# ChemComp: Compiling and Computing with Chemical Reaction Networks

Nicolas Bohm Agostini, Connah G. M. Johnson, William R. Cannon, Antonino Tumeo
Pacific Northwest National Laboratory,
Richland, WA 99354
(nicolas.agostini, connah.johnson, william.cannon, antonino.tumeo)@pnnl.gov

***Abstract*—The exponential growth in computing demands driven by scientific computing, data analytics, and artificial intelligence is pushing conventional CMOS-based high-performance computing systems to their physical and energy efficiency limits. As we approach the era of post-exascale computing, disruptive approaches are necessary to overcome these barriers and achieve substantial gains in energy efficiency. Analog and hybrid digital-analog computing systems have emerged as promising alternatives, offering the potential for orders-of-magnitude improvements in efficiency. Among these, biochemical computing stands out as a novel paradigm capable of leveraging the natural efficiency of chemical reactions, which have shown promise in solving optimization problems by converging to steady states. By scaling up reaction networks or reaction vessel sizes, biochemical systems present an opportunity to meet the high-performance demands of modern computing tasks. Despite their promise, significant theoretical and practical challenges remain, particularly in formulating and mapping computational problems to chemical reaction networks (CRNs) and designing viable biochemical computing devices. This paper addresses these challenges by introducing new ideas to ChemComp, a compilation and emulation framework for chemical computation. This work describes the mechanisms through which solutions to ordinary differential equations (ODEs) that can be represented as CRN systems can be achieved. Furthermore, we explain the design principles of an ODE dialect implemented as a multi-level intermediate representation (MLIR) compiler extension that will be coupled with existing infrastructure. We demonstrate the potential of our framework through a case study emulating a simplified chemical reservoir computing device. This work establishes foundational tools and methodologies necessary to harness the computational power of chemistry, paving the way for the development of energy-efficient, high-performance computing systems tailored to contemporary and future computational needs.**

***Index Terms*—MLIR, ODEs, Chemical Reactions**

## I. Introduction

Biological cells exhibit exceptional capabilities in solving optimization problems efficiently, with low energy consumption, and operate in parallel [1]. This is accomplished through complex chemical reactions that enable the cell to adapt optimally for survival within its environment. These biochemical processes can be detached from their natural context and integrated into designed chemical reaction networks (CRNs) [2] to harness their computational power. CRNs have been demonstrated to compute nonlinear functions implementing logic gates [3] and can theoretically be employed as a computing machine that excels at solving complex problems. Recent research has proven the Turing completeness of carefully designed CRNs, making them capable of performing arbitrary computations [4]–[6].

Chemical reaction computing boasts inherent parallelism and scalability, making it an attractive alternative for high-performance computation. Each chemical reaction system can be considered a computational unit, with input, output, and intermediate concentrations (or their derivatives) determining the solution. For instance, a cup of water contains approximately 13 moles of molecules ($10^{24}$ molecules in total), each potentially participating in various reactions concurrently. This offers the potential for numerous parallel computing units within a compact device. The biochemical breakdown of glucose during the glycolysis process within cells exemplifies the computational energy scale [7]. During glycolysis, through a sequence of 10 chemical reactions, one mole of glucose is processed into two moles of pyruvate while generating a net gain of 4 moles of ATP (adenosine triphosphate) used to drive further cellular reactions. The process consumes roughly 68 kJ of energy to drive the reactions. The slowest rate - determined phosphorylation reaction step - takes about one second. This equates to $10^{23}$ potential operations (OPS) per reaction in a cup of water. Comparatively, Oak Ridge Leadership Computing Facility's Frontier exascale supercomputer performs on the order of $10^{18}$ floating-point OPS with a power consumption of 29,000 kW [8]. The gain in energy efficiency is derived from comparing the glycolysis reaction's OPS/W to Frontier's floating-point OPS/W. Consequently, the biochemical OPS/W (energy efficiency) gain over Frontier is roughly $4 \times 10^7$. Note that this makes the unrealistic assumption of perfect exploitation of chemical computation, which, in fact, relies on challenging and noisy single-molecule measurements. Nevertheless, even a modest utilization of available computational capacity would yield power and performance gains over modern supercomputers.

However, challenges still exist in understanding how to leverage these reactions for computational purposes effectively. End-to-end solutions for efficiently compiling mathematical equations onto CRNs remain elusive. Despite these obstacles, the potential benefits offered by biochemical operations and CRNs are substantial – promising reduced energy consumption and increased scalability (both in terms of mapped operation complexity and chemical system size).

In this paper, we expand upon ChemComp [9], which introduced a comprehensive compilation and emulation framework

978-3-9826741-0-0/DATE25/© 2025 EDAA

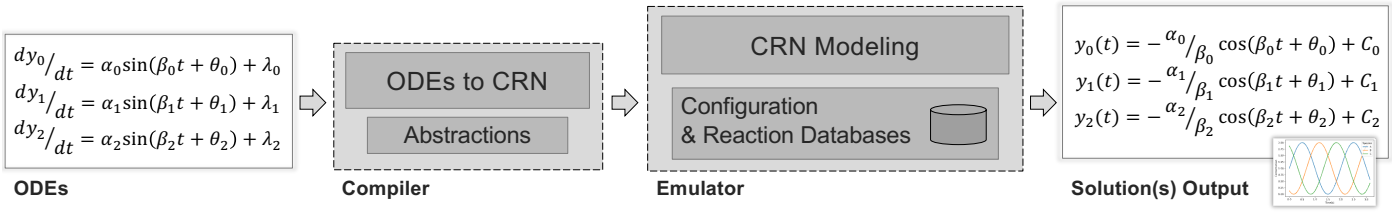


Fig. 1. ChemComp proposed compilation and emulation workflow takes as input a system of ODEs (ordinary differential equations) and converts them to a set of chemical reactions for emulation. The ODEs are abstracted into mathematical patterns and abstract chemical reactions. These abstract reactions can be instantiated using databases of biochemical reactions. These reactions can then be emulated or implemented onto a chemical computing device.

as depicted in Figure 1. This extension addresses existing challenges by describing a Multi-Level Intermediate Representation (MLIR) [10] abstraction specifically designed for representing ODEs. We also propose a method to translate ODEs into CRNs following the mass action chemical kinetics law. This design decomposes ODEs into modular chemical reactions that symbolize basic arithmetic operations, such as addition and multiplication, that are implemented through changes in species concentrations. The objective is to create a compositional representation where complex ODEs are realized as networks of interconnected chemical reactions, thereby facilitating efficient compilation and transformation within our framework.

## II. Background

Chemical computing leverages chemical reactions as fundamental components for computational architectures by utilizing existing software abstractions and compilation frameworks. Chemical reservoirs have been explored as promising devices for such computations [11]–[14]. When addressing continuous problems like ODEs, a "chemical advantage" emerges from the fact that executing a chemical reaction system provides time-dependent solutions for the ODE implemented by their reactions and species' concentration. This inherent capability highlights their foundational role in scientific applications and underscores their potential for solving complex computational challenges.

In computational models, representing chemical reactions involves several critical aspects. The *reactants and products* must be clearly identified to determine which chemicals or species are involved in each reaction. Next, understanding the *reaction mechanisms* is essential for establishing how these reactions proceed and linking reactants with their respective products. Proper attention to *stoichiometry* ensures that the equation coefficients accurately represent the correct proportions of reactants, maintaining mass balance to achieve precise product outcomes. Additionally, incorporating *kinetics and thermodynamics* involves including rate constants to define reaction rates and considering energy changes for a comprehensive description. These elements collectively form the foundation of chemical computing architectures.

Multiple chemical reactions can be modeled as "reaction graphs", where nodes represent chemical species and edges represent reactions. Networks of these reactions, or CRNs, facilitate modeling reaction rates using ODEs [15], allowing emulation to solve typical scientific problems.

MLIR [10] provides a flexible compiler framework for representing computational graphs across various domains, such as machine learning and quantum computing [16]. It allows defining dialects to encapsulate different concepts. While MLIR lacks native support for chemical reactions, ChemComp utilizes it to design the compilation flow by describing biochemical processes as composable operations, enabling higher-level mathematical representations.

Previous work [9], [17] introduced ChemComp, along with the foundational concepts of an MLIR dialect designed for abstract chemical reactions, termed `acr`. In this dialect, abstract chemical reactions are articulated using specific operations and types. The `acr` operations reason about generic biochemical species and their interactions. The ChemComp framework aims to encapsulate these concepts at both the chemical system level and within a compilation infrastructure. This is achieved by enabling relevant abstractions and establishing the necessary compilation flow.

## III. Methods

In this section, we present biochemical principles and compiler methodology that will facilitate computations using CRNs. We explore the modeling of chemical reactions — an essential element for our emulation platform — and detail our strategy for abstracting critical elements during ChemComp's compilation process.

### A. Modeling Chemical Reaction Networks

Equation 1 represents a chemical reaction $r$, where $I_i^r$ is the number of molecules of biochemical species $i$ consumed (inputs), $O_i^r$ is the number of molecules of species $i$ produced (outputs), and $p_r(X)$ is the propensity or rate of the reaction depending on the concentrations $X$.

$$\sum_i I_i^r X_i \xrightarrow{p_r(X)} \sum_i O_i^r X_i \tag{1}$$

We expand the rules governing chemical reactions from Equation 1. Equation 2 shows an example with four biochemical species.

$$\mathrm{aA} + \mathrm{bB} \longrightarrow \mathrm{cC} + \mathrm{dD} \tag{2}$$

This equation illustrates how chemicals $A$ and $B$, known as reactants, interact to form products $C$ and $D$. The stoichiometry coefficients $\{a, b, c, d\}$ indicate the proportions of each reactant and product involved in the reaction. When reactions occur

concurrently at the same location, they can combine to create CRNs, as demonstrated by the CRN comprising Equations 3 and 4.

$$\mathrm{A} + \mathrm{B} \longrightarrow \mathrm{C} \tag{3}$$
$$\mathrm{C} \longrightarrow \mathrm{D} \tag{4}$$

The rules governing CRN dynamics can be modeled using a system of ODEs. Equation 5 describes the rate of change of species $X_i$ over time, calculated as the net production $O_i^r - I_i^r$ multiplied by the reaction rate $p_r(X)$, summed over all reactions $r$.

$$\frac{dX_i}{dt} = \sum_r (O_i^r - I_i^r) p_r(X) \tag{5}$$

Equations 6 - 9 present the detailed form for our example, where the rate of change is directly proportional to the concentration of chemicals. Here, $[x]$ denotes the concentration of chemical $x$, and $k_i$ represents the kinetic coefficient.

$$\frac{d[A]}{dt} = -k_1[A][B] \tag{6}$$
$$\frac{d[B]}{dt} = -k_1[A][B] \tag{7}$$
$$\frac{d[C]}{dt} = k_1[A][B] - k_2[C] \tag{8}$$
$$\frac{d[D]}{dt} = k_2[C]. \tag{9}$$

These equations can be resolved using standard ODE solvers (see Figure 2). The reactions are applicable for performing logical operations [18] at steady state. For instance, a high concentration of $D$ is achievable only when both $A$ and $B$ are supplied to the reaction system, analogous to an AND operation at steady state.

Running the described chemical reactions provides a time-dependent solution for the corresponding system of ODEs. The ODEs model the dynamic changes in concentrations of biochemical species involved in the reactions and, consequently, running these chemical reactions in a real-world setup would provide an experimental solution to the corresponding system of ODEs. In practice, by observing and measuring the dynamic changes in concentrations of biochemical species involved in these reactions, we can directly obtain how concentrations (the time-dependent variables) evolve over time. This practical approach offers insights into the system behavior under specified initial conditions and kinetic parameters. This methodology demonstrates a powerful tool: leveraging chemical reactions to find solutions for ODEs that can be described using CRNs.

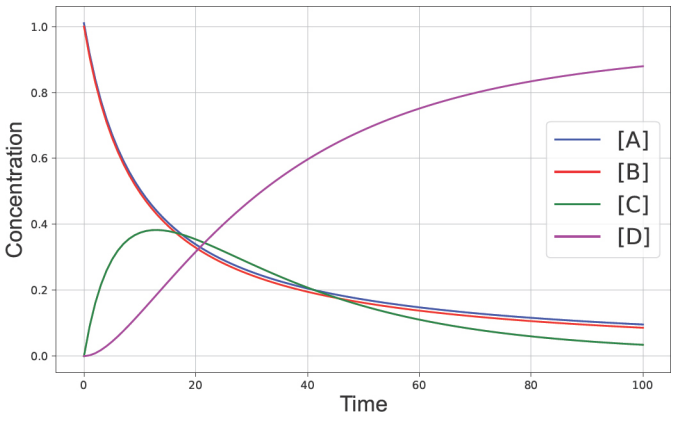


Fig. 2. The CRN Equations 6-9 are solved using odeint from the scipy package. The initial concentrations were $([A],[B],[C],[D]) = (1.01, 1.0, 0.0, 0.0)$ with reaction kinetic coefficients of $(k_1, k_2) = (0.1, 0.05)$.

*Chemical reactions, ODEs, and Solutions*

Table I presents a collection of fundamental CRNs along with their corresponding ODEs, and analytical solutions. These CRN motifs represent basic patterns that can be used to systematically compose and model more complex reactions, allowing us to construct specific ODEs with simple biochemical systems.

The concept is to leverage these pre-defined motifs as building blocks in our framework. By composing these motifs, we can create intricate chemical reaction networks capable of emulating the dynamics described by user-specified ODEs. This approach allows us to harness the time evolution behavior of these CRNs during numerical evaluation to yield solutions for complex ODE systems.

For instance, a simple unimolecular decay or reversible reaction can be directly mapped using their respective motifs. More complex reactions, such as chain reactions and dimerization, can similarly be modeled by composing appropriate sequences or combinations of these basic units. This method will provide a path to translate any given ODE into a corresponding CRN representation, thereby facilitating the solution process through chemical simulation or, in the future, execution of the CRN system.

In future work, we will extend this methodology by developing more sophisticated compositions of these CRN motifs, enabling great flexibility in implementing diverse user-defined ODEs.

### B. Expressive Abstractions

MLIR [10] is a powerful framework designed for building reusable and extensible compilers and infrastructure. It enables developers to define domain-specific dialects that can represent high-level abstractions, allowing for efficient transformations and optimizations across different levels of abstraction. MLIR supports composability by enabling the construction of complex models from simpler components, interoperability through integration with existing dialects, and expressiveness in capturing a wide range of computational patterns.

In order to represent input ODEs and CRNs, we need MLIR abstractions that are composable, interoperable, and expressive. Composability in an MLIR dialect allows for the representation of complex ODE models by combining simple mathematical operations. In doing so, complex behaviors are constructed from well-understood foundational building blocks. Interoperability permits leveraging other MLIR dialects when available and necessary. Expressivity is a key feature for representing

TABLE I
CHEMICAL REACTION NETWORKS AND ODES.

| **Name of the Reaction** | **Reaction System or CRN** | **ODE Representation** | **Solution** |
|---|---|---|---|
| Unimolecular Decay | $A \xrightarrow{k} B$ | $\frac{d[A]}{dt} = -k[A]$ | $[A](t) = [A]_0 e^{-kt}$ |
| Reversible Reaction | $A \leftrightarrow B$ | $\frac{d[A]}{dt} = -k_f[A] + k_r[B]$<br>$\frac{d[B]}{dt} = k_f[A] - k_r[B]$ | $[A](t) =$<br>$\frac{k_r C}{k} + \left([A]_0 - \frac{k_r C}{k}\right) e^{-kt}$<br>where $k = k_f + k_r$, $C = [A]_0 + [B]_0$ |
| Irreversible Bimolecular Reaction | $A + B \xrightarrow{k} C$ | $\frac{d[A]}{dt} = -k[A][B]$<br>$\frac{d[B]}{dt} = -k[A][B]$<br>$\frac{d[C]}{dt} = k[A][B]$ | $[A](t) = \frac{[A]_0}{1+k[A]_0 t}$<br>(for $[A]_0 = [B]_0$) |
| Chain Reaction | $A \xrightarrow{k_1} B,\ B \xrightarrow{k_2} C$ | $\frac{d[A]}{dt} = -k_1[A]$<br>$\frac{d[B]}{dt} = k_1[A] - k_2[B]$<br>$\frac{d[C]}{dt} = k_2[B]$ | $[A](t) = [A]_0 e^{-k_1 t}$ |
| Dimerization Reaction | $2A \xrightarrow{k} B$ | $\frac{d[A]}{dt} = -2k[A]^2$<br>$\frac{d[B]}{dt} = k[A]^2$ | $[A](t) = \frac{[A]_0}{1+2k[A]_0 t}$ |

our selected range of complex ODE models, including systems with linear and nonlinear terms. Finally, representing ODEs and CRNs requires operations akin to MLIR's `builtin.module` and `func.func`, which hold other MLIR operations describing the desired functionality.

In ChemComp, we are implementing a compilation flow that converts mathematical operations to generic bio-chemical operations of well-known reaction behavior, providing translations for relevant classes of ODEs. We describe some of the compilation steps and infrastructure next. The `math` and `arith` dialects in MLIR are used to *represent underlying patterns* inside ODEs as a set of mathematical operations. These mathematical operations will then be *canonicalized* and *pattern-matched* to identify the suitable abstract chemical reaction motifs and species involved in the ODE's representation.

## IV. COMPILATION AND EMULATION

The ChemComp framework initiates its process when a user specifies an ODE representing a computational problem. Upon receiving this input, the compiler selects suitable chemical reactions to encapsulate each component and interaction within the ODE. This selection involves mapping mathematical functions and operations to their biochemical counterparts (described in Section III-A) through a predefined dialect of abstract chemical reactions. These selected reactions are represented using the MLIR framework, ensuring that the computational semantics of the original ODE are faithfully captured in the form of CRNs. Finally, a simulator interprets this CRN to generate solutions for the initial ODE. By emulating the dynamic behavior of chemical reactions, the simulation provides valuable insights into how the system evolves through time, representing solution outcomes.

This section describes the high-level ideas of an `ode` dialect to abstract ODEs and recalls concepts of the `acr` dialect relevant to the transformation of ODEs into CRNs.

### A. The ODE Dialect

An ODE expresses the relationship between dependent variables (e.g., $y$) and their derivatives with respect to an independent variable. ODEs are commonly used to model dynamic systems, where the rate of change of a quantity depends on the quantity itself and/or other factors. In the design of the ODE dialect, we aim to capture several key concepts that are essential for representing ODEs within a structured framework.

Firstly, it is crucial to define dependent variables, which represent quantities whose changes over time are being modeled. For instance, in an equation where $y$ changes with respect to another variable, $y$ would be identified as the dependent variable. Further, there must be an independent variable, such as $t$, against which derivatives of the dependent variable are computed.

Additionally, parameters or constants form a vital component of ODEs. These are fixed values like $k$ and $b$ that play significant roles in determining the behavior and solutions of differential equations without themselves being functions of time or other variables within the equation.

The relationships between these elements are encapsulated through equations that express how derivatives relate to each other and other terms within the system. These equations form the backbone of ODE modeling, defining how a system evolves over time based on the input parameters and initial conditions.

To facilitate this representation in the MLIR framework, a range of types has been introduced within the ODE dialect:

- The type `!ode.dependent` is used to represent dependent variables, which will be mapped to our species concentrations.
- The type `!ode.independent` captures independent variables, which will be mapped into the time variable.
- The type `!ode.param` denotes parameters or constants, exemplified by values like $k$ representing reaction rates.
- Derivatives of the dependent variables are represented using the type `!ode.derivative`, which would be used for expressions like $\frac{dy}{dt}$.

In terms of operations, the dialect incorporates basic arithmetic functions derived from the `arith` and `math` dialects. These include operations for addition, subtraction, multiplica-

tion, division, and others. These foundational operations allow for the representation of complex ODEs systems.

To represent the logic of an ODE, we utilize the `ode.equation` operation. This operation accepts derivatives, dependent variables, independent variables, and parameters as inputs, enabling the definition of a single differential equation within the system. For defining a collection of equations that interact with each other, we use the `ode.system`. This operation maps inputs to outputs, effectively creating a structured representation of multiple interacting ODEs.

By adhering to this structured approach, we ensure that the complex relationships and behaviors inherent in systems described by differential equations are accurately represented within the MLIR framework. This approach will enable us to analyze and transform these systems into our CRN representation through pattern matching and re-write rules and, in the future, map this representation into other targets.

### B. The Abstract Chemical Reaction Dialect

In previous work [17], we introduced an `acr` dialect for abstract chemical reactions, featuring operations with variable arguments (like *species* and *rate constant*) and constant bodies (such as *stoichiometry*). This design enables flexible composition and efficient transformations in the compilation process. The framework facilitates the translation of mathematical patterns into chemical reactions through operations such as `acr.react` and `acr.reaction`, which define reaction stoichiometry and kinetics. A mathematical function `f(x)` can be represented by multiple instantiated reactions within a network, encapsulated using the `acr.crn` operation. This allows for complex interactions modeled as independent reactions that ultimately contribute to a single CRN. These CRNs are further transformed into their generic forms via the `acr.crn_generic` operations, leveraging stoichiometric matrices for compact representation. Our methodology builds upon previous work and is designed to integrate with external databases of chemical reactions, like BiGG [19], streamlining species and reaction instantiation in future iterations.

### C. Emulation of a Chemical Computing Device

To demonstrate ChemComp's backend, we emulate a reservoir computing device [20] using the mapped reactions. Reservoir computing is a computational framework that leverages dynamic systems—often referred to as "reservoirs"—to process temporal patterns and perform complex computations. In this context, we utilize chemical reaction networks CRNs as our reservoir, exploiting their inherent dynamics to solve ODEs.

We emulate the chemical reactions, as discussed in section III-A, and we combine the concentrations of these species as a linear basis with coefficient constants $\alpha \in \mathcal{R}$, which are learned via linear regression. This approach enables us to apply linear corrections to ODEs, effectively fine-tuning the solution by adjusting for discrepancies between the emulated and desired outputs.

$$y(t) = \sum_{i=0}^{|\mathcal{C}|} \alpha_i c_i(t) \tag{10}$$

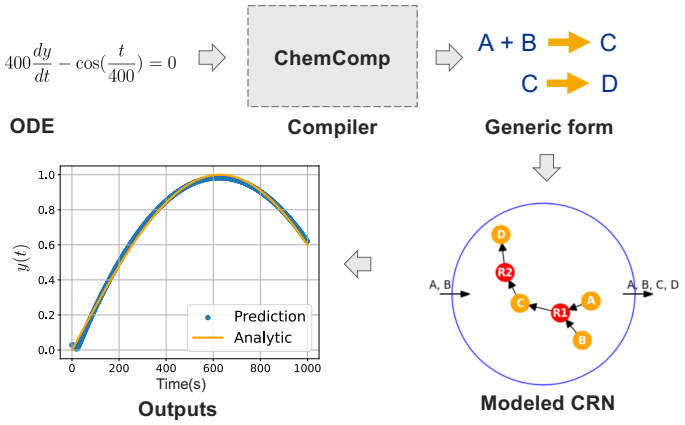


Fig. 3. Demonstration of the complete ChemComp workflow. ODEs can be mapped to a set of chemical reactions, which can be provided to a chemical reservoir. The reservoir then learns the linear basis coefficients to map the concentrations to the target problem.

Here, the basis coefficients $\alpha_i$ and concentrations $c_i(t)$ compose the ODE solution $y(t)$. In Figure 3, we illustrate the compilation of a target ODE:

$$400\frac{dy}{dt} - \cos\left(\frac{t}{400}\right) = 0 \tag{11}$$

The ODE can be represented by a set of generic reactions. These are then emulated and utilized to determine the computational solution through the linear basis in Equation 10. The predicted solution is compared to the analytic solution of the ODE given by,

$$y(t) = \frac{1}{400}\sin\left(\frac{t}{400}\right) \tag{12}$$

showing that the CRN is capable of representing the target solution using the ChemComp workflow. This demonstrates the utility of reservoir computing in achieving accurate and efficient emulation of ODEs through chemical systems.

## V. Related Work

Others have investigated how to leverage CRNs for computational purposes. One prominent example is CRN++ [21], which introduces a language and compiler for programming deterministic chemical kinetics to facilitate computation. It converts imperative programming constructs into sets of chemical reactions that execute the corresponding commands, transforming initial species concentrations into their steady-state values. The composability of each set is ensured by maintaining input concentrations at steady state.

However, CRN++ focuses on achieving these steady states without considering the temporal dynamics during concentration changes. This contrasts with ChemComp's approach, which seeks to track the concentration evolution such that a linear combination of these concentrations represents dependent variables of an ODE over time.

While CRN++ successfully translates several discrete algorithms into chemical reactions, it does not provide guidelines for implementing hypothetical one-way reactions chemically or

account for the temporal progression of species concentrations. To address this limitation, our emulation platform inputs compiled CRNs representing target systems and outputs emulations that implement these equations dynamically. Previous work demonstrates how physics-based methods, constraining reaction kinetics through thermodynamics, ensure realistic reaction emulation [22].

In specific contexts, CRNs can model probability density functions when the maximum probability is one [23]. This approach provides a novel method for simulating stochastic processes. A related concept is Markov chains, where future states depend solely on the current state, not past ones [24]. By mapping state transitions to chemical reactions, CRNs can effectively model population dynamics over time.

Quantum extensions such as quantum Markov processes [25] and quantum kinetic theory [26] draw parallels with mass-action kinetics, focusing on the probabilistic evolution of quantum states through interactions and decoherence. This research area is promising due to faster reaction timescales, suggesting potential for high-performance computing solutions using alternative reaction platforms.

## VI. Conclusion

The acceleration of scientific computation requires disruptive computing paradigms to provide energy-efficient modes of computation. Here, we have discussed extensions to ChemComp, a compilation and emulation workflow that seeks to utilize chemistry for energy-efficient and scalable computation. We show design concepts for an ordinary differential dialect, `ode`, and we describe CRN motifs that can be used to compose user-defined ODEs. The CRN output of our workflow was then emulated to demonstrate the computing capabilities of ODE solutions with chemical reactions. In the future, we will map generic reactions used in this emulation to concrete reactions found in biochemical systems. This work outlines a strategic plan to leverage chemical computing to address mathematically significant scientific problems. By following this plan and studying the necessary hardware, we can attain energy-efficient chemical computation.

## Acknowledgment

This research was supported by funding from the U.S. Department of Energy (DOE), Office of Advanced Scientific Computing Research (ASCR), as part of the ChemComp project under the Exploratory Research for Extreme-Scale Science (EXPRESS2023) program. Pacific Northwest National Laboratory (PNNL) is managed and operated by Battelle under contract DE-AC05-76RL01830 for the U.S. DOE.

## References

[1] D. A. Sivak and M. Thomson, "Environmental statistics and optimal regulation," *PLOS Computational Biology*, vol. 10, no. 9, pp. 1–12, 09 2014.

[2] M. Wen, E. Spotte-Smith, S. Blau, and et al., "Chemical reaction networks and opportunities for machine learning," *Nat Comput Sci*, vol. 3, pp. 12–24, 2023.

[3] M. Vasić, C. Chalk, A. Luchsinger, S. Khurshid, and D. Soloveichik, "Programming and training rate-independent chemical reaction networks," *Proceedings of the National Academy of Sciences*, vol. 119, no. 24, 2022.

[4] F. Fages, G. Le Guludec, O. Bournez, and A. Pouly, "Strong turing completeness of continuous chemical reaction networks and compilation of mixed analog-digital programs," in *Computational Methods in Systems Biology*, J. Feret and H. Koeppl, Eds., 2017, pp. 108–127.

[5] O. Bournez, D. S. Graça, and A. Pouly, "Polynomial time corresponds to solutions of polynomial ordinary differential equations of polynomial length," *J. ACM*, vol. 64, no. 6, oct 2017.

[6] D. Soloveichik, M. Cook, E. Winfree, and J. Bruck, "Computation with finite stochastic chemical reaction networks," *Natural Computing: An International Journal*, vol. 7, no. 4, p. 615–633, Dec. 2008.

[7] B. Alberts, A. Johnson, J. Lewis, and et al., *Molecular Biology of the Cell*. New York, NY, USA: Garland Science, 2002.

[8] D. Schneider, "The exascale era is upon us: The frontier supercomputer may be the first to reach 1,000,000,000,000,000,000 operations per second," *IEEE Spectrum*, vol. 59, no. 1, pp. 34 – 35, 2022.

[9] C. Johnson, N. B. Agostini, W. Cannon, and A. Tumeo, "Computing with a chemical reservoir," in *2024 IEEE International Conference on Rebooting Computing (ICRC)*, San Diego, CA, 2024, pp. 1–9.

[10] C. Lattner, M. Amini, U. Bondhugula, A. Cohen, A. Davis, J. Pienaar, R. Riddle, T. Shpeisman, N. Vasilache, and O. Zinenko, "Mlir: Scaling compiler infrastructure for domain specific computation," in *2021 IEEE/ACM International Symposium on Code Generation and Optimization (CGO)*. Seoul, South Korea: IEEE, 2021, pp. 2–14.

[11] *Reservoir Computing with Random Chemical Systems*, ser. Artificial Life Conference Proceedings, vol. ALIFE 2020: The 2020 Conference on Artificial Life, 07 2020.

[12] S. Kan, K. Nakajima, T. Asai, and M. Akai-Kasaya, "Physical implementation of reservoir computing through electrochemical reaction," *Advanced Science*, vol. 9, no. 6, p. 8, 2022.

[13] G. Abdi, L. Alluhaibi, E. Kowalewska, T. Mazur, K. Mech, A. Podborska, A. Sławek, H. Tanaka, and K. Szaciłowski, "Reservoir computing and photoelectrochemical sensors: A marriage of convenience," *Coordination Chemistry Reviews*, vol. 487, p. 215155, 2023.

[14] M. G. Baltussen, T. J. de Jong, Q. Duez, W. E. Robinson, and W. T. S. Huck, "Chemical reservoir computation in a self-organizing reaction network," *Nature*, vol. 631, pp. 549–555, 2024.

[15] L. Cardelli, "From processes to odes by chemistry," in *Fifth Ifip International Conference On Theoretical Computer Science – Tcs 2008*, G. Ausiello, J. Karhumäki, G. Mauri, and L. Ong, Eds. Boston, MA: Springer US, 2008, pp. 261–281.

[16] A. McCaskey and T. Nguyen, "A mlir dialect for quantum assembly languages," in *2021 IEEE International Conference on Quantum Computing and Engineering (QCE)*. Broomfield, CO, USA: IEEE, 2021, pp. 255–264.

[17] N. B. Agostini, C. Johnson, W. Cannon, and A. Tumeo, "Chemcomp: A compilation framework for computing with chemical reaction networks," in *30th Asia and South Pacific Design Automation Conference (ASP-DAC)*, Tokyo, Japan, 2025, pp. 1–9.

[18] E. Katz, "Boolean logic gates realized with enzyme-catalyzed reactions – unusual look at usual chemical reactions," *ChemPhysChem*, vol. 20, no. 1, pp. 9–22, 2018.

[19] C. J. Norsigian, N. Pusarla, J. L. McConn, and et al., "Bigg models 2020: multi-strain genome-scale models and expansion across the phylogenetic tree," *Nucleir Acids Research*, vol. 48, no. 1, pp. 402–406, 2020.

[20] M. Yan, C. Huang, P. Bienstman, P. Tino, W. Lin, and J. Sun, "Emerging opportunities and challenges for the future of reservoir computing," *Nature Communications*, vol. 15, no. 1, p. 2056, 2024.

[21] M. Vasic, D. Soloveichik, and S. Khurshid, "Crn++: Molecular programming language," in *DNA Computing and Molecular Programming: 24th International Conference, DNA 24, Jinan, China, October 8–12, 2018, Proceedings*. Berlin, Heidelberg: Springer-Verlag, 2018, p. 1–18.

[22] W. R. Cannon and S. E. Baker, "Non-steady state mass action dynamics without rate constants: dynamics of coupled reactions using chemical potentials," *Phys Biol*, vol. 14, no. 5, p. 055003, 2017.

[23] C. Gardiner, *Elements of Stochastic Methods*. Melville, NY: AIP Publishing LLC, 2021. [Online]. Available: https://doi.org/10.1063/9780735423718

[24] F. López-Caamal and T. T. Marquez-Lago, "Exact probability distributions of selected species in stochastic chemical reaction networks," *Bulletin of mathematical biology*, vol. 76, pp. 2334–2361, 2014.

[25] S. Gudder, "Quantum Markov chains," *Journal of Mathematical Physics*, vol. 49, no. 7, p. 072105, 07 2008. [Online]. Available: https://doi.org/10.1063/1.2953952

[26] M. Bonitz, *Quantum kinetic theory*. Springer, 2016, vol. 412.