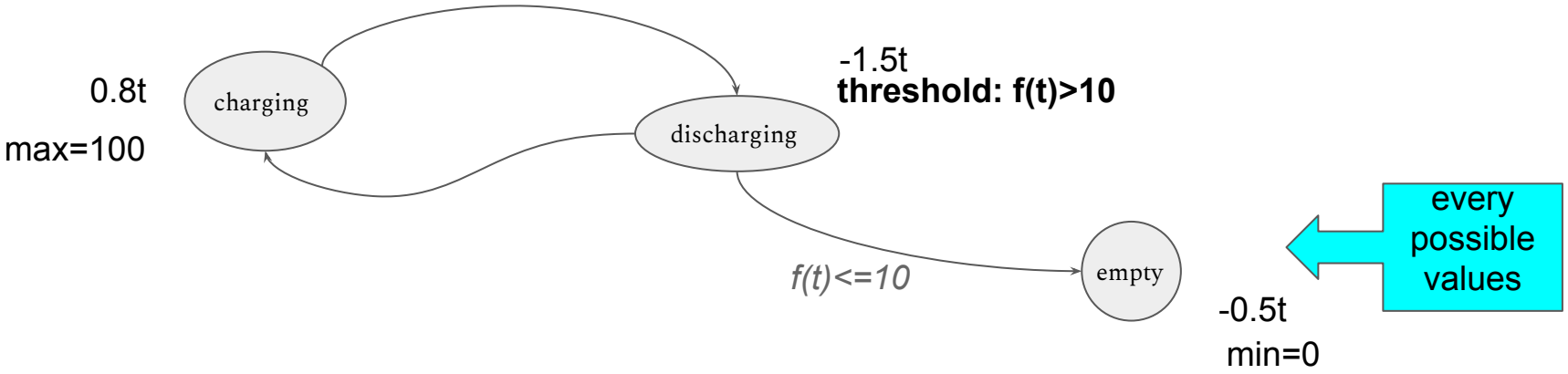
for example



decidability vs undecidability

PROBLEM: the algorithm does not always terminates. *Why?*

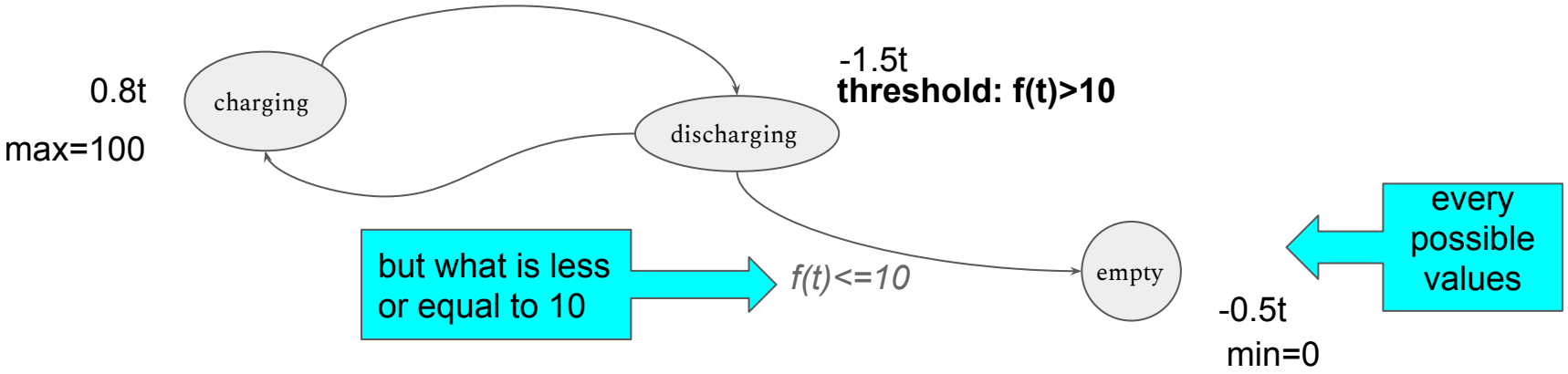
for example



decidability vs undecidability

PROBLEM: the algorithm does not always terminates. *Why?*

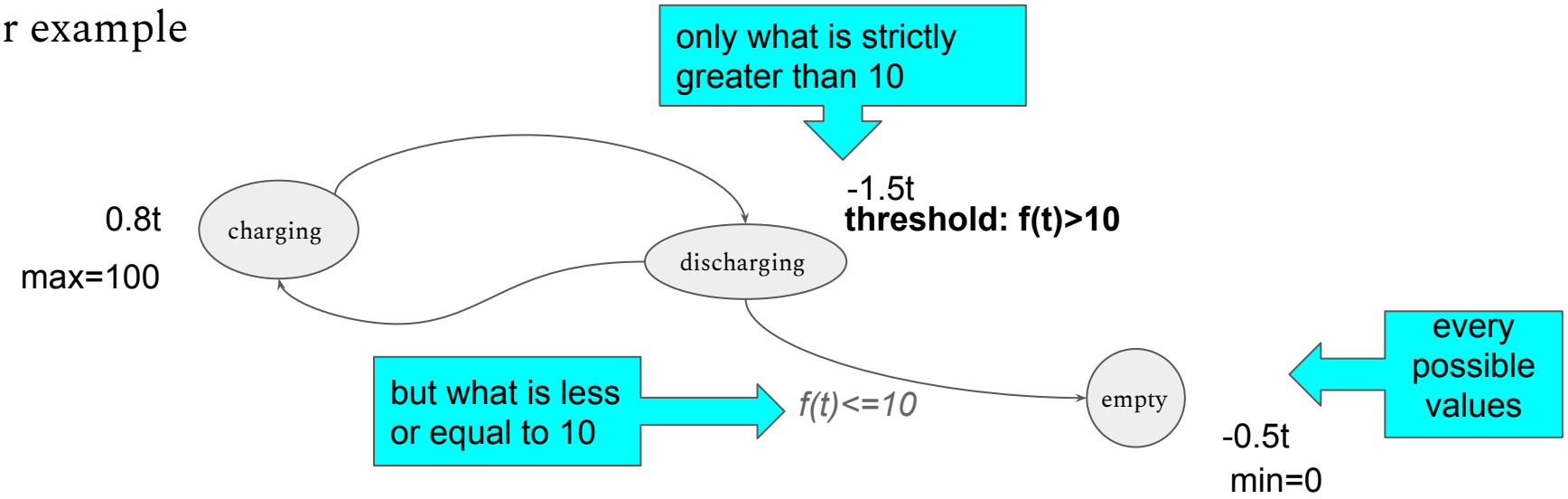
for example



decidability vs undecidability

PROBLEM: the algorithm does not always terminates. *Why?*

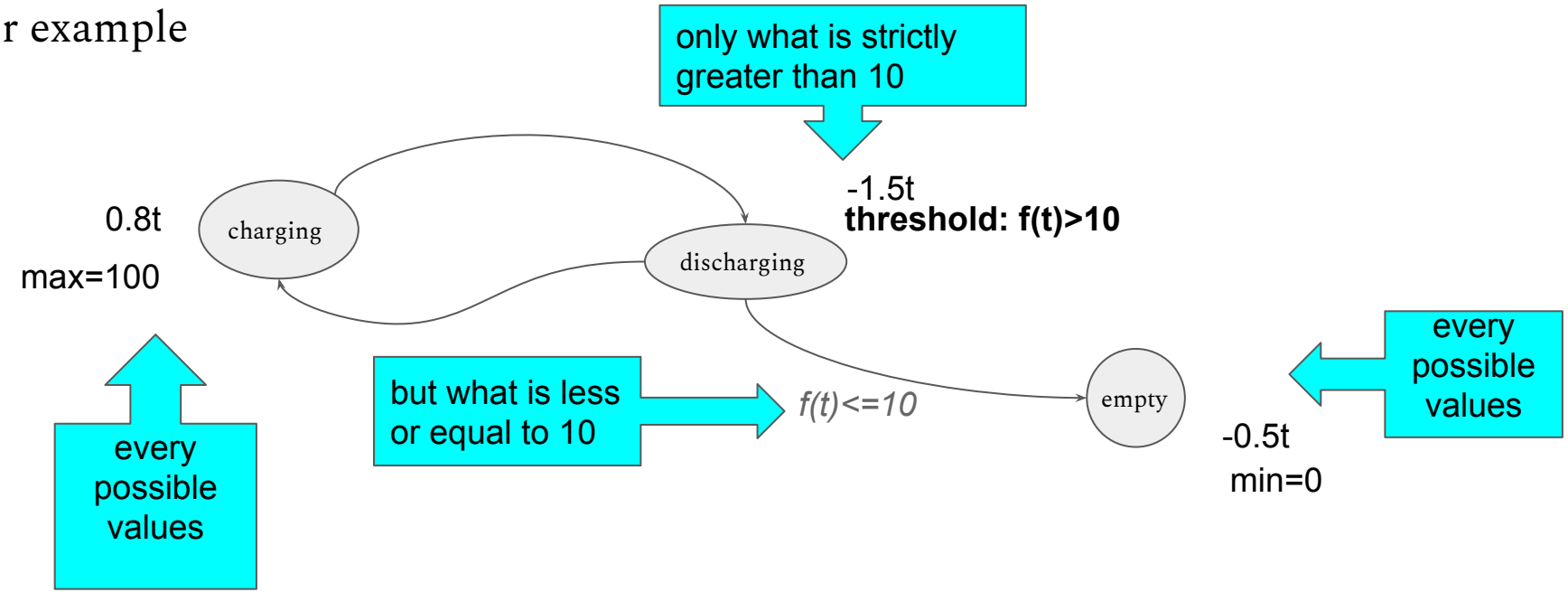
for example



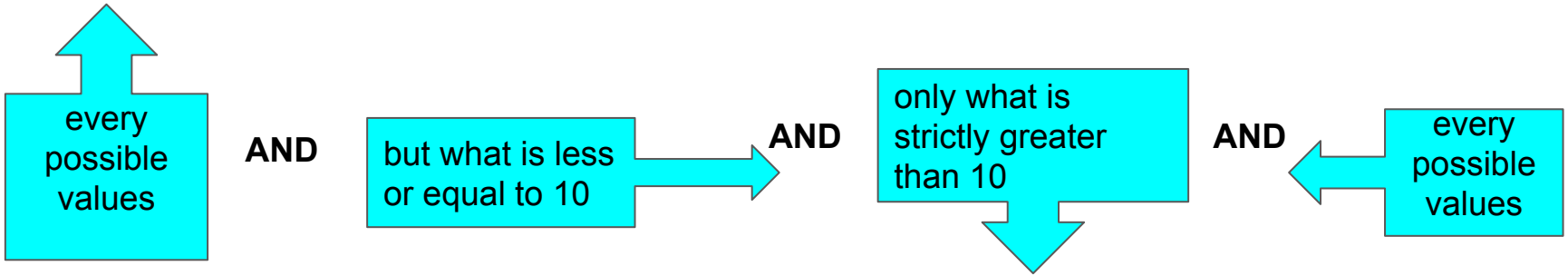
decidability vs undecidability

PROBLEM: the algorithm does not always terminates. *Why?*

for example

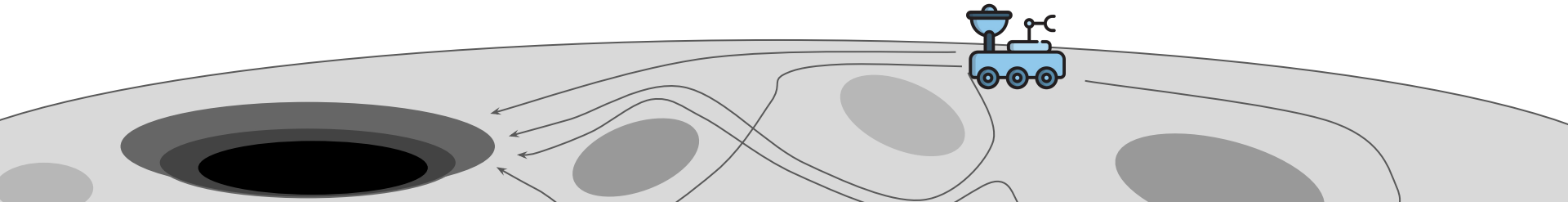


decidability vs undecidability



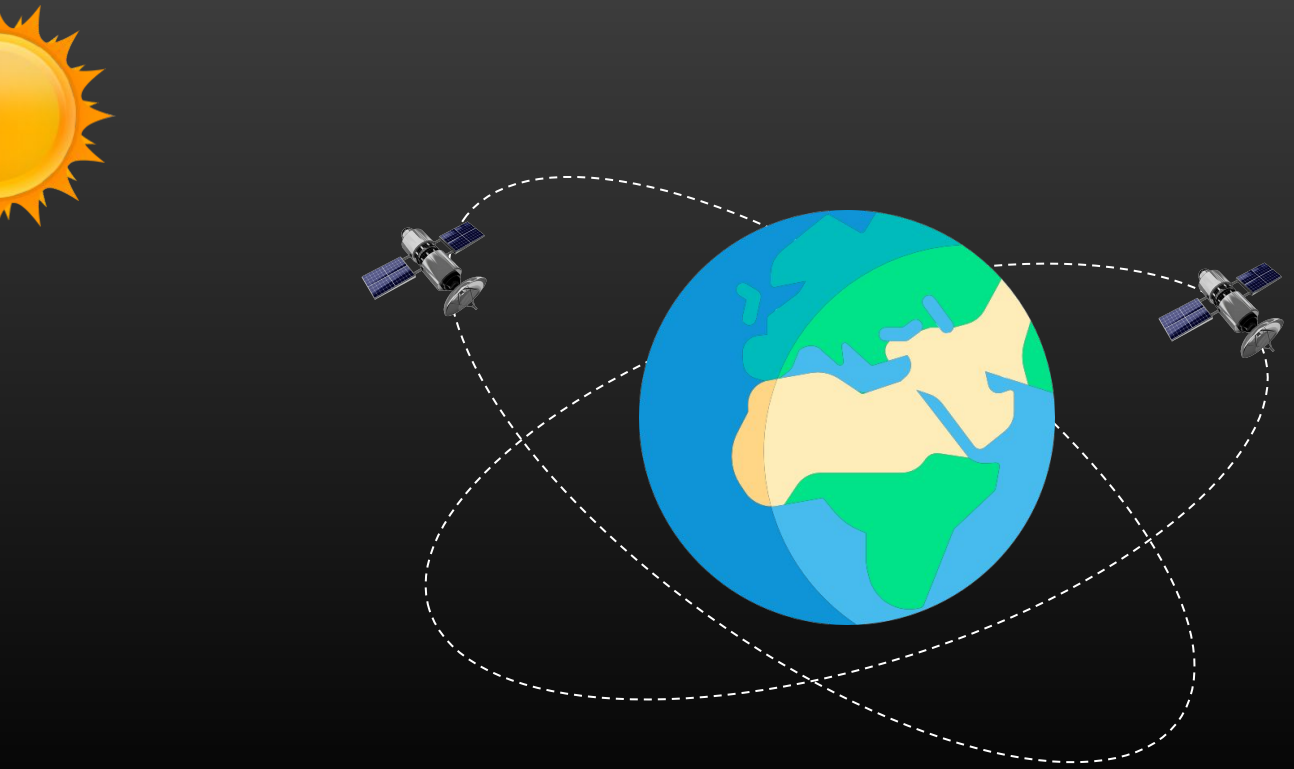
- Usually expressed as mathematical expressions
- Algorithm runs, and refines until it does not add or remove values
- Very costly, a lot of computation steps
- May not terminate because of loops

intuition: VIPER Moon rover trajectories



1. NASA's VIPER Moon rover: requirements and design specifications
2. Case study: Cubesat constellation

Cubesat constellation



model-checking

- Several efficient tools, used in industry
- most famous is Uppaal
- Event B and B method

demo on COMPASS?

Conclusion

testing

formal verification

testing

-fast
-efficient
-bugs sometimes

formal verification

-costly (money+computation power+time)
-very technical
-trustworthy

testing

-fast
-efficient
-bugs sometimes

formal verification

-costly (money+computation power+time)
-very technical
-trustworthy

ultimate goal: find a hybrid technique

testing

-fast
-efficient
-bugs sometimes

formal verification

-costly (money+computation power+time)
-very technical
-trustworthy

ultimate goal: find a hybrid technique

good balance between performances, cost, efficiency in removing bugs