VRule - A Pixel Rewriting System for End-User Modeling

Rick L. Vinyard, Jr.

Dept. of Computer Science, New Mexico State University, Las Cruces NM, USA
rick.vinyard@gmail.com

Abstract. Since the introduction of pixel rewriting systems with Furnas' Bitpict system numerous approaches have been explored to extend the concept of pixel rewriting computation. VRule is a new pixel rewriting framework designed to support investigation of new formalisms for end-user modeling with pixel rewriting systems (PRSs). VRule is implemented as a Pattern Directed Inference System (PDIS) with specialized classes of antecedent and consequent Pattern Directed Modules (PDM) components. This overview briefly examines VRule's key elements resulting from exploration of the architectural aspects of rule-based engines to include matching, rule resolution, and execution in a spatial environment, as well as numerous new PDM formalisms. Examples representing classes of modeling domains have been developed to include Turing machines, 1D/2D/3D cellular automata, a game of Fox and Hounds, Langton's Ant (turmites), a climatological gas model originally written in C#, and a virological infection model originally written in Python.

Keywords: pixel · rewriting · modeling · end-user · inference engine.

1 Topic and Approach Taken

As noted by Blackwell[2] graphical rewrite systems are "an influential family of development environments for end-users." In addition to the groundbreaking contributions by Furnas, et. al., others [10, 7] further extended PRSs with logical conjunction and disjunction concepts, and showed Turing completeness.

Furnas and Qu [3] note the similarity between the raster based pixel rewriting systems and raster based cellular automata (CAs), but also distinguish the two approaches as "classical CAs are, however, often considered hard to program to achieve desired (as opposed to explore emergent) results."

VRule embraces both pixel rewriting systems and CAs and introduces new formalisms for PRSs that enable the programming of CAs and agent based pixel models, and improve upon the existing notational forms to demonstrate the potential of image based computation for some forms of end-user modeling.

2 VRule Architecture and Work Completed

VRule has been designed to support investigation into pixel rewriting environments and has been informed by many of the aforementioned graphical systems.

VRule has been implemented as a Pattern Directed Inference System (PDIS) where a "program is better viewed as a loosely organized collection of pattern-directed modules"[6] and employs a familiar *match* → *resolve* → *act* cycle. Like Bitpict, VRule's workspaces and rules are images, as is the raw or annotated output as seen in the latter images generated directly by VRule.

Examples of VRule's engine innovations include a new spatial resolver, and providing Visulan's extended pairs with a total ordering not only between rules, but also between spatial matches and introduction of intra-rule matches with independent spatial priority. Key/Value workspace layers allow rules to match and act against specific layers, and also boundary matches and hierarchical rules for specification of intra-rule strategies.

While Anderson's Inter-Diagrammatic Reasoning [1] evolved a macro level approach to BitPict, VRule's PDM approach takes a diametrical evaluation of the rules to the micro level of the pixels involved during the match and act phases while preserving BitPict's image to image computation. New PDM formalisms support paradigms common to classes of modeling domains including transparent pixel PDMs providing punctured lattices such as CA Moore neighborhoods. Anchor PDMs allow the programmer to specify transforms between the antecedent match and the consequent application. Other PDMs include null consequents, negation antecedent, border matching, and extended affine transforms. Non-image PDMs have been implemented such as event based antecedents, halt consequents, and snapshot consequents. Event based antecedents provide user interaction with the executing model, while snapshots allow programming specific points in the model evolution that can be used for further numerical analysis beyond the scope of VRule. New stochastic elements have been introduced as well as Trilobite notation inspired by the Schrodinger notation of Spider diagrams[5] to specify concepts such as "no more than n Moore neighbors," or "at least n Von Neumann neighbors with range less than r" used in the SEIR model. Also examined is the natural alignment between the parallelized architecture of *General-Purpose computing on Graphics Processing Units* (GPGPU) environments such as CUDA and OpenCL, and the execution engines of PRSs.

3 Expected Contribution, Future Work, and Advice

Figure 1 depicts three examples of VRule programs in different modeling domains created with VRule's new formalisms, and selected for computational characteristics in the domain. The three upper left images depict the Model of Ebullition and Gas storAge (MEGA) [8] (originally in C#), the three upper right images depict the progression of the Wolfram code 82 CA [9], and the nine lower images depict a Human Immunodeficiency Virus (HIV) SEIR cellular model [4] (originally in Python) progressing from a healthy state, to an infected state, to a delayed immune response over seven weeks. These examples demonstrate rule sets that include common concepts such as random populations or events enabled by VRule's stochastic rules, or such as cyclic events which are implemented using a cycle timer in a layer separate from the execution layers.

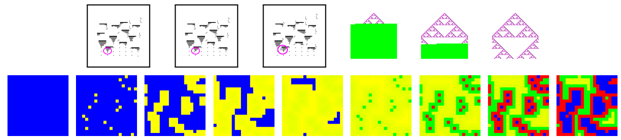

Fig. 1. MEGA, CA Wolfram code 82, and HIV SEIR model executing in VRule

Initial work has been done upon tessellations of hexagons to support macroscopic Navier-Stokes fluid flows, as have antecedent color component thresholds and consequent color-based operators such as dodge, burn, etc. While VRule has been demonstrated to be Turing complete, and capable of supporting many modeling domains including CAs, climate change, and epidemiology there remains a multitude of possibilities to explore additional formalisms supporting primitives enabling new and emerging modeling approaches. Advice and discussion regarding these, possible directions, and domains would be invaluable.

Acknowledgements

I am thankful to my advisor Professor Enrico Pontelli for supporting this work.

References

1. Anderson, M., Furnas, G.: Relating two image-based diagrammatic reasoning architectures. In: International Conference on Theory and Application of Diagrams. pp. 128–143. Springer (2010)
2. Blackwell, A.F.: Psychological issues in end-user programming. In: End user development, pp. 9–30. Springer (2006)
3. Furnas, G.W., Qu, Y.: Using pixel rewrites for shape-rich interaction. In: CHI '03: Proceedings of the SIGCHI conference on Human factors in computing systems. pp. 369–376. ACM Press, New York, NY, USA (2003)
4. Giabbanelli, P.J., Freeman, C., Devita, J.A., Rosso, N., Brumme, Z.L.: Mechanisms for cell-to-cell and cell-free spread of hiv-1 in cellular automata models. In: Proceedings of the 2019 ACM SIGSIM Conference on Principles of Advanced Discrete Simulation. pp. 103–114. ACM (2019)
5. Gil, J., Howse, J., Kent, S.: Formalizing spider diagrams. In: Proceedings 1999 IEEE Symposium on Visual Languages. pp. 130–137. IEEE (1999)
6. Hayes-Roth, F., Waterman, D., Lenat, D.B.: Principles of pattern-directed inference systems. In: Pattern-directed inference systems, pp. 577–601. Elsevier (1978)
7. Lathrop, S.D., Laird, J.E.: Extending cognitive architectures with mental imagery. In: Proceedings of the 2nd Conference on Artificial General Intelligence (2009). Atlantis Press (2009)
8. Ramirez, J.A., Baird, A.J., Coulthard, T.J., Waddington, J.M.: Testing a simple model of gas bubble dynamics in porous media. Water Resources Research **51**(2), 1036–1049 (2015)
9. Wolfram, S.: A new kind of science. Wolfram Media, Inc. (2002)
10. Yamamoto, K.: Visulan: A visual programming language for self-changing bitmap. In: Proceedings of International Conference on Visual Information Systems. pp. 88–96. Springer (1996)