Machine Learned Domain Decomposition Scheme Applied to Parallel Multi-scale Simulations

Journal Title
XX(X):1–11

DOI: 10.1177/ToBeAssigned
www.sagepub.com/
SAGE

Miloš Ivanović[1], Ana Kaplarević-Mališić[1], Boban Stojanović[1], Marina Svičević[1] and Srboljub Mijailovich[2]

Abstract

Since multi-scale models of muscles rely on the integration of physical and biochemical properties across multiple length and time scales, they are highly CPU consuming and memory intensive. Consequently, their practical implementation and usage in real-world applications is limited by high computational requirements. There are various reported solutions to the problem of parallel computation of various multi-scale models, but due to their inherent complexity, load balancing remains a challenging task. In this paper, we present a novel load balancing method for multiscale simulations. The method uses computatonally simple single-scale model in order to predict computational weights of the integration points within the complex multi-scale model. Employing obtained weights, it is possible to improve domain decomposition prior to the complex multi-scale simulation run and consequently reduce computation time. The method is applied to the two-scale muscle model, where finite element on macro scale is coupled with Huxley's model of cross-bridge kinetics on the micro scale. Our massive parallel solution is based on the static domain decomposition policy and operates in heterogeneous (CPU+GPU) environment. The approach was verified on the real-world example of the human tongue, showing high utilization of all processors and ensuring high scalability, thanks to the proposed load balancing scheme. The performance analysis shows that the inclusion of the prediction of the computational weights reduces execution time by about 40% compared to the run which uses trivial load balancer which assumes identical computational weights of all micro models. Proposed domain decomposition approach posseses a high capability to be applied in a variety of multi-scale models, not only in the field of bioengineering.

Keywords

Multi-scale modeling, parallel computing, load-balancing, muscle simulation, Huxley's muscle model

Introduction

Multi-scale models that are characterized by a range of spatial-temporal scales arise widely in many scientific domains. Despite the diversity in subject areas and terminology, there are many common challenges in multi-scale modelling, especially their real-world validation and high computational requirements Karabasov et al. (2014).

In the field of bioengineering, the models can be classified as phenomenological and biophysical. Phenomenological models predict the response to a specified input based on experimental measurements. Biophysically based models attempt to predict the tisssue response as emerging from the underlying physiology of the system and often take multi-scale approach. The large category of the multi-scale models uses an approach where reliable finite element (FE) method covers macro-scale behavior, while micro-scale molecular interaction model acts in each FE integration point in order to provide instantaneous macroscopic constitutive material characteristics. Ivanović et al. (2015) demonstrates mentioned approach on the skeletal muscle two-scale model.

However, greater accuracy of the multi-scale models comes at certain price. Their practical implementation and usage in real-world applications is limited by their high computational requirements. Feasible usage of these models could be reached only by employing massive

[1]Faculty of Science, University of Kragujevac, Kragujevac, Serbia
[2]Department of Chemistry and Chemical Biology, Northeastern University, Boston, USA

Corresponding author:
Miloš Ivanović, Faculty of Science, University of Kragujevac, R. Domanovića Str. 12, 34000 Kragujevac, Serbia.
Email: mivanovic@kg.ac.rs

parallellel techniques in high performance computing (HPC) environment, demonstrated by Chopard et al. (2011), Ivanović et al. (2015) and Heidlauf and Röhrle (2013).

Although these tools and frameworks provide significant acceleration, the utilization of the CPUs and GPUs is not perfect. The main reason lies in imperfect domain decomposition, due to the fact that most of the realistic models have complex geometries and inhomogeneous structure. Due to variations in structure and material properties coupled with different external conditions and loads, the computational weight varies significantly between the model parts and negativelly affects the balance of load. In the absence of any knowledge regarding these spatial and temporal imbalance, the domain decomposition algorithms usually consider all model parts equally computationally complex.

However, if one could obtain domain knowledge regarding dynamic behavior of each model part, it would be possible to assess their computational weights and employ this knowledge to determine the optimal domain decomposition. Although we cannot know the exact computational weights of the model parts prior to multi-scale simulation run, we claim that it is possible to estimate them using a simple machine learning technique, significantly improving domain decomposition and consequently speed-up.

To address this opportunity, in this paper, we propose a novel domain decomposition method which uses computationally undemanding phenomenological model (with identical setup as the complex multi-scale model) and machine learning in order to predict the computational weights of the integration points. Using our method consists of three distinct steps: *training step*, *prediction step* and *domain decomposition step*. In *training step*, we use real multi-scale data to train our ML model to predict computational weights of the integration points given the model state. We perform training only once. In the second, *prediction step*, we first run a fast single-scale phenomenologcal model with identical setup as the complex multi-scale model in order to obtain the estimate of the model state time history. Then, we employ trained ML model to predict computational weights of the integration points based on those states. In last, *domain decomposition step*, we use obtained computational weights in order to improve domain decomposition of the complex multi-scale model.

We evaluate entire machine learned domain decomposition approach on the parallel two-scale muscle model, based on Huxley's kinetics, the very same one as introduced by Ivanović et al. (2015). In our use case, single-scale phenomenological model run lasts more than 500 times faster then the multi-scale run, so we can safely neglect the overhead of the *prediction step*.

The rest of the article is organized as follows. Section *Two-scale muscle model* gives an introduction to two-scale model based on Huxley's muscle kinetics, together with simplistic phenomenological Hill's model used as a part of the predictor. Section *Parallelising the two-scale muscle model* gives an overview of the strategy used to parallelize computations of the micro models. Section *Predicting computational weights of the micro-models* contains details regarding developement of the innovative domain decomposition and its usage, while Section *Results and Discussion* presents the comparison between old and new domain decomposition technique on the real-world problem.

Two-scale muscle model

The aim of this section is to describe the components of two-scale model of muscle kinetics, which we use to test and evaluate our novel data decomposition technique.

Mechanical behavior of muscles is derived from the behavior of many individual components, such as cell membrane electrical conductivity and action potential, calcium dynamics, chemical reaction kinetics, and the actomyosin cycle, working together across spatial and temporal scales. Generally speaking, existing computational muscle models fall into two classes: (1) phenomenological, which evaluates the performance of the whole muscle and (2) biophysical, which investigates the ability of contractile proteins to generate force and movement at the cellular level. The most widely used phenomenological model, the Hill model described in Hill (1938); Zajac (1989); Kojic et al. (1998), only takes into account the relationship between active stress and strain rate, so its use is limited to isometric and steady state contractions. Thus, while practically useful, the Hill model is often inadequate for simulations of motor physiology. Most biophysical models evolve from the hypothesized cross-bridge kinetic concepts originally formulated by A.F. Huxley in Huxley (1957). Simulations of these kinetic processes, in the context of whole muscle models are tremendously computationally intensive and require simplifications of geometry, composition, and activation (Razumova et al. (1999)). These deficiencies in phenomenological and biophysical approaches invite the development of multi-scale models of muscle contraction that employ models of molecular interactions to calculate the instantaneous macroscopic constitutive material characteristics of muscle

necessary for quantitative models of whole muscle functional behavior.

A multi-scale muscle model provides a method to characterize deformation and force generation based on instantaneous material properties and component geometry. At the molecular scale, strain-dependent transitions between molecular contractile states define the instantaneous capacity of a muscle to generate active force and stiffness. This instantaneous muscle stiffness directly contributes to local constitutive relationships and thus the balance of forces. Instantaneous material attributes of the muscular tissue, derived from local actomyosin (molecular) interactions within muscle fiber and the material characteristics of the surrounding connective tissue, were prescribed in the FE integration points (Figure 1). Using this comprehensive methodology, the configuration at a given time was obtained from the displacement field, which is incrementally calculated in an iterative scheme by the equilibrium of internal forces, originating from muscle contraction and connective tissue elasticity, and external forces originating from the boundary and loading conditions. Macroscopic stresses were obtained from the active stress acting in the direction of muscle fibers and the components of elastic tensor representing connective tissue resistance to deformation.

Macroscopic model

From a mechanical point of view, a muscle can be considered as a mechanical system. The most common method for solving complex materially and/or geometrically nonlinear structural problems is the finite element method. In an incremental-iterative scheme, equilibrium configuration of a muscle can be calculated, considering the muscle as a structure composed of active fiber elements, able to contract under activation within the deformable connective tissue continuum as demonstrated by Kojic et al. (1998).

The governing equilibrium equation of a FE structure in deformed configuration at a time step (t) and iteration (i) is formulated as:

$$\begin{aligned}&\left({}^{t+\Delta t}\boldsymbol{K}_{el} + {}^{t+\Delta t}\boldsymbol{K}_{mol}\right)^{(i-1)}\delta\boldsymbol{U}^{(i)} = \\ &{}^{t+\Delta t}\boldsymbol{F}_{ext}^{(i-1)} + {}^{t+\Delta t}\boldsymbol{F}_{int}^{(i-1)} + {}^{t+\Delta t}\boldsymbol{F}_{activ}^{(i-1)}\end{aligned} \tag{1}$$

where ${}^{t+\Delta t}\boldsymbol{F}_{ext}^{(i-1)}$, ${}^{t+\Delta t}\boldsymbol{F}_{int}^{(i-1)}$ and ${}^{t+\Delta t}\boldsymbol{F}_{activ}^{(i-1)}$ are vectors of external physiological loads, internal (structural) nodal forces, and integrated active molecular forces lumped into FE nodal forces, respectively; ${}^{t+\Delta t}\boldsymbol{K}_{el}$ and ${}^{t+\Delta t}\boldsymbol{K}_{mol}$ are stiffness matrices of the passive components of constitutive FE and of cumulative stiffness of actomyosin bonds, respectively; $\delta\boldsymbol{U}^{(i)}$ are the increments of nodal displacements at iteration (i) and the left-upper index $t+\Delta t$ indicates that the equilibrium equations correspond to the end of the time step. The key step in a standard FE formulation is the evaluation of the element nodal internal and active forces:

$$\begin{aligned}&{}^{t+\Delta t}\boldsymbol{F}_{int}^{(i-1)} + {}^{t+\Delta t}\boldsymbol{F}_{activ}^{(i-1)} = \\ &\int_{t+\Delta t V^{(i-1)}}^{n+1} {}^{t+\Delta t}\boldsymbol{B}_L^{T\,(i-1)} \quad {}^{t+\Delta t}\boldsymbol{\sigma}^{(i-1)}\, dV,\end{aligned} \tag{2}$$

where ${}^{t+\Delta t}\boldsymbol{B}_L^{T\,(i-1)}$ is the geometric linear strain-displacement matrix (superscript T means transpose), ${}^{t+\Delta t}\boldsymbol{\sigma}^{(i-1)}$ is the 2^{nd} Piola-Kirchhoff stress tensor within the muscle, and ${}^{t+\Delta t}\boldsymbol{V}^{(i-1)}$ is the volume of a FE. The contribution of variable stiffness of actomyosin bonds, ${}^{t+\Delta t}\boldsymbol{K}_{mol}^{(i-1)}$, and of the passive component representing connective tissue, ${}^{t+\Delta t}\boldsymbol{K}_{el}^{(i-1)}$, define the material resistance to deformation across spatial scales and can be calculated as:

$$\begin{aligned}&{}^{t+\Delta t}\boldsymbol{K}_{mol}^{(i-1)} + {}^{t+\Delta t}\boldsymbol{K}_{el}^{(i-1)} = \\ &\int_{t+\Delta t V^{(i-1)}}^{n+1} ({}^{t+\Delta t}\boldsymbol{B}_L^{T} \quad {}^{t+\Delta t}\boldsymbol{C} \quad {}^{t+\Delta t}\boldsymbol{B}_L)^{(i-1)}\, dV + \\ &\int_{t+\Delta t V^{(i-1)}}^{n+1} ({}^{t+\Delta t}\boldsymbol{B}_{NL}^{T} \quad {}^{t+\Delta t}\boldsymbol{S} \quad {}^{t+\Delta t}\boldsymbol{B}_{NL})^{(i-1)}\, dV,\end{aligned} \tag{3}$$

where $\boldsymbol{C}$ is is the constitutive matrix representing the stressstrain relationship, ${}^{t+\Delta t}\boldsymbol{B}_{NL}$ is nonlinear strain-displacement transformation matrics, ${}^{t+\Delta t}\boldsymbol{S}$ is matrix of 2^{nd} Piola-Kirchhoff stresses.

After assembling the element balance equations, the FE equilibrium equations for the entire muscle are solved, securing the equilibrium of $\boldsymbol{F}_{ext}$, $\boldsymbol{F}_{int}$ and $\boldsymbol{F}_{active}$ within the prescribed tolerance at the end of each time step, as shown by Bathe (1982), Bathe and Kojic (2005). The displacement vector $\boldsymbol{U}^i$ is updated during iterations by the current increment $\delta\boldsymbol{U}^{(i)}$ until $\delta\boldsymbol{U}^{(i)} \approx 0$ at the convergence. Active force generation, ${}^{t+\Delta t}\boldsymbol{F}_{activ}^{(i-1)}$, and stiffness, ${}^{t+\Delta t}\boldsymbol{K}_{mol}^{(i-1)}$, are directly dependent on the rate of muscle deformation in the principal direction of muscle fibers (Kojic et al. (1998)).

Evaluation of nodal internal and active forces in Equation (2) demands determination of stresses in muscle fibers corresponding primarily to their stretch rates. This can be performed by employing a simplistic phenomenological model like Hill's, or more precise and computationally complex physiological model such as Huxley's.

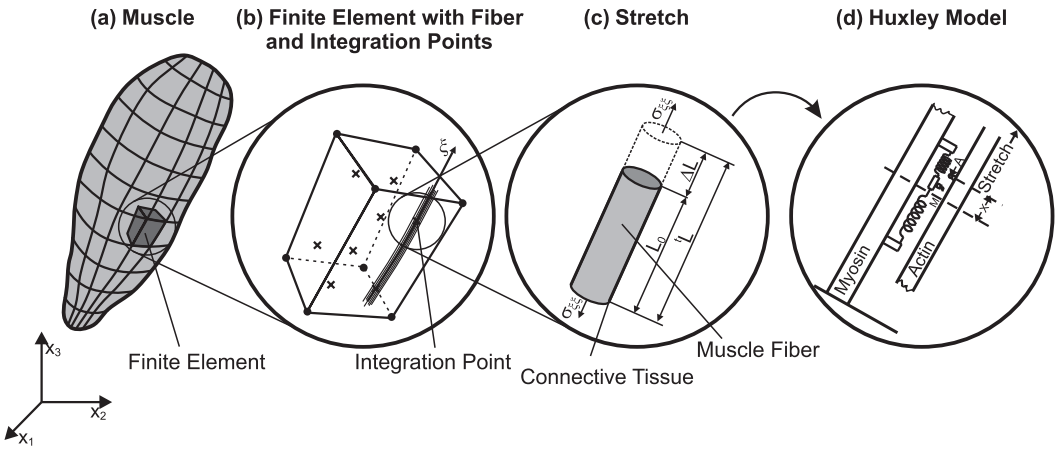

Figure 1. Multi-scale model of muscle contraction: (a) muscle FE discretization; (b) diagram depicting the muscle fibers contained within a characteristic tree-dimensional FE, including denoted FE integration points and the principal direction of muscle fibers, ξ; (c) elongation of an individual muscle fiber, ΔL, at the indicated spatial scale and under stress $\sigma_{\xi\xi}$; ΔL is calculated from current length, ${}^{t}L$, and slack (relaxed) length L_0; the current time is denoted as t; (d) Huxley's cross-bridge kinetics model.

Hill's phenomenological model

Hill's equation (Hill (1938)) is derived from the quick-release experiments on a muscle in tetanized condition, which can be written in dimensionless form as

$$\frac{S}{S_0} = \frac{1 - v/v_0}{1 + cv/v_0} \tag{4}$$

where S represents tension in muscle, v is the velocity of the contraction, v_0 is maximum velocity, S_0 is maximum tension in tetanized condition and c is constant. Muscle composite structure consisting of different fiber types can be presented as series of contractile and non-linear serial elements, representing an active part of a muscle, coupled in parallel to the linear elastic element representing the connective tissue (passive part), following Stojanovic et al. (2007). Contractile element behavior is described by Equation (4) and curve established by Gordon et al. (1966). The tension-stretch relationship for the non-linear elastic element is given by:

$$S = (S^* + \beta)e^{\alpha(\eta - \eta^*)} - \beta \tag{5}$$

where S^* represents the tension corresponding to a stretch η^* while α and β are the material constants. From the condition that the stresses in contractile and serial elements are equal, we can obtain the active muscle stress σ_m at the material point.

The total stress σ is expressed as the contribution of active muscle forces and the contribution of (passive) elasticity of collagenous connective tissue, cell membrane, and muscle noncontractile cytoskeleton in parallel to muscle cells:

$$\boldsymbol{\sigma} = \boldsymbol{\sigma}_m \phi + (\boldsymbol{\sigma}^E + b \cdot \dot{\boldsymbol{e}}), \tag{6}$$

where ϕ is the fraction of muscle fibers in the total muscle volume and σ^E is the stress in passive part of the muscle, $b \cdot \dot{\boldsymbol{e}}$ is a damper element that provides resistance in mechanical motion and b is a damping factor. The member of the tangent constitutive matrix in the fiber direction can be calculated as

$$\bar{C}_{11} = \phi \frac{\partial \sigma_m}{\partial \bar{e}_{11}} + (\bar{C}^E_{11} + b \cdot \frac{\dot{e}_{11}}{\partial \bar{e}_{11}}), \tag{7}$$

where $\bar{C}^E_{11}$ is the element of elastic tangent constitutive matrix of connective tissue in the fiber direction.

Huxley's microscopic model

Another way to obtain stress in Eq. (2) follows Huxley (1957), where a muscle fiber constitutive unit is represented by interacting actin and myosin filaments. Elastic spring like connections between these filaments, formed via so-called cross-bridges, generate in aggregate the active muscle force and stiffness proposed by Torelli (1997). Depending on boundary conditions, over time the filaments can slide relative to each other so cross-bridges can experience both tension and compression. Following these rules, McMahon (1984) proposed Huxley's sliding filament theory defined by the following partial differential equation defined over the domain Ω:

$$\frac{\partial n}{\partial t}(x,t) - v\frac{\partial n}{\partial x}(x,t) = \mathcal{N}(n(x,t),x), \tag{8}$$

where n is fraction of attached cross-bridges displaced for x from its strain free position at time t, $v = dx/dt$ is the shortening velocity of the thin filament with respect to the thick filament, $\mathcal{N}(n(x,t),x) = [1 - n(x,t)]\, f(x) -$

$n(x,t)g(x)$ is compounded transition flux between attached and detached states where $f(x)$ and $g(x)$ are attachment and detachment rates, strictly dependent on the distance x. For solving the first order hyperbolic Equation (8), we used method of characteristics following Lister (1960); Mijailovich et al. (1996).

The specific muscle tension arises from the distortion of the cross-bridge represented as a linear spring and can be calculated as $\mathcal{F}(t) = \kappa \cdot \int_{-\infty}^{\infty} x \cdot n(x,t)dx$, where κ is the cross-bridge stiffness. Instantaneous specific muscle stiffness is then defined by integral $\mathcal{K}(t) = \kappa \cdot \int_{-\infty}^{\infty} n(x,t)dx$. The active stress generated in muscle is calculated as

$$\sigma_m = \mathcal{F}\frac{\sigma_{iso}}{\mathcal{F}_{iso}}, \tag{9}$$

where σ_{iso} is a maximal isometric stress and $\mathcal{F}_{iso}$ is a specific maximal force calculated by Huxley's model for isometric contraction $(v = 0)$.

Determining the total stress σ and constitutive equation can be performed in the same manner as it has been done in case of Hill's model, using Equation (6) and (7). The computational difference between Hill's and Huxley's model is significant and counts to two to three orders of magnitude. While the phenomenological model represents constitutive law by a single equation, Huxley's model requires the entire molecular simulation to be conducted and PDE Equation (8) solved using iterative method of characteristics.

Parallelizing the two-scale muscle model

The starting point in the applied parallelization strategy is dividing each iteration, during incremental two-scale model simulation, into two distinct sets of algorithm steps, following Ivanović et al. (2015). The first set considers the finite element calculations such as assembling the element balance equations and solving the FE equilibrium equations for the entire muscle model. The second set considers micro-scale models of muscle fibers for each integration point within the FE mesh. As stated by Ivanović et al. (2015), our solution applies parallelization only to the second set, i.e. to micro-scale model calculations, while the FE algorithm runs in sequential manner. Parallel run assumes a set of MPI processes, with a basic idea being to statically decompose the entire set of micro models into smaller chunks. One of the processes, having the role of a manager, performs FE calculations. When the algorithm reaches the point where micro model computations should start, each process performs simulations over the micro models from its own domain of responsibility. Using this approach, we reduce the process intercommunication to only two operations: *scattering of the deformations* and *gathering of the stresses together with stress derivatives*, Figure 2.

The static domain decomposer acts only at the beginning of the simulation by adjusting workload distribution to the number of available resources. The previous implementation demonstrated by Ivanović et al. (2015) assumes identical computational weights of all micro-models, which is not true, since muscle structure and the activation is inhomogeneous. In a real tissue, each micro-model assigned to an integration point has its own material characteristics. Some of these points respond purely elastically, while the others also contain muscle tissue of certain characteristics. In addition, different parts of the muscle model undergo varying degrees of fiber activation.

Consequently, the uniform domain decomposition based only on determination of the size of micro model chunks cannot result in the balanced workload. Assessing computational weight of each micro model would largely help improving data decomposition and reaching significantly better load balance. In the next section, we present the method for predicting the computational weight of the integration points using domain knowledge obtained from the simpler Hill's single-scale phenonmenological model.

Predicting computational weights of the micro-models

We base our predictor on the premise that it's possible to obtain approximate number of iterations of each micro-model (computational weight) from the time history of model states given in strain rate, stress and degree of muscle fiber activation. Equipped with such relation, it is feasible to employ simpler Hill's phenomenological model with identical setup as the complex two-scale model, in order to provide estimations of time histories for each integration point state. The final output of the proposed scheme is the assesment of the total number of Huxley's micro model iterations for each integration point. The simplified predictor scheme is given in Figure 3.

In the absence of any analytical relation, we employ a machine learning technique in order to obtain approximate number of Huxley's micro model iterations out of micro-model states. As shown in Figure 3, the predictor training occurs only once, using input data obtained from real two-scale simulation. On the contrary, it is necessary to perform a single-scale simulation run prior to each two-scale simulation run, in order to obtain integration points state

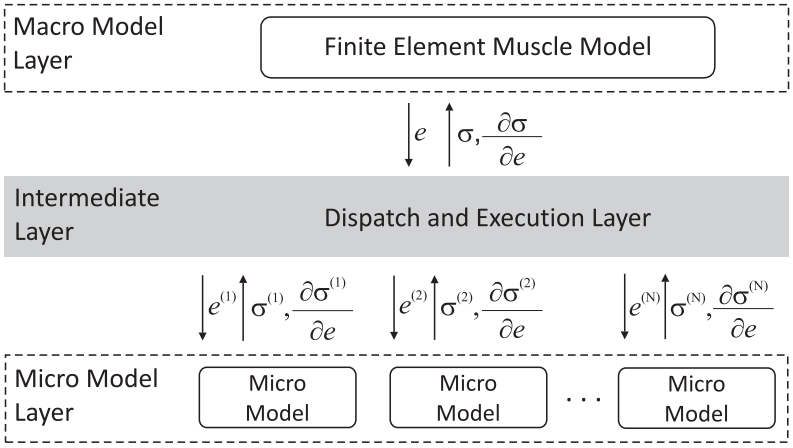

Figure 2. The proposed system embodies the interface between a macro-scale layer, including the FE model and its implementation, a micro-scale layer, including the material model that determines muscle behavior at the integration points, and an intermediate layer, which acts to mediate the macro- and micro-scale functions.

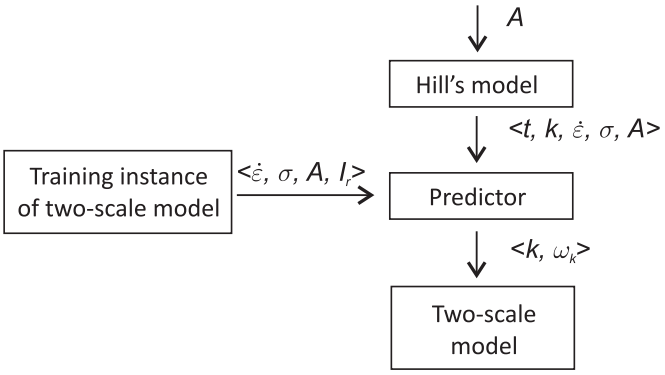

Figure 3. Predictor of the micro-model weights. We train the predictor using records that contain the strain rate $\dot{\epsilon}$, stress σ, activation A and the number of the micro-model iterations I_r (class variable). Hill's single scale model produces the estimate of time history states for each micro-model k and each time step t. Using this data, the predictor produces the computational weighs w_k for each micro-model k.

history. Finally, the predictor outputs the estimation of each micro model computational weight, which is then employed during the domain decomposition phase.

The additional step is not significantly burdensome for the entire simulation process, since the duration of single-scale simulation run using Hill's model is over 500 times shorter compared to the complex two-scale model with identical setup.

Predictor training

The dynamic parameters that change throughout the muscle simulation and dictate the behavior of the micro models states are strain rate, stress in previous time step, and degree of activation degree of the muscle fibers. The initial multivariate analysis shows that these parameters have significant impact on the micro model weight, quantified as the number of iterations of the micro model (Huxley's) within a single iteration of the macro (finite element) model.

Among various ML techniques, we chose simple *K*-nearest neighbor method (KNN). In our case, KNN turned out to be more efficient and flexible than ANN (Artificial Neural Networks). Complete set of training data was obtained from the real two-scale model and consisted of 1165056 records in the form

$$(\dot{\epsilon}, \sigma, A, I_r),$$

where $\dot{\epsilon}$ denotes the strain rate, σ denotes stress, A activation, and I_r number of the micro-model iterations to converge. We divided data into training test (75% samples) and test set (25% samples) and tested for optimal number of neighbors to be taken. We express the error as

$$Err[\%] = \frac{\sum_{i=1}^{n} \left| I_r^i - I_{knn}^i \right|}{\sum_{i=1}^{n} I_r^i} \cdot 100, \qquad (10)$$

where n represents the test set length, I_r^i the number of iterations from i-th sample, and I_{knn}^i predicted number of iterations.

As shown in Table 1, the estimation on the test set works well for any number of neighbors, with all errors below 1%. We set the number of neighbors to 3 for further benchmarks.

Table 1. Estimation error with various number of neighbors in KNN

Number of neighbors (k)	Error (%)
2	0.530
3	0.528
4	0.580
5	0.608
10	0.795
15	0.930

Another confirmation of the validity of our KNN model comes from the Table 2. The algorithm estimates the majority of test set correctly. Even when wrong, it's for a relatively small number of iterations compared to the average of 28 iterations.

Table 2. Error in number of micro-model iterations. Average number of iterations is 28.

Error (*iterations*)	Frequency (%)
<2	97.50
2	0.97
3	0.48
4	0.30
5	0.20
>5	0.03

Predicting and decomposing

As shown in Figure 3, the predictor is executed prior to each two-scale simulation run. The major assumption in this phase is that Hill's single-scale model outputs similar mechanical states $(\dot{\epsilon}, \sigma, A)$ compared to the complex two-scale model with the identical setup. The time history of the integration points states predicted by the single-scale Hill's model approximate corresponding states in two-scale model sufficiently well, as shown in Figure 4. In 99.3% of the cases, the strain rate given by Hill's model deviates less than 10% from the strain rate given by the complex two-scale model. Similar applies to the value of stress. Of course, the activation is identical in both models, since it acts as an input variable. The final output of the predictor is predicted number of the iterations w_k of each micro-model k:

$$w_k = \sum_{i=1}^{n_t} I_{knn}^k(i), \qquad k = \overline{1,n}, \tag{11}$$

where n_t designates the number of macro-model simulation steps.

In the following two subsections we present two decomposition cases (1) simple case with **homogenious computing environment** (only CPUs), and (2) more complicated **heterogeneous computing environment** (CPUs+GPUs).

Homogeneous computing environment. In a pure CPU computing environment, we determine the average number of iterations that should be carried out by each CPU, based on the predicted total number of iterations and the number of the CPUs as:

$$I_{proc} = \frac{1}{p}\sum_{k=1}^{n} w_k, \tag{12}$$

where n is the number of all micro-models and p is the CPU count. The domain decomposition is performed by dividing micro models into chunks, starting from the most weighted down to the least weigted micro model, keeping the size of chunks under the limit of the total number of iterations per chunk, I_{proc}. Finally, the algorithm allocates the remaining, yet unallocated micro models, using simple round robin rule. Surely, described algorithm does not provide fully optimal task allocation in all possible cases, but is sufficient for the verification of the efficiency of the proposed predictive load-balancer.

Heterogeneous computing environment. Decomposing in the heterogeneous environment (CPUs+GPUs) is a bit more complex. The scheme which we present here is a slightly modified technique given by Ivanović et al. (2015). Instead of treating all micro-models equiponderant, we take into account the predictor output designated as w_k, $\forall k = \overline{1,n}$. Besides predictor output, the algorithm requires the following input data for each MPI process: v_j - ideal process speed (in number of micro-model iterations per second) and m_j - the memory limit of j-th process representing the number of micro-models that can be stored within the device memory. The algorithm is given in Figure 5.

The first step in the iterative scheme is to obtain the idealistic execution time $T^{(0)}$ (when $\forall m_j \to \infty$):

$$T^{(0)} = \frac{\sum_{k=1}^{n} w_k}{\sum_{j=1}^{p} v_j}. \tag{13}$$

Then, for each process, we determine the chunk size (number of micro models), that the process with speed $V_j^{(0)} \equiv v_j$ is capable to complete in time $T^{(0)}$. The expected number of the micro-model iterations a process j should perform is then

$$L_j^{(0)} = T^{(0)} \cdot V_j^{(0)}, \qquad j = \overline{1,p}. \tag{14}$$

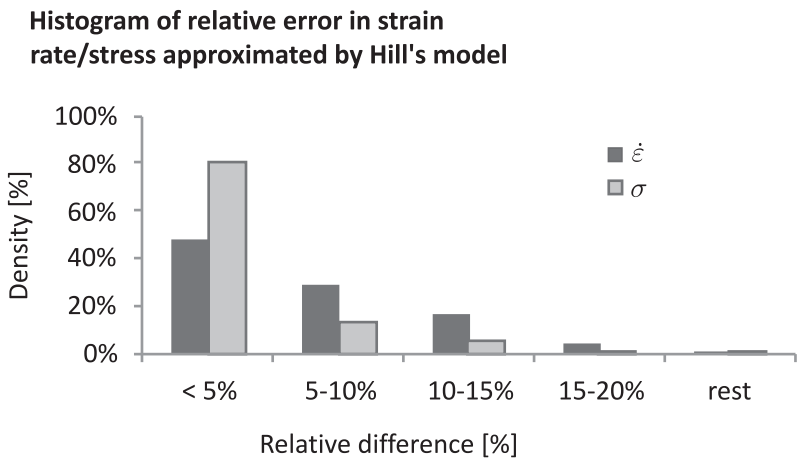

Figure 4. Histogram of relative difference between two-scale model and Hill's model in terms of strain rate $\dot{\epsilon}$ and stress σ

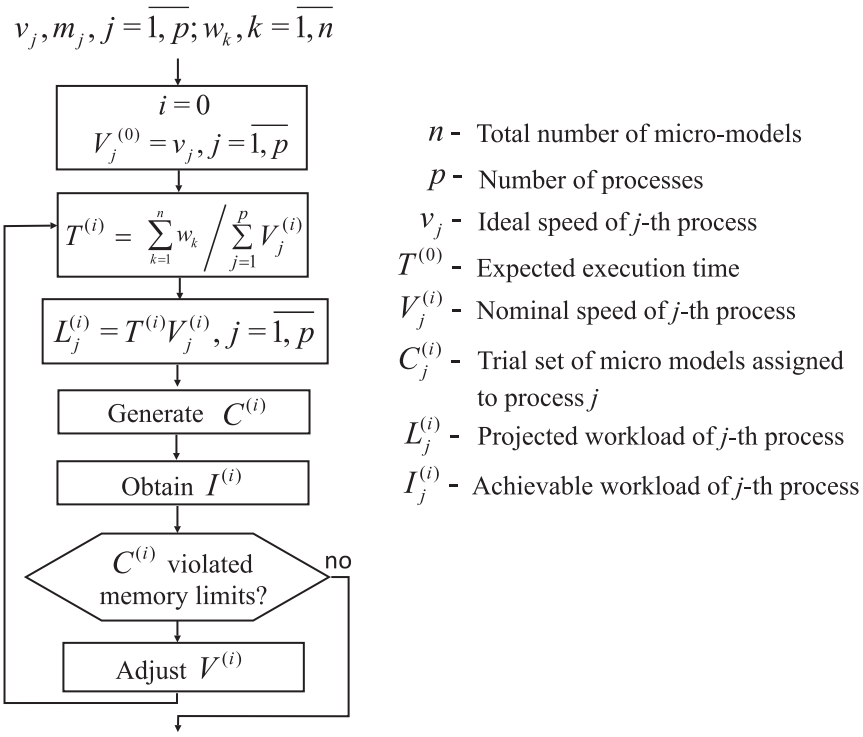

Figure 5. Decomposition algorithm used in heterogeneous computing environment (CPUs+GPUs)

Now we create a **trial distribution** of the micro-models $C_j^{(0)}$, $\forall j = \overline{1,p}$, trying to reach projected $L_j^{(0)}$ iteratoins.

If any process j (usually those tied to GPUs) is incapable to store $\left|C_j^{(0)}\right|$ micro-models without reaching its own memory limit m_j, then the process j is incapable to perform calculations with its nominal speed $V_j^{(0)}$. Consequently, the real workload of the process j should be reduced to $I_j^{(0)} \leq L_j^{(0)}$ micro-model iterations. The corrected process speed distribution is then:

$$V_j^{(1)} = \frac{I_j^{(0)}}{T^{(0)}}, \qquad j = \overline{1,p}. \tag{15}$$

According to the values $V_j^{(1)}$, $j = \overline{1,p}$, we estimate the new synchronization time $T^{(1)}$. The algorithm repeats iteratively until all projected $L_j^{(i)}$ are satisfied without violating prescribed memory limits, namely $L_j^{(i)} = I_j^{(i)}, \forall j = \overline{1,p}$.

Results and Discussion

We conducted the performance analysis of the real-world model of the complex muscle structure of a tongue. The multi-scale tongue model setup is identical to that employed previously by Ivanović et al. (2015). The model consists of 873 finite elements, with 3492 integration points. We modeled lingual shape changes occurring following lingual

tip contact with the hard palate during swallowing, with the total duration of 0.5 seconds divided into 50 FE (macro model) time steps. As a hardware platform, we employed a cluster consisted of 22 nodes, each of them equipped with dual Intel Xeon E5-2670@2.6GHz 8-core CPU and 64GB memory.

Figure 6 shows the normalized value of the total number of iterations on the integration points during 0.5s, estimated by our predictor compared to the values obtained from the real two-scale simulation.

Figure 7 shows the comparison of the total number of Huxley iterations carried out by each of 128 processes using proposed predictor compared to the equiponderant case. It is obvious that using data produced by the predictor in determining the distribution of tasks among CPUs leads to a significantly more balanced workload.

Consequently, incorporating our predictor within domain decomposition policy results in reduced execution time by about 40% compared to the same model carried out using old equiponderant domain decomposition policy. For the sake of comparison, the sequential execution of the simulation took approximately 65.5 h, while 256 processes completed the same task in only 20 minutes.

Figure 8 shows the results of the speed-up benchmark which oposes domain decomposer employing proposed predictor and the old using equiponderant scheme. Each speed-up value is obtained as an average of total execution times taken from ten simulation runs.

Conclusion

We presented a novel machine learned domain decomposition scheme for multi-scale simulations in the parallel computing environment. The scheme uses computatonally simple single-scale model in order to predict computational weights of the integration points within the complex multi-scale model, thus trying to improve load balance of the parallel run.

Our benchmark case employs data obtained from simple Hill's phenomenological model in order to predict computational weights of the integration points within the multi-scale model. Massive parallel solution, based on decomposition of micro model domain and static domain decomposition policy was verified on the realistic example of lingual shape changes occurring following lingual tip contact with the hard palate during swallowing. The novel domain decomposition scheme ensures high utilization of all CPUs and reduction of runtime by more than 40% compared to the same model carried out using decomposer assuming equal weights of all micro-models bound to the integration points.

Achieved performance boost paves the path for introducing such models into clinical practice with a reasonable price-performance ratio, but also opens the door for new scientific discoveries in the field of biomedicine.

Acknowledgements

Part of this research is supported by The Ministry of Science in Serbia, Grants III41007, OI174028, III44010, and TR14005.

References

Bathe K and Kojic M (2005) *Inelastic analysis of solids and structures*. Springer.

Bathe KJ (1982) *Finite element procedures in engineering analysis*. Prentice-Hall.

Chopard B, Falcone JL, Hoekstra AG, Borgdorff J et al. (2011) A framework for multiscale and multiscience modeling and numerical simulations. In: *UC*. Springer, pp. 2–8.

Gordon A, Huxley AF and Julian F (1966) The variation in isometric tension with sarcomere length in vertebrate muscle fibres. *The Journal of physiology* 184(1): 170.

Heidlauf T and Röhrle O (2013) Modeling the chemoelectromechanical behavior of skeletal muscle using the parallel open-source software library opencmiss. *Computational and mathematical methods in medicine* 2013.

Hill A (1938) The heat of shortening and the dynamic constants of muscle. *Proceedings of the Royal Society of London B: Biological Sciences* 126(843): 136–195.

Huxley AF (1957) Muscle structure and theories of contraction. *Prog. Biophys. Biophys. Chem* 7: 255–318.

Ivanović M, Stojanović B, Kaplarević-Mališić A, Gilbert R and Mijailovich S (2015) Distributed multi-scale muscle simulation in a hybrid mpi–cuda computational environment. *simulation* : 0037549715620299.

Karabasov S, Nerukh D, Hoekstra A, Chopard B and Coveney PV (2014) Multiscale modelling: approaches and challenges.

Kojic M, Mijailovic S and Zdravkovic N (1998) Modelling of muscle behaviour by the finite element method using hill's three-element model. *International journal for numerical methods in engineering* 43(5): 941–953.

Lister M (1960) The numerical solution of hyperbolic partial differential equations by the method of characteristics. *Mathematical methods for digital computers* 1: 165–179.

McMahon TA (1984) *Muscles, reflexes, and locomotion*. Princeton University Press.

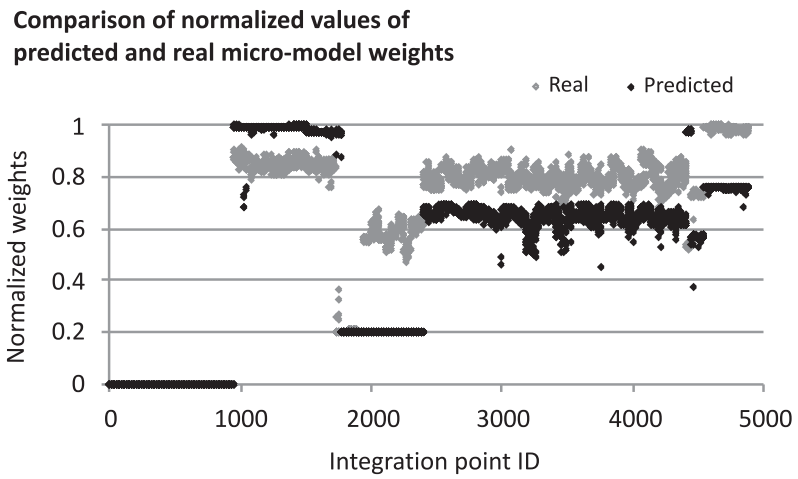

Figure 6. Predicted vs. real computational weights of the micro models bound to the integration points

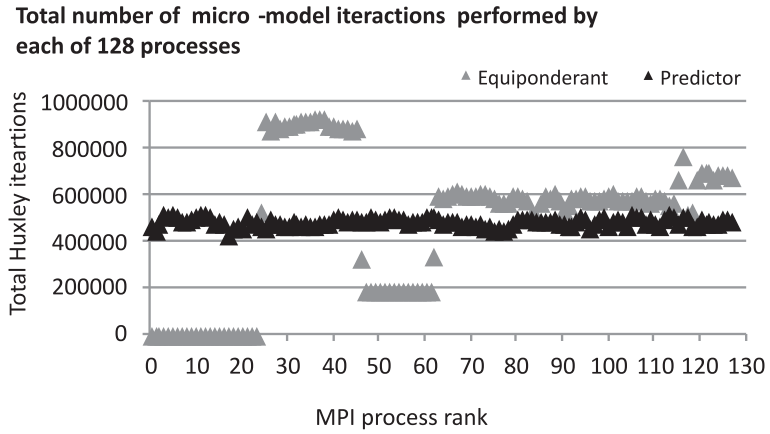

Figure 7. Total number of micro-model iterations executed by each MPI process throughout the simulation run

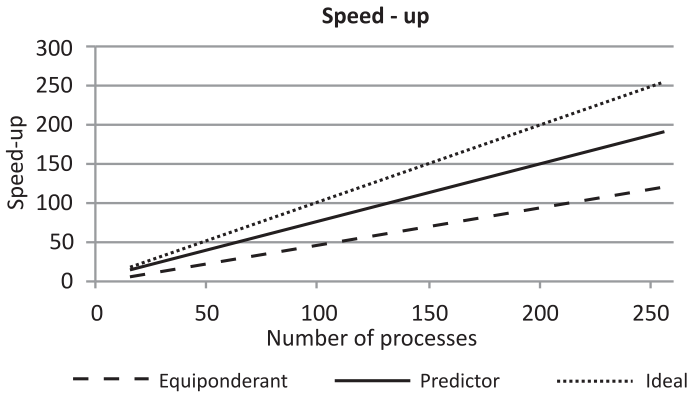

Figure 8. Speed-up obtained using (1) domain decomposer assuming all micro models equiponderant and (2) decomposer employing proposed predictor. Each speed-up value is obtained as an average of total execution time taken from 10 subsequent runs.

Mijailovich SM, Fredberg JJ and Butler JP (1996) On the theory of muscle contraction: filament extensibility and the development of isometric force and stiffness. *Biophysical Journal* 71(3): 1475–1484.

Razumova MV, Bukatina AE and Campbell KB (1999) Stiffness-distortion sarcomere model for muscle simulation. *Journal of Applied Physiology* 87(5): 1861–1876.

Stojanovic B, Kojic M, Rosic M, Tsui C and Tang C (2007) An extension of hill's three-component model to include different fibre types in finite element modelling of muscle. *International journal for numerical methods in engineering* 71(7): 801–817.

Torelli A (1997) Study of a mathematical model for muscle contraction with deformable elements. *Rend. Semin. Mat.(Torino)* 55: 241–271.

Zajac FE (1989) Muscle and tendon properties models scaling and application to biomechanics and motor. *Critical reviews in biomedical engineering* 17(4): 359–411.