A Formal Approach to Computational Creativity

Claudia Elena Chiriţă and José Luiz Fiadeiro

Department of Computer Science, Royal Holloway University of London, UK
claudia.elena.chirita@gmail.com, jose.fiadeiro@rhul.ac.uk

Computational creativity is a subdomain of AI developed in the last decades to explore the potential of computational systems to create original artefacts and ideas. Overlapping with the broader field of cognitive science, it encompasses "the philosophy, science and engineering of computational systems which [...] exhibit behaviours that unbiased observers would deem to be creative" [3].

In this work, we discuss computational creativity from an algebraic point of view, showing how we can give a mathematical formalization of creative systems and their components. We start from the tenet that creativity can be seen in essence as the identification or location of new conceptual objects in a conceptual space [1], and present creative systems in an institutional setting built around the notion of concept. We adopt Goguen's understanding of a conceptual space [8,7] as an algebraic specification [11], and develop our study based on institution theory [9], which formalizes logical systems by capturing their syntax, semantics, and the satisfaction relation between them in an algebraic manner. This allows us to maintain the generality of previous descriptions of creative systems [12], and at the same time to use formal definitions for concept abstraction, concretization, and blending [7] that enable reasoning about creative processes.

We first define creative systems by means of many-valued specifications written over institutions endowed with residuated lattices as truth spaces [2], and of abstract strategies for the discovery and evaluation of concepts based on the notion of graded semantic consequence between specifications [4].

We then focus on a subclass of creative systems modelled as complex dynamic systems. The idea that creative processes can be explained in terms of the evolution of complex systems has been extensively studied in works connecting creativity with biology-inspired algorithms (swarm optimization, genetic algorithms, etc.), nonlinear dynamical systems of equations and multi-agent systems. We continue this line of contributions by investigating a new connection between service-oriented architectures viewed as complex systems [5] and improvisation performances.

In service-oriented computing [6], software applications evolve dynamically to meet their goals through discrete reconfigurations triggered by the need for an external resource and realized through a three-fold process of discovery, selection and binding of service modules. A salient feature of these systems is the unpredictability of their actual architectural configuration (i.e. the entities and connectors through which entities exchange information) at design time. This is due to their openness to reconfiguration: binding new services could trigger subsequent processes of discovery, selection, and binding. The dynamic aspect of these reconfigurations is endogenous, intrinsic to the systems; their

evolution is not driven by external factors such as the change of the environment, but originates from the design of the components themselves. We argue that this gives rise to an emergent behaviour similar to what has been observed for computational creative systems that model, for example, improvisations.

In modelling creative processes as service-oriented computing, we regard concepts as modules and concept discovery as service discovery. In this context, we evaluate the usefulness of a concept through the mechanism of service selection, and recast concept blending in terms of service binding. This permits us to study properties of improvisation processes within the framework of service-oriented systems: by analysing service repositories from the perspective of quality-of-service profiling we can examine the system's reliability (to what extend the user's expectations of delivering an artefact can be met?), determine the reachability of some concepts (which modules are never or seldom used during the execution of an/any application?), identify relationships between concepts, such as redundancy, within the universe of concepts involved in the process (does the presence of certain service modules prohibit the use of other modules?), and assess the cohesion of a performance or creative act (via the soundness of resolution for many-valued logic programming).

While most of the current research in computational creativity seems to adopt a connectionist view on cognitive and in particular on creative processes, our approach adheres to the computational theory of mind. This opens naturally a series of questions related to the everlasting paradigm dispute between the subsymbolic and symbolic views on the philosophy of mind that one should not ignore. To answer the concern on the origin of concept specifications, we investigate the connection between the abstract representations of concepts in neural networks and their algebraic definition [10]. Establishing a formal relation between the two would provide a solution: the automatic learning from examples could alleviate the user's task of writing complex specifications of concepts.

References

1. Margaret A Boden. *The creative mind: Myths and mechanisms*. Psychology Press, 2004.
2. Claudia Elena Chiriţă, José Luiz Fiadeiro, and Fernando Orejas. Many-valued institutions for constraint specification. In *Fundamental Approaches to Software Engineering*, volume 9633 of *LNCS*, pages 359–376. Springer, 2016.
3. Simon Colton and Geraint A. Wiggins. Computational creativity: The final frontier? In Luc De Raedt, Christian Bessière, Didier Dubois, Patrick Doherty, Paolo Frasconi, Fredrik Heintz, and Peter J. F. Lucas, editors, *ECAI 2012 - 20th European Conference on Artificial Intelligence. Including Prestigious Applications of Artificial Intelligence (PAIS-2012) System Demonstrations Track, Montpellier, France, August 27-31 , 2012*, volume 242 of *Frontiers in Artificial Intelligence and Applications*, pages 21–26. IOS Press, 2012.
4. Răzvan Diaconescu. Graded consequence: an institution theoretic study. *Soft Comput.*, 18(7):1247–1267, 2014.
5. José Luiz Fiadeiro. The many faces of complexity in software design. In *Conquering Complexity*, pages 3–47. Springer, 2012.

6. José Luiz Fiadeiro and Antónia Lopes. A model for dynamic reconfiguration in service-oriented architectures. *Software and System Modeling*, 12(2):349–367, 2013.
7. Joseph Goguen. An introduction to algebraic semiotics, with application to user interface design. In *Computation for metaphors, analogy, and agents*, pages 242–291. Springer, 1999.
8. Joseph A. Goguen. What is a concept? In Frithjof Dau, Marie-Laure Mugnier, and Gerd Stumme, editors, *Conceptual Structures: Common Semantics for Sharing Knowledge, 13th International Conference on Conceptual Structures, ICCS 2005, Kassel, Germany, July 17-22, 2005, Proceedings*, volume 3596 of *Lecture Notes in Computer Science*, pages 52–77. Springer, 2005.
9. Joseph A. Goguen and Rod M. Burstall. Institutions: Abstract model theory for specification and programming. *Journal of the ACM*, 39(1):95–146, 1992.
10. Michael John Healy and Thomas Preston Caudell. Ontologies and worlds in category theory: implications for neural systems. *Axiomathes*, 16(1):165–214, 2006.
11. Donald Sannella and Andrzej Tarlecki. *Foundations of Algebraic Specification and Formal Software Development*. Springer, 2012.
12. Geraint A. Wiggins. A preliminary framework for description, analysis and comparison of creative systems. *Knowl.-Based Syst.*, 19(7):449–458, 2006.