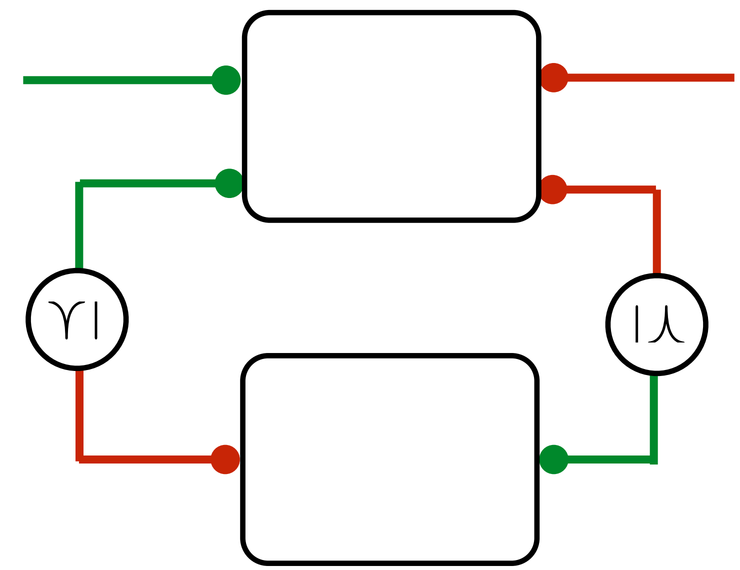
A Mathematical Theory of Co-Design

Andrea Censi



Research Scientist, Principal Investigator

Laboratory for Information and Decision Systems
Massachusetts Institute of Technology

About me

- Previous experience:
  - M.Eng in Automation & Robotics at **Sapienza** University
  - Ph.D. in Control & Dynamical System at **Caltech**
  - Visiting scholar at **UZH** (with Scaramuzza)

- Interests: **"Science of embodied autonomy"**
  - perception
  - control
  - calibration
  - learning

- **Today: formalizing the problem of "co-design"**

- There is a **lack of formal approaches** to the design of autonomous robotic systems.

robotic system

“robots”

hardware	**software**
actuators	perception
sensors	inference
energetics	planning
communication	control
computation	coordination
…	…

“humans”

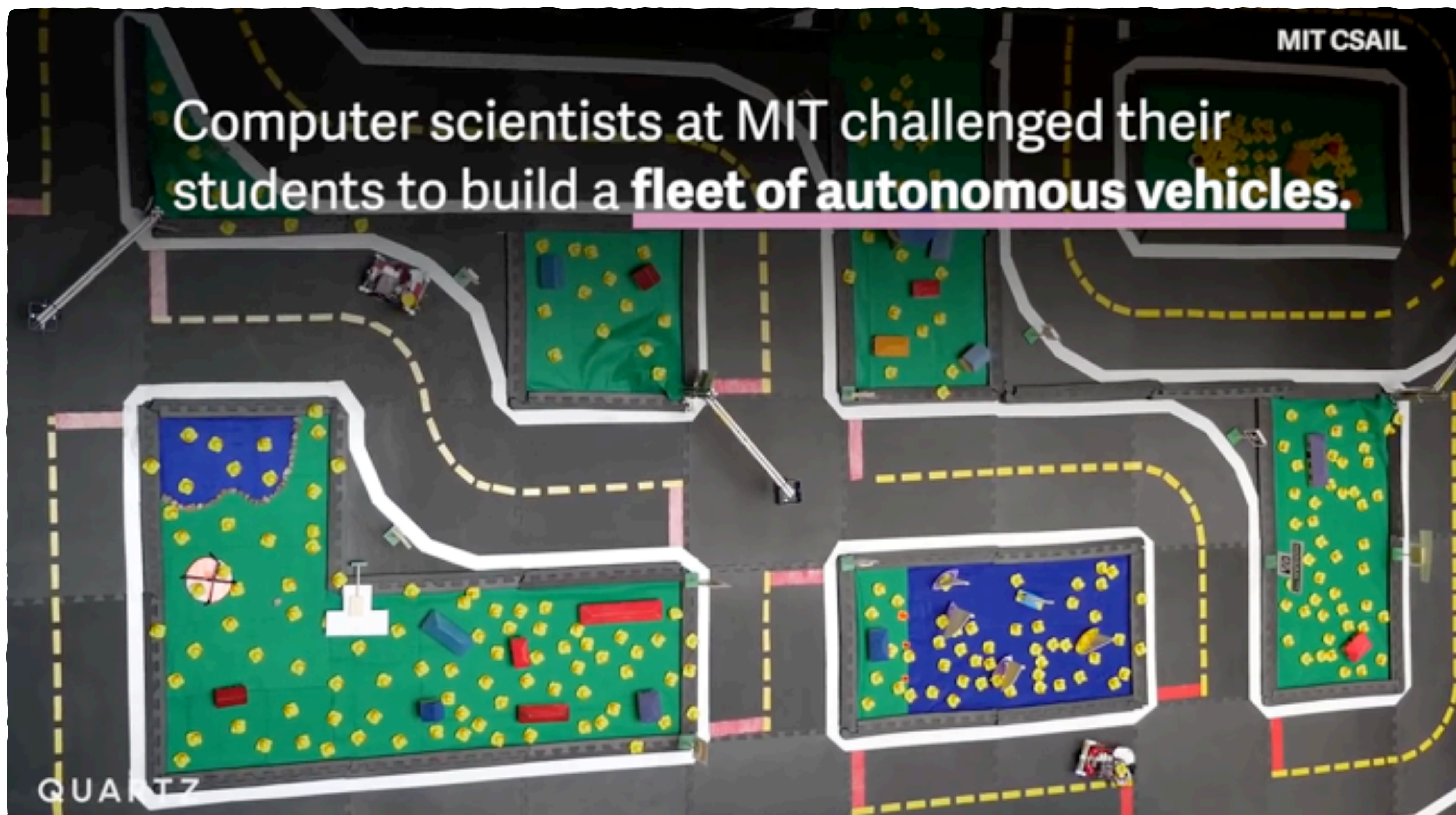
languages
interfaces
rules
procedures

“infrastructure”

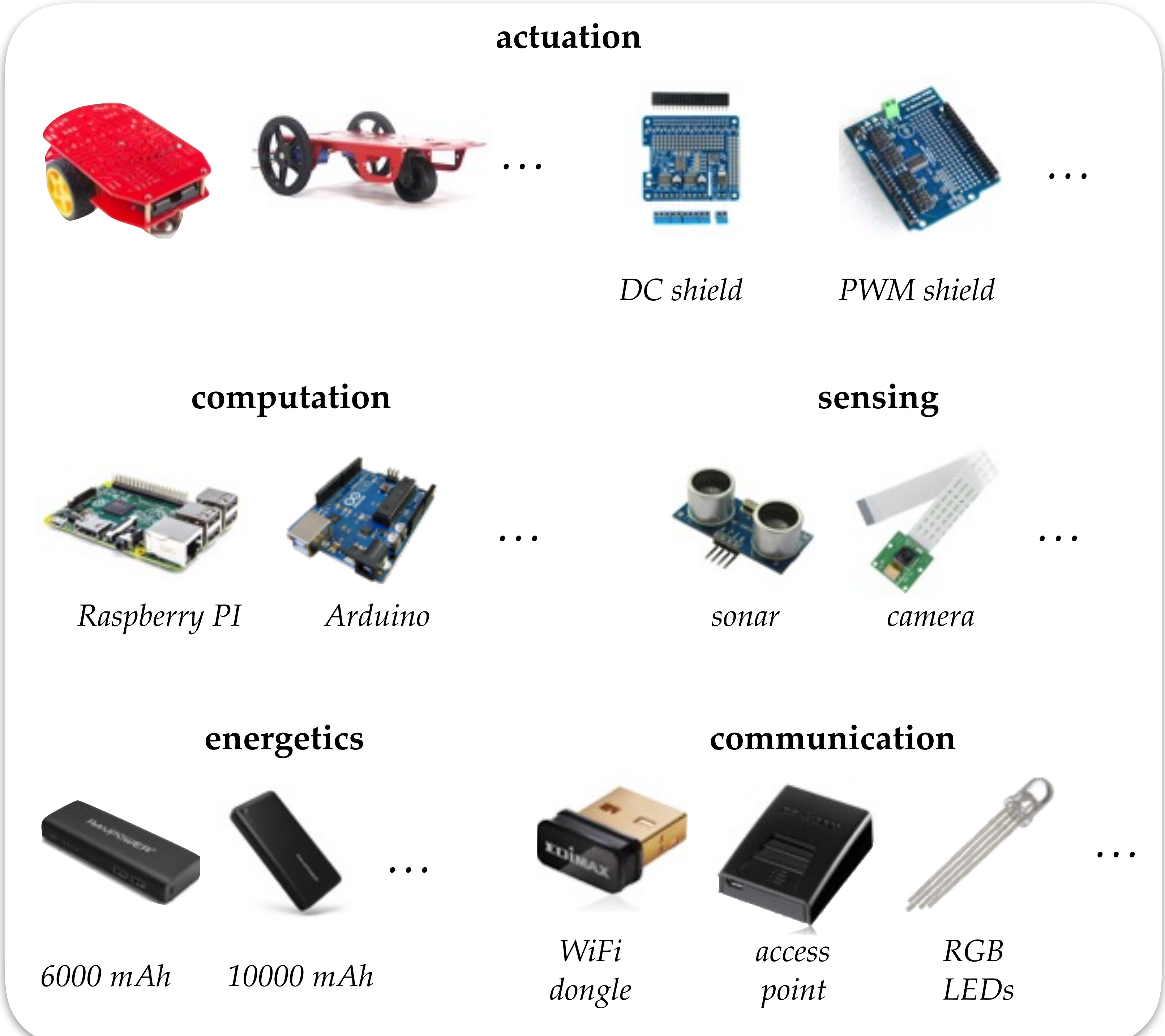
environment
prior knowledge
data
tests
simulations

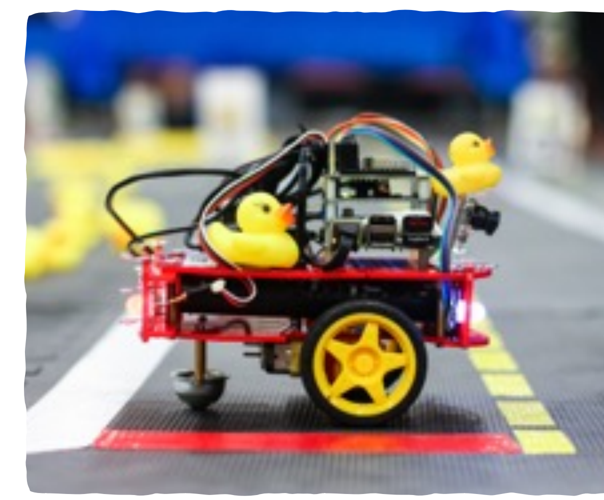
Kiva / Amazon Robotics (D'Andrea)

Duckietown: *a case study in minimality*



minimize cost
subject to teaching goals





minimize
(resources usage)

subject to
(functionality constraints)

task specification → “automated roboticist” → **optimal design**

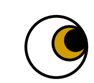
catalogue of parts → “automated roboticist”

- **Contribution**: **a theory of co-design**

heterogenous domains

propulsion

computation



sensing

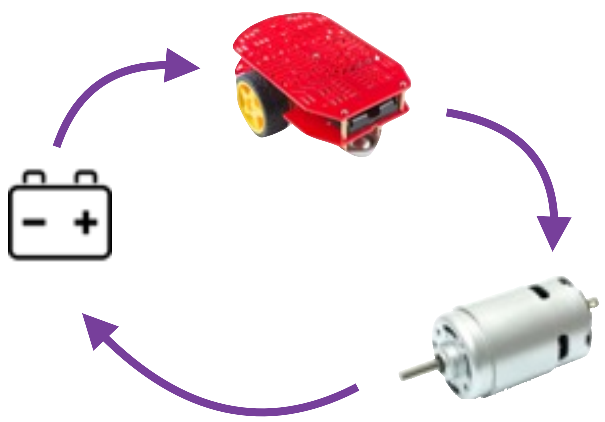
energetics

communication

...

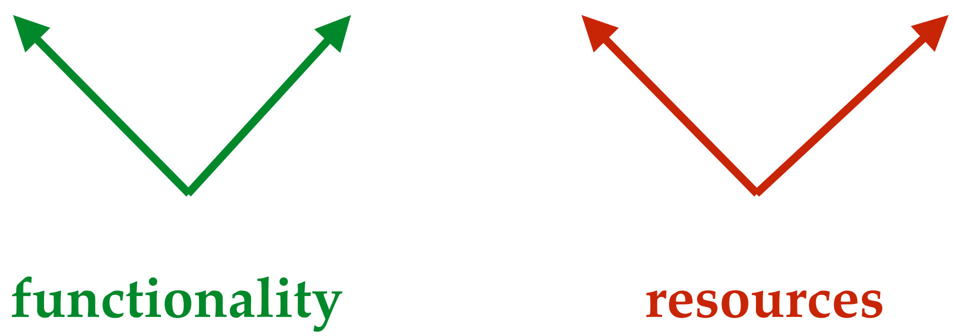
- **Contribution**: **a theory of co-design**

recursive
co-design constraints



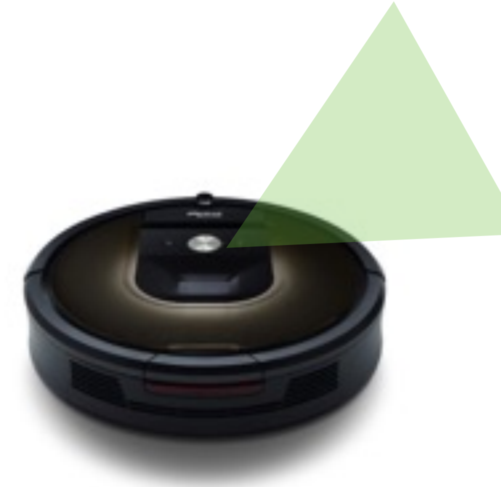
- **Contribution: a theory of co-design**

trade-offs of
functionality and resources



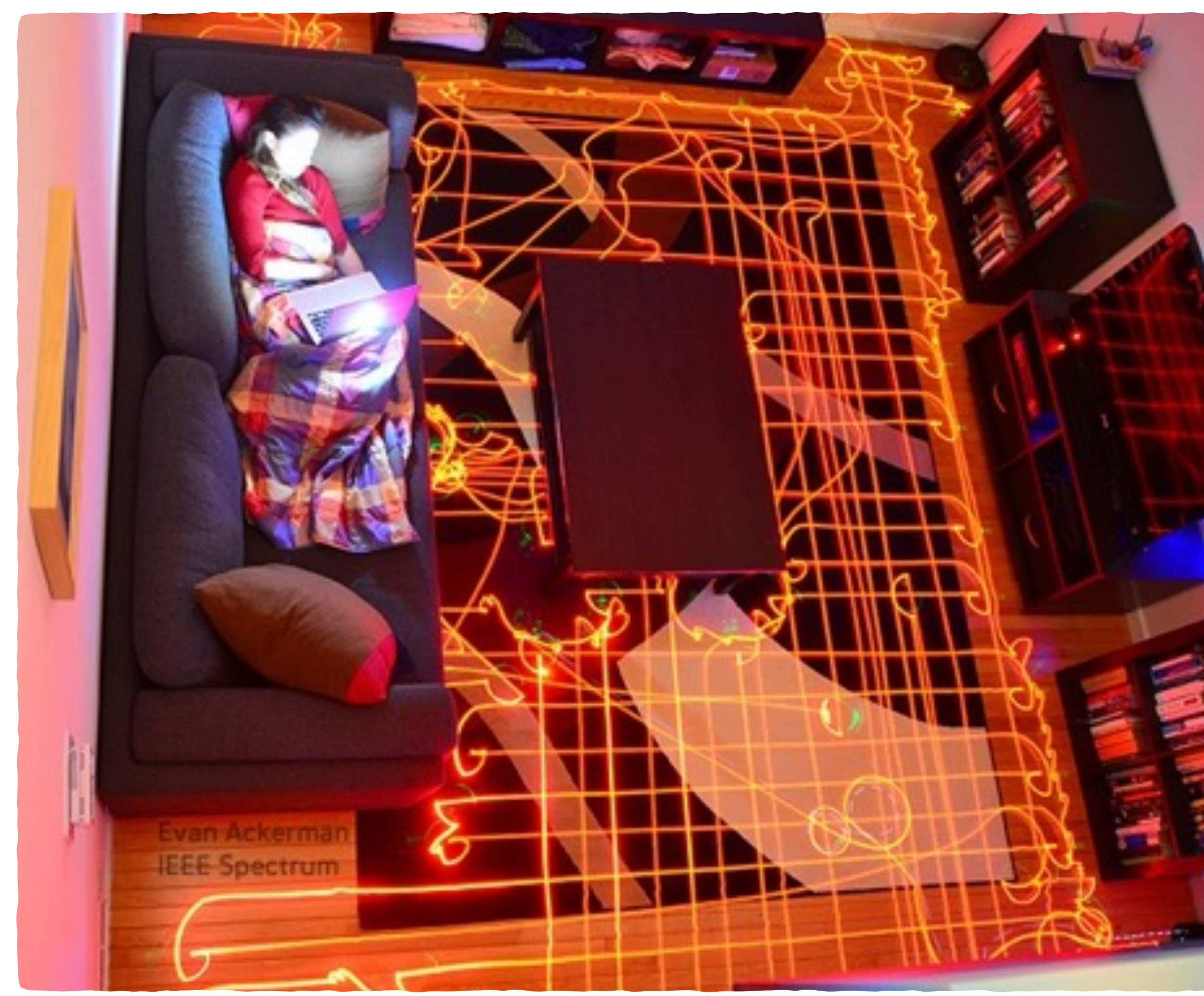
Roomba

$200

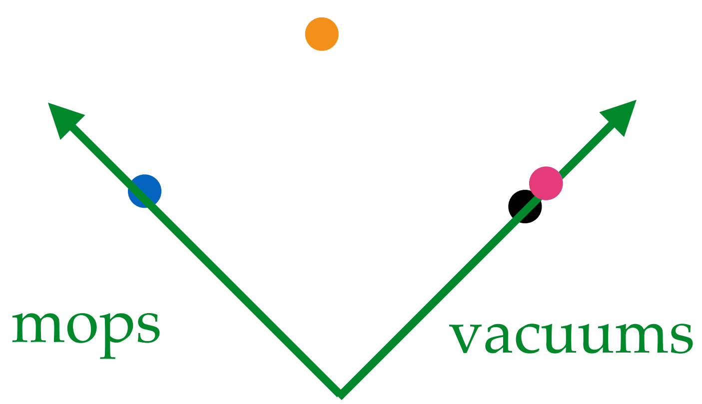


Roomba 980

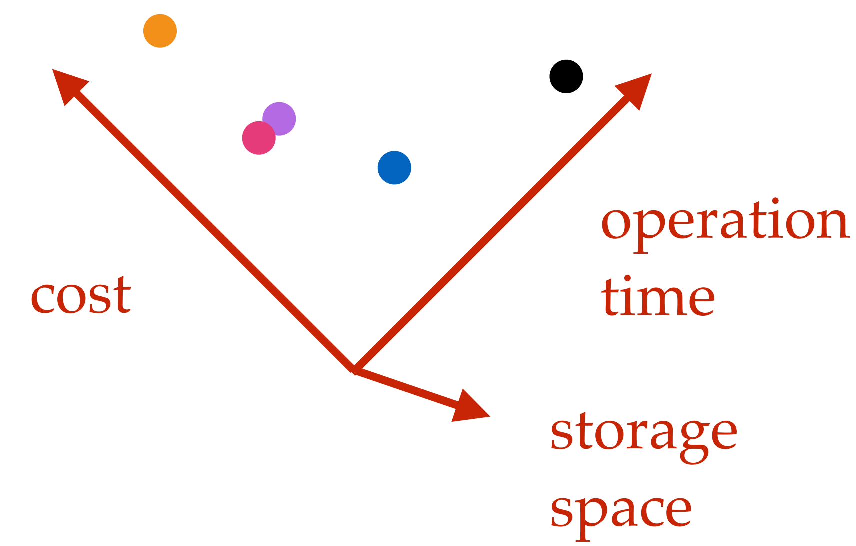
$900



cleaning robots

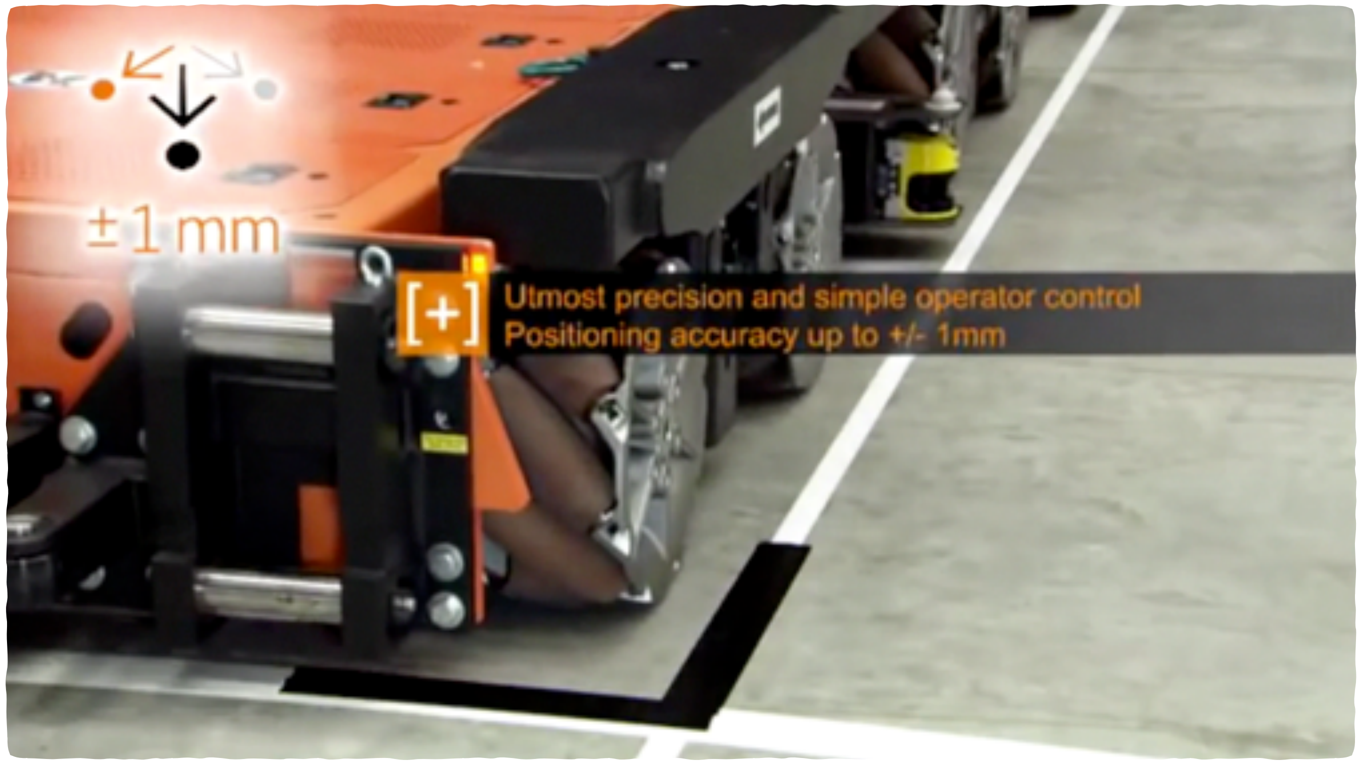


functionality

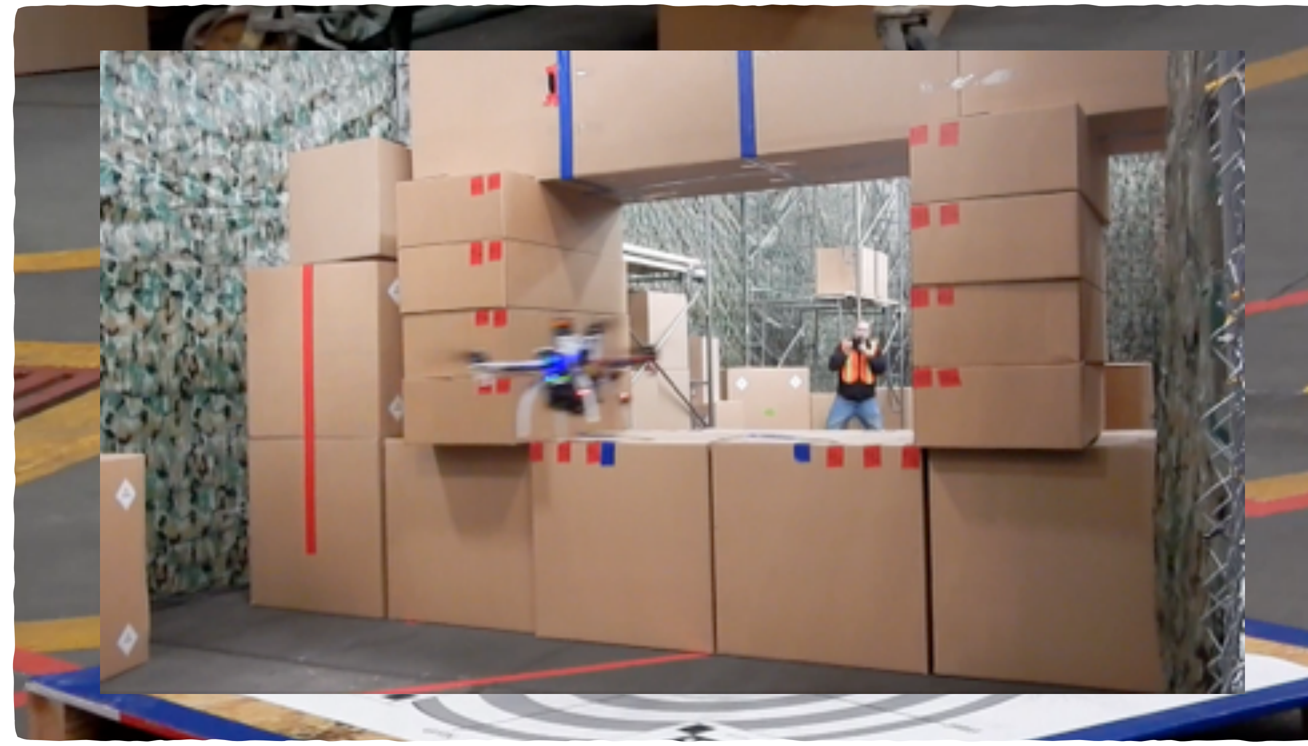


resources

Kuka's Omnimove



MIT/DRAPER (PI: Roy)



+ PLICP scan matcher [Censi ICRA'07,08,09]

platform velocity

estimation accuracy

functionality

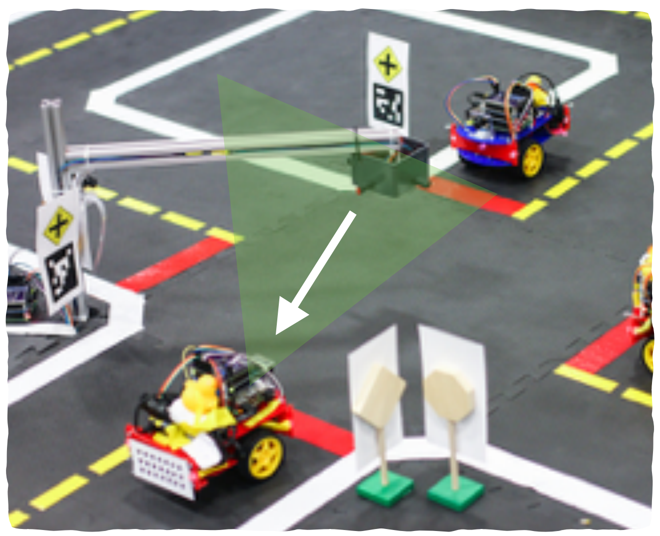
cost

payload

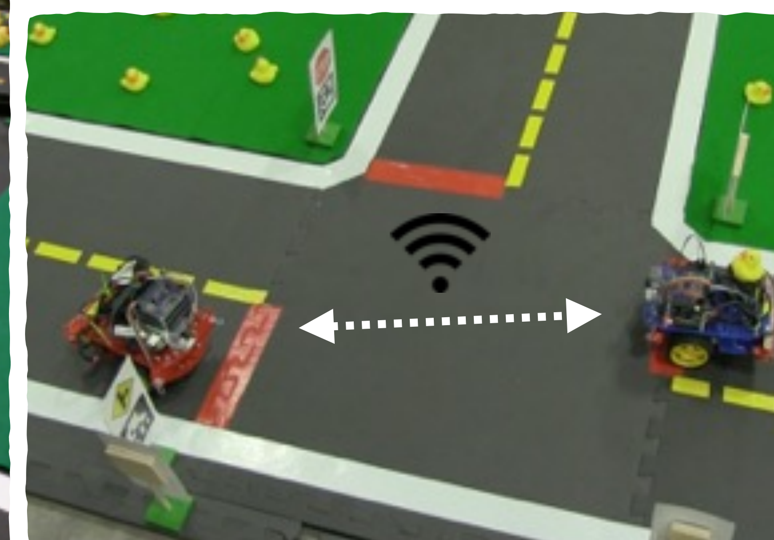
resources

distributed / vision

centralized / vision



distributed / network



safety

performance

functionality

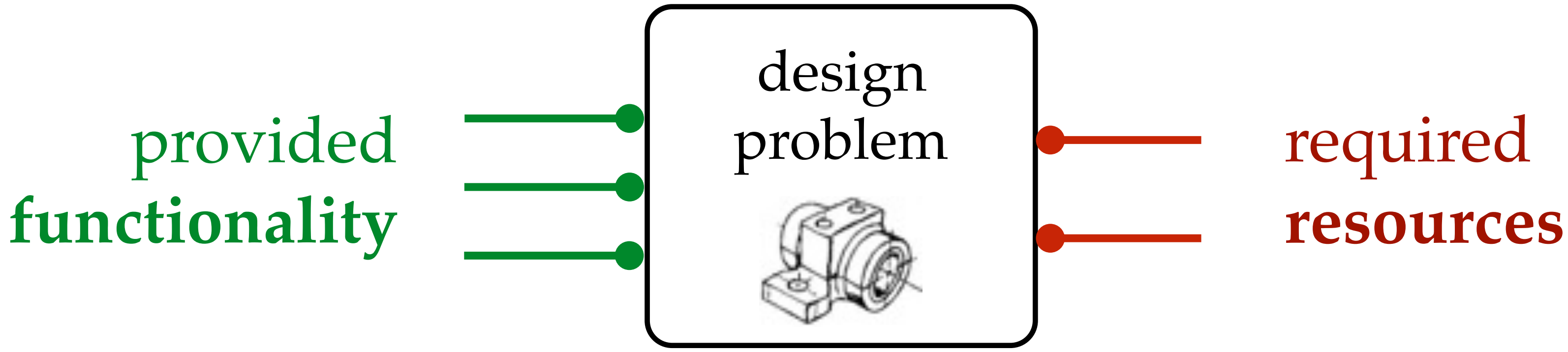
complexity
of **infrastructure**

complexity
of platform

resources

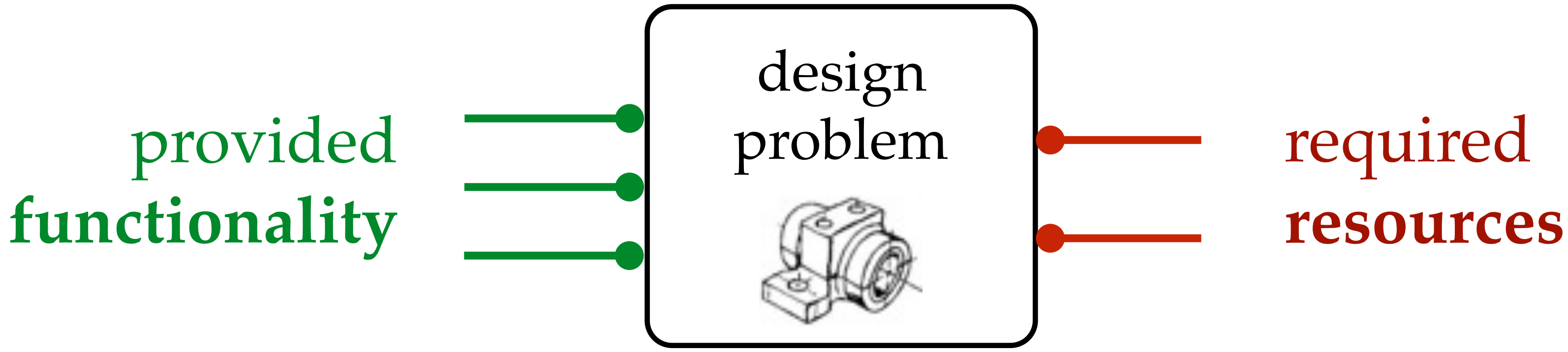
- The quest for "**minimality**" in robotics
  - **actuation/control:** Bicchi, Mason, Rodriguez, Goldberg, Wood, Fearing, Ijspeert, …
  - **sensing:** Floreano**,** Lavalle, O'Kane, Davison, …
  - **representations/inference**: Soatto, Milford, …

- A **design problem** is a **relation** between **provided functionality** and **required resources.**



“functional requirements”
“desired behavior”
“required performance”
“specifications”

- A **design problem** is a **relation** between **provided functionality** and **required resources.**



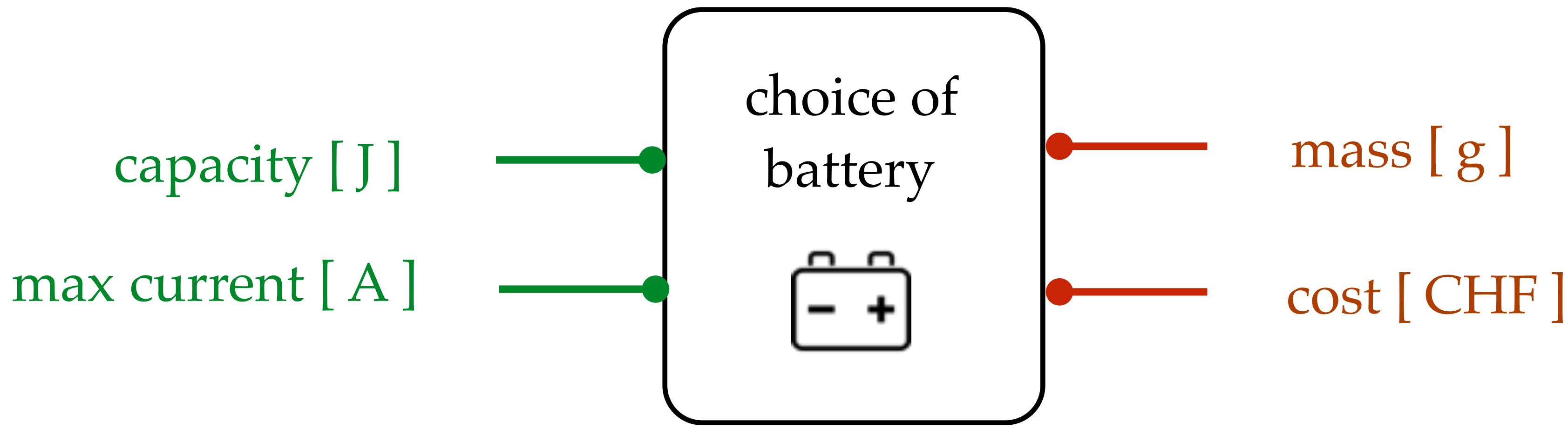
$\langle \mathcal{F}, \preceq_{\mathcal{F}} \rangle$

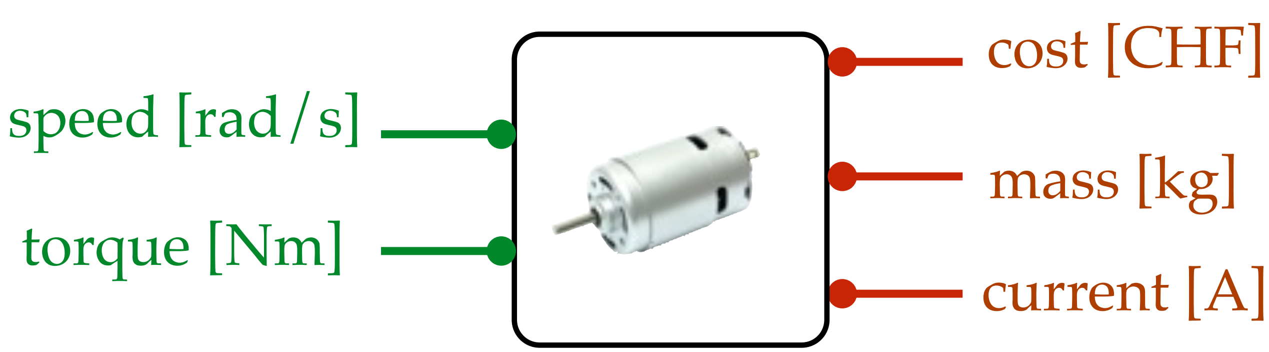
any partially ordered set

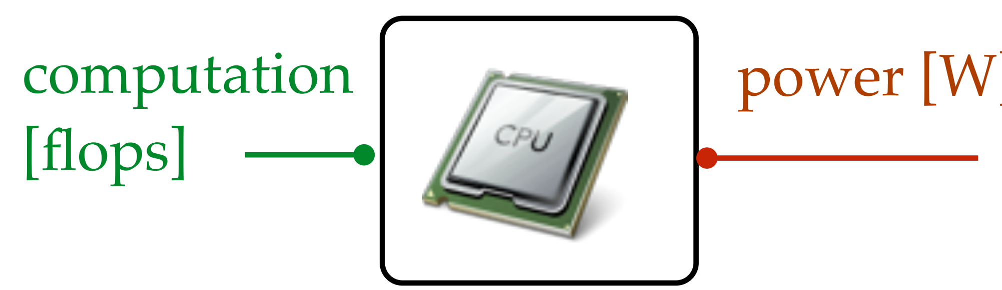
$\langle \mathcal{R}, \preceq_{\mathcal{R}} \rangle$

any partially ordered set

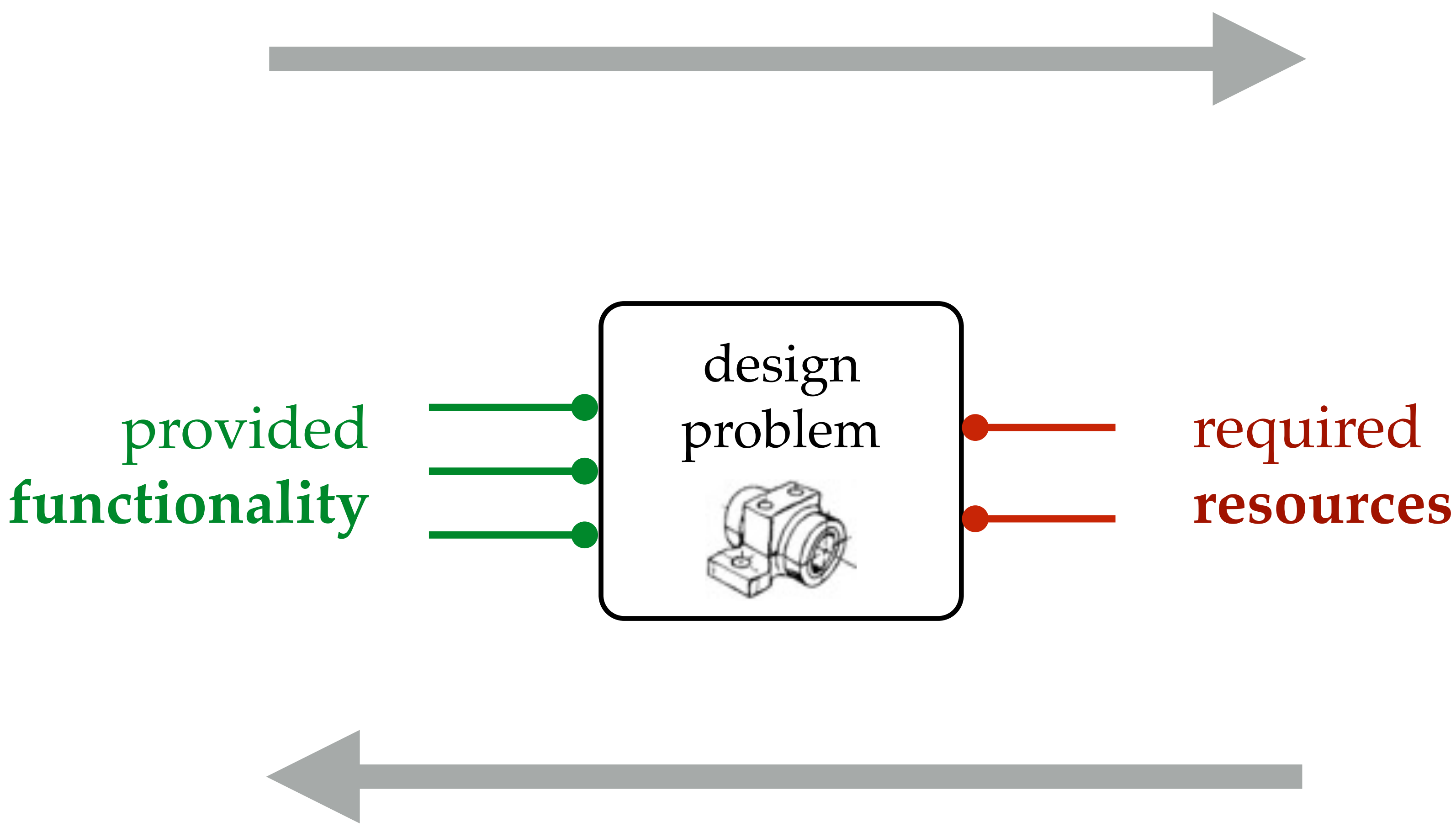
- A **design problem** is a **relation** between **provided functionality** and **required resources.**





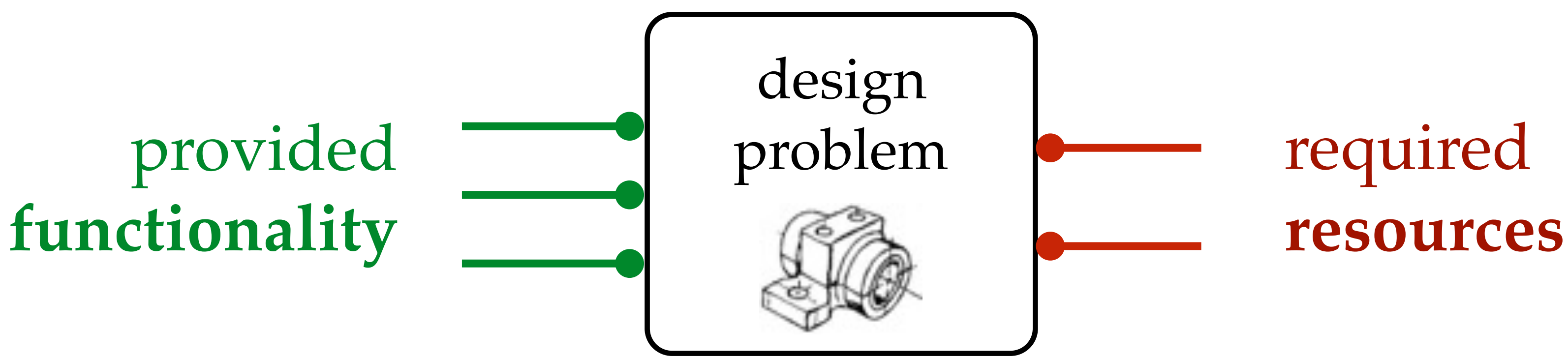


- **Given the functionality** to be provided,
  what are the **minimal resources** required?

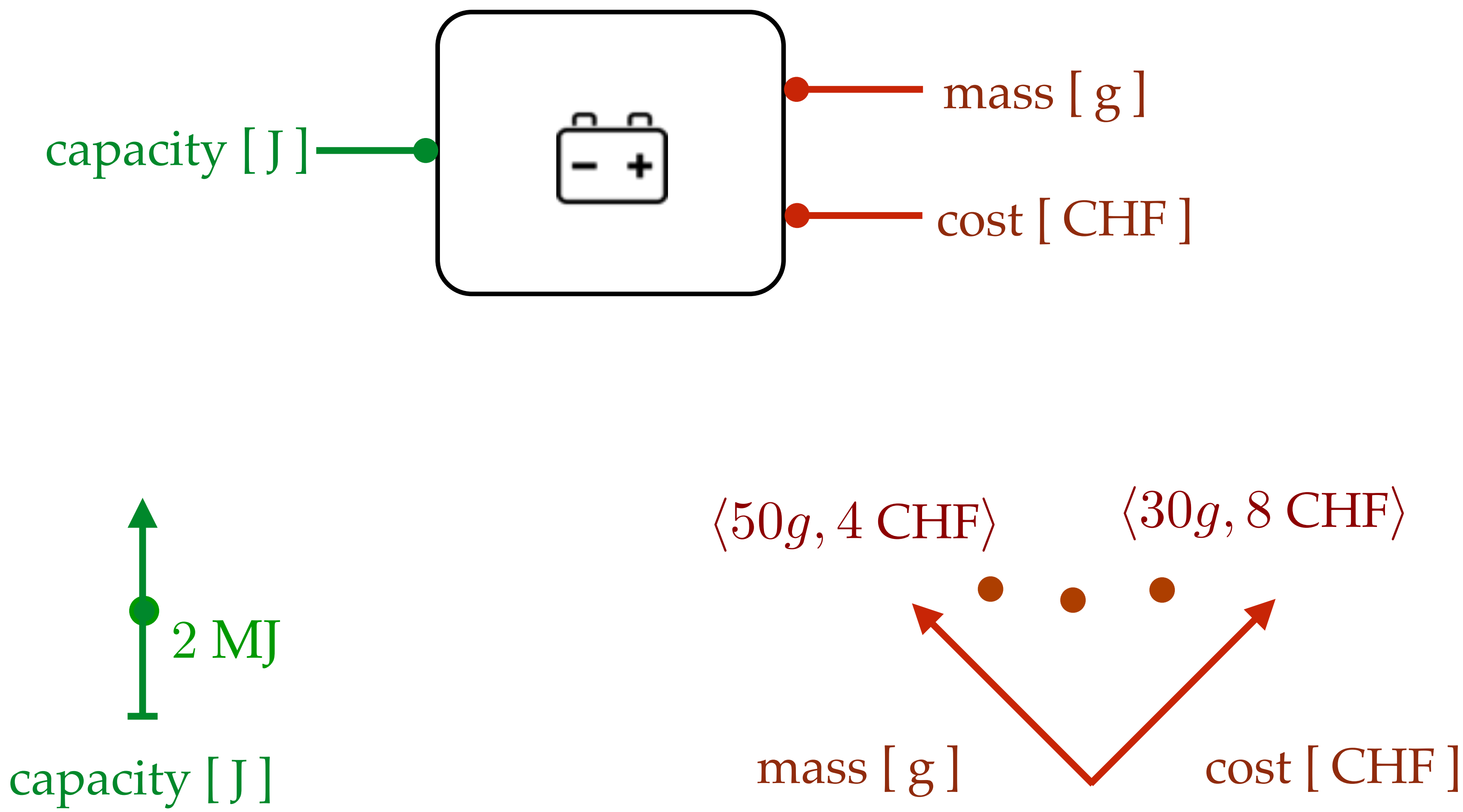


- **Given the resources** that are available,
  what is the **maximal functionality** that can be provided?

- **Given the functionality** to be provided, what are the **minimal resources** required?



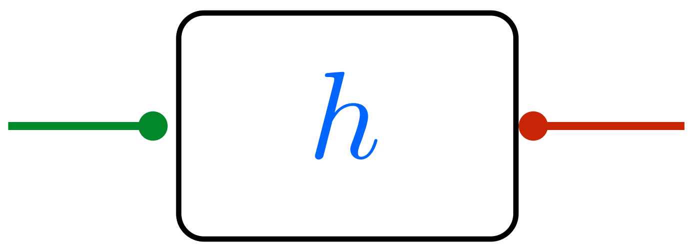
- **Given the functionality** to be provided, what are the <u>**minimal**</u> **resources** required?

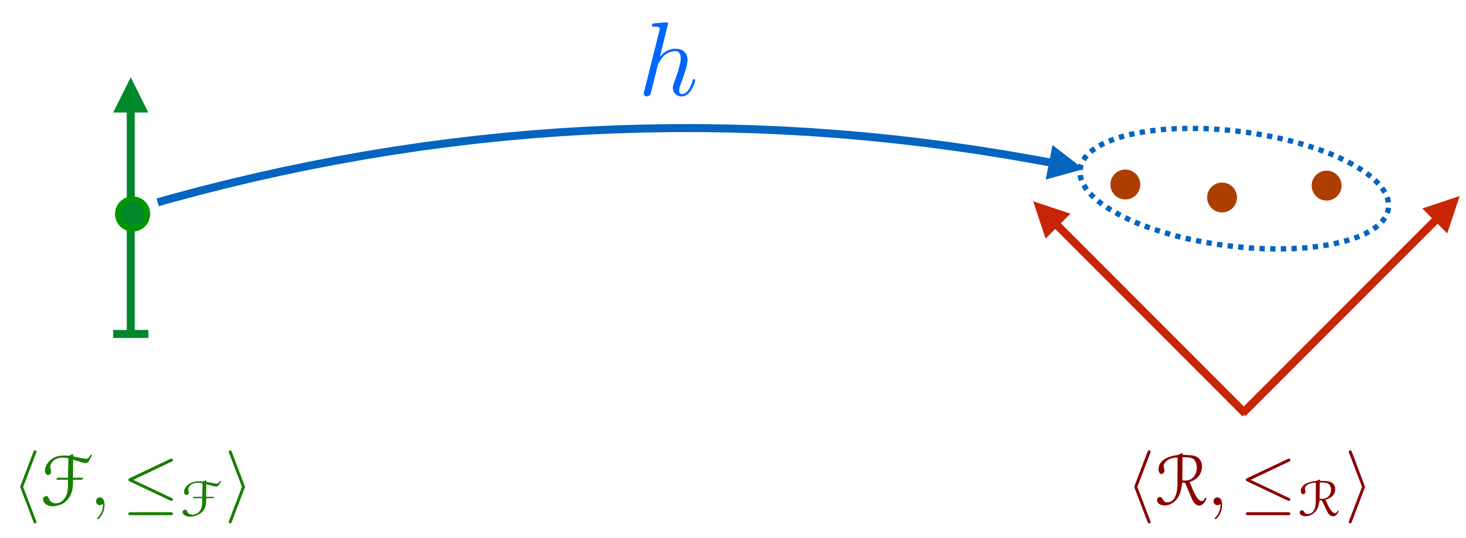


- A design problem can be concretely represented as a map

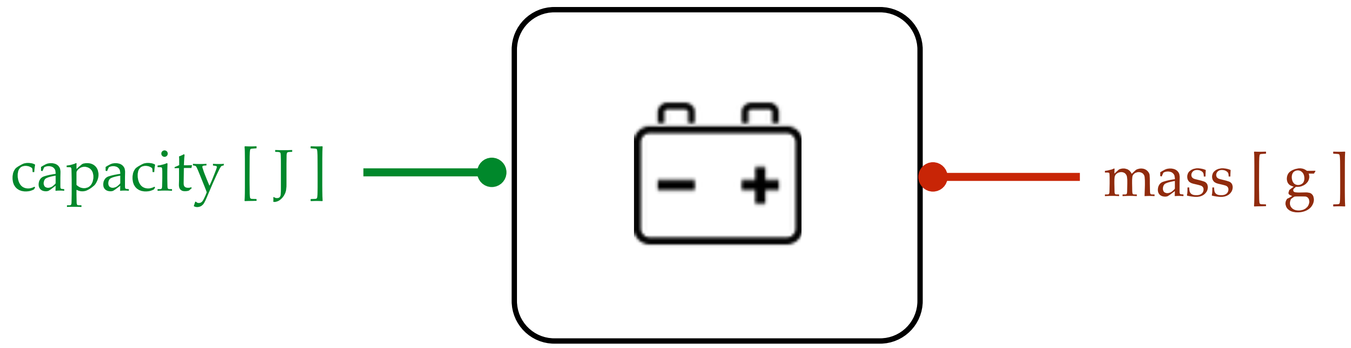
$$h : \mathcal{F} \to \text{subsets}(\mathcal{R})$$

from **functionality** to "**minimal subsets**" of **resources**.

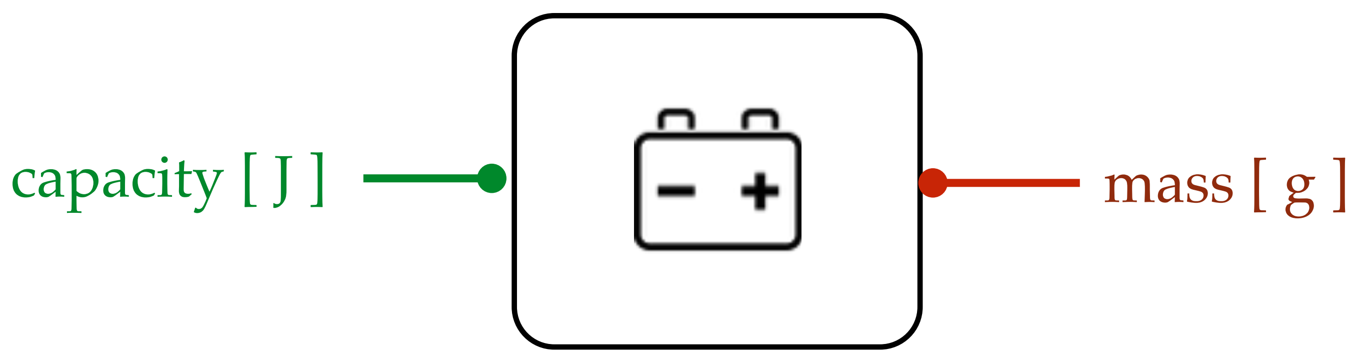


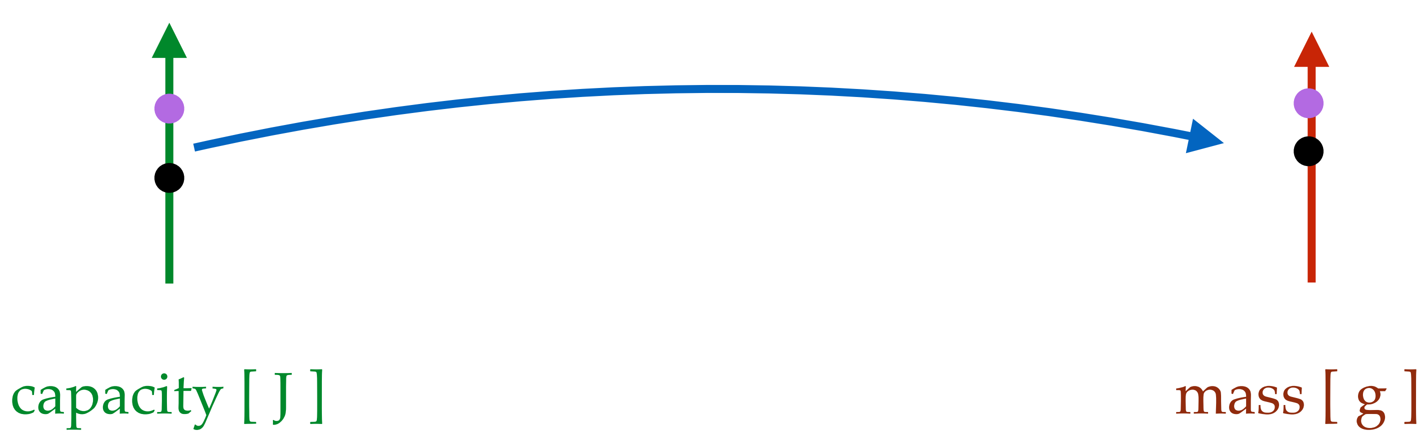


- **Monotonicity**: increasing the functional requirements does not decrease the resources required.

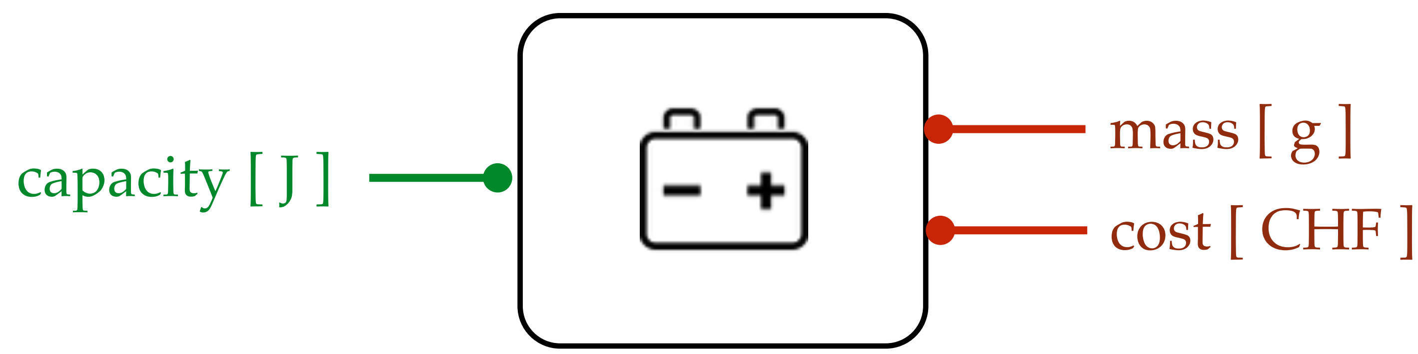


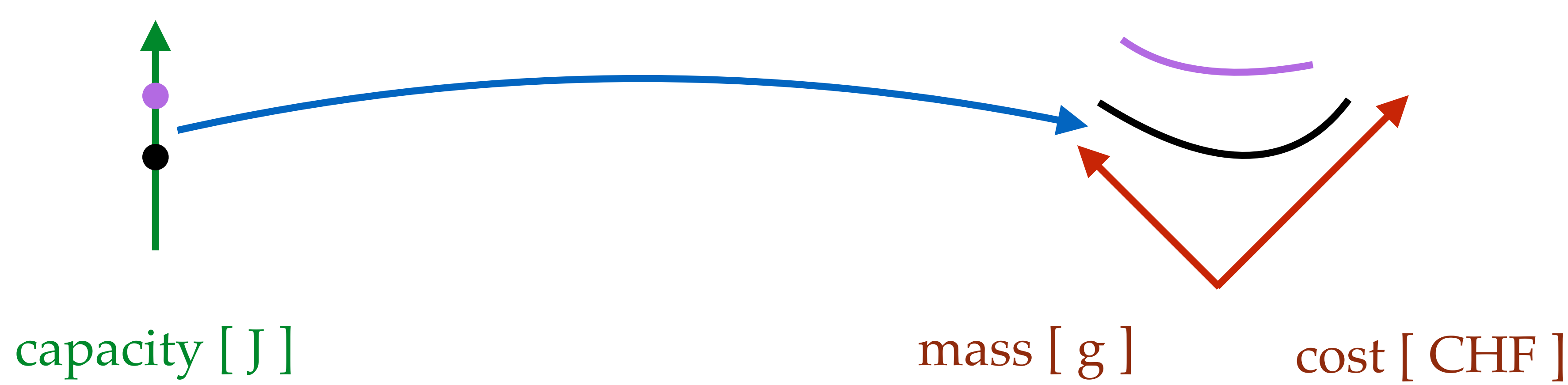
- **Monotonicity**: increasing the functional requirements does not decrease the resources required.





- **Monotonicity**: increasing the functional requirements does not decrease the resources required.





- **Monotonicity**: increasing the functional requirements does not decrease the resources required.

What is the order on trade-off curves?

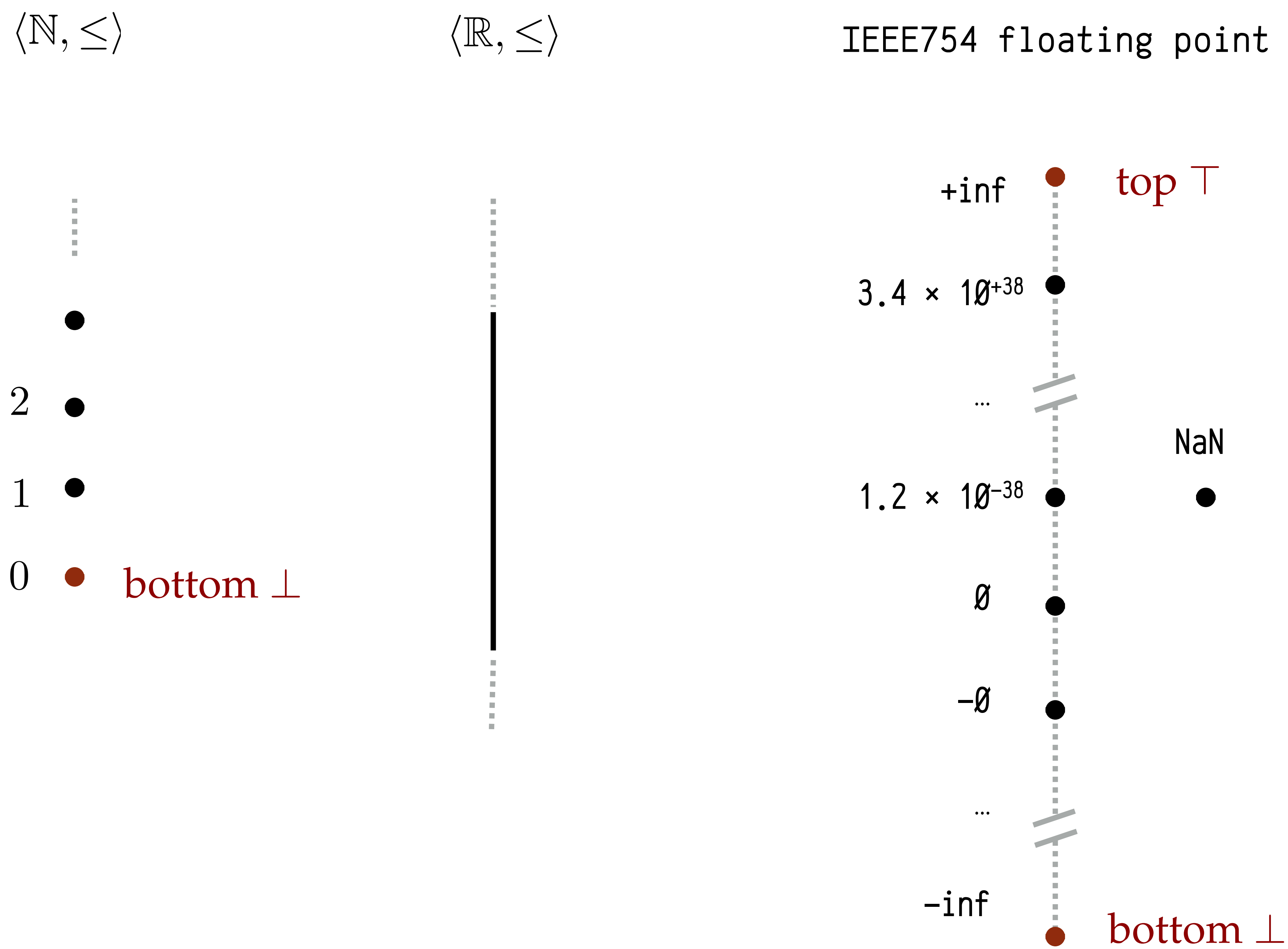
- **Background needed to understand the problem:**
  - posets
  - minimal elements
  - antichains
  - monotonicity
  - upper sets and upper closure

- **Background needed to understand the solution:**
  - Scott-continuity
  - Kleene's fixed point theorem
  - order theory: height and width of poset

- **Background not needed:**

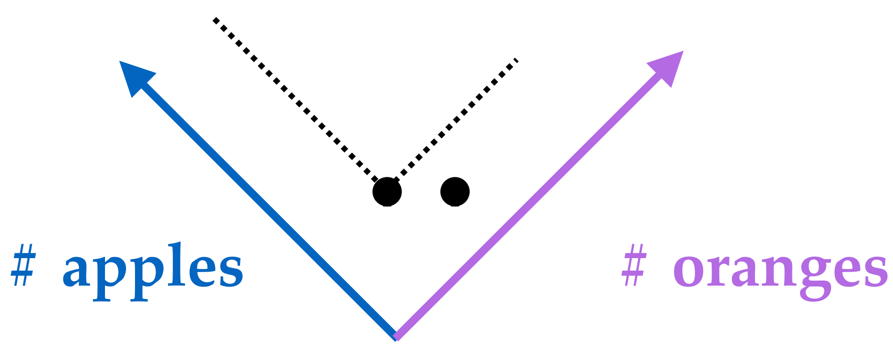
$$\partial \quad \nabla \quad \nabla^2$$

- **Poset** = a set together with a **partial order** $\preceq$



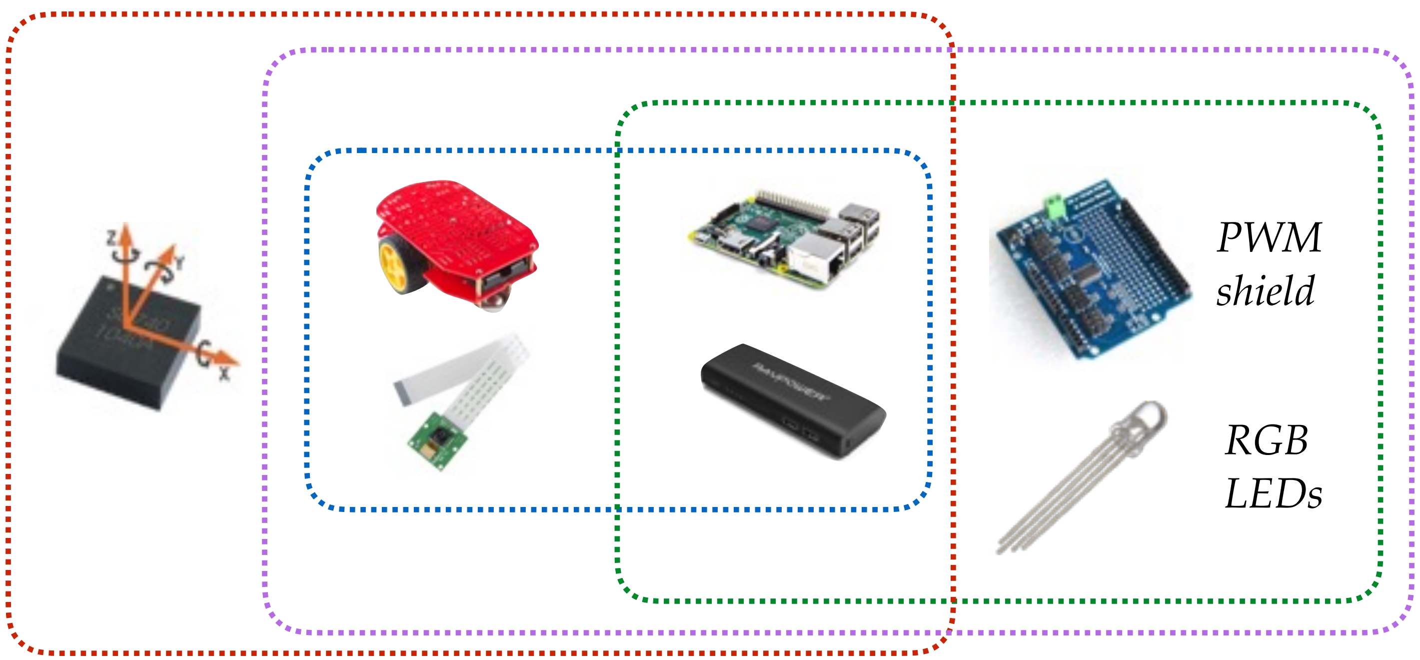
- **Poset** = a set together with a **partial order** $\preceq$

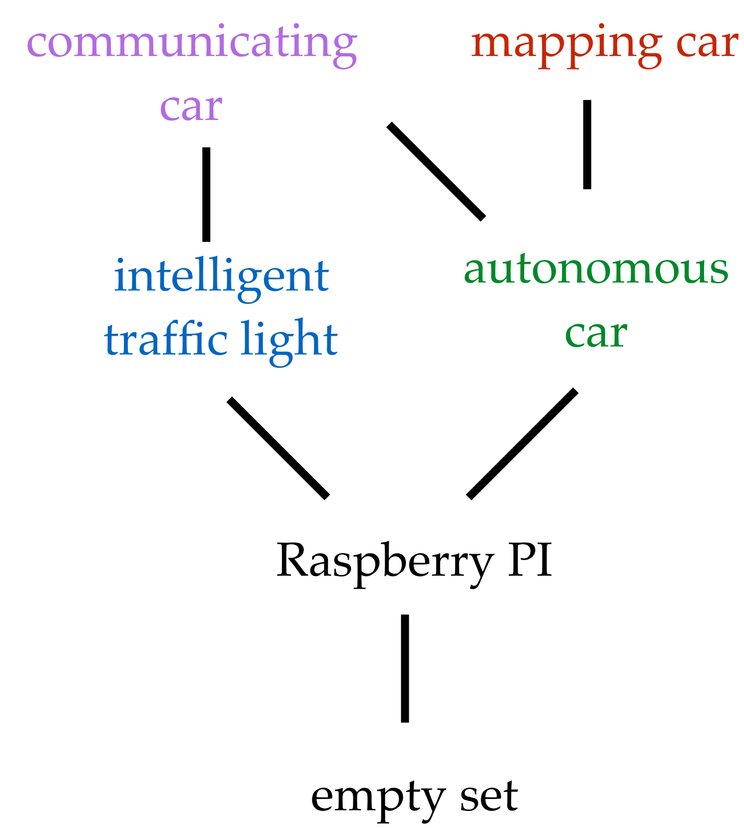
- Example: **products of posets**



- **Poset** = a set together with a **partial order** $\preceq$

- Example: **subsets, ordered by inclusions**

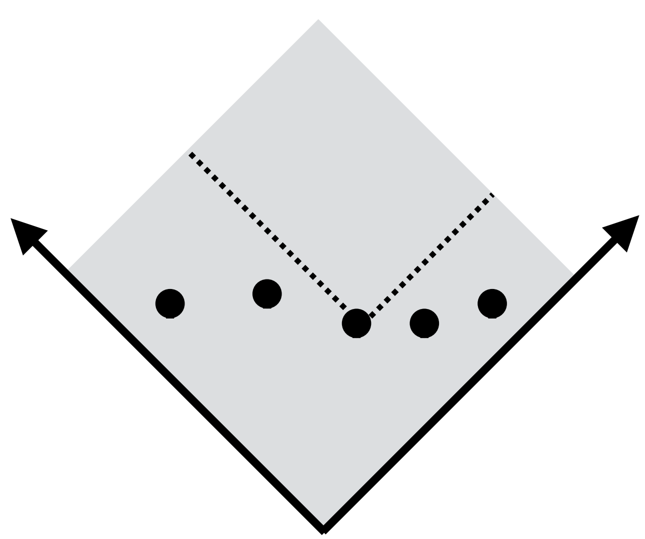




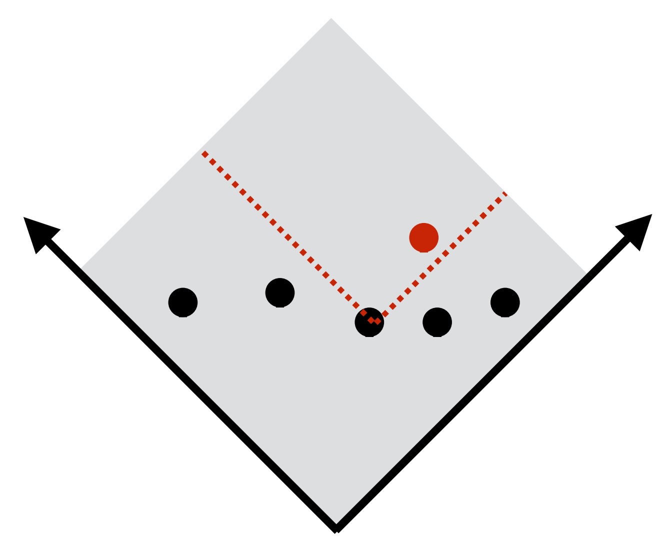
- **Antichain** = a set of elements that are mutually incomparable:

$$(a \preceq b) \Rightarrow (a = b)$$

example of antichain

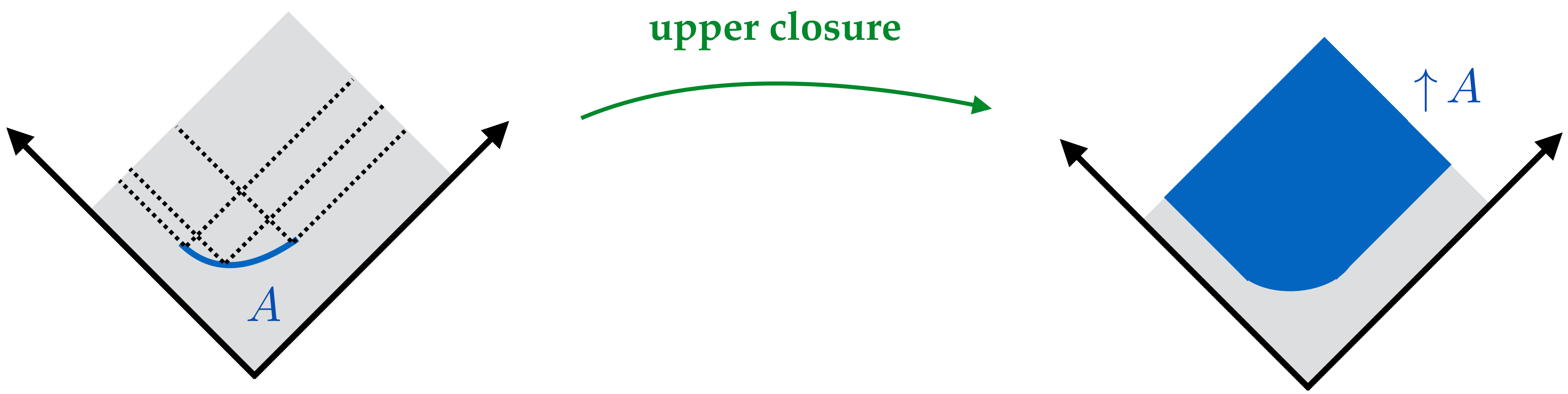


not an antichain



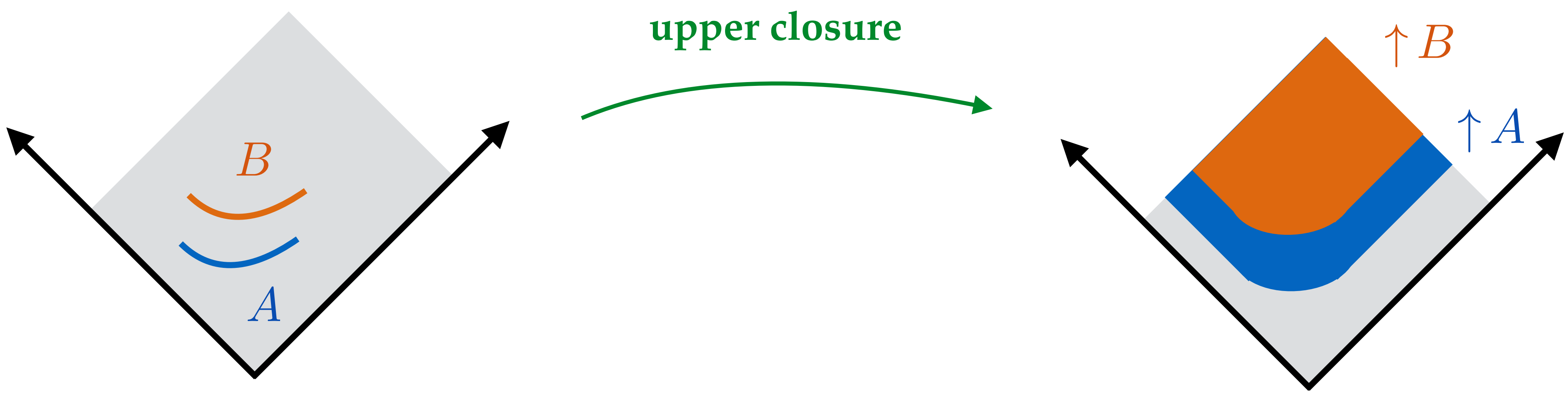
- **Upper closure** (↑) of a subset S of a poset P:

$$\uparrow S = \{y \in P : \exists x \in S : x \preceq y\}$$



- **Definition.** Choose this partial order for antichains:

$$A \preceq B \quad \doteq \quad \uparrow A \supseteq \uparrow B$$

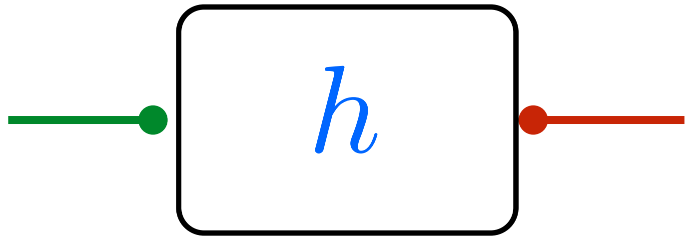


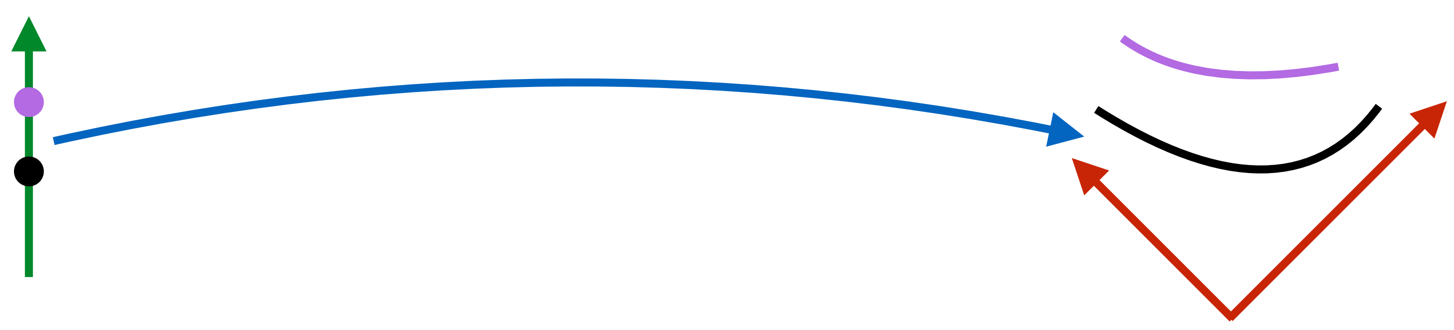
add definition of monotonic?

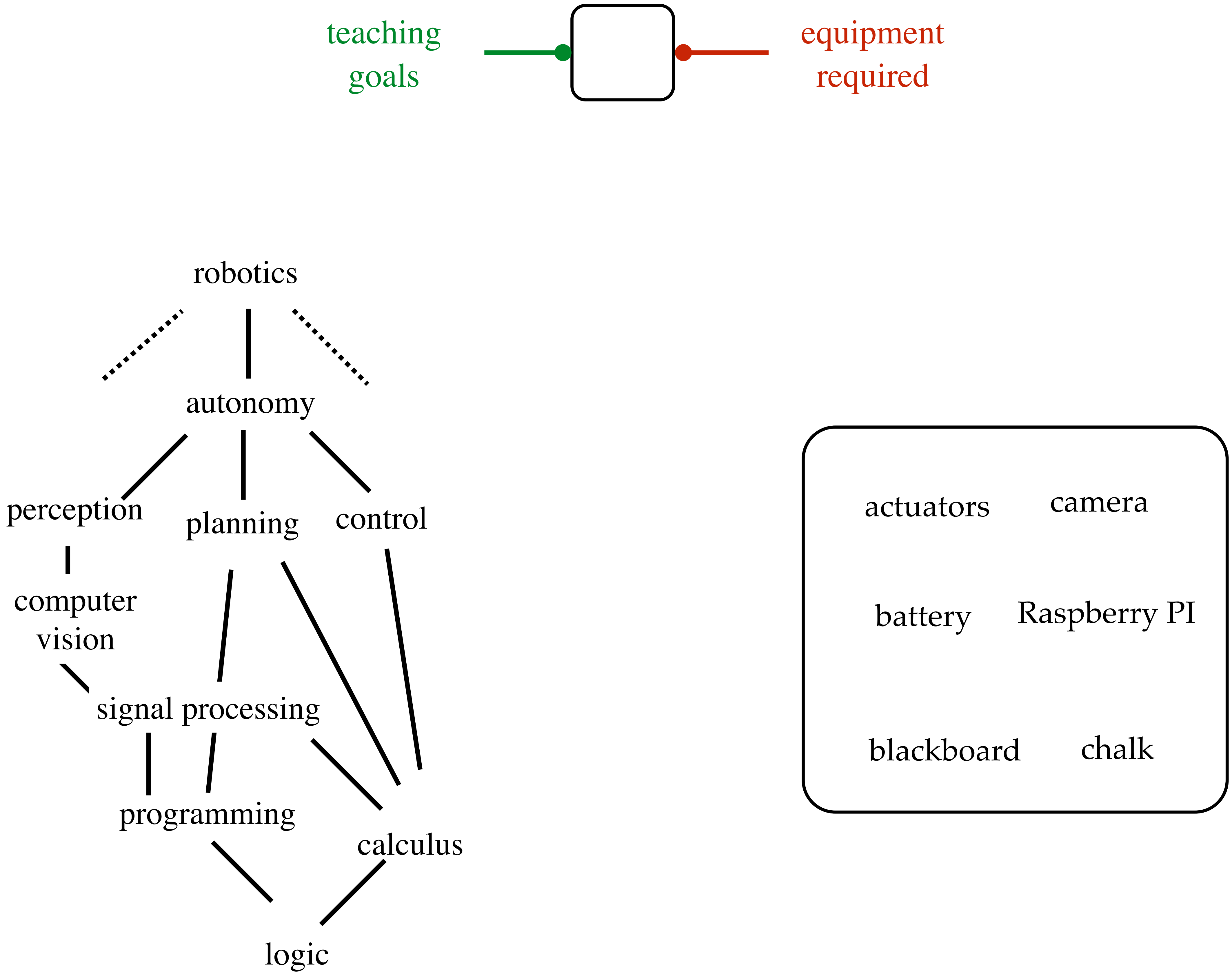
- **Definition.** A design problem is **monotone** if the map

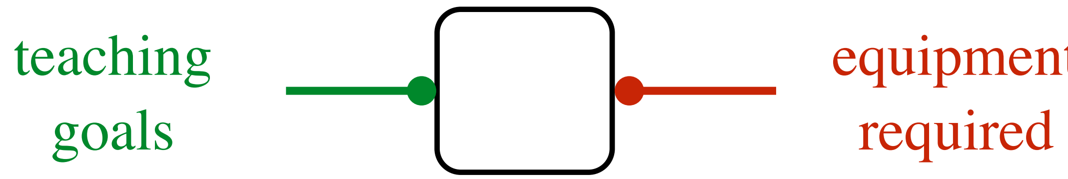
$$h : \mathcal{F} \to \text{antichains}(\mathcal{R})$$

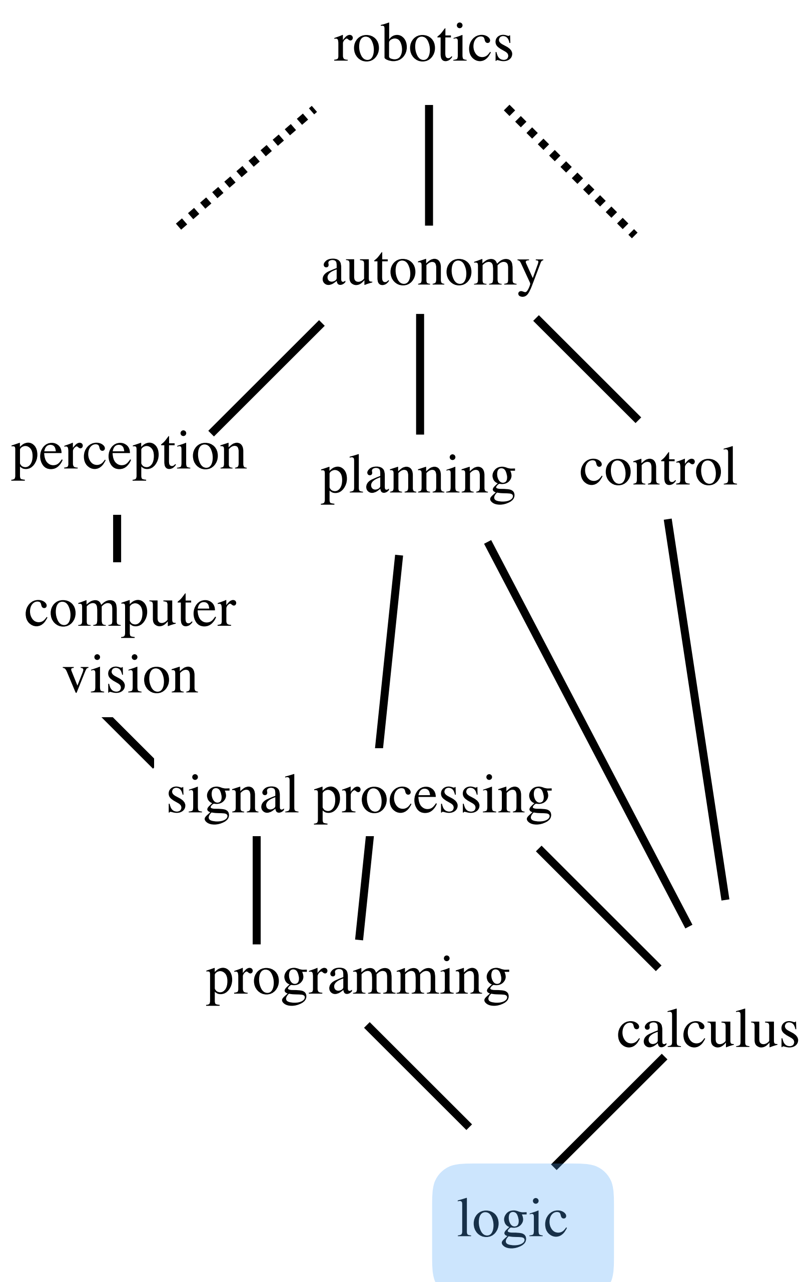
is monotone.

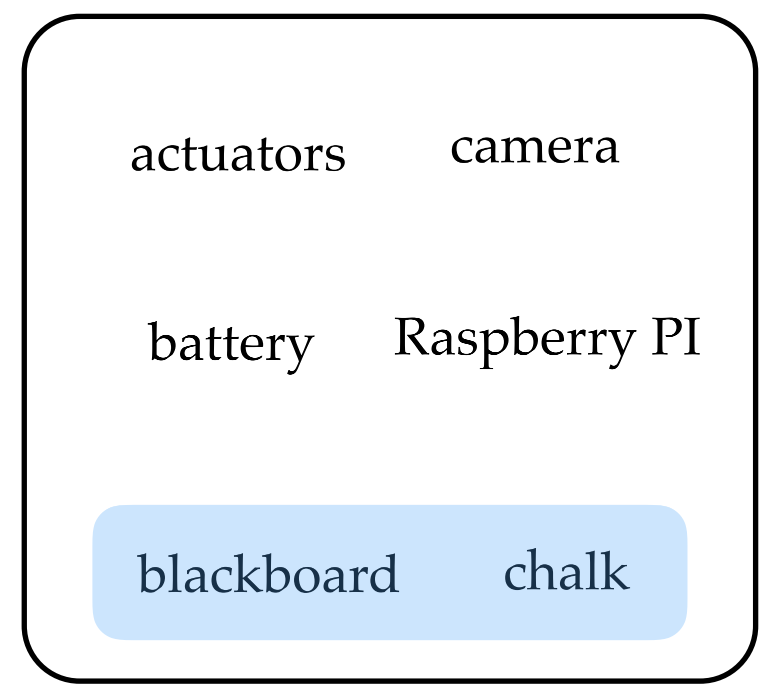


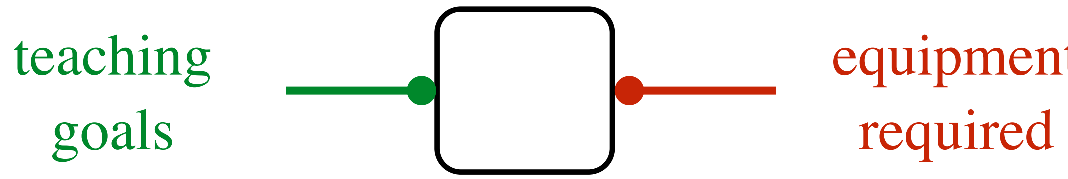


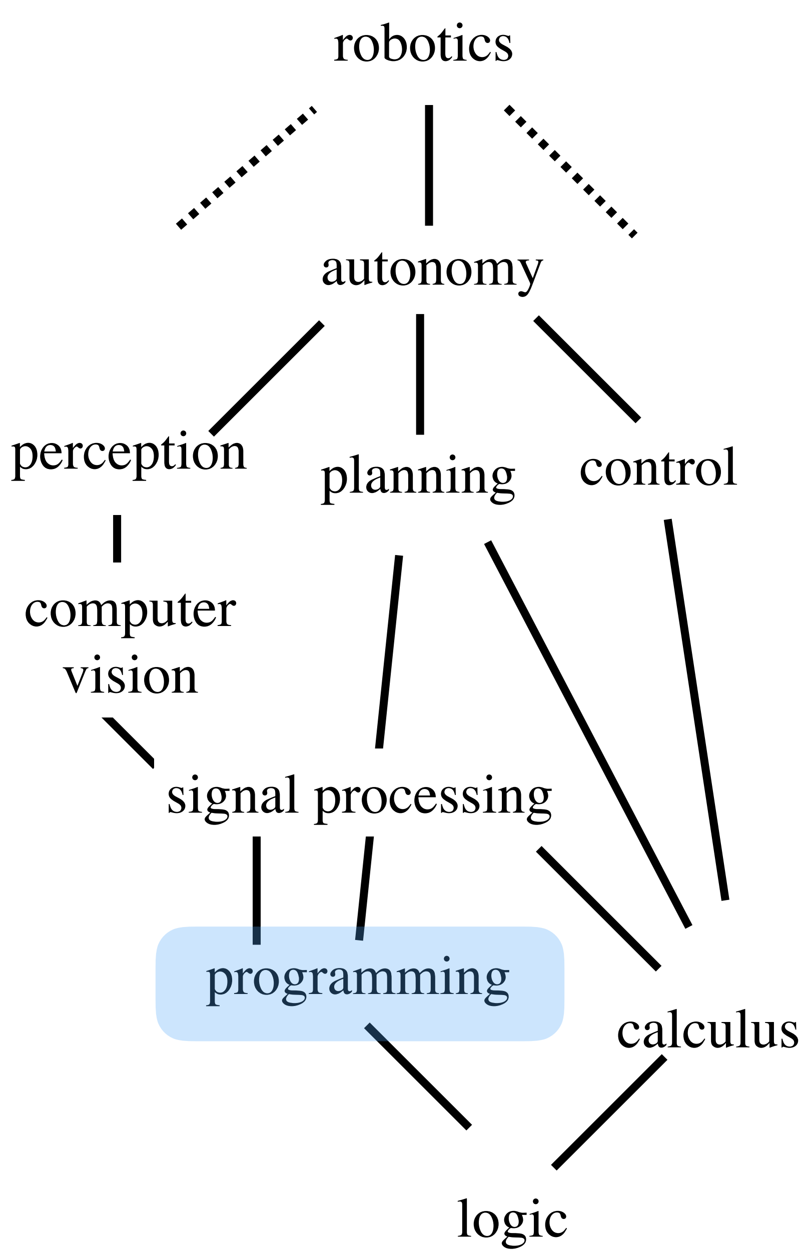

teaching goals
equipment required
robotics
autonomy
perception
planning
control
computer vision
signal processing
programming
calculus
logic
actuators
camera
battery
Raspberry PI
blackboard
chalk

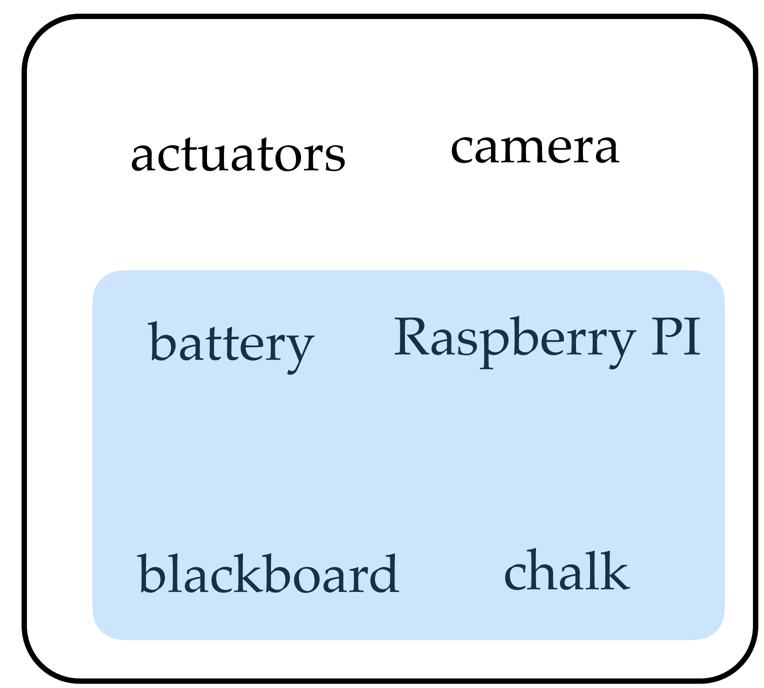

teaching
goals
equipment
required

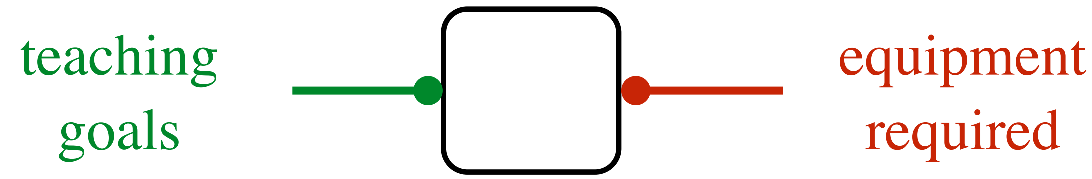

robotics
autonomy
perception
planning
control
computer
vision
signal processing
programming
calculus
logic

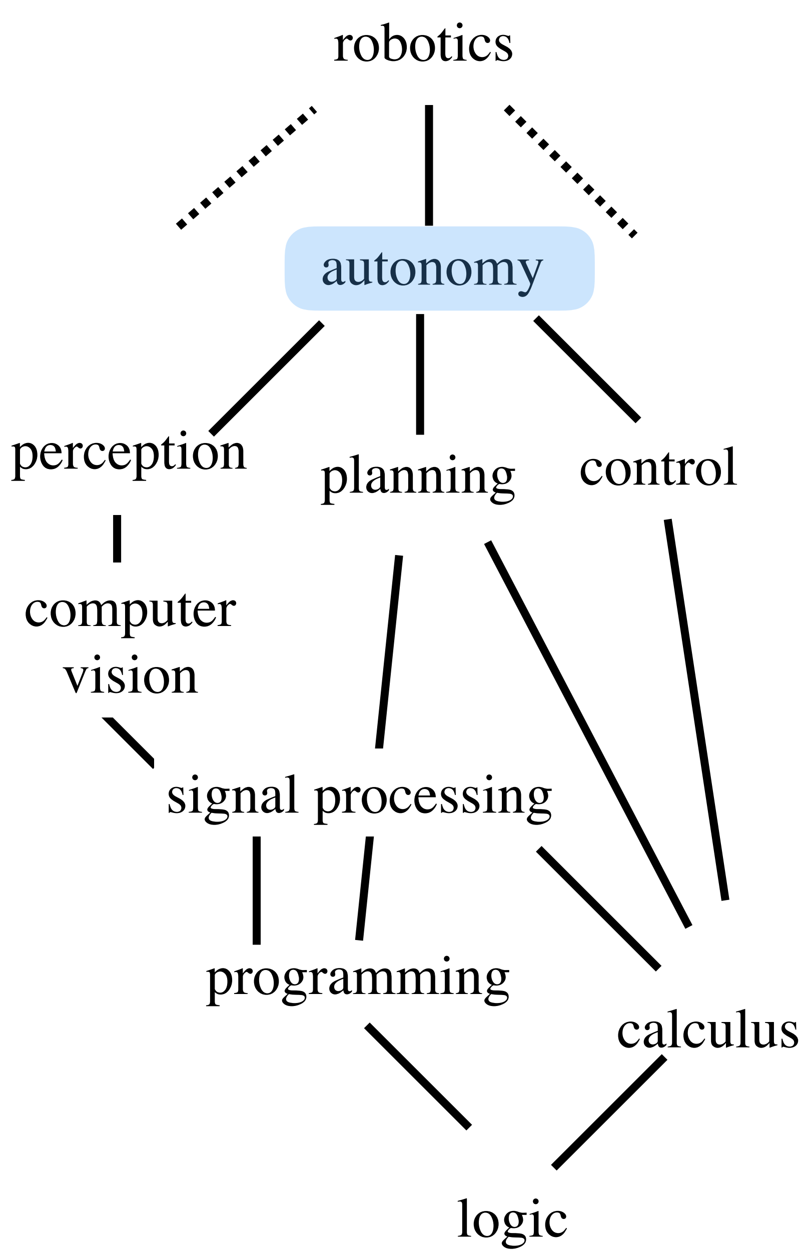

actuators
camera
battery
Raspberry PI
blackboard
chalk

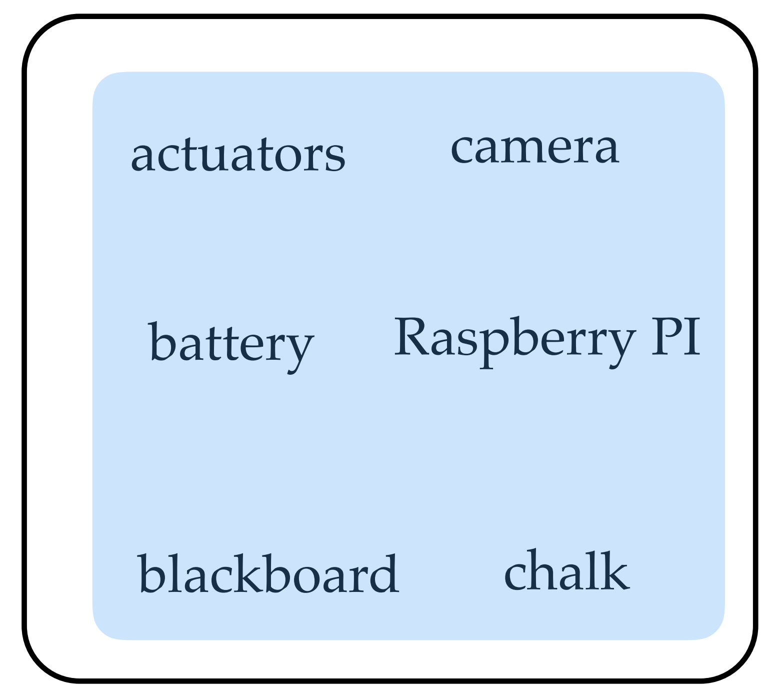

teaching
goals
equipment
required


robotics
autonomy
perception
planning
control
computer
vision
signal processing
programming
calculus
logic


actuators
camera
battery
Raspberry PI
blackboard
chalk

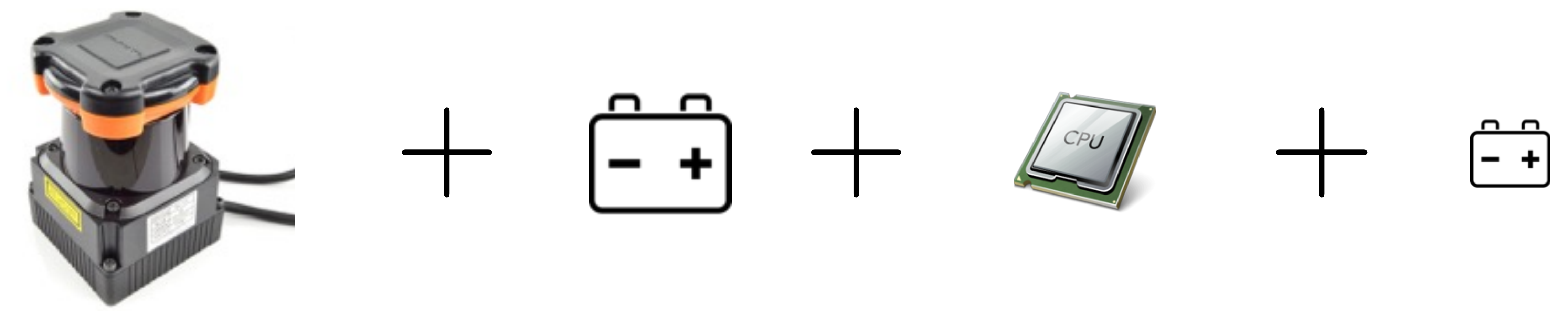

teaching
goals
equipment
required


robotics
autonomy
perception
planning
control
computer
vision
signal processing
programming
calculus
logic

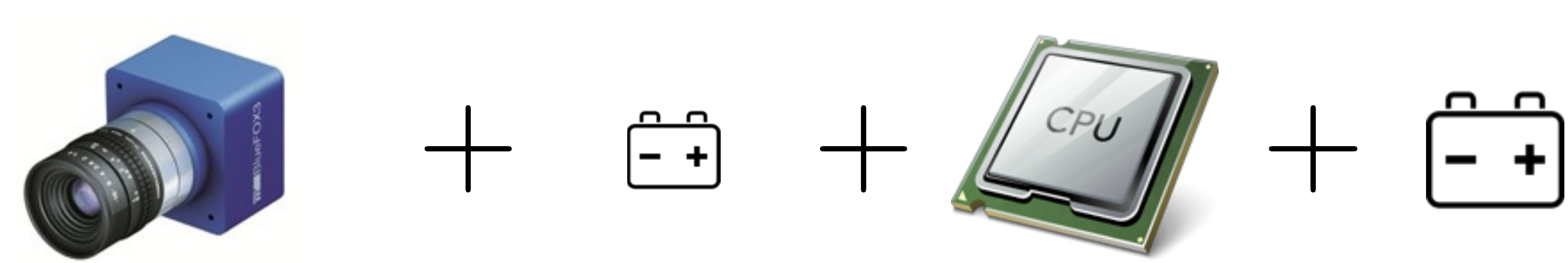

actuators
camera
battery
Raspberry PI
blackboard
chalk

Co-design constraints

range-finder



camera



Co-design constraints

chassis must
carry battery

chassis requires
motors to move

battery

chassis

motor

battery must power motor

budget must
be sufficient
for components

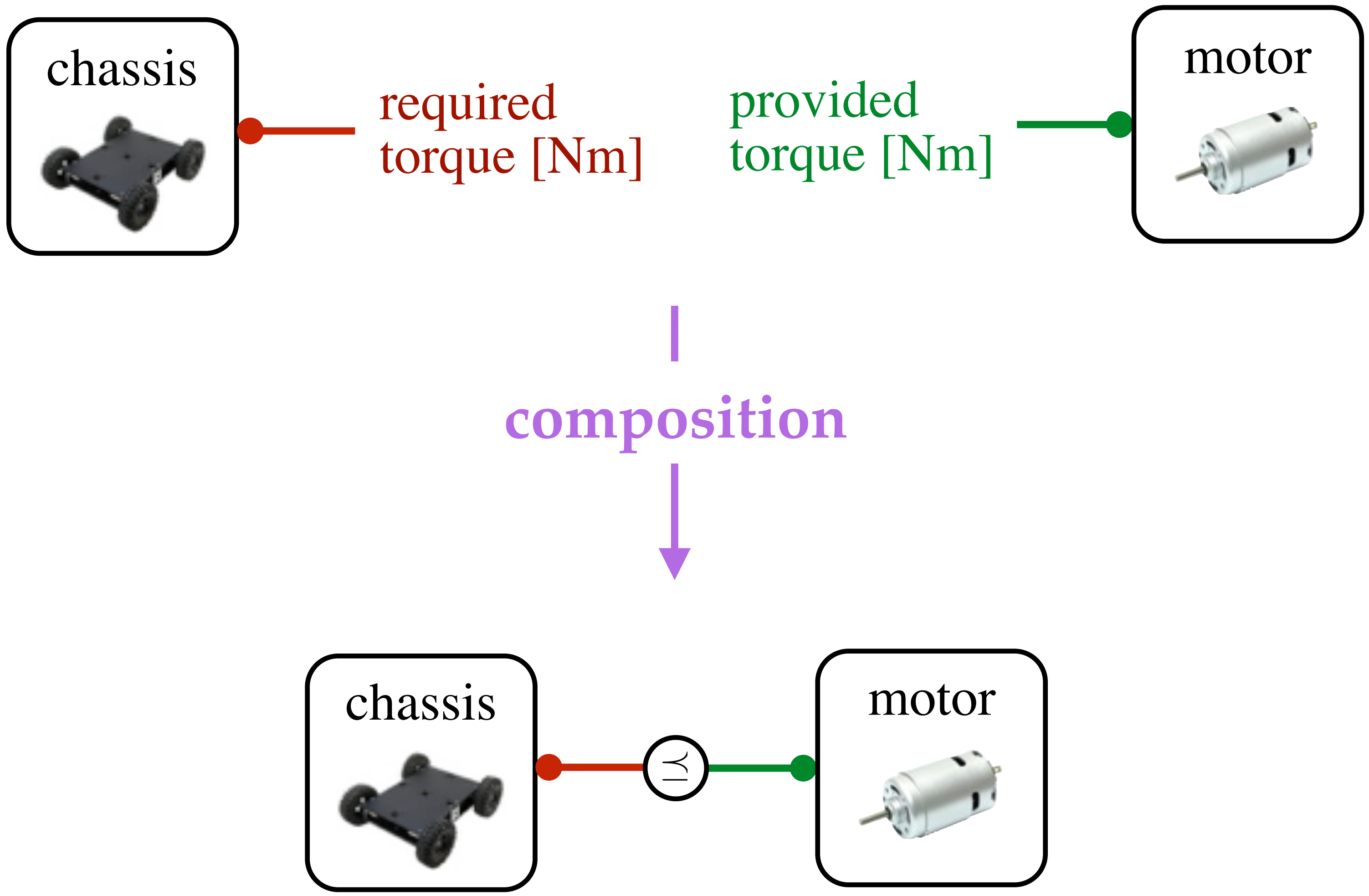
components

components must
support behaviors

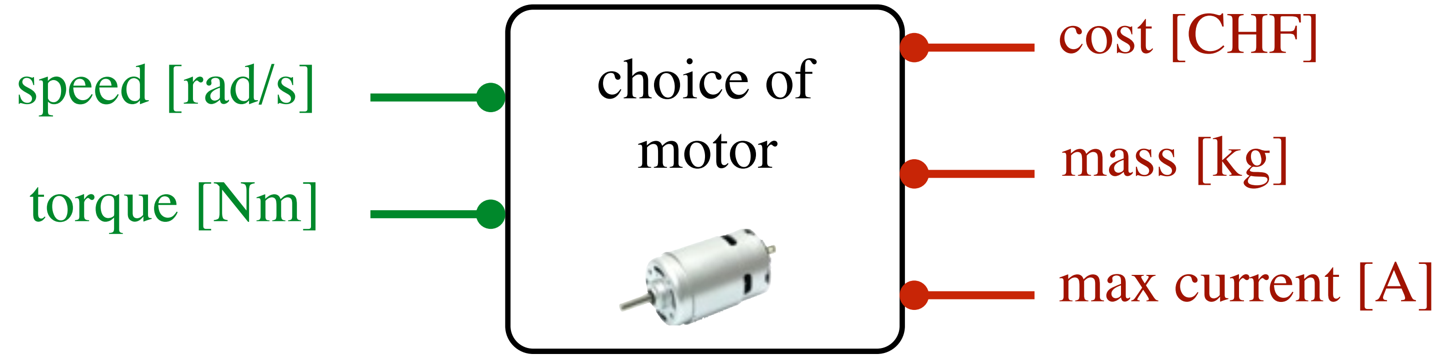
budget

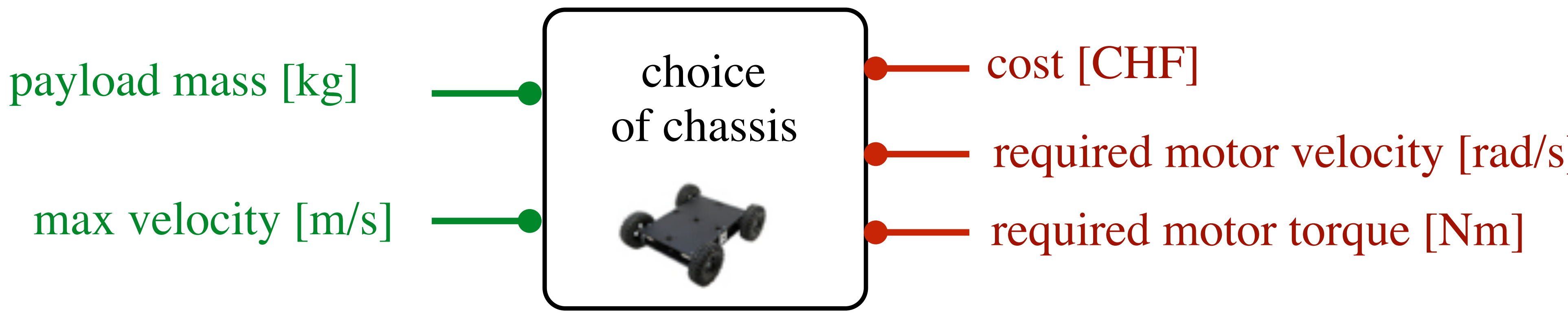
behaviors

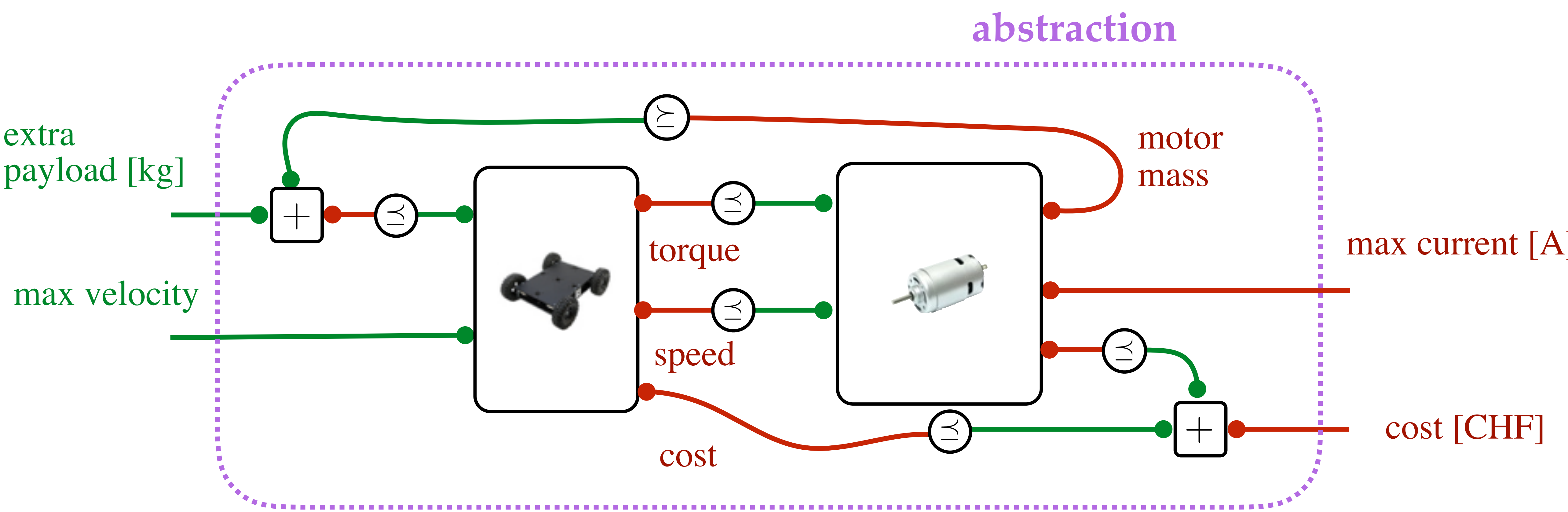
behaviors implemented
should justify the cost



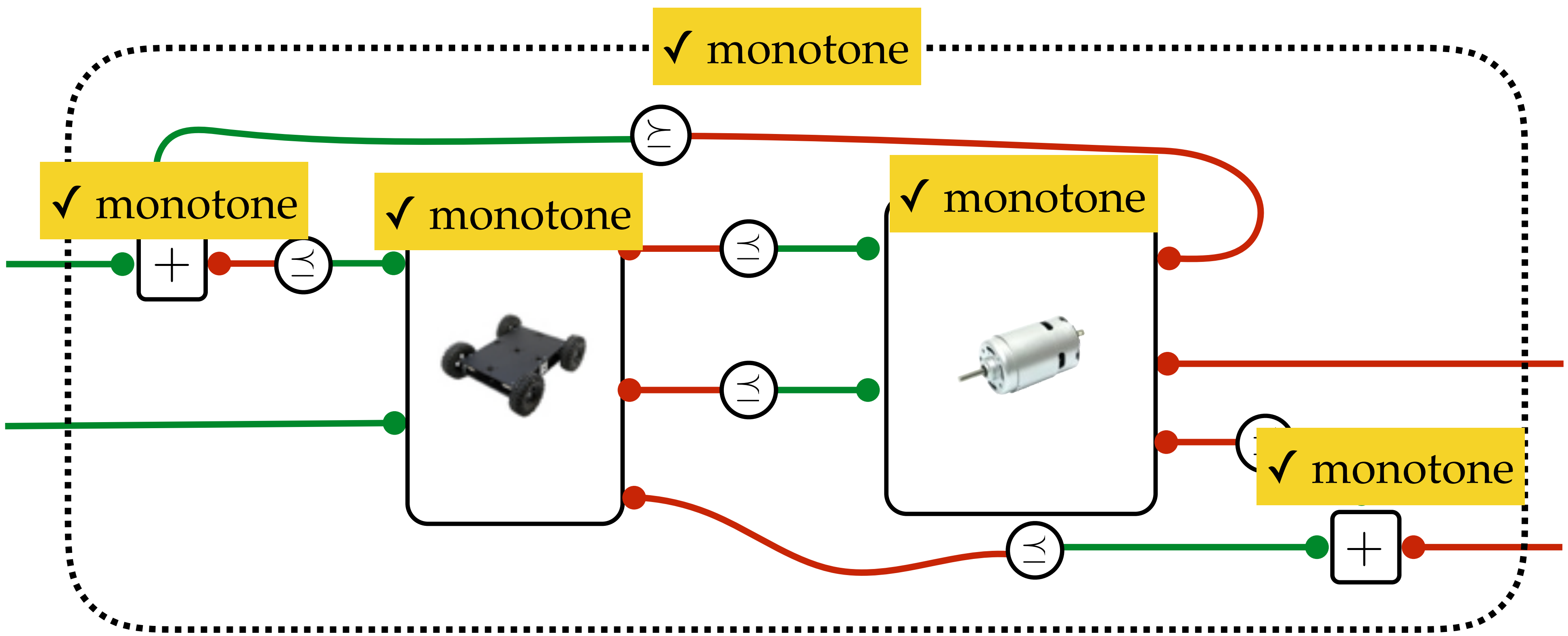
resources required by the first system $\preceq$ **functionality provided by the second system**


speed [rad/s]
torque [Nm]
choice of motor
cost [CHF]
mass [kg]
max current [A]


payload mass [kg]
max velocity [m/s]
choice of chassis
cost [CHF]
required motor velocity [rad/s]
required motor torque [Nm]

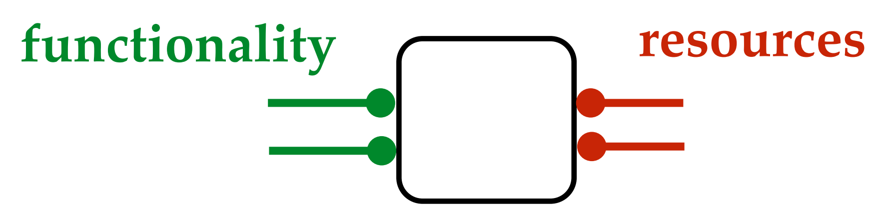

abstraction
extra payload [kg]
max velocity
motor mass
torque
speed
cost
max current [A]
cost [CHF]

- **Theorem.** The interconnection of any number of monotone design problems is monotone.

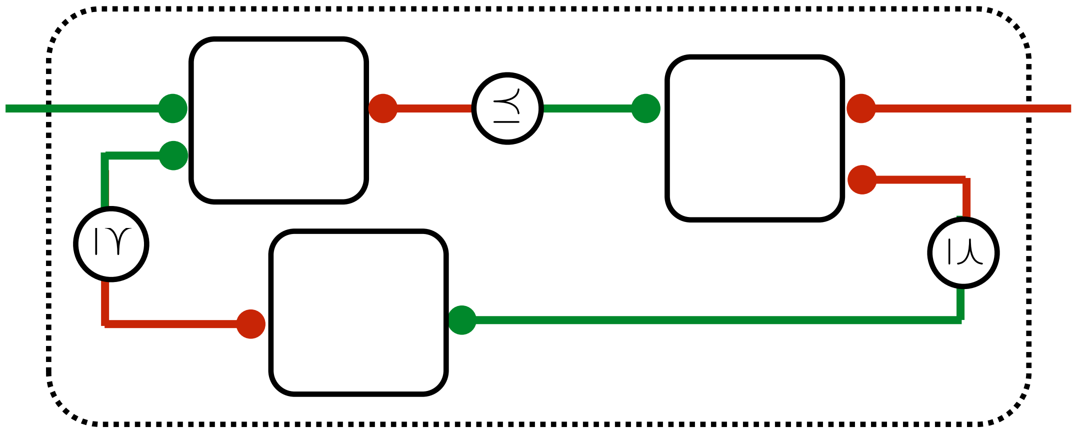


A mathematical theory of co-design

- ✓ Design problem = **monotone** relations between functionality and resources.



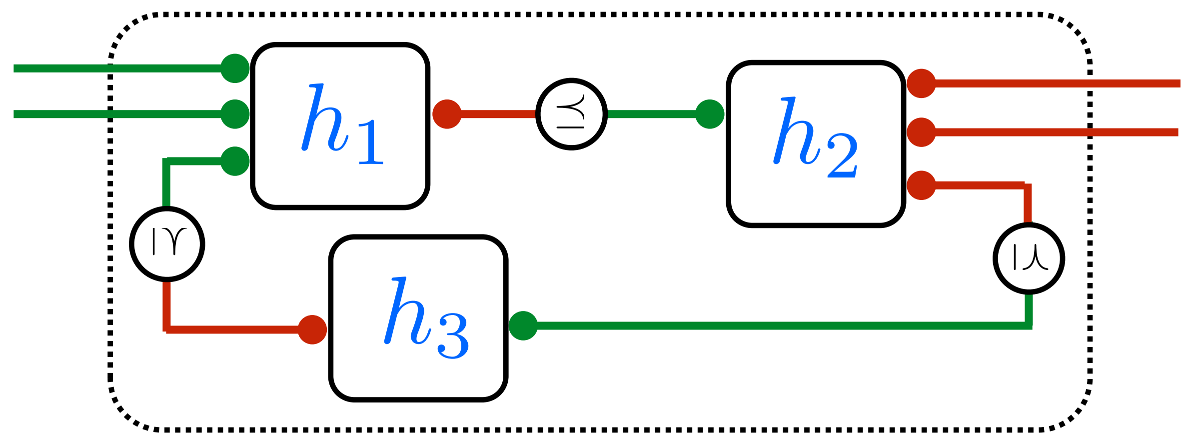
- ✓ Co-design problem = interconnection of design problems



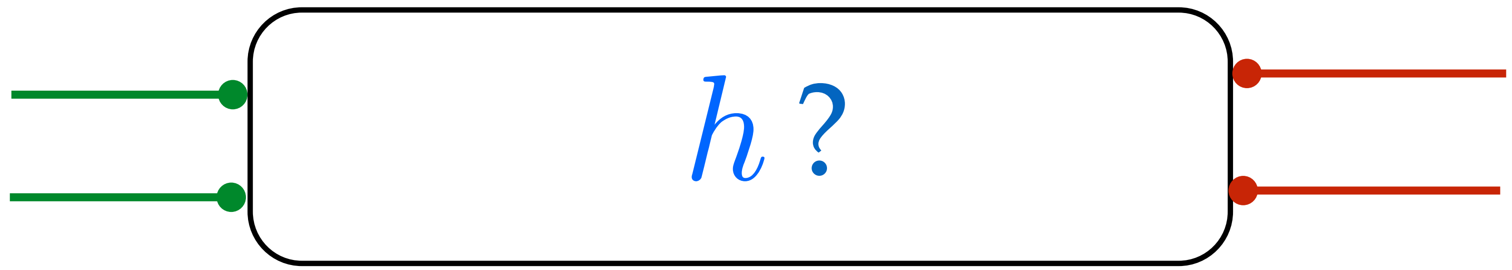
- ✓ Interconnection preserves monotonicity.

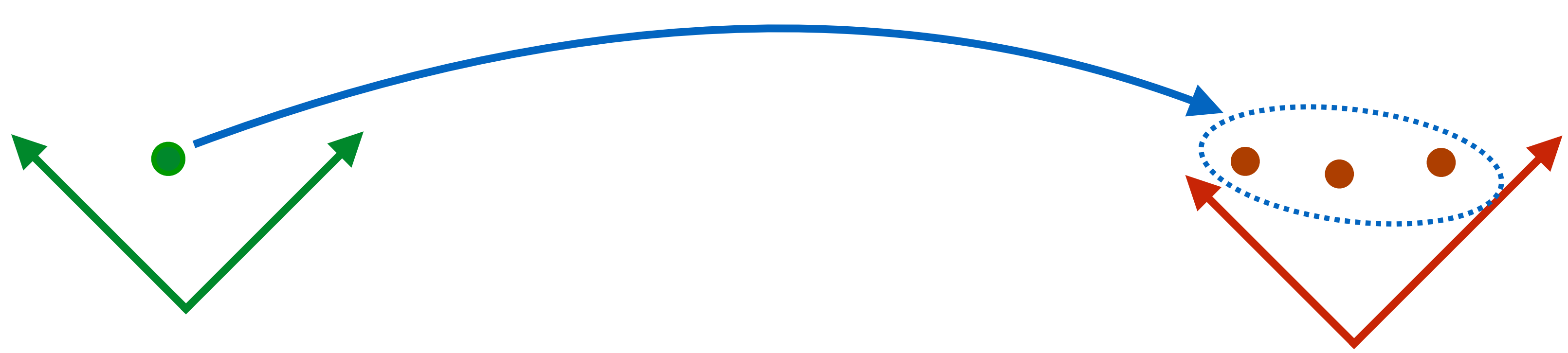
- Semantics as an optimization problem
- Solution techniques

Semantics as an optimization problem

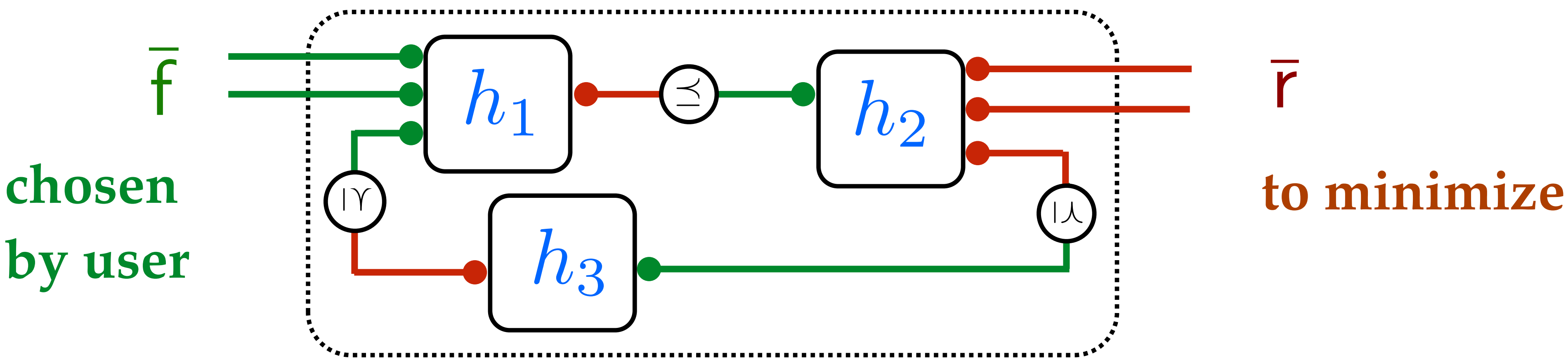


- **Assuming we know the maps *h*** for the subproblems,
- …we need to **evaluate the map *h* for the entire graph.**





Semantics as an optimization problem

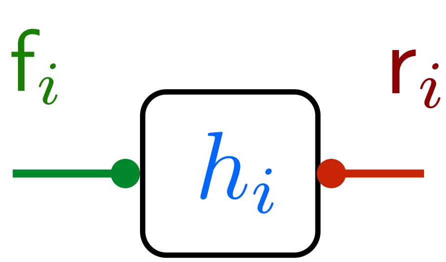


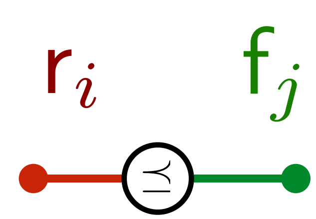
parameters $\bar{\mathsf{f}}$

variables $\{(\mathsf{f}_i, \mathsf{r}_i)\},\ i = 1, \ldots, n$

objective $\underset{\preceq}{\mathrm{Min}}\ \bar{\mathsf{r}}$

constraints *for each node i:* *for each edge (i, j):*





$\mathsf{r}_i \in h_i(\mathsf{f}_i)$ $\qquad$ $\mathsf{f}_j \succeq \mathsf{r}_i$

! not convex

! not differentiable

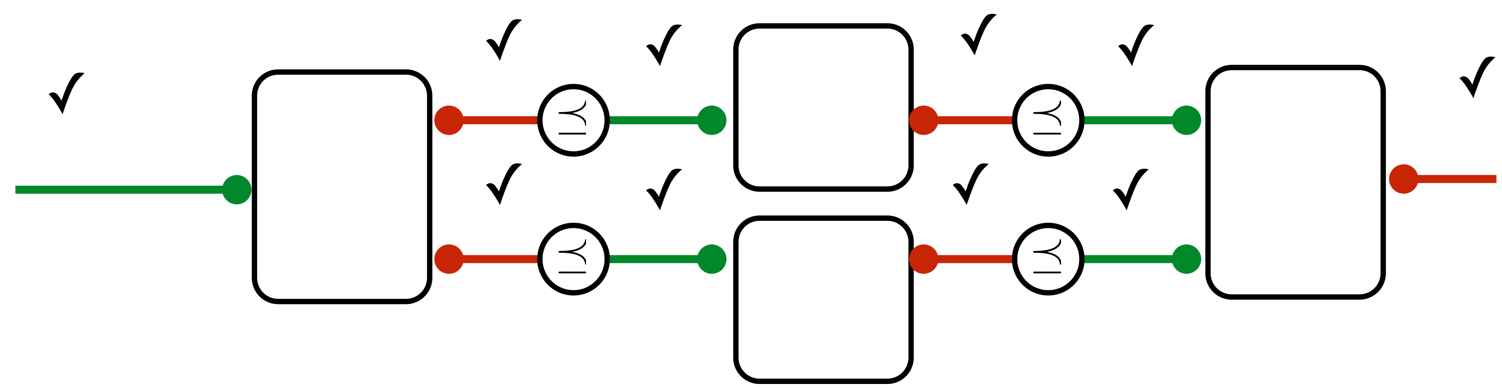
! not continuous

! not even defined on continuous spaces

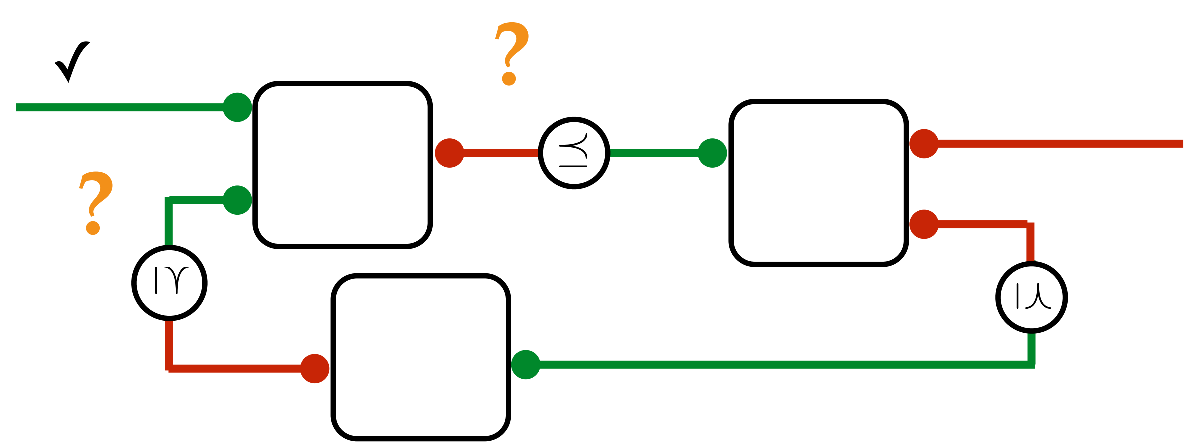
- **The problem is loops** (“feedback”)

- **The problem is loops** (“feedback”)

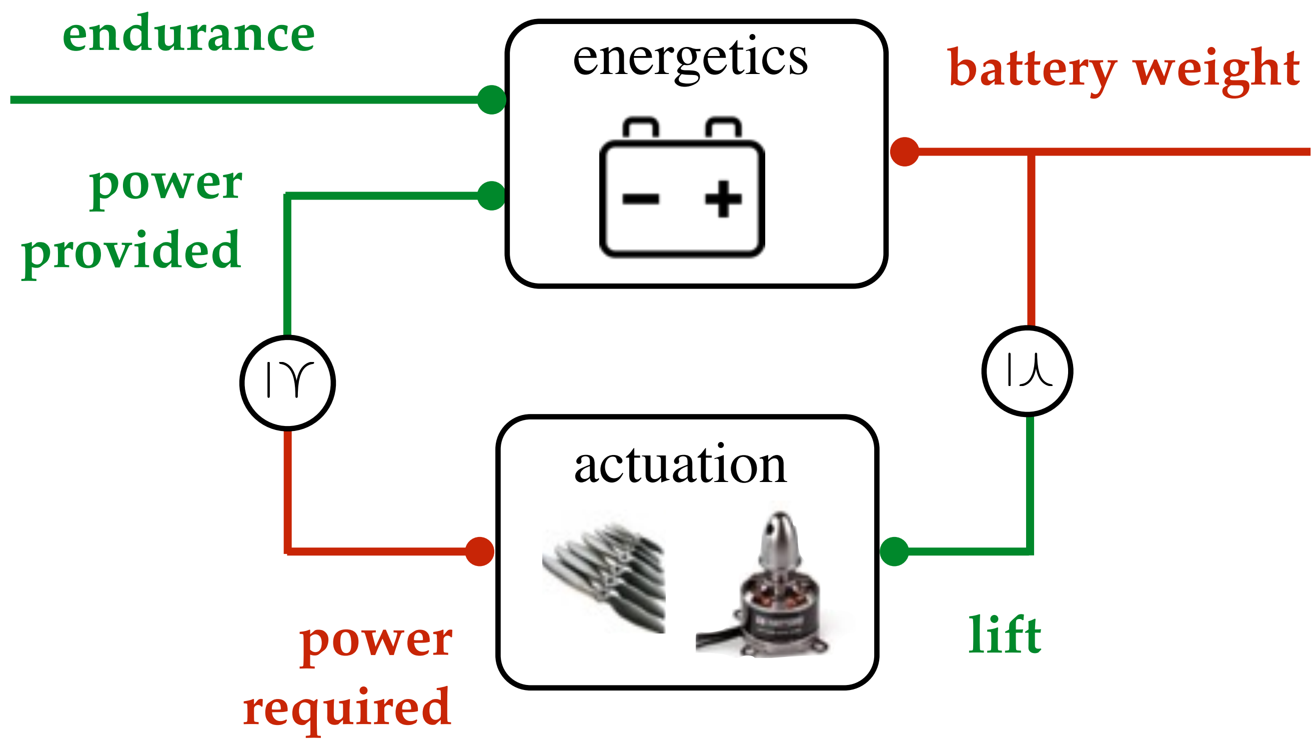
without loops:



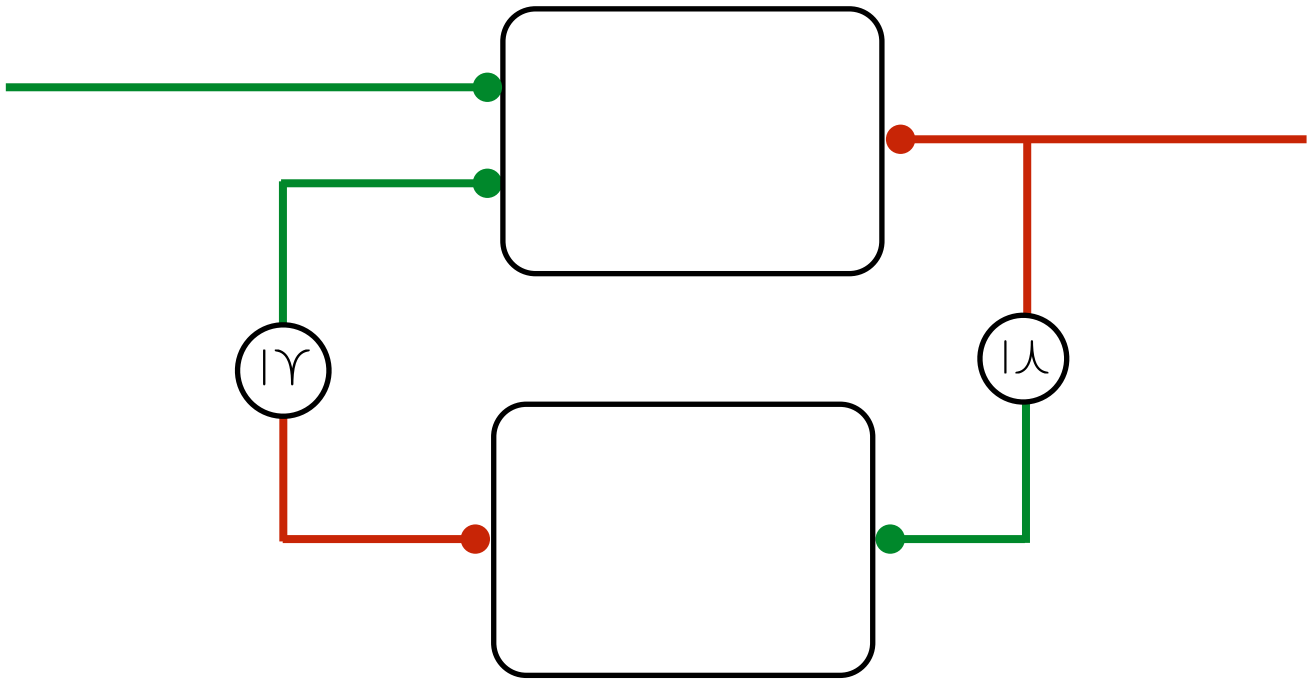
with loops:



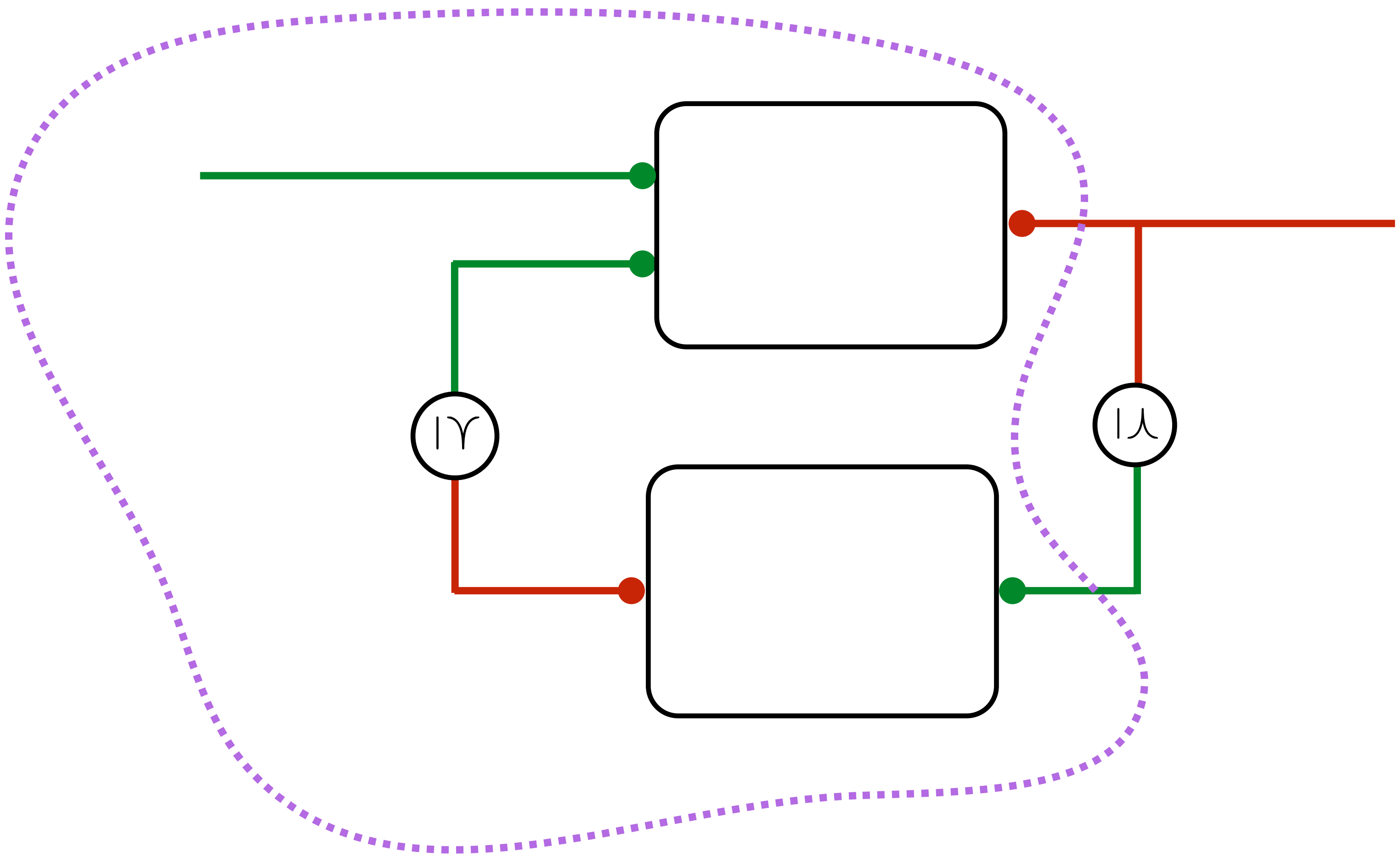
- Consider the case of **1 loop, with a scalar variable.**



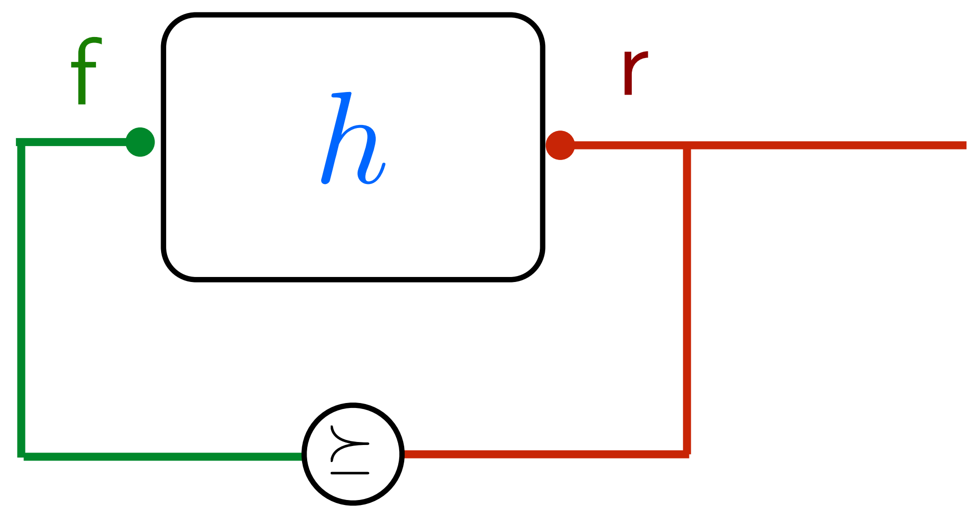
- Consider the case of **1 loop, with a scalar variable.**



- Consider the case of **1 loop, with a scalar variable.**



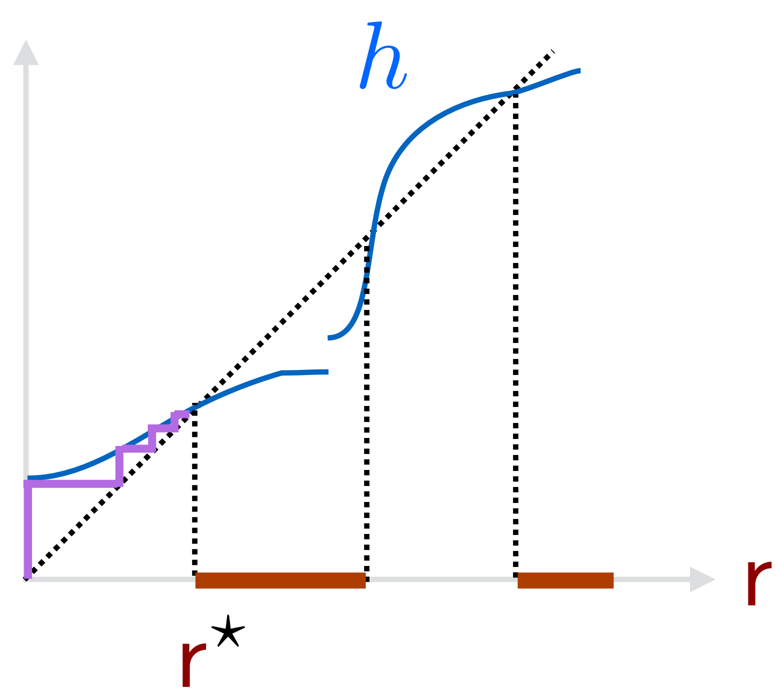
- Consider the case of **1 loop, with a scalar variable.**



"least fixed point"

$$\min \mathsf{r}$$
$$\mathsf{f} \geq \mathsf{r}$$
$$\mathsf{r} = h(\mathsf{f})$$

$$\min \mathsf{r}$$
$$\mathsf{r} \geq h(\mathsf{r})$$



"Kleene's algorithm"

$$\mathsf{r}_0 = 0$$
$$\mathsf{r}_{k+1} = h(\mathsf{r}_k)$$
$$\mathsf{r}^\star = \lim_{k \to \infty} \mathsf{r}_k$$

Banach fixed-point theorem

If:

- $\langle \mathcal{X}, d \rangle$ complete metric space
- $f : \mathcal{X} \to \mathcal{X}$ contraction

$$d(f(x), f(y)) \leq c\, d(x,y), \quad 0 \leq c < 1$$

Then:

- $\exists!\, \overline{x} : \overline{x} = f(\overline{x})$

- $\overline{x} = \lim_{n\to\infty} f^n(x_0), \quad \forall x_0$

Kleene fixed-point theorem(s)

(Knaster-Tarski)

If:

- $\langle \mathcal{P}, \preceq \rangle$ complete partial order
- $f : \mathcal{P} \to \mathcal{P}$ monotone

$$x \preceq y \;\Rightarrow\; f(x) \preceq f(y)$$

Then:

- $\exists\, \mathsf{lfp}(f) = \min_{\preceq} \{x \mid f(x) = x\}$

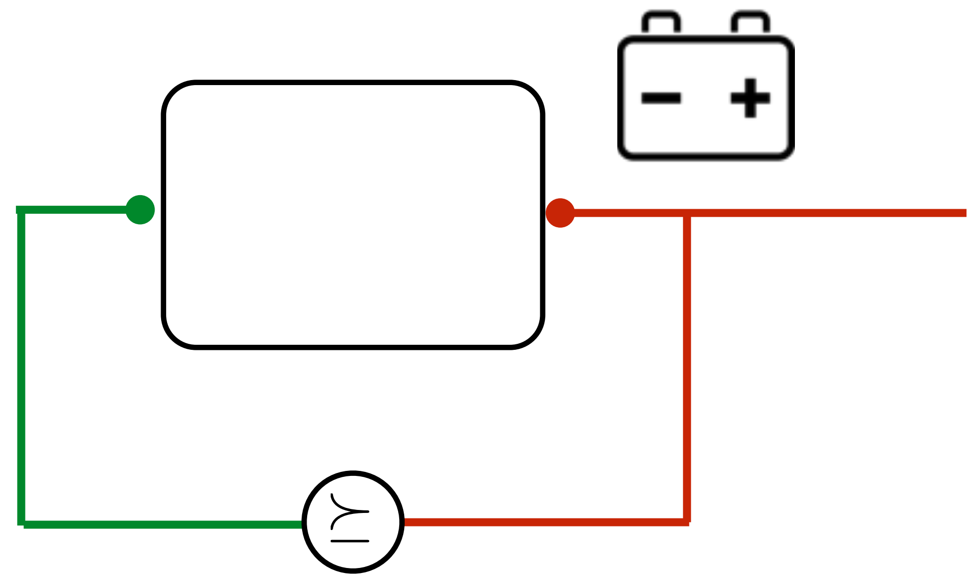
If also:

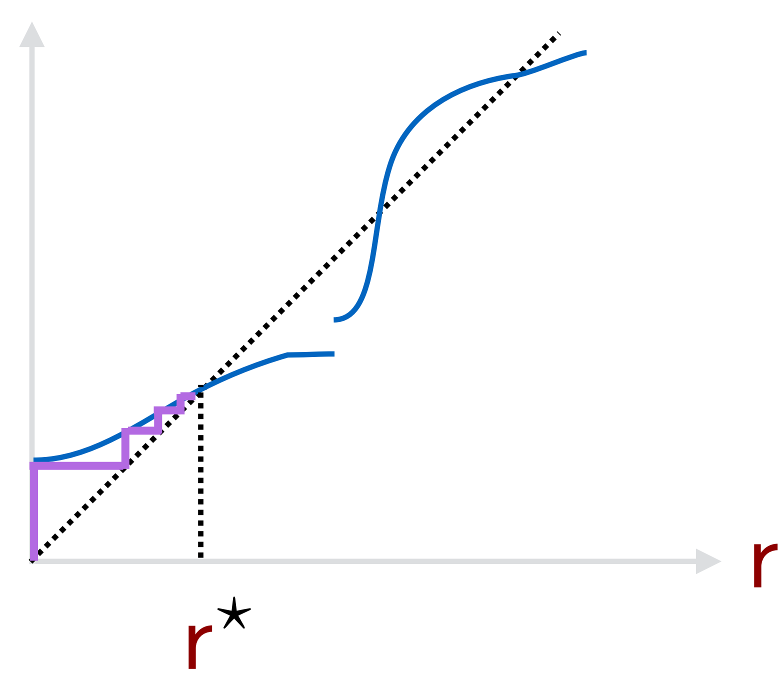
- f Scott-continuous

Then:

- $\mathsf{lfp}(f) = \lim_{n\to\infty} f^n(\perp)$

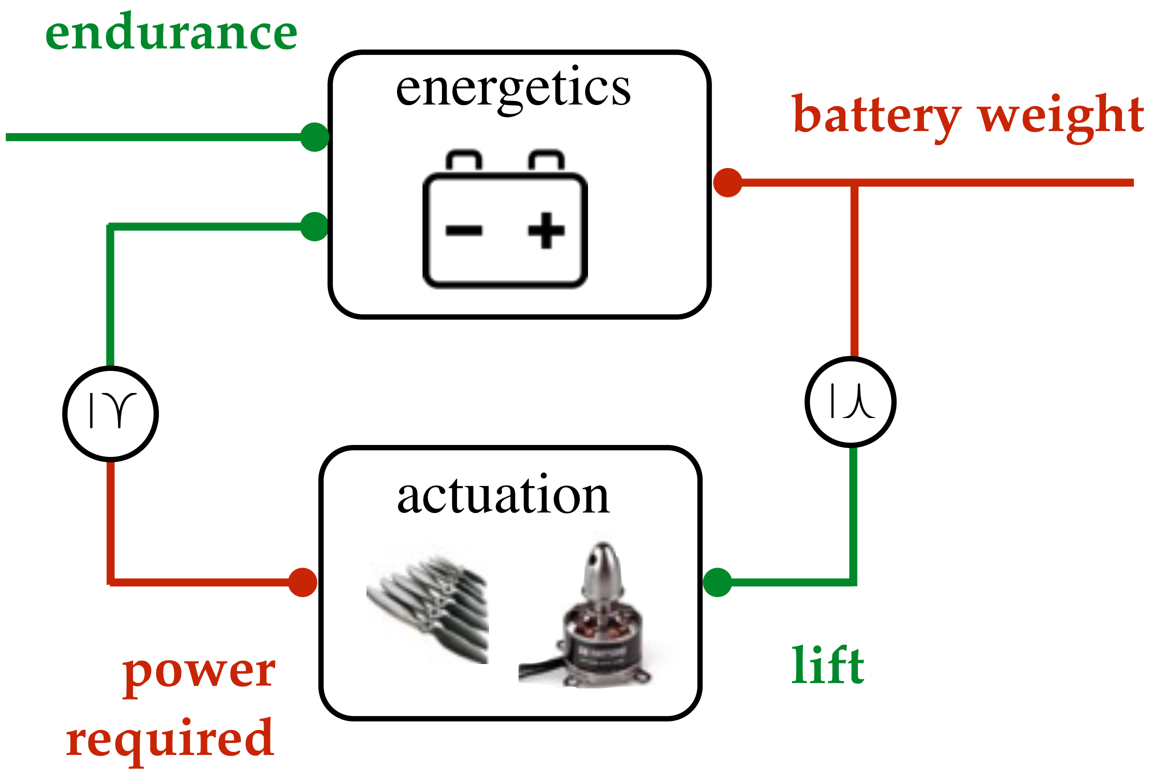
- "Start with no resources; increase as needed".



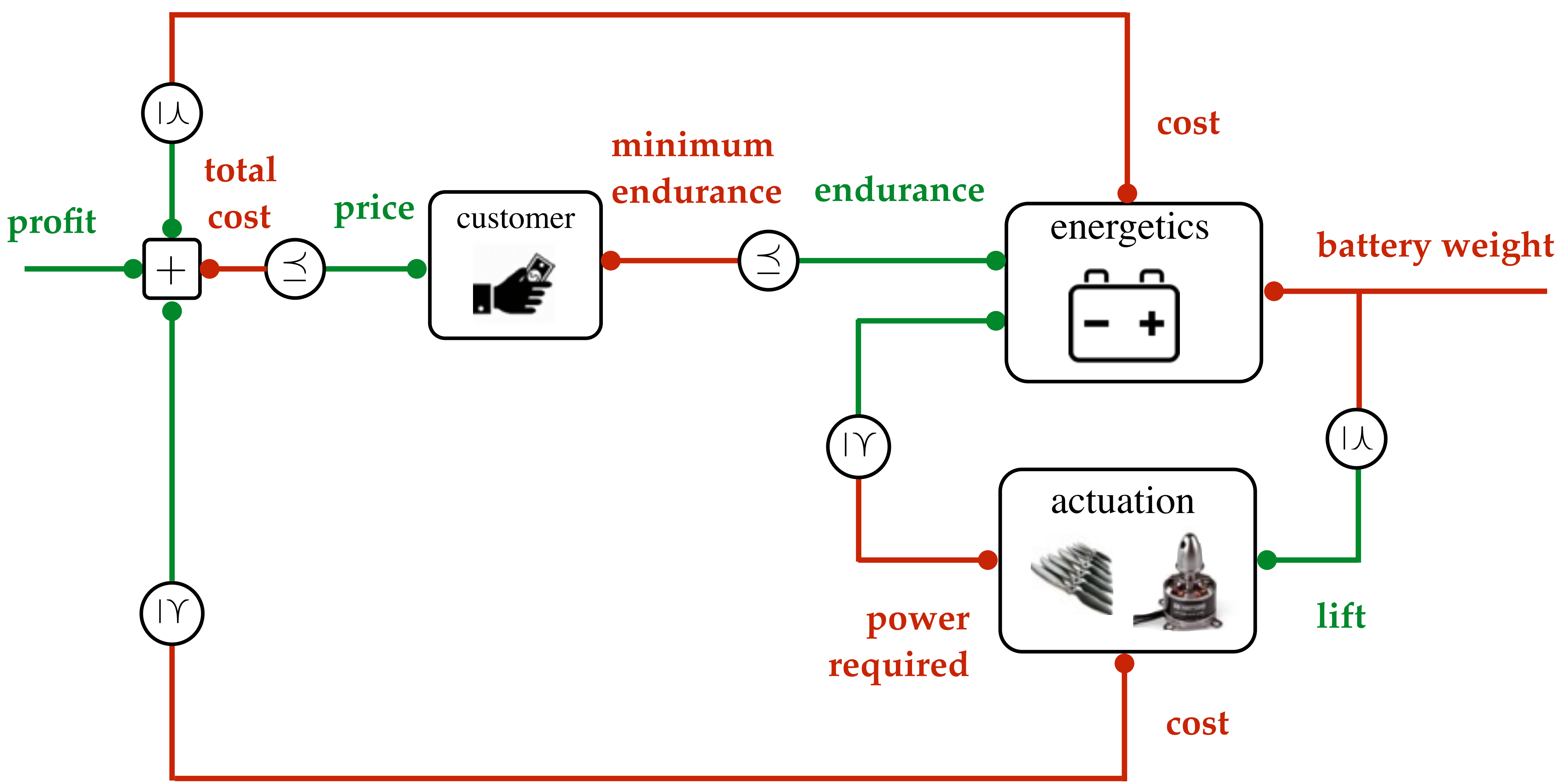


- Everything generalizes to the case with **multiple cycles**.

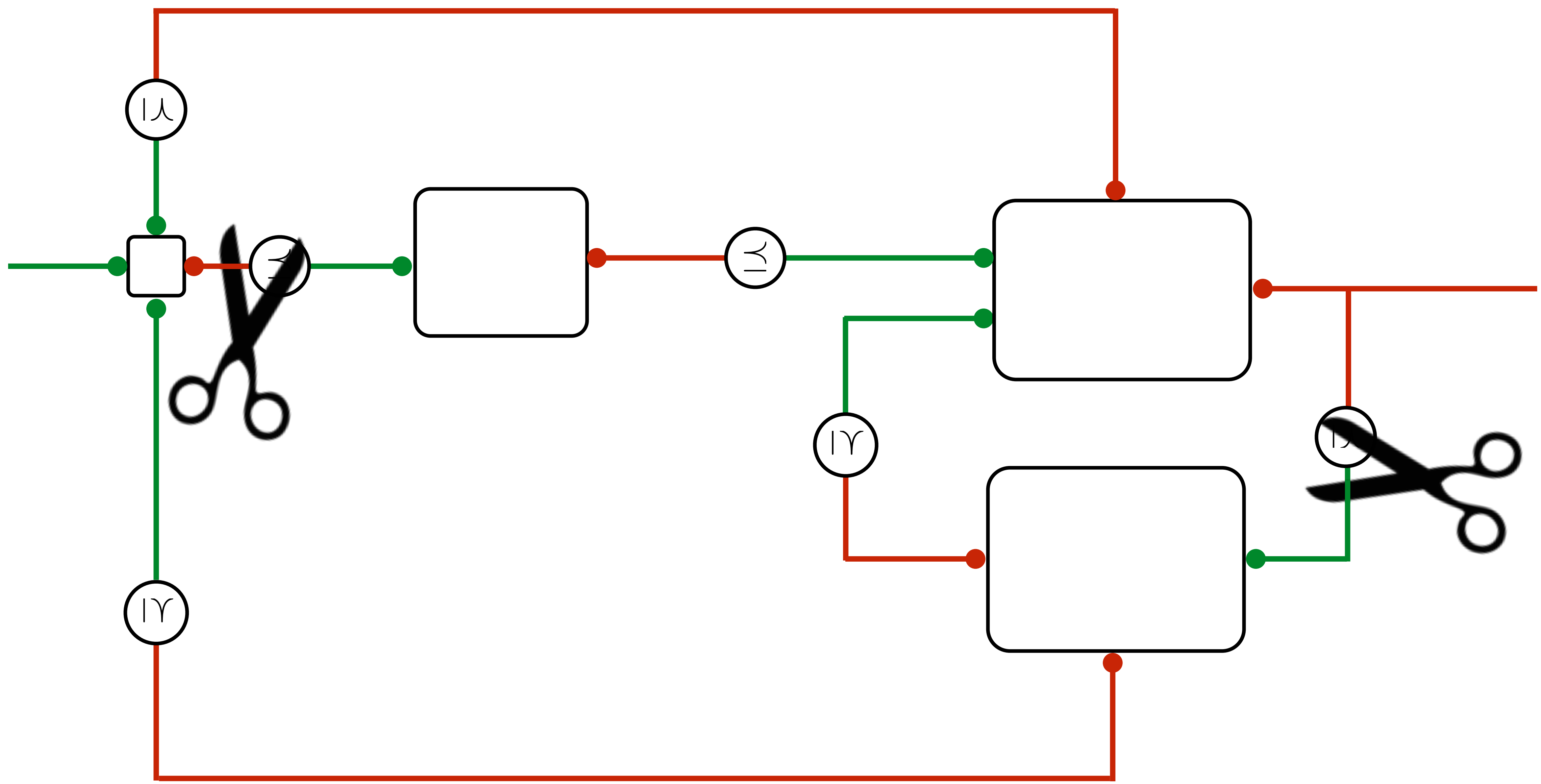
- Everything generalizes to the case with **multiple cycles**.

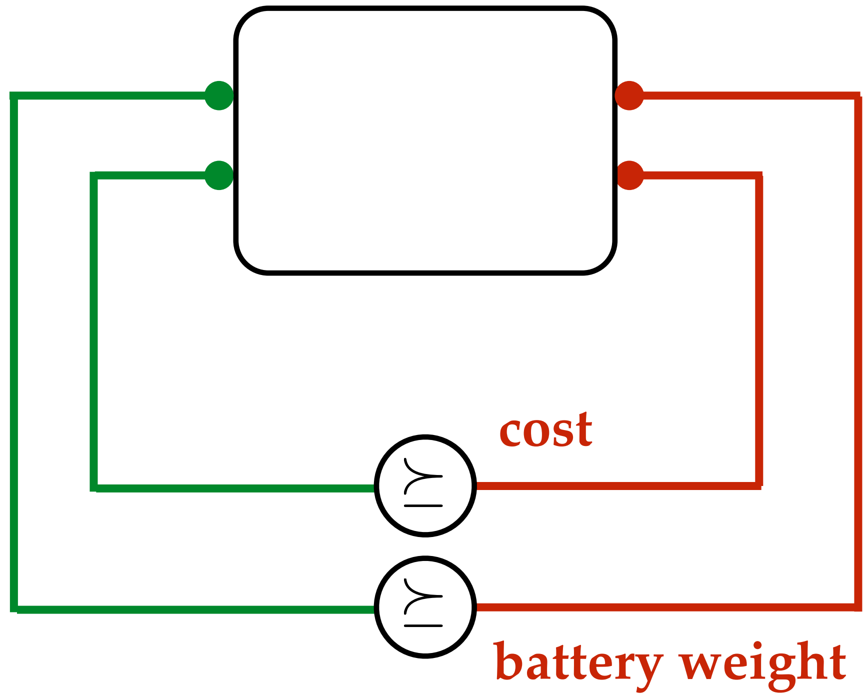


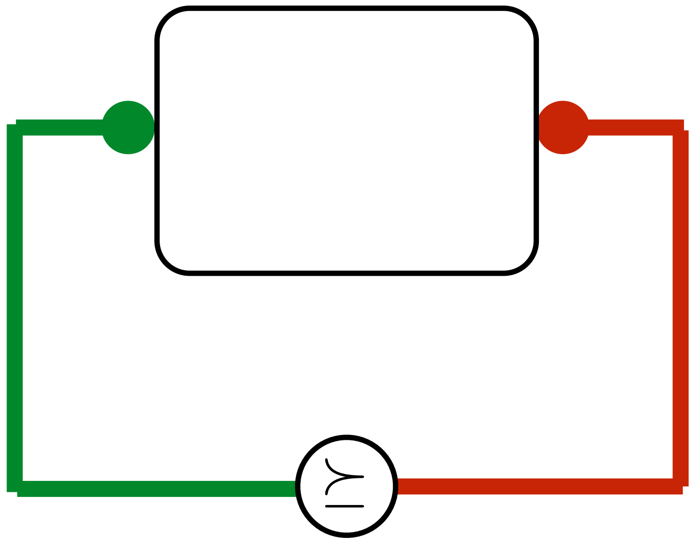
- Everything generalizes to the case with **multiple cycles**.



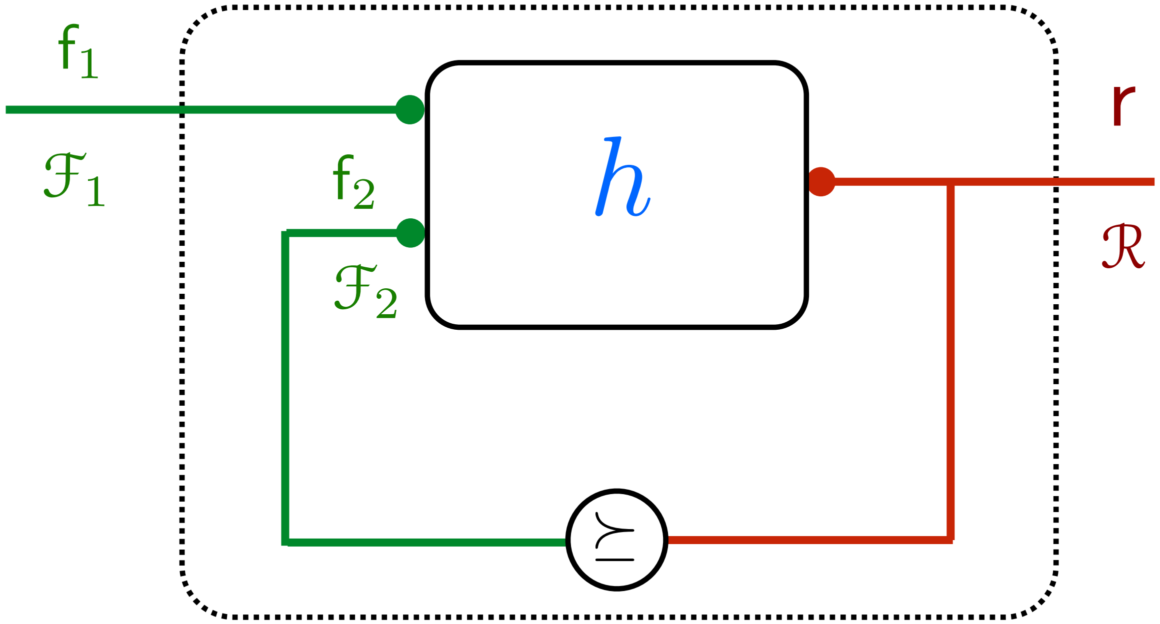
- **Removing 2 edges removes all oriented cycles.**
  (“minimal feedback arc set”)



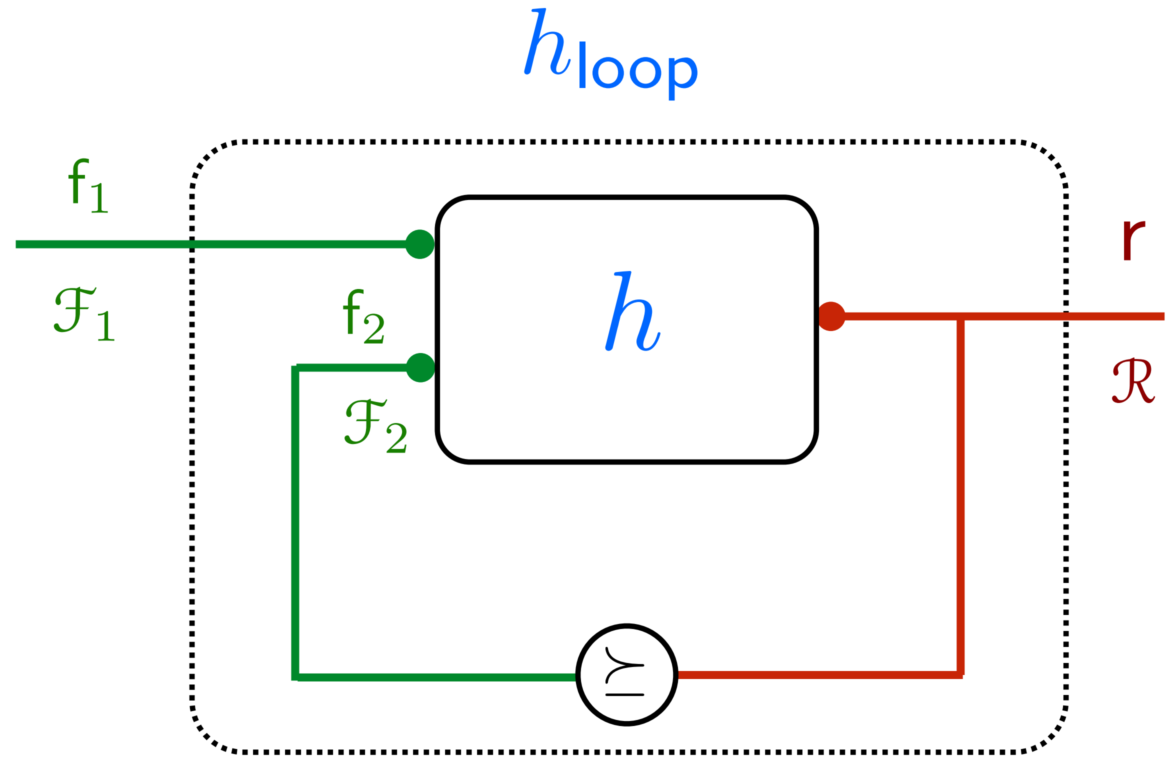

cost
battery weight

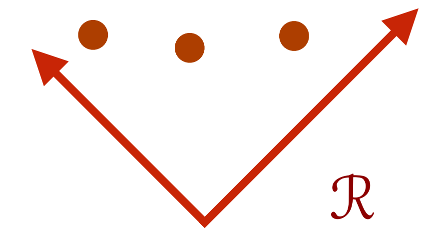


cost × battery weight



f_1
$\mathcal{F}_1$
f_2
$\mathcal{F}_2$
h
r
$\mathcal{R}$
$\succeq$

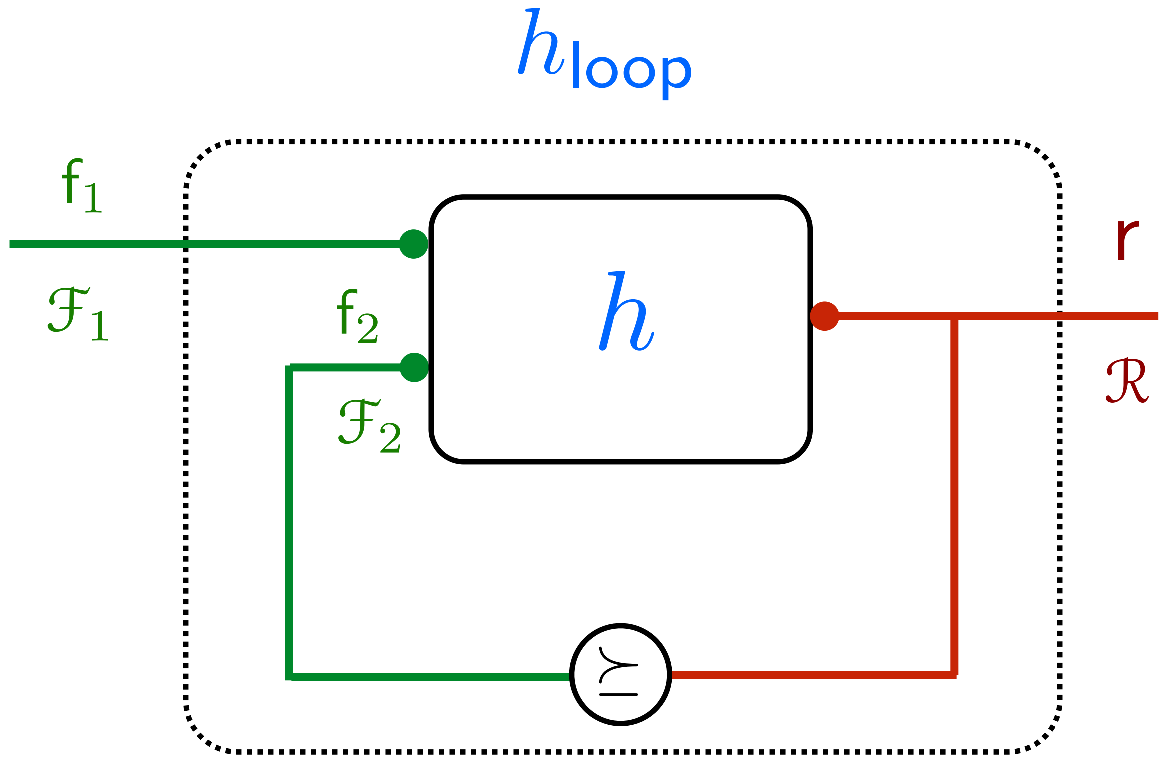


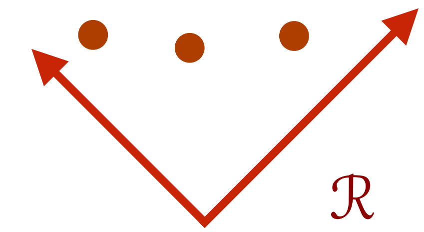


Theorem. The **set of minimal feasible resources** can be obtained as the least fixed point of a monotone function in the space of antichains.

$$\begin{aligned} h_{\text{loop}} : \mathcal{F}_1 &\rightarrow \text{antichains}(\mathcal{R}) \\ \mathsf{f}_1 &\mapsto \text{least-fixed-point}(\Phi_{\mathsf{f}_1}) \end{aligned}$$

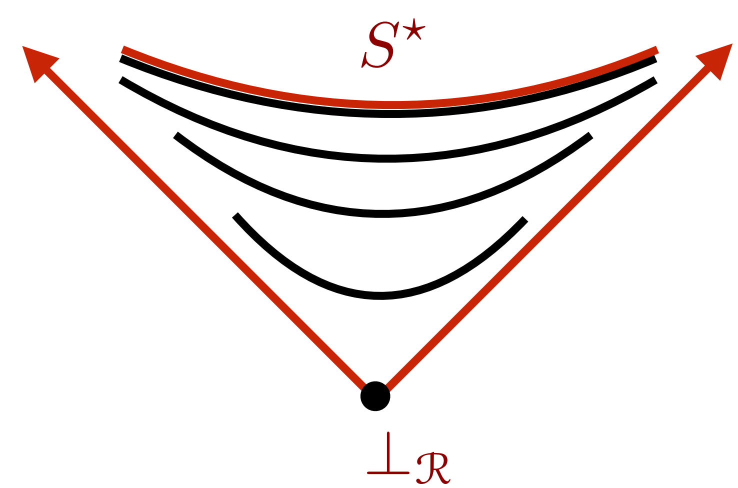
$$\begin{aligned} \Phi_{\mathsf{f}_1} : \text{antichains}(\mathcal{R}) &\rightarrow \text{antichains}(\mathcal{R}) \\ S &\mapsto \operatorname*{Min}_{\preceq_{\mathcal{R}}} \bigcup_{\mathsf{r} \in S} h(\mathsf{f}_1, \mathsf{r}) \cap \uparrow \mathsf{r} \end{aligned}$$





Corollary. The set of minimal solutions can be found using Kleene's algorithm.

$$S \subset \text{antichains}(\mathcal{R})$$
$$S_0 = \{\perp_{\mathcal{R}}\}$$
$$S_{k+1} = \Phi_{\mathsf{f}_1}(S_k)$$

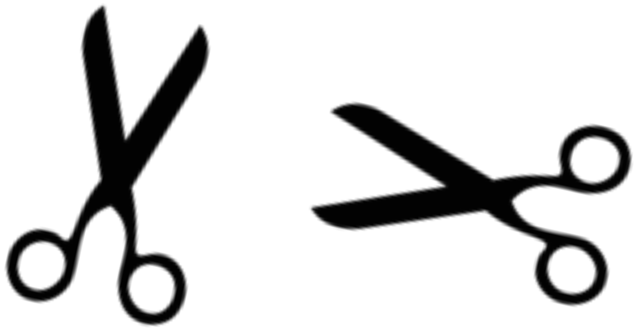


If the iteration diverges, it is a certificate of infeasibility.

✓ There exists a **systematic procedure** to solve MCDPs
that finds **all minimal solutions**, or a **certificate of infeasibility**.

- Surprising because not convex (or differentiable or continuous).

- Performance depends on the structure of the graph.
  - "thickness" of the edges that must be removed.



storage: $\mathrm{width}(\mathcal{R})$

computation: $\mathrm{height}(\mathrm{A}\mathcal{R}) \times (\mathrm{width}(\mathcal{R}))^2$

number of steps: $\mathrm{height}(\mathrm{A}\mathcal{R})$

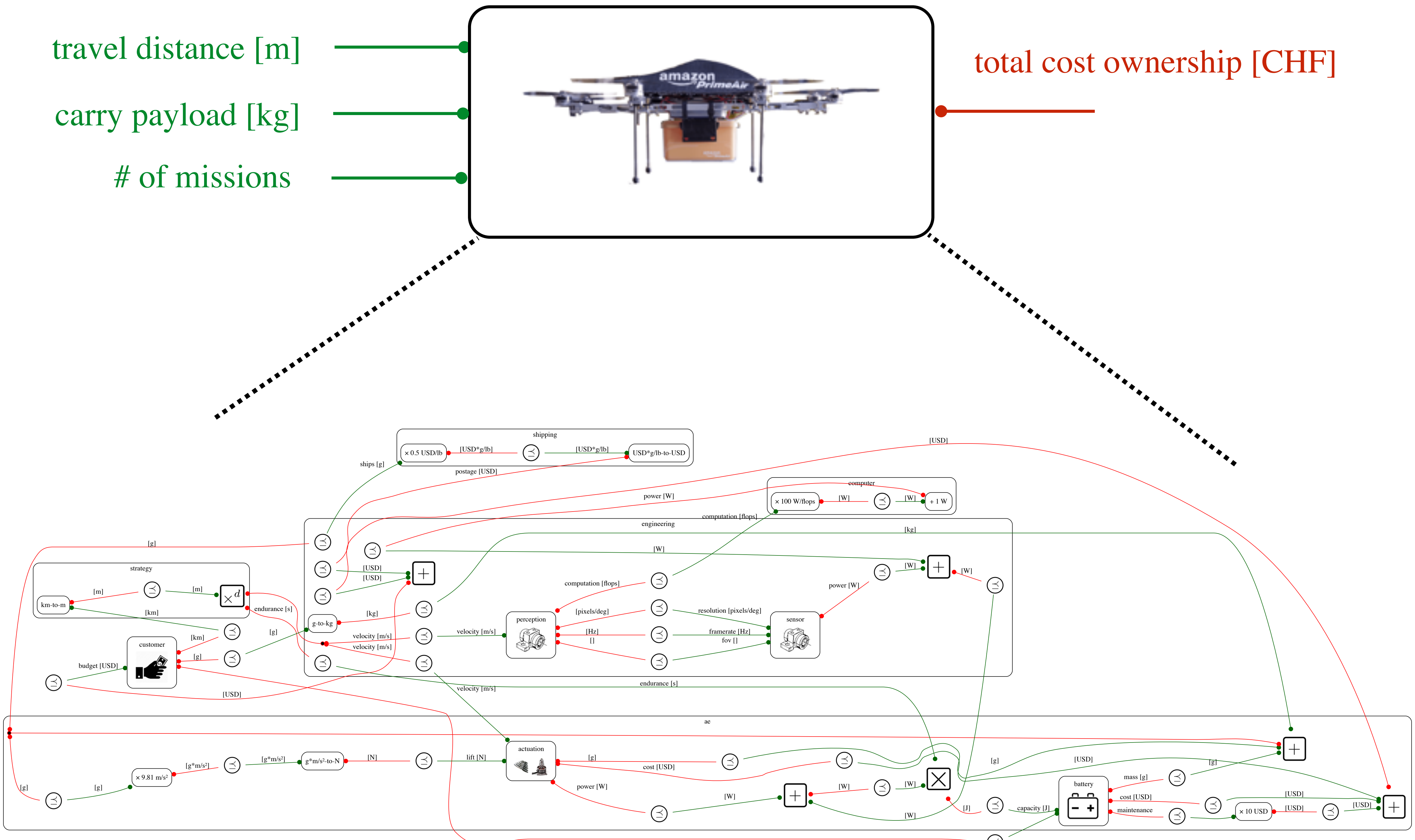
complexity of each step: $(\mathrm{width}(\mathcal{R}))^2$

animation

A mathematical theory of co-design

- ✓ Design problem = **monotone** relations between functionality and resources.
- ✓ Co-design problem = interconnection of design problems
- ✓ Interconnection preserves monotonicity.
- ✓ Semantics as an optimization problem
- ✓ Solution techniques
- ‣ Formal language and more examples

add citations


travel distance [m]
carry payload [kg]
# of missions
total cost ownership [CHF]
shipping
strategy
customer
engineering
computer
perception
sensor
actuation
battery
ae

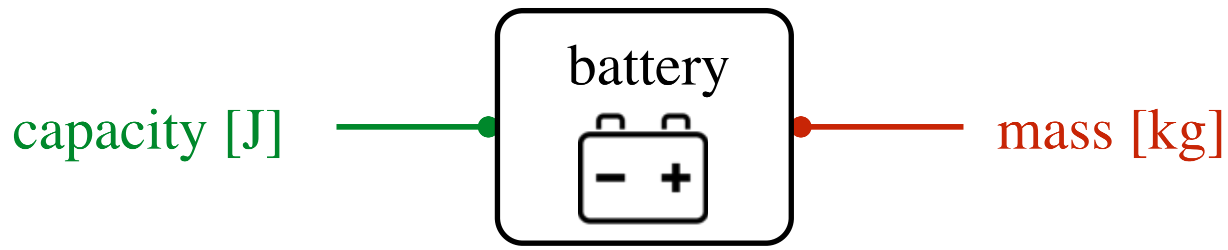
The user's perspective

- **MCDPL**: A user-friendly language to describe MCDPs.
  - inspired by Disciplined Convex Programming (CVX) [Grant & Boyd]

- **PyMCDP**: An interpreter and solver.

  http://mcdp.mit.edu/

  http://github.com/AndreaCensi/mcdp

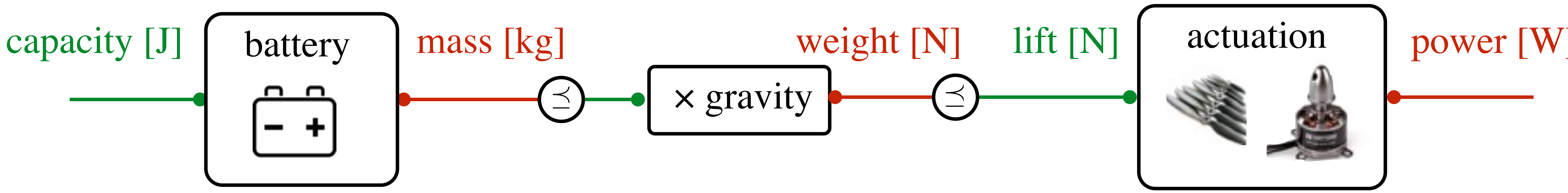


“interface”

```
mcdp {
    provides capacity [J]
    requires mass     [kg]

    specific_energy = 100 Wh / kg

    mass >= capacity / specific_energy
}
```

monotonic constraint

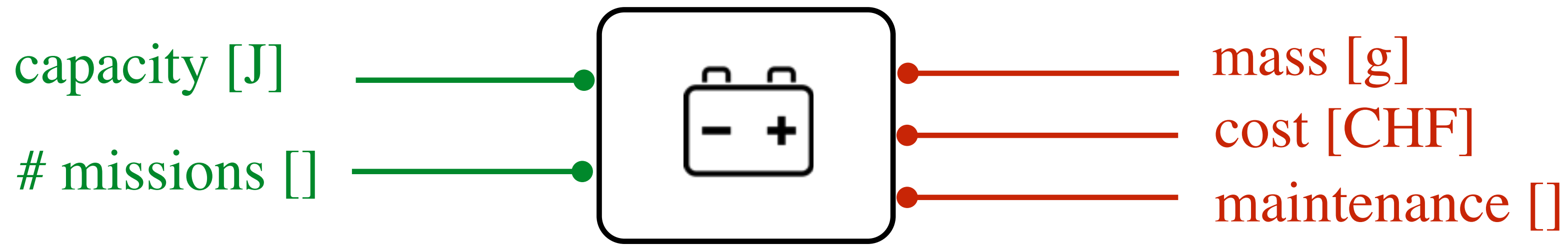

```
mcdp {
    provides lift  [N]
    requires power [W]


    # Maximum lift provided
    lift <= 10 N

    # Power as a function of lift
    p0 = 1 W
    p1 = 1.5 W/N^2
    power >= p0 + p1 * (lift^2)
}
```




```
mcdp {
  battery = new Battery
  actuation = new Actuation

  gravity = 9.81 m/s^2
  weight = (mass required by battery) * gravity

  lift provided by actuation >= weight
}
```

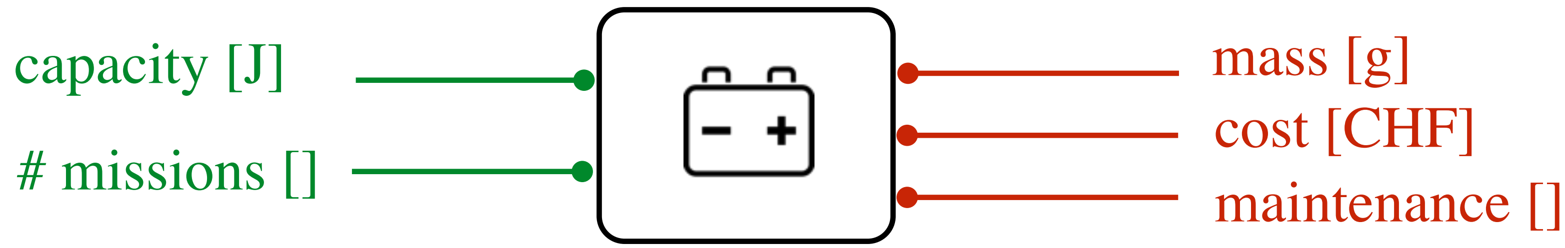
compositionality

interconnection




```
mcdp {
    provides capacity [J]
    # Number of missions to be flown
    provides missions [R]

    requires mass       [g]
    requires cost       [CHF]
    # Number of replacements needed
    requires maintenance [R]

    specific_energy = 150 Wh/kg
    specific_cost =  2.50 Wh/CHF
    cycles = 600 []

    # How many times should it be replaced?
    num_replacements = ceil(missions / cycles)
    maintenance >= num_replacements

    mass >= capacity / specific_energy

    unit_cost = capacity / specific_cost
    cost >= unit_cost * num_replacements
}
```

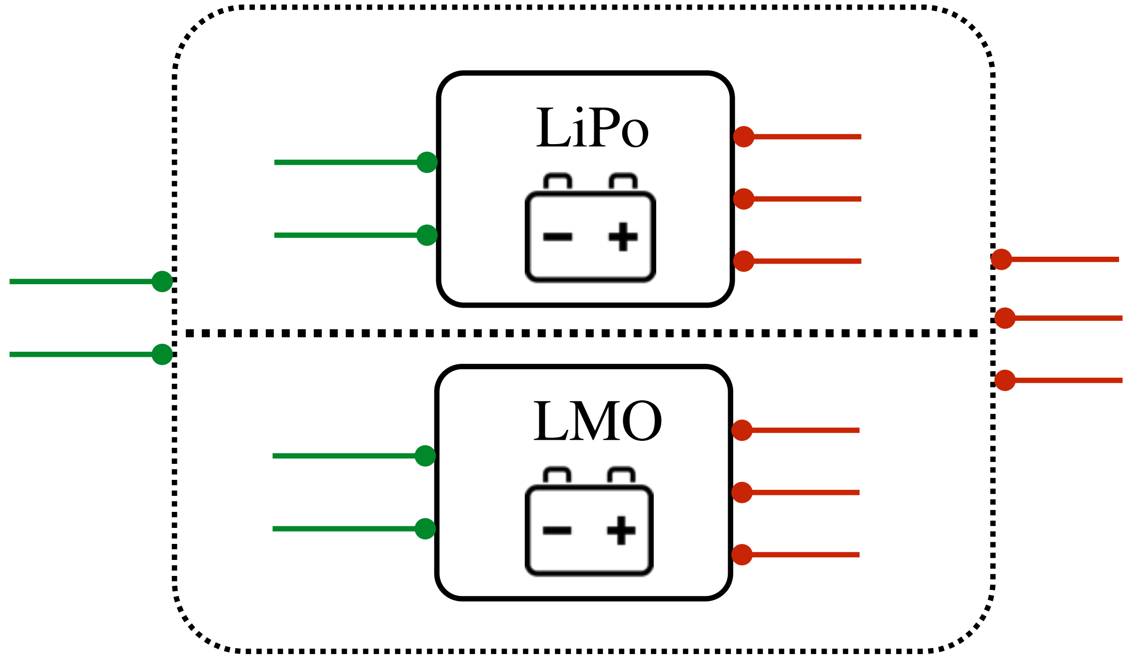
parameters for specific type of battery



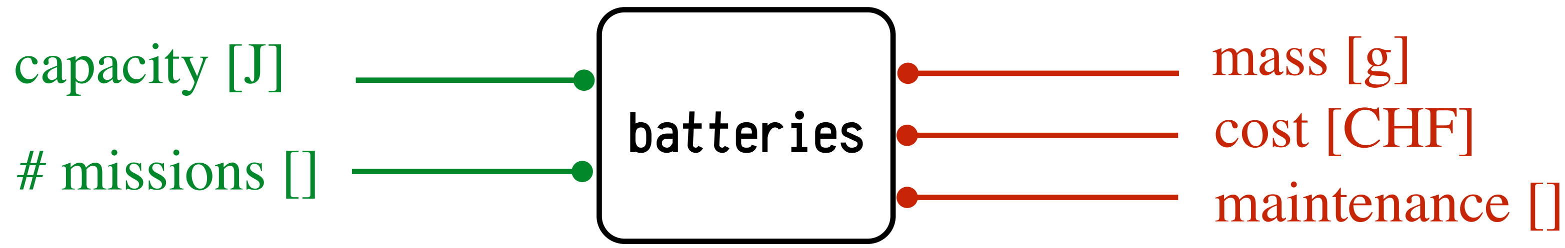

```
mcdp {
    provides capacity [J]
    # Number of missions to be flown
    provides missions [R]

    requires mass        [g]
    requires cost        [CHF]
    # Number of replacements needed
    requires maintenance [R]

    specific_energy = 150 Wh/kg
    specific_cost =  2.50 Wh/CHF
    cycles = 600 []

```

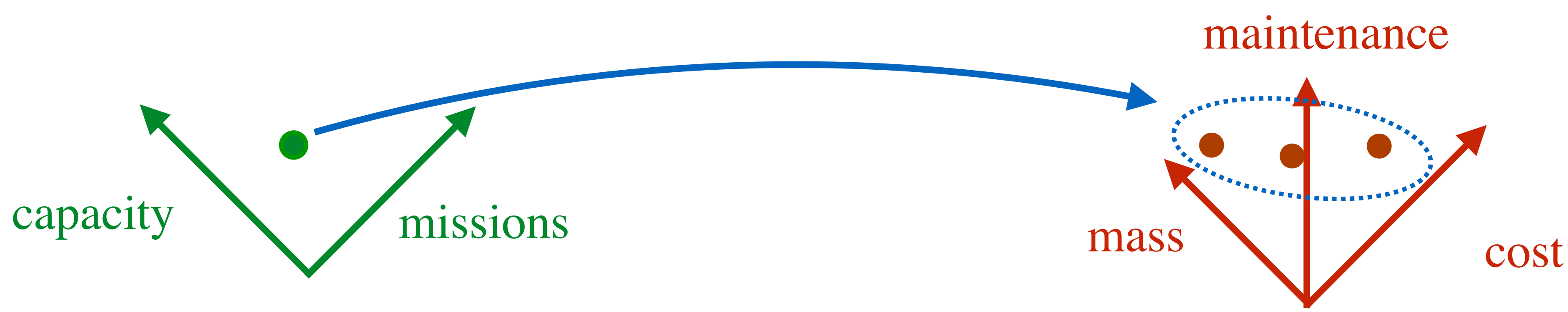
```
 NiMH,  100 Wh/kg,   3.41 Wh/$,    500 cycles,  "Nickel-metal hydride"
 NiH2,   45 Wh/kg,  10.50 Wh/$,  20000 cycles,  "Nickel-hydrogen"
  LCO,  195 Wh/kg,   2.84 Wh/$,    750 cycles,  "Lithium cobalt oxide"
  LMO,  150 Wh/kg,   2.84 Wh/$,    500 cycles,  "Lithium manganese oxide"
NiCad,   30 Wh/kg,   7.50 Wh/$,    500 cycles,  "Nickel-cadmium"
  SLA,   30 Wh/kg,   7.00 Wh/$,    500 cycles,  "Lead-acid"
 LiPo,  150 Wh/kg,   2.50 Wh/$,    600 cycles,  "Lithium polymer"
  LFP,   90 Wh/kg,   1.50 Wh/$,   1500 cycles,  "Lithium iron phosphate"
```

(Wikipedia)

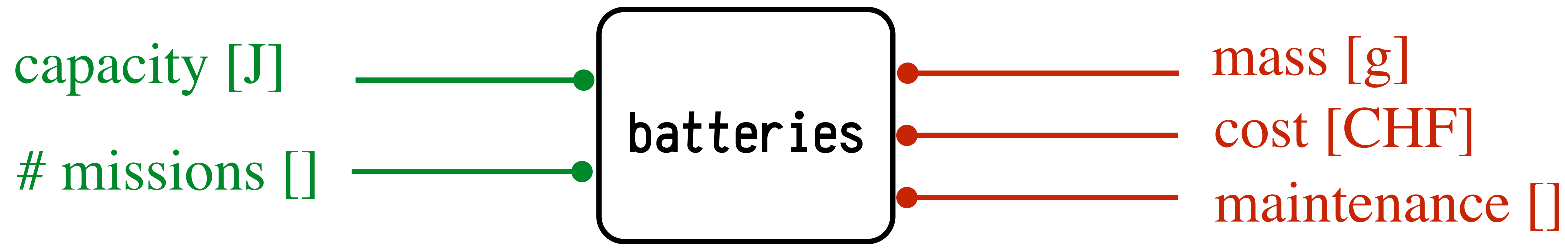


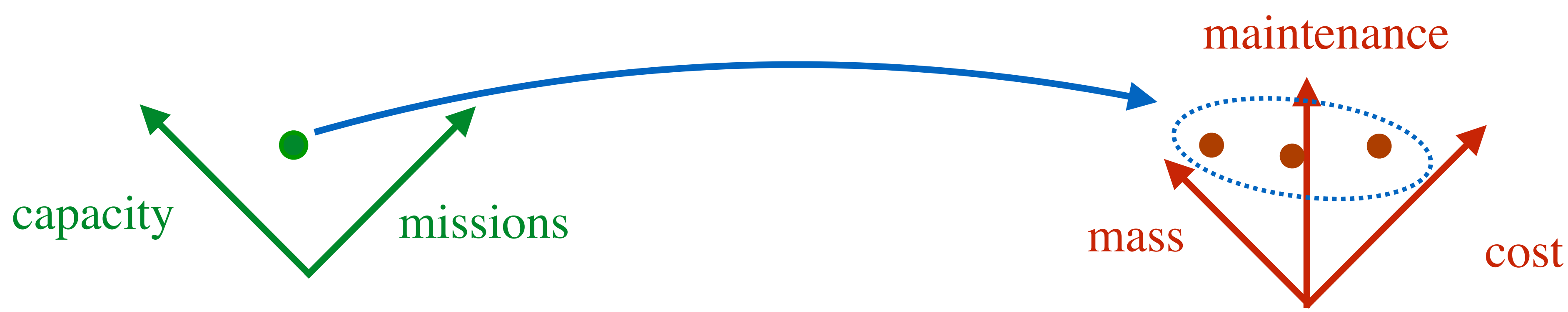

```
choose (
    LiPo: new Battery_LiPO,
    LMO:  new Battery_LMO
)
```

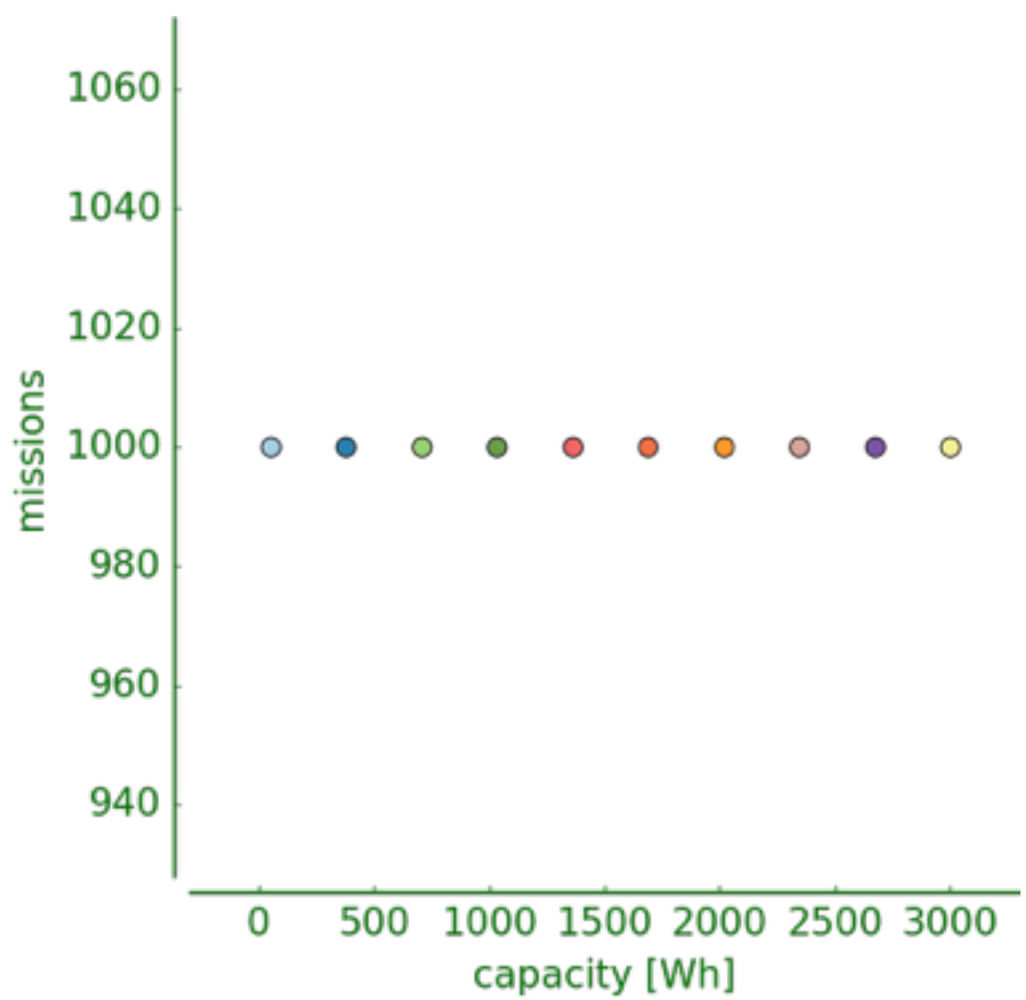


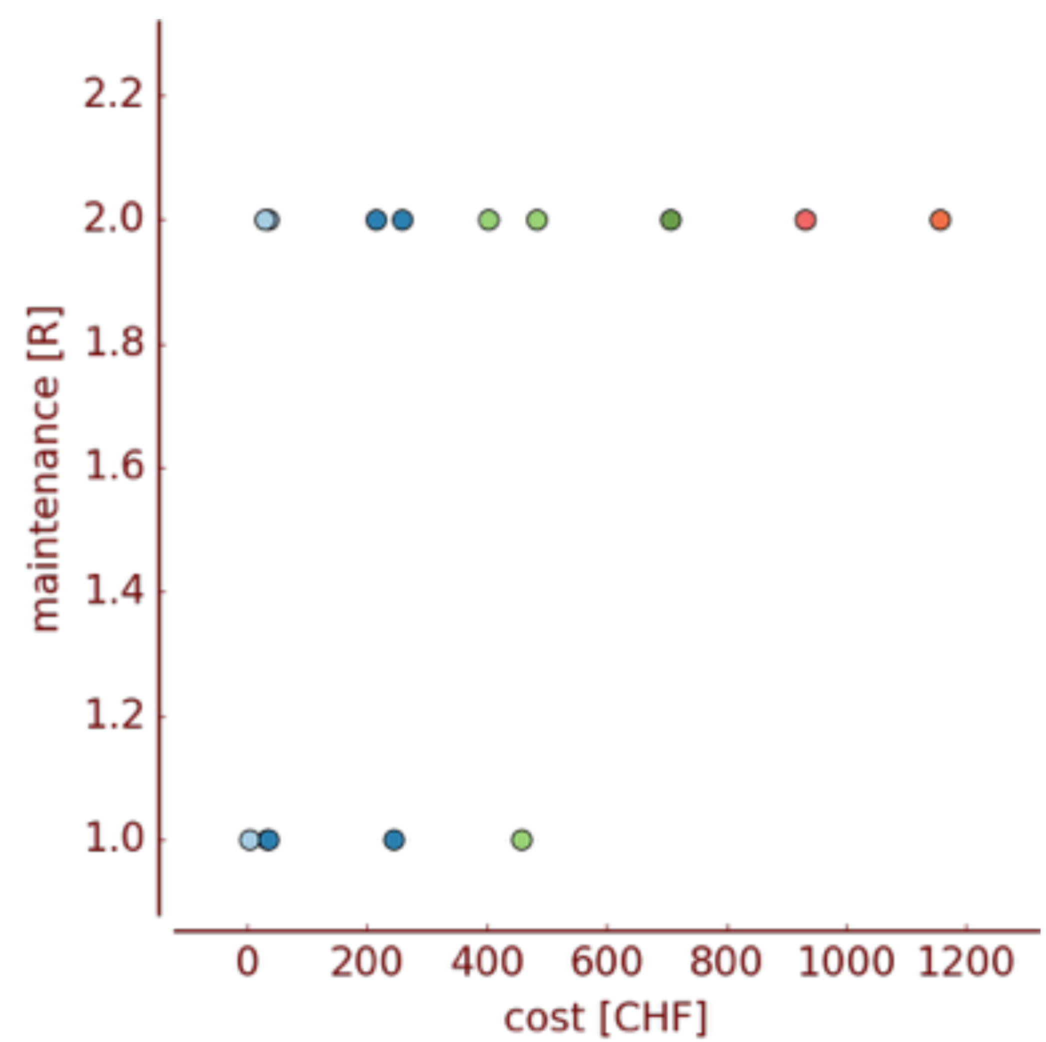


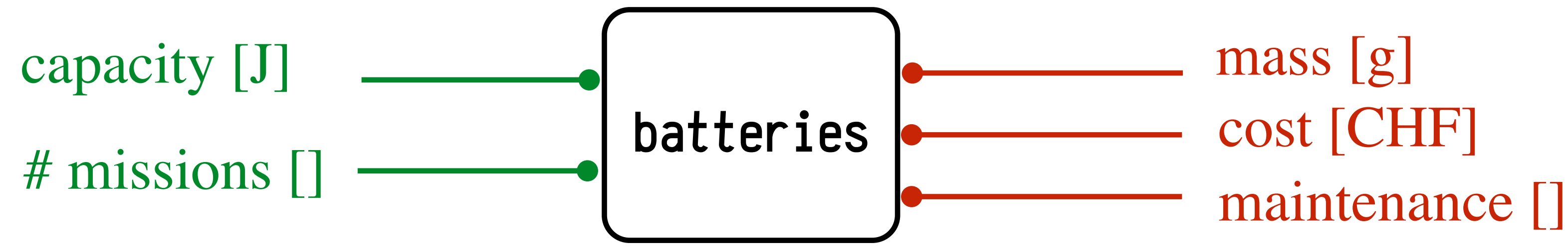

```
$ mcdp-solve batteries ”<100 Wh, 500 []>”
query: ⟨capacity:360000 J, missions:500 ⟩
Minimal resources needed:
maintenance, cost, mass = ↑{
    ⟨1, 10 CHF, 2230 g⟩,
    ⟨1, 30 CHF, 1000 g⟩,
    ⟨1, 36 CHF, 520 g⟩
}
```

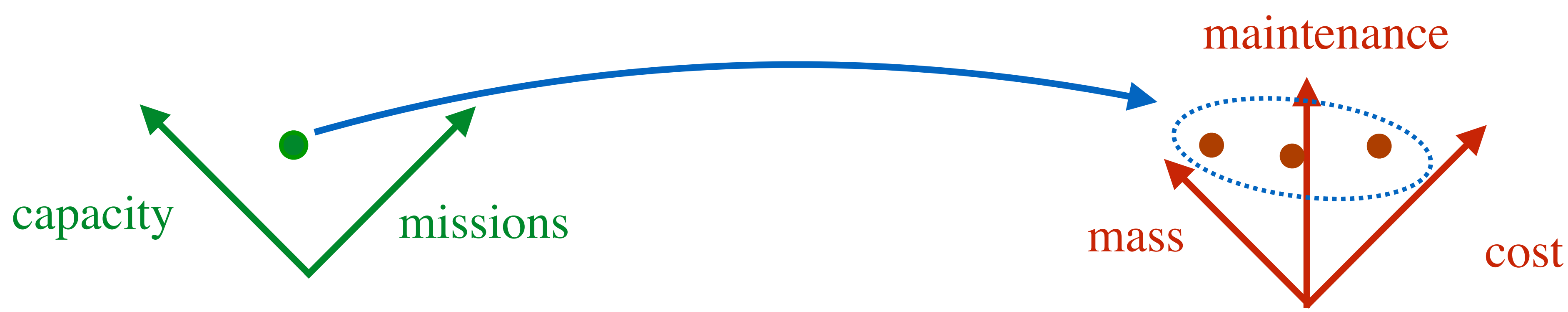

capacity [J]
missions []
batteries
mass [g]
cost [CHF]
maintenance []

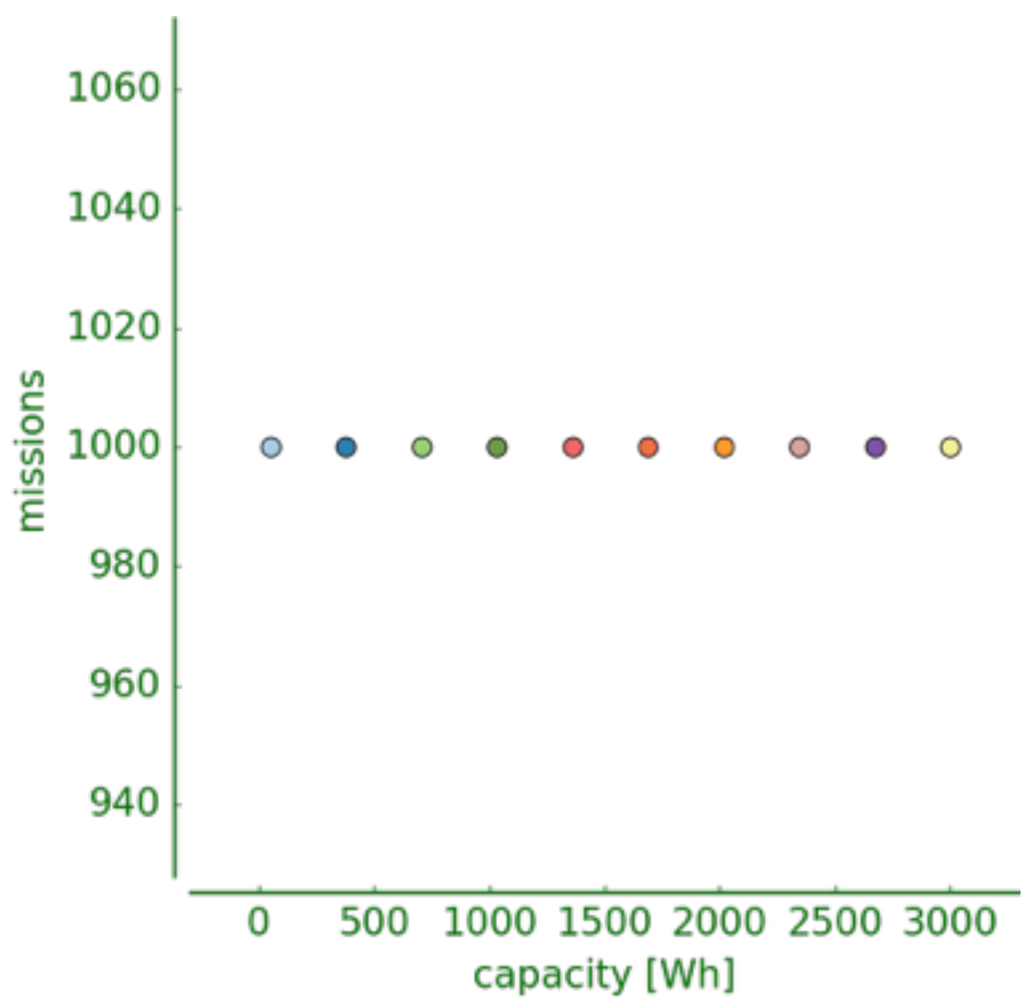

capacity
missions
maintenance
mass
cost

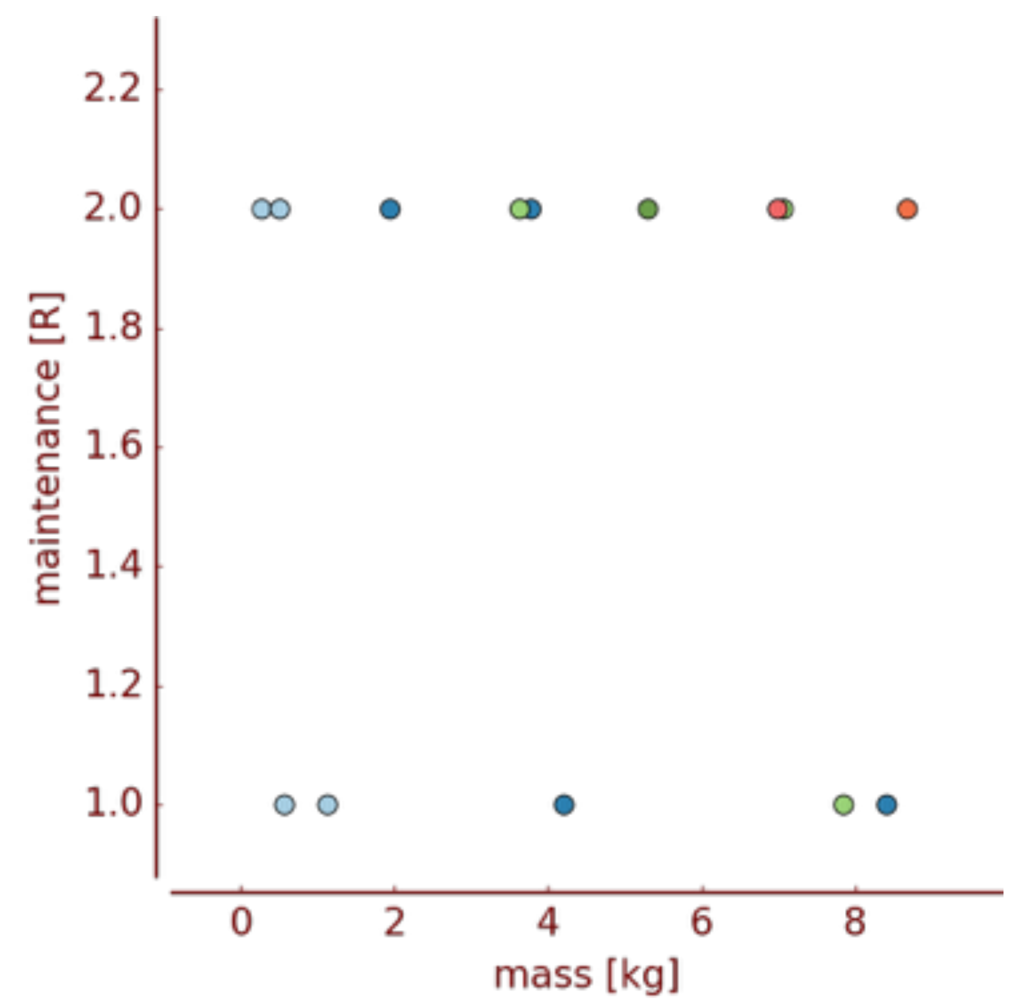

missions
1060
1040
1020
1000
980
960
940
0
500
1000
1500
2000
2500
3000
capacity [Wh]

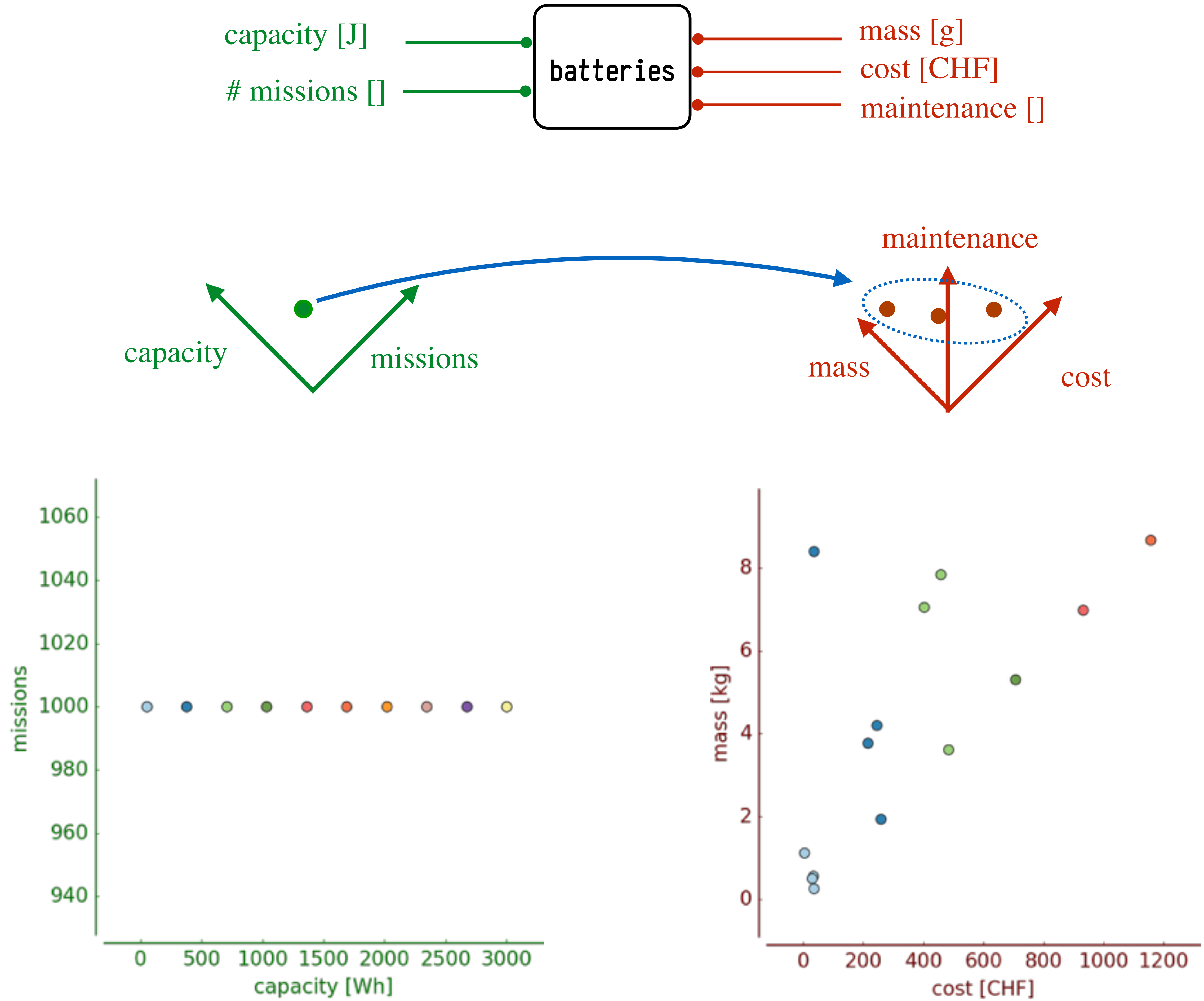

maintenance [R]
2.2
2.0
1.8
1.6
1.4
1.2
1.0
0
200
400
600
800
1000
1200
cost [CHF]

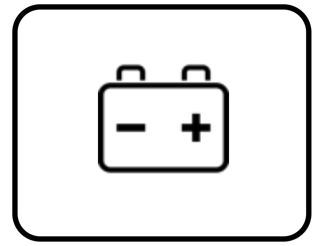

capacity [J]
missions []
batteries
mass [g]
cost [CHF]
maintenance []

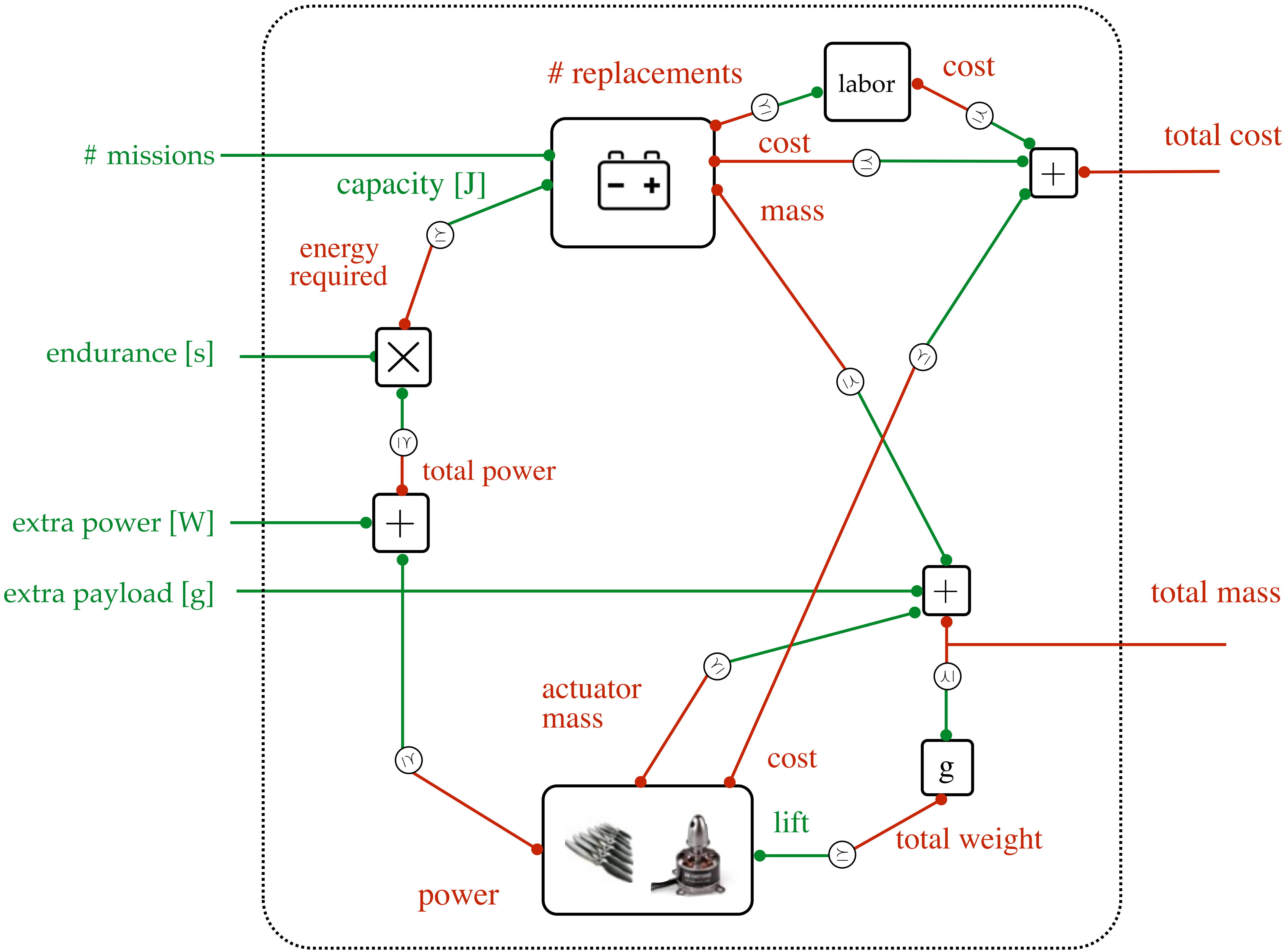

capacity
missions
maintenance
mass
cost


1060
1040
1020
1000
980
960
940
missions
0
500
1000
1500
2000
2500
3000
capacity [Wh]

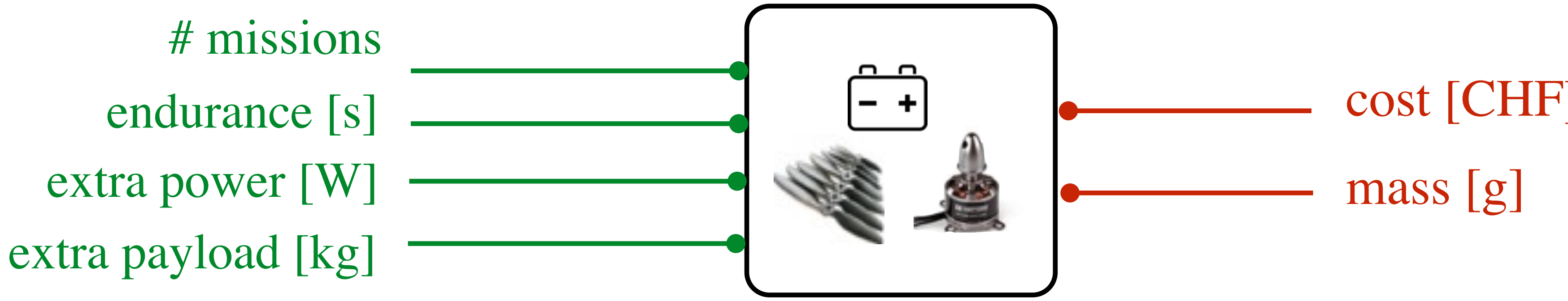

2.2
2.0
1.8
1.6
1.4
1.2
1.0
maintenance [R]
0
2
4
6
8
mass [kg]


capacity [J]
missions []
batteries
mass [g]
cost [CHF]
maintenance []
maintenance
capacity
missions
mass
cost
1060
1040
1020
1000
980
960
940
missions
0 500 1000 1500 2000 2500 3000
capacity [Wh]
8
6
4
2
0
mass [kg]
0 200 400 600 800 1000 1200
cost [CHF]

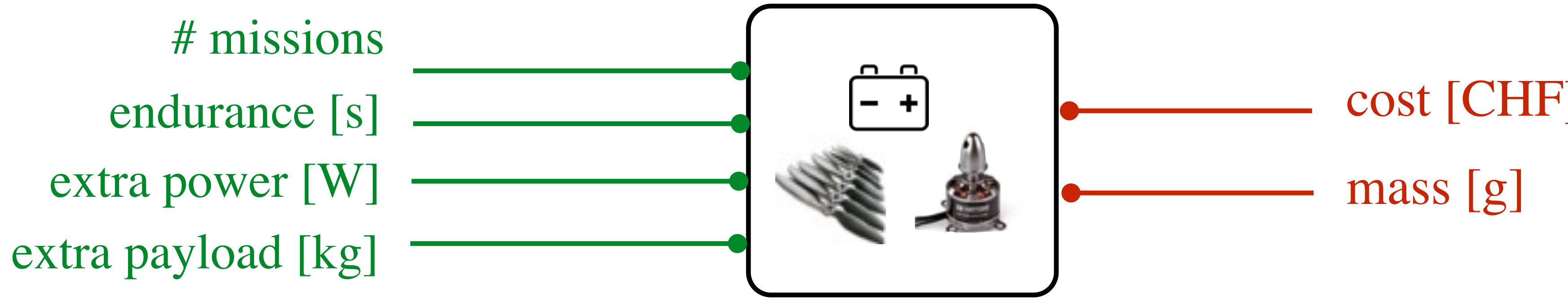


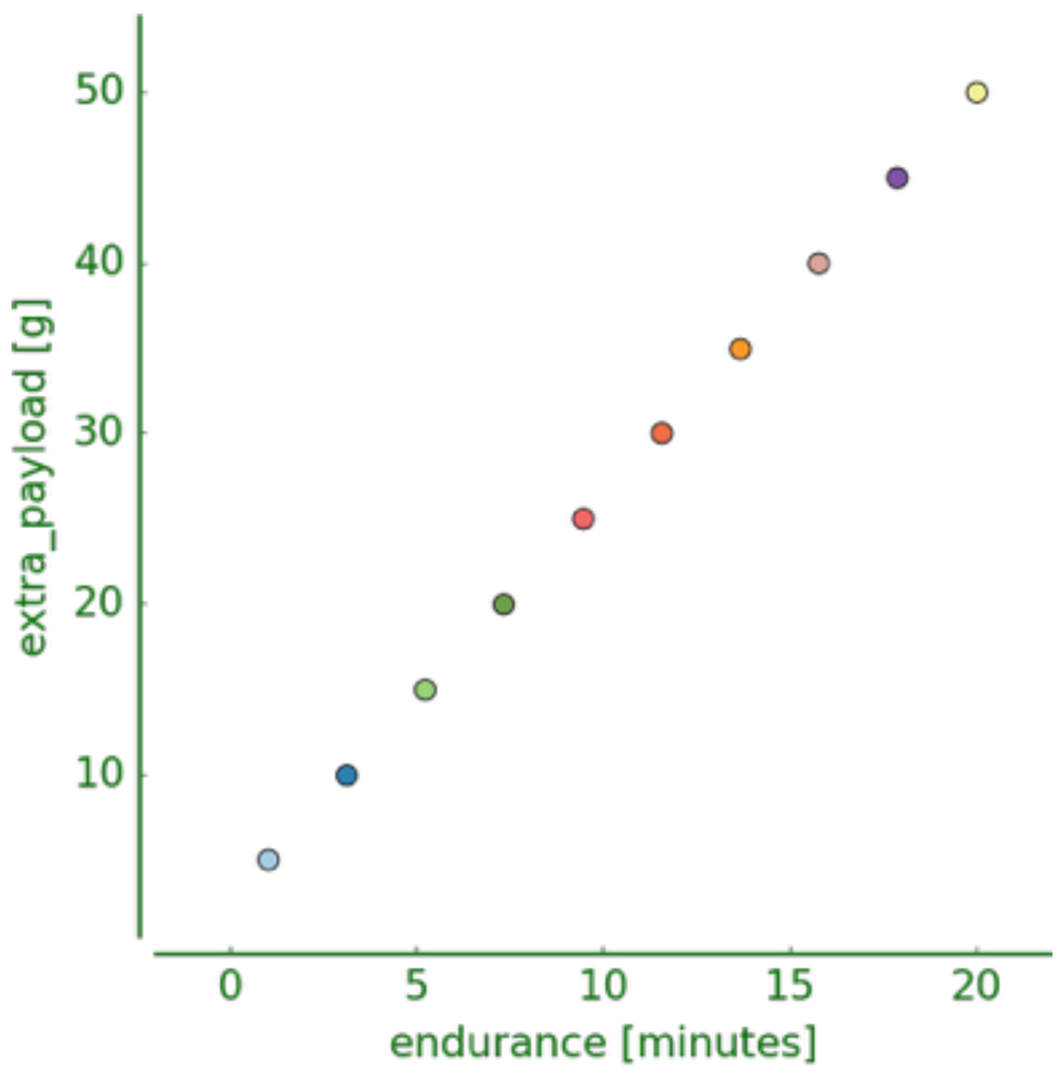


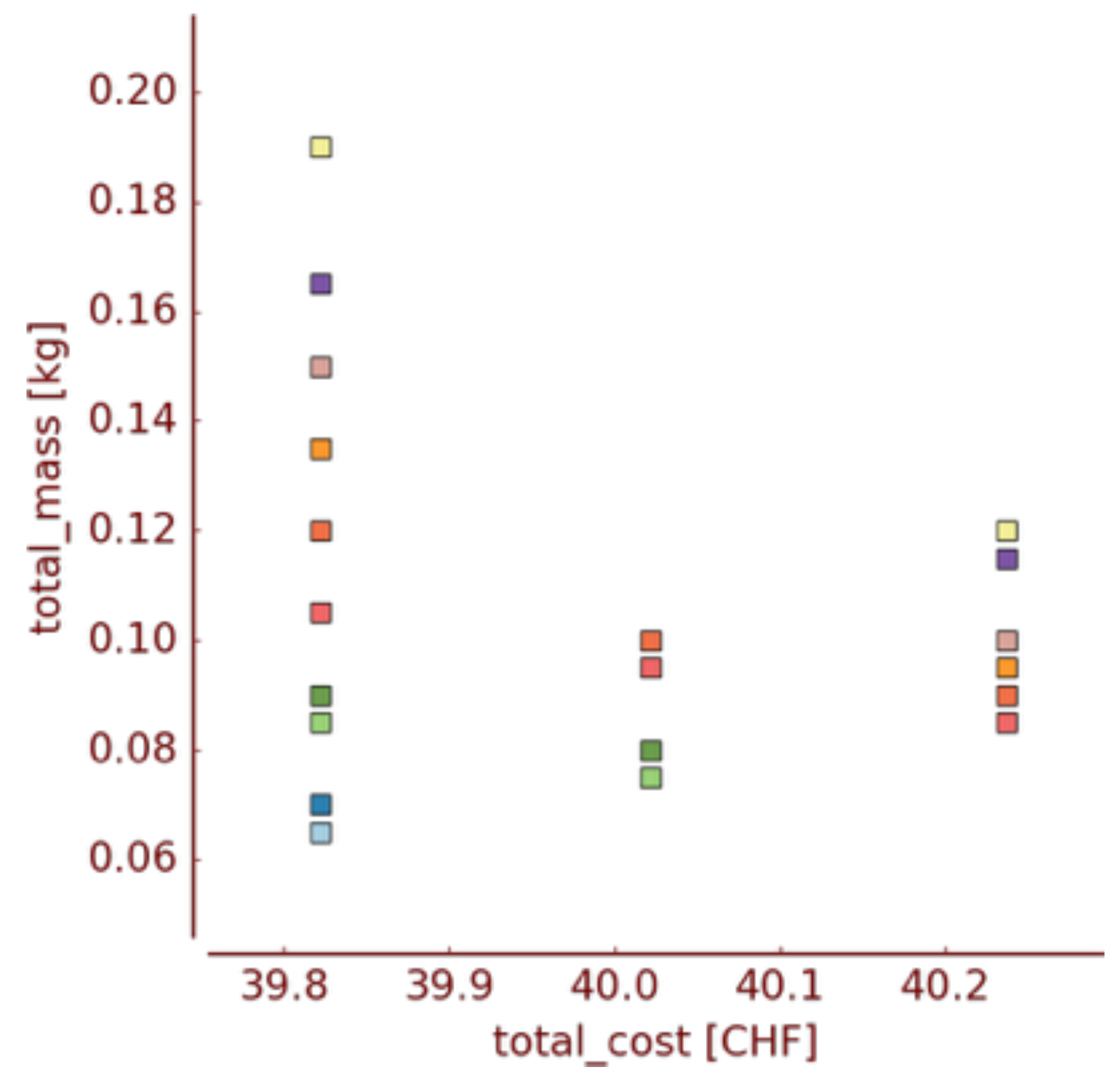
replacements
labor
cost
total cost
missions
cost
capacity [J]
mass
energy required
endurance [s]
total power
extra power [W]
extra payload [g]
total mass
actuator mass
cost
g
lift
total weight
power



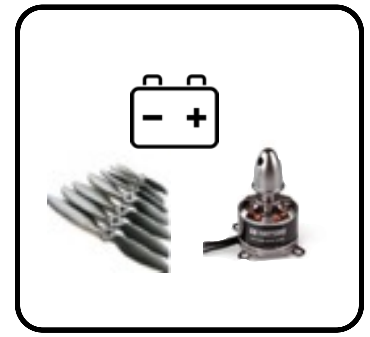
missions
endurance [s]
extra power [W]
extra payload [kg]
cost [CHF]
mass [g]

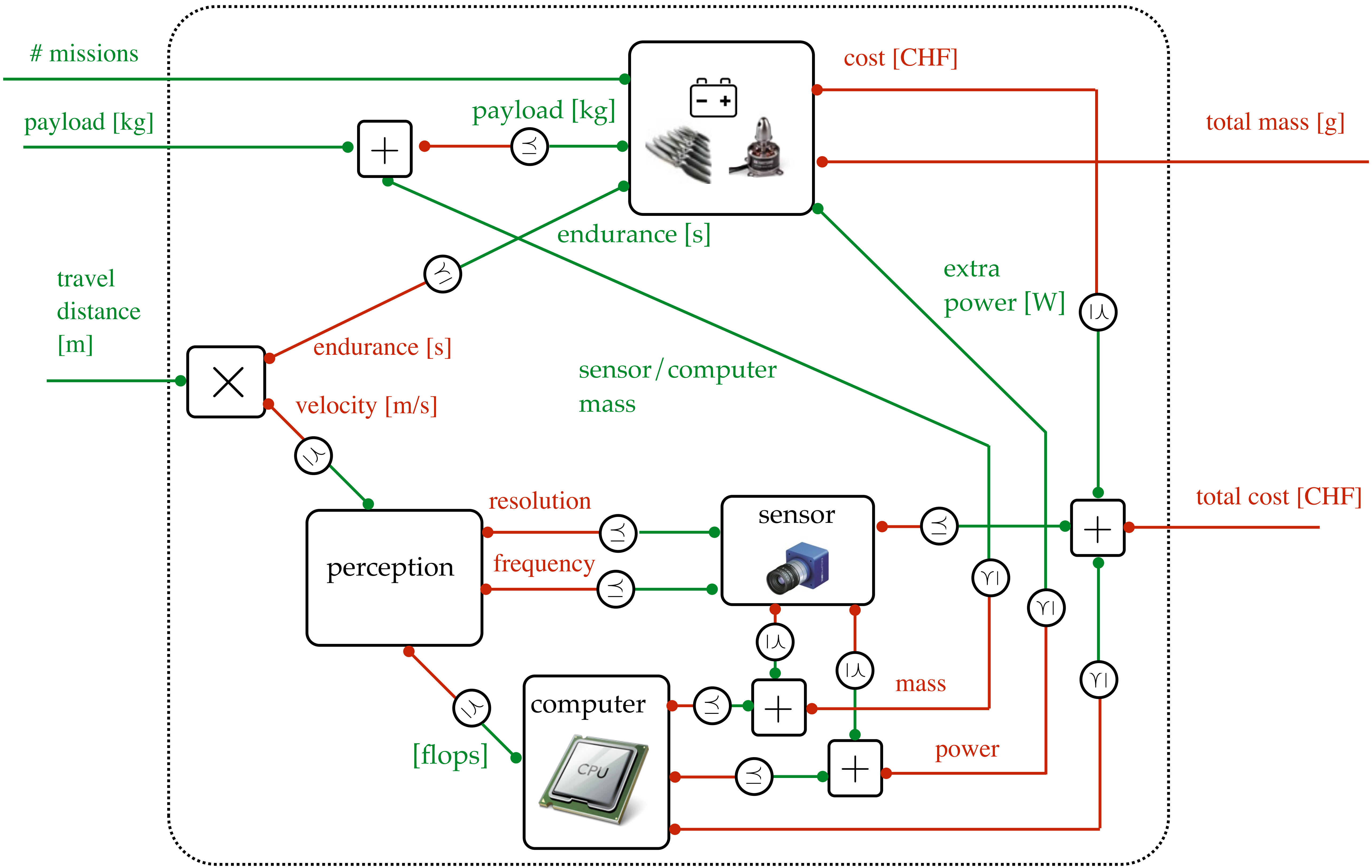






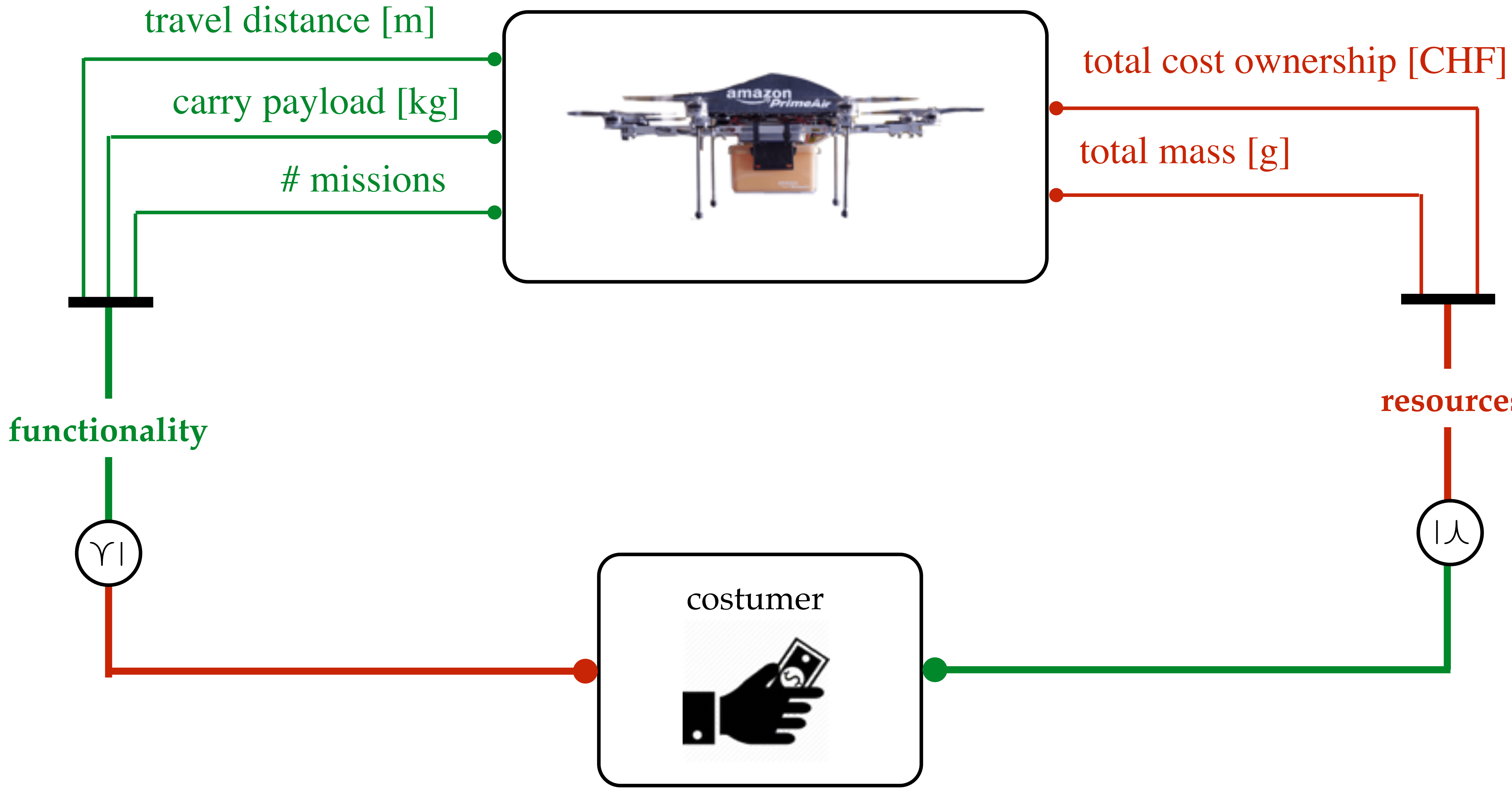
add mcdp-command



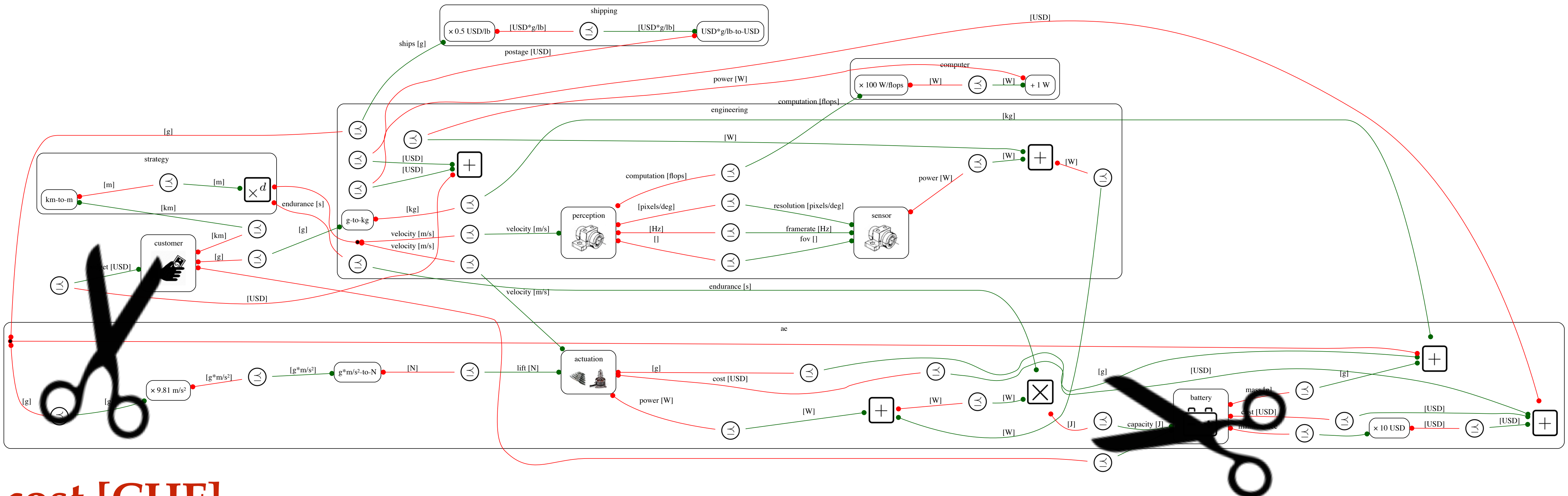


missions
payload [kg]
payload [kg]
cost [CHF]
total mass [g]
endurance [s]
travel
distance
[m]
extra
power [W]
endurance [s]
velocity [m/s]
sensor / computer
mass
resolution
sensor
total cost [CHF]
perception
frequency
mass
computer
CPU
[flops]
power


travel distance [m]
carry payload [kg]
missions
total cost ownership [CHF]
total mass [g]


travel distance [m]
carry payload [kg]
missions
total cost ownership [CHF]
total mass [g]
functionality
resources
costumer

- Removing 2 edges removes all 22 oriented cycles.



cost [CHF]

battery capacity [J]

- These are the **co-design constraints** that tie everything together.

Summary

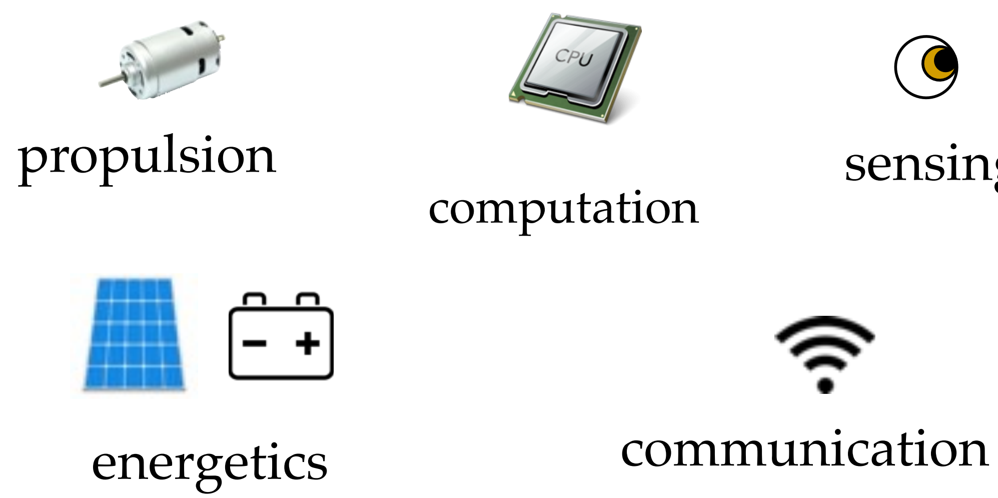
- **Need**: formal design methods for complex autonomous systems.

A mathematical theory of co-design

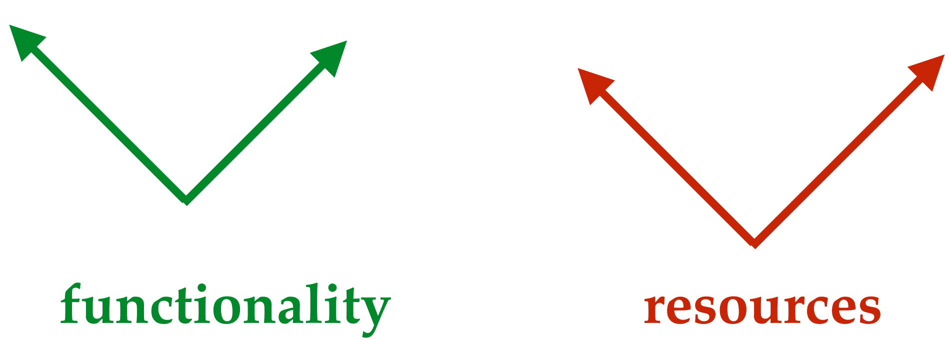
resources constraints

Watts, CHF, …

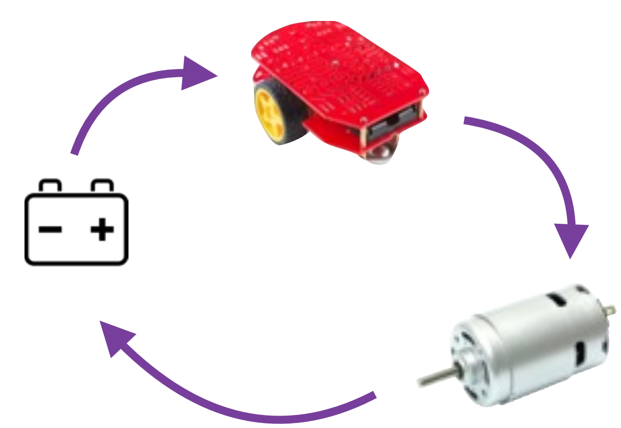
heterogenous domains



trade-offs of
functionality and resources



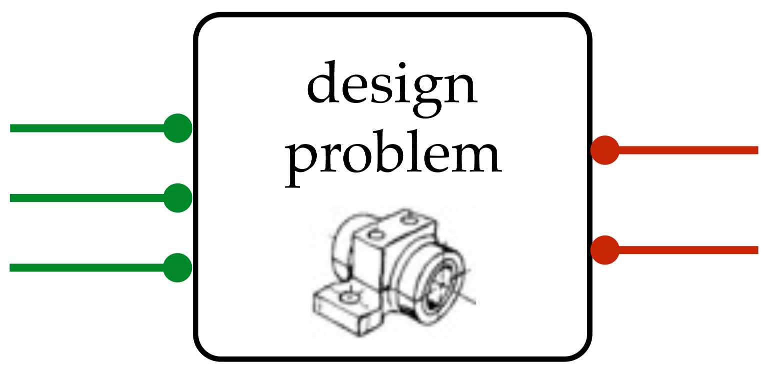
recursive
co-design constraints



A mathematical theory of co-design

- A **design problem** is abstracted as a **relation** between **provided functionality** and **required resources.**

provided **functionality**



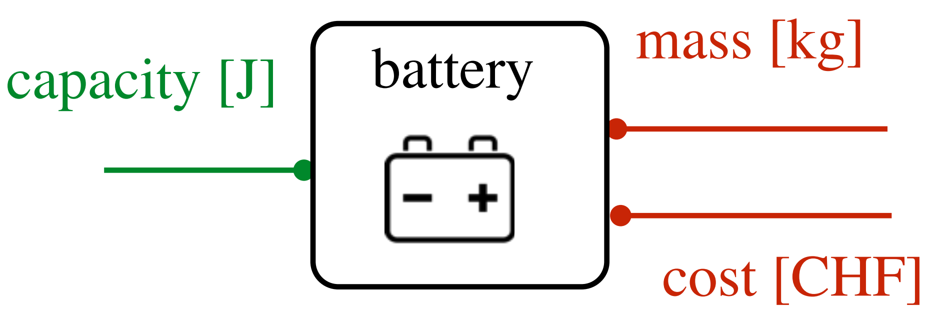
required **resources**

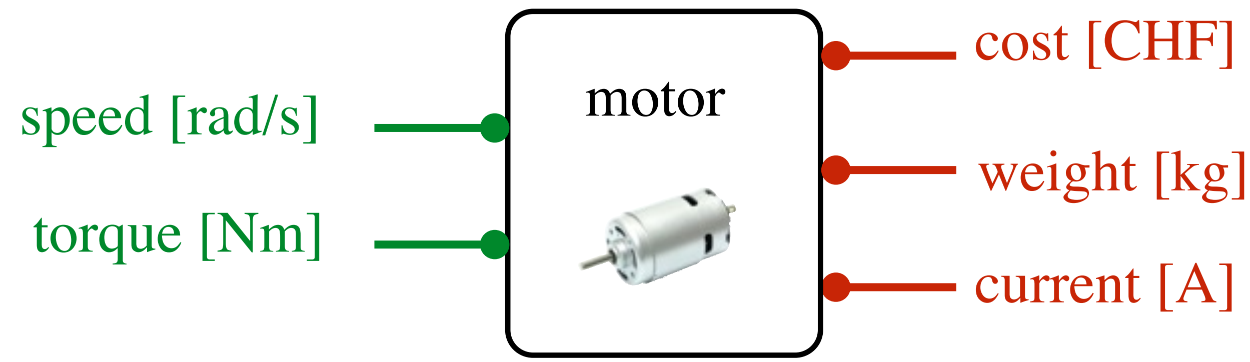
$\langle \mathcal{F}, \leq_{\mathcal{F}} \rangle$

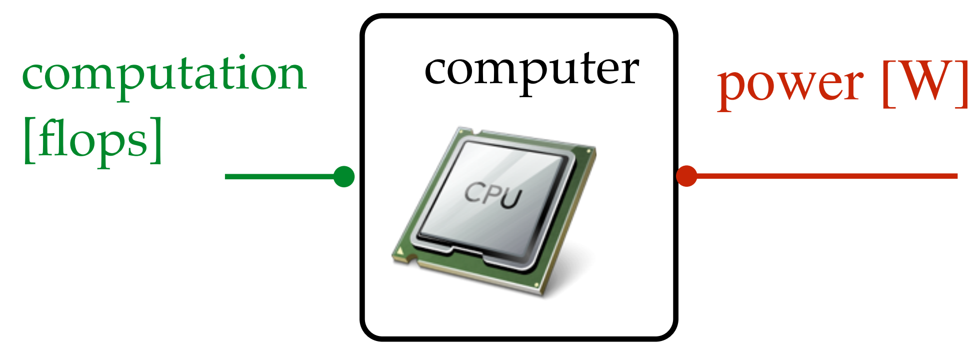
$\langle \mathcal{R}, \leq_{\mathcal{R}} \rangle$

A mathematical theory of co-design

- **Multi-scale**: from components to systems.

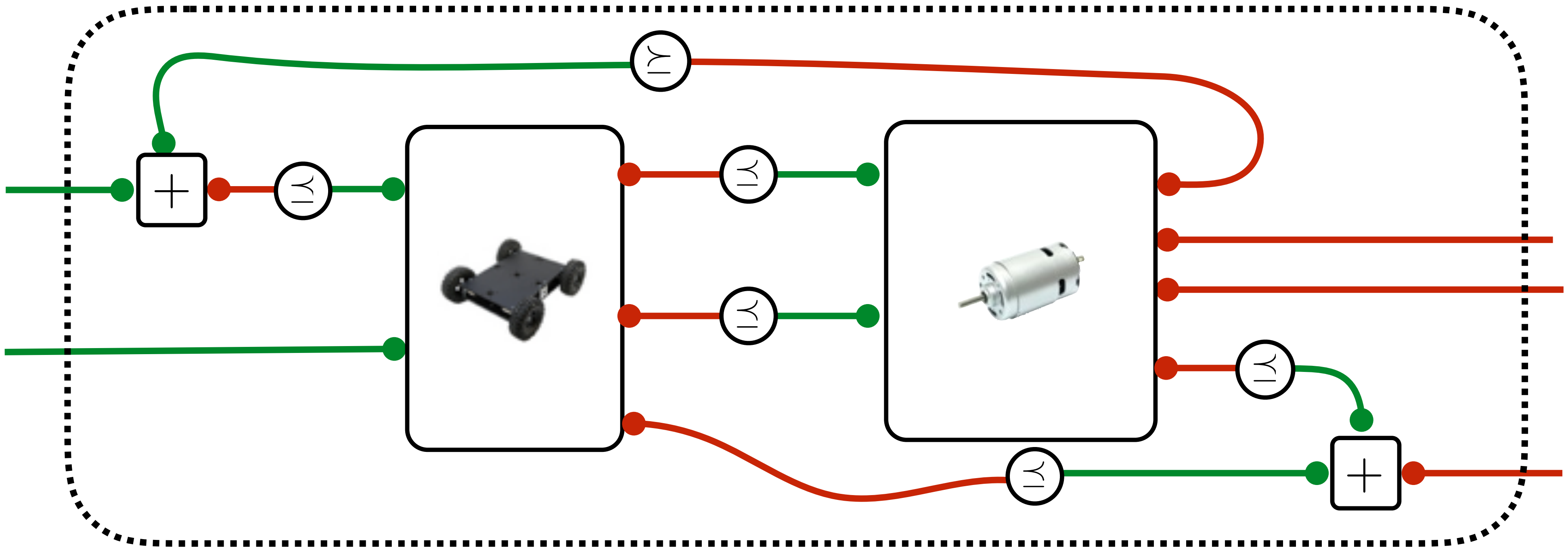




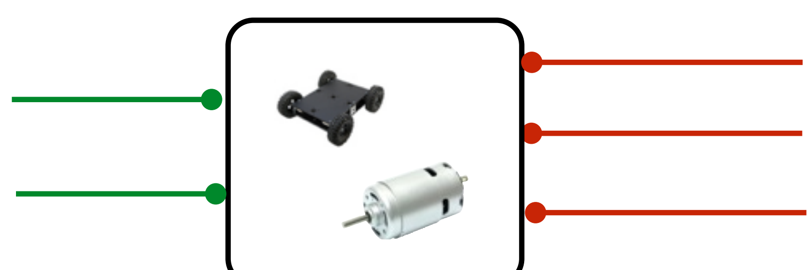


A mathematical theory of co-design

- **Compositionality** and abstraction properties

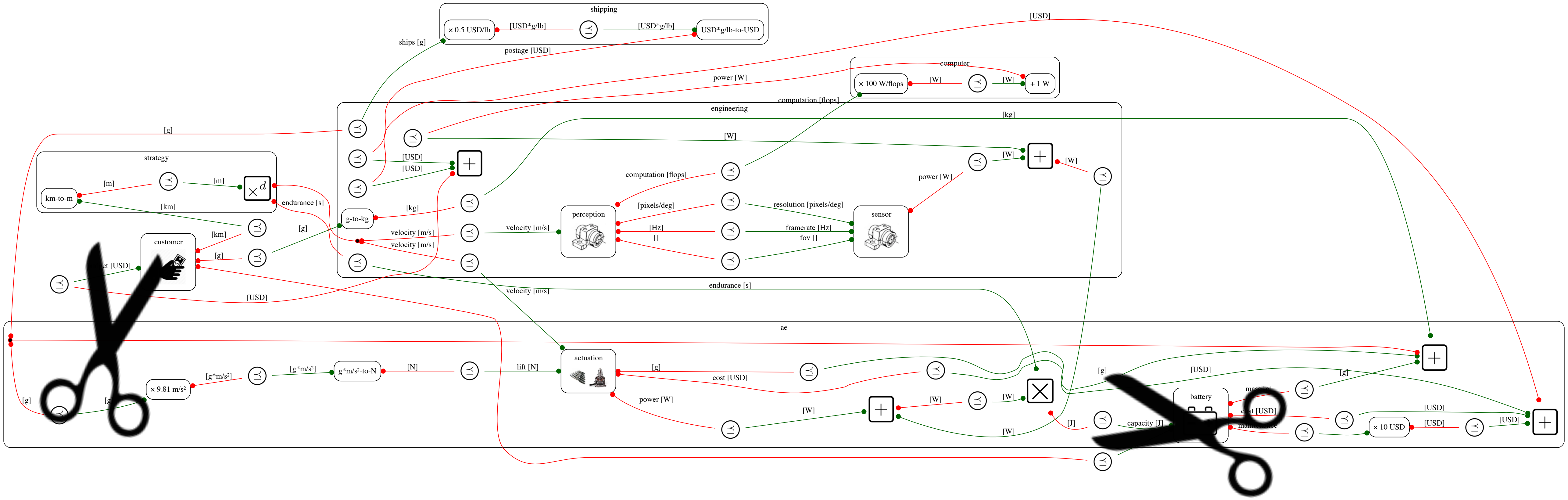


abstraction

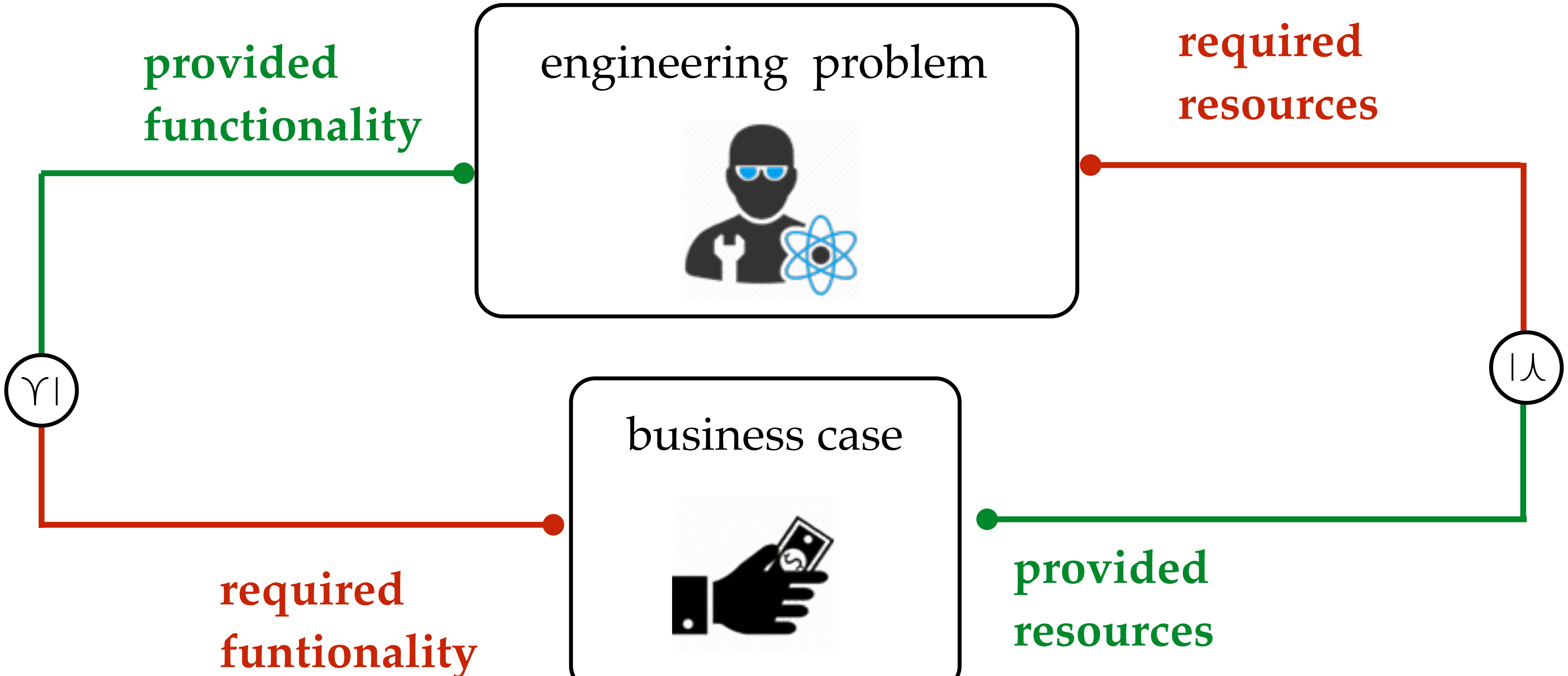


A mathematical theory of co-design

- **Algorithmic results:**
  - There exists a systematic solution guaranteed to **find all minimal solutions, or certificate of infeasibility.**
  - Complexity depends on the **structure of the co-design graph.**



A mathematical theory of co-design

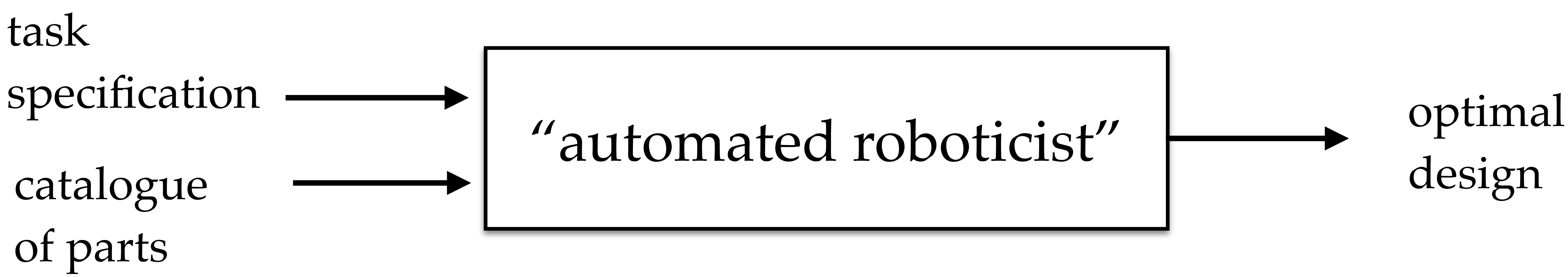


A mathematical theory of co-design

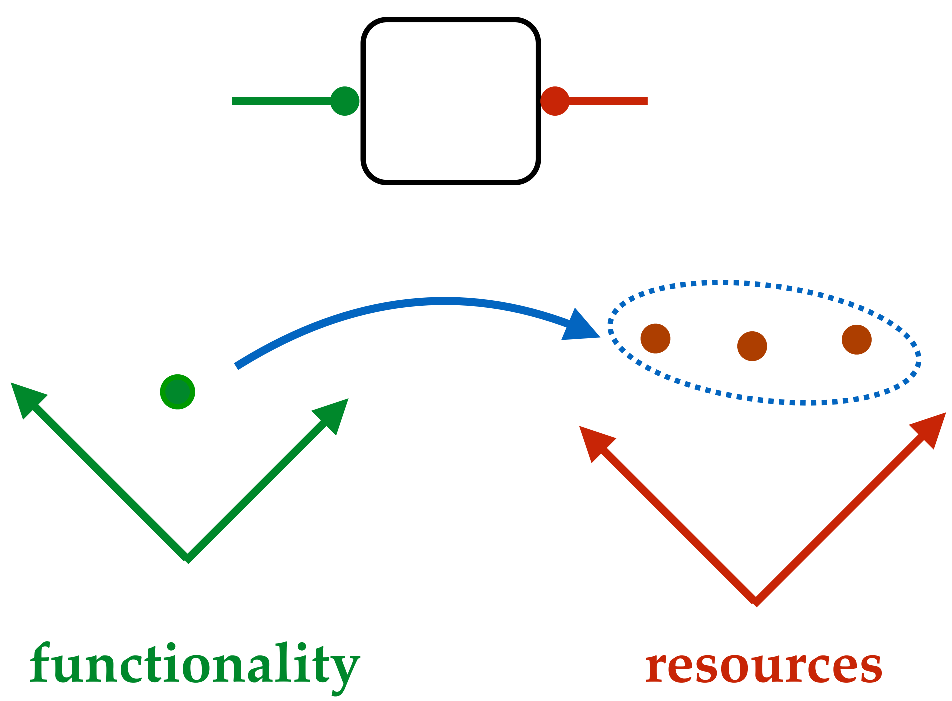
- Concrete **implementation** as a formal language.

```
mcdp {
    provides capacity [J]
    # Number of missions to be flown
    provides missions [R]

    requires mass     [g]
    requires cost     [CHF]
    # Number of replacements needed
    requires maintenance [R]

    specific_energy = 150 Wh/kg
    specific_cost =  2.50 Wh/CHF
    cycles = 600 []

    # How many times should it be replaced?
    num_replacements = ceil(missions / cycles)
    maintenance >= num_replacements

    mass >= capacity / specific_energy

    unit_cost = capacity / specific_cost
    cost >= unit_cost * num_replacements
}
```

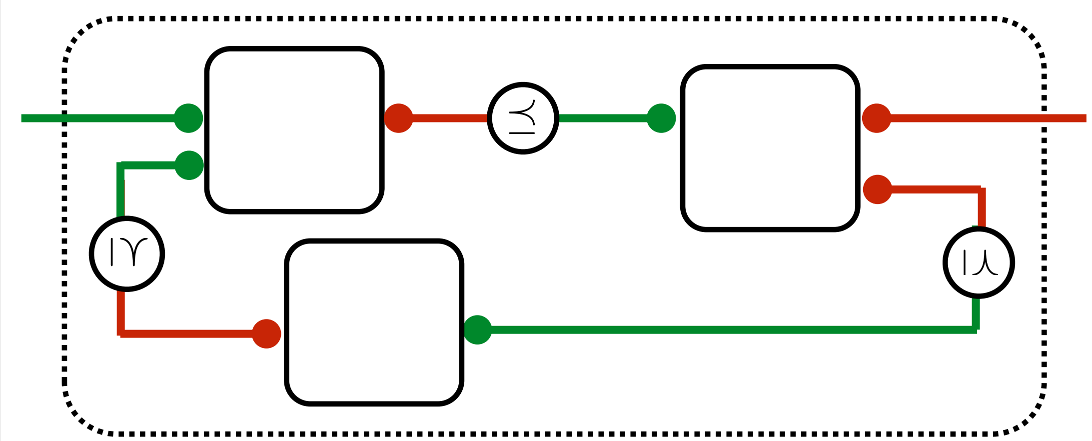
```
mcdp {
    provides lift  [N]
    requires power [W]


    # Maximum lift provided
    lift <= 10 N

    # Power as a function of lift
    p0 = 1 W
    p1 = 1.5 W/N^2
    power >= p0 + p1 * (lift^2)
}
```



What components do.

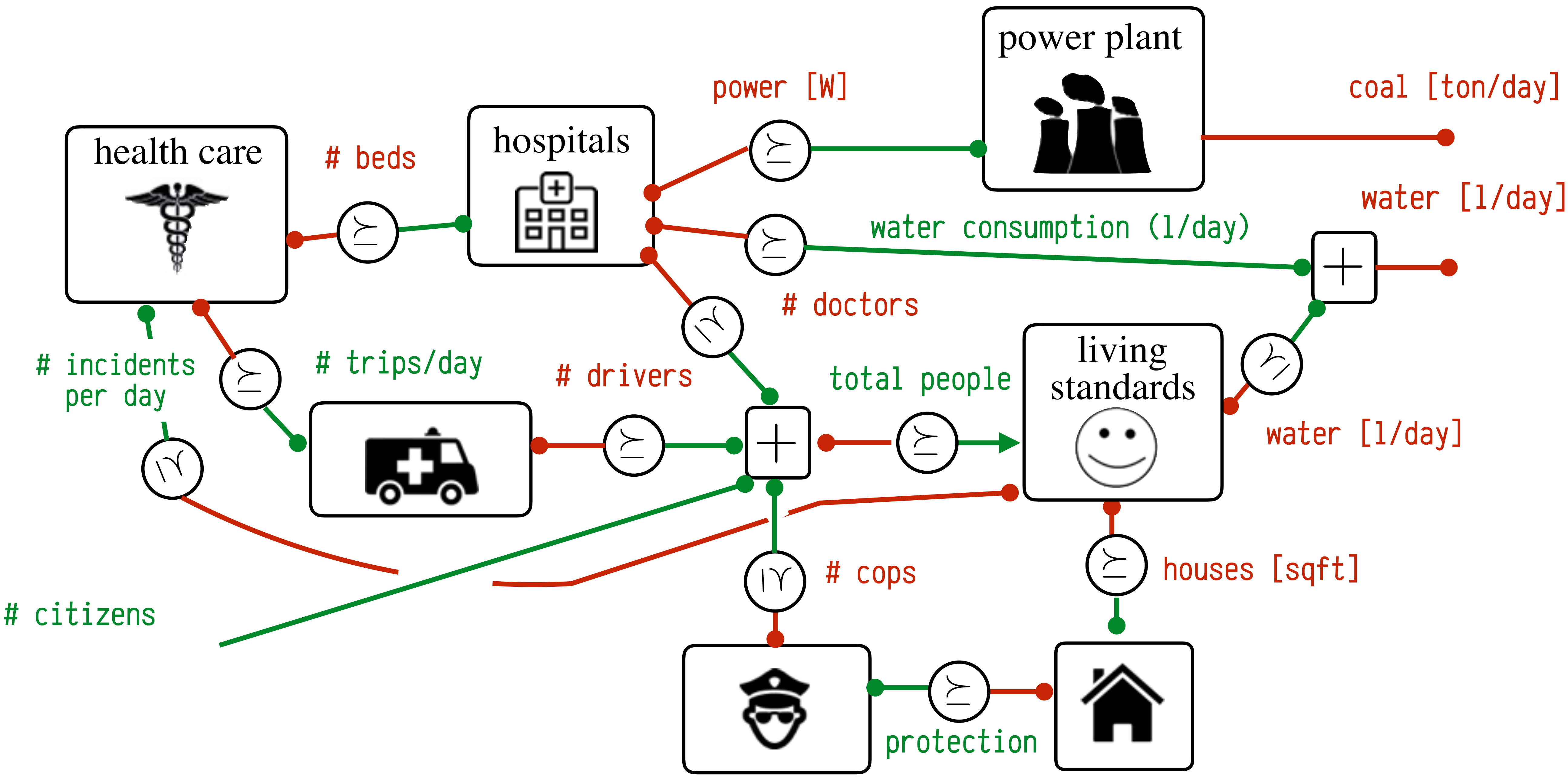


How components work together.



```
power = actuation.power + extra_power
energy = power * endurance
battery.capacity >= energy
```

- **Future work**: theory - tools - robotics - other fields



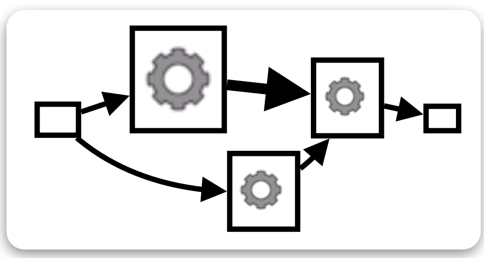
health care
beds
hospitals
power [W]
power plant
coal [ton/day]
water [l/day]
water consumption (l/day)
doctors
incidents per day
trips/day
drivers
total people
living standards
water [l/day]
cops
houses [sqft]
citizens
protection

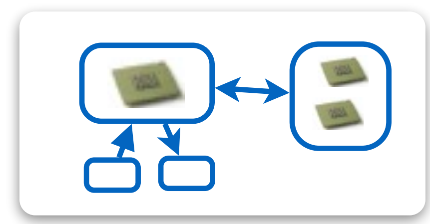
- max latency [-s]
min throughput [s]
computation graph

resource allocation

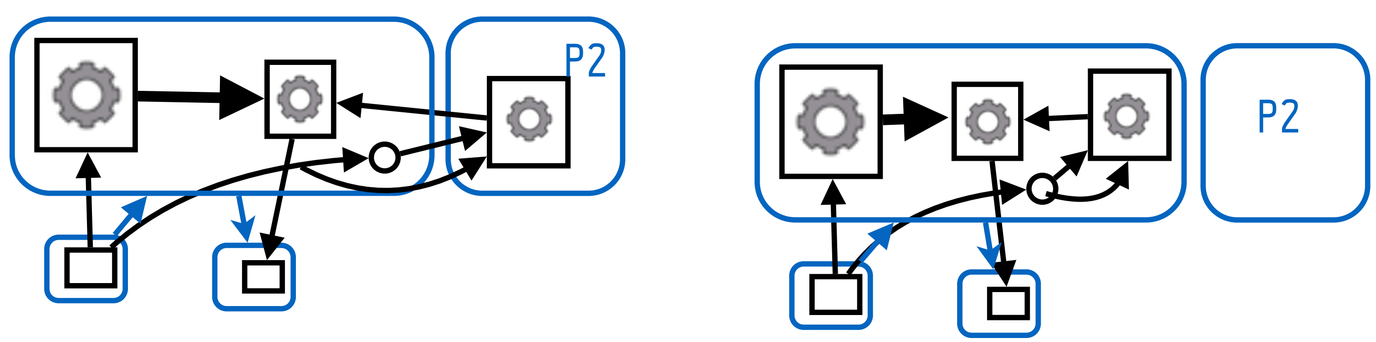
power consumption [W]
architecture graph

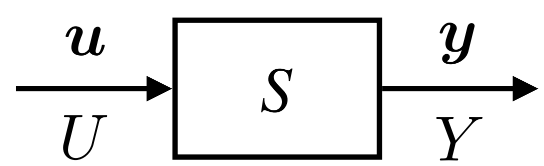
nodes: components
edges: signals



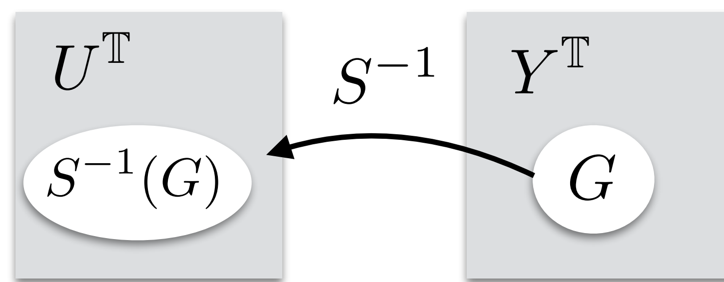


nodes: processors
edges: network links



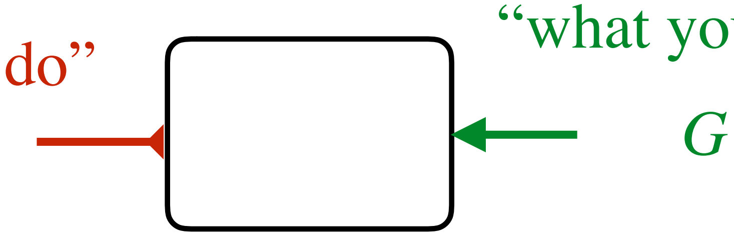


$$S \subseteq U^{\mathbb{T}} \times Y^{\mathbb{T}}$$



"what you need to do"

"what you want to see"



$\mathcal{R} = \langle \mathsf{powerset}(U^{\mathbb{T}}), \supseteq \rangle$

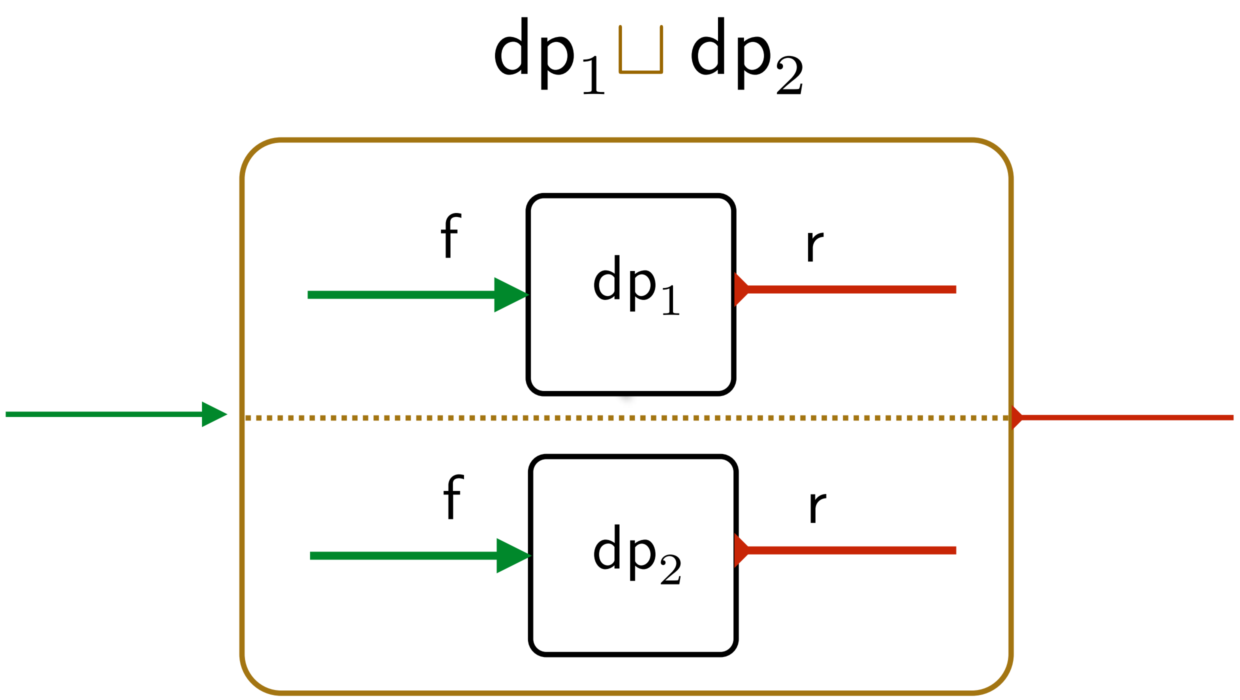
$\mathcal{F} = \langle \mathsf{powerset}(Y^{\mathbb{T}}), \supseteq \rangle$

- **Convergence speed?**
  - Linear, quadratic, … convergence do not make sense **without a metric.**

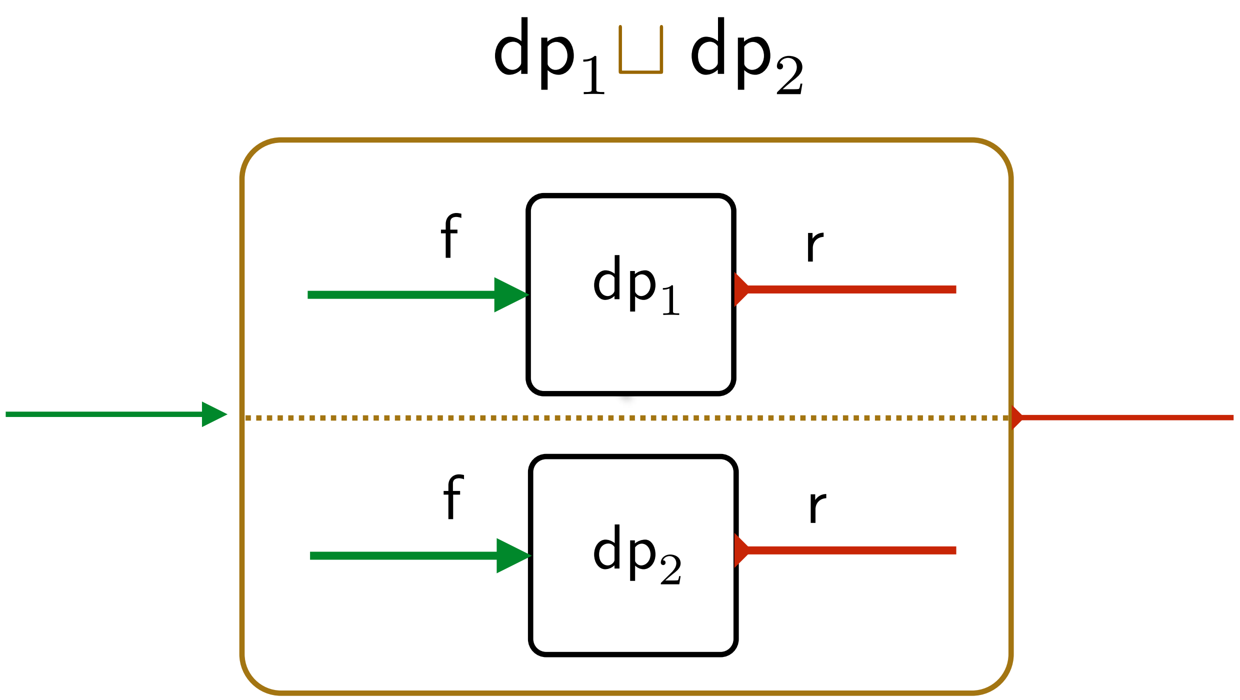
- **Option 1:** add a metric (additional assumptions)
- **Option 2:** derive bounds for the “pure” theory that are **parametrization invariant.**

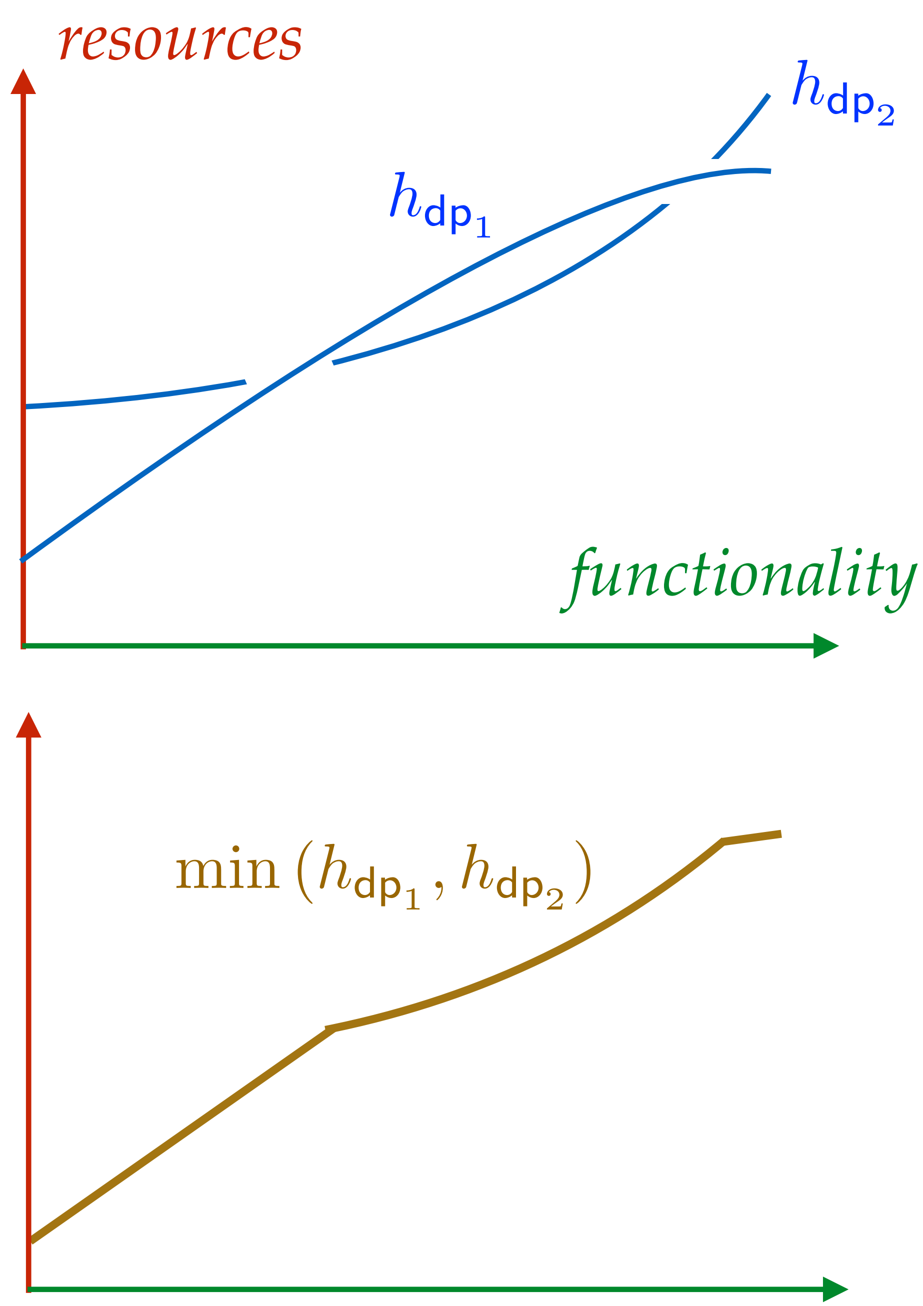
group = order isomorphisms

- Coproduct **- "Choose between technologies"**



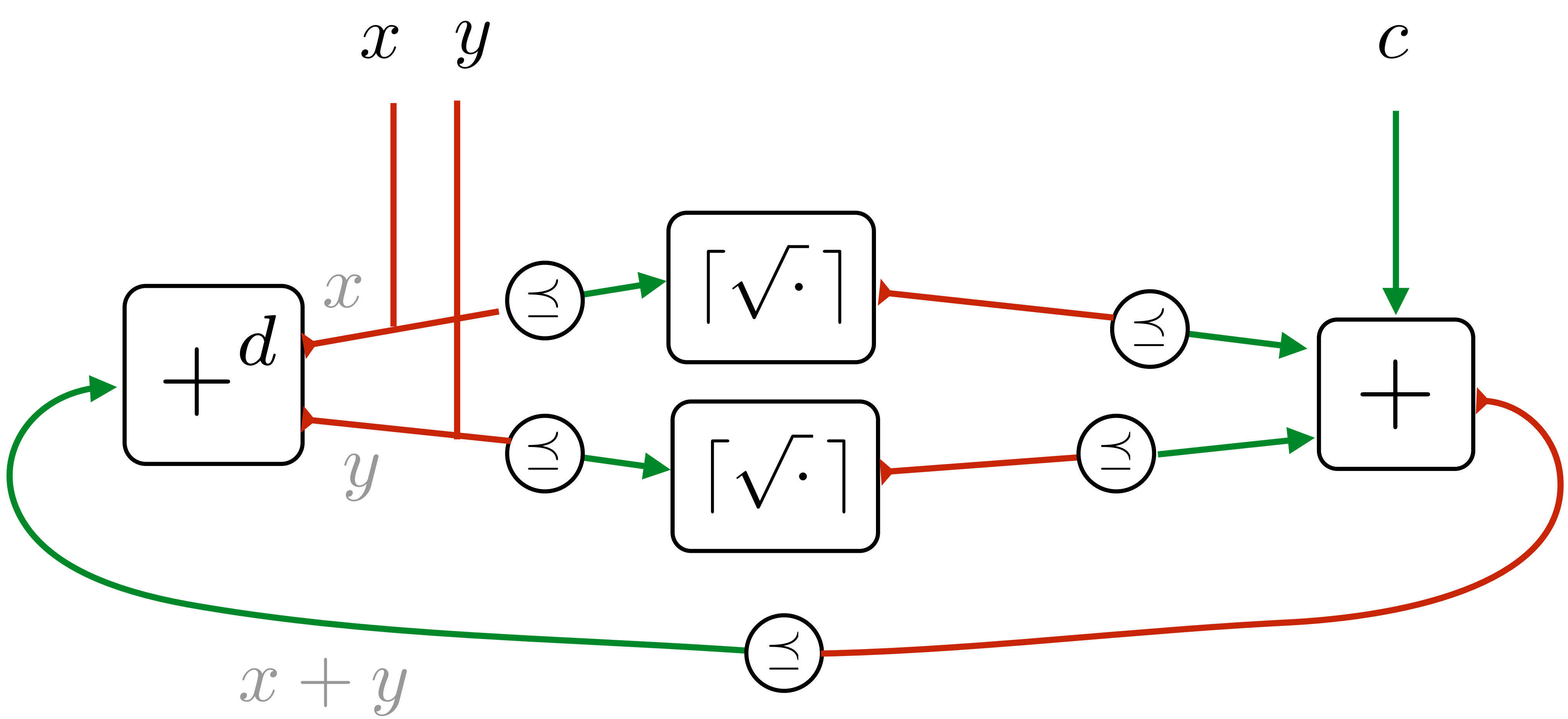
- Coproduct **- "Choose between technologies"**





$$h_{\mathsf{dp}_1 \sqcup \mathsf{dp}_2}(\mathsf{f}) = \operatorname*{Min}_{\preceq_{\mathcal{R}}} h_{\mathsf{dp}_1}(\mathsf{f}) \cup h_{\mathsf{dp}_2}(\mathsf{f})$$

$$\begin{aligned} &\underset{\preceq_{\mathbb{N}\times\mathbb{N}}}{\text{Min}} \;\langle x, y\rangle \\ &\text{s.t.} \quad x + y \geq \lceil \sqrt{x}\,\rceil + \lceil \sqrt{y}\,\rceil + c \end{aligned}$$

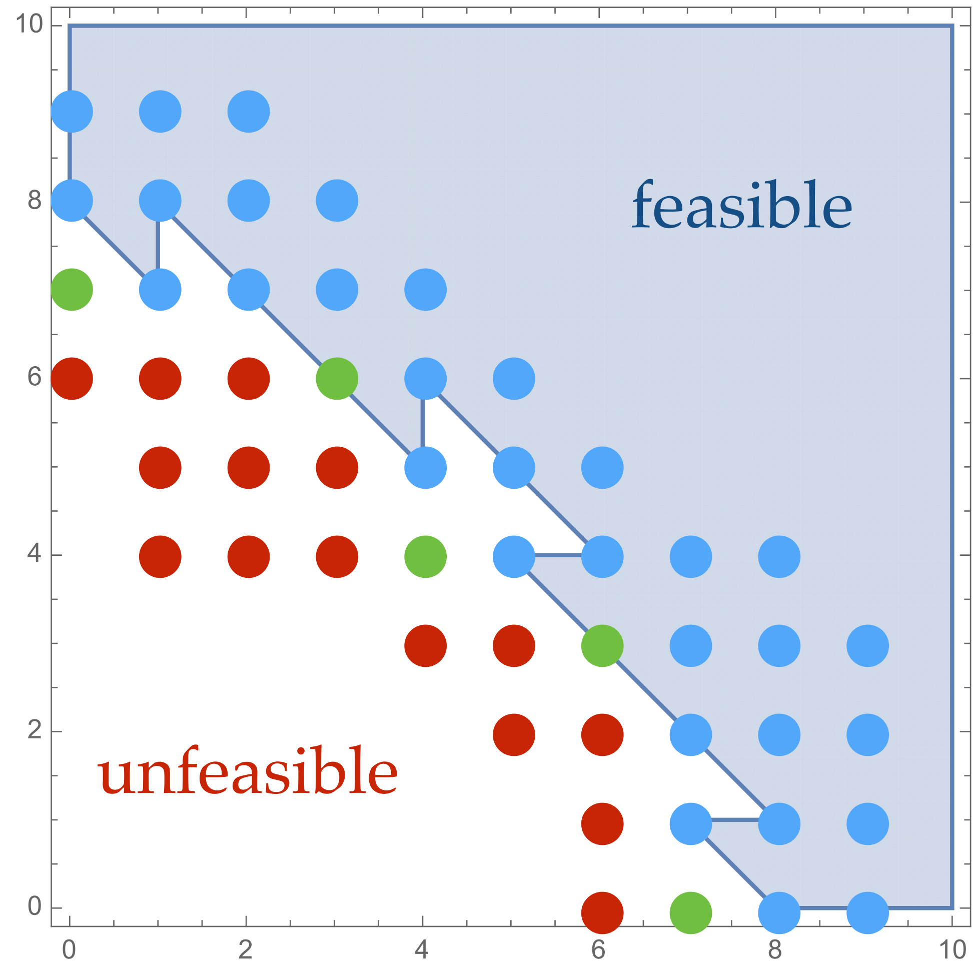


$$\operatorname*{Min}_{\preceq_{\mathbb{N}\times\mathbb{N}}} \langle x, y \rangle$$

$$\text{s.t.} \quad x + y \geq \lceil \sqrt{x} \rceil + \lceil \sqrt{y} \rceil + c$$

$$c = 4$$

minimal solutions

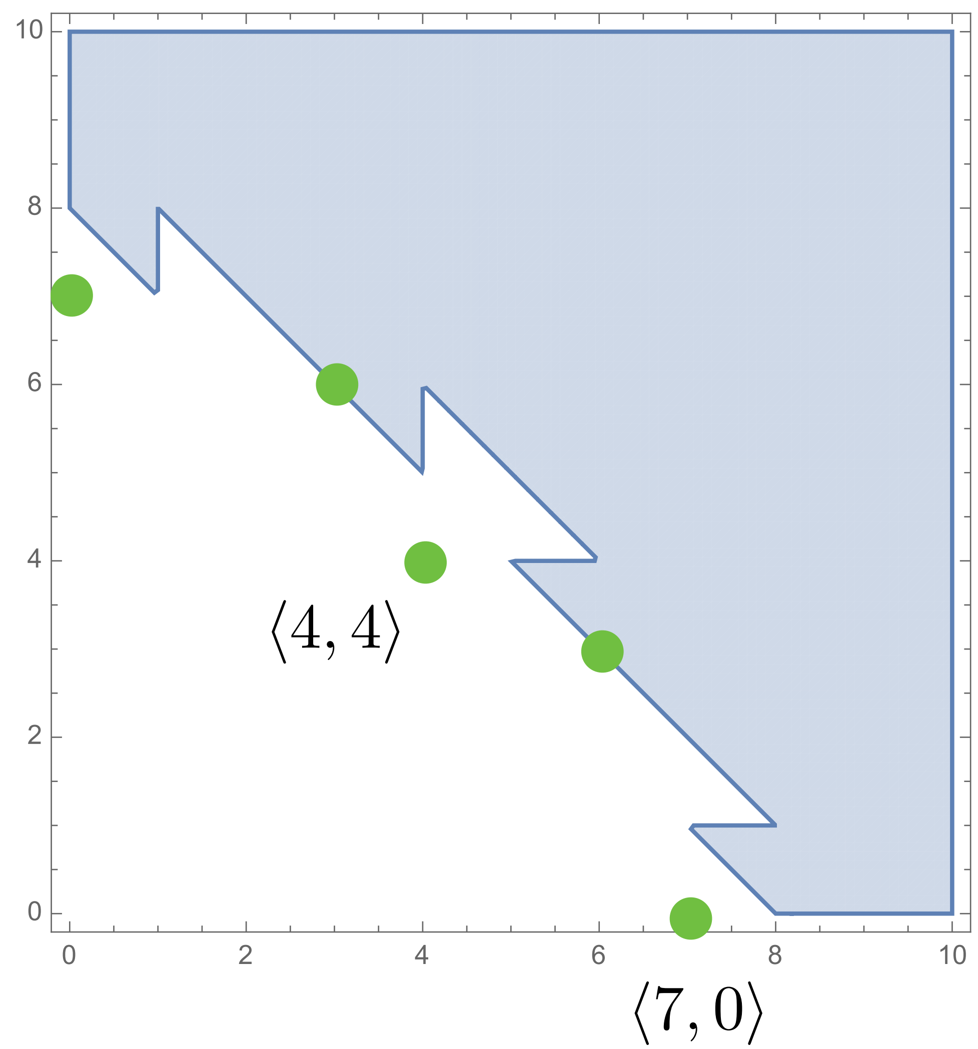


$$\operatorname*{Min}_{\preceq_{\mathbb{N}\times\mathbb{N}}} \langle x, y\rangle$$

$$\text{s.t.} \quad x + y \geq \lceil \sqrt{x}\, \rceil + \lceil \sqrt{y}\, \rceil + c$$

$c = 4$

minimal solutions

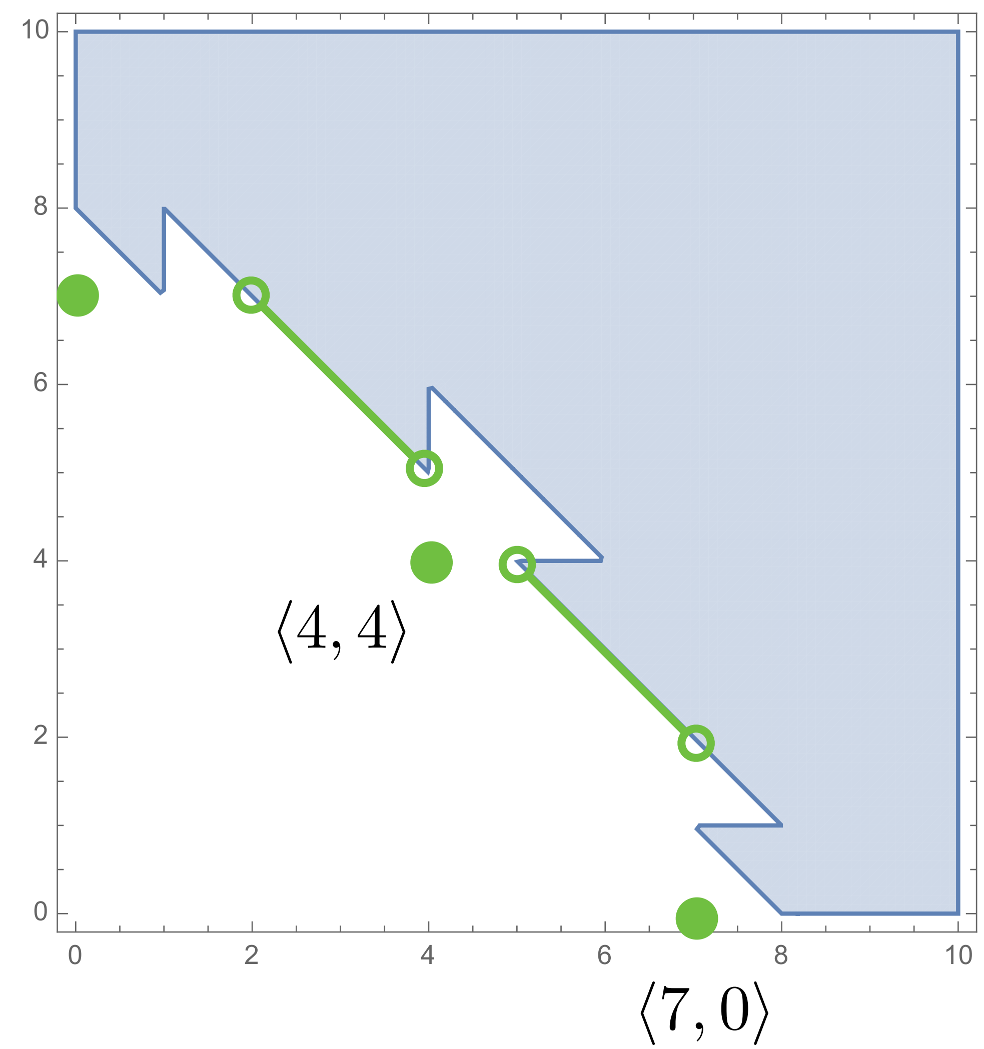


$$\underset{\preceq_{\mathbb{R}\times\mathbb{R}}}{\mathrm{Min}}\ \langle x, y\rangle$$

$$\text{s.t.}\quad x + y \geq \lceil \sqrt{x}\, \rceil + \lceil \sqrt{y}\, \rceil + c$$

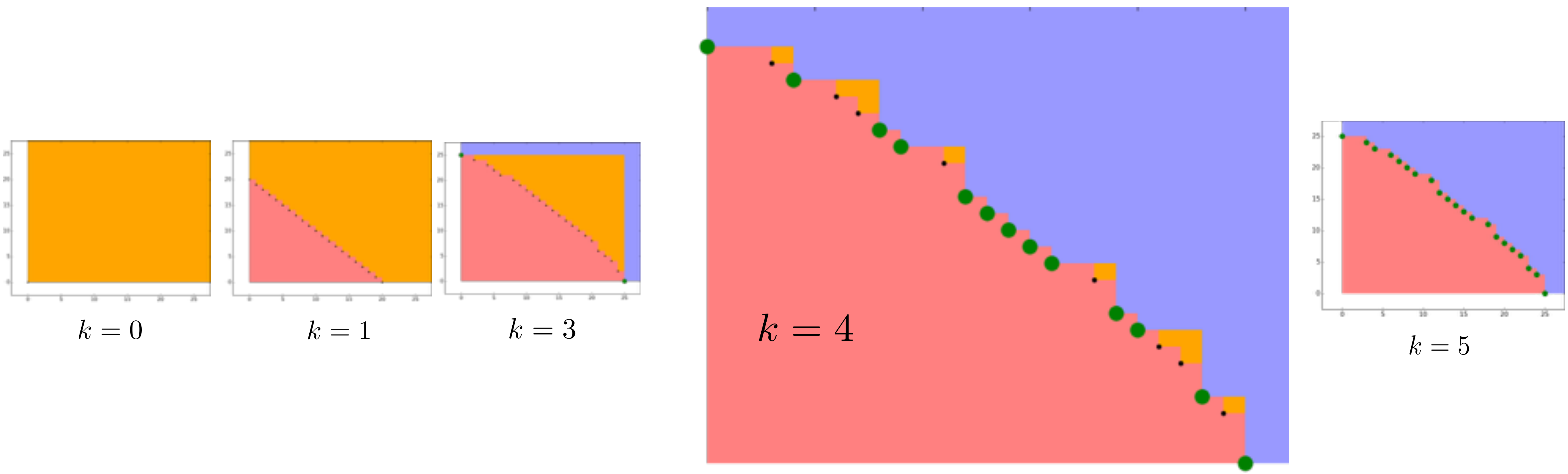
$c = 4$

minimal solutions



- The **set of all minimal solutions** can be found as a **fixed point**

$$
\begin{array}{ll}
\displaystyle\min_{\preceq_{\mathbb{N}\times\mathbb{N}}} & \langle x, y\rangle \\
\text{s.t.} & x + y \geq \lceil\sqrt{x}\,\rceil + \lceil\sqrt{y}\,\rceil + 20
\end{array}
$$



current antichain

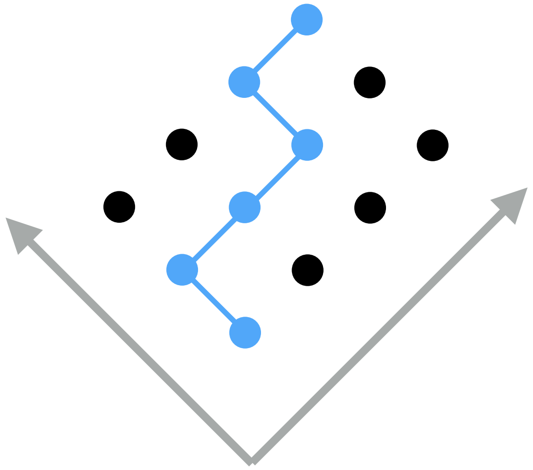
unfeasible

unclassified

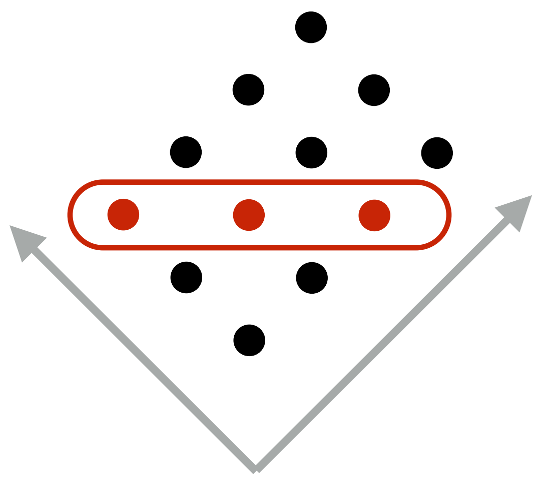
minimal solutions

"Don't care" - There are no feasible points here not dominated by ones already found.

- **Height** of a poset:
  maximum cardinality of its **chains**.



- **Width** of a poset:
  maximum cardinality of its **antichains**.



- **Heights and widths of products**

$$\text{height}(\mathcal{P} \times \mathcal{Q}) = \text{height}(P) + \text{height}(Q) - 1$$

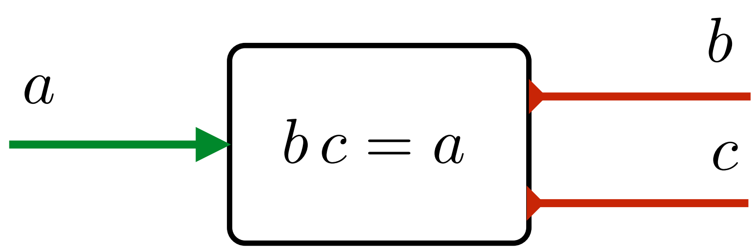
$$\text{width}(\mathcal{P})\text{width}(\mathcal{Q}) \leq \text{width}(\mathcal{P} \times \mathcal{Q}) \leq \min\{|\mathcal{P}|\text{width}(\mathcal{Q}), |\mathcal{Q}|\text{width}(\mathcal{P})\}$$

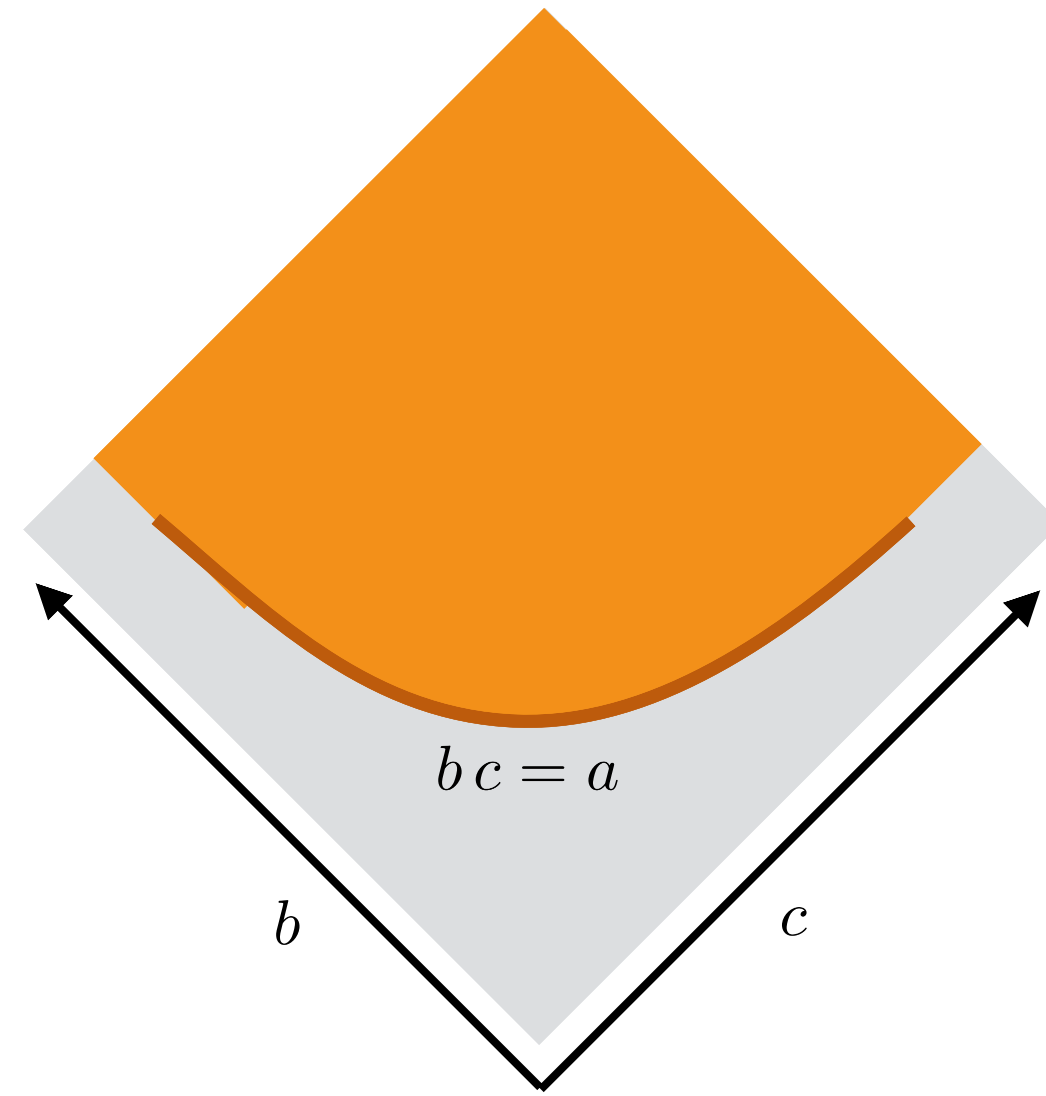
Griggs. *Maximum antichains in the product of chains. 1984*

Bezrukov, Roberts. *On antichains in product posets. 2008*

.

- Dealing with **infinite solutions**.



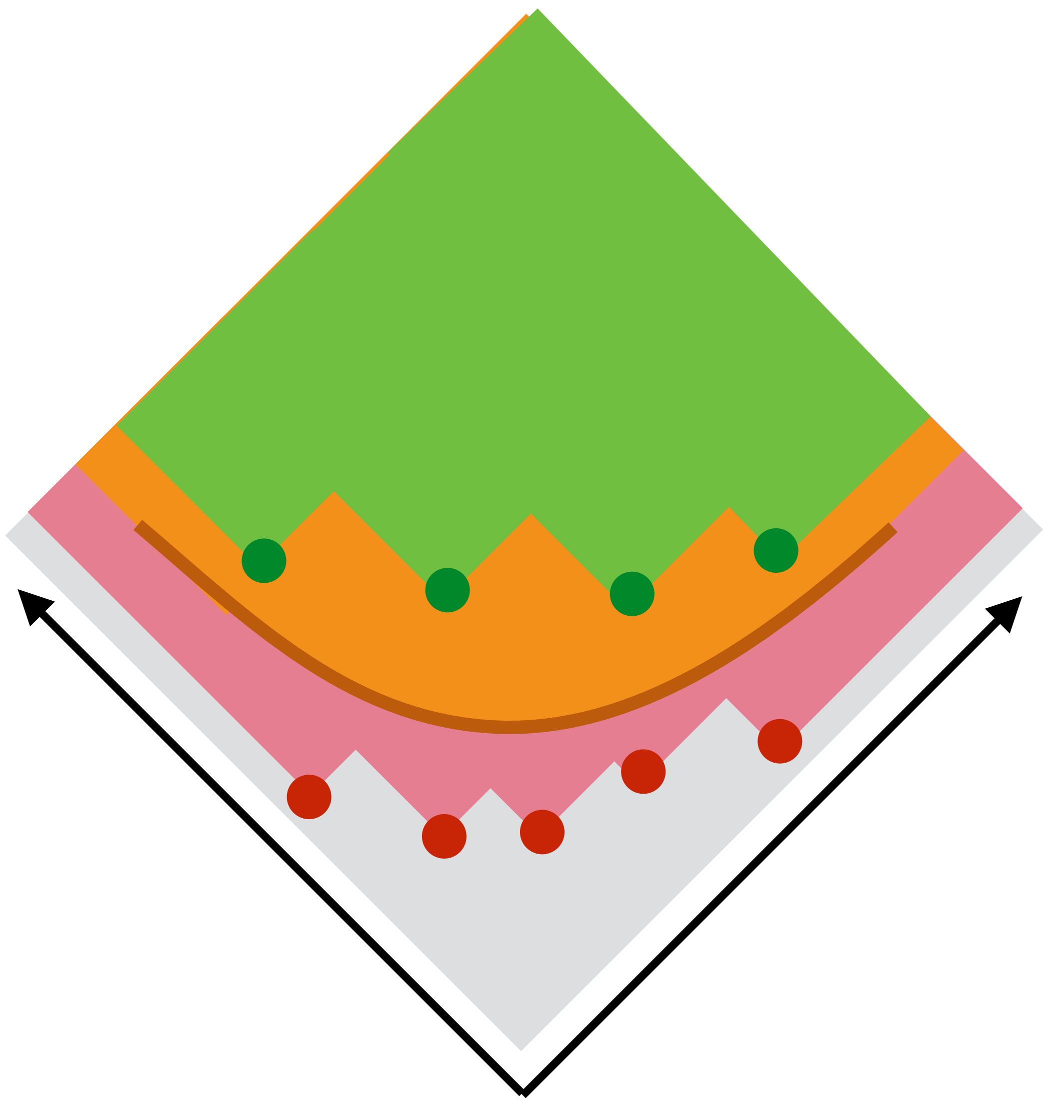


  - **Option 1:** Restrict attention to sets that are finitely representable
  - **Option 2:** Work out generic approximation bounds.

- **Finite lower/upper** (inner / outer) **approximations.**

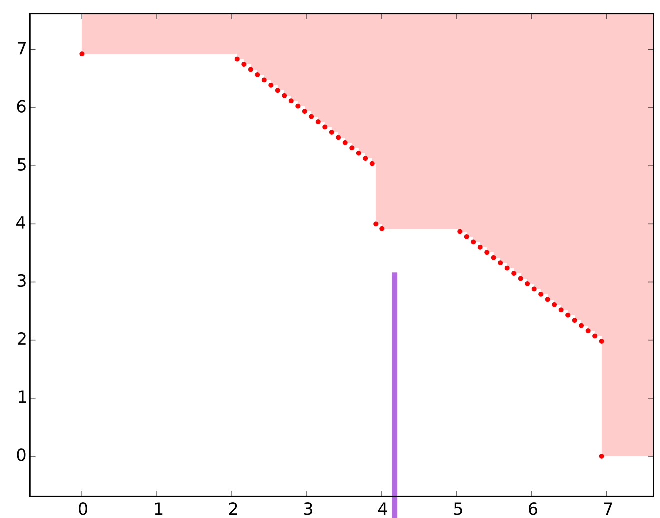
$$h_L(n_L, \mathsf{f}) \preceq h(\mathsf{f}) \preceq h_U(n_U, \mathsf{f})$$

$$\lim_{n_L, n_U \to \infty} h_L(n_L, \mathsf{f}) = h(\mathsf{f}) = h_U(n_U, \mathsf{f})$$



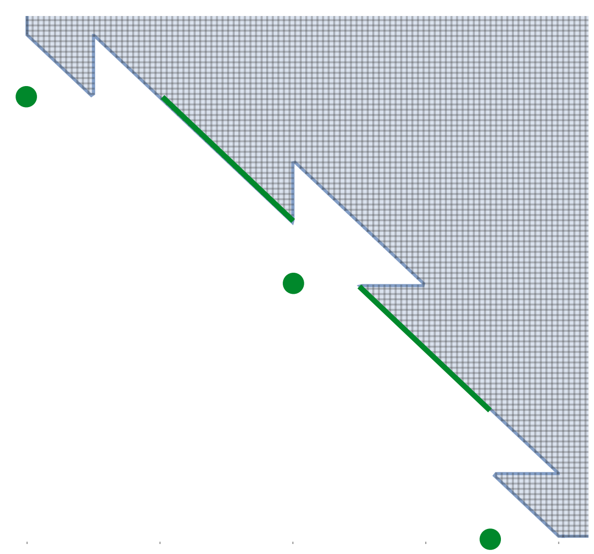
$$\underset{\preceq_{\mathbb{R}\times\mathbb{R}}}{\text{Min}}\ \langle x, y\rangle$$
$$\text{s.t.}\quad x + y \geq \lceil\sqrt{x}\,\rceil + \lceil\sqrt{y}\,\rceil + c$$

computed
finite lower bound



$\preceq_{\mathsf{A}\mathcal{R}}$

exact solution



$\preceq_{\mathsf{A}\mathcal{R}}$

computed
finite upper bound