NoC-Enabled Multicore Architectures for Stochastic Analysis of Biomolecular Reactions

Turbo Majumder[1], Xian Li[2], Paul Bogdan[3], Partha Pande[2]
[1] Department of Electrical Engineering, Indian Institute of Technology Delhi, Hauz Khas, New Delhi 110016, India
[2] School of EECS, Washington State University, PO BOX 642752, Pullman, WA, 99164-2752 USA
[3] Department of Electrical Engineering, University of Southern California, 3740 McClintock Avenue, Los Angeles, CA 90089-2560
[1] turbo@ee.iitd.ac.in [2] xli1@eecs.wsu.edu [3] pbogdan@usc.edu [4] pande@eecs.wsu.edu

ABSTRACT
Recent medical challenges such as cancer, drug-resistant microbes or diabetes crucially affect human health. To tackle these, modern medicine must analyze molecular interactions and rely on powerful computational platforms for the design and performance evaluation of medical therapies. Towards this end, we propose a Network-on-Chip (NoC)-based multicore platform enabling the efficient analysis of stochastic molecular interactions among biological entities. Our in-depth analysis of the stochastic interactions among biological components and the characterization of their computational and communication requirements allows us to design a high-performance NoC architecture sustaining a throughput of over 1.36E5 events/ms, while consuming only 15 mJ per 1E5 stochastic events. Our proposed NoC-based multicore can offer a throughput improvement of 23% over a regular mesh-based NoC, while consuming 20% less energy.

Keywords
Network-on-Chip, multicore, cyber-physical system, personalized medicine, gene therapy, stochastic simulation.

1. INTRODUCTION
There is a growing recognition of the importance of molecular markers and their behavior as disease precursors. Consequently, there is a growing interest in developing computational platforms that can assist disease diagnosis, drug development, measure drug efficacy, and monitor patient treatment. In addition, there are also urgent needs for computational platforms that can perform vast amounts of bio-chemical stochastic simulation efficiently in order to understand the cellular biological processes. Of note, bio-chemical stochastic simulation refers to running a Monte-Carlo type of investigation in which a set of chemical species interact in space and time by following probabilistic rules. It requires a significant number of interactive simulations, which can take a long time via software implementations. In addition, spatio-temporal interactions introduce dependencies that prevent one from simply executing independent runs in parallel.

Towards this end, in this paper, we propose a highly optimized NoC architecture for multi-scale spatio-temporal stochastic simulation and analysis of biological processes (e.g., gene regulation, biochemical cascades, cell-cell interactions). We begin by developing a mathematical framework for bio-chemical stochastic reactions, which accounts for the spatial and temporal interactions. This represents a major improvement compared to traditional Gillespie stochastic simulation algorithm [5][6]. More precisely, we tessellate the 3D space into small regions that contain specific types and number of bio-chemical reactants and investigate how the physico-chemical interactions (molecule generation, molecular diffusion, molecular absorption) dictate the communication and computation workloads. Then, we optimize both the processing elements and the communication infrastructure in order to provide the highest application throughput. In summary, the salient contributions of this paper are as follows:

A. We propose a non-trivial enhancement to the traditional Gillespie stochastic simulation algorithm by incorporating spatial information in order to account for the heterogeneity of biological systems, which generates a complex communication pattern.

B. We profile our algorithm and determine the characteristics of computation and communication workloads. In particular, we identify local and long-range communication spawned from this model, which we encapsulate in the form of a weighted dependency graph (WDG).

C. We develop an optimization algorithm for NoC design based on the above, which proceeds as follows: (i) We analyze the WDG and perform a clustering of the vertices representing reaction channels in order to balance computation and communication workload. (ii) We propose an algorithm for deploying on-chip wireless links for long-range communication dictated by the spatial interactions between bio-chemical reaction systems.

D. We demonstrate the validity and merits of our approach by considering a pertinent use-case, namely gene therapy. We demonstrate significant improvements in performance metrics (e.g., computation throughput, energy consumption) over traditional mesh-based NoC implementations.

The remainder of this paper is organized as follows: Section 2 briefly reviews the research work in the field of NoC optimization for biomedical applications, and compares and contrasts related computational approaches for studying biomolecular reaction systems with our approach. Section 3 outlines the design of NoC architecture for bio-chemical stochastic analysis and describes the gene therapy problem, which will be used to evaluate performance of the proposed NoC design. Section 4 summarizes our experimental and validation results. Finally, Section 5 concludes the paper and outlines some future work.

2. RELATED WORK
NoCs have emerged as communication backbones to enable a high degree of integration in multicore SoCs [2] [9]. In spite of all the advantages, the existing method of implementing a NoC with planar metal interconnects is deficient due to high latency and significant power consumption arising out of multi-hop links used in data exchanges. NoCs have been shown to perform better by inserting long-range wired links following the principles of small-world graphs [10]. NoC-enabled multicores have been shown to be very beneficial for many computation-intensive bio-informatics applications, delivering orders of magnitude speedup and lower energy consumption [3][4], even compared with other popular acceleration platforms, such as GPUs and large FPGAs. A

This work was supported in part by the US National Science Foundation (NSF) grants CCF-0845504, CNS-1059289, and CCF-1162202, and Army Research Office grant W911NF-12-1-0373. PB acknowledges the support by US National Science Foundation (NSF) under Grant CCF-1331610. TM acknowledges support by Govt. of India DST SERB Grant No. SERB/F/1512/2014-15.

comprehensive survey [14] of various wireless NoC architectures and their design principles shows the possibilities of creating novel architectures enabled by on-chip wireless links.

Many of the biological processes we aim to control exhibit a highly dynamic spatio-temporal variation and heterogeneous structure imposing new constraints on the design methodologies [1]. A very relevant application in this domain pertains to the development of the gene therapy for HIV-1 [11], which has to tackle the dynamic variation in parameters of the biological interactions when searching for the best configuration and amount of engineered cells. Consequently, we explore the computation kernel that studies the interactions between cells and virions and enables the design of a gene therapy for HIV-1 infection, and consider it as a use-case to design and evaluate the potentials of a NoC-based multicore platform.

3. NOC DESIGN FOR BIOMOLECULAR REACTIONS

3.1 Modeling Stochastic Reactions

Reactants in biomolecular reactions, typically cells or protein macromolecules, interact with one another through stochastic processes that change the concentration of each reactant in the system. Since reactants are discrete molecules (or cells), the number of reactant molecules (or cells) completely describes the state of the system at any point in time. The dynamics of biomolecular reactants (say A, B, C) can be expressed through the mathematical formalism of stochastic simulations as follows:

$$A + B \underset{k}{\rightarrow} B + C \quad (1a)$$

$$B \underset{m}{\rightarrow} C \quad (1b)$$

$$C \underset{n}{\rightarrow} \phi \quad (1c)$$

A, B and C in the above reactions denote the number of biomolecular entities corresponding to reactants A, B and C respectively; k, m and n are the reaction rates. It has been shown that Gillespie's Stochastic Simulation Algorithm (GSSA) [5][6] is a very efficient method for modeling such coupled reactions. In the following, we exemplify our approach by considering as a use-case a set of reactions representing personalized gene therapy using protected T cells for treating HIV-1 infection, an application that is of great interest and significance to the research community.

3.2 Example: Biomolecular Reaction System Modeling HIV-1 Gene Therapy

Gene therapy, a promising alternative to the conventional highly active antiretroviral therapy (HAART) approach for treating HIV-1 infection, consists of infusion of engineered genes (i.e., gene based inhibitors [11]) into so-called protected cells (PC) with the goal of disrupting the viral spread within the body system. The design of a successful clinically relevant personalized gene therapy requires a comprehensive understanding of the interactions between the various biomolecular entities, which necessitates numerous stochastic simulations for exploring the functionality and efficacy of a particular strategy.

Consequently, we consider the well-known computational model describing the dynamics of the HIV's resistance to gene-modified cells [1][8][12] for *in vivo* infusion. This computational model consists of five biomolecular entities: (1) CD4+ T unprotected cells (U) denoting cells that are not genetically modified and naturally belong to the patient through homeostatic self-proliferation; (2) CD4+ T protected cells (P) referring to gene modified cells that are infused into the patient's body; (3) infected unprotected T cells (I_U); (4) infected protected T cells (I_P); and (5) free virions (V) that spread the viral infection throughout the T cell population. Following the conventions from Equation (1), the dynamics of these 5 entities can be described as follows:

$$\begin{aligned} &\phi \underset{\lambda}{\rightarrow} U, \quad U \underset{d}{\rightarrow} \phi, \quad U + V \underset{k}{\rightarrow} I_U, \quad I_U \underset{\delta}{\rightarrow} \phi \quad U \underset{d}{\rightarrow} \phi, \\ &P \underset{d}{\rightarrow} \phi, \quad U + P \underset{h+U+P}{\overset{r}{\rightarrow}} U, \quad U + P \underset{h+U+P}{\overset{r}{\rightarrow}} P, \\ &P + V \underset{\varepsilon k}{\rightarrow} I_P, \quad I_P \underset{\delta}{\rightarrow} \phi, \quad I_U \underset{\rho}{\rightarrow} V, \quad I_V \underset{\rho}{\rightarrow} V, \quad V \underset{c}{\rightarrow} \phi \end{aligned} \quad (2)$$

where λ is the thymus-rate of self-proliferation of U cells, d is the death rate, k and εk are the rates at which the virus infects an unprotected (U) and a protected (P) cell, respectively, δ is the death rate of both infected unprotected and protected cells, r and h are the Michaelis-Menten coefficients, ρ is the rate at which either infected unprotected cells or infected protected cells generate virions, and c is the death rate of the virions.

Each kind of biomolecular entity, i.e., U, P, I_U, I_P and V, has an initial concentration (or cell count), which varies over the course of time based on the Equation (2). We model the stochastic dynamics of the gene therapy based HIV treatment using GSSA first reaction method [5][6], based on which we decompose Equation (2) into a set of 11 "reaction channels" denoted by *Cj (j=1...11)* having propensities P_j as follows (given in the format *Cj: Pj*).

C1: λ *C2:* $rU/(h+U+P)$ *C3:* dU *C4:* kUV
C5: $rP/(h+U+P)$ *C6:* dP *C7:* εkPV *C8:* δI_U
C9: δI_P *C10:* $\rho(I_U+I_P)$ *C11:* cV (3)

3.3 Modeling Migration of Biological Entities

Each set of reaction channels described in Equation (2) represents the interaction among biomolecular entities within a specific reaction volume. A biophysical system is intrinsically heterogeneous and dynamic, implying that biological entities move (or migrate) in space over time as a result of interactions. We model this by the following approach.

A. We tessellate a biophysical system into a number of small regions, each of which independently carries out a system of reactions obeying Equation (2) and updates propensities (*update step*, Equation (3)).
B. Each such region R also interacts with *neighboring regions* (short-range communication) with one hop in 3D space and *non-neighboring regions* (long-range communication) with 2 to 4 hops in 3D space in the system to exchange a number of biological entities based on specific probabilistic rules. At the end of this step (which we call *migration step*), the number of cells of each biological entity in any region will get updated, leading to propensities (Equation (3)) being re-evaluated.

Hence, the biophysical system presents a case of several concurrent update steps (within different regions) and coupled migration steps (across regions). Both steps involve extensive computation and communication. In the actual system, each region has a fluid boundary. However, in our abstraction, we model these regions as cubes in three-dimensional space. Each cubic region has six neighbors in this model, and it exchanges

Table 1. Adjacency list representation of dependency graph

Vertex	Adjacency list	Vertex	Adjacency list
1	{2, 3, 4, 5}	7	{2, 5, 6, 9, 10}
2	{3, 4, 5}	8	{10}
3	{2, 4, 5}	9	{10}
4	{2, 3, 5, 8, 10}	10	{4, 7, 11}
5	{2, 6, 7}	11	{4, 7}
6	{2, 5, 7}		

Table 2. Computation load in each node of a region during one update step and a migration step

Node	ADD	MUL	DIV	RANDOM NUMBER	LOG	COMPARE
U0(1,2,4)	2	6	4	3	3	2
U1(3,5,6,7)	2	10	5	4	4	3
U2(8,9,10,11)	1	8	4	4	4	3
M	30	60	0	0	0	2

cells/molecules with them following a certain probability. We denote this kind of migration as *neighborhood communication*. However, there will be some exchange beyond the six immediate neighbors of each cubic region, albeit with progressively lower probabilities depending on the distance between the regions. We denote this kind of migration as *non-neighborhood communication*. It is to be noted that migration is a slower phenomenon compared to the update step. As such, in our model, the migration step occurs after every k update steps, or T/k denotes the migration step. For example, the migration step is $0.1T$ if it occurs after every 10 update steps.

3.4 Design of NoC-Based Multicore

3.4.1 Interaction among reaction channels

The interaction among the channels with propensities defined in Equation (3) can be modeled based on the dependency graph inferred from Equation (2). According to the GSSA first reaction method, each reaction channel computes a putative time τ_j of its own given by the following:

$\tau_j = -log(r) / P_j$, where r is a uniform random number, $0 \le r \le 1$ (4)

For every iteration, we denote the minimum value of τ_j as τ_j* and the corresponding channel as C_j*, which updates all its *dependent* neighbors in the dependency graph for that iteration. The data contained in these updates represent the new propensity values calculated in C_j*.

Pursuant to this discretized stochastic model of biomolecular reaction system with HIV-1 in the presence of gene therapy as a use-case, we create a (directed) dependency graph with 11 vertices corresponding to 11 reaction channels represented in adjacency list format in Table 1 where vertex j represents reaction channel C_j. Each vertex in this graph computes the new propensity function (Equation (3)) in every iteration. The edges of this graph represent the cumulative data exchange communication occurring from C_j*, for all j*.

The dependency graph of the reaction channels could provide the broad topological framework for inter-core data exchange. However, due to random numbers being generated at every step of the discretized reaction model, and the concomitant dependence of the update step described above, the traffic pattern subtends a unique topology in every step (basically, a subset of the dependency graph centered around Cj*). We carried out an extensive range of simulations with a variety of input conditions to cover a wide spectrum of reaction trajectories.

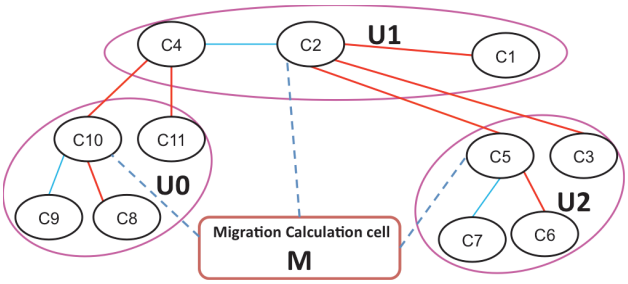

Figure 1. The solid lines connecting C_j vertices represent the MST derived from the weighted dependency graph (WDG). These vertices are then clustered into nodes *U0*, *U1* and *U2*, where the update step in computed. The fourth node *M* is the one where migration step is carried out. The dashed lines indicate the links between *M* and each *Ux*.

We construct an undirected weighted dependency graph (WDG) out of the large number of reaction simulations mentioned above (and further in Section 4.1). We achieve this by assigning edge weights to the directed graph in Table 1 based on the characterized communication volume over a wide range of reaction scenarios. The weights here correspond to the amount and frequency of data exchange that occurred for the adjacencies over the simulation period. We carry out topology inference from this WDG using a Minimum Spanning Tree approach based on Kruskal's algorithm [7]. The inferred topology is shown in Fig. 1.

3.4.2 Clustering of reaction channels

As explained earlier, there is extensive computation and communication during the update step in each of the regions, as well as during the migration step. This on-chip communication bandwidth requirement can be met only through a NoC. While the straightforward interaction between reaction channels presents an interesting case for mapping to a NoC, each reaction channel involves a rather small amount of computation to warrant an entire processing element. As such, we need to cluster the reaction channels (i.e. the tasks of propensity computation) into a few nodes. The nature of the dependency graph and the MST derived from it (cf. Section 3.4.1 and Fig. 1) and an analysis of the amount of computation (see Table 2) in each of the reaction channels led to clustering of the reaction channels within one region into three nodes (*U0*, *U1* and *U2*) following Algorithm 1 shown below. The result is shown in Fig. 1. The time complexity of this algorithm is $O(V \log V)$ where V is the number of vertices in the MST.

Algorithm 1

Initiate the MST with adjacency list structure TreeAL
Store all edges in a structure array TreeE
Store all nodes in the array ClusterR
Calculate the connection value for each edge stored in TreeE and store the value in TreeE
Find the edge Ex with maximum connection value in TreeE
While (true)
  Set the vertex on one side of the Ex with more connected edges to vertex TarV
  Set the vertex on one side of the Ex with less connected edges to vertex MoV
  If (the sum of number of nodes in TarV and MoV <= LimitN)
    Move all nodes in vertex MoV to TarV in array CLusterR.
    Mark Vertex MoV with invalid sign
    For each neighboring vertex NV connected to MoV
      Modify the edge between NV and MoV, set MoV to TarV
      Recalculate the connection value
    Find the new edge Ex with maximum connection value in TreeE
  Else Break
End While

Output all vertex in ClusterR without invalid mark.

The migration step also involves exchange of cells across regions, and we dedicate one node per region to undertake the computation involved for this step. This node, *M*, communicates with all other nodes in the same region, and the corresponding node in other (neighboring and non-neighboring) regions.

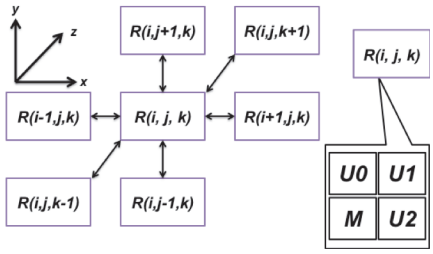

Figure 2. Orientation of semi-independent neighboring regions in 3-D space, and placement of computation nodes within a region.

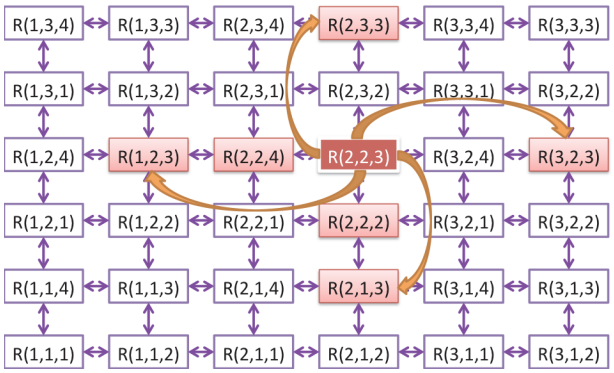

Figure 3. This diagram shows the global placement of a subset (36) of all the 64 regions we have modeled. Six neighbors of R(2,3,3) are lightly shaded. The straight arrows indicate direct wired connectivity. The curved arrows (just one set shown for clarity) indicate pass-transistor-enabled bypass connectivity through the switchbox.

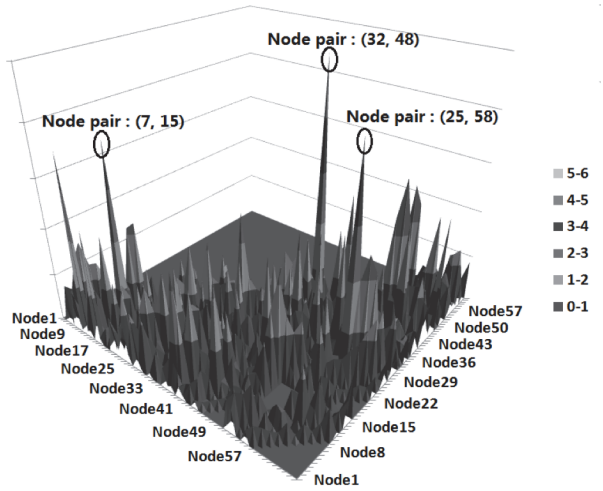

Figure 4. Plot showing intensity of *only* non-neighborhood communication among 64 regions. Note that the communication is symmetric; hence only one diagonal half of the matrix is plotted for clarity. The peak intensity pairs, indicated with circles, are the locations chosen for insertion of wireless links.

3.4.3 Communication during the migration step

A pair of neighboring and non-neighboring regions exchange a few cells of each type. More specifically, this exchange is carried out between the nodes handling the migration step across different regions. Since each region *R* is modeled as a cube in 3-D space (see Fig. 2), the migration step node in *R(i, j, k)*, i.e., *R(i, j, k).M* communicates with each of its neighbors, i.e. *R(i+1,j,k).M, R(i-1,j,k).M, R(i,j+1,k).M, R(i,j-1,k).M, R(i,j,k+1).M, R(i,j,k-1).M.* This communication happens deterministically after a fixed number of GSSA update steps. For non-neighboring nodes, communication occurs with a probability *P* that decreases with the Manhattan distance between the pair of nodes, which is given by the following:

$$P_A(x,y) = N_A * e^{-Dis3} \qquad (5)$$

where $P_A(x,y)$ is the probability of migration from a region centered at x to a non-neighboring region centered at y (in 3-D space) separated by a Manhattan distance *Dis3*, and N_A is the number of molecules/cells of reactant *A*.

3.4.4 Design of NoC architecture

In the following, we describe the proposed NoC design. Specifically, this constitutes the computation cores and the on-chip interconnection network.

3.4.4.1 Computation core and interconnection fabric

Each node, i.e., *U0*, *U1*, *U2* and *M*, is mapped to a computation core in the multicore platform. Hence, the total number of cores required is four times the number of regions modeled. Our model envisages a 4×4×4 subset of regions (independent reaction volumes) in 3-D space. Hence the multicore platform we design contains 256 cores catering to these 64 regions. The mapping of the regions to the cores is shown in Fig. 3. Computation loads in *Ux* nodes are roughly balanced but that in *M* node is much higher. However, the migration step usually takes place after a few update steps. Hence, the effect is not pronounced unless the migration frequency is high enough (cf. Section 4).

The placement of cores within one region is shown in Fig. 2, and the mapping and placement of regions is done in a tiled fashion as shown in Fig. 3. For nodes belonging to different regions, connectivity is required among the nodes of type *M* in each region (tile). Fig. 3 also shows the connectivity between tiles representing neighboring and next-to-neighbor regions, through direct wired links and pass-transistor-enabled bypass links [13].

For non-neighborhood communication, we follow the communication intensity given by Equation (5) and carry out extensive simulations to determine the interaction probabilities across 64 regions. The plot shown in Fig. 4 depicts the intensity of non-neighborhood (i.e. beyond next-to-neighbor) communication that results from the use-case application. The peaks in Fig. 4 represent the highest intensity of long-range communication.

3.4.4.2 Wireless long-range links

The interconnect topology in Fig. 3 is not equipped to support this long-range communication (cf. Fig. 4) without incurring a significant latency and energy penalty through multi-hop wired paths (cf. Section 4 for details). To overcome this challenge, we choose to implement these long-range links using wireless channels [14]. Implementing long-range links using different wireless technologies have been shown to deliver significant savings in latency and energy consumption [14][15][16]. In [14], it is demonstrated that it is possible to create three non-overlapping channels with on-chip mm-wave wireless links working in the 10-100 GHz range. Three different channels can be

designed with 3dB bandwidths of 16 GHz and center frequencies of 31, 57.5 and 120 GHz respectively, and each channel supports a data rate of 16 Gbps over a communication range of 20 mm. We add three wireless links corresponding to the three highest peaks in Fig. 4 to satisfy the most intensive long-range communication requirements.

3.4.5 On-chip communication

Within a region, nodes *U0*, *U1* and *U2* communicate among themselves during the update step. The new propensities at the end of this step are also communicated to node *M*. During the migration step, node *M* in each region communicates with the same node in each of its neighbors (and next-to-neighbors) the number of cells/molecules of each reactant in that region that migrate across the region boundary. This data exchange is bidirectional and takes place through wired links. Data exchange between (*M* nodes of) non-neighboring regions takes place through wired and wireless links. The wireless links (cf. Fig. 4) not only directly cater to the most intensive long-range communication, but they also act as shortcuts for (less intensive) long-range traffic between other source-destination region pairs. There is a marked improvement in latency and energy consumption of the NoC after the introduction of these wireless links, as we present in detail in the next section.

4. EXPERIMENTAL RESULTS

In this section, we present the detailed methodology for evaluating our proposed design and the obtained results.

4.1 Intra- and inter-region communication

As discussed in Section 3.4.1, we simulated over a wide range of initial cell counts (50, 100 and 500 for each cell type, i.e., *U*, *P*, I_U, I_P and *V*) and reaction parameters to determine the communication dependency graph. The reaction parameters that we experimented with are reported in Table 3 representing a realistic spectrum of infection and gene therapy treatment initiation scenarios [1]. We generated communication dependency graphs based on each scenario, and built a common subgraph (i.e. the WDG) and the subsequent MST as described in Section 3.4.1. The probability of inclusion of each edge of the inferred MST in each of the individual dependency graphs is very close to 1, indicating that there is an extremely high degree of confidence of the edges included in the final MST (cf. Fig. 1). Intra-region communication, i.e., among nodes *U0*, *U1*, *U2* and *M* of one region, follows directly from the original communication graph and the clustering result. Of note, in an ordinary mesh, communication between diagonal nodes (e.g., *M-U1*) takes 2 hops, while in our custom topology, the same takes 1 hop.

4.2 Performance Analysis

We use CMOS 65 nm technology node for our design and a system clock frequency of 1 GHz. Inter-core communication takes place with 16-bit wide flits. The wired links are adequately buffered so that the delays are less than one clock period (i.e. one hop). Energy dissipation of the network switches was obtained from the synthesized netlist by running Synopsys™ Prime Power, while the energy dissipated by wireline links was obtained through HSPICE simulations taking into consideration the length and layout of the buffered wireline links. The wireless links (including the transceiver circuitry) dissipate 1.95 pJ/bit. Wireless links can sustain a data rate of 16 Gbps over a 20 mm range of communication [14].

The key performance metrics that we evaluate are computation throughput and energy consumption. We present the results obtained on our custom-designed NoC topology with and without wireless links, and compare them with those obtained on a regular mesh.

4.2.1 Computation throughput

To measure the throughput of our design we report the number of GSSA update steps executed every 1 ms (or 1 million clock cycles). As mentioned earlier, the migration step has higher computation latency than the update step, and occurs after every few update steps. We show the sustained throughput achieved during our experiments over the run-time of the application when the migration step occurs after every 10, 5 and 2 update steps respectively in Fig. 5. As expected, the overall throughput metric shows a decline with increasing frequency of the migration step.

Table 3. Reaction parameters used in our simulations

Parameters	λ	r	h	d	k	ε	δ	ρ	c
Values	0.2	20	90	0.015	0.00008	0.01	0.5	120	6

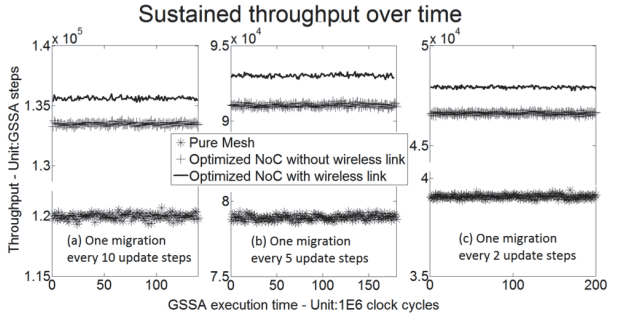

Figure 5. Sustained throughput delivered by our proposed NoC design with and without wireless links vis-à-vis pure mesh while simulating the use-case application with different migration step intervals. Each plotted point represents the number of GSSA update steps executed in 1 million clock cycles (i.e. 1 ms).

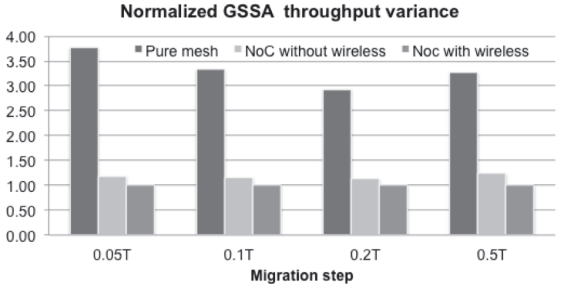

Figure 6. GSSA throughput variance normalized with respect to that on a NoC with wireless links for different migration steps

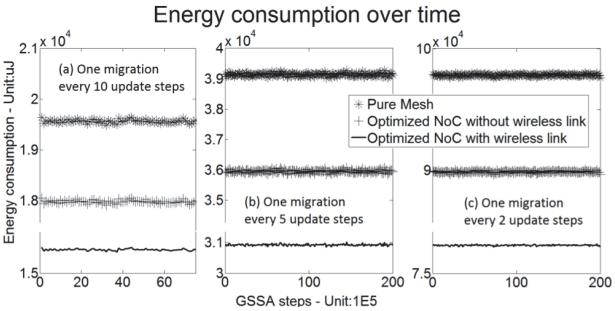

Figure 7. Energy consumed every 10^5 GSSA update steps by our proposed NoC design with and without wireless links vis-à-vis pure mesh while simulating the use-case application with different migration intervals.

The highest throughput of ~1.36E5 updates/ms is obtained on our NoC with wireless links for a migration step of *0.1T*. This represents more than 13% improvement over a pure mesh and a slight improvement over NoC without wireless links. This improvement goes up to 17% for migration step *0.2T* and 23% for migration step *0.5T*. This conclusively proves that our NoC indeed enables high intensity communication traffic, and that introduction of wireless links provides some throughput gain.

Another interesting observation is that there is some variation in throughput over the simulation time but this variation decreases with higher migration frequency. For the same migration frequency, NoC with wireless links also delivers the least variation in throughput over time. To further investigate this, we analyzed the variance of throughput for each migration step and topology. We normalize these variances with respect to the lowest value (on NoC with wireless) and present the results in Fig. 6. The low variances in throughput obtained on our NoCs (with and without wireless) also point to the fact that the network responds better in avoiding transient congestions and deadlocks.

4.2.2 Energy consumption

We define our energy consumption metric as the energy consumed over the time taken by 10^5 GSSA update steps. We plot the energy consumption over the simulation time in Fig. 7. As expected, increasing migration frequency leads to higher energy consumption, because as mentioned earlier the migration step is more computation- and communication-intensive than the update step. For migration step *0.1T*, the energy consumed every 10^5 update steps on our NoC with wireless is ~15 mJ. Note that this number incorporates the communication energy from 10^5 update steps and the same from 10^4 migration steps. For this migration frequency, NoC with wireless links consumes 23% less energy than a pure mesh, and ~17% less energy than the same NoC without wireless links. Although the overall energy consumption on each topology goes up with increasing migration frequency, our NoC with wireless always saves more than 20% energy compared to a pure mesh. The savings due to wireless links alone (vis-à-vis the optimized wired NoC topology) are more than 14% in each case. This proves that introduction of wireless links indeed makes the design significantly more energy-efficient.

5. CONCLUSIONS AND FUTURE WORK

In this paper, we present an efficient NoC architecture for biochemical analysis of cell physiology. We develop a novel spatio-temporal stochastic model of biological interactions occurring at the cellular level that account for biological heterogeneity. We comprehensively profile and analyze the computation and communication workloads. This enables us to develop an optimization algorithm for NoC design that balances the computation and communication loads. To reduce the communication energy consumption and node-to-node latencies, we propose an algorithm for application-specific long-range wireless link insertion. To prove the merits of our approach, we consider a realistic application, namely gene therapy, which aims to manage the HIV-1 infection and keep it under control (minimizing the HIV-1 virion population and slowing its mutational evolution). Our results show significant improvement in throughput and energy consumption over conventional mesh-based NoC implementations. This represents a first step towards realization of fast, efficient multicore platforms enabling the era of mobile health and personalized medicine.

REFERENCES

[1] S. Aviran, P.S. Shah, D.V. Schaffer, A.P. Arkin, “Computational Models of HIV-1 Resistance to Gene Therapy Elucidate Therapy Design Principles,” *PLoS Computational Biology,* vol. 6, issue 8, August 2010.

[2] L. Benini and G. D. Micheli, “Networks on Chips: A New SoC Paradigm,” *IEEE Computer,* Vol. 35, Issue 1, January 2002, pp. 70-78.

[3] T. Majumder, S. Sarkar, P.P. Pande and A. Kalyanaraman, "NoC-Based Hardware Accelerator for Breakpoint Phylogeny," *Computers, IEEE Transactions on* , vol.61, no.6, pp.857,869, June 2012.

[4] T. Majumder, M.E. Borgens, P.P. Pande and A. Kalyanaraman, "On-Chip Network-Enabled Multicore Platforms Targeting Maximum Likelihood Phylogeny Reconstruction," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on* , vol.31, no.7, pp.1061,1073, July 2012.

[5] M. A. Gibson and J. Bruck, “Efficient Exact Stochastic Simulation of Chemical Systems with Many Species and Many Channels,” *The Journal of Physical Chemistry A* **2000** *104* (9), 1876-1889.

[6] D. T. Gillespie, “A general method for numerically simulating the stochastic time evolution of coupled chemical reactions, “ *Journal of Computational Physics*, Volume 22, Issue 4, December 1976, Pages 403-434, ISSN 0021-9991.

[7] J.B. Kruskal, Jr., “On the Shortest Spanning Subtree of a Graph and the Traveling Salesman Problem,” *Proceedings of the American Mathematical Society*, Vol. 7, No. 1 (Feb., 1956), pp. 48-50.

[8] O. Lund, et al., “Gene Therapy of T Helper Cells in HIV Infection: Mathematical Model of the Criteria for Clinical Effect,” *Bull Math Biol.*, vol. 59, issue 4, pp. 725-45, July 1997.

[9] R. Marculescu, P. Bogdan, “The Chip Is the Network: Toward a Science of Network-on-Chip Design,” Foundations and Trends in Electronic Design Automation, vol. 2, no. 4, pp. 371-461, March 2009.

[10] U. Y. Ogras and R. Marculescu, “It’s a small world after all: NoC performance optimization via long-range link insertion,” *IEEE Trans. Very Large Scale Integr. Syst.*, vol. 14, no. 7, 2006, pp. 693-706.

[11] J.J. Rossi, C.H. June, and D.B. Kohn, “Genetic Therapies Against HIV,” *Nature Biotechnology*, vol. 25, pp. 1444-1454, 2007.

[12] G.F. Weber, “Gene Therapy – Why Can It Fail?”, *Medical Hypothesis,* vol. 80, pp. 613-616, 2013.

[13] S. Sarkar, G.R. Kulkarni, P.P. Pande, A. Kalyanaraman, "Network-on-Chip Hardware Accelerators for Biological Sequence Alignment," *Computers, IEEE Transactions on* , vol.59, no.1, pp.29,41, Jan. 2010.

[14] S. Deb, A. Ganguly, P.P. Pande, B. Belzer and D. Heo,"Wireless NoC as Interconnection Backbone for Multicore Chips: Promises and Challenges," *Emerging and Selected Topics in Circuits and Systems, IEEE Journal on* , vol.2, no.2, pp.228,239, June 2012.

[15] A. Ganguly, K. Chang, S. Deb, P.P. Pande, B. Belzer and C. Teuscher, "Scalable Hybrid Wireless Network-on-Chip Architectures for Multicore Systems," *Computers, IEEE Transactions on* , vol.60, no.10, pp.1485,1502, Oct. 2011.

[16] T. Majumder, P.P. Pande and A. Kalyanaraman, "Wireless NoC Platforms with Dynamic Task Allocation for Maximum Likelihood Phylogeny Reconstruction," *Design & Test, IEEE*, vol.31, no.3, pp. 54,64, June 2014.