Q:Applied Compositional Thinking for Engineers (ACT4E)

Session 9 - Feasibility - Q&A
Questions & Answers

Q: Where does ethics fit in the abstract view of design problems?

Q: Where does aesthetics fit in the abstract view of design?

GZ: The above questions are both interesting to be answered at the end. For the case of aesthetics, I can imagine a personal functional requirement to express aesthetics

MH: It will be interesting to see that related to a category

Q: Why does the bin packing example have a dotted green line ?
GZ: It should be a solid line

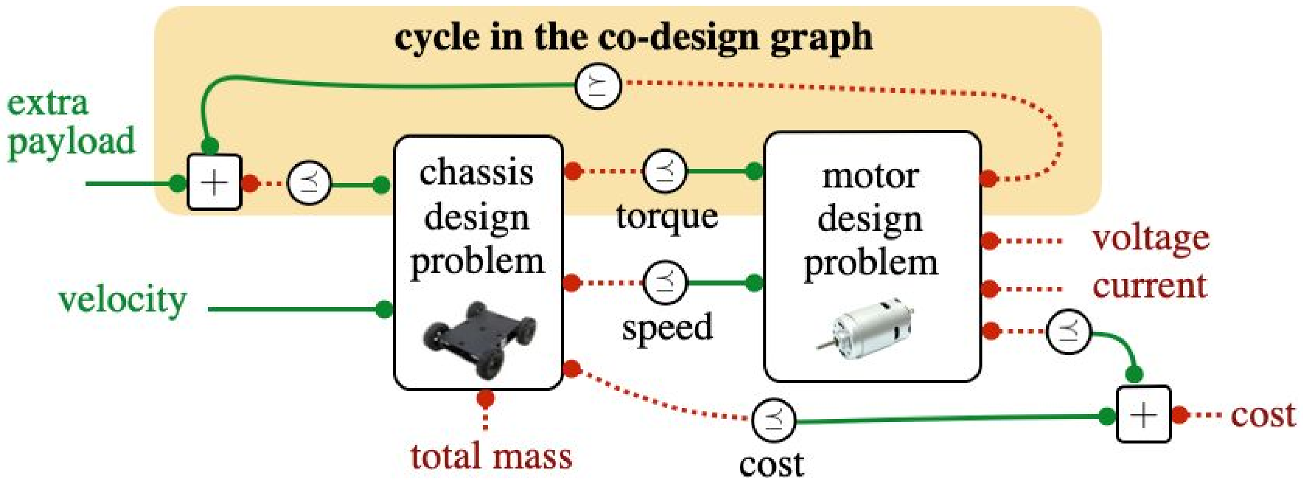
Q: composition of profunctors is not really as composition in categories right? In particular composed profunctors have not common codomain and domain ?

GZ: In couple of slides you will see that this composition is precisely the composition in the category DP, which has posets as objects and boolean profunctors as morphisms. A morphism here is written as F-|->R.

NM: The composition of profunctors is defined by the existence of a span: https://ncatlab.org/nlab/show/span. These spans would be defined in a category that somehow includes all possible domains and codomains as objects (it may be simpler than the cartesian product of the set of domains and codomains since they sometime coincide).
- Alternatively there is a definition using something called a left Kan extension (over the yoneda functor) which might be more rigorous under composition of profunctors: https://en.wikipedia.org/wiki/Profunctor. This uses a trick that the functors P: **Cop** x **D** -> **set** are equivalent to the functors P: **D**-> **set** ^(**Cop**), and then does something like the span before to define the composition of profunctors. [Cop is the opposite category of C]
- Also note that the composition in a category can refer to any set of rules used to define the composition of two morphisms, even just a random truth table, just as long as these rules respect associativity.

Q: Does this mean the category of functionality and the category of requirements have the same set of objects? (or are possible the same categories)
GZ: What do you mean with category of functionality and category of requirements?
OP: when you define the DPI you have a category of implementations with maps into a category of functionality and a category of requirements.
GZ: Yes, these are posets here (which are categories). They don't need to have the same set of objects.
OP: Though if you joint two strings you have an object B in requirements and an object B in your functionalities (which are joined by <=)
GZ: Ok, I now understand your question. Yes, composition requires that the codomain of the first morphism is the domain of the second one (exactly as in Set, you can compose two functions f:A->B and g:B->C if the Bs agree). The <= has the semantics "this first DP requires up to what is provided by the next DP".
This does not mean that F and R always have to be the same in general. As Andrea was showing, you can have multiple functionalities and multiple resources (F and R represent the "product poset" of them). Composition could require that you are able to "bridge" them (see example AC made).

In the picture below you see that the "chassis DP" has 3 resources, and that only 2 of these are involved in the composition with the "motor DP". (The complete theory to understand the diagram will be covered in the next lectures).



Q: if a Set can be seen as a poset and A as A=A^op^op we get a boolean profunctor from a relation right? Then Rel subcat of DP? Right? Something like that..;o)
GZ: Yes, I think AC just answered this

Q: Where/when does this setup give an algorithmic advantage compared to traditional approaches? Is its advantage in organising large networks of dependencies/letting the project be organised by a computer, and/or is there something else?
GZ: To be seen in next lectures.

Q: I follow perfectly the "mathematical" side, I'm short on the engineering side, can you give 2-3 examples more?... (I'm not an engineer…;o)) . Human resources - Jobs - skills..? THANKS!!!!!

GZ: See worked out examples on HR/Toilets

Q: How can a poset A be seen as a boolean profunctor Aop x A to Bool?

JL: Is the question about how to view a poset structure on a set A as a boolean profunctor?
OP: It seems we introduce boolean profunctors given that we know what posets are. But it also seems that a poset can be defined as a boolean profunctor itself.
JL: "But it also seems that a poset can be defined as a boolean profunctor itself." Yes, there is in principle a way do think/do this (but I'm confused as to why one would want to :)). Here's how I would think about this:

1. A poset structure on a set X is a special kind of relation R \subset X x X
2. A relation R \subset X x X can be viewed as a special kind of Boolean profunctor f_R: X x X → Bool (here, in the domain of f_R, we are thinking of X as a poset with trivial poset structure, i.e. no elements are related).
3. So, combining the first two points, we can encode a poset structure R on X as a boolean profunctor f_R.

OP: Ok, so we defined a composition of profunctors Aop x B to Bool and Bop x C to Bool. But what if we use the above to Aop x A to bool composed with Aop x B to bool?

GO: what can represent a natural transformation between profunctors?
GZ: We will see this when talking about locally latticeal categories. We will have a way of comparing design problems (boolean profunctors).