Do software architectures evolve too fast ? A biologist's point of view

Philippe Grosjean <Philippe.Grosjean@umons.ac.be>

University of Mons, Complexys Institute
Numerical Ecology of Aquatic Systems Laboratory (EcoNum)

IWSECO-WEA 2014

complexys
INSTITUT DE RECHERCHE
SUR LES SYTEMES COMPLEXES
DE L'UMONS

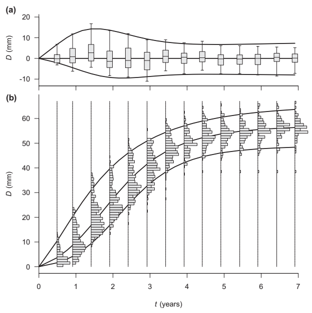
UMONS

Overview

A short presentation of the speaker

Marine biologist, biostatistician and software developer

- **Bioengineer** with a Ph.D. thesis in **marine biology** (growth model of sea urchins)
- Additional skills developed in **(bio)statistics** during post-docs and consultancy during 4 years all around Europe (France, Ireland, Spain, U.K.)
- **EcoNum lab** created in 2004 at UMONS
- Interested by **interdisciplinary work**: biology, chemistry, modelling, statistics, computing science
- Write **software for ecology** in Java, R, ...



Growth model of sea urchins

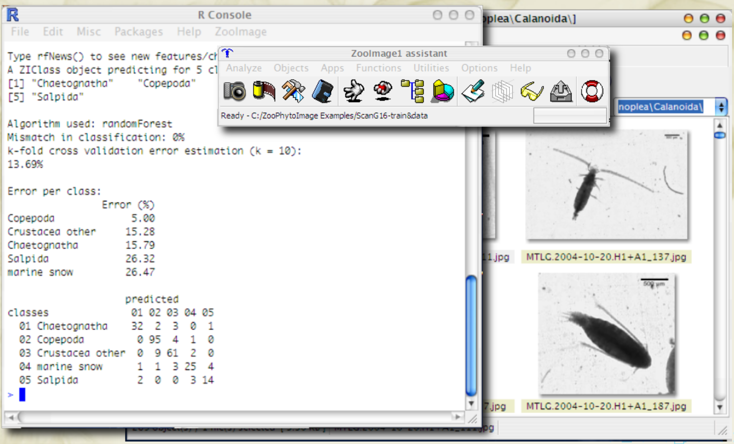
Ecophysiology of corals in mesocosms

Tropical coral reefs constitute beautiful and diverse ecosystems, but they are endangered by climate changes, overfishing and pollution.
At EcoNum lab, we study how the environment affects growth, reproduction and health (ecophysiology) of tropical corals in artificial reef mesocosms.

Automatic identification of plankton

Plankton (the organisms that drift in the middle of the column water) is a diverse community. There are easily thousands of species in 1L of seawater.

At EcoNum lab, we develop tools to automatically enumerate plankton using image analysis and supervised classification algorithms.


R Console
File Edit Misc Packages Help ZooImage
ZooImage1 assistant
Analyze Objects Apps Functions Utilities Options Help
oplea\Calanoida\]
noplea\Calanoida\
1.jpg
MTLG.2004-10-20.H1+A1_137.jpg
7.jpg
MTLG.2004-10-20.H1+A1_187.jpg
500 µm

```
Type rfNews() to see new features/ch
A ZIClass object predicting for 5 cl
[1] "Chaetognatha"      "Copepoda"
[5] "Salpida"

Algorithm used: randomForest
Mismatch in classification: 0%
k-fold cross validation error estimation (k = 10):
13.69%

Error per class:
                 Error (%)
Copepoda              5.00
Crustacea other      15.28
Chaetognatha         15.79
Salpida              26.32
marine snow          26.47

                    predicted
classes              01 02 03 04 05
  01 Chaetognatha    32  2  3  0  1
  02 Copepoda         0 95  4  1  0
  03 Crustacea other  0  9 61  2  0
  04 marine snow      1  1  3 25  4
  05 Salpida          2  0  0  3 14
>
```

Ready - C:/ZooPhytoImage Examples/ScanG16-train&data

Software development

Most of the time (>99%) I am developing **solutions in R**

- Author and **maintainer** of 11 R packages
- Main **translator** of R in French
- **SciViews** (GUI), **tinn-r** (Code editor)
- **mlearning** and **zooimage** (machine learning)
- **aurelhy** (multivariate spatial interpolation), ...

Tinn-R Editor – GUI for R Language and Environment

José Cláudio Faria
Philippe Grosjean
Enio Galinkin Jelihovschi
Ricardo Pietrobon

Editora da UESC

UMONS

The ECOS project

Ecological Studies of Open Source Software Ecosystems (http://informatique.umons.ac.be/genlog/projects/ecos/) 2011-2016.

In ECOS we look at Open Source Software ecosystems using tools and ideas gathered from biological ecosystems

We currently work with GNOME and CRAN (Comprehensive R Archive Network, http://cran.r-project.org).

Two research questions:

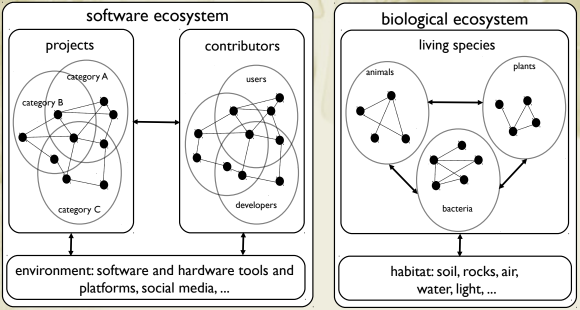
1. Which theories for biological evolution can be used to understand and explain software project evolution?
2. Which control mechanisms driving biological ecosystems can be used to explain and change the way in which software ecosystems evolve?

Comparison of biological and software ecosystems

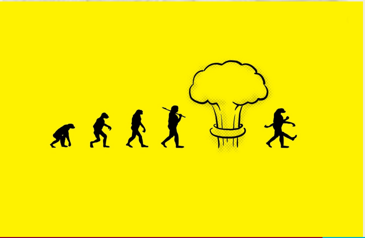
Comprehension of software ecosystems could also benefit from the study of biological systems considering:

- Many complex systems share some **common emergent properties**
- **Analogies** exist in software and biological ecosystems components

Two points of view on software ecosystems compared to biological ecosystems:



Biological architectures and evolution



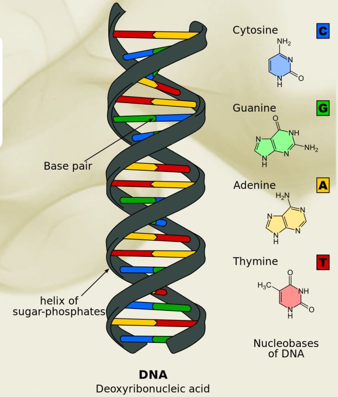
DNA

As you know, the genetic code is contained in special molecules named **“deoxyribonucleic acid” (DNA)**

DNA forms a double helix of a succession of **nucleotides** linked by hydrogen bounds between base pairs. There are four possible bases:

- **Adenine (A)** and **Thymine (T)** that make two hydrogen bounds, forming T-A (or A-T) structures
- **Guanine (G)** and **Cytosine (C)** that make three hydrogen bounds forming a G-C (or C-G) structures

One DNA helix is thus a succession of bases (say AATGTC...) that binds and coils with its complement, that is, TTACAG...

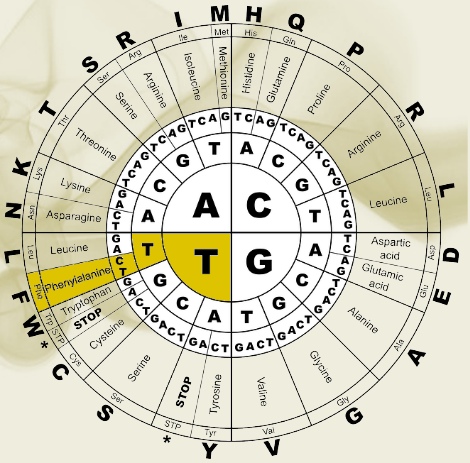


DNA and codons

- 2-bit based system of information. Encoding sequences of **20 different amino-acids to build proteins**
- Two nucleotides not enough for 20 amino acids (4^2=16)
- Three are OK (4^3=64 possibilities) => **codons**
- Several codons for the same amino acid.

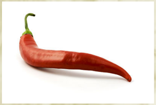
Phenylalanine

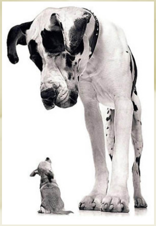
TTT and TTC code both for the same amino acid: phenylalanine.



Universality of the genetic code

The genetic code is **universal**. Same genetic code for everybody!





This contrasts with computing architectures where different standards that co-exist: big endian versus little endian, character encodings, etc.

The genetic code in extreme conditions

May be is it just because there is no need to change it?

Let’s look at **hyperthermophilic conditions**, where “normal” DNA starts to degrade at very high temperatures...

Hyperthermophiles

- Hyperthermophile bacteria resist to very high temperatures

- G-C pairs with three hydrogen bounds are more stable than A-T pairs.
- Switch to G-C bound everywhere it is possible in hyperthermophile's DNA

Phenylalanine

Recall that both TTT and TTC code for phenylalanine. TTC is more stable. It is preferentially found in hyperthermophile's DNA

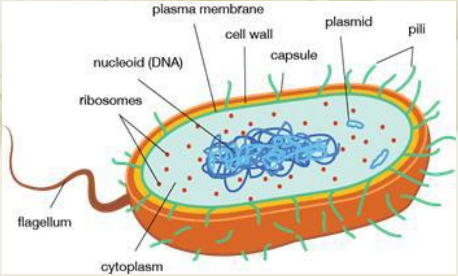
An even better approach would be to redefine the genetic code in favour of more G-C bounds everywhere.

It is not the case!

Hyperthermophiles strictly conform to standard genetic code, despite the selective pressure towards a break of the "standards"

And when a new standard is really required?

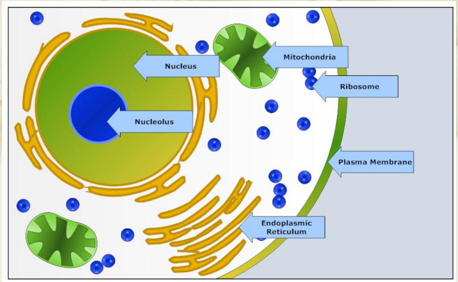
- Simplest and most primitive organisms are **"prokaryote"**.
- Their genetic material (DNA) is not in a separated compartment.
- This makes the expression of the genetic code to build proteins relatively easy.
- However, there is little control of interactions between the DNA and the proteins in the cell.



As cells became more and more complex, that started to be problematic!

Version 2: the eukaryote

- Another type of cells appeared: the **eukaryotes**.
- Those cells have a **nucleus** that is a specific compartment containing the DNA (plus other compartments called **organelles**).
- DNA is now protected and a specific mechanism allows to filter molecules going in and out of it.



Organelles of the Cell

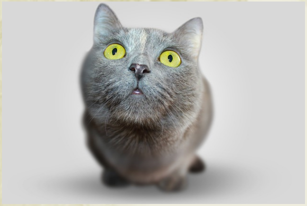
Prokaryotes versus eukaryotes

In computer science terminology, eukaryotes look like refactored prokaryotes with an object-oriented organisation of the molecules inside the cell

Note two important points:

- *version 1 (prokaryote) is still used today* whenever it is sufficient: there are lots of bacteria still living around us!
- *there is only one version 2 (eukaryote)* with the same nucleus and same mechanism for "message passing" (**nucleopores**) for all eukaryotes

Comparison with software architectures



UMONS

Too rapid changes, too many standards?

- Software architectures evolution looks more “chaotic” to me: *it is a moving target build on shifting sands*
- There are **too many standards**, and best ones are not necessarily enforced/adopted by users
- Often, **change is forced by obsolescence** (sorry! no support any more... you *have* to upgrade!)

Software architectures on shifting sands

Looking at software architectures, it seems that changes were required more frequently. **Many standards co-exists.** For instance:

- Little endian *versus* big endian,
- 8bit then 16bit then now 32bit *versus* 64bit,
- Unixes and Linuxes *versus* Windows *versus* Mac OS *versus* Android, ...
- etc, etc.

This situation contrasts with the foundations of life, which appears much more conservatives with time

New standards enforcement

There are certainly many reasons to enforce new standards:

- Avoidance of old code handling
- Obsolescence of old hardware technologies
- Incompatibilities with old standards

But there is a big one that is not beneficial to the end user:

- **Commercial strategies and concurrence.** A few companies are expert in inventing new standards just to outcompete concurrents!

For life, the main driving force is the ability to produce organisms with superior capabilities (Darwin's struggle for life)

UMONS

One example: character encoding

Character encoding is a basic standard. There are too many different character encodings... for historical reasons, they say:

- ASCII, one of the ancestor, widespread but obviously too limited
- code pages with 8bit encoding like ISO 8859, Mac Roman, etc.
- UTF-8, UTF-16, UTF-32

UTF-8 could potentially become *the* standard character encoding (the Web, Linuxes, Mac OS X, ...)

But it is still not in that position of being the default system used everywhere.

Rich-text formats

Here again, the history is rich in many different standards: RTF, HTML, etc.
Currently, there is:

- OpenDocument *versus* Open XML
- XML DocBook
- LaTeX
- Plain text formatting like Markdown

What will be used in, say 5 or 10 years from now?

In which format should I record my texts to keep them readable/editable as much as possible in the future?

What to use if I want to do reproducible analysis with computations, plots and tables generated in the document directly?

iPython notebook? R with Sweave/knitr? What else?

Suboptimal standards

Suboptimal standards enforced by either commercial strategies, or simply inertia are also problematic.

A classical example: the **QWERTY/AZERTY keyboards**.

They were developed for old typewriter machines **to avoid internal clashing of typebars**

This constraint does not exists any more and there are more efficient configurations: Dvorak, Colemak, Bépo, ...

How many people use them?

Discussion

Discussion

A comparison of standard for the foundation of biological *versus* software architectures highlights large differences:

1. Biological systems use one unique genetic code; there are too many standards for encoding in software systems
2. New standards appears rather infrequently in biological systems (eukaryotes *versus* prokaryotes)
3. The driving force in biological systems is **performance**. In software architectures, commercial interests and historical reasons can impose suboptimal standards
4. Keeping this in mind, would it be possible to enforce a few standards that are optimised and designed to last longer in the software industry? Complex systems like life on earth have demonstrated it is possible; what should be done to move in the same direction for software architectures? *(open discussion...)*

UMONS

Discussion

A comparison of standard for the foundation of biological *versus* software architectures highlights large differences:

1. Biological systems use one unique genetic code; there are too many standards for encoding in software systems
2. New standards appears rather infrequently in biological systems (eukaryotes *versus* prokaryotes)
3. The driving force in biological systems is **performance**. In software architectures, commercial interests and historical reasons can impose suboptimal standards
4. Keeping this in mind, would it be possible to enforce a few standards that are optimised and designed to last longer in the software industry? Complex systems like life on earth have demonstrated it is possible; what should be done to move in the same direction for software architectures? *(open discussion...)*

Discussion

A comparison of standard for the foundation of biological *versus* software architectures highlights large differences:

1. Biological systems use one unique genetic code; there are too many standards for encoding in software systems
2. New standards appears rather infrequently in biological systems (eukaryotes *versus* prokaryotes)
3. The driving force in biological systems is **performance**. In software architectures, commercial interests and historical reasons can impose suboptimal standards
4. Keeping this in mind, would it be possible to enforce a few standards that are optimised and designed to last longer in the software industry? Complex systems like life on earth have demonstrated it is possible; what should be done to move in the same direction for software architectures? *(open discussion...)*

Discussion

A comparison of standard for the foundation of biological *versus* software architectures highlights large differences:

1. Biological systems use one unique genetic code; there are too many standards for encoding in software systems
2. New standards appears rather infrequently in biological systems (eukaryotes *versus* prokaryotes)
3. The driving force in biological systems is **performance**. In software architectures, commercial interests and historical reasons can impose suboptimal standards
4. Keeping this in mind, would it be possible to enforce a few standards that are optimised and designed to last longer in the software industry? Complex systems like life on earth have demonstrated it is possible; what should be done to move in the same direction for software architectures? *(open discussion...)*

Discussion

A comparison of standard for the foundation of biological *versus* software architectures highlights large differences:

1. Biological systems use one unique genetic code; there are too many standards for encoding in software systems
2. New standards appears rather infrequently in biological systems (eukaryotes *versus* prokaryotes)
3. The driving force in biological systems is **performance**. In software architectures, commercial interests and historical reasons can impose suboptimal standards
4. Keeping this in mind, would it be possible to enforce a few standards that are optimised and designed to last longer in the software industry? Complex systems like life on earth have demonstrated it is possible; what should be done to move in the same direction for software architectures? *(open discussion...)*

So, are software architectures evolving too fast...
at the cost of multiple standards that do not last long enough?
Could biology inspire you here?