Applied Compositional Thinking for Engineers

Session 8

Design

What is design?

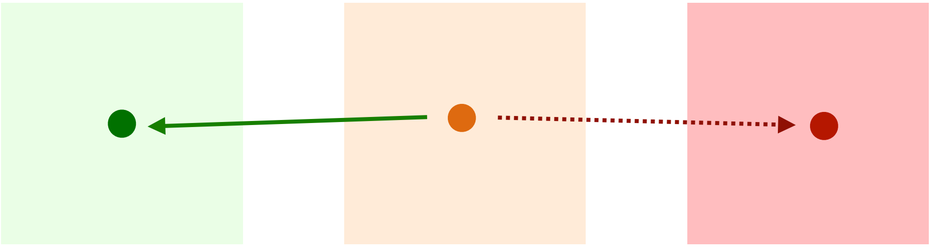
- We take a broad view of what design is, the same as Herbert Simon:

> ***Engineers are not the only professional designers.***
> ***Everyone designs who devises courses of action aimed at changing existing situations into preferred ones.***
> *The intellectual activity that produces material artifacts is no different fundamentally from the one that prescribes remedies for a sick patient or the one that devises a new sales plan for a company or a social welfare policy for a state. Design, so construed, is the core of all professional training; it is the principal mark that distinguishes the professions from the sciences. Schools of engineering, as well as schools of architecture, business, education, law, and medicine, are all centrally concerned with the process of design.*

Herbert Simon, The sciences of the artificial, Chapter 5

An abstract view of design problems

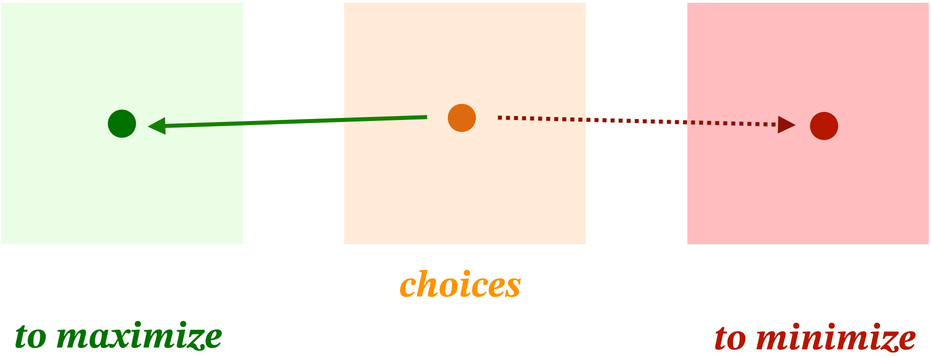
- Across fields, design or synthesis problems are defined with 3 spaces:
  - **implementation space**: the options we can choose from;
  - **functionality space**: what we need to provide/achieve;
  - **requirements/costs space**: the resources we need to have available;



functionality *(provided)*	***implementations***	***requirements, resources*** *required*
desired behavior	*choices*	*costs*
specifications	*plans*	*dependencies*
objectives	*blueprints*	
	decision variables	
"function"	*"form"*	*"function"*
"conclusions"	*"proof"*	*"assumptions"*

An abstract view of design problems

- Across fields, design or synthesis problems are defined with 3 spaces:
  - **implementation space**: the options we can choose from;
  - **functionality space**: what we need to provide/achieve;
  - **requirements/costs space**: the resources we need to have available;



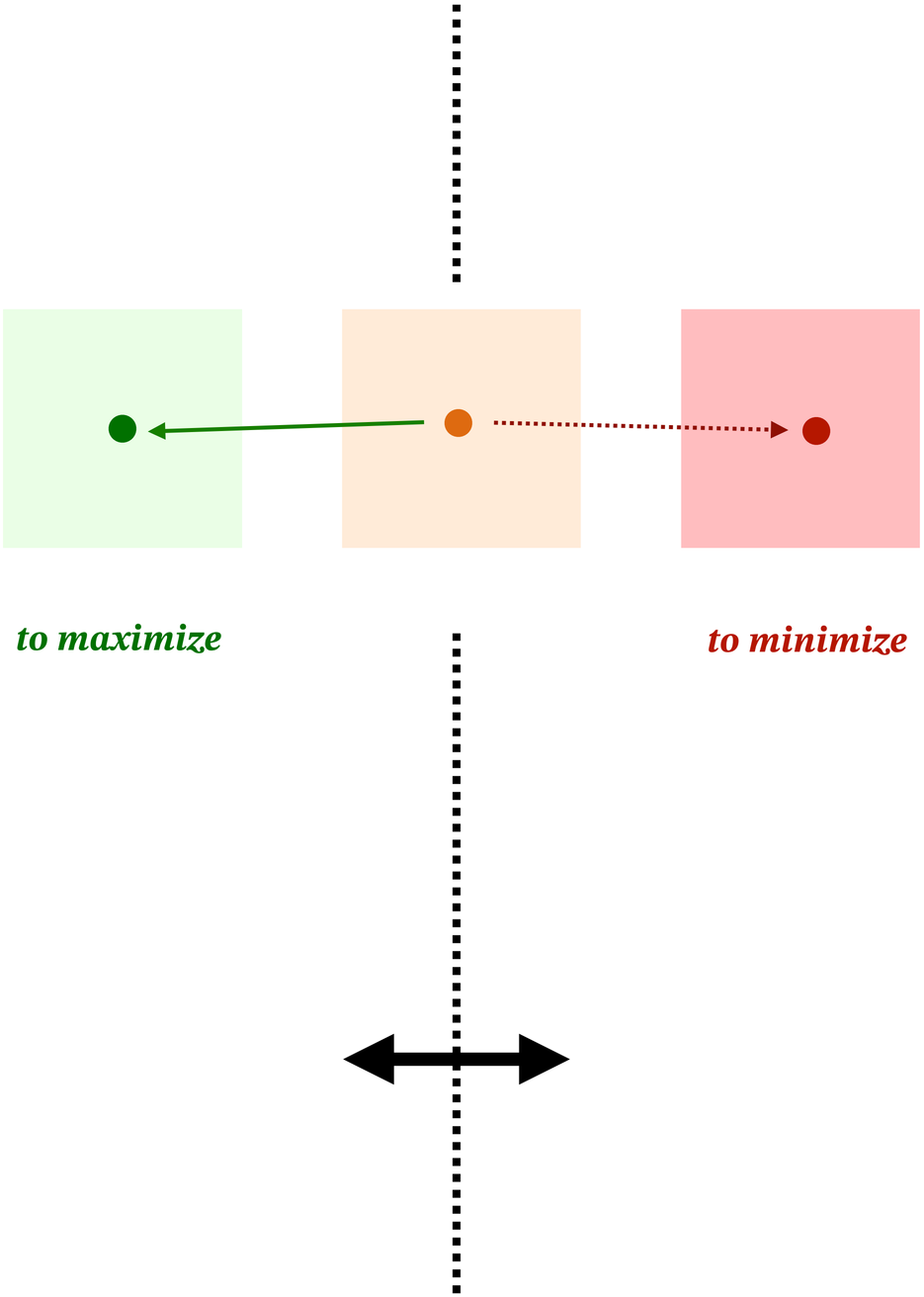
$\langle \mathcal{F}, \leq_{\mathcal{F}} \rangle$

any partially ordered set

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

any partially ordered set

A symmetric theory



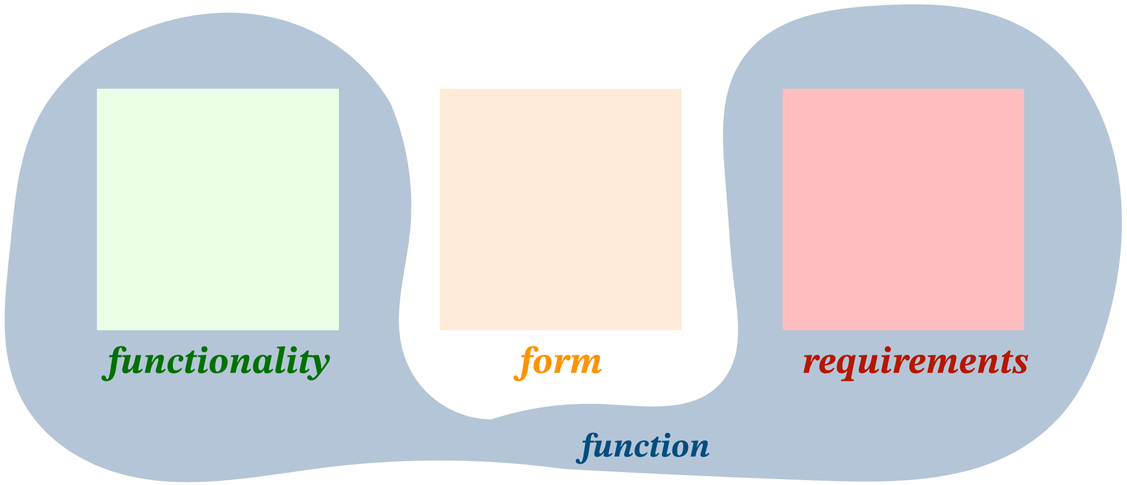
Form ever follows function

Whether it be the sweeping eagle in his flight, or the open apple-blossom, the toiling work-horse, the blithe swan, the branching oak, the winding stream at its base, the drifting clouds, over all the coursing sun, ***form ever follows function, and this is the law****. Where function does not change, form does not change. The granite rocks, the ever-brooding hills, remain for ages; the lightning lives, comes into shape, and dies, in a twinkling.*

It is the pervading law of all things organic and inorganic, of all things physical and metaphysical, of all things human and all things superhuman, of all true manifestations of the head, of the heart, of the soul, that the life is recognizable in its expression, that form ever follows function. This is the law.

Louis Sullivan

"function" conflates two dual aspects



Function beyond the strictly technical meaning

- **Law of successful products**: at equilibrium, in an efficient and free market, no product completely dominates another by both functionality and costs.
  (Otherwise, the dominated product wouldn't sell.)

Casio F-91W

Patek Philippe

"watchiness"

tells time,
multiple alarms

tells the time

signaling

?

I am somebody
who can afford
a *Patek Philippe*
and I chose to
buy one.

price

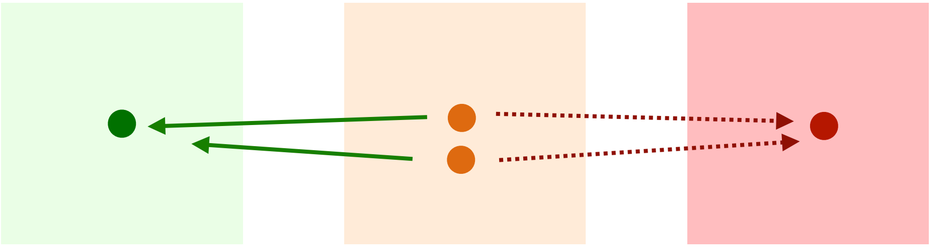
a lot

much less

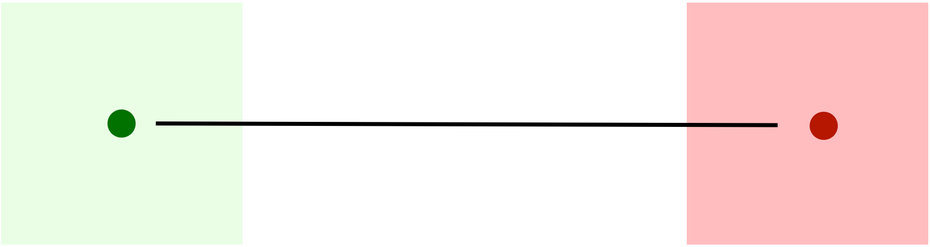
Transparent vs opaque models

- For the purpose of design, we **need to know how something is is done**, not just that it is possible to do something.

- We need to know what are the implementation(s), if any, that relate functionality and costs.



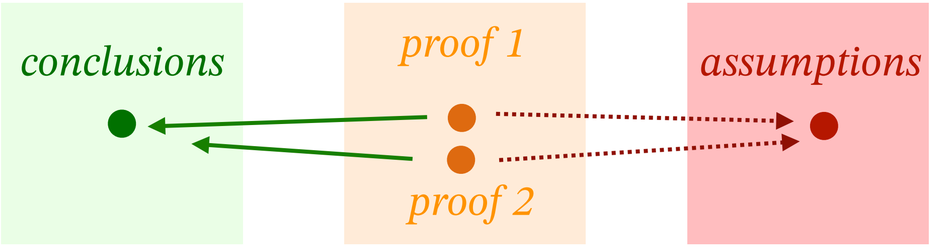
- Algorithmically, it will also be useful to consider a **direct feasibility relation** from functionality to costs.



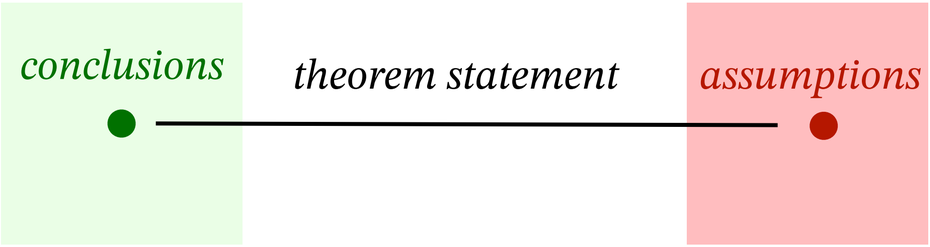
Engineering is constructive

- For the purpose of design, we **need to know how something is is done**, not just that it is possible to do something: **engineering is constructive**.

"I care about the proof"



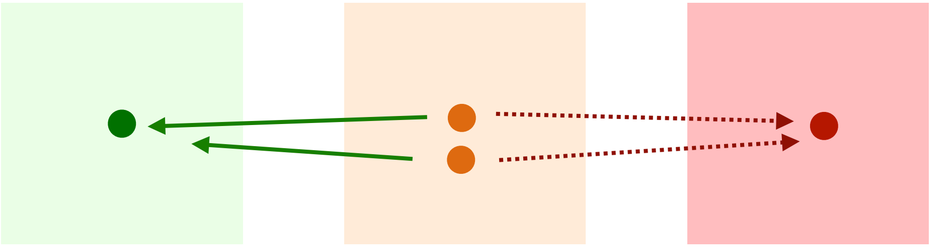
"I see the theorem as a black box"



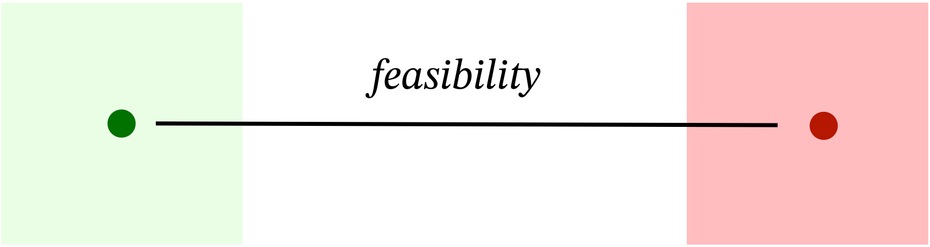
Engineering is constructive

- For the purpose of design, we **need to know how something is is done**, not just that it is possible to do something: **engineering is constructive**.

- We need to know what are the implementation(s), if any, that relate functionality and costs.



- For the algorithmic solution of co-design problem, **it will also be useful to consider a direct feasibility relation** from functionality to costs.

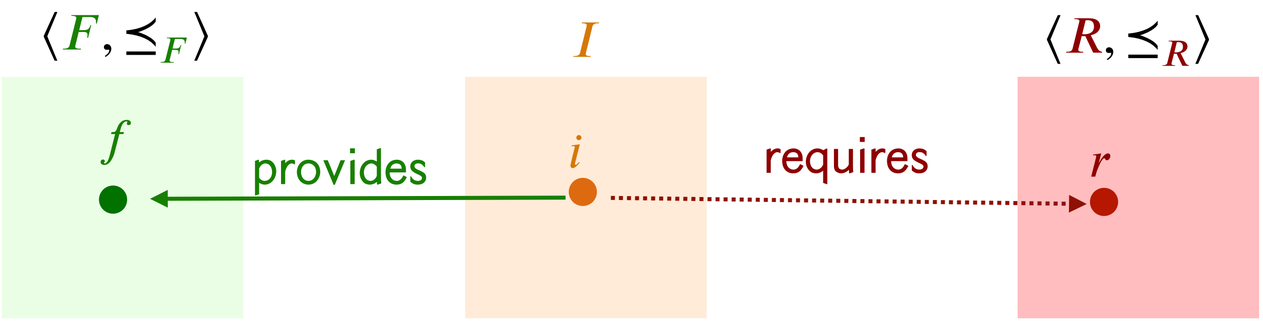


- We will call these **boolean profunctors.**

Design problem with implementation

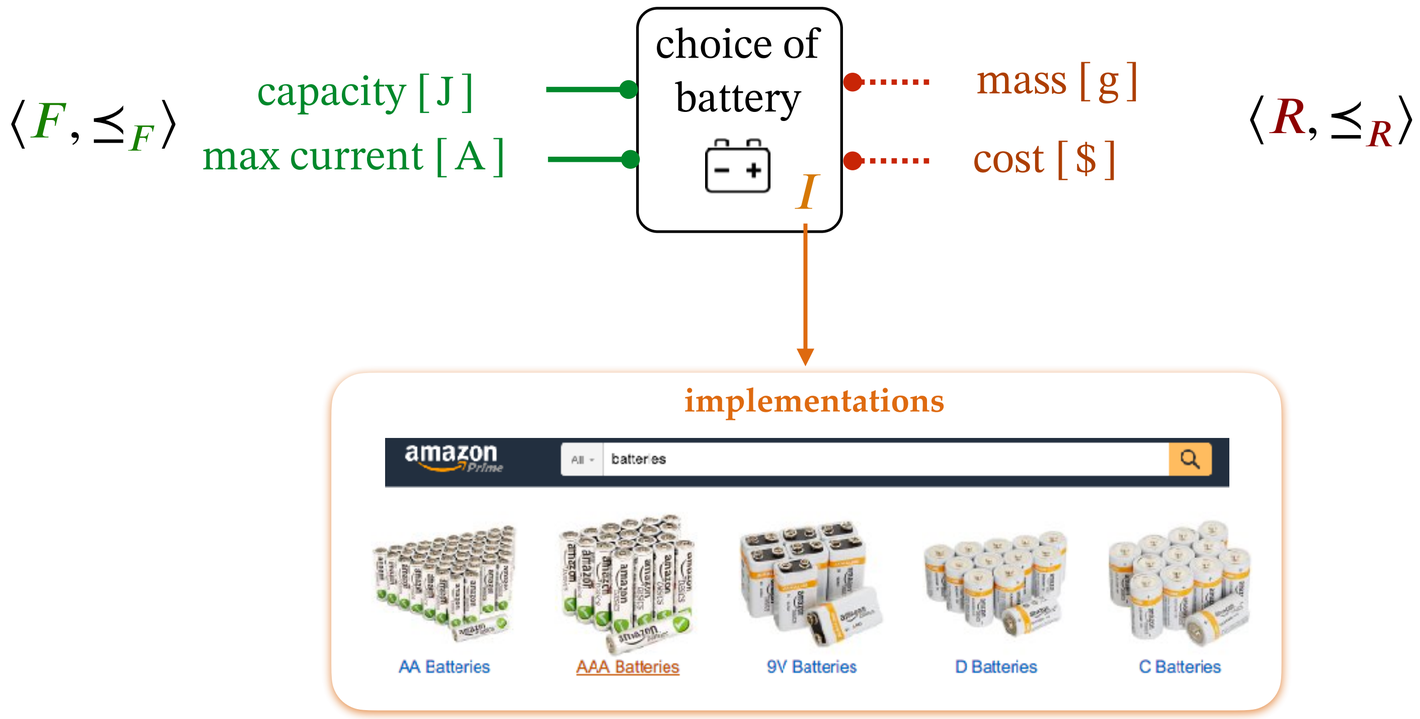
- A **design problem with implementation (DPI)** is defined as a tuple

$$\langle F, R, I, \text{provides}, \text{requires} \rangle$$

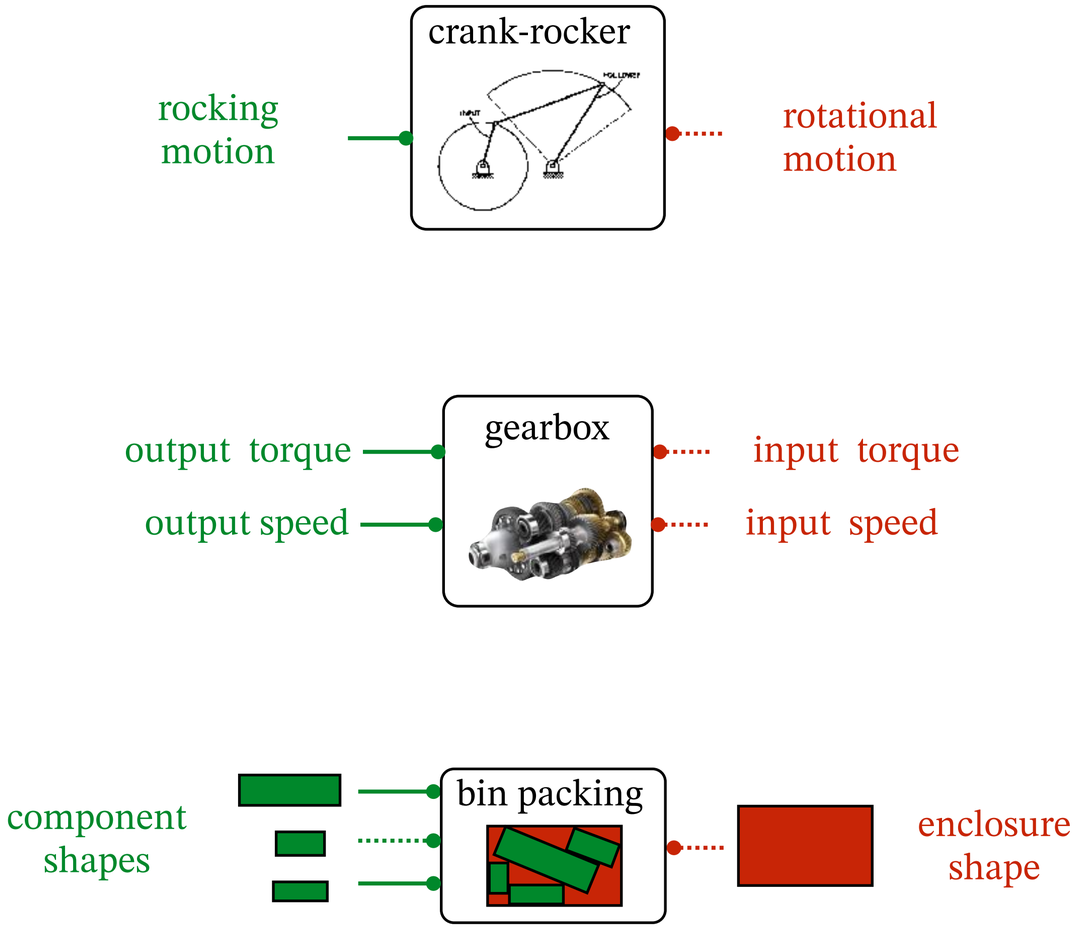


Graphical notation for DPIs

- We use this graphical notation:
  - functionality: **green continuous wires** on the left
  - requirements: **dashed red wires** on the right.



Mechanical Engineering



Inference

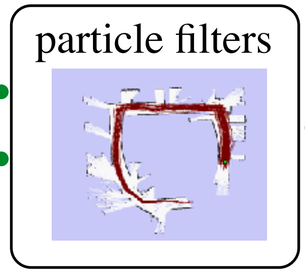
resolution

latency [s]

Alex Locher, Michal Perdoch and Luc Van Gool. Progressive prioritized multi-view stereo. CVPR 2016.

accuracy

robustness



# of particles

From DPI to feasibility relation

- From a DPI we can find a monotone function

$$r : F^{\mathrm{op}} \times R \to_{\mathbf{Pos}} \mathbf{Bool}$$

defined as follows:

$$\langle f^{\mathrm{op}}, r \rangle \quad \mapsto \quad \exists i \in I : (f \preceq_F \mathsf{provides}(i)) \wedge (\mathsf{requires}(i) \preceq_R r)$$

- Alternative: first define the set of all feasible implementations:

$$\langle f^{\mathrm{op}}, r \rangle \quad \mapsto \quad \{ i \in I : (f \preceq_F \mathsf{provides}(i)) \wedge (\mathsf{requires}(i) \preceq_R r) \}$$

then ask if non empty.

Boolean profunctors

- We call these ***boolean profunctors*** or ***design problems***

$$r : F^{\mathrm{op}} \times R \to_{\mathbf{Pos}} \mathbf{Bool}$$

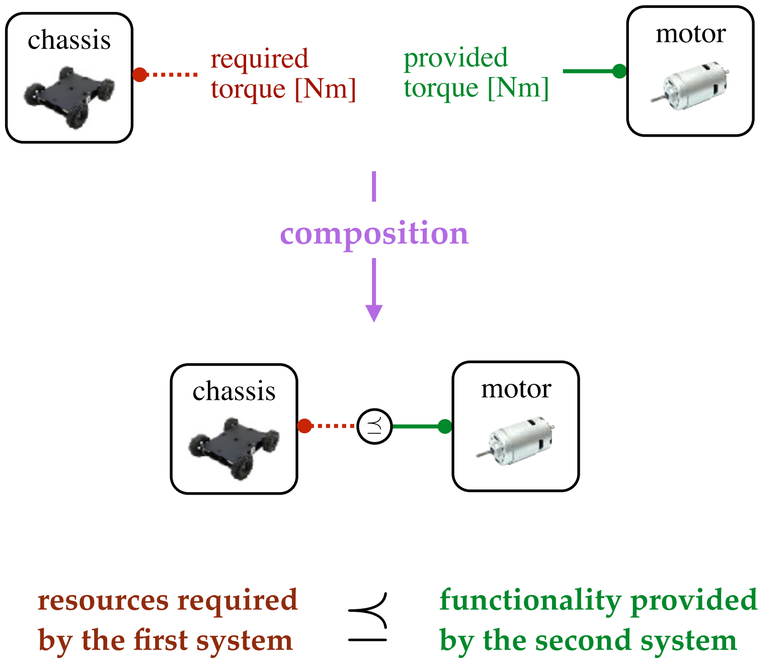
- There is a category of these, called **Feas** or **DP**.

- We use these slashed arrows:

$$r : F \nrightarrow R$$

Composition semantics

- Composition in DP reflects an intuitive notion of composition in engineering:

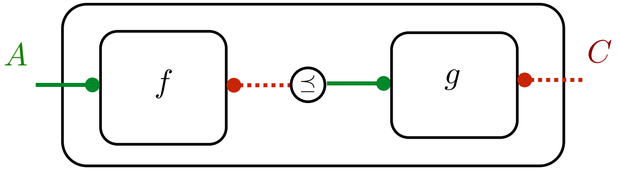


resources required by the first system $\preceq$ **functionality provided by the second system**

Defining composition for boolean profunctors

- Definition of composition:

$$\frac{f : A \nrightarrow B \qquad g : B \nrightarrow C}{(f\,;g) : A \nrightarrow C}$$



Defining composition for boolean profunctors

- Definition of composition:

$$\frac{f : A^{\mathrm{op}} \times B \to_{\mathbf{Pos}} \mathbf{Bool} \qquad g : B^{\mathrm{op}} \times C \to_{\mathbf{Pos}} \mathbf{Bool}}{(f;\, g) : A^{\mathrm{op}} \times C \to_{\mathbf{Pos}} \mathbf{Bool}}$$

$$\langle a^{\mathrm{op}}, c\rangle \mapsto \bigvee_{b_1 \leq b_2} f(a^{\mathrm{op}}, b_1) \wedge g(b_2^{\mathrm{op}}, c)$$

- Identities:

$$\mathrm{Id}_A : A^{\mathrm{op}} \times A \to_{\mathbf{Pos}} \mathbf{Bool}$$

$$\langle {a_1}^{\mathrm{op}}, a_2\rangle \;\mapsto\; (a_1 \leq_A a_2)$$

Boolean Profunctors as generalization of relations

- A relation is a map

$$r : A \times B \to_{\mathbf{Set}} \mathbf{Bool}$$

- A boolean profunctor is a map

$$\overline{f} : A^{\mathrm{op}} \times B \to_{\mathbf{Pos}} \mathbf{Bool}$$

- Boolean profunctors are generalization of relations.
- Are profunctors special relations, or are relations special profunctors?
- Which one is true?
  - DP is a subcategory of Rel.
  - Rel is a subcategory of DP.

Profunctors

- **Profunctors** are functors of type

$$P : \mathbf{A}^{\mathrm{op}} \times \mathbf{B} \to_{\mathbf{Cat}} \mathbf{Set}$$

where **A**, **B** are generic categories.

- You can see profunctors as a general case of Boolean profunctors:
  - Think of Bool as the subcategory of Set consisting of the empty set and a singleton.

$$\begin{tikzcd} \top \arrow[r, phantom, "\in"] & \{\circ\} \\ \bot \arrow[u] \arrow[r, phantom, "\in"] & \emptyset \arrow[u] \end{tikzcd} \qquad \cong$$

  - In the other direction: consider the functor

nonempty: Set → Bool

The Hom Profunctor

- **Profunctors** are functors of type

$$P : \mathbf{A}^{\mathrm{op}} \times \mathbf{B} \to_{\mathbf{Cat}} \mathbf{Set}$$

  where **A**, **B** are generic categories.

- Claim: **Hom** can be seen as a profunctor:

$$\mathrm{Hom}_{\mathbf{C}} : \mathbf{C}^{\mathrm{op}} \times \mathbf{C} \to_{\mathbf{Cat}} \mathbf{Set}$$

- We are going to check this: technical but insightful.

Defining the Hom Profunctor

- Let's define the profunctor

$$\mathrm{Hom}_{\mathbf{C}} : \mathbf{C}^{\mathrm{op}} \times \mathbf{C} \to_{\mathbf{Cat}} \mathbf{Set}$$

  - It maps pairs of objects to their Hom-sets :

$$\langle x^{\mathrm{op}}, y \rangle \mapsto \mathrm{Hom}_{\mathbf{C}}(x; y)$$

  - What does it do to a morphism?

Defining the Hom Profunctor

- Let's define the profunctor

$$\mathrm{Hom}_{\mathbf{C}} : \mathbf{C}^{\mathrm{op}} \times \mathbf{C} \to_{\mathbf{Cat}} \mathbf{Set}$$

  - It maps pairs of objects to their Hom-sets :

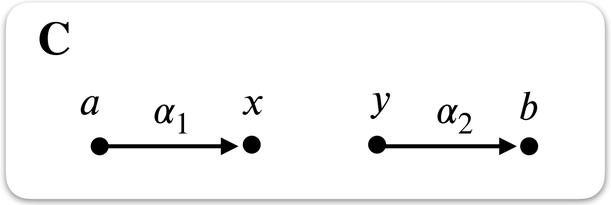
$$\langle x^{\mathrm{op}}, y \rangle \mapsto \mathrm{Hom}_{\mathbf{C}}(x; y)$$

  - What does it do to a morphism?

morphism in $\mathbf{C}^{\mathrm{op}} \times \mathbf{C}$

$$f = \langle \alpha_1^{\mathrm{op}}, \alpha_2 \rangle \longmapsto \begin{array}{l} \mathrm{Hom}_{\mathbf{C}}\, f : \mathrm{Hom}_{\mathbf{C}}(x; y) \to \mathrm{Hom}_{\mathbf{C}}(a; b) \\ z \mapsto (\alpha_1 \fatsemi z \fatsemi \alpha_2) \end{array}$$

morphism in **Set**



Checking that Hom is a profunctor

- Let's check the property:

$$F(f \fatsemi g) = F(f) \fatsemi F(g)$$

$$f = \langle \alpha_1^{\mathrm{op}}, \alpha_2 \rangle$$

$$g = \langle \beta_1^{\mathrm{op}}, \beta_2 \rangle$$

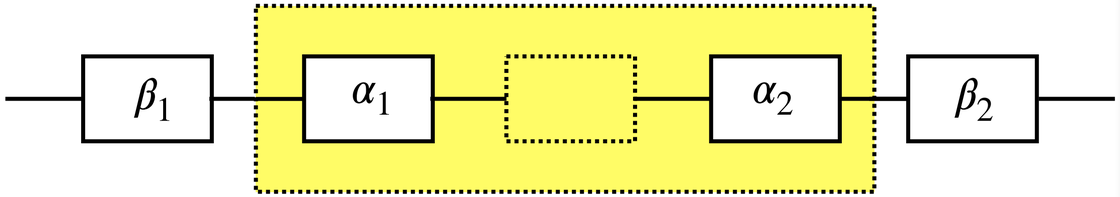
$$f \fatsemi g = \langle \alpha_1^{\mathrm{op}} \fatsemi \beta_1^{\mathrm{op}}, \alpha_2 \fatsemi \beta_2 \rangle$$
$$= \langle (\beta_1 \fatsemi \alpha_1)^{\mathrm{op}}, \alpha_2 \fatsemi \beta_2 \rangle$$

$\mathrm{Hom}_{\mathbf{C}}\, f$

$\mathrm{Hom}_{\mathbf{C}}\, g$

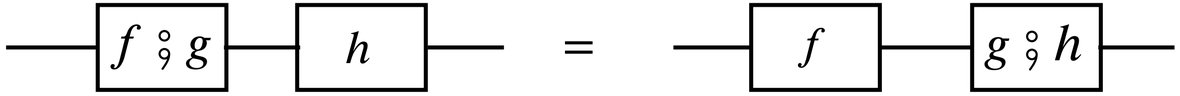
$\mathrm{Hom}_{\mathbf{C}}(f \fatsemi g)$

$\mathrm{Hom}_{\mathbf{C}}\, f \fatsemi \mathrm{Hom}_{\mathbf{C}}\, g$

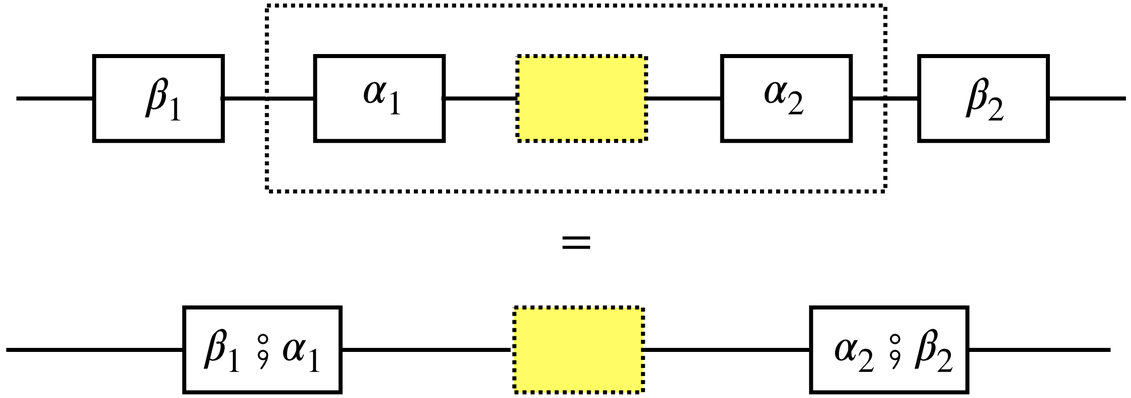


Associativity

$$f \fatsemi (g \fatsemi h) = (f \fatsemi g) \fatsemi h$$



$$(\beta_1 \fatsemi \alpha_1) \fatsemi x \fatsemi (\alpha_2 \fatsemi \beta_2) = \beta_1 \fatsemi (\alpha_1 \fatsemi x \fatsemi \alpha_2) \fatsemi \beta_2$$



$$(\beta_1 \fatsemi \alpha_1); x; (\alpha_2 \fatsemi \beta_2) = \beta_1; (\alpha_1; x; \alpha_2); \beta_2$$

DPI as profunctors?

- From a DPI we can find a monotone function

$$r : F^{\mathrm{op}} \times R \to_{\mathbf{Pos}} \mathrm{PowerSet}(I)$$

defined as follows:

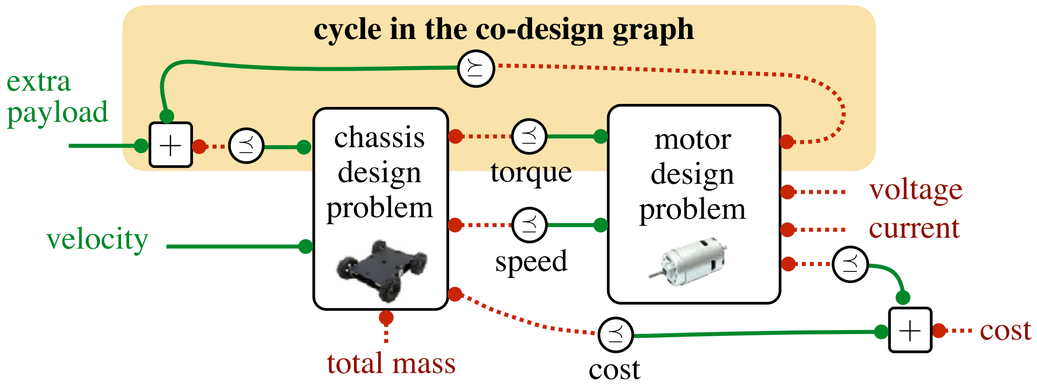
$$\langle f^{\mathrm{op}}, r \rangle \quad \mapsto \quad \{ i \in I : (f \preceq_F \mathsf{provides}(i)) \wedge (\mathsf{requires}(i) \preceq_R r) \}$$

- Can we make this into a profunctor?

Coming up: Co-design

- So far, we have only composed morphisms in "**series**".

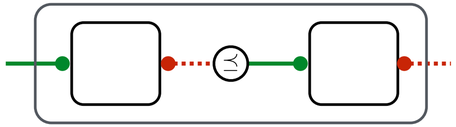
- In **co-design** we ask how to optimize over very complex graphs:



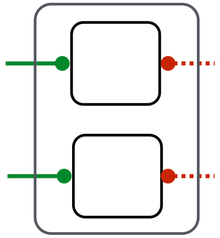
Coming up: Co-design

- So far, we have only composed morphisms in **“series”**.
- To give a categorical semantics to complex co-design diagrams, we will define:
  - monoidal categories (**parallelism**)
  - traced monoidal categories (**feedback**)
  - locally posetal / lattical categories (**and**, **or**)

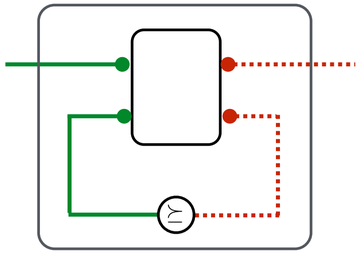
“series”



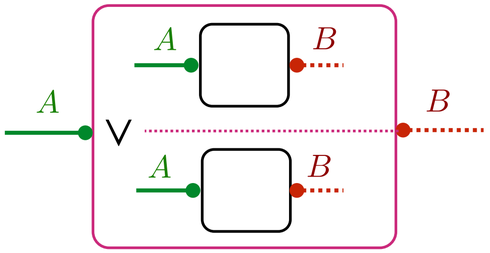
“parallel”



“feedback”



“choose between two options”



“convince two experts”