DOI:10.1145/2633031

Uri Wilensky, Corey E. Brady, and Michael S. Horn

Education

Fostering Computational Literacy in Science Classrooms

An agent-based approach to integrating computing in secondary-school science courses.

There is widespread and growing agreement that computing should play a more prominent role throughout our education system. The next generation of learners will require a high level of fluency with modes of thinking and inquiry in which computers act as interactive partners. While many students experience computing (via the Web and apps), few students understand computation, and even fewer have experience using computers as tools for scientific inquiry. These skills and perspectives are essential for full and effective participation in today's (and tomorrow's) society.

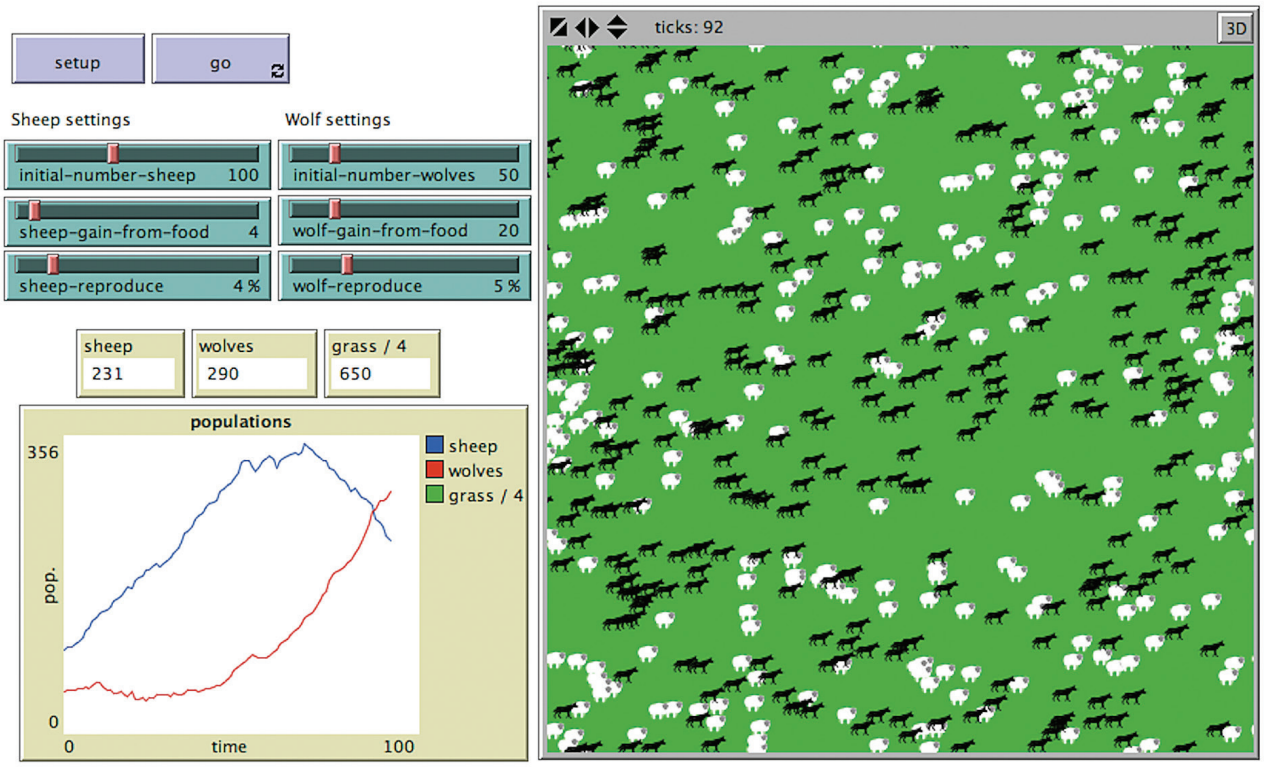
The development of computer science curricula, standards, and course requirements for secondary schools is an important and useful direction actively being pursued in a variety of initiatives. However, the success of such initiatives will depend heavily on schools' ability to hire and retain qualified teachers; on teachers' ability to implement curriculum that is applicable to scientific inquiry; and on students' ability to make room for new coursework in their already-packed schedules. Although large-scale efforts such as Code.org are offering support for computer programming instruction in schools, the magnitude of the challenge is enormous. Even more worrisome is the possibility that an elective-only CS course sequence will fail to attract a diverse population of students, exacerbating the low participation rates of women and other underrepresented groups in computing fields.

Let's introduce real computational literacy through the science classes every student takes, rather than solely through computer science classes.

A complementary approach we advocate here is to integrate computing across the range of secondary-school science courses.[a] Let's introduce real computational literacy through the science classes every student takes, rather than solely through computer science classes. Our goal is to treat computation as a core component in a broad-based cultural literacy, rather than the exclusive province of a single academic department or field of study. This approach is consonant with the views of visionary educators such as Papert[3] and diSessa[2] who have advocated for universal computational literacy as a means to provide access to "powerful ideas." While this approach faces its own set of challenges, integrating computation into science learning has several distinct advantages: it increases access to computing for all students in all schools; it enhances students' motivation for and depth of understanding of scientific principles by using computing in powerful ways; it brings science education in line with authentic scientific practice and the needs of 21st-century science; and it provides students with experiences of computers beyond searching and sorting, demonstrating the power of computation to help them make sense of their world. We can reach more students more quickly by getting science teachers to add computing into their classes, than we can by developing a nationwide cohort of computer science teachers and placing them in all our high schools. We have found that brief but intensive professional development experiences, accompanied by carefully crafted curricular materials, are sufficient to

a For more, see the Computational Thinking in STEM project at Northwestern University (http://ct-stem.northwestern.edu).

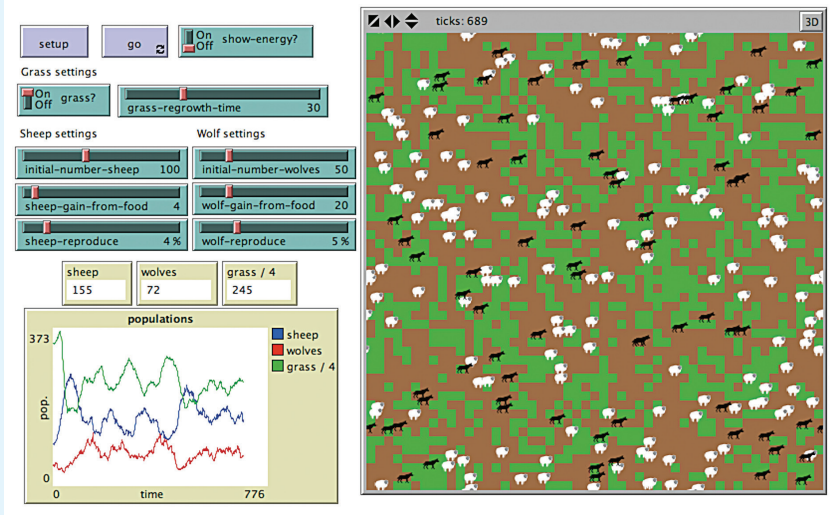
Figure 1a. An ecosystem with sheep and wolves (see http://ccl.northwestern.edu/netlogo/models/wolfsheeppredation).



support high school science teachers in bringing computational modeling into their courses. We are currently completing an NSF-funded project working successfully in this way with over 100 science classrooms in the Chicago area. [b]

In this column, we describe the use of agent-based modeling (ABM) as a powerful way to introduce computation across the secondary science curriculum. ABM is a form of computational modeling in which individual entities in a computer simulation (the agents) are given rules defining their behavior. There are two classes of agents, mobile agents that typically represent individuals such as animals or molecules, and a grid of stationary agents, as in a cellular automaton, that typically represent parts of the environment such as grass or other elements of the terrain. The collective interactions of a multitude of agents, each concurrently acting out its behavioral rules, can reveal complex, emergent patterns. The "game" of ABM is to try to generate known phenomena by defining a set of agents and rules for their behavior, or to explore the possible phenomena that can be generated with such a set by varying conditions or parameters.

Figure 1b. Adding grass adds stability.



Over the past 25 years, the Center for Connected Learning and Computer-

b For more, see the Enabling Modeling and Simulation in the Classroom project at Northwestern University with partnerships at Stanford and Vanderbilt universities (http://ccl.northwestern.edu/modelsim).

Figure 2a. Behavioral rules for each "breed" of agent.

```
ask sheep [
        wander
        try-eating-grass
        maybe-starve
        maybe-reproduce-sheep
]
```

```
ask wolves [
        wander
        try-catching-sheep
        maybe-starve
        maybe-reproduce-wolves
]
```

Figure 2b. A possible NetLogo implementation of the wolf behavior "try-catching-sheep."

```
to try-catching-sheep                   ;; A wolf-specific procedure
  if any? sheep-here [                  ;; Are there any sheep here with me?
        let prey one-of sheep-here      ;; If so, select one...
        ask prey [ die ]                ;; ...kill it (and eat it)...
        set energy energy + 20          ;; ...and gain some energy from it.
  ]
end
```

Based Modeling (CCL) at Northwestern University has developed ABM tools for education and scientific practice. The NetLogo ABM environment[c] (free and open source) is a product of this effort, and currently has hundreds of thousands of users, ranging from students in middle schools to researchers in scientific laboratories. The CCL continues to develop software and materials to support the integration of NetLogo in science classrooms.[d] This includes a wide variety of contexts and grade levels from middle school through undergraduate education and graduate school, suggesting we have made some progress toward our goal of broad reach, but we still have a long way to go.

Why Agent-Based Modeling?

One reason ABM approaches hold great potential to support science learning is that so many of the concepts students find most challenging involve connecting micro and macro aspects of scientific phenomena. The science education literature describes the many misconceptions students have about what connects these levels (see Wilensky and Resnick[5] and Chi[1]). For instance, in chemistry and physics, gas molecules collide elastically at the micro level, leading to the properties of pressure and temperature at the macro level. In biology, individual animals struggle to survive and reproduce at the individual level, leading to phenomena such as evolution, natural selection, and population dynamics at the ecosystem level. Agent-based modeling provides the means to build on intuitive understandings about individual agents acting at the micro level in order to grasp the mechanisms of emergence at the aggregate, macro level.

Because the individual-level behavior of agents is relatively simple, ABMs feature relatively simple computer programs that control the behaviors of their computational agents. On the other hand, swarms or aggregates of interacting agents can produce complex, emergent patterns that require computational power beyond the human capacity to simulate. (Thanks to decades of life under Moore's Law, however, such power is now available in virtually all personal computers and mobile devices.) Working in partnership with a computational model within an ABM environment such as NetLogo, learners can explore the connections between the micro-level behavior of individuals and the macro-level patterns that emerge from their interaction. As they work with NetLogo, students can articulate their own provisional thinking in an executable form. Running their models reveals the implications of their ideas, provokes new conjectures, and drives "debugging" cycles of modeling, execution, and refinement. This iterative process is a motivating and intellectually exciting activity, driven by

c See http://ccl.northwestern.edu/netlogo.
d See http://ccl.northwestern.edu/modelsim/ and http://ccl.northwestern.edu/simevolution/.

interactive feedback and dynamic visualizations. Working with NetLogo, kids can explore existing models by changing initial conditions and sweeping the parameter space of key variables. They can also explore what-if questions by modifying or adding behavioral rules to an existing model or creating their own models from scratch.

But ABM is not only a tool for the classroom. NetLogo is used in a wide variety of research laboratories and professional contexts, and hundreds of scientific papers using it have been published.[e] Thus, the work students do in agent-based modeling prepares them for authentic inquiry in the scientific disciplines. In the examples here, we present several uses of NetLogo and agent-based modeling to engage with core concepts across a range of topics from secondary science.

Agent-Based Modeling Across the Sciences

Population Biology/Ecology. Common topics in middle- and high-school biology include the dynamics of populations within ecosystems and food webs. Figures 1a and 1b show agent-based models of predator-prey relations between populations of wolves

The work students do in agent-based modeling prepares them for authentic inquiry in the scientific disciplines.

(who eat sheep) and sheep (who eat grass). Both wolf and sheep are modeled as having energy, losing energy by moving, gaining energy by eating, dying if they have too little energy, and expending energy to reproduce. As shown in Figure 2a, for each "breed" of agent, a simple collection of behavioral rules describes their actions and interactions. These behaviors are then defined computationally. Thus, a NetLogo implementation of the wolf behavior "try-catching-sheep" might be as shown in Figure 2b.

Learners can run their in-progress models to see how the system behaves. For instance, in this simulation learners have found that adding logic to describe the depletion and replenishment of the virtual grass led to increased stability in the ecosystem, damping the oscillations in the wolf and sheep populations.

Physics and Chemistry. In physics and chemistry, the Ideal Gas Laws provide an elegant mathematical explanation of regularities in the world, but students rarely connect these macro-level phenomena with the molecular interactions that generate them. Students using ABMs can leverage agent-based intuitions to understand the Kinetic Molecular Theory (KMT). Models like the ones in the CCL's GasLab suite[f] begin from simple collision rules for particle agents and show how pressure and temperature arise as emergent properties in the aggregate.[g] Because they are working with individual entities, high school students can reason about their interactions, going beyond the familiar topics and touching on advanced concepts, such as the Maxwell-Boltzmann distribution and phenomena of statistical mechanics (see Figure 3).

Earth Sciences. In middle school Earth science classrooms, it is common to study natural disasters such as volcanoes and forest fires. With the *Fire* model,[h] students can investigate the systems principles of critical parameters and nonlinear dynamics in the context of a simulated forest fire. Here, there are tree agents and fire agents. The trees are randomly distributed at a given density, which is controlled by a slider. The leading edge of a forest fire is represented by red agents; the rules for the trees are very simple: they "look" at their neighbors to the North, South, East, and West. If they see a tree on fire, they ignite.

Most people's intuition about this system is that a small increase in tree density should result in a little more burn (that is, a linear relationship). However, that is not what the simulation reveals. Instead, there is a critical density value, below which the forest fire dies out quickly (see Figure 4a) and above which it consumes almost the entire forest (see Figure 4b). Such critical parameters are common in complex systems throughout science, and have been recently popularized as "tipping points."

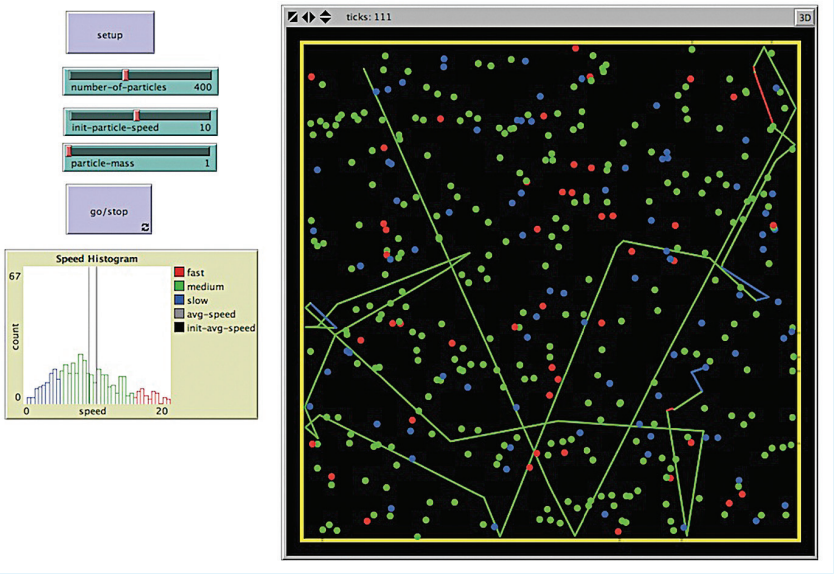
e See http://ccl.northwestern.edu/netlogo/references.shtml.

f See http://ccl.northwestern.edu/curriculum/gaslab/.

g Or, in other words, the molecules "compute" pressure and temperature.

h See http://ccl.northwestern.edu/netlogo/models/fire.

Figure 3. GasLab model of molecules in a box, colored by their speed. One of the molecules leaves a trace to facilitate following its trajectories.



Students find the simulated fire to be visually compelling, and the code behind the NetLogo model is extremely simple. Thus, the Fire model can act as an excellent introduction to agent-based modeling. Indeed, learners who engage with it often modify or extend the model to explore their own questions, including the effects of wind, alternative rules by which the fire might spread, or strategies for arresting the spread of an existing forest fire.

Figure 4a. Critical parameters. The run (57% density) is just below the critical density and the fire has burned only 7.6% of the forest.

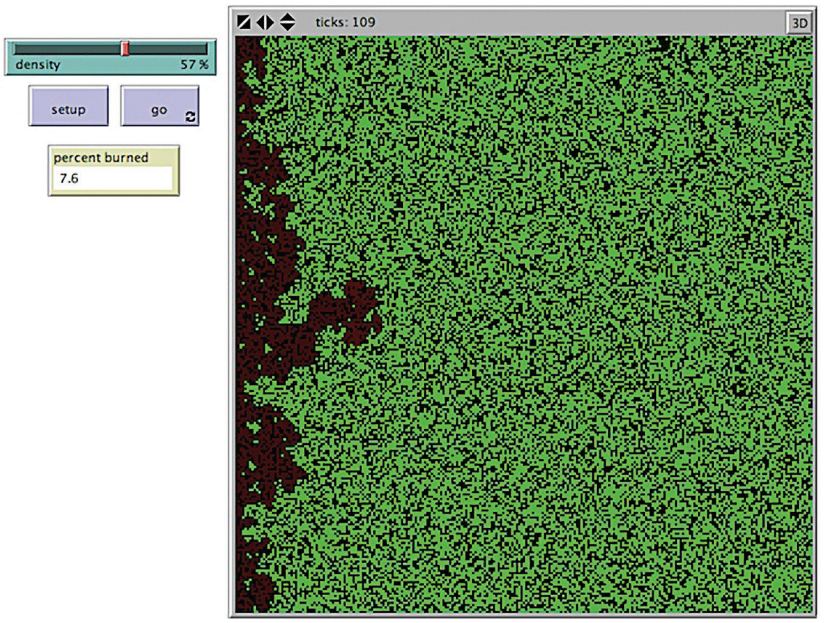
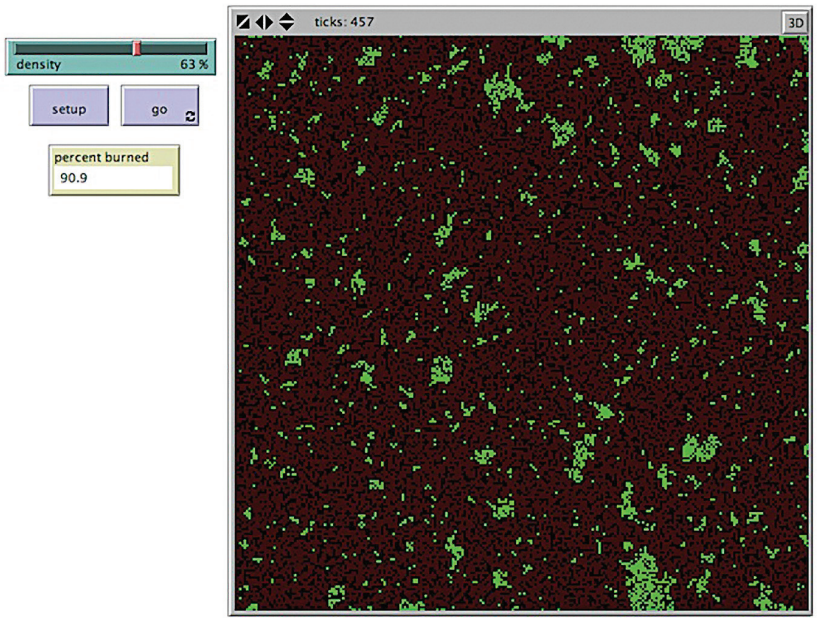


Figure 4b. Critical parameters. In contrast, the run (63% density) is above the critical value, and the fire rages on. (In the end, 90.9% of the trees burned.)



Conclusion: Agent-Based Modeling and Computation for All

We are now well into the 21st century, and we need to bring the efforts of our educational system in line with our shared sense of the growing importance of computing for all members of our society. Thinking effectively about and with computational processes is a broad-based literacy needed by all citizens to support their effective social, economic, and political participation. We have argued in this column that secondary science classrooms[i] present a compelling opportunity to build this literacy, and that ABM tools like NetLogo are an effective means to do so. By letting go of the idea that literacy with computation is the sole responsibility of educators officially working within computer science departments, we can widen our reach and arrive at our goals more quickly and effectively. C

i In this column, we have argued for and presented examples of the use of ABM in middle and high school science. However, our efforts have not been limited to either science or secondary school. There has also been considerable enthusiasm for using ABM in the *social sciences*, where we have also created materials for modeling phenomena such as wealth accumulation, segregation, and language change. We have also worked to integrate ABM into a wide range of university courses (see Wilensky and Rand[4]).

References

1. Chi, M. Commonsense conceptions of emergent processes: Why some misconceptions are robust. *The Journal of the Learning Sciences 14* (2005), 61–199.
2. diSessa, A.A. *Changing Minds: Computers, Learning, and Literacy.* MIT Press, Cambridge, MA, 2000.
3. Papert, S. *Mindstorms: Children, Computers, and Powerful Ideas.* Basic Books, New York, 1980.
4. Wilensky, U. and Rand, W. (in press). *Introduction to Agent-based Modeling: Modeling Natural, Social and Engineered Complex Systems with NetLogo.* MIT Press, Cambridge, MA.
5. Wilensky, U. and Resnick, M. Thinking in levels: A dynamic systems approach to making sense of the world. *Journal of Science Education and Technology 8*, 1 (1999), 3–19.

Uri Wilensky (uri@northwestern.edu) is a professor at Northwestern University with a joint appointment in Computer Science and the Learning Sciences. He is the author of NetLogo and the director of the CCL lab.

Corey E. Brady (cbrady@northwestern.edu) is a research assistant professor of Learning Sciences at Northwestern University.

Michael S. Horn (michael-horn@northwestern.edu) is an assistant professor at Northwestern University with a joint appointment in Computer Science and the Learning Sciences.