v1.0:2008.04.20

On the applicability of the phenotype framework implemented in ICES

Ricardo Aron T. F. Birmann

April, 2008

This article is part of the ICES project documentation and is available at http://ricardobirmann.com/ices/docs

1 Introduction

In ICES, just like in the real world, `LivingBeing`-s[1] have a bunch of genes that are translated into a phenotype, which will then determine the creature's metabolism and behavior.

In order to maintain the biological coherence of ICES' simulation, it was critical to formulate a framework that would allow complete distinction between genotype and phenotype. Genotype had to be inheritable, while phenotype couldn't. Metabolism and behavior had to dependent on phenotype, and never on genotype.

Additionally, as "bonus" goals, the implementation should allow skills to be acquired (or lost) throughout generations, simply due to genetic differentiation between `LivingBeing`-s, as well as to contemplate the possibility of new abilities arising in the population as the lineage evolves and to be free from hard-coded connotation, allowing phenotypic structures to acquire new functionality as generations come and go.

On top of it all, it should be a platform from which all sorts of emergent complexity could be dreamed about, from collective hunting to speciation.

This article does not necessarily justifies the implemented code. Instead, it proposed a discussion over the described framework, envisioning improvement and amelioration of all constructed models, concepts and algorithms from such discussion.

Since many references are made throughout the text to classes on the ICES code, a quick glance over the implemented structure is needed beforehand.

2 Contextual overview

In ICES, each `LivingBeing` has a **genotype** represented by a collection of objects of the class `Gene` and a **phenotype** represented by a collection of objects of the class `PhenotypicExpression`.

`LivingBeing`-s also have a metabolism and react to the surrounding environment; both metabolism and behavior are symbolized by objects of the `Skill` class, which represent in a very broad manner, the complete collection of skills, abilities, deficiencies, reactions and metabolic processes that can be acquired by a certain `LivingBeing`, as long as permitted by its phenotype.

In order to tie `Skill`-s to the `LivingBeing`'s phenotype, objects of an auxiliary class, `PhenotypicModel`, are used. In ICES notation, a `Skill` requires[2] a group of `PhenotypicModel`-s, and `PhenotypicModel`-s are represented by `PhenotypicExpression`-s.

[1] Some of the notation in this article is used in accordance to the computer code to allow better referencing. All text displayed in `monospace font` refers to ICES code.

[2] Underlined text is used herein to introduce specific notation or terminology used within ICES.

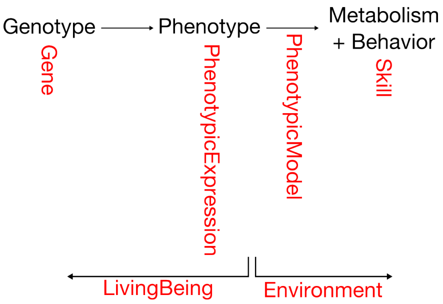


Figure 1: *Most important classes (in red) of the phenotype framework implemented in ICES.*

PhenotypicModel-s are abstract representations of what kind of body structure a LivingBeing would need to be able to perform a specific Skill. For instance, a group of PhenotypicModel-s could describe the structure of a functional pair of legs that would allow bipedism and LivingBeing-s with PhenotypicExpression-s that could represent these PhenotypicModel-s would be biped.

Furthermore, a Skill's set of required PhenotypicModel-s indicates the excel phenotypic structures needed to perform such ability in the most efficient manner possible.

A concept of **similarity** between phenotypic models and expressions is contemplated in the ICES algorithm, creating a sense of scale in how well is a certain PhenotypicExpression represents a PhenotypicModel. This way, LivingBeing-s capable of one same Skill will not necessarily perform it identically[3], but we'll get back to this.

3 The phenotype model, as implemented

3.1 A geometric optics into phenotype

In any model and, particularly, in any simulation, simplifications are made in order to:

a) Reduce the number of involved variables, allowing easier understanding of the problem;

b) Isolate the aspects that are relevant for the research intended, focusing the resulting data on information that has as little interference as possible from uninteresting details; and

c) Minimize the computational effort needed to run the simulation.

With that in mind, the **description** of a phenotype can be simplified to a finite set of Ω relevant characteristics c_1, c_2, $\ldots$, c_Ω and the simplified **description** of each of these c_i, as imposed by the fact the phenotype is being modeled, would take no more than a finite amount of information.

Each of these sets of information (that describe each of the phenotypic characteristics) should have a finite number Λ_i of degrees of freedom $y_{i,1}$, $y_{i,2}$, $\ldots$, y_{i,Λ_i}, as shown in the decomposition on Table 1.

c_1	c_2	c_3	$\cdots$	c_Ω
$y_{1,1}$	$y_{2,1}$	$y_{3,1}$	$\cdots$	$y_{\Omega,1}$
$y_{1,2}$	$y_{2,2}$	$y_{3,2}$	$\cdots$	$y_{\Omega,2}$
$\vdots$	$\vdots$	$\vdots$		$\vdots$
y_{1,Λ_1}	y_{2,Λ_2}	y_{3,Λ_3}	$\cdots$	$y_{\Omega,\Lambda_\Omega}$

Table 1: *Decomposition of phenotypic characteristics c_i into several degrees of freedom $y_{j,k}$. Note that Λ_1, Λ_2, $\ldots$, Λ_Ω are not necessarily the same.*

Instances of the PhenotypicExpression class represents the atomic aspects that constitute the LivingBeing's phenotype, and it is consequently natural to understand each PhenotypicExpression as one of the above mentioned **degrees of freedom** of the system.

Since degrees of freedom are, by definition, unidimensional, so should PhenotypicExpression-s, and it is tempting, to say the least, to represent these entities as points in a unidimensional space.

Being so, in ICES notation, each degree of freedom $y_{j,k}$ is understood as a point in the 1D space referred to as the Phenotypic Space and the value assumed by each coordinate is called Phenotypic Space Value or, simply, PSV.

[3] All humans are bipeds, but some run faster than others.

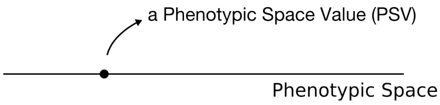


Figure 2: *The unidimensional Phenotypic Space.*

So remember: `Skill`-s require `PhenotypicModel`-s, which are ideal phenotypic configurations for `PhenotypicExpression`-s. These are abstract atomic entities used to describe a `LivingBeing`'s phenotype according to the simplifications made necessary by the model; they have to be somehow dependent on the creature's genotype, composed of objects of the class `Gene`.

3.2 Representativeness of phenotypic models by phenotypic expressions

Because our `LivingBeing`-s have `PhenotypicExpression`-s to be matched with a set of `PhenotypicModel`-s required by each `Skill`, gradual phenotypic changes along the generations eventually lead to acquisition (or loss) of a certain ability.

As a consequence, as the phenotype is altered throughout the lineage, a concurrent gradual increase (or decrease) in the efficiency of the creature's abilities is expected to be observed in the generations right after (or before) the acquisition (or loss) of the given skill.

ICES could not afford to have a **boolean** skills framework, where either a living being **had** or **hadn't** a skill and it was, therefore, needed that the implementation of `PhenotypicExpression`-s and `PhenotypicModel`-s allowed for two of these to be not only **equal** or **different**, but also **similar** to one another. This similarity is the basis for the concept of representation, mentioned before.

A `PhenotypicExpression` can be a representation of a certain `PhenotypicModel`, but that can be a good, a great, a bad or a lousy representation. To quantify "how good of a representation it is" a function $R(y)$ that takes the `PhenotypicExpression`'s PSV (y) as argument needed to be defined.

The determination of **which** function best fits this purpose is far from easy and the topic is completely open for discussion. However, ICES needed something to be coded and a few criteria were defined to choose a function; namely, $R(y)$ had to be symmetrical around y_m, maximum at y_m, balanced to only assume values between 0 and 1 and should asymptotically go to zero for PSV's far from y_m.

At the end of the day, a gaussian curve was chosen:

$$R(y) = G(y) = \exp\left[-\frac{(y-y_m)^2}{2\sigma^2}\right]$$

The parameter σ is user-defined and can be different for each case. The curve is trimmed at $y_m + 3\sigma$ and $y_m - 3\sigma$ to define a limited interval of representativeness by the given model[4] (red and dotted on Figure 3).

The height of the gaussian curve at the PSV of a `PhenotypicExpression` describes how well it represents the `PhenotypicModel` at hand[5]. This parameter can be used by `Skill`-s requiring this model to calibrate the response accordingly.

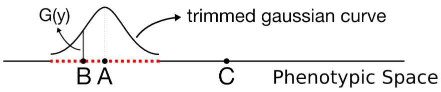


Figure 3: *Gaussian curves centered at the `PhenotypicModel`'s PSV (A) are used to determine whether it is represented by given `PhenotypicExpression`-s. A PSV inside the interval of radius 3σ (B) indicates the `PhenotypicExpression` is representative, while a PSV outside such interval (C) imply that it isn't.*

The simulation shown in Figure 4, was prepared to illustrate this concept of representativeness. In it, the only existing `Skill` was "to move horizontally to the right", but the speed with which this movement took place was determined by the representation of the `Skill`'s `PhenotypicModel` by the creature's

[4] $G(y_m \pm 3\sigma) \sim 0.01$

[5] $0 \leq G(y) \geq 1$; $\max G(y) = G(y_m) = 1$

PhenotypicExpression[6]. PSVs closer to the center of the gaussian would imply in more efficient runners.

Twenty five LivingBeing-s were created with random genotype and positioned as shown in Figure 4(a). The different phenotypes were analyzed by ICES to see which ones were adequate for performing the existing Skill and the simulation was put to run.

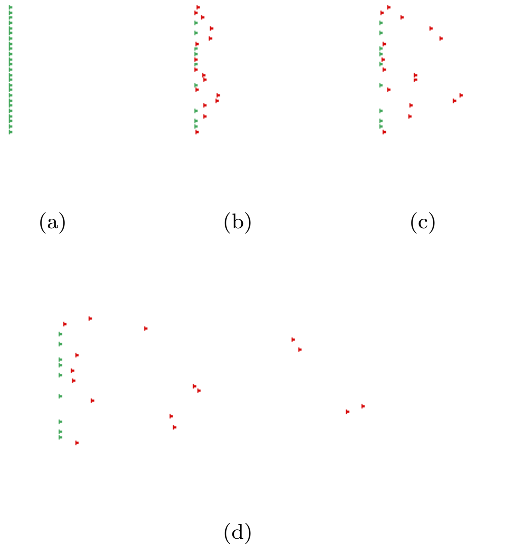


Figure 4: *Simulation of twenty five LivingBeing-s with random genotype in a world where there was only one Skill available (move to the right); shown at simulation cycles 0 (a), 15 (b), 50 (c) and 200 (d); Behavior differentiation caused by phenotype variation can lead to an evolutionary advantage of a certain lineage and is, therefore, a crucial feature of ICES.*

The Skill was coded to also change the color of the LivingBeing from green to red while it was moving so it was easy to see the ones that did not have an appropriate phenotype to perform such Skill.

3.3 Multiple representations

Interestingly, this kind of structure automatically allows for a single PhenotypicExpression to represent more than one model, as shown on Figure 5 below.

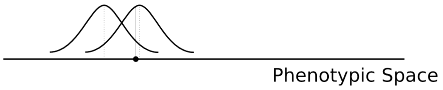


Figure 5: *A single PhenotypicExpression may represent multiple PhenotypicModel-s*

This kind of situation is fascinating given the emergent evolutionary strategies that are made possible. In the long run, this means ICES can simulate situations where the function of a certain phenotypic characteristic changes along generations, such as fish fins that start to work as legs, or penguin wings that were once used for flying and are now for swimming.

Allowing the same PhenotypicExpression to assume different roles gives the simulated creature an evolutionary "choice", in the sense that, depending on what the environmental situation is, one or other path may be taken by the upcoming generations.

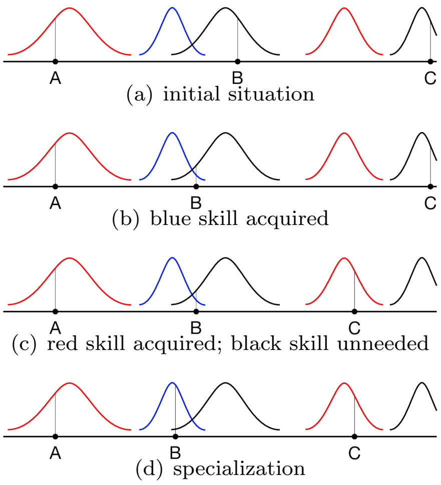
In the example shown in Figure 6, the PhenotypicModel-s required by three different Skill-s are identified by a code of colors. Lets imagine a certain simulated creature needs either the black Skill or the blue and red Skill-s **together** to survive. Additionally, lets assume LivingBeing-s with both red and blue Skill-s have an evolutionary advantage over those with only the black Skill.

Because of the overlapping between required PhenotypicModel-s for the blue and black Skill-s, a mutation might allow the blue Skill to appear in the lineage without the black one being lost (note change in position of PhenotypicExpression B in Figure 6(b)). Later on, a second mutation can move the PhenotypicExpression C and thus granting

[6] The Skill was configured to only require one PhenotypicModel and each LivingBeing had fifteen PhenotypicExpression-s; it would suffice for the Skill to be acquired to have one of these PhenotypicExpression-s representing the given PhenotypicModel.

the creature the red `Skill` (Figure 6(c)).

Finally, now no longer needing the black `Skill`, the lineage can evolve to specialize in its novel abilities, as shown by the migration of the `PhenotypicExpression` B towards the optimal PSV (Figure 6(d)).



(a) initial situation

(b) blue skill acquired

(c) red skill acquired; black skill unneeded

(d) specialization

Figure 6: *Sequence of phenotypical configurations where the overlapping between `PhenotypicModel`-s allows sophisticated evolutionary strategies to emerge.*

3.4 The implications of having a single phenotypic space

It might be needless to say at point, but all `PhenotypicModel`-s and `PhenotypicExpression`-s are represented on a single phenotypic space. In order to make possible all versatility and flexibility that is expected of ICES, it was understood that there could be no segregation per `Skill`, per `LivingBeing` or anything in this sense, mainly because, conceptually:

1. Having a segregation would indicate the use of a certain criterion to distinguish two `PhenotypicModel`-s from one another, while these are expected to be completely connotation-free, equivalent entities.

…and, pragmatically, with a single phenotypic space:

2. It is simpler to allow multiple `Skill`-s to require the same `PhenotypicModel`.

3. Lineages are completely free to, through evolution, have phenotypes that represent any combination of `PhenotypicModel`-s.

4. Superposition of the representativeness ranges of two or more `PhenotypicModel`-s are made possible (and evident).

It is mandatory, however, to analyze the effects of putting all `PhenotypicExpression`-s and, more importantly, all `PhenotypicModel`-s on the same space.

On a unidimensional space, any given set of points is a **totally ordered set**. That means that, for any given pair of elements, there exists an implicit order between them (e.g. if $a \neq b$, then either $a > b$ or $a < b$). More than that, given three points a, b and c, where $a < b$ and $b < c$, it is impossible to go from a to c without passing by b.

In the phenotypic space, one can imagine a sense of **movement** of `PhenotypicExpression`-s, as genotype changes during reproduction and `PhenotypicExpression`-s of newborn `LivingBeing`-s are not exactly at the same PSVs as in the case of the parent generation[7].

A premature interpretation would lead to the conclusion that `PhenotypicExpression` A (as shown in Figure 7) would unavoidably represent one of the black `PhenotypicModel`-s before being able to represent anything else (such as the blue `PhenotypicModel`). However, genetic alterations are far from being continuous functions and do cause jumps in the `PhenotypicExpression`'s PSV from one generation to the other.

[7] Although interesting and handy in some cases, the understanding of this phenomenon as **movement** is misleading, since `PhenotypicExpression`-s of the parent generation do not correlate with the `PhenotypicExpression`-s of its offspring. Even if child `LivingBeing`-s are identical to their parents (clones), the identical phenotype would be a reflex of the inherited identical genotype and never of some sort of phenotypical inheritance.

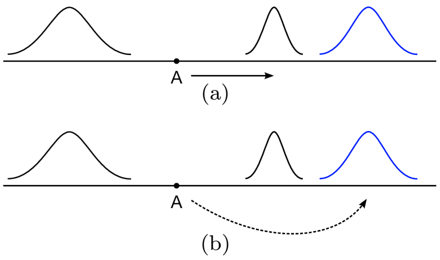


Figure 7: *One of the limitations of a unidimensional phenotypic space is the implied ordered between `PhenotypicModel`-s, however, instead of changing continuously throughout the generations (a), `PhenotypicExpression`-s jump from one PSV to another (b).*

More than that, unless the simulation at hand contemplates very simple `Skill`-s with only one required `PhenotypicModel` per `Skill`, the risk of a `PhenotypicExpression` being ”trapped” between two `PhenotypicModel`-s wouldn't imply a risk of a certain phenotypic **characteristic** being ”trapped” between other two, since the creature's phenotype (as a whole) would actually exist in a n-dimensional world[8] and sets of points in spaces with cardinality greater than one are not ordered.

Still, there are a few problems with using a single phenotypic space, mostly related to a probabilistic analysis of acquisition of `Skill`-s.

Just to mention one example: given a `PhenotypicModel` M represented by a `PhenotypicExpression` E at PSV y, if genetic alterations incurred during reproduction have an equal probability of shifting y by $+\Delta y$ or by $-\Delta y$ and if all `PhenotypicModel`-s have the same σ; it is impossible to have more than two `PhenotypicModel`-s with the same probability of being acquired by a shift in E's position.

This is clearly a very specific situation, dependent on many variables, such as the distribution of the probability of phenotypic shifts to occur, as a function of Δy, that could be asymmetrical or periodical or discontinuous; or the variance of σ values along the universe of existing `PhenotypicModel`-s; and so on.

These issues have been looked at as minor difficulties incurred by the simplifications[9] made during the creation of the model and considered to be acceptable, given the features made possible by the use of this implementation.

4 Proposed discussion

The disadvantage of modeling reality into a crazy and complex (even if flexible and powerful) framework is that sometimes it can get really hard to trace a straight line between the represented subject in the real world and its representation inside the model.

Specifically in the case of ICES, there have been a few times when the implementation of a certain feature had direct impact in the model used, such as when a concept of similarity was needed between `PhenotypicModel`-s and `PhenotypicExpression`-s and a complex mechanism (gaussian representation curves) was coded.

It is extremely important to always reexamine the implications of these structures and algorithms, even if this leads to code deletion and re-implementation.

This article was written with the intention of stimulating a discussion over the concepts and models adopted in the phenotype framework in ICES. The implemented strategy is absolutely not definitive and will hopefully be revised after new ideas emerge from dialogue.

[8] Where n = number of `PhenotypicModel`-s represented by the `LivingBeing`'s `PhenotypicExpression`-s.

[9] Simplifications are necessary on any process of modeling, as discussed earlier.