



# Technical Perspective

## Programming Microfluidics to Execute Biological Protocols

By Nada Amin

REPRODUCIBILITY OF EXPERIMENTAL results is a cornerstone of biology research. Today, many of these experiments are done using automated machines such as robots and microfluidic chips. However, published reports about the work explain the experimentation method in plain English, which must be interpreted by other groups to reproduce the experiment.

Biological protocols give a recipe for a biological experiment. Ideally, we would like these protocols to be specified rigorously and precisely. Once we do that, we are a step away from automation, reproducibility, and also repurposing.

Microfluidics are diverse technologies to conduct precise and repeatable experiments on small quantities of fluids. Just like integrated circuits have allowed automation of computation, microfluidic chips—coin-sized media that manipulate small quantities of liquid—promise to automate biological and chemical experiments. A common application is DNA replication, enabling small amounts of DNA to be amplified for larger-scale analyses. Reagents are held in chambers on the chip and the interconnecting fluid pathways dictate how and in what proportions reagents are to be mixed; this can be accomplished differently across chip technologies, for example, through microchannels etched into the medium or through electric fields that manipulate discrete droplets. The key benefit of a microfluidic chip is that precise analyses can be performed despite their incredibly small size; this enables massively parallel experiments, low reagent consumption, and overall lower experiment setup costs. Since fabrication of microfluidic devices is cheap, experimenters tend to iterate through their designs quickly. However, the production of lab-on-chips is not purely a matter of manufacturing—bottlenecks exist throughout the microfluidic chip design workflow, including microfluidics-aware computer-aided design tools, design verification, and barriers to entry.


With the advent of microfluidics, there is now a constrained medium in which to explore executable biological protocols. BioStream and BioCoder are the first programming languages for biological protocols in the wake of microfluidics; both are embedded in C++. BioStream separates the specification of the protocol from its realization on a microfluidic chip. BioCoder focuses on expressing high-level protocols and encompasses a variety of biological experiments, leaving the realization of those protocols on microfluidic chips or other media as future work. Developed more recently, BioScript is a simpler standalone language with an operational semantics and type system. The type system is based on a table of real hazards and can statically guarantee the experiment does not cause hazards, for example, by mixing incompatible fluids. Even more recently, Puddle, an automation platform based on microfluidics, relies on dynamic feedback rather than static checks to run experiments from bio-computing to medical diagnostics.

The hazard-free guarantee approach taken in the following paper is an example of how programming languages can help develop executable protocols that are conforming, understandable, safe, and retargetable. As in software reuse, one might be able to ‘tweak’ an executable protocol to a new purpose, and one should ensure the guarantees still carry.

Within the next decades, we can imagine that medicine, biology, and chemistry papers that contain results from wet lab experiments come with their own ‘protocol’ artifact and that such artifacts will be more formally specified. It will also be possible to formally analyze the protocols and the results of a paper, to evaluate the claims, and beyond, to evaluate the protocols’ safety, retargetability (running on different hardware), modularity (plugging multiple protocols) and repurposability (running a variant of the protocol).

In another direction, having a programming language to specify a biology protocol that is one step removed from the medium would enable a proliferation of digital media (as in Lab-on-Chips) and compilers-to-chips for more custom media (as in custom chips). We can see the trend that microfluidic chips are either general digital chips that can capture a range of experiments, or custom high-throughput chips that target a particular experiment. From a biology protocol, a compiler could automatically propose how to run the protocol on a prefabricated digital chip or how to fabricate a custom high-throughput chip layout to run the protocol.

Programming microfluidics to execute biological protocols remains an exciting avenue, with a promise still to be fulfilled. Ideally, one should be able to run the same biological protocol code on a variety of potential platforms. Works in programming languages for biological protocols can ensure a separation of concerns between the specification of a biological protocol and its realization on biological media such as microfluidic chips, fostering advances on both sides of the separation.

Finally, creating a faster loop from medical problem to diagnostic to informed decision is a robot scientist’s dream. Programming microfluidics could be put to use at various levels of safety when filtering through the myriad of potential cures to the one cure that will work for the here and now. For example, people are mining SARS-CoV-2-Human Protein-Protein Interaction for drug repurposing. Though robots are less used in high facility labs for obvious ‘gone rogue’ reasons, given the myriad of potential experiments, it would be wonderful to have a tight feedback between the future robot scientist and the robot or human experimenters. 

**Nada Amin** is an assistant professor of computer science at Harvard SEAS, Cambridge, MA, USA.

Copyright held by author.