The Evolution of Identity and Modularity in Nature and Computation

Lee Spector
Cognitive Science
Hampshire College
Overview

- Modularity
- Identity
- Evolving computer programs
- Evolving modular programs
- Implications
Modularity is Everywhere
http://wyss.harvard.edu/viewevent/37/wyss-seminar-series-kasper-stoy
http://www.hizook.com/blog/2012/01/16/ted-talks-about-robots-and-robotics-part-1
http://www.synthtopia.com/content/2007/04/04/moog-55-modular-synthesizer/
http://mappery.com/map-of/Tokyo-Metro-Map
Modularity in Software

- Pervasive and widely acknowledged to be essential
- Modules may be functions, procedures, methods, classes, data structures, interfaces, etc.
- Modularity measures include coupling, cohesion, encapsulation, composability, etc.
Cognitive Science

• Long history of modularity theories: Gall, ... Simon, ... Fodor, ... Cermak and Craik, ... Gardner, ... Jackendoff, ... Grafman, ...

• Simon’s “nearly decomposable systems”

• Fodor’s features: domain specific, mandatory, fast, encapsulated, fixed architecture, characteristic patterns of ontogeny and failure

• Central vs. input systems

• Modest vs. massive
module | mˈæjʊl |
noun
each of a set of standardized parts or independent units that can be used to construct a more complex structure, such as an item of furniture or a building.
- [ usu. with adj. ] an independent self-contained unit of a spacecraft.
- Computing any of a number of distinct but interrelated units from which a program may be built up or into which a complex activity may be analyzed.

ORIGIN late 16th cent. (in the senses ‘allotted scale’ and ‘plan, model’): from French, or from Latin modulus (see modulus). Current senses date from the 1950s.
Questions

• **Why** are modules everywhere?

• What are they good for?

• Where do they come from?

• What conditions permit or facilitate their emergence?
Identity

• How are modules recognized by other components of a system?
• Where do module identities come from?
• How can module identity co-evolve with modular architecture?
Holland’s Tags

- Initially arbitrary identifiers that come to have meaning over time
- Appear to be present in some form in many different kinds of complex adaptive systems
- Examples range from immune systems to armies on a battlefield
- A general tool for the support of emergent complexity
Evolution of Altruism

• Puzzles/challenges/results since Darwin

• Explanations of altruism toward:
  • Kin
  • Reciprocating partners
  • Agents with good reputations
Tag-Based Altruism

- Individuals have tags and tag-difference tolerances
- Donate when $\Delta \text{tags} \leq$ tolerance
- Riolo et al. (Nature, 2001) showed that tag-based altruism can evolve; Roberts & Sherratt (Nature, 2002) claimed it would not evolve under more realistic conditions
Spector, L., and Klein, J. Genetic stability and territorial structure facilitate the evolution of tag-mediated altruism. In *Artificial Life*. 

Tuesday, April 16, 13
Evolutionary Computation

Random Generation

Assessment

Selection

Solution

Variation
Genetic Programming

- Evolutionary computing to produce executable computer programs.
- Programs are tested by executing them.
Evolving Modular Programs

With “automatically defined functions”

- All programs in the population have the same, pre-specified architecture
- Genetic operators respect that architecture
- Complicated, brittle, limited...
- Architecture-altering operations: more so
Evolving Modular Programs

With “execution stack manipulation”

- Code queued for execution is stored on an “execution stack”
- Allow programs to duplicate and manipulate code that on the stack
- Simple types and uses of modules can be evolved easily
- Does not scale well to large/complex systems
Evolving Modular Programs

With tags

- Include instructions that tag code (modules)
- Include instructions that recall and execute modules by closest matching tag
- If a single module has been tagged then all tag references will recall modules
- The number of tagged modules can grow incrementally over evolutionary time
Dirt-Sensing, Obstacle-Avoiding Robot Problem
## DSOAR Instructions

<table>
<thead>
<tr>
<th>Condition</th>
<th>Instructions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basic</td>
<td>if-dirty, if-obstacle, left, mop, v8a, frog, $R_{v8}$</td>
</tr>
<tr>
<td>Tag</td>
<td>if-dirty, if-obstacle, left, mop, v8a, frog, $R_{v8}$, tag.exec.[1000], tagged.[1000]</td>
</tr>
<tr>
<td>Exec</td>
<td>if-dirty, if-obstacle, left, mop, v8a, frog, $R_{v8}$, exec.dup, exec.pop, exec.rot, exec.swap, exec.k, exec.s, exec.y</td>
</tr>
</tbody>
</table>
DSOAR Effort

Problem Size

Computational Effort

Tag
Exec

Problem Size

Tuesday, April 16, 13
Evolved DSOAR Architecture (in one environment)
Evolved DSOAR Architecture (in another environment)
Conclusions

• Tags provide an effective mechanism for the evolution of modular programs that solve difficult problems

• Tags may provide or explain mechanisms that support the evolution of modularity in a range of other systems, both natural and artificial