УНИВЕРЗИТЕТ У КРАГУЈЕВЦУ
ПРИРОДНО-МАТЕМАТИЧКИ ФАКУЛТЕТ

мр Ана Капларевић-Малишић

Развој и анализа метода паралелизације вишескалних модела мишића

докторска дисертација

Крагујевац, 2016.

I. Аутор
Име и презиме: Ана Капларевић-Малишић
Датум и место рођења: 21.01.1974. Крагујевац
Садашње запослење: асистент на Природно-математичком факултету Универзитета у Крагујевцу
II. Докторска дисертација
Наслов: Развој и анализа метода паралелизације вишескалних модела мишића
Број страница: 114
Број слика: 52
Број библиографских података: 83
Установа и место где је рад израђен: Природно-математички факултет Универзитета у Крагујевцу
Научна област (УДК): 004 (рачунарство)
Ментор: др Милош Ивановић, доцент Природно-математичког факултета Универзитета у Крагујевцу, ужа научна област: Рачунарске комуникације
III. Оцена и обрана
Датум пријаве теме: 23. мај 2012.
Број одлуке и датум прихватања докторске дисертације:
Комисија за оцену подобности теме и кандидата: 1. др Александар Цветковић, ванредни професор Машинског факултета у Београду 2. др Бобан Стојановић, ванредни професор Природно-математичког факултета Универзитета у Крагујевцу 3. др Милош Ивановић, доцент Природно-математичког факултета Универзитета у Крагујевцу
Комисија за оцену докторске дисертације: 1. др Антун Балаж, научни саветник у Институту за физику, Београд 2. др Ненад Филиповић, редовни професор Факултета инжењерских наука Универзитета у Крагујевцу 3. др Бобан Стојановић, ванредни професор Природно-математичког факултета Универзитета у Крагујевцу 4. др Милош Ивановић, доцент Природно-математичког факултета Универзитета у Крагујевцу
Комисија за одбрану докторске дисертације: 1. др Антун Балаж, научни саветник у Институту за физику, Београд 2. др Ненад Филиповић, редовни професор Факултета инжењерских наука Универзитета у Крагујевцу 3. др Бобан Стојановић, ванредни професор Природно-математичког факултета Универзитета у Крагујевцу 4. др Милош Ивановић, доцент Природно-математичког факултета Универзитета у Крагујевцу
Датум одбране дисертације:

Јовани

Захвалности

Ова дисертација представља први обимнији документ на српском језику о резултатима истраживања у области вишескалног моделирања мишића, која се изводе на Природно-математичком факултету у Крагујевцу у сарадњи са Northeastern универзитетом у Бостону. Главни покретач и руководилац истраживања је др Бобан Стојановић, ванредни професор Природно-математичког факултета у Крагујевцу. Ова дисертација не би имала циљ и смисао без његових усмеравања и безусловног преношења свега што је научио и искусио у истраживањима везаним за биомеханику мишића. Зато, као прво, желим да изразим велику захвалност професору Стојановићу за све идеје и инспирације, али и редовне прозивке, несебично пружану подршку и вољу да препозна тренутке када је помоћ потребна.

Део резултата приказаних у овој дисертацији је настао као плод сарадње са професорима др Србољубом Мијаиловићем и др Ричардом Гилбертом са Northeastern универзитета у Бостону, па им се овом приликом захваљујем. Посебно се захваљујем професору Мијаиловићу, без чије подршке и огромног знања везаног за молекуларне, а и биомеханичке моделе мишића читаво истраживање вероватно не би било ни започето. Желим да му се захвалим и на свим корисним саветима везаним за писање научних радова, као и за адекватно представљање научних резулатата.

Захвалност дугујем и професору др Антуну Балажу, са Института за физику у Београду, за труд уложен у читање ове дисертације и корисним сугестијама везаним за сам текст. Такође се захваљујем и професору др Ненаду Филиповићу са Факултета инжењерских наука у Крагујевцу, зато што је омогућио коришћење рачунарских ресурса BioIRC-а, као и за пружање подршке развоју истраживачког потенцијала Природно-математичког факултета и Универзитета у Крагујевцу уопште.

Највећу захвалост дугујем свом ментору, др Милошу Ивановићу, доценту Природно-математичког факултета у Крагујевцу. Као ментор, професор Ивановић је био укључен у све фазе рада и истраживања приказаних у овој дисертацији, почевши од дефинисања теме, преко увођења у област рачунарства високих перформанси и сталне бриге о томе да сви технички услови за тестирање развијеног софтвера буду обезбеђени, до давања

конструктивних сугестија и подршке током израде самог текста дисертације. Посебно му се захваљујем за стрпљење и упорност у указивању на то који су битни, а који споредни циљеви, који су добри путеви и где треба тражити решења проблема.

Такође, желим да се захвалим колегиници Марини Свичевић, колегама Дарку Антонијевићу и Ђорђу Недићу, са Природно-математичког факултета у Крагујевцу, за њихов допринос у истраживањима која су довела до објављивања научних радова везаних за тему ове дисертације. Додатно, захваљујем се колегиници Марини и колеги Дарку зато што је заједнички рад са њима увек био насмејан и конструктиван.

У име свих учесника пројеката желим да се захвалим и Министратству за науку и заштиту животне средине Републике Србије за финансирања пројеката 41007 и 37013 у оквиру којих су настали и резултати ове дисертације.

На крају, желим да се извиним својој породици и пријатељима, али и сарадницима и студентима/ђацима којима сам ускратила велики део пажње и стрпљења које заслужују. Ако је школа балансирања научена, онда штета на крају неће бити велика. Време ће показати.

На крају, сваки је стил добар и свака форма прихватљива, ако су
у служби нечег што није само стил ни само форма.
Знакови поред пута, Иво Андрић

Крагујевац, јануар 2016.
Ана Капларевић-Малишић

Abstract

Multi-scale models of muscle contraction rely on the integration of physical and biochemical properties across multiple length and time scales, consequentially they are highly CPU consuming. Feasible usage of these models can be reached only by employing high performance computing environments. Existing efforts for accelerating multi-scale muscle simulations lean on general purpose parallelization techniques and frameworks. These solutions imply the usage of expensive large-scale computational resources, which produces overwhelming costs for the everyday practical application of such models. Additionally, none of them employ any methodology based on specific model domain knowledge to provide customized resource utilization. The thesis resulted from the investigations addressed to both of these deficiencies.

The dissertation introduces concept of the distributed calculations of the multi-scale muscle models in a mixed CPU–GPU environment in order to improve computational speed within a reasonable budget. The concept is applied to a two-scale muscle model, in which a finite element macro model is coupled with the microscopic Huxley kinetics model. The parallel solution is based on decomposition of the micro model domain and static scheduling policy. Computations related to Huxley models run on both CPUs and GPUs, while finite element calculations are executed sequentially and solely on the CPU. Introduced concept is implemented as a software platform, Mexie, which represents a portable and extensible software solution with modular architecture. The platform uses a specific load balancing method customized for heterogeneous execution environments. It takes into account differences in computational speeds and memory limits of available computing resources. Additionally, the dispatch algorithm considers computational weights of each micro model, providing more balanced workload. Since these computational weights could not be known before exact two-scale simulation run, a novel methodology for assessing their values is introduced. The methodology uses simple Hill phenomenological model in order to predict model states and input parameters of all micro models during two-scale simulation. These estimated data is then used by specific predictor tool, trained by machine learning techniques, for assessing micro model computational complexities.

Implemented solution was verified on both benchmark and real-world example, showing high utilization of involved processing units, strong scalability and speed-up of two orders of magnitude compared to the sequential CPU run. This major improvement in computational feasibility and significant lowering of price-to-performance ratio of the multi-scale muscle models paves the way for new discoveries in the field of muscle modeling and usage of shuch models in the future clinical applications, as well.

Keywords

multi-scale simulations, muscle modeling, finite element model, Huxley muscle model, Hill phenomenological muscle model, graphic processing unit, scheduling policy, hybrid programming model, MPI, CUDA

Сажетак

Вишескални модел мишићне контракције обухвата физичке и биохемијске процесе који се одвијају на више просторних и временских скала. Због велике рачунске сложености, овакви модели могу бити употребљиви само применом техника паралелизације и употребом рачунарских ресурса високих перформанси. Постојећа решења за убрзавање вишескалних симулација мишића су ослоњена на универзалне софтверске оквире за паралелизацију вишескалних модела. Она подразумевају употребу скупе рачунарске опреме високих перформанси, што ствара високу цену трошкова експлоатације и чини прескупом њихову употребу у свакодневној клиничкој пракси. Поред тога, ниједно од решења не користи никакву методологију којом би се специфична знања о домену искористила у сврху бољег искоришћења расположивих рачунарских ресурса. Ова дисертација је резултат истраживања везаних за превазилажење ова два недостатка постојећих решења.

У дисертацији је уведен концепт дистрибуирања прорачуна вишескалних модела мишића у окружењу састављеном од стандардних процесорских језгара и графичких процесорских јединица. Концепт је примењен у контексту двоскалног модела мишића, у којем је модел коначних елемената на макроскали упарен са Хаксли моделом кинетике попречних мостова на микро скали. Паралелно решење је базирано на статичкој декомпозицији домена микромодела. Уведени концепт је имплементиран као софтверска платформа, *Mexie*, која представља преносиво и прошириво софтверско решење модуларне архитектуре. Платформа користи специфичан метод распоређивања послова прилагођен хетерогеном извршном окружењу, где се у обзир узимају различите брзине израчунавања и меморијска ограничења ресурса. Алгоритам расподеле узима у обзир и различитост у рачунској комплексности микромодела. Како комплексност микромодела не може бити прецизно одређена пре извршења двоскалне симулације, дефинисана је методологија за процену. Методологија је базирана на употреби Хиловог феноменолошког модела за предвиђање стања и улазних параметара микромодела током двоскалне симулације. На основу процењених вредности, специфичан алат за процену, обучен техникама машинског учења, процењује рачунску комплексност свих микромодела.

Имплементирано решење је верификовано на тестним моделима, као и на реалном моделу, показујући високу искоришћеност расположивих ресурса, чврсто скалирање и убрзања од два реда величине у односу на секвенцијалну варијанту. Захваљујући ефикасности модела и значајном побољшању односа цена/перформансе, отворен је пут за нова истраживања у области физиологије мишића и направљен значајан корак у смеру употребе вишескалних модела мишића у будућој клиничкој пракси.

Кључне речи

вишескалне симулације, моделирање мишића, модел коначних елемената, Хаксли модел, Хилов модел, графичке процесорске јединице, политика расподеле, хибридни програмски модели, MPI, CUDA

Садржај

Листа слика

Листа табела

Листа скраћеница

ГПЈ	графичке процесорске јединице
КНС	К-најближих суседа
МП	мултипроцесор
МКЕ	метод коначних елемената
РВП	рачунарство високих перформанси
СП	скаларни процесор
ЦПЈ	централна процесорска јединица
DRAM	*Dynamic Random Access Memory*
FLOPS	*floating point operations per second*
FPGA	*Field Programmable Gate Array*
GPGPU	*General Purpose computing on Graphics Processing Units*
HPC	*High Performance Computing*
MISD	*Multiple Instruction Single Data*
MIMD	*Multiple Instruction Multiple Data*
MPI	*Message Passing Interface*
NUMA	*Non-Uniform Memory Access*
SIMD	*Single Instruction Multiple Data*
SISD	*Single Instruction Single Data*

SMP *Symmetric MultiProcessor*

SIMT *Single Instruction Multiple Thread*

UMA *Uniform Memory Access*

1

Увод

Истраживања у области физиологије мишића, испитивања и анализе функционалних и структурних карактеристика махом се ослањају на *in vitro* или *in vivo* експерименте. Ограничено знање о комплексним унутрашњим механизмима и њиховим међусобним утицајима повлачи за собом испитивања фокусирана на изоловане процесе или компоненте система. Ширење знања о механизмима који дефинишу физиологију мишића ослоњено само на *in vitro* и *in vivo* експерименте је споро и ограничено. *In silico* анализе детаљних модела омогућавају јефтинији и бржи механизам тестирања и евалуације хипотеза. Њима се могу идентификовати важни аспекти или корелације које захтевају даља испитивања, и одатле обезбедити, *a priori*, вредне информације за *in vitro* и *in vivo* експерименте. Ограничавајући фактор у комбиновању *in vitro* или *in vivo* са *in silico* експериментима је често сложеност постојећих биофизичких модела.

Када су у питању поремећаји у функционисању мишићно-скелетног система као последице повреда или карактеризација и прогнозирање развоја обољења неуромишићног система, истраживања морају бити опсежна и захтевају дубоко разумевање структурних и функционалних карактеристика мишића. Употреба рачунарских симулација у таквим истраживањима захтева сложене моделе који описују унутрашње физиолошке процесе са више просторних и временских скала. Упркос чињеници да су вишескални модели реалистичнији и информативнији од једноскланих, њихова ефикасност је лимитирана њиховом великом комплексношћу. Вишескалне симулације су рачунски и меморијски захтевне, па је њихова пра-

ктична употреба могућа само ако се извршавају применом техника рачунарства високих перформанси и у одговајућем окружењу.

Да би вишескалне симулације мишића могле да пруже употребљиве резултате у временском оквиру прихватљивом за апликације које би се користиле у клиничкој пракси, убрзања морају бити веома висока. Како секвенцијално извршење ових симулација може трајати данима, па и недељама, за свођење времена на неколико сати потребно је ангажовати неколико десетина или стотина процесора. Постојећа решења за убрзавање вишескалних модела мишића се ослањају на софтверске оквире опште намене и опште технике паралелизације. Она подразумевају употребу рачунарских средстава широког обима, што је превисок однос између цене и времеске уштеде да би се уопште размишљало о употреби истих модела у свакодневној клиничкој пракси. Додатно, оквири опште намене немају висок степен прилагођавања природи проблема и повећања убрзања које није везано за повећање броја процесирајућих јединица. Такође, модерна рачунарска окружења високих перформанси су типично хетерогена. Ефикасно искоришћење доступних архитектура тражи пажљиво планирање стратегије паралелизације да би се превазишле разлике у пропусном опсегу и меморијским капацитетима доступним свакој процесирајућој јединици.

Предмет ове дисертације је развој метода паралелизације вишескалних симулација мишића који је прилагођен природи модела и хетерогеном рачунарском окружењу у којем се симулација извршава. Методологија је развијана и тестирана на двоскалном моделу скелетних мишића, где је на макро скали мишић моделиран методом коначних елемената, а механичке карактеристике материјала у интеграционим тачкама се одређују молекуларним моделом мишићне контракције.[1]

1.1 Циљеви

Циљ дисертације је дефинисање методологије паралелизације и у складу са њом развој софтверске платформе за дистрибуирано извршавање двоскалне симулације мишића у хетерогеном рачунарском окружењу. Методологија треба да обезбеди преносивост и високу скалабилност, што подразумева:

- употребу, колико је то могуће, универзалних модела паралелизације;
- прилагођавање хетерогеној природи ресурса са аспекта брзине рада и меморијских капацитета извршних јединица (процесорска језгра, графички процесори, …);
- издвајање доменског знања о моделу и његову употребу у бољем искоришћењу ресурса.

Сама софтверска платформа треба да задовољи следеће захтеве:

- паралелно извршавање симулација двоскалног модела и убрзање од најмање два реда величине у односу на секвенцијално извршавање;

[1] У даљем тексту ће се под мишићем подразумевати скелетни мишић, уколико другачије није наглашено.

- извршавање симулација на хетерогеним рачунарским ресурсима који укључују произвољан број централних и графичких процесорских јединица;
- једноставно проширивање новим микромоделима и компонентама за подршку извршавању на другачијим паралелним програмским моделима.

1.2 Полазне хипотезе

Полазну основу докторске дисертације чине резултати досадашњих истраживања у области једноскалног и вишескалног моделирања мишића, као и истраживања на пољу развоја софтвера за употребу дистрибуираних рачунарских ресурса и софтвера за развој вишескалних модела уопште. Полазна претпоставка јесте да уобичајен секвенцијални модел софтверског решења симулације које се базира на вишескалном моделу мишића, у смислу перформанси, није довољан да у разумном времену изведе било какву реално корисну анализу.

Основне хипотезе дисертације су:

- паралелизацијом се може достићи убрзање од, минимум, два реда величине и тиме постићи практична употребљивост модела,
- могуће је издвајање доменског знања о конкретном моделу који се израчунава на основу којег је могуће постићи бољи степен искоришћења расположивих ресурса.

1.3 Преглед садржаја

Дисертација се састоји из девет поглавља.

Друго поглавље уводи појам вишескалних модела, даје преглед стања у области вишескалног моделирања у билошким системима, постојећих методологија, као и опис софтверских оквира за развој вишескалних модела уопште.

Треће погравље даје преглед математичких алата који су коришћени за дефинисање симулационог алгоритма двоскалног модела мишића. Описани су основни концепти метода коначних елемената, као универзалог метода за моделирање различитих физичких система. Затим је описан метод карактеристика као метод за приближно решавање парцијалних диференцијалних једначина, често коришћеног у инжењерској анализи.

На почетку **четвртог поглавља** су дате основне информације о скелетним мишићима, укључујући физиолошке функције, начин генерисања силе, као и опис пратећих биохемијских процеса. Затим је дат опис основних врста једноскалних модела мишића, праћен прегледом постојећих вишескалних модела скелетних мишића описаних у литератури. Остатак поглавља садржи детаљан опис двоскалног модела коришћеног у развоју програмске платформе.

Пето поглавље садржи преглед основних појмова паралелног рачунарства, детаљније описе касније коришћених паралелних програмских модела, а затим и метрика намењених анализи

перформанси паралелних решења. На крају поглавља, дат је преглед постојећих решења паралелизације вишескалних модела скелетних мишића.

Шесто поглавље садржи детаљан приказ програмског модела, архитектуре и начина извршавања развијене софтверске платформе за дистрибуирано и паралелно извршавање двоскалних симулација скелетних мишића. Посебна пажња је посвећена специфичном алгоритму распоређивања послова.

Додатна новина коју доноси ова дисертација представља методологија стицања доменског знања о двоскалном моделу мишића описана у **седмој глави.** Циљ методологије је добијање процене сложености израчунавања микромодела у конкретној двоскалној симулацији, који се затим користе за побољшање ефикасности паралелизације.

Осма глава садржи резултате анализе перформанси дефинисане методологије и развијеног система на тестним примерима. Затим је дат приказ перформанси система анализираним у студији случаја на моделу деформације језика.

Закључна разматрања о употребној вредности и простору за даљи развој како методологије, тако и самог софтверског решења су дата у **деветом поглављу**.

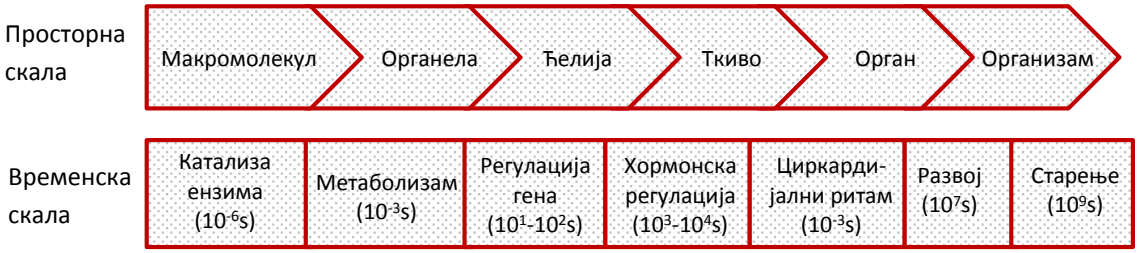
2

Вишескално моделирање у биолошким системима

У последње две деценије биомедицинске науке су доживеле револуцију. Напредак у биотехнологији, подржан огромним скоком у развоју рачунарских ресурса, је обезбедио богатство података о биолошким системима на свим нивоима огранизације. Сакупљање великих колекција експерименталних података има за циљ да подржи испитивања билошких и физиолошких функција у живим организмима. Пуно разумевање биолошких функција је условљено интегрисањем свих релевантних информација са свих скала система и њиховом употребом у реконструисању динамике интеракција просеца на свим скалама.

Биолошки системи и процеси у њима имају велику сложеност и могу се посматрати и описивати на више различитих просторних и временских скала. Процеси на свакој скали зависе од утицаја са скале изнад и испод. У раду [1], аутори дефиниши просторну скалу биолошких процеса на основу “нивоа биолошке организације” система у којима се одвијају. Она обухвата процесе у системима почевши од гена, преко проетина, појединачних ћелија, ткива, органа, па све до читавог организма. Повезана са функционалном и просторном класификацијом, временска скала биолошких процеса се простире од микросекуде ($\sim 10^{-6}\mathrm{s}$), у којима се дешавају молекуларне интеракције, до десетина година ($\sim 10^{9}\mathrm{s}$) којима се мери дужина животног века људског бића (Слика 2.1). Постојање већег броја процеса који се одвијају на ра-зли-

читим скалама и који међусобно утичу једни на друге чине билошке системе изузетно сложеним. Успешна физиолошка анализа биолошких система захтева моделе који обухаватају разматрања процеса на појединачним скалама, али и интеракције међу свима њима.



Слика 2.1 Посторне и временске скале у биолошким системима [82]

Модели који укључују описивање процеса неког система са више просторних и временских скала се називају **вишескалним моделима** (енг. *multi-scale models*). Вишескални биолошки модели, према референци [1], се могу дефинисати као модели који обухватају подмоделе компоненти са више нивоа организације и/или процеса од којих се једни одвијају много брже него други (процеси са различитих временских скала). По својој природи, ови модели често имају велику временску сложеност и, упркос перформансама данас доступних рачунарских ресурса, захтевају посебна прилагођавања са циљем смањења рачунске комплексности и добијања резултата у разумном времену.

У литератури су забележени различити приступи моделирању са циљем обезбеђивања рачунске ефикасности симулација над вишескалним моделима. У референци [2] је предложен *equation-free* метод који представља оквир за рачунарски потпомогнуту вишекалну анализу. Он предвиђа употребу микромодела на релативно малим "прозорима" у временској и/или просторној скали на којима није могуће одредити решење система из макромодела у затвореној форми. Резултати из "прозора" се интерполирају или на неки други начин апрокасимативно уклапају у понашање макромодела. *E W.* и *Engquist B.* су у референци [3] представили хетерогени вишекални метод који је реализиван као софтверски оквир опште намене. Њихов метод је заснован на идеји употребе микроскопских модела у случајевима где макроскопски модел не постоји или модел није исправан. *Brandt A.* је дефинисао *multi-grid* метод, намењен решавању парцијалних диференцијалних једначина, погодан за примену на вишескалне проблеме. Основна идеја тог метода је решавање једначина над хијерархијом мрежа различитих густина [4], при чему се при поставци модела могу узети у обзир и различите природе модела на различитим скалама [5].

Поред наведених решења везаних за одређену врсту проблема, одређен број истраживача се бавио дефинисањем општијих концепата и оквира за дефинисање вишекалних модела. Полазећи од претпоставке да се сваки вишескални систем може декомпоновати на одређен број ћелијских аутомата који међу собом комуницирају (при чему је сваки од њих дефинисан на по једној скали) *Hoekstra A.* је 2008. дефинисао *сложене аутомате* (*Complex Automata*

CxA), као општи оквир за моделирање система на више скала [6]. Као градивне блокове сложених аутомата *Hoekstra* је користио моделе базиране на агентима[2] и ћелијским аутоматима. Овај концепт је затим искоришћен за развој софтверског окружења *Multi-Scale Coupling Library and Environment* (MUSCLE), које служи дефинисању вишескалних модела и покретању симулација над њима [7,8]. Језгро MUSCLE оквира је *Distributed Space Time Coupling Library* библиотека која омогућава дводимензионо (просторно и временско) упаривање једноскалних подмодела.

Када су у питању билогија и биомедицинске науке, значајни резултати су добијени у оквиру активности везаних за VPH/Physiome пројекат [9]. Учињен је напор у развоју механизма за подршку дефинисању и интеграцији физиолошких модела дефинисаних на различитим просторним скалама система људског организма. Додатно, развијена је и софтверска библиотека OpenCMISS [10] која се користи за постављање модела и њихово упаривање, при чему модел и механизам повезивања подлежу VPH/Physiome стандардима.

[2] Моделирање засновано на агентима (*agent-based modelling*) је техника намењена моделирању система представљањем његових градивних елемената агентима, а активности унутар система интеракцијом међу агентима.

3

Математичке основе рачунарског моделирања мишића

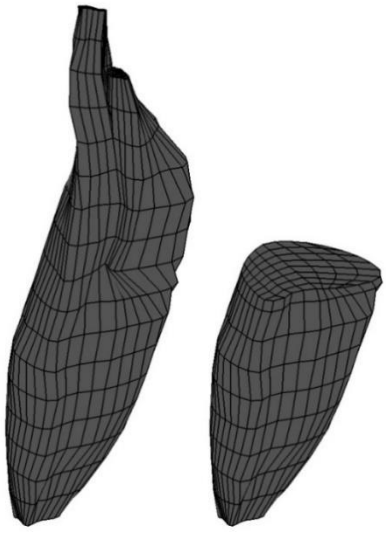
Рачунарско моделирање представља метод за одређивање тачног или приближног решења математичких модела којима су апстраховани комплексни реални системи [11]. Када је у питању моделирање понашања мишића, за описивање њихових активних и пасивних делова нелинеарних карактеристика се користе различити математички модели. Ови модели укључују различите параметре, од стимулације која контролише активацију мишића до механичких карактеристика везивног ткива која одређују пасивни напон. У овом поглављу су представљени математички алати који су употребљени за израду модела мишића развијеног у оквиру истраживања дисертације.

3.1 Метод коначних елемената

Метод коначних елемената (МКЕ) је метод намењен приближном решавању парцијалних диференцијалних једначина са задатим почетним и граничним условима. Имајући у виду да се оваким једначинама могу описивати најразличитији физички системи, метод је веома популаран у области инжењерске анализе и дизајна.

Моделирање механике биосолида спада у механику континуума којом се изучавају општи проблеми кретања и равнотеже континуално распоређеног скупа материјалних тачака. Основна идеја МКЕ анализе јесте дискретизација домена на поддомене на које се примењују општа знања и искуства механике континуума и нумеричке анализе. Анализа добијеног система омогућава нумеричку симулацију одзива континуума на задате побуде [12].

МКЕ омогућава просторно моделирање објекта чије се особине и понашање проучавају. Објекат се моделира мрежом састављеном од једноставних геометријских форми као што су троуглови и четвороуглови у дводимензионом, или различити полиедри у тродимензионом простору (Слика 3.1). Те елементарне форме које чине мрежу се називају **коначним елементима**. Захваљујући декомпозицији на мање целине, метод се може успешно применити на објекте сложене геометрије и структуре. Користећи МКЕ, објекти комплексне стуктуре се дискретизују великим бројем коначних елемената чије је понашање могуће описати тачно или приближно тачно. Елементи су међусобно повезани чворовима, а њихове међусобне релације су дефинисане законима о одржању масе, енергије и количине кретања, као и конститутивним једначинама материјала. Релације су описане системом једначина који се решава директним или итеративним методима. Решавањем система се добија конфигурација структуре у равнотежном стању [13].



Слика 3.1 Шематски приказ мишића *biceps brachii* човека дискретизованог на 3D коначне елементе [13]

Мишићи као и сва мека ткива имају сложену композитну структуру, деформишу се при спољашњем оптерећењу и унутрашњој побуди, показују анизотропне карактеристике и могу трпети велике деформације. Зато се за њихово моделирање методом коначних елемената морају користити методе нелинеарне анализе.

Проналажење равнотежног стања је основни проблем у моделирању механичког одзива тела састављеног од материјала нелинеарних карактеристика. Ако су спољашња оптерећења временски промењљива, тада се услов равнотеже система коначних елемената, којим је представљено тело, може изразити као

$$ {}^{t}\mathbf{F}^{\text{ext}} - {}^{t}\mathbf{F}^{\text{int}} = 0, \tag{3.1} $$

где су ${}^{t}\mathbf{F}^{\text{ext}}$ и ${}^{t}\mathbf{F}^{\text{int}}$, редом, вектор спољашњих чворних сила и вектор чворних сила које одговарају напонима елемената у конфигурацији у тренутку t. Важи да је

$$ {}^{t}\mathbf{F}^{\text{ext}} = {}^{t}\mathbf{F}_{C}{}^{\text{ext}} + {}^{t}\mathbf{F}_{S}{}^{\text{ext}} + {}^{t}\mathbf{F}_{V}{}^{\text{ext}} \tag{3.2} $$

где су ${}^{t}\mathbf{F}_{C}{}^{\text{ext}}$, ${}^{t}\mathbf{F}_{S}{}^{\text{ext}}$ и ${}^{t}\mathbf{F}_{V}{}^{\text{ext}}$ вектори чворних, површинских и запреминских сила редом.

Када моделирано тело има нелинеарни одзив, тада равнотежна једначина (3.1) мора важити током целе историје оптерећивања. У случају да се геометрија тела и карактеристике материјала нелинеарно мењају у времену и зависе од историје оптерећења, одређивање равнотежне конфигурације у одређеном временском тренутку захтева решавање равнотежних једначина у целокупном временском интервалу од интереса. Једна од ефикасних нумеричких метода која се користи у анализи таквих система јесте инкрементална шема „корак-по-корак” [14].

Инкрементална шема „корак-по-корак” примењена на решавање временски зависних проблема нелинеарне анализе подразумева поделу временског интервала у коме се систем разматра на одређен број подинтервала. Одређивање конфигурације система, тј. решавање равнотежне једначине се врши на крају сваког подинтервала, при чему се претпоставља да је решење у дискретном тренутку са почетка интервала познато. На основу њега се налази решење за конфигурацију која одговара наредном дискретном тренутку, тј. крају посматраног подинтервала.

Нека су сви подинтервали исте дужине Δt, где је Δt погодно одабран временски корак, и нека је решење за конфигурацију у тренутку t познато. За конфигурацију у тренутку $t+\Delta t$ једначина (3.1) гласи

$$ {}^{t+\Delta t}\mathbf{F}^{\text{ext}} - {}^{t+\Delta t}\mathbf{F}^{\text{int}} = 0. \tag{3.3} $$

За ${}^{t+\Delta t}\mathbf{F}^{\text{int}}$ важи

$$ {}^{t+\Delta t}\mathbf{F}^{\text{int}} = {}^{t}\mathbf{F}^{\text{int}} + \mathbf{F}^{\text{int}}, \tag{3.4} $$

где је $\mathbf{F}^{\text{int}}$ прираштај сила у чворовима елемената који одговара прираштају померања и напона од тренутка t до тренутка $t+\Delta t$. Вектор $\mathbf{F}^{\text{int}}$ се може апроксимирати употребом тангентне матрице, ${}^{t}\mathbf{K}$, која одговара геометријским и материјалним условима конфигурације у тренутку t,

$$ \mathbf{F}^{\text{int}} \approx {}^{t}\mathbf{K}\mathbf{U} \tag{3.5} $$

где је $\mathbf{U}$ вектор прираштаја померања у чворовима, а

$$ {}^{t}\mathbf{K} = \frac{\partial\, {}^{t}\mathbf{F}^{\text{int}}}{\partial\, {}^{t}\mathbf{U}}. \tag{3.6} $$

На основу (3.3), (3.4) и (3.5) се добија

$$ {}^{t}\mathbf{K}\mathbf{U} = {}^{t+\Delta t}\mathbf{F}^{\text{ext}} - {}^{t}\mathbf{F}^{\text{int}} \tag{3.7} $$

чијим се решавањем по $\mathbf{U}$ могу добити приближна померања у конфигурацији у тренутку $t+\Delta t$,

$$ {}^{t+\Delta t}\mathbf{U} \approx {}^{t}\mathbf{U} + \mathbf{U}, \tag{3.8} $$

на основу њих се могу израчунати напони и одговарајуће чворне силе, чије ће вредности, такође, бити приближне тачним. Због апроксимације (3.5) добијене вредности померања, сила и напона могу имати недовољну тачност. Из тог разлога се у пракси користе итеративни поступци којима се тражене величине доводе до задовољавајуће тачности.

Један од често коришћених итеративних метода јесте *Newton-Raphson* [14]. Итеративна процедура према том методу је описана следећим једначинама

$$ \begin{aligned} {}^{t+\Delta t}\mathbf{K}^{(i-1)}\Delta\mathbf{U}^{(i)} &= {}^{t+\Delta t}\mathbf{F}^{\text{ext}} - {}^{t+\Delta t}\mathbf{F}^{\text{int}\,(i-1)} \\ {}^{t+\Delta t}\mathbf{U}^{(i)} &= {}^{t+\Delta t}\mathbf{U}^{(i-1)} + \Delta\mathbf{U}^{(i)} \end{aligned} \tag{3.9} $$

где је $i = 1, 2, 3\ldots$, а почетни услови су

$$ {}^{t+\Delta t}\mathbf{U}^{(0)} = {}^{t}\mathbf{U},\ {}^{t+\Delta t}\mathbf{K}^{(0)} = {}^{t}\mathbf{K},\ {}^{t+\Delta t}\mathbf{F}^{\text{int}\,(0)} = {}^{t}\mathbf{F}^{\text{int}}. \tag{3.10} $$

Како вектор ${}^{t+\Delta t}\mathbf{F}^{\text{ext}} = {}^{t+\Delta t}\mathbf{F}^{\text{int}\,(i-1)}$ одговара силама које нису уравнотежене са напонима у елементима, то је потребно одредити инкремент чворних померања, такав да се ова неуравнотеженост елиминише. Корекција померања се врши у свакој итерацији. Поступак се зауставља када прираштаји померања и/или неуравнотежених сила буду довољно мали (Слика 3.2).

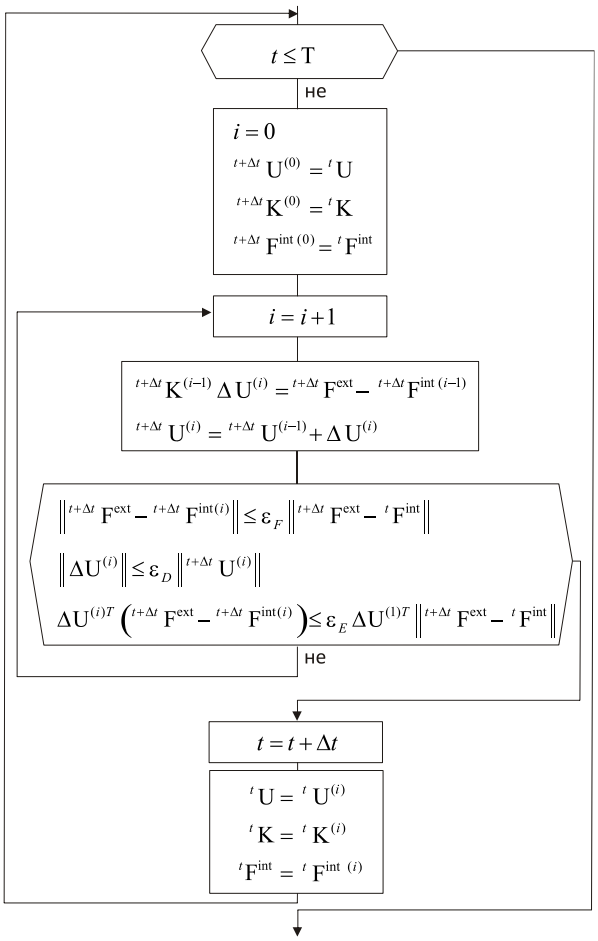
Вектор унутрашњих сила ${}^{t+\Delta t}\mathbf{F}^{\text{int}\,(i-1)}$ и матрица крутости ${}^{t+\Delta t}\mathbf{K}^{(i-1)}$ се добијају на основу једначина

$$ {}^{t+\Delta t}\mathbf{F}^{\text{int}\,(i-1)} = \int_{{}^{t+\Delta t}V^{(i-1)}} \left({}^{t+\Delta t}\mathbf{B}^{T}\ {}^{t+\Delta t}\boldsymbol{\sigma} \right)^{(i-1)} dV \tag{3.11} $$

$$ {}^{t+\Delta t}\mathbf{K}^{(i-1)} = \int_{{}^{t+\Delta t}V^{(i-1)}} \left({}^{t+\Delta t}\mathbf{B}^{T}\ {}^{t+\Delta t}\mathbf{C}\ {}^{t+\Delta t}\mathbf{B} \right)^{(i-1)} dV \tag{3.12} $$

где је ${}^{t+\Delta t}\boldsymbol{\sigma}^{(i-1)}$ напон, ${}^{t+\Delta t}\mathbf{C}^{(i-1)}$ тангентна конститутивна матрица, а ${}^{t+\Delta t}\mathbf{B}$ матрица извода интерполационих функција. Слика 3.2. даје графички приказ управо описане инкременталноитеративне процедуре.

У случају нелинеарних материјалних модела, на основу датих једначина се може видети да је основни задатак одредити напон и тангентну конститутивну матрицу у одговарајућим тачкама материјала. Тачност резултата зависи од тачности одређивања унутрашњих сила на основу померања, тј. интеграције напона. Тачност тангентне матрице не утиче на тачност решења, али значајно утиче на број итерација потребних да би се достигла конвергенција.



Слика 3.2 Алгоритамски приказ инкрементално-итеративне шеме

3.2 Метод карактеристика

Други метод за приближно решавање парцијалних диференцијалних једначина често коришћен у инжењерској анализи јесте **метод карактеристка** [15,16]. Основна идеја метода јесте свођење проблема решавања парцијалних диференцијалних једначина на решавање обичних диференцијалних једначина. У овој секцији ће бити описан поступак примене метода на хиперболичке парцијалне једначине првог реда.

Општи облик хиперболичке једначине првог реда је

$$a\frac{\partial u}{\partial x} + b\frac{\partial u}{\partial t} = c, \tag{3.13}$$

где је u функција од x и t, док су a,b,c функције од x,t,u, али не и од $\frac{\partial u}{\partial x}$ и $\frac{\partial u}{\partial t}$. Тотални диференцијал u се одређује према

$$du = \frac{\partial u}{\partial x}dx + \frac{\partial u}{\partial t}dt. \tag{3.14}$$

Елиминацијом $\frac{\partial u}{\partial x}$ из једначине (3.14) се добија

$$\frac{\partial u}{\partial t}(adt - bdx) + (cdx - adu) = 0. \tag{3.15}$$

Ако претпоставимо да једначина (3.15) важи дуж криве C која припада $xt-$ равни, тј. да C задовољава једнакост

$$adt - bdx = 0 \tag{3.16}$$

тада дуж криве C важи и

$$cdx - adu = 0. \tag{3.17}$$

Једнакости (3.16) и (3.17) се могу записати и у облику једнакости

$$\frac{dx}{a} = \frac{dt}{b} = \frac{du}{c} \tag{3.18}$$

која се назива канонским обликом једначине.

Како a и b имају јединствене вредности у било којој тачки (x,t), то и $\frac{dt}{dx}$ такође има само једну вредност, што значи да је C једина крива $xt-$ равни која пролази кроз (x,t) и задовољава једначину (3.15). Ова крива се назива **карактеристиком** или **карактеристичном кривом** кроз (x,t). Дуж карактеристике, u има константну вредност.

Како се једначине (3.16) и (3.17) ретко могу решити аналитичким путем, то се за њихово решавање често користе итеративни поступци. Нека је вредност u позната у N тачака $T_m^{(0)}\,(m=1,2,...,N)$ у $xt-$ равни. За сваку тачку $T_m^{(0)}$ се може одредити карактеристика C_m којој та тачка припада. Поступак интеграције се спроводи дуж карактеристика C_m. Карактеристика C_m се апроксимира дужима $T_m^{(k)}T_m^{(k+1)}(k=1,2,...)$, где тачке $T_m^{(k)}$ припадају $C_m(k=1,2,...)$.

Нека $x_m^{(k)}, t_m^{(k)}, u_m^{(k)}, a_m^{(k)}, b_m^{(k)}, c_m^{(k)}$ представљају редом вредности x,t,u,a,b,c у $T_m^{(k)}$ и нека су све те вредности познате. Ако претпоставимо да је $t_m^{(k+1)}$ познато, тада се вредности $x_m^{(k+1)}$ и $u_m^{(k+1)}$ могу проценити следећом итеративном процедуром:

1) Из једначине (3.16) следи

$$x_m^{(k+1)} = x_m^{(k)} + \frac{a_m^{(k)}}{b_m^{(k)}}\left(t_m^{(k+1)} - t_m^{(k)}\right) \tag{3.19}$$

одакле се одређује почетна процењена вредност за $x_m^{(k+1)}$;

2) Из једнакости (3.17) следи

$$u_m^{(k+1)} = u_m^{(k)} + \frac{c_m^{(k)}}{a_m^{(k)}}\left(x_m^{(k+1)} - x_m^{(k)}\right) \tag{3.20}$$

на основу чега се одређује почетна процењена вредност за $u_m^{(k+1)}$;

3) На основу $x_m^{(k+1)}, u_m^{(k+1)}, t_m^{(k+1)}$ се одређују $a_m^{(k+1)}, b_m^{(k+1)}, c_m^{(k+1)}$;

4) Коригована вредност $x_m^{(k+1)}$ се добија из

$$\frac{1}{2}\left(a_m^{(k+1)} + a_m^{(k)}\right)\left(t_m^{(k+1)} - t_m^{(k)}\right) - \frac{1}{2}\left(b_m^{(k+1)} + b_m^{(k)}\right)\left(x_m^{(k+1)} - x_m^{(k)}\right) = 0 \tag{3.21}$$

док се коригована вредност $u_m^{(k+1)}$ одређује из

$$\frac{1}{2}\left(c_m^{(k+1)} + c_m^{(k)}\right)\left(x_m^{(k+1)} - x_m^{(k)}\right) - \frac{1}{2}\left(a_m^{(k+1)} + a_m^{(k)}\right)\left(u_m^{(k+1)} - u_m^{(k)}\right) = 0 \tag{3.22}$$

5) Кораци 3) и 4) се понављају док корекције $x_m^{(k+1)}$ и $u_m^{(k+1)}$ не буду мање од захтеване прецизности.

4

Биомеханички модели мишића

Мишићи обухватају највећу групу ткива у телу која чине око половине телесне масе човека. Они су способни да енергију добијену оксидацијом хране и ускладиштену у организму ослобађају у виду механичке и тополотне енергије. На тај начин се развија сила за покретање делова тела, целокупног организма као и одржавање виталних функција.

Основне карактеристике мишићног ткива су: ексцитабилност (надражљивост), контрактилност и еластичност. Ексцитабилност је способност мишића да, захваљујући специфичној ћелијској структури, на нервне импулсе реагује контракцијом.

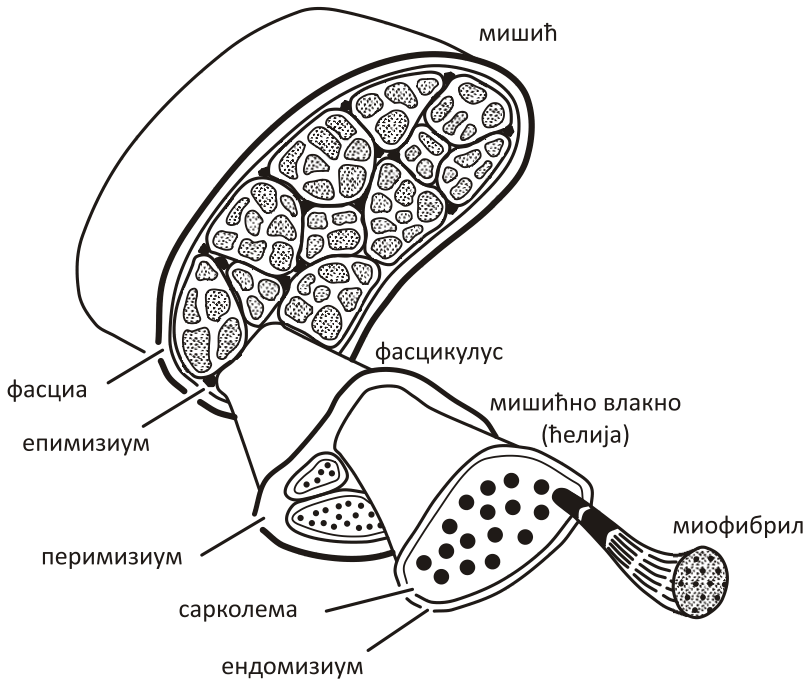
На основу структурних (грађе, инервације, начина контракције) и функционалних карактеристика, дефинишу се три врсте мишића: **скелетни**, **глатки** и **срчани**. Срчани и скелетни мишићи се називају попречно-пругастим мишићима због оптичког феномена условљеног њиховом молекуларном структуром. Друга категоризација се врши према врсти инервације, тј. начину контролисања. Срчани и глатки мишићи су контролисани од стране вегетативног нервног система, док су скелетни мишићи инервисани соматским нервним системом, па спадају у вољно контролисане. Глатко мишићно ткиво гради зидове унутрашњих органа, крвних судова и других структура које нису вољно контролисане. Имају константан напон, чак и при истезању, и у односу на скелетне мишиће, њихове контракције су спорије и правилније. Срчани мишић гради срце и изводи ритмичне покрете срца. Скелетни мишићи чине мишићни систем и карактерише их ткиво специјализовано за краткотрајне снажне контракције. Иако се категоришу као вољно контролисани, велики део активности скелетних мишића је последица

несвесних регулација, као што је одржавање равнотеже. У људском организму их има преко 600, и са малим бројем изузетака, они су везани тетивама за кости, захваљујући чему се сила произведена у мишићу преноси на скелетни систем. Обично раде у пару и то тако што у тренутку контракције један другог истеже. Упарене активности омогућавају прављење различитих покрета тела, као и мимике лица.

У првом одељку овог поглавља је детаљније описана структура и механика скелетних мишића. Затим је дат преглед постојећих модела, једноскалних и вишескалних, док последњи одељак, 4.3, садржи опис двоскалног модела развијеног у оквиру истраживања ове дисертације.

4.1 Структура и физиологија скелетних мишића

Структуру скелетних мишића сачињава неколико врста везивног ткива и попречно-пругасто мишићно ткиво. Мишић је обавијен опном, тј. слојем везивног ткива које се назива *fascia* испод којег се налази везивни омотач *епимизиум* (Слика 4.1). На епимизијум се наставља *перимизиум*, везивно ткиво које продире у тело мишића делећи га на мишићне снопове (*фасцикуле*). Фасцикуле се састоје од великог броја, од стотину до преко хиљаду, мишићних *влакана*, тј. мишићних ћелија. Свако влакно се протеже дужином целог мишића и често је постављено дијагонално. Влакна су обавијена танком опном *ендомизиумом*, везивним ткивом које повезује влакна у фасцикули.



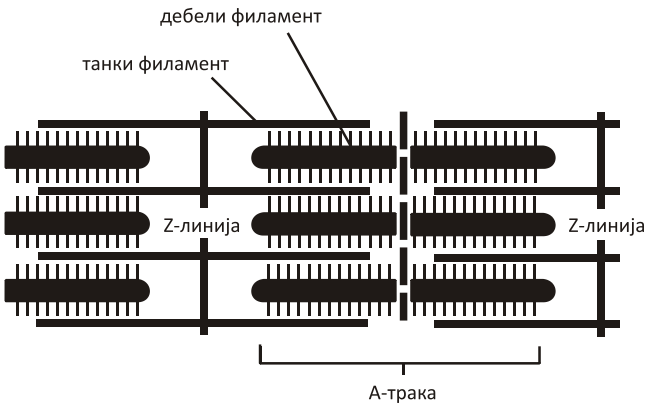
Слика 4.1 Структура мишића

Мишићна влакна су ћелије специфичне структуре. Спољашњи слој ових ћелија представља плазматична мембрана *сарколема*. Одмах испод сарколеме се налази велики број једара (и

до 150). Унутрашњост ових ћелија је испуњена полутечном цитоплазмом, *саркоплазмом*, у којој се налазе спаковане митохондрије и стотине увезаних штапићастих структура које се називају *миофибрили*. Миофибрили представљају контрактилну машинерију мишића.

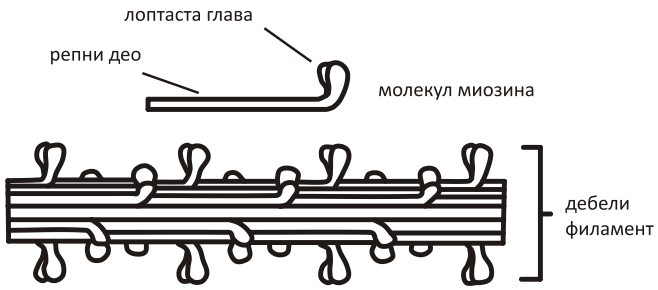
Посматрана под микроскопом, мишићна влакна имају пругаст изглед. Он потиче од правилног распореда миофибрила и њихове специфичне структуре. Миофибрили садрже снопове протеинских ланаца, *миофиламенте*, који су постављени паралелено са уздужном осом мишићне ћелије. Постоје две врсте миофиламената, дебели (*миозински*) и танки (*актински*) и заступљени су у односу 2:1.

Основна организациона јединица елемената од којих је саграђен миофибрил је *саркомера*. Саркомере се надовезују једна на другу и тако граде миофибрил (Слика 4.2). Свака саркомера је са оба краја ограничена такозваним Z линијама. Оне представљају протеинске снопове који фиксирају танке филаменте са једног краја и постављене су попречно у односу на уздужну осу мишићне ћелије. Дебели филаменти у саркомерама су са једног краја везани такозваним M линијама, које су такође попречно постављене. Групе миофиламената су у правилним интервалима пресечене Z линијама. Описани распоред миофиламената даје утисак постојања попречних пруга на мишићним влакнима.



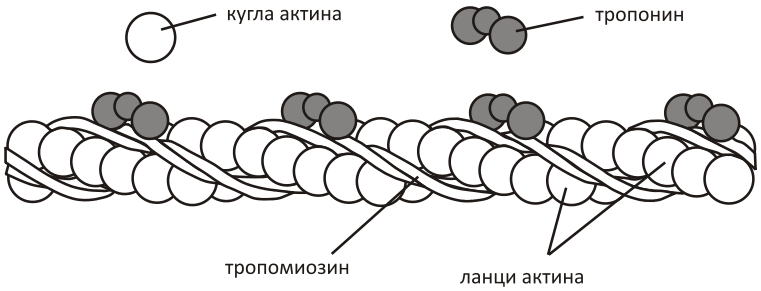
Слика 4.2 Шематски приказ основне контрактилне јединице мишића, саркомере

Дебели миофиламенти, названи *миозинским* по протеину који их гради, су влакна која се налазе у средишњем делу саркомере. У том делу саркомера има тамнију боју, која делује као тамна пруга на миофибрилу, која је позната као A-трака. Између A-трака налазе се светлије I-траке. Миозински филамент се састоји од великог броја молекула миозина (и до 180). Сваки молекул миозина се састоји од лаког меромиозина издуженог облика на који се на једном крају наставља тешки меромиозин лоптастог облика. Лоптасте главе се издвајају из филамента и то у паровима (Слика 4.4). На свакој глави постоји место за везивање за актин, као и место за катализацију хидролизе аденозинтрифосфата (АТП) који ослобађа енергију за мишићну контракцију. Како се на миозинским главама врши везивање актинских и миозинских филамената, оне се називају и *попречним мостовима* или крос-бриџевима (*cross-bridges*).



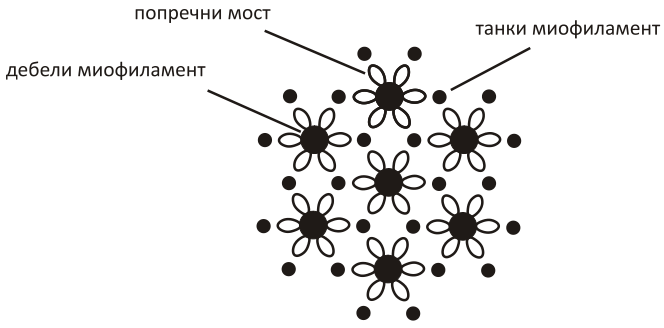
Слика 4.3 Шематски приказ дебелог миофиламента

Актинска влакана се састоје од два спирално уплетена ланца кугли актина пречника 5-6 nm. У жлебу који формирају ланци актина налази се дугачки влакнасти протеин *тропомиозин*, а на њему на сваких 38.5 nm налазе се молекули *тропонина*.



Слика 4.4 Шематски приказ танког филамента

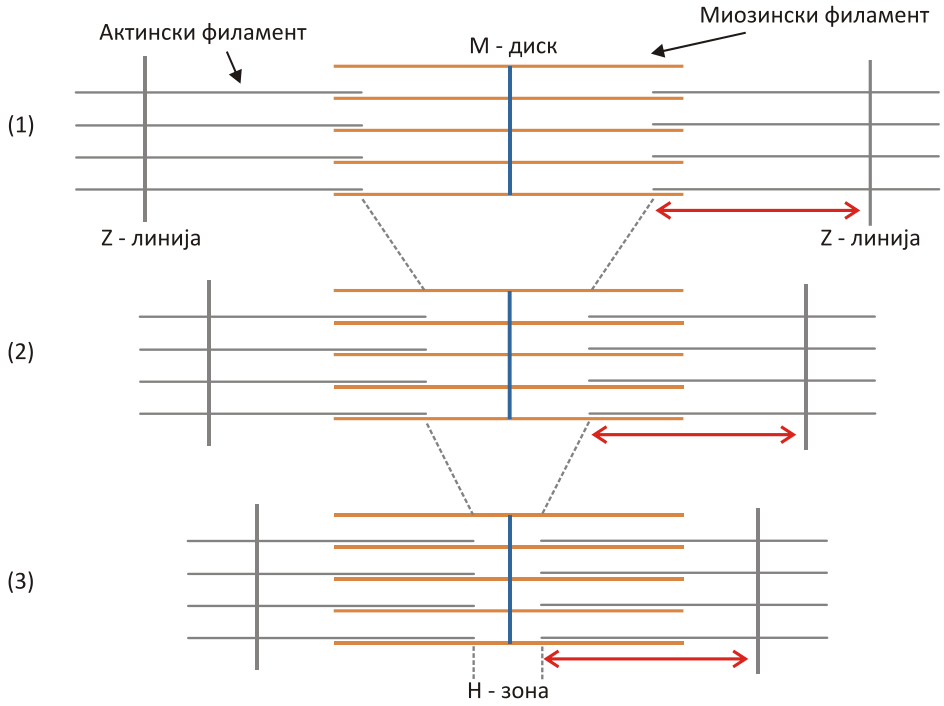
На попречном пресеку миофибрила у зони преклапања танких и дебелих филамената се може видети да је свако дебело влакно окружено са шест танких влакана (Слика 4.5).



Слика 4.5 Шематски приказ распореда танких и дебелих миофиламената у попречном пресеку кроз зону преклапања миофиламената

4.1.1 Механизам генерисања силе у скелетном мишићу

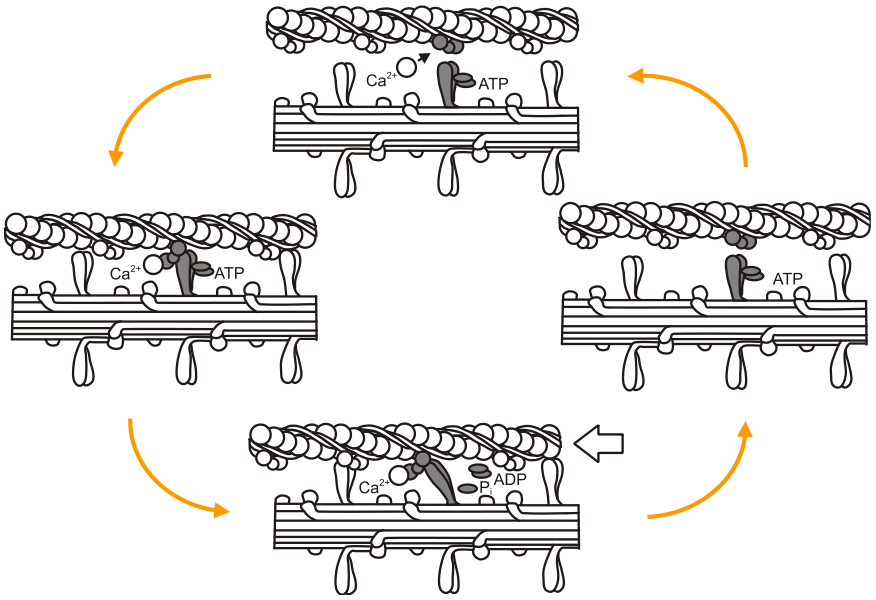
Развијање силе у скелетном мишићу је диктирано сигналима које мишић прима од соматских моторних неурона, тј. мишићних нерава. Сваки неурон се грана и на тај начин инервише више миофибрила. По пријему сигнала, миофибриле се контракују и на тај начин се ствара сила. Мишићни нерв садржи еферентне аксоне, који преносе сигнале о жељеној контракцији од централног нервног система до мишића. Примарни еферентни путеви се називају α мотонеуронима. α мотонеурон заједно са миофибрилама које иневрише чини фукционалну јединицу која се назива *моторном јединицом*. Како су сва влакна у моторној јединици инервисана истим неуроном, њихове реакције на неуро сигнале су синхронизоване [17].



Слика 4.6 Модел клизајућих филамената. (1) релаксиран мишић (2) делимично контракован мишић (3) мишић у пуној контракцији [83]

Постоји више теорија којима се објашњавају основни принципи генерисања силе у мишићу на молекуларном нивоу. Најшире прихваћена је теорија *клизећих филамената*, касније названа теоријом *попречних мостова*, чије је основе поставио A.F. Huxley [17]. Његова хипотеза представља основу тумачења механике мишића у модерној науци. Експериметално је утврђено да се при контракцији мишића А-трака саркомера не скраћује, већ је сво скраћење миофибрила последица скраћења I-траке, при чему се дужине и актинских и миозинских филамената не скраћују [18,19]. До скраћења долази тако што актински филаменти клизе дуж миозинских (Слика 4.6), а померање је последица цикличних интеракција миозинских главица са влакнима актина, такозваним *циклусима попречних мостова* или крос-бриџ циклусима (*cross-bridge cycle*). Попречни мостови миозинског влакна се везују за специјална места на актинским влакнима (*актин сајтови*). Након качења, покрети и производња силе се јављају услед ротације главица, које на тај начин вуку танка влакна ка средини саркомере.

Контракција мишића се дешава по његовој ексцитацији. Током ексцитације се у сарколеми ствара акциони потенцијал којим се иницира ослобађање молекула Ca^{2+} у интрацелуларну течност. Калцијум се везује за једну од три врсте молекула тропонина (тропонин С) који се налазе на актинском филаменту. Тада долази до конформацијске промене која помера тропомиозин ван његовог лежишта и тиме открива *активна места* на актинским нитима (*actin sites*) за које се могу везати миозинске главице дебелих филамената. Тада настаје услов за започињање *циклуса попречних мостова* (*cross-bridge cycle*), а самим тим и контракције (Слика 4.7).



Слика 4.7 Циклус попречних мостова

Пре него што започне контракција, за главице попречних мостова се вежу АТП молекули који одмах бивају разграђени на аденозиндифосфат (АДП) и неорганski фосфат, али остају везани уз главицу [20]. При разградњи се мења конформација главице тако да се она усправи према актинској нити. У таквом положају попречни мостови могу да се закаче за актинске нити у тренутку када су активна места актинске нити откривена. Веза попречног моста са активним актинским местом узрокује да се главица нагне ка репу и тиме повуче актинско влакно за собом низ миозинско влакно, тј. ка средини саркомере. Ово повлачење, названо *power stroke*, изазива контракцију, а енергија којом се активира описани замах је она коју је главица акумулирала при разградњи АТП молекула. Након замаха отпуштају се АДП и фосфатни молекул, и тиме се ослобађа место на главици за везивање новог АТП молекула. Тек по везивању новог АТП молекула, главица попречног моста се раздваја о актинске нити. По раздвајању од актинске нити, врши се АТП разградња и попречни мост је спреман за ново везивање и замах. Процес качења, замаха и раздвајања се понавља све док је калцијум везан за тропонин С и/или док је могуће померање актинских нити низ миозински ланац.

4.2 Модели скелетних мишића

Већина постојећих модела скелетних мишића се може сврстати у једну од две широке категорије: **феноменолошки** и **биофизички** модели. Феноменолошки модели се при одређивању одзива мишића ослањају на емпиријским путем одређене односе између улазних и излазних параметара модела. Најчешће се користе за моделирање на макро скали. Биофизичким моделима се тренутне карактеристике или одзив мишића одређују на основу унутрашњих процеса, тј. процеса који се одвијају у микроструктури, најчешће на нивоу миофибрила или саркомере. Упркос чињеници да су биофизички модели прецизнији, у пракси се чешће користе феноменолошки модели, јер су рачунски знатно једноставнији. Обе врсте изводе закључак о реакацији мишића на одређени стимуланс на основу понашања мишића на само једној скали.

Пут од побуде до контракције мишића и његовог одговора на спољашњи утицај подразумева међусобно зависне процесе који се одвијају на више временских и просторних скала. У случајевима када се спроводи истраживање функционалиних и физиолошких аспеката промене рада мишића у случају болести или повреде, једноскални модели нису довољни. За те потребе је неопходно дефинисати одговарауће вишескалне моделе. Њихова комплексност је драстично већа, па се паралелно са валидношћу поставља питање њихове употребљивости.

4.2.1 Феноменолошки модели мишића

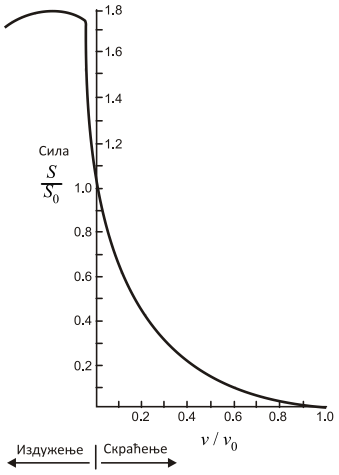
Механичко понашање мишића је опсежно изучавано још крајем XIX века, у радовима *M. Blix*-а ([21–23]), на чије се радове касније надовезао велики број научника. 1938. године, *A. V. Hill* је у [24] описао способност контракције тетанизованог мишића једначином која представља основу за већину данашњих феноменолошких модела.

Хилова једначина дефинише везу између напона и брзине контракције и то на следећи начин

$$\frac{S}{S_0} = \frac{1 - v / v_0}{1 + cv / v_0} \tag{4.1}$$

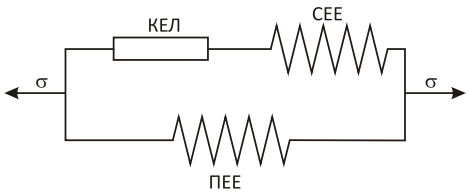
где је S напон у мишићу, v брзина контракције, S_0 максимални изометријски напон, а v_0 максимална брзина која се развија при напону $S = 0$, при чему су S_0 и v_0 константе добијене емпиријским путем. Њом је описана особина мишића да при већим брзинама контракције мишић ствара мању силу и обрнуто (Слика 4.8).

Веза дата једначином (4.1) се користи за одређивање механичког одговора мишића тако што се уграђује у **Хилов трокомпонентни функционални модел**. Трокомпонентни модел представља активни мишић састављен од три елемента (Слика 4.9). Контрактилни (КЕЛ) и нелинеарни еластични (СЕЕ) су везана редно. Контрактилни елемент има моћ скраћивања када је мишић активиран, а његово понашање је дефинисано Хиловом једначином. Серијски везаним еластичним елементом је представљена еластичност спојева актина и миозина, као и еластичност тетива. Да би се описала еластичност мишића када он није активиран, паралелно са КЕ и СЕЕ паром је везан један елестични елемент , тзв. паралелни еластични елемент (ПЕЕ), који репрезентује околно везивно ткиво.



Слика 4.8 Релација између силе и брзине контракције описана Хиловом једначином

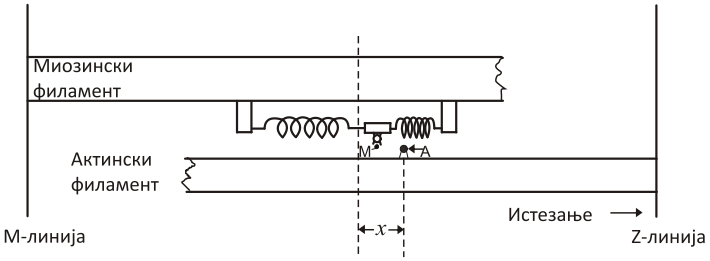
Од тада до данас представљени модел је прилагођаван новим сазнањима [25–31]. Без обзира на модификације, модели Хиловог типа користе неколико макроскопски мерених конститутивних параметара и не узимају у обзир промене напона мањих редова величине. Неосетљивост на фине промене резултира неадекватним баласирањем активних, пасивниих и спољашњих сила, нарочито када је деформација неуниформна. Ови модели нису погодни за тестирање хипотеза о процесима и појавама, скали једног мишића, али могу бити веома корисни као део модела већих система, као што је мишићно-скелетни систем.



Слика 4.9 Хилов функционални модел мишића

4.2.2 Биофизички модели мишића

Највећи број биофизичких модела се ослања на модел који су предложили *Andrew F. Huxley* и *Alan Hodgkin* 1957. године [32]. Они су поставили теорију, названу теоријом попречних мостова, којом су објаснили механизам генерисања мишићне силе. У оквиру теорије се претпоставља да мишићна контракција настаје услед цикличне интеракције миозинских главица са влакнима актина. Закачени попречни мостови граде еластичну везу између актинских и миозинских влакана, а те везе удружене генеришу активну мишићну силу и напон. Миозинске главе се каче за најближа активна места актиских нити и сваки закачен попречни мост доприноси сили контраковања. Током времена, у зависности од граничних услова, филаменти могу клизити релативно једни у односу на друге, па попречни мостови могу трпети и истезање и скраћивање.



Слика 4.10 Шематски приказ Хакслијевог модела контракције ћелије попречно-пругастог мишића

Слика 4.10 даје шематски приказ модела. Репрезентативни дебели филамент је фиксиран у простору, а стационарни референтни систем везан за M линију. Када се актински филамент помери ка Z линији, мишић се истеже, док се у супротном скупља. За попречни мост се сматра да може да се помера у правцу M или Z линије. Један попречни мост може бити закачен за само једно активно место актина у једном тренутку. Величина x представља тренутно растојање закачене актинске лоптице (активног места) A такног филамента од равнотежне позиције попречног моста M (његова централна или позиција нулте силе), тј. x представља дужину везаног попречног моста у аксијалном правцу. Да би остао закачен за активно место A, дужина попречног моста не може бити већа од одређене вредности, h. Сваки попречни мост који је закачен доприноси сили коју производи миофиламент у мери која зависи од његове дужине x. Попречни мостови се у циклусима спајају и одвајају од актинских нити, при чему се током једног циклуса њихова дужина мења. Број закачених попречних мостова одређене дужине x из домена Ω у одређеном временском тренутку t се према Хакслијевој теорији може одредити на основу

$$\frac{dn(x,t)}{dt} = \left[1 - n(x,t)\right] f(x) - n(x,t) g(x), \forall x \in \Omega \tag{4.2}$$

где је $n(x,t)$ вероватноћа да је случајно изабран попречни мост дужине x у тренутку t закачен, док су f и g редом стопе успостављања и раскидања везе миозинских и актинских глава [33]. Функције f и g су дефинисане тако да зависе само од дужине x.

Хакслијева теорија кинетике попречних мостова је током времена била модификована укључивањем различитих стања у којима се могу наћи активна места и дефинисањем правила преласка из једног у друго стање ([34–37]). Такође, оригинални Хакслијев модел подразумева да су актински и миозински филаменти крути. Касније је показало да филаменти показују одређен степен еластичности, што је требало додатно разматрати ([38–41]). Сврха ових модела је, пре свега, квалитативна анализа процеса који се одвијају на нивоу молекула. Њихова употреба у одређивању механичког одговора мишића као целине захтева узимање у обзир геометрије, композиције и активације. Симулације таквог типа су вишескалне и велике комплексности, која се може редуковати поједностављивањем појединих или већине подмодела [42].

4.2.3 Постојећи вишескални модели скелетних мишића

У последњих десет година, у истраживањима из области биомеханике мишића се интензивније ради на развоју вишескалних модела мишића. У [43] *Makssoud* представља квантитативни модел реакције скелетних мишића на електростимулацију. Ослањајући се на резултате приказане у [44,45], он дефинише модификацију Хиловог функционалног модела у којем је контрактилни елемент описан Хаксли моделом. Побуђивање контрактилног елемента се описује сложеним моделом активације који прима информације о електростимулацијама и прослеђује их микромоделу у виду степена активације [46]. За разлику од описаног модела у којем геометрија мишића није била од интереса, у контексту информативности и рачунске комплексности, већи изазов представља извођење механике пуног просторног модела скелетног мишића на основу електрофизиолошких принципа. Тако је, *Fernandez* у [47] поставио тродимензиони модел КЕ, који користи модел неурона за симултано генерисање акционог потенцијала. У свакој од интеграционих тачака користи ћелијски модел преузет из механике срчаног мишића [48], па тако активни напон у свакој од интеграционих тачака добија применом везе калцијум-напон. *Böl* је публиковао модел у којем је упарио једначине тродимензионог електричног поља са феноменолошким моделом влакана [49].

До сада, најсвеобухватнији вишескални модел скелетних мишића је развијан и публикован од стране *O. Röhle*-а и др. [50,51]. Мишић је представљен тродимнезионом мрежом КЕ. Мишићна влакна су представљена једнодимензионом мрежом правилно распоређених чворова, смештених у простор мреже КЕ. Свако влакно је придружено одређеној моторној јединици, где се подразумева да она влакна која припадају истој моторној јединици имају јединствено моделовано пропагирање акционог потенцијала. Влакна нису директно везана за интеграционе тачке мреже КЕ, јер су због смањења сложености израчунавања мање бројна и ређе распоређена. Одговор на дистрибуцију потенцијала у влакну се одређује употребом модела саркомере, који представља варијацију Хаксли модела [52]. Механички одоговор материјала се са микро нивоа преноси на макро ниво процесом хомогенизације [50]. За имплементацију целог модела коришћена је *OpenCMISS* бибилиотека [53] са готовим реализацијама хомогенизације употребом *FieldML* структура [54] и модела саркомере преузетог са *OpenCMISS* репозиторијума.

Заједничка карактеристика свих описаних вишескалних модела јесте контролисано увођење биомеханичких микромодела, чиме се балансира однос презизности и комплексности и тиме само решење држи у границама употребљивости. Један од резултата ове дисертације јесте дефинисање методологије којом се у великој мери решава проблем утицаја рапидног увећања сложености модела на његову употребљивост.

4.3 Вишескални модел мишића КЕ-ХАКСЛИ

У оквиру истраживања на које се ова дисертација односи дефинисан је двоскални модел којим се може представити један скелетни мишић или мишићни орган, као што је језик [55]. Мишић се посматра као конструкција сачињена од активних влакнастих елемената, који имају могућност контраковања при активацији унутар деформабилног континуума везивног ткива.

На макро скали, континуални модел је изграђен **методом коначних елемената** (Слика 4.11а). Понашање активних делова влакнастих елемената је описано **Хоџкин-Хаксли микро-моделом**.

Модел омогућава израчунавање деформације и генерисане силе на основу тренутних карактеристика материјала и геометрије мишића. На молекуларној скали, тренутна способност мишића да генерише силу, као и његова крутост, су дефинисани променама стања молекула, која зависе од деформације. Тренутна крутост мишића директно утиче на локалне конститутивне релације, а тиме и на равнотежу сила. Ове карактеристике материјала у мишићном ткиву, изведене на основу актомиозинских интеракција унутар мишићног влакна, као и материјалне карактеристике окружујућег везивног ткива се затим користе у интеграционим тачкама за израчунавање унутрашњих сила континуума. Ова комплексна методологија се користи при одређивању поља померања на основу којег се добија конфигурација мишића у датом тренутку. Према методологији описаној у Одељку 3.1, померања се израчунавају инкрементално, тако што се у сваком кораку по итеративној шеми достиже стање равнотеже унутрашњих и спољашњих сила. Напон у макроскопском моделу се одређује на основу активног напона који делује у правцу миофибрила и еластичног тензора којим је представљен отпор везивног ткива деформацији.

4.3.1 Макроскопски модел - модел КЕ

Једначина равнотеже деформисане конфигурације КЕ у тренутку $t+\Delta t$ и $i-$тој итерацији се може формулисати као

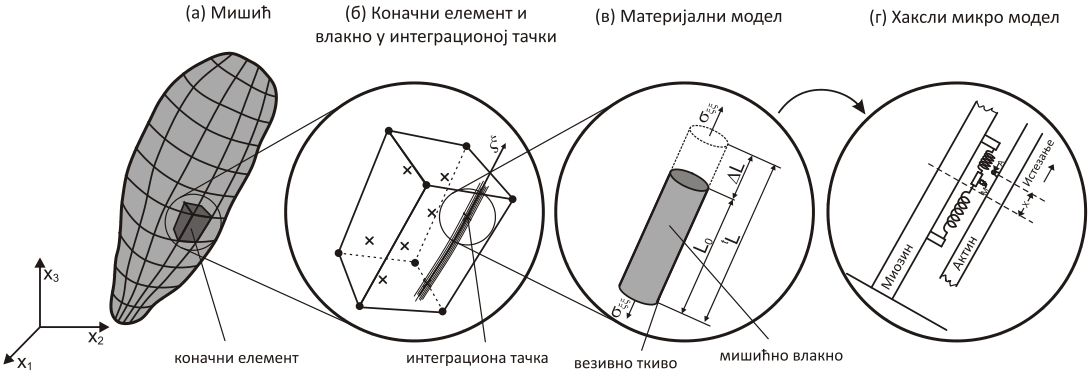
$$\left({}^{t+\Delta t}\mathbf{K}_{\text{el}}{}^{(i-1)} + {}^{t+\Delta t}\mathbf{K}_{\text{mol}}{}^{(i-1)}\right)\Delta\mathbf{U}^{(i)} = {}^{t+\Delta t}\mathbf{F}_{\text{ext}}{}^{(i-1)} + {}^{t+\Delta t}\mathbf{F}_{\text{int}}{}^{(i-1)} + {}^{t+\Delta t}\mathbf{F}_{\text{active}}{}^{(i-1)}, \tag{4.3}$$

где су ${}^{t+\Delta t}\mathbf{F}_{ext}^{(i-1)}$, ${}^{t+\Delta t}\mathbf{F}_{int}^{(i-1)}$ и ${}^{t+\Delta t}\mathbf{F}_{active}^{(i-1)}$ редом вектори спољашњих оптерећења, унутрашњих сила, и интегрисаних молекуларних сила у чворовима КЕ; ${}^{t+\Delta t}\mathbf{K}_{\text{el}}^{(i-1)}$ и ${}^{t+\Delta t}\mathbf{K}_{\text{mol}}^{(i-1)}$ су редом матрица крутости пасивних компоненти КЕ и матрица кумулативне крутости актомиозинских веза, редом; $\Delta\mathbf{U}^{(i)}$ представља инкремент чворних померања у $i-$тој итерацији. Кључни корак у стандардној формулацији коначног елемета је одређивање унутрашњих и активних сила у чворовима:

$${}^{t+\Delta t}\mathbf{F}_{\text{int}}{}^{(i-1)} + {}^{t+\Delta t}\mathbf{F}_{\text{active}}{}^{(i-1)} = \int_{{}^{t+\Delta t}V^{(i-1)}} {}^{t+\Delta t}\mathbf{B}_L{}^{T(i-1)}\,{}^{t+\Delta t}\boldsymbol{\sigma}^{(i-1)}dV, \tag{4.4}$$

где је ${}^{t+\Delta t}\mathbf{B}_L{}^{T(i-1)}$ транспонована матрица извода интерполационих функција, ${}^{t+\Delta t}\boldsymbol{\sigma}^{(i-1)}$ је тензор напона, а ${}^{t+\Delta t}V^{(i-1)}$ запремина елемента. Сабирањем равнотежних једначина појединачних елемената се добија равнотежна једначина целог мишића, која се решава тако да се обезбеди равнотежа сила $\mathbf{F}_{\text{ext}}$, $\mathbf{F}_{\text{int}}$ и $\mathbf{F}_{\text{active}}$ у границама задате толеранције на крају сваког временског корака $t+\Delta t$ ([14,56]). Вектор помераја $\mathbf{U}^{(i)}$ се током итерисања ажурира текућим инкрементом $\Delta\mathbf{U}^{(i)}$ све до испуњења $\Delta\mathbf{U}^{(i)} \approx 0$, услова конвергенције. Генерисање активне

силе, ${}^{t+\Delta t}\mathbf{F}_{\text{active}}{}^{(i-1)}$, и крутост, ${}^{t+\Delta t}\mathbf{K}_{\text{mol}}^{(i-1)}$, директно зависе од деформације мишића у правцу мишићних влакана [27].



Слика 4.11 Вишескални модел мишића. (а) Дискретизација мишића мрежом коначних елемената. (б) Тродимензиони коначни елемент са приказаним једним мишићним влакном, означеним интеграционим тачкама и оријентацијом мишићних влакана у правцу ξ. (в) Издужење мишићног влакна, ΔL, при напону од $\sigma_{\xi\xi}$, одређено као разлика текуће дужине ${}^{t}L$ и дужине у релаксираном стању L_0. (г) Хакслиев кинетички модел попречних мостова.

Имајући у виду влакнасту структуру скелетних мишића и чињеницу да је сила створена у мишићу последица контракције миофибрила, да би се одредила укупна генерисана сила у модел су уграђене информације о правцу пружања, тј. оријентацији, влакана. Слика 4.11б садржи шематски приказ коначног елемента, у којем је оријентација влакна у простору дефинисана јединичним вектором у правцу осе ξ. Овај правац одговара интеграционој тачки коришћеној за нумеричко одређивање матрица и вектора коначног елемента у једначинама (4.3) и (4.4). Напон у правцу влакна, означен са $\sigma_{\xi\xi}$, је састављен од активног дела σ_S и пасивног дела σ^E и зависи од издужења ΔL, или релативне дужине мишићног влакна $\lambda = 1 + \Delta L / L_0$ (Слика 4.11в). Релативна дужина представља однос текуће и почетне дужине влакна, тј. дужине када је напон једнак нули, L_0. Зависност $\sigma_{\xi\xi}(\lambda)$ представља **конститутивну релацију** мишића која се дефинише микромоделом придруженим интеграционој тачки.

4.3.2 Микромодел – Хоџкин-Хаксли

За моделирање мишића на микро нивоу употребљен је Хоџкин-Хаксли модел попречних мостова. Као што је већ напоменуто, према том моделу конститутивна јединица мишића је представљена међусобно интерагујућим актинским и миозинским филаменатима. Попречни мостови су носиоци еластичних веза између филамената, које у збиру генеришу активну силу у мишићу и одређују његову крутост (Слика 4.11г). Једначина (4.2) којом су описани односи битних параматера система према Хакслијевој теорији се може записати и у облику парцијалне диференцијалне једначине над доменом Ω:

$$\frac{\partial n}{\partial t}(x,t) - v\frac{\partial n}{\partial x}(x,t) = \mathcal{N}\left(n(x,t),x\right), \forall x \in \Omega \tag{4.5}$$

где $v = -dx/dt$ представља брзину клизања танког актинског филамента у односу на дебели миозински филамент; $\mathcal{N}\left(n\left(x,t\right),x\right)$ је укупни флукс промене стања пропречних мостова од незакаченог до закаченог који се одређује према

$$\mathcal{N}\left(n\left(x,t\right),x\right) = \left[1 - n(x,t)\right] f(x) - n(x,t) g(x). \tag{4.6}$$

Једначина (4.5) представља хиперболичку парцијалну диференцијалну једначину првог реда и на њено решавање је примењен метод карактеристика (Одељак 3.2). Вредности x и n се у итеративном поступку процењују следећим једначинама [57]:

$$\begin{aligned} {}^{t+\Delta t}x^{(i)} &= {}^{t}x + \frac{1}{2}\left({}^{t+\Delta t}v^{(i-1)} + {}^{t}v\right)\Delta t \\ {}^{t+\Delta t}n^{(i)} &= {}^{t}n + \frac{1}{2}\left({}^{t+\Delta t}\mathcal{N}^{(i-1)} + {}^{t}\mathcal{N}\right)\Delta t \end{aligned} \tag{4.7}$$

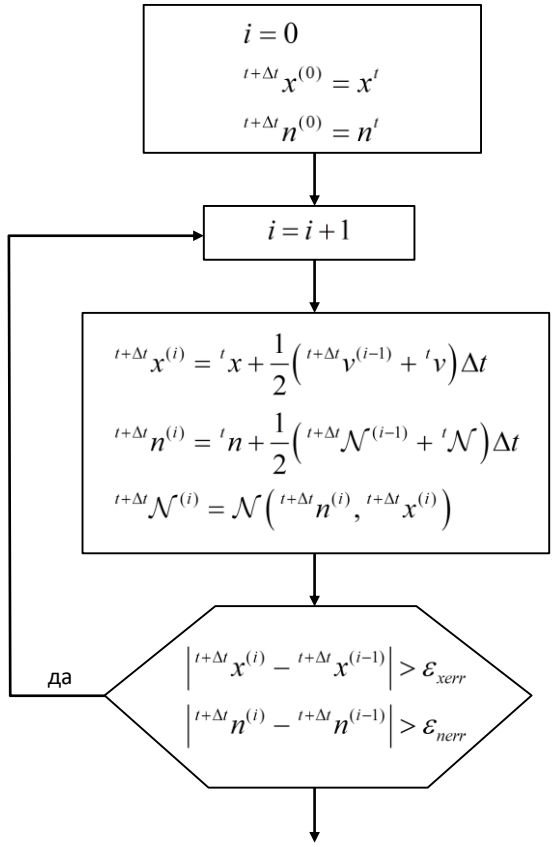
По завршеној процени вредности, може се одредити генерисана специфична сила и то према

$$\mathcal{F}(t) = k\int_{-\infty}^{\infty} x \cdot n\left(x,t\right) dx \tag{4.8}$$

где је k коефицијент крутости попречног моста, док се специфична крутост мишића одређује на основу

$$\mathcal{K}(t) = k\int_{-\infty}^{\infty} n\left(x,t\right) dx. \tag{4.9}$$

Слика 4.12 даје детаљанији приказ итеративног поступка.



Слика 4.12 Алгоритам решавања Хакслијеве хиперболичке једначине

4.3.3 Упаривање микромодела и макромодела

Активни напон генерисан у мишићу, σ_m, се одређује као

$$\sigma_m = \mathcal{F}\frac{\sigma_{iso}}{\mathcal{F}_{iso}} \tag{4.10}$$

где су σ_{iso} и $\mathcal{F}_{iso}$ редом максимални напон и максимална сила добијени Хакслијевим моделом при изометријској контракцији. Да би се добио укупан напон $\boldsymbol{\sigma}$ у правцу миофибрила сабирају се доприноси активне мишићне силе, пасивног елестичног везивног ткива, ћелијских мембрана и мишићног неконтракујућег цитоскелетона, па се он може изразити овако

$$\boldsymbol{\sigma} = \boldsymbol{\sigma}_m\phi + \boldsymbol{\sigma}^E\left(1-\phi\right) \tag{4.11}$$

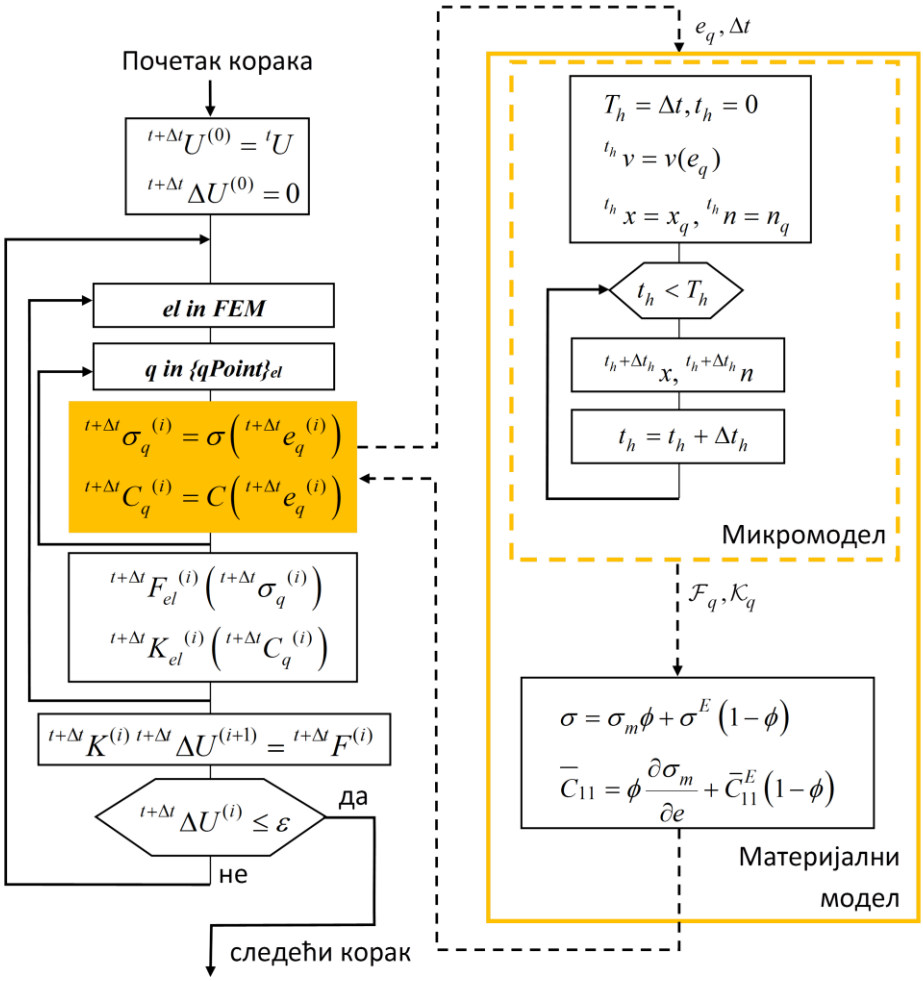
где је ϕ заступљеност мишићних влакана у укупној запремини мишића, а $\boldsymbol{\sigma}^E$ је напон у пасивном делу мишића. Члан тангентне конститутивне матрице у правцу миофибрила се одређује као

$$\overline{C}_{11} = \phi\frac{\partial\sigma_m}{\partial e} + \overline{C}_{11}^E\left(1-\phi\right), \tag{4.12}$$

где је $\bar{C}_{11}^{E}$ елемент тангенте конститутивне матрице везивног ткива у правцу миофибрила, а $\frac{\partial \sigma_m}{\partial e} \sim \mathcal{K}$ тренутна крутост мишићних влакана, такође у правцу миофибрила.

4.3.4 Опис итеративно-инкременталне шеме за вишескални модел

Имајући у виду предложену методологију повезивања макромодела и микромодела, сада је могуће дефинисати комплетну инкрементално-итеративну шему представљеног двоскалног модела мишића (Слика 4.13).



Слика 4.13 Итеративни алгоритам двоскалног модела

У складу са алгоритмом описаним у одељку 3.1 (Слика 3.2) биомеханичко понашање мишића током одређеног временског периода се симулира инкременталним прорачуном равнотежних конфигурација у више временских тренутака. С обзиром да се ради о високо нелинеарном понашању, равнотежна конфигурација у сваком кораку се одређује *Newton-Raphson* итеративном методом. Свака итерација подразумева израчунавање унутрашњих сила у моделу и одређивање тангентне матрице (матрице крутости) система за посматрану

конфигурацију, према једначинама (3.11) и (3.12). Да би било могуће израчунати силе и тангентну матрицу, неопходно је претходно израчунати напоне σ и конститутивне релације C у свим интеграционим тачкама модела. Итеративним кориговањем померања чворова модела, врши се корекција деформација у моделу, што последично изазива промену напона и конститутивних релација, а самим тим и унутрашњих сила и матрице крутости. Читав поступак се понавља до постизања равнотеже унутрашњих и спољашњих сила које делују на модел.

Кључни тренутак у инкрементално-итеративном алгоритму двоскалног модела представља израчунавање напона σ и конститутивне релације C у појединачној интеграционој тачки. Уместо коришћења аналитичке формуле, као што је случај код једноставнијих материјалних модела, у случају предложеног модела је за израчунавање напона и конститутивне релације неопходно извршити симулацију на микромоделу. Циљ симулације на микромоделу је одређивање напона и конститутивне релације при задатим деформацијама у интеграционој тачки макромодела.

Одређивање напона σ_m и конститутивне релације $\frac{\partial \sigma_m}{\partial e}$ микромодела се врши према методологији описаној у 4.3.3, при чему се брзина клизања рачуна на основу деформација у тренутном и претходном временском кораку. Израчунавање понашања микромодела се врши у временским корацима, који су због динамике посматраних хемијских процеса знатно краћи од временског корака симулације макромодела Δt. Из тог разлога се понашање микромодела током једног временског корака макромодела врши у више „микрокорака“ Δt_h. Резултујућа специфична активна сила $\mathcal{F}$ и специфична крутост $\mathcal{K}$ се затим, према методологији описаној у 4.3.2, користе за одређивање укупних напона σ и конститутивне релације C на нивоу материјалног модела интеграционе тачке.

5

Методи паралелизације програмског кода

Рачунарство високих перформанси (РВП, HPC – *High Performance Computing*) је област рачунарства развијана око идеје суперрачунара. Од првог електронског рачунара за општу употребу ENIAC-а (конструисаног 1944. за потребе прорачуна балистичких табела артиљеријских јединица војске САД-а), преко чувеног Cray-1 (најпознатији суперрачунар, постављен 1976. у Нацинонаналној лабораторији у Лос Аламосу), до, у тренутку писања овог текста, Tianhe-2 (развијен од стране 1300 научника и инжењера, смештен у Националном суперкомпјутинг центру Кине), брзине суперрачунара су расле у великим скоковима. Далеко најбољи у своје време, ENIAC је био способан да изведе 5000 операција у покретном зарезу у секунди (FLOPS – *floating point operations per second*), Cray-1 100 милиона, док је Tihane-2 на LINPACK тестовима достигао брзину 33.8 PFLOPS[3] [58].

Од ENIAC-а до данас, начин на који се велике брзине суперрачунара постижу се значајно мењао. Тако је Cray-1 представљао једнопроцесорски систем у којем брзу обраду обезбеђује векторски процесор потпомогнут брзом широкопојасном меморијом. Данашњи суперрачунари постижу брзину спрезањем великог броја вишејезгарних процесора којима обезбеђују подршку паралелном извршавању задатих послова. Не само када су у питању скупи и ретки

[3] PFLOPS = 10^{15} FLOPS

суперрачунари, концепт паралелизације као решење за постизање већих брзина обраде података и превазилажење ограничења појединачних рачунарских ресурса је постао стандард у рачунарству високих перформанси [59]. Основни недостатак паралелизације јесте то што захтева значајно додатно ангажовање програмера. У којој мери је потребно изменити секвенцијалну верзију програма зависи од проблема који се решава, архитекуре система које користи, као и програмског модела који се примењује. У овој глави је дат кратак преглед паралелних архитекура, а затим и преглед програмских модела који се користе у паралелном извршавању послова.

5.1 Архитектуре паралелних рачунара

Под паралелним рачунаром се подразумева вишепроцесорски систем који подржава паралелно извршавање послова [60]. Његова архитектура је дефинисана начином спрезања процесора и организације меморије. Постоји више критеријума по којима се може вршити класификација паралелних архитектура. Општеприхваћена Флинова таксономија (1966.) је дефинисана на основу тога колико токова података и инструкција архитектура може истовремено да подржи [61]. Тако према Флину постоје четири категорије паралелних архитектура:

- **SISD** (*Single Instruction Single Data*) или "једна инструкција, један податак" - у ову групу спадају серијски једнопроцесорки рачунари који су способни да извршавају један ток инстукција над једним током података у једном тренутку.
- **SIMD** (*Single Instruction Multiple Data*) или "једна инструкција, више података" – подразимева скуп више идентичних процесора са једном заједничком контролном јединицом, који свако на свом току података синхроно извршавају исту инструкцију. На оваквој архитектури су изграђени модерни графички процесори.
- **MISD** (*Multiple Instruction Single Data*) или "више инструкција, један податак" – према овој архитектури низ процесора, сваки са по једном контролном јединицом деле исти ток података. Сваки процесор извршава своју инструкцију и прослеђује добијено следећем у низу. Овакву архитектуру имају такозвани *dataflow* уређаји, као што програмабилни логички уређаји (FPGA - *Field Programmable Gate Array*) на чијем развоју и комерцијализацији се последњих година интензивно ради.
- **MIMD** (*Multiple Instruction Multiple Data*) или "више инструкција, више података" – подразумева организацију у којој процесори раде асинхроно, при чему сваки процесор има свој ток инструкција и ради над сопственим током података. MIMD рачунари обухватају широку групу рачунара који у свом саставу могу садржати елементе који имају другачију локалну архитектуру. Већина савремених паралелних архитектура спада у ову категорију (суперрачунари, рачунарски кластери, рачунарски грид, вишејезграни процесори рачунара опште намене).

Класификација паралелних рачунара се може извршити и према меморијској архитектури, па тако постоји [60]:

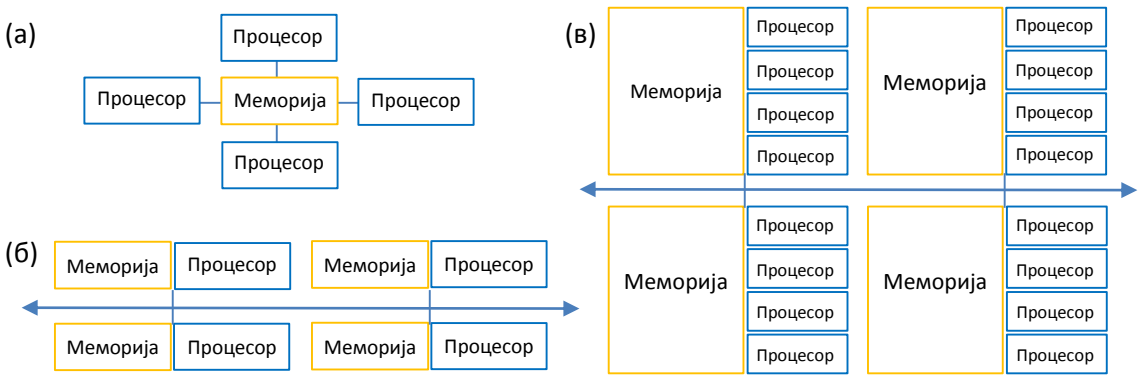
- **Архитектура дељене меморије** – меморија је заједничка за све процесоре и припрада глобалном адресном простору, па су све измене од стране једног процесора на

тривијалан начин доступне и осталим (Слика 5.1а). У зависности од тога да ли је време приступа меморији уједначено или не, постоје:

- Архитектуре са **униформним меморијским приступом** (**UMA** - *Uniform Memory Access*) – подразумевају идентичне процесоре са једнаким временом приступа заједничкој меморији. Рачунари са оваквом архитектуром се још називају и симетричним мултипроцесорима (**SMP** - *Symmetric MultiProcessor*). Типичан пример ове врсте јесу вишејезгарни и вишепроцесорски серверски системи.
- Архитектуре са **неуниформним меморијским приступом** (N**UMA** – *Non-Uniform Memory Access*) – подразумевају структуру у којој је везано више јединица симетричних мултипроцесора. Све меморије су доступне свим процесорима, у оквиру истог адресног простора, али је време приступа неуниформно, због тога што је приступ меморији суседне јединице спорији од приступа сопственој меморији.

Употреба дељене меморије програмеру обезбеђује једноставан приступ меморијским ресурсима, брзо дељење података међу процесима, док су ефикасно паралелно читање и упис брига оперативног система. Одговорност програмера је да обезбеди синхронизацију конкурентних процеса. Највећа мана архитектуре је недостатак скалабилности између меморије и процесора, јер са повећањем броја процесора саобраћај на магистали расте геометријском прогресијом.

- **Архитектура дистрибуиране меморије** – сваки процесор поседује сопствену локалну меморију и нема директни приступ меморијама других процеса (Слика 5.1б). Добављање података из адресног простора другог процесора се обавља разменом порука између процеса и то преко комуникационе мреже. Успостављање комуникације међу процесорима/процесима је одговорност програмера. Предности ове архитектуре су скалабилност меморије у односу на број процесора (пропорционално расту), бржи приступ процесора својој локалној меморији и исплативост. Највеће мане су апсолутна одговорност програмера за све детаље међупроцесне комуникације, неуниформно време приступа подацима и солидан ниво прилагођавања кода заснованог на употреби глобалне меморије режиму дистрибуираних података, тј. режиму размене порука.
- **Хибридна архитектура** – доминантна архитектура савремених паралелних рачунара, која подразумева употребу архитектура дељене и дистрибуиране меморије (Слика 5.1в). У суштини она представља више SMP машина међу собом везаних комуникационом мрежом. Очекује се да ће у даљој будућности ова архитектура бути апсолутно доминантна.



Слика 5.1 Меморијске архитектуре паралелних рачунара: (а) архитектура дељене меморије; (б) архитектура дистрибуиране меморије; (в) хибридна архитектура.

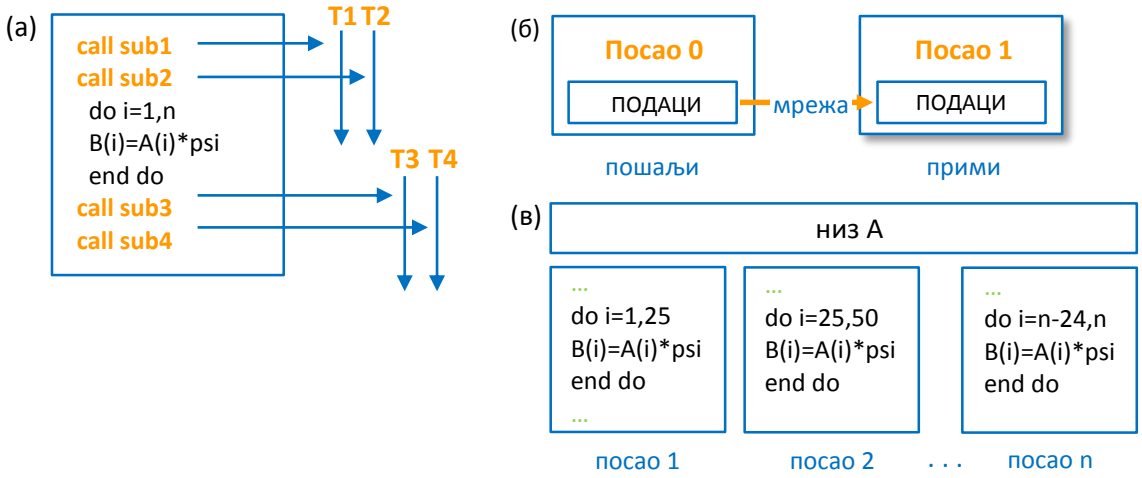
5.2 Паралелни програмски модели

Паралелизација програма захтева декомпозицију укупног посла на делове који се могу извршавати истовремено. У зависности од објекта декомпозиције, дефинишу се два основна нивоа паралелизма:

- **Функционални паралелизам** – који подразумева функционалну декомпозицију целог или дела програма и паралелно/конкурентно извршавање декомпонованих делова.
- **Паралелизам на нивоу података** – подразумева раздвајање колекције података који се обрађују на мање порције и њихово дистрибуирање процесорима на симултану обраду, при чему сваки од процесора извршава исти задатак над својом порцијом. Овај приступ се још назива **декомпозицијом домена**.

У зависности од природе проблема, могућа је примена комбинације функционалне и доменске декомпозиције. У зависности од тога да ли се подела послова извршава само једном, на почетку, или се током извршавања подела мења, декомпозиција се назива **статичком,** односно **динамичком**.

Постоји више начина којима би могла да се омогући реализација апликативног софтвера на парелелним рачунарима: проширење постојећих компајлера, проширење постојећих језика, додавање паралелно-програмског нивоа или конструисање новог језика и компајлерског система [62]. Најлакши, најбржи, најјефтинији и најпопуларнији начин јесте проширење секвенцијалних програмских језика функцијама које омогућују програму да диктира извршавање паралелних процеса, синхронизује их и контролише комуникацију [60]. Недостатак примене оваквог модела лежи у отежаном конструисању кода и откривању грешака.



Слика 5.2 Паралелени програмски модели

Програмски модел по којем се развија паралелни софтвер се бира у складу са хардверском архитекутром на којој ће се извршавати. Зато се класификација паралелних програмских модела може извршити аналогно класификацији хардверске архитекутре и то поделом на:

- **Модел дељене меморије** – паралелни задаци деле заједнички адресни простор из којег читају и у који пишу асинхроно (Слика 5.2а). За контролу конкурентности се користе механизми за контролу приступа дељеној меморији, као што су семафори. Најпознатији модел је модел нити, где сваки процес може имати вишреструке, конкурентне путеве извршавања имплеметиране у оквиру нити. Свака нит поседује сопствене локалне податке, али такође приступа и ресурсима матичног процеса. Најпознатије стандардне имплементације нити су *POSIX Threads* [63], и *OpenMP* (*Open Multi-Processing*), базиран на компајлерским директивама [64].

- **Модел порука** – паралелни задаци се обављају од стране скупа процеса, од којих сваки користи само сопствену меморију приликом извршавања, док се њихова међусобна комуникација обавља искључиво путем размене порука (Слика 5.2б). Задаци могу бити извршавани на процесорима једног или више различитих рачунарских система. Трансфер података обично захтева извођење кооперативних операција од процеса учесника. Са програмерске тачке гледишта, имплементације модела порука су обичне библиотеке комуникационих рутина које се повезују са изворним кодом. Стандардизација библиотека за модел порука је резултовала MPI (*Message Passing Interface*) стандардом [65].

- **Паралелизам података** - Највећи део паралелног посла извршава се извођењем операција над широким скупом података (Слика 5.2в). Подаци су обично организовани у једнодимензоне или вишедимензионе низове, при чему сваки процес оперише над својом порцијом података. Најпознатије имплементације су HPF (*High Performance Fortran*) и сада у све широј примени разне платформе за програмирање графичких

процесора, попут *NVIDIA CUDA* (*Compute Unified Device Architecture*) [66] или *OpenCL* библиотека[4] [67].

Архитектура	CC-UMA	CC-NUMA	Дистрибуирана
Комуникација	MPI Pthreads OpenMP shmem (MPI преко дељене меморије)	MPI Pthreads OpenMP shmem (MPI преко дељене меморије)	MPI
Скалабилност	До неколико десетина процесора	до неколико стотина процесора	хиљаде процесора
Мане	Брзина CPU-меморија магистрале	Брзина CPU-меморија магистрале Неуниформно време приступа	Системска администрација Програмирање компликовано у развоју и одржавању

Табела 5.1 Поређење дељене и дистрибуиране меморијске архитектуре [68]

Сваки од ових програмских модела има своју ширину домена примене. Тако се, на пример, модел дељене меморије заснован на нитима, може користити искључиво на SMP машинама, док се модел порука може употребити како на машинама са дељеном меморијом, тако и на кластерима и супер-рачунарима.

5.2.1 MPI

MPI представља најпопуларнију спецификацију бибилотеке намењене реализацији паралелних модела са прослеђивањем порука [60]. Први део MPI стандарда издат је 1994. године, док је други део, који дефинише паралелни улаз и излаз, издат 1996. године. MPI стандард подржава:

- Појединачне (*point-to-point*) и колективне комуникације међу процесорима, како над системским и тако и над кориснички дефинисаним типовима;
- Дефинисање групе процеса, као и контекста групе којим се дефинишу заједничке особине група и који може бити мењан од стране било ког процеса из групе у току извршавања.

Тренутно је актуелан MPI-3, објављен 2012. године [65]. Стандард је независан од од платформе и оперативног система на којем се извршава. Одатле све имплементације стандарда морају додатно да садрже дефиниције улазно/излазних операција и начин извршавања апликација на конкретном оперативном систему за који је имплементација писана. Све платформе за паралелно рачунарство поседују бар по једну MPI имплементацију. Најпознатије имплементације су MPICH [69] и OpenMPI [70] и обе подржавају језике C/C++ и *Fortran* 77/90. За

[4] Ова библиотека није намењена само имплементацији паралелизма података, али представља платформу за употребу графичких процесора.

превођење и исправљање грешака се користе преводиоци самих језика, мада постоје и посебни алати конфигурисани за MPI апликације.

5.2.2 Модел паралелизма података на графичким процесорским јединицама

Графичке процесорске јединице (ГПЈ) представљају специјализоване микропроцесоре чија је главна намена брзи приказ тродимензионе графике. Темин ГПЈ је популаризован 1999. године када је *NVIDIA* избацила на тржиште *GeForce* 256 картицу и промовисала је као прву ГПЈ на свету [66]. Период од 2000. до 2006. је обележио интентзиван развој хардверске подршке убрзавању ГПЈ, пре свега уграђивањем програмабилних јединица за извођење захтевних прорачуна везаних за сенчење и растеризацију. 2001. *NVIDIA* је први пут учинила доступним за употребу у развоју софтвера сет инструкција које је извршавао програмабилни процесор специјализован за прорачуне трасформација и осветљења, а нешто касније и за процесор који извршава процес растеризације. Њихова употреба за општију намену није била једноставна, јер је захтевала превођење проблема у контекст приказивања слике, а сам сет инструкција није садржао логичке, нити операције са целим бројевима [62]. 2006. године је хардверски концепт ГПЈ промењен тако да је за масивне графичке прорачуне почео да се користи унификован низ процесора. Први такав уређај била је *NVIDIA GeForce* 8800 картица. Увезивање унификованих процесора способних за ефикасно извршавање операција над целим и бројевима у фиксном зарезу, као нови сет инструкција за приступ меморији са адресном резолуцијом на нивоу бајта, створили су услове за интензивнију употребу ГПЈ у прорачунима опште намене (енг. *General Purpose computing on Graphics Processing Units* - GPGPU).

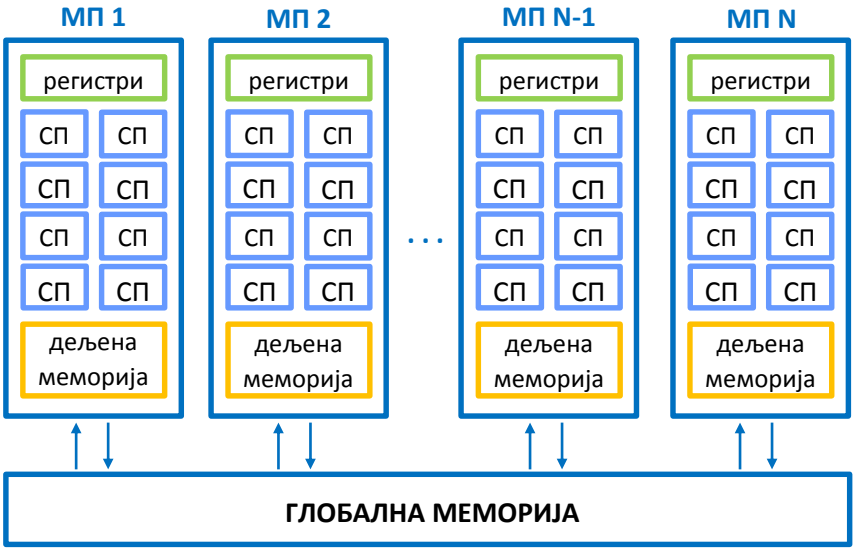
Суштина идеје за убрзавање извршавања послова коју имплементира ГПЈ јесте употреба великог броја релативно спорих[5] процесних јединица за симултано обављање истог сета инструкција над различитим подацима. Део кода који се састоји из примене одређене функције на сваки елемент великог скупа међусобно независних података је добар кандидат за извршавање на ГПЈ. Тада се примењује тзв. SIMT (*Single Instruction Multiple Thread* – једна инструкција више нити) концепт извршавања. Функција којом се обрађује један елемент скупа података добија улогу **кернел функције** (енг. *kernel function*). Извршавање функције над једним елементом представља независну јединицу посла који се извршава у оквиру нити (енг. *thread*). Нити има онолико колико има елемената у скупу података. Број процесних јединица је најчешће мањи од броја нити, па тако свака процесна јединица добија неколико нити на извршавање. У једном тренутку свака од процесних јединица извршава једну нит, и то тако што све јединице истовремено извршавају исту команду[6].

[5] Спорих у односу на ЦПЈ.

[6] Осим у случају гранања, када се прво извршавају инструкције по једном, а затим по другој. У том случају је одређен сет процесних јединица привремено беспослен.

ГПЈ архитектура и меморијски модел[7]

Са аспекта паралелних архитектура, модерна ГПЈ спада у тзв. масивно паралелне архитектуре и може бити посматрана као скалабилни **низ мултипроцесорских јединица** (МП) способних за истовремено извршавање више нити. Картица/уређај поседује **глобални управљач нитима**, енг. *thread execution manager*, који врши поделу послова и додељује колекције нити МП јединицама на извршење[8]. МП једница, тзв. мултипроцесор токова[9] (енг. *streaming multiprocessor* или *Compute Unit*), садржи колекцију **скаларних процесорских** јединица (СП), названих процесорима тока (енг. s*treaming processor*) или језгрима (енг. *cores*). СП је основни процесирајући елемент који у једном тренутку извршава једну нит. СП су способне да обаве операције над целим и бројевима у покретном зарезу, изврше операције поређења и гранање. Поред скаларних процесорских јединица сваки МП поседује јединицу за распоређивање додељених јој нити (енг. *multithreaded instruction unit*) и неколико рачунских јединица специјалне намене.



Слика 5.3 Архитектура ГПЈ

Подаци који се обрађују од стране ГПЈ морају бити смештени у меморију самог уређаја. Постоји неколико врста меморија у самом уређају различитих капацитета, брзина и доступности. Највећа, али и најспорија је глобална DRAM (*Dynamic Random Access Memory*) меморија која је доступна за читање и упис на било коју локацију свим нитима. Свака МП поседује тзв. дељену меморију, чији је садржај доступан за читање и упис свим нитима које

[7] Преглед архитектуре и меморијског модела који су дати у овом одељку одговарају графичким картицама развијеним од стране NVIDIA-е, што је у складу са општим конецпетима примењеним у свим модерним ГПЈ.

[8] У неким верзијама ГПЈ постоји организациони слој изнад МП, тзв. кластер процесора нити (*thread processing cluster*) који се састоји из неколико МП. У главном тексту се не помиње, јер није заједнички свим архитектурама и не утиче на разумевање општег концепта функционисања ГПЈ.

[9] Процесор тока података.

су јој додељене. Сваки СП има свој сет регистара доступних само нити која се тренутно на њему извршава.

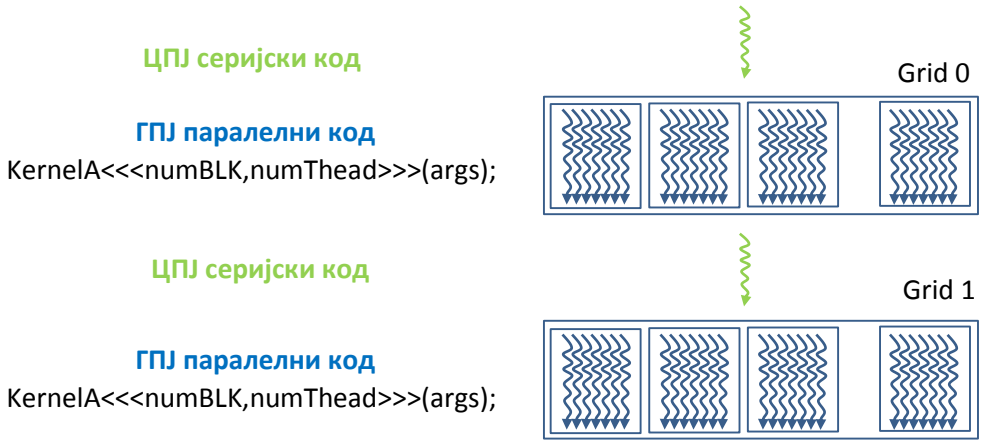
CUDA програмски модел

Развој ГПЈ прати и развој програмских модела и платформи који чине ГПЈ доступним из виших програмских језика и поједностављују процес паралелизације. Два најпопуларнија програмска модела су:

- CUDA (*Compute Unified Device Architecture*) на којем је изграђена платформа компаније *NVIDIA* намењена извршавању на ГПЈ које *NVIDIA* производи,
- модел дефинисан OpenCL (*OPEN Computing Language*) стандардом, отвореним стандардом (одржаван од стране *Khronos group* [67]) са великим бројем имплементација који дефинише јединствени модел за хетерогене хардверске платформе (ЦПЈ, ГПЈ, FPGA,...) на које се може применити[10].

У овом одељку је дат опис CUDA програмског модела који је коришћен за потребе ове дисертације.

CUDA модел извршавања подразумева да се делови програма који показују висок степен паралелизма података извршавају на ГПЈ, тзв. ***уређају*** (енг. *device*), док се сви остали изводе на централној процесорској јединици, ***домаћину*** (енг. *host*). CUDA програм представља јединствен код у којем се секвенце кода који се извршава на уређају смешта у посебно означене кернел функције или, краће, *кернеле*.

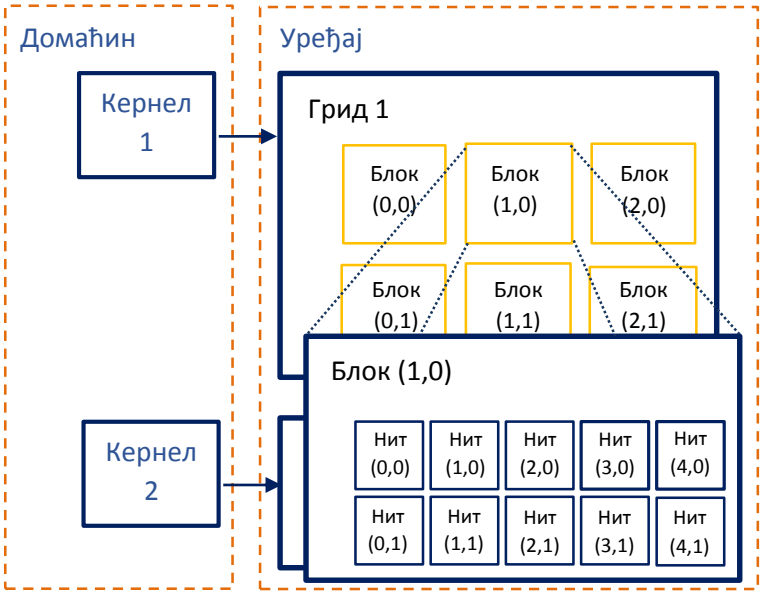


Слика 5.4 Ток извршавања типичног CUDA програма [62]

Извршавање CUDA програма почиње на *домаћину*. Када кернел функција буде позвана контролу преузима *уређај* који извршава мрежу (енг. *grid*) нити, од којих свака извршава кернелску функцију над одређеном порцијом података (Слика 5.4). У самој мрежи, нити су логички

[10] Поменути модели захтевају програмирање ниског нивоа, тј. прилагођавање кода карактеристикама хардвера на којем се извршава. Постоје и програмски модели високог нивоа као што је *OpenACC* [81], у којима се прилагођавања паралелном извршавању своде на задавање компајлерских директива.

организоване у низ блокова који садрже једнак број нити. Блокови могу бити организовани у 1Д или 2Д низ унутар мреже, док се нити унутар блока организују у 1Д, 2Д или 3Д низ. Свака појединачна нит има јединствене координате у оквиру блока којем припада, а блок у оквиру мреже. Димензије мреже и блокова представљају параметре извршне конфигурације и дефинишу се при позиву кернела.



Слика 5.5 Логичка организација нити у оквиру мреже

Када CUDA *уређај* учита кернел и формира мрежу нити, додељивање нити извршним ресурсима се врши по принципу блок по блок. Сваком мултипроцесору се на извршавање додељује одређен број блокова. Један блок може бити додељен једном МП. Како је број блокова који се могу доделити једном МП ограничен, *runtime* систем одржава листу недодељених блокова из које се преузимају нови скупови када претходно додељени буду извршени.

У којој мери ће употреба CUDA уређаја донети убрзање зависи он и од ефикасности употребе меморија доступних на *уређају*. Регистри и дељена меморија су МП меморије и може им бити приступљено великом брзином са високим степеном паралелизације. Регистри се алоцирају за сваку нит појединачно и свака нит може приступити само сопственим регистрима. Дељена меморија се алоцира за блок нити, па тако све нити у блоку могу приступити подацима у њој. Временски најзахтевнија за приступ јесте глобална меморија уређаја, па се зато посебна пажња у писању кернела и дефинисању конфигурације у којој се извршава посвећује планирању величине и локације порција података над којима ће једна нит кернела оперисати.

5.3 Анализа перформанси

Разматрање могућности учинка било каквог паралелног програмског система се врши употребом тзв. модела перформанси, чијом се применом могу вршити квалитативне или квантита-

тивне анализе [71]. Употребом модела перформанси паралелних програма могуће је добијање одговора на питања као што су: Које су границе у постизању убрзања? Да ли је убрзање ограничено количином непаралелизованог кода или масивним комуникацијама? итд. Постоји широк спектар различитих метрика које се употребљавају у таквим моделима, као што су време извршења, убрзање, ефикасност, скалабилност, меморијска захтевност, цена хардвера, однос цена/перформансе. У наставку овог одељка су дате дефиниције и описи метрика и модела који су коришћени у истраживању представљеном у дисертацији.

5.3.1 Време извршавања и убрзање

Под **временом извршавања** T паралелног програма се подразумева време које протекне од тренутка када први процесор почне са радом до тренутка када последњи заврши. Време рада појединачног, $i-$тог, процесора се састоји из врмена проведеног у прорачуну, времена проведеног у комуникацији и времена када је процесор био неупошљен, тзв. *idle time*.

Како време извршавања паралелног алгоритма варира у зависности од величине проблема, некада је представа о успешности убрзавања реалнија када се добијено време изрази релативно у односу на време извршавања секвенцијалне верзије алгоритма, тј. **убрзањем** (Ψ). Убрзање представља однос

$$\Psi = \frac{T_{sek}}{T_{par}} \tag{5.1}$$

где су T_{sek} укупно време извршења секвенцијалног, а T_{par} укупно време извршења паралелног алгоритма.

У моделирању перформанси, треба имати у виду да се у паралелним алгоритама могу наћи три врсте операција [60]:

- оне које се морају извршавати секвенцијално,
- оне које се могу извршавати паралелно,
- и тзв. **додатне операција** (енг. *parallel overhead*) у које спадају операције комуникације између процеса.

Имајући у виду наведено, могуће је увести једноставан модел убрзања. Нека је $\psi(n,p)$ убрзање које се постиже за проблем димензије n на p процесирских јединица, $\sigma(n)$ време потребно за извршење секценцијално извршаваног дела алгоритма, $\pi(n)$ време потребно за извршење оног дела алгоритма који се може паралелизовати, а $\xi(n,p)$ укупни временски трошкови комуникација. Укупно време извршења секвенцијалне верзије алгоритма за проблем димензије n се може изразити као

$$T_{sek}(n) = \sigma(n) + \pi(n) \tag{5.2}$$

док се, уз претпоставку да су порције прорачуна једнако распоређене, а процесори исти, време извршавања паралелног алгоритма за проблем димензије n и p процесора може изразити као

$$T_{par}(n,p) = \sigma(n) + \frac{\pi(n)}{p} + \xi(n,p) \tag{5.3}$$

Однос ове две величине даје процену идеалне, максималне, вредности убрзања, па тако важи

$$\psi(n,p) \le \frac{\sigma(n)+\pi(n)}{\sigma(n)+\frac{\pi(n)}{p}+\xi(n,p)}. \tag{5.4}$$

Повећање броја процесора смањује члан $\pi(n)/p,$ али повећава време комуникације, које ће од одређеног броја процесора почети да има доминантан утицај у односу на утицај већег броја процесора. Зато је очекивано да ће свака крива зависности убрзања од броја процесора имати максимум након којег ће показати тенденцију пада.

Пошто је $\xi(n,p) > 0,$ онда важи

$$\psi(n,p) \le \frac{\sigma(n)+\pi(n)}{\sigma(n)+\frac{\pi(n)}{p}+\xi(n,p)} \le \frac{\sigma(n)+\pi(n)}{\sigma(n)+\frac{\pi(n)}{p}}. \tag{5.5}$$

Нека је $f = \sigma(n)/(\sigma(n)+\pi(n)),$ тада из (5.5) следи

$$\psi(n,p) \le \frac{1}{f+(1-f)/p}. \tag{5.6}$$

На основу последње неједнакости се формулише **Амдалов закон** [72] којим се тврди да за максимално убрзање ψ које је могуће остварити упошљавањем p процесора важи неједнакост

$$\psi \le \frac{1}{f+(1-f)/p}, \tag{5.7}$$

где је f однос времена које протекне током извршавања секвенцијалних операција и укупног времена извршења. У већини проблема, функција $\xi(n,p)$ има мању комплексност од $\pi(n)$. Одатле, повећањем величине проблема, време прорачуна расте већом брзином од времена које протиче у комуникацији између процесора. Следи да је за фиксиран број процесора убрзање најчешће растућа функција величине проблема. Ова појава се назива **Амдаловим ефектом** [60].

Поред убрзања, метрика која се често користи у анализама јесте **ефикасност**. Ефикасност паралелног програма представља меру искоришћености процесора која се може описати као однос постигнутог убрзања и броја процесора. Дефинише се као

$$\mathrm{E} = \frac{T_{sek}}{pT_{par}} \tag{5.8}$$

На основу (5.4) за ефикасност $\varepsilon(n, p)$ паралелног алгоритма извршеног на p процесора и примењеног на проблем димензије n важи следеће

$$\varepsilon(n, p) \le \frac{\sigma(n) + \pi(n)}{p\sigma(p) + \pi(n) + \xi(n, p)}. \tag{5.9}$$

5.3.2 *Karp-Flatt* метрика

Један од недостатака Амдаловог закона јесте занемаривање утицаја $\xi(n, p)$, које у зависности од проблема може знатно спустити горњу границу коју максимално убрзање може имати. *Karp-Flatt* метрика исправља тај недостатак. Она се заснива на експериментално одређеној тзв. **серијској фракцији** e, која се дефинише као

$$e = \frac{(p-1)\sigma(n) + p\xi(n, p)}{(p-1)T_{par}(n,1)}, \tag{5.10}$$

где $T_{par}(n,1)$ одговара времену извршавања серијског програма, у којем коминикационе и синхронизационе операције између процеса немају утицаја.

Како из (5.10) следи $\sigma(n) + \xi(n, p) = T(n,1)e$, време паралелног извршавања се може записати као

$$T(n, p) = eT(n,1) + (1-e)\frac{T(n,1)}{p}. \tag{5.11}$$

Како је $\psi(n, p) = T(n,1) / T(n, p)$, може се извести да је

$$e = \frac{\frac{1}{\psi} - \frac{1}{p}}{1 - \frac{1}{p}}, \tag{5.12}$$

где је $\psi = \psi(n, p)$. Према ***Karp-Flatt* метрици**, за паралелни алгоритам извршен на p процесора који показује убрзање ψ, **експериментална серијска фракција** се дефинише према (5.12). За проблем фиксне димензије, ефикасност паралелног извођења од неке вредности броја процесора пада са његовим даљим повећањем. Употребом *Karp-Flatt* метрике могуће је утврдити шта је узрок те појаве. Уколико са повећањем броја процесора фракција e не расте, недовољан раст убрзања је последица ограничених могућности за паралелизацију. Са друге стране, уколико e расте са повећањем броја процесора, то значи да пораст додатних операција ограничава пораст убрзања.

5.4 Постојећа решења паралелизације вишескалних модела скелетних мишића

Колико је аутору ове дисертације у знању, једини до сада описан вишескални модел скелетних мишића у чијем су извршавању примењене паралелизационе технике јесте модел представљен у [51]. Аутори, *O. Röhle*-а и др. , су за његово постављање користили OpenCMISS софтверски оквир ([10,53]), који у себи има подршку за паралелно извршавање. Оквир је опште намене и прилагођен извршавању симулација у хетерогеним вишепроцесорским окружењима. Подржава моделе дељене и дистрибуиране меморије, где се за имплементацију дељене користи OpenMP, а дистрибуиране MPI стандард. У раду [73] се наводи да је паралелизација поменутог модела мишића спроведена применом модела дистрибуиране меморије. Над моделом је изведена статичка декомпозиција домена мреже једнодимензиоиних мишићних влакана, па је свако влакно јединствено додељено једном процесору. Израчунавања везана за модел КЕ нису паралелизована, што аутори објашњавају чињеницом да се у њиховој поставци прорачуни дифузије акционог потенцијала обављају 1000 пута чешће од прорачуна механичког модела. Они, даље, показују резултате анализе убрзања која је изведена на 1, 2 и 4 једнаких процесора, што, и поред чињенице да добијене вредности делују обећавајуће у смислу скалабиности, није довољно да би се утврдио глобални тренд.

6

Систем за дистрибуирано и паралелно извршавање двоскалних симулација мишића

Главни резултат ове дисертације јесте софтверски систем који представља оквир за извршавање симулација вишескалних модела мишића у хетерогеним рачунарским окружењима високих перформанси. Развијена платформа, названа ***Mexie***, представља проширив и скалабилно софтверско решење изграђено на MPI и CUDA програмским моделима [74]. Конструисан да омогући дистрибуирано и паралелно извршавање вишескалних модела мишића, систем је изграђен и тестиран у контексту КЕ-ХАКСЛИ модела, описаног у Одељку 4.3. У овом тренутку је постављен да подржи извршавање симулација на произвољном броју ЦП и ГП јединица, али је његова архитектура отворена за једноставно проширивање модулима који би омогућили употребу додатних програмских модела и рачунарских ресурса високих перформанси. У овој глави су представљени принципи развоја *Mexie* система, његова архитекутра са детаљним приказом свих компоненти и начина функционисања.

6.1 Принципи развоја *Mexie* система

Софтверска платформа је развијена са циљем да задовољи следеће захтеве:

- Омогућити извођење симулација над описаним двоскалним моделом;
- Омогућити паралелно извршавање симулација и убрзање од најмање два реда величине у односу на секвенцијално извршавање;
- Омогућити извршавање симулација на хетерогеним рачунарским ресурсима који укључују произвољан број ЦПЈ и ГПЈ;
- Обезбедити високу скалабилност и преносивост;
- Прилагодити рад система хетерогеној природи ресурса са аспекта брзине рада и меморијских капацитета извршних елемената;
- Омогућити једноставно проширивање новим микромоделима;
- Обезбедити висок ниво апстракције модела, тј. генерализовати макромоделе и микромоделе тако да се двоскални модел апстрахује на проблем сложено тело-материјални модел;
- Омогућити услове за једноставно проширивање система компонентама за подршку извршавању на другачијим паралелним програмским моделима, као што су модели дефинисани за *dataflow* уређаје;
- Независност компоненти задужених за паралелизацију од природе макромодела и микромодела, што отвара могућност за употребу истог система на системе другачије од мишићног.

Полазна тачка у поставци *Mexie* решења је било дефинисање стратегије паралелизације. Стратегија је постављена имајући у виду горе наведене захтеве и следеће чињенице везане за конкретан КЕ-ХАКСЛИ модел:

- Анализа временске комплексности секвенцијалне верзије симулације је показала да се 99.89% укупног времена извршавања троши на прорачуне материјалних микромодела. Применом Амдаловог закона (5.7) се може закључити да би горња граница убрзања добијеног паралелним извршавањем тих прорачуна била око 900. На основу тога је реално очекивати да добро постављен план паралеленог извршавања тог дела алгоритма може дати убрзања нивоа од неколико стотина пута у односу на секвенцијално изведен алгоритам.

- Операције различитих инстанци микромодела нису међусобно зависне, па је проблем паралелизације извршавања симулација микромодела погодан за примену декомпозиције домена.

- Два вектора од по неколико хиљада бројева у покретном зарезу описују стање једне инстанце микромодела. Узимајући у обзир да реални модели садрже најмање неколико хиљада таквих инстаници (једна за сваку интеграциону тачку), статичка декомпозиција домена се показује као погоднија, јер не повлачи за собом очигледно велику количину додатних комуникационих операција.

У складу са свим наведеним, имплементирана је стратегија која подразумева:

1. **Паралелно извршавање прорачуна материјалних модела** у маниру паралелизма података, при чему је на домен материјалних модела примењена статичка декомпозиција. Остатак алгоритма се изводи секвенцијално.
2. Употребу **модела порука** као основног модела комуникације између паралелних процеса. Овај модел је, као једини универзални механизам конкурентности, одабран због обезбеђивања преносивости решења.
3. Додатно убрзавање захтевних операција микромодела изведено применом **паралелизма података на ГПЈ**.
4. **Оптимизацију распоређивања** послова осетљиву на разлике у **перформансама** и **меморијска ограничења** извршних јединица. Политиком распоређивања је обухваћено и **знање о домену микромодела**, чиме је обезбеђено боље прилагођавање решења карактеристикама двоскалног модела и конкретном сценарију симулације која се извршава.

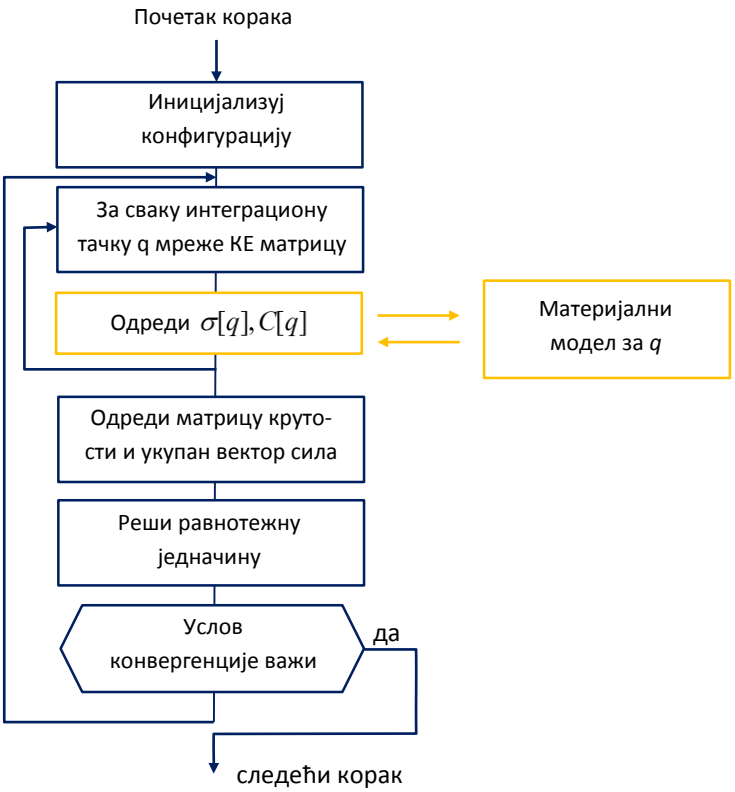
6.2 Програмски модел

Као основа за развој програмског модела употребљена је господар-слуга паралелна парадигма са моделом порука као начином комуникације између процеса. Један од учесничких процеса је господарски процес, у даљем тексту **Руководилац,** док су остали процеси слуге, у даљем тексту **Радници**. Сваки процес, без обзира на то да ли је Руководилац или Радник, има сопствени извршни контекст реализован од стране једног **процесирајућег елемента (ПЕ)**. Под процесирајућим елементом се подразумева:

- једна ЦП јединица/језгро,
- пар ЦП/ГП јединица, где је ГПЈ задужена за извођење захтевних операција у маниру паралелизма података, док је за остале операције задужена ЦПЈ.

Процеси за чије се извршавање користи искључиво ЦП јединица ће у даљем тексту бити називани **ЦП процесима**, док ће процеси који користе ГП јединицу за убрзавање захтевних операција бити означавани као **ГП-подржани процес**и.

Подела послова током извођења симулације је организована тако да је Руководилац задужен за прорачуне КЕ модела, док се прорачуни материјалних модела расподељују свим учесничким процесима. Оваква подела је захтевала извесна прилагођавања симулационог алгоритма описаног у одељку 4.3 (Слика 4.13). Наиме, захтеви за одређивање тренутних напона и извода напона упућених материјалним моделима, тј. сви прорачуни материјалних модела у текућој итерацији се обављају пре рачунања унутрашњих сила и матрице крутости, чиме је омогућено да се симулације више микро модела истовремено покрену и извршавају (Слика 6.1).



Слика 6.1 Двоскални модел - измењени КЕ алгоритам

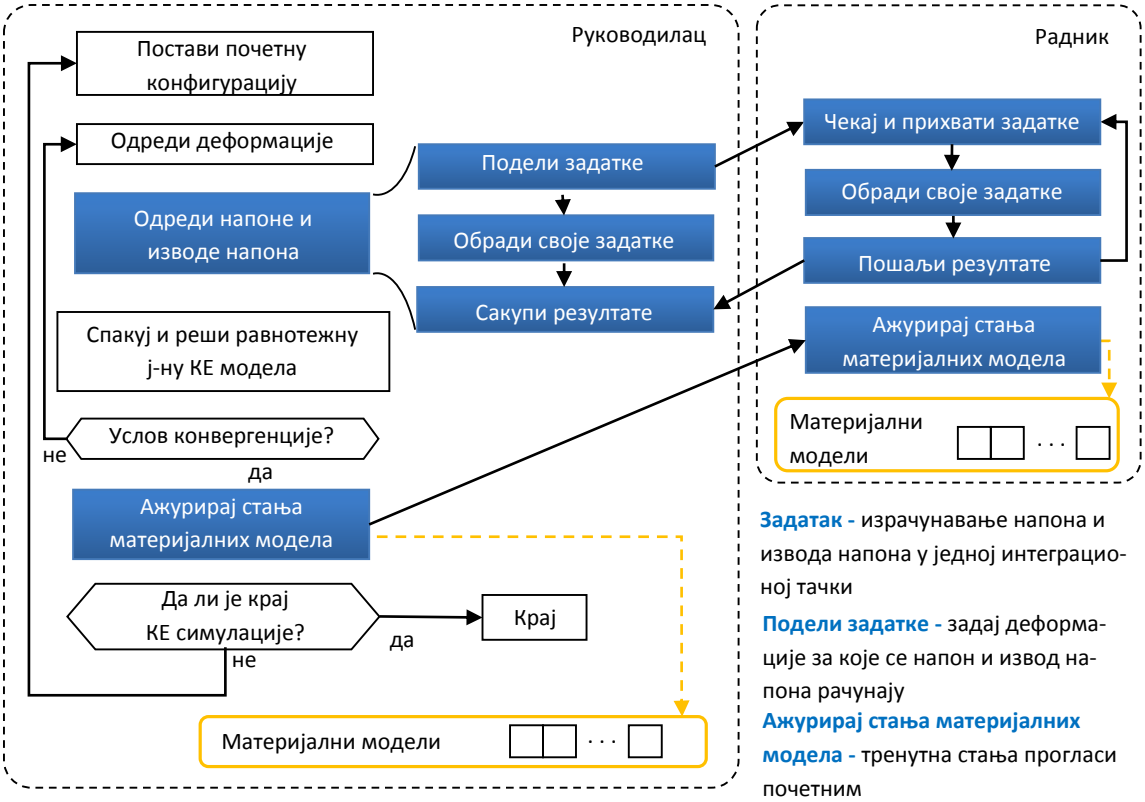
На домен материјалних модела се примењује статичка декомозиција. Сваки процес је задужен да спроводи прорачуне над истим скупом материјалних модела током читаве симулације. Оваквом поделом је омогућено да сваки процес за себе чува параметре и стања модела за које је задужен, а подаци који се размењују између процеса су сведени на колекције улазних и излазних параметара материјалних модела. Због дистрибуције материјалних модела, комуникација између Руководиоца и Радника подразумева још једну тачку комуникације. На крају сваког временског корака КЕ симулације, стање у коме се материјални модел нашао на крају последње итерације је потребно поставити као почетно за наредни корак. Процес Руководилац има задатак да о томе обавести раднике (Слика 6.2).

Јединицом посла који се дистрибуира учесничким процесима, у даљем тексту **задатком**, се сматра прорачун тренутног напона и извода напона у једној интеграционој тачки модела КЕ. У свакој итерацији КЕ симулације, Руководилац укупну количину задатака дели између свих учесвујућих процеса, укључујући и себе самог, и по обављеном послу сакупља резултате (Слика 6.2). Пре почетка двоскалне симулације, Руководилац спроводи декомпозицију тако да количина посла коју сваки од процеса добије буде што уравнотеженија. Оптимизација расподеле посла се убавља у складу са меморијским ограничењима и брзинама процесирајућих елемената на којима се ти процеси извршавају, као и рачунским комплексностима примерака микромодела, уколико су такви подаци доступни.

Комуникација међу процесима се одвија по моделу порука, имплементираним MPI стандардом. Руководилац и Радници представљају MPI процесе који током двоскалне симулације

комуницирају позивима синхроних рутина. Асинхрона комуникација би донела врло мало уштеде, из разлога што ни један процес не може добити нови сет задатака док Руководилац не прикупи све одговоре које је захтевао у једној итерацији и не реши равнотежну једначину. Синхрона комуникација је омогућила поједностављење делова алгоритма који се односе на комуникацију Руководиоца са Радницима. Наиме, за слање деформација и прикупљање података о напонима и крутостима су коришћене MPI функције One-to-All типа, које саме према задатој шеми величина порција разбијају податке на порције које се шаљу од једног процеса свима и обрнуто. Чекања се одвијају и на подели задатака и на прикупљању резултата.

Треба још нагласити, да како се задаци који се шаљу увек односе на исте материјалне моделе, то су одлуке донете при декомпозицији домена од кључне важности за ефикасност паралелизације. Зато је посебна пажња у имплементацији описаног програмског модела посвећења поступку доношења одлука о распоређивању задатака.



Слика 6.2 Програмски модел Руководилац - Радник

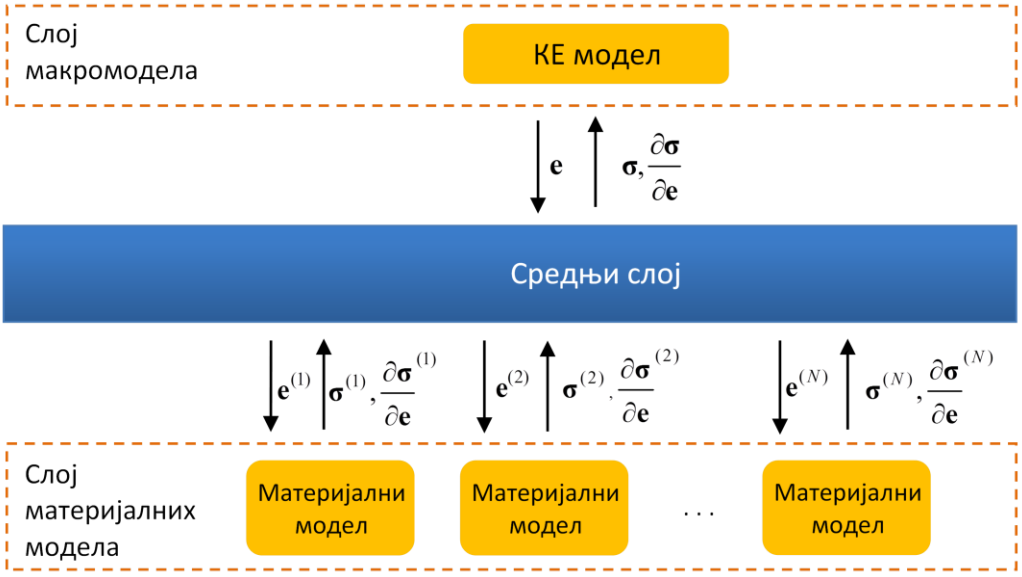
6.3 Архитектура програмског решења

Mexie систем је развијан у објектно-оријентисаном парадигмом, по моделу заснованом на компонентама. У пројектовању се тежило обезбеђивању једноставне проширивости система у смислу материјалних модела и додатних паралелних програмских модела. Постигнут је висок степен модуларности, а тиме поједностављен развој и тестирање сваке компоненете појединачно.

На високом нивоу апстаракције, посматрано из контекста двоскалне симулације, развијени систем поседује трослојну архитектуру која обухвата: **слој макромодела**, **средњи слој** и **слој материјалних модела** (Слика 6.3).

У слоју макромодела се поставља модел КЕ и извршава симулација. Захтеви за одређивањем тренутних карактеристика материјала се прослеђују средњем слоју. Након обављених прорачуна, средњи слој одговара израчунатим вредностима напона и извода напона у свакој од интеграционих тачака.

Слој материјалних модела садржи материјалне моделе који се користе за описивање понашања мишића на молекуларном нивоу. Сваки примерак материјалног модела одговара једној интеграционој тачки и има сопствене вредности параметара и варијабли стања којима су дефинисане карактеристике и стање материјала у тој тачки. Овај слој комуницира искључиво са средњим слојем. На захтев који стиже из средњег слоја материјални модел се преводи у ново стање које одговара задатој деформацији, на основу којег се даље одређују вредности напона и његовог извода у правцу мишићног влакна.



Слика 6.3 Трослојна архитектура *Mexie* система

Са аспекта двоскалног модела, средњи слој игра улогу посредника између макро слоја и слоја материјалних модела. Он је одговоран за:

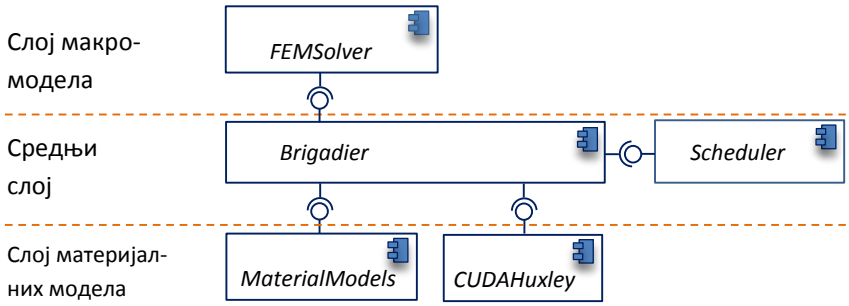
- **прослеђивање захтева** из макро слоја одговарајућим примерцима материјалних модела,
- **преузимање одговора** материјалних модела и слање тражених вредности напона и извода напона у интеграционим тачкама макро слоју.

У извршном контексту, главна улога средњег слоја јесте подршка паралелизацији у складу са дефинисаном стратегијом, па се тако у овом слоју врши декомпозиција домена материјалних модела на подскупове и њихово придруживање учесничким процесима. Резултат декомпозиције је тзв. **ПМИ мапа** (Процес - Материјални модел - Интеграциона тачка) у којој

су забележени подаци о томе ком процесу су који материјални модели додељени и којим интеграционим тачкама који модел одговара. На основу ПМИ мапе, средњи слој врши пресликавање података из захтева макро слоја на захтеве које прослеђује микро слоју, као и паковање одговора добијених из микро слоја у редоследу који одговара интеграционим тачкама.

Детаљном анализом свих задатака система и њихових међусобних условљености је издвојено пет функционалних целина, па је свака од њих имплементирана у оквиру посебне програмске компоненте (Слика 6.4), и то:

1. **Компонента макромодела (*FEMSolver*)** - реализује активности које се одвијају у Слоју макромодела, тј. омогућава постављање модела КЕ и изводи целокупну симулацију;
2. **Управљач (*Brigadier*)** – управља извршавањем целе двоскалне симулације, припада средњем слоју;
3. **Распоређивач (*Scheduler*)** – на захтев управљача одређује оптимални распоред послова, креира ПМИ мапу и врши распоређивање података према ПМИ мапи, припада средњем слоју;
4. **Компонента материјалних модела (*MaterialModels*)** – реализује активности Слоја материјалних модела које се одигравају на ЦПЈ;
5. **ГПЈ Хаксли симулатор (*CUDAHuxley*)** – представља имплментацију Хаксли модела на ГПЈ и припада слоју материјалних модела.

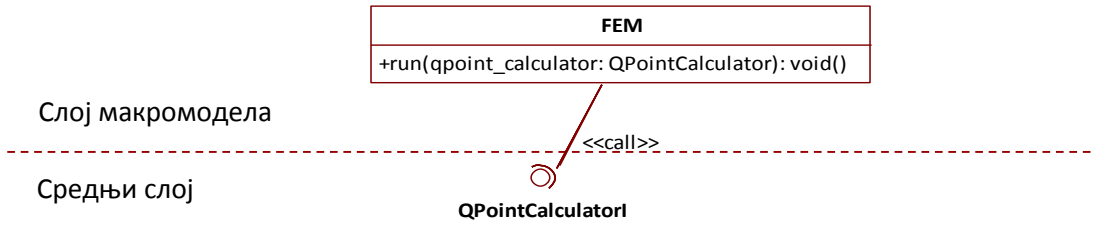


Слика 6.4 Компоненте *Mexie* система

6.3.1 Детаљи слоја макромодела

Све активности макромодела изводи компонента ***FEMSolver***. Централна класа ове компоненте је *FEM*, која је способна да учита податке о моделу КЕ и изврши измењени алгоритам КЕ симулације описан у претходном одељку (Слика 6.1). Подаци о моделу КЕ подразумевају геометрију мреже КЕ, материјалне карактеристике елемената, граничне услове и оптерећења. Компонента ***FEMSolver*** комуницира са компонентама средњег слоја преко интерфејса *QPointCalculatorI*, који користи класа *FEM*. Комуникација се одвија у оквиру следећих активности:

1. Учитавање модела и извршавање симулације иницира реализатор *QpointCalculatorI*.
2. На местима где је потребно извршити прорачуне материјалних модела, FEM пакује потребне податке у одговарајуће колекције и шаље *QpointCaluclatorI*–у на обраду.
3. На крају сваког корака КЕ симулације, FEM обавештава QpointCaluclator да је потребно ажурирати стања материјалних модела.



Слика 6.5 *QpointCalculatorI* као интерфејс средњег слоја према слоју макромодела

6.3.2 Детаљи слоја материјалних модела

Слој материјалних модела садржи класе којима се имплементирају различити модели материјала. Сваки модел је одређен реализацијама следећих апстрактних типова (Слика 6.6):

- *MaterialModel* – реализатор овог типа је класа која, за задато стање модела и задату деформацију, врши промену стања модела и одређује вредности излазних параметара, тензора напона и извода напона ,
- *MaterialModelParameter* – реализатор је класа која памти параметре модела,
- *MateralModelState* – реализатор је класа којом се описује стање модела.

Одређивање тренутних карактеристика материјала се одвија тако што се одговарајућем примерку *MaterialModel* упућује позив *calculate()* метода којем се у аргументу прослеђује последње стање модела материјала, тј. инстанца класе *MateralModelState*.

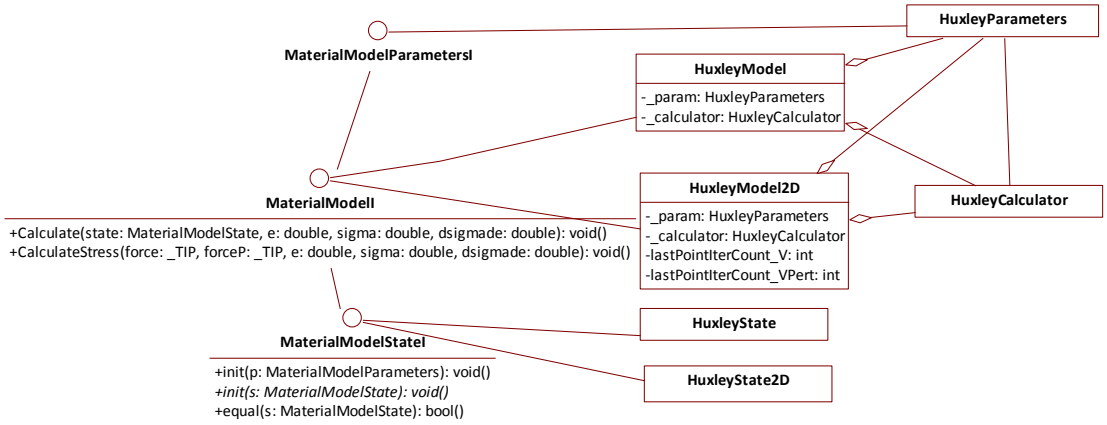
За потребе описаног двоскалног модела мишића, дефинисани су једнодимензиони и дводимензиони **Хаксли материјални модели (*HuxleModel*, *HuxleModel2D*)**. Додатно, дефинисани су и модели који описују понашање линеарно еластичних материјала, који су коришћени за описивање механичког одзива везивног ткива.

Хаксли материјалним моделима се описује нелинеарно понашање мишићног ткива. За одређивање доприноса активног дела мишићног влакна (4.10), они користе **Хакслијев микромодел**. Материјални модел покреће симулацију над микромоделом и по завршеној симулацији добија податак о сили коју генерише активирани мишић. Добијеним вредностима ова класа додаје утицај везивног ткива и формира укупан тензор напона и извода напона у координатном систему мишићног влакна (4.12).

Дефинисане су две имплементације Хакслијевог микромодела. Прву врсту чини имплементација која се извршава искључиво на ЦПЈ, а другу она која за масивна израчунавања користи ГПЈ. ЦПЈ имплементација Хаксли микромодела је смештена у

компоненету *MaterialModels*, док је ГПЈ имплементација издвојена у посебну компоненту *CUDAHuxley*.

Обе имплементације подразумевају програмску реализацију делимично проширеног алгоритма описаног у Одељку 4.3.2 (Слика 4.12). Наиме, на крају сваке симулације микромодела је одређивање вредности $\mathcal{F}$ и $\mathcal{K}$ (једначине (4.8) и (4.9)), за које је потребно израчунавање вредности интеграла од n, односно xn по x. Сама функција n је различита од нуле у релативно уском интервалу вредности x, па њихове дискретизоване репрезентације, вектори $\mathbf{N}$ и $\mathbf{X}$, могу бити сведене на један интервал вредности x. Током прорачуна, вредности x се увећавају или смањују у зависности од брзине деформације, па сам вектор $\mathbf{X}$ полако излази из оквира ненула вредности $\mathbf{N}$. Да би се избегла непотребна алокација меморије за векторе $\mathbf{N}$ и $\mathbf{X}$ у широком интервалу, на крају сваког временског корака микро симулације се врши ажурирање оба вектора тако да покривају област са ненула вредностима.



Слика 6.6 Дијаграм класа компоненте *MaterialModels*

ЦПЈ имплементација Хаксли микромодела

ЦПЈ имплементацију Хаксли микромодела (Одељак 4.3.2) изводи класа *HuxleyCalculator*. Позивом метода *calculate()* се за задате почетне вредности вектора *X* и *N* и задату брзину деформације по итеративном поступку, описаном у Одељку 4.3.2, одређује стање у саркомерама, а затим и сила коју мишићно влакно генерише.

ГПЈ имплементација Хаксли микромодела

Овом имплементацијом Хаксли модела је предвиђено да ГПЈ, у SIMD маниру, изводи рачунски интезивне делове Хаксли симулације, који подразумевају операције над векторима стања Хаксли микромодела. Остале кораке алгоритма изводи ЦПЈ, секвенцијално. Вектори са којима кернел функције оперишу садрже неколико хиљада бројева у покретном зарезу. Два од тих вектора, X и N, описују стање једног примерка микромодела. Како ГПЈ може изводити операције искључиво над подацима који се налазе у меморији уређаја, то је цена вишка операција које се тичу преноса података о стању микромодела релативно висока.

Пошто ГП јединицу користе ГП-подржани процеси за оперисање над колекцијом микромодела, то се у меморији ГПЈ подсистема чувају стања свих микромодела из колекције за коју је процес који је користи задужен. На тај начин се избегава непотребна размена стања микромодела између ЦПЈ и ГПЈ у свакој итерацији макромодела.

У тренутку када је потребно извршити симулације над моделима из колекције, у ГП меморију се само копирају задате деформације (Слика 6.7). Затим се појединачно за сваки микромодел, чија се стања чувају на ГПЈ подсистему, изводе симулације. Током симулације над једним микромоделом се за најзахтевније операције позивају кернели, и то за:

- иницијализацију вектора брзине деформације $\mathbf{V}$,
- израчунавање текућих вредности вектора $\mathbf{X}$ и $\mathbf{N}$,
- одређивање услова конвергенције итеративног алгортима и
- одређивање вредности силе $\mathbf{F}$ и крутости $\mathbf{K}$.

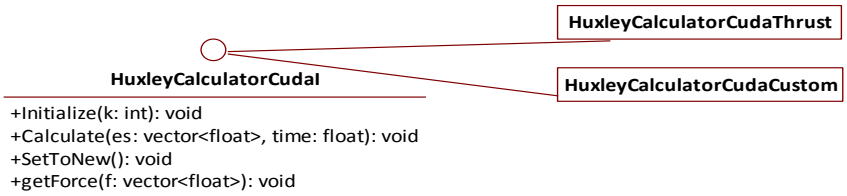
Израчунате вредности силе и крутости за све микромоделе се чувају у меморији уређаја, па се по завршетку симулације над свим моделима копирају у меморију хоста.

Додатно, на крају сваког временског корака се врше ажурирања вектора $\mathbf{X}$ и $\mathbf{N}$. Ажурирање је изведено тако што се одређивање параметара потребних за израчунавање нових вредности изводи на ЦПЈ, док се само постављање нових вредности изводи позивима кернелских функција.

```
copy deformation e from the host memory to the GPU device memory;
 foreach Micromodel in chunk do
 begin
   invoke kernel to initialize shortening velocity V;
   repeat
      repeat
         invoke kernel to calculate X;
         invoke kernel to calculate N;
         invoke kernel to calculate IterationStopCondition;
      until IterationStopCondition is true;
      recalculate X,N;
   until SimulationStep is last;
   invoke kernel to calculate F;
   invoke kernel to calculate K;
  end;
copy F,K from GPU device memory to host memory;
```

Слика 6.7 Псеудо код ГПЈ имплементације алгоритма Хаксли симулације над једним микро моделом.

ГПЈ имплементације наведених кернела (Слика 6.7) су изграђене на CUDA програмском моделу и представљају реализације типа *HuxleyCalculatorCudaI.* Битно је нагласити да изведене тако да једна ГПЈ може да буде коришћена од стране само једног процеса током трајања читаве двоскалне симулације. Као што је и приказано на дијаграму класа компоненте *CUDAHuxley* (Слика 6.8), постоје две реализације *HuxleyCalculatorCudaI* типа, једна која за извршавање векторских операција користи *Thrust* бибилиотеку, *HuxleyCalculatorCudaThrust*, док се друга, *HuxleyCalculatorCudaCustom*, ослања на самостално имплементиране кернелске функције.



Слика 6.8 Дијаграм класа комопоненте *CUDAHuxley*

6.3.3 Детаљи средњег слоја

Компоненте средњег слоја управљају извршавањем двоскалне симулације, дефинишу и спроводе стратегију паралелизације. Активности које се обављају у овом слоју се могу разврстати у три основне групе:

1. комуникација са макро слојем,
2. комуникација са микро слојем,
3. декомпозиција домена материјалних модела, креирање ПМИ мапе и преуређивање колекција захтева и одговора у складу са ПМИ мапом.

Наведене активности обављају класе које су смештене у компонентама Управљача и Распоређивача (Слика 6.4).

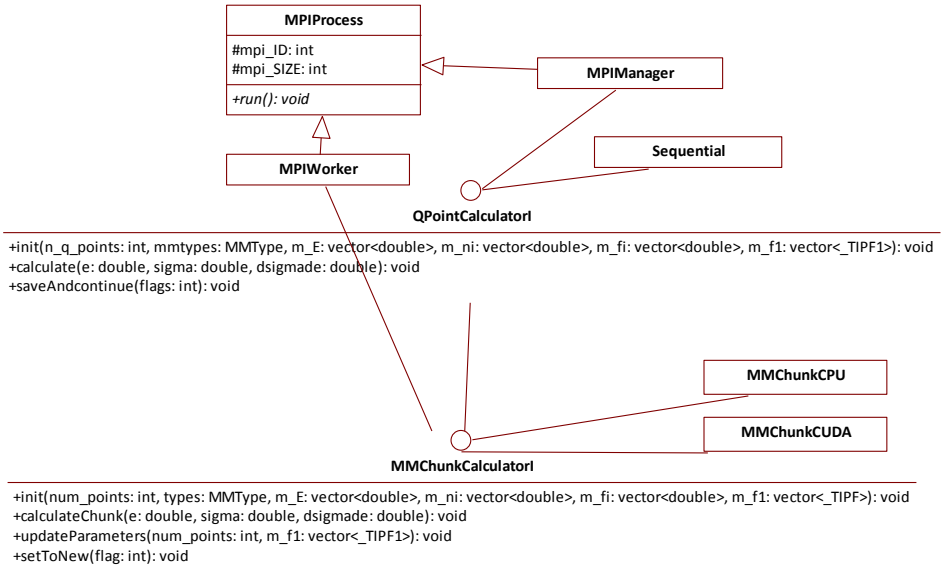
Компонента Управљача

Управљач је централна компонета *Mexie* система, којом се остварује посредовање између макро слоја и слоја материјалних модела и руковођење извршавањем двоскалне симулације. Два кључна типа ове компоненте су *QPointCalculatorI* и *MMChunkCalculatorI* (Слика 6.9).

QPointCalculatorI прима захтев од *FEM* инстанце да за задате деформације одреди напоне и изводе напона у свим интеграционим тачкама. Добијени захтев преводи у захтеве упућене материјалним моделима које прослеђује *MMChunkCalculatorI*-у. *MMChunkCalculatorI* поседује референце на примерке материјалних модела којима су захтеви упућени, па има задатак да сваком од њих пошаље одговарајућу поруку. По добијању, одговор о вредности напона и њихових извода прослеђује *QPointCalculatorI*-у који их смешта у контекст КЕ мреже, тј. мапира их на интеграционе тачке и враћа назад инстанци *FEM* класе.

Поред ова два типа, за потребе реализације сценарија паралелног извршавања *Mexie* система је дефинисан тип *MPIProcess*. Сваки од учесничких процеса садржи по један примерак овог типа, чији метод *run()* дефинише шта ће процес радити. Како је већ раније наведено (Одељак 6.2), извршни сценарио подразумева покретање више MPI процеса, при чему је један процес Руководилац, а остали процеси Радници. Зато постоје две специјализације типа *MPIProcess*:

- *MPIManager*, који користи процес Руководилац и
- *MPIWorker*, који користе процеси Радници.



Слика 6.9 Дијаграм класа компоненте *Brigardier*.

Класа *MPIManager* представља *QPointCalculatorI*[11] и дефинише кључне активности процеса Руководиоца, и то:

- покретање КЕ симулације;
- декомпозиција домена материјалних модела и придруживање добијених порција учесничким процесима, тј. стварање ПМИ мапе;
- подела колекције задатака добијених од КЕ модела на одговарајуће порције које се прослеђују учесничким процесима;
- сакупљање одговора учесничких процеса, паковање на одговарајући начин у јединствен одговор и слање КЕ моделу.

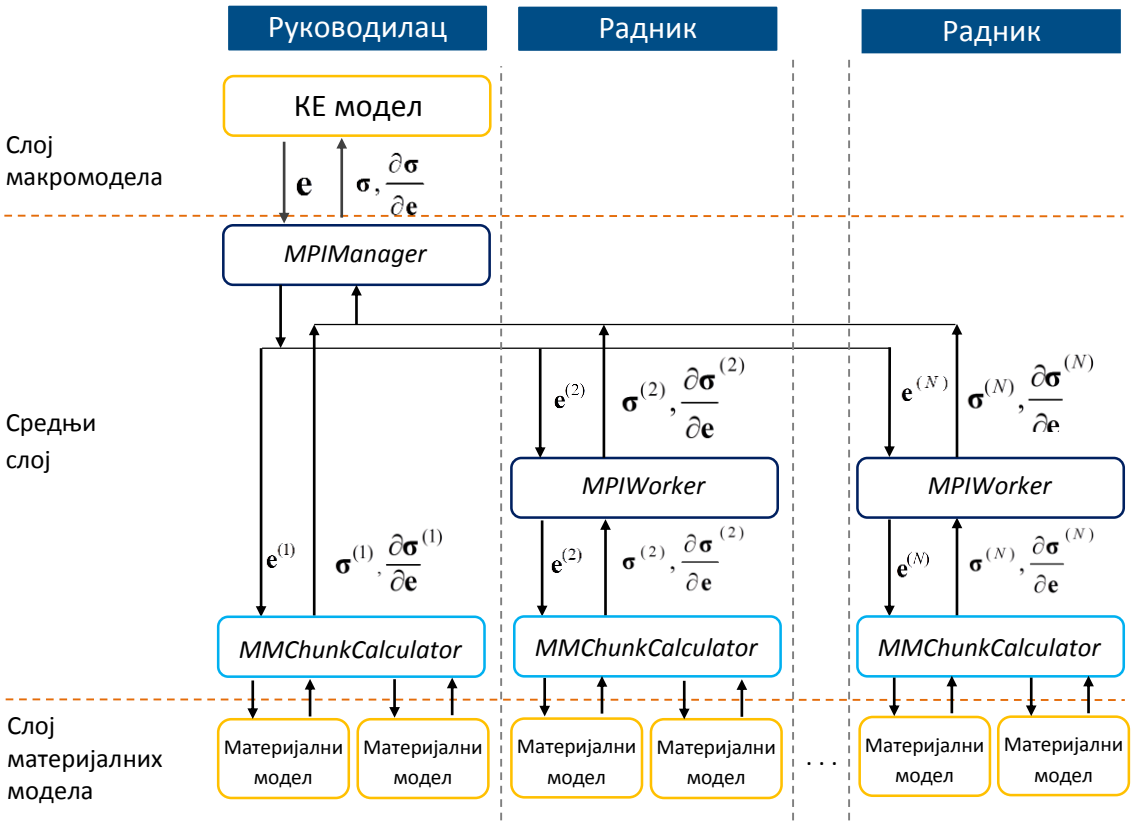
За потребе конструисања ПМИ мапе, разбијања захтева и паковања одговора према њој, *MPIManager* користи компоненту Распоређивача, којој шаље податке битне за креирање распореда материјалних модела, а касније и захтеве и одговоре који се враћају спаковани у одговарајућем редоследу.

MPIWorker класа дефинише понашање процеса Радника и задужена је само за пријем захтева упућених материјалним моделима и враћање одговора. Ова класа, а самим тим ни процес Радник, нема никакве информације о макро слоју и макромоделу.

Како и Руководилац и Радник обављају прорачуне над делом домена материјалних модела, класе *MPIManager* и *MPIWorker* садрже по једну референцу на тип *MMChunkCalculatorI*, којем се током симулације прослеђују захтеви упућени материјалним моделима.

[11] Поред класе *MPIManager*, као реализација *QPointCalculatorI* дефинисана је и класа *Sequential*. Она се користи у секвенцијалном сценарију извршавања двоскалне симулације и неће бити разматрана у даљем тексту.

Посматрано кроз призму описане тролојне архитектуре, активности процеса Руководиоца се протежу кроз сва три слоја, док процеси Радници имају активности само у средњем слоју и слоју материјалних модела (Слика 6.10).



Слика 6.10 Реализација програмског модела Руководилац-Радници

Независно од тога да ли је Руководилац или Радник, учеснички процес може као ПЕ за своје операције да користи једну ЦПЈ, тј. да буде **ЦП процес**, или да поред ЦПЈ користи и једну ГПЈ јединицу и буде **ГП-подржан процес**.

ЦП процес на месту *MMChunkCalculatorI* има примерак *MMChunkCPU* класе, која подразумева извршавање прорачуна материјалних модела искључиво на ЦПЈ јединици. *MMChunkCPU* садржи референце на примерке материјалних модела који су додељени процесу. По пријему захтева добијеног од *QPointCalculatorI*, за сваки материјални модел се упућује захтев за прорачуном одговарајућем примерку *MaterialModel* класе, којој се том приликом прослеђује и стање посматраног материјалног модела.

ГП-подржан процес користи *MMChunkCUDA* имплементацију *MMChunkCalculator*-а. *MMChunkCUDA* садржи *HuxleyCalculatorCuda*. Класа *HuxleyCalculatorCuda* извршава симулације над микромоделима и даје вредности сила и крутости активних делова мишића, датих једначинама (4.8) и (4.9). Поред симулатора микромодела, *MMChunkCUDA* садржи референце на материјалне моделе које користи за добијање вредности тензора укупног напона и извода напона у координатном сиситему влакна, датих једначинама (4.11) и (4.12).

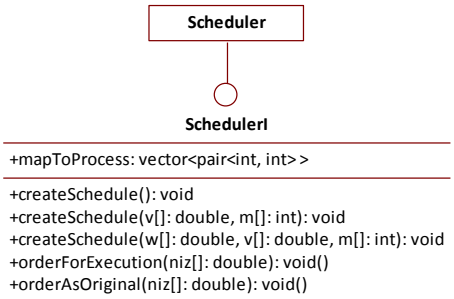
Распоређивач

Компонента Распоређивача садржи класу *Scheduler* која има задатак да дати број задатака распореди на задати број процеса, оптимизујући оптерећења, при чему узима у обзир:

- **тежинске факторе задатака** – које представљају меру рачунске комплексности,
- **брзине процеса** – изражене бројем задатака јединичне комплексности које процес може да обради у јединици времена,
- **меморијска ограничења** процеса у броју микромодела које могу чувати.

Сама класа *Scheduler* и читава компонента Распоређивача су услужне, па у њима нема информација о остатку система. Сва комуникација се своди на позиве метода:

- *createSchedule()* – служи креирању мапе распореда задатака по процесима, која у контексту *Mexie* система има значење ПМИ мапе.
- *orderForExecution()* – врши преуређивање добијеног низа елемената у редослед којим су задаци додељени процесима. Позива се при паковању података, потребних за обављање задатака, у редослед који одговара **извршном распореду.**
- *orderAsOriginal()* – врши преуређивање добијеног низа елемената из извршног распореда у **оригинални редослед задатака**. Позива се при паковању резултата задатака за враћање потражиоцу.



Слика 6.11 Компонента Распоређивача

6.4 Алгоритам распоређивања послова

Као што је већ речено, у компоненти *Scheduler* је имплементиран статички распоређивач, који има задатак да изведе иницијалну декомпозицију скупа свих микромодела на порције и додели их учесничким MPI процесима, били они везани за ЦПЈ или за ГПЈ. Политика распоређивања је прилагођена хетерогеној природи окружења у којем се извршава симулација. У покушају да пронађе што оптималнији распоред, алгоритам распоређивања узима у обзир перформансе ПЕ и капацитет меморије која је доступна сваком појединачном ПЕ. Овде је битно нагласити да се брзина рада ПЕ дефинише бројем итерација Хаксли микромодела које дати ПЕ може да изврши у јединици времена.

Циљ алгоритма јесте да процесима додели количину посла која је у складу са њиховом брзином, тако да сви процеси проведу приближно исто време у раду, чиме би се постигао одговарајући баланс оптерећења хетерогених ПЕ. Потребно је дефинисати следеће величине:

- **Очекивано време** T, које представља приближно време за које би учеснички MPI процеси требало да, према датој расподели, заврше додељене задатке.
- **Реална брзина процеса** као број итерација микромодела које процес треба да обави у јединици времена да би стигао да у времену, не већем од T, изврши њему додељене симулације. Реалне брзине су означене са $V_j, j=\overline{1,p}$, где је p укупни број MPI процеса у комуникатору.

Алгоритам подразумева следеће улазне параметре:

- **Идеалне брзине MPI процеса** изражене бројем итерација Хаксли симулације које сваки процес понаособ може да обави у јединици времена и које се као почетне вредности додељују реалним брзинама, $V_j, j=\overline{1,p}$.
- **Меморијски капацитети** ПЕ на којима се процеси извршавају, означени са $m_j, j=\overline{1,p}.$ Меморијски капацитет је одређен максималним бројем Хаксли микромодела које ПЕ може да смести у своју меморију. У случају ГП-подржаних процеса капацитет је одређен величином глобалне меморије ГПЈ, док је ЦПЈ капацитет ограничен доступном количином радне меморије, која се дели са осталим MPI процесима на датом чвору.
- **Тежински фактори микромодела,** изражени укупним бројем итерација Хаксли микромодела које се изведу током целе двоскале симулације, означени са $w_k, k=\overline{1,n}$ где је n укупан број микромодела. Ови подаци се не могу знати пре извођења саме двоскалне симулације. Ови подаци се не могу знати пре извођења саме двоскалне симулације. Међутим, уколико постоје механизми процене ових вредности, алгоритам ће их узети у обзир. У супротном се за све моделе сматра да су подједнаких рачунских тежина.

Основна идеја алгоритма је да итеративним поступком одреди **очекивано време извршења** свих симулација микромодела, T, и да у складу са њим одреди распоред микромодела по процесима. Почетна вредност $T^{(0)}$, тзв. **идеално време извршења**, се одређује на основу почетних вредности брзина $V_j^{(0)}, j=\overline{1,p}.$ Уколико се при креирању распореда за неки процес установи да му је досегнут меморијски лимит m_j, његова реална брзина се коригује спрам количине посла који му је могуће доделити. На основу нових вредности $V_j^{(i)}, j=\overline{1,p},$ се одређује ново очекивано време, $T^{(i)}$. Очекивано време се сматра коначно одређеним када се при подели не досегне меморијско ограничење ниједног процеса.

6.4.1 Детаљи алгоритма

Свака итерација i алгоритма (Слика 6.12) почиње одређивањем **очекиваног времена извршења** потребног за извођење свих симулација микромодела, $T^{(i)}$, при чему се уопште не узимају у обзир меморијска ограничења ПЕ. $T^{(i)}$ се одређује као однос укупне количине посла

мерене бројем Хаксли итерација и количине посла коју сви процеси заједно могу да обаве у јединици времена, и то као

$$T^{(i)} = \frac{\sum_{k=1}^{n} w_k}{\sum_{j=1}^{p} V_j^{(i)}}. \tag{6.1}$$

На основу добијеног времена и реалног радног оптерећења, за сваки процес се одређује **очекивано оптерећење** $L_j^{(i)}$, које представља број итерација микромодела које процес j за дато време треба да изведе, и то као:

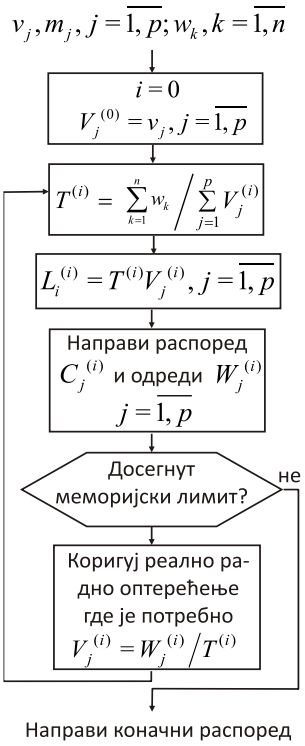
$$L_j^{(i)} = T^{(i)} \cdot V_j^{(i)}. \tag{6.2}$$

Распоређивач затим прави пробну декомпозицију домена, односно **пробну расподелу** микромодела по учесничким процесима. Распоред $C_j^{(i)}, j = \overline{1, p}$ се формира тако што ни за један процес број итерација $W_j^{(i)}$ не сме прећи очекивано оптерећење у броју итерација, $L_j^{(i)}$. По *round robin* принципу, почевши од најбржег ка најспоријем, процесима се редом додељује један по један микромодел. Микромодели се додељују идући од оних са највишим тежинским факторима ка онима са мањим тежинским факторима. Одређивање пробне расподеле се завршава када су сви модели додељени, или када више није могуће наставити поступак а да се при томе не превазиђу очекивано оптерећење и меморијски капацитети процеса. Уколико су сви микромодели додељени, распоређивање је завршено и сматра се **коначним**. У супротном алгоритам наставља са радом.

Ако распоред није коначан, прво се проверава да ли је на неком од процеса број додељених микромодела $\left|C_j^{(i)}\right|$ достигао меморијски капацитет $m_j^{(i)}$, а да при томе **укупан број итерација додељених модела** $W_j^{(i)}$ није достигао очекивано оптерећење $L_j^{(i)}$. Тада се реално радно оптерећење коригује према формули

$$V_j^{(i+1)} = \frac{W_j^{(i)}}{T^{(i)}}. \tag{6.3}$$

Прерачунавање се изводи за све процесе који испуњавају описани услов. Затим се прелази на нову итерацију, која подразумева нову вредност очекиваног времена, $T^{(i+1)}$, према коригованим брзинама.



v_j – идеална брзина j – тог процеса

m_j – меморијско ограничење j–тог процеса

w_k – тежински фактор k–тог модела

$T^{(i)}$ – очекивано време извршења у i–тој итерацији

$V_j^{(i)}$ – реална брзина j – тог процеса у i–тој итерацији

$L_j^{(i)}$ – очекивано оптерећење j – тог процеса у i–тој итерацији

$W_j^{(i)}$ – укупан број итерација модела додељених j–том процесу у i– тој итерацији

Слика 6.12 Алгоритам одређивања распореда микро модела по процесима

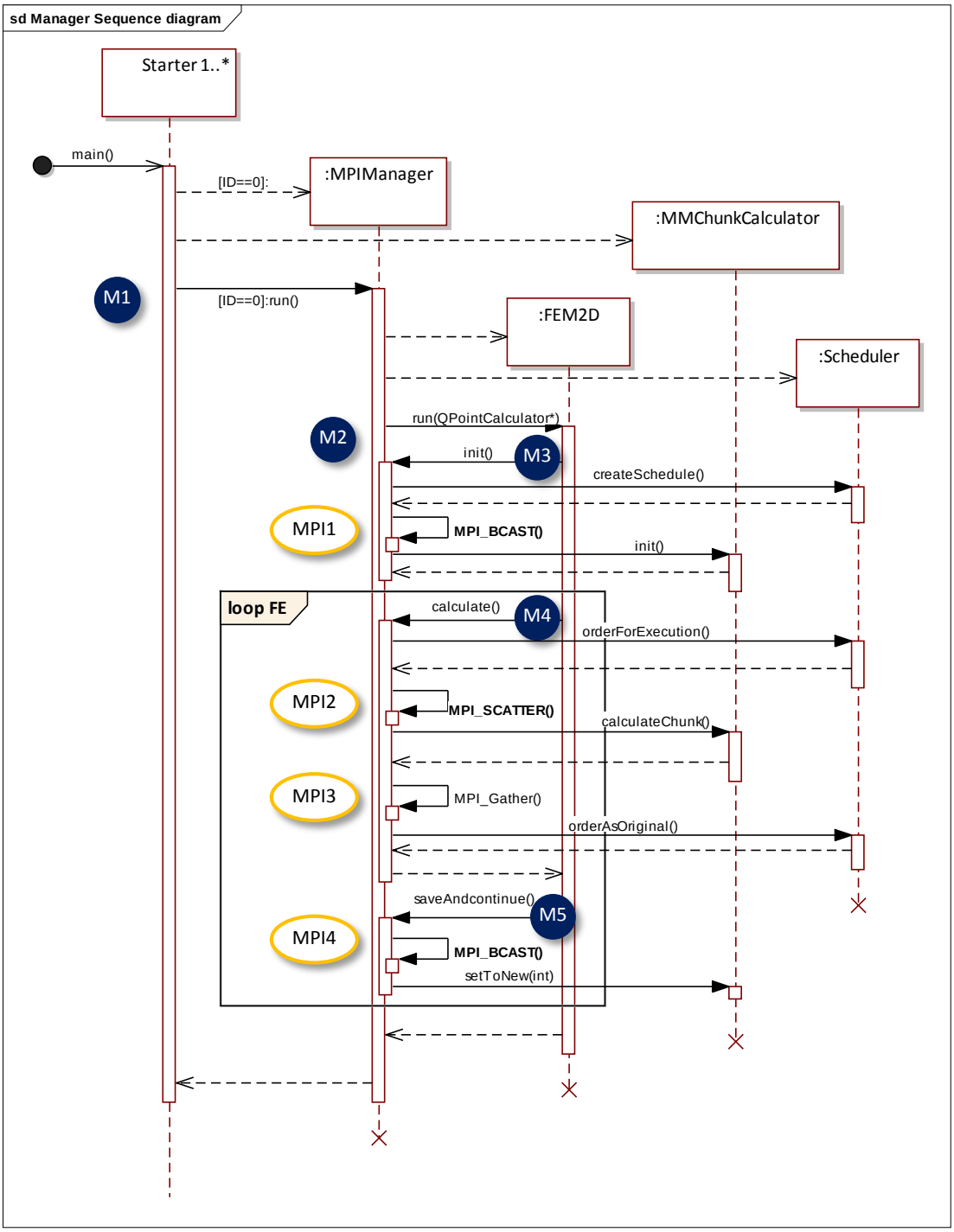
У тренутку када пробна расподела буде таква да ниједном процесу више није потребно кориговати радно оптерећење, та пробна расподела се проширује до коначне. Проширивање подразумева да се сви недодељени микромодели редом, по *round robin* принципу, распоређују процесима водећи рачуна о меморијским ограничењима.

6.5 Ток рада

У овом одељку су детаљније приказани динамички аспекти *Mexie* система. Дати су дијаграми секвенци процеса Руководиоца и Радника и описана је интеракција између скупова објеката које сваки од процеса током рада користи.

Како је већ речено, читава симулација почиње покретањем скупа MPI процеса. Сви процеси користе један комуникатор за размену порука у којем сваки има свој јединствени идентификатор (ранг процеса). Процес којем је додељен идентификатор 0 добија улогу Руководиоца, а остали улоге Радника. Информацију о томе да ли је ЦПЈ или ГП-подржан, стартовани процес чита из конфигурационог фајла, на основу чега креира одговарајући примерак *MMChunkCalcualtorI*-а.

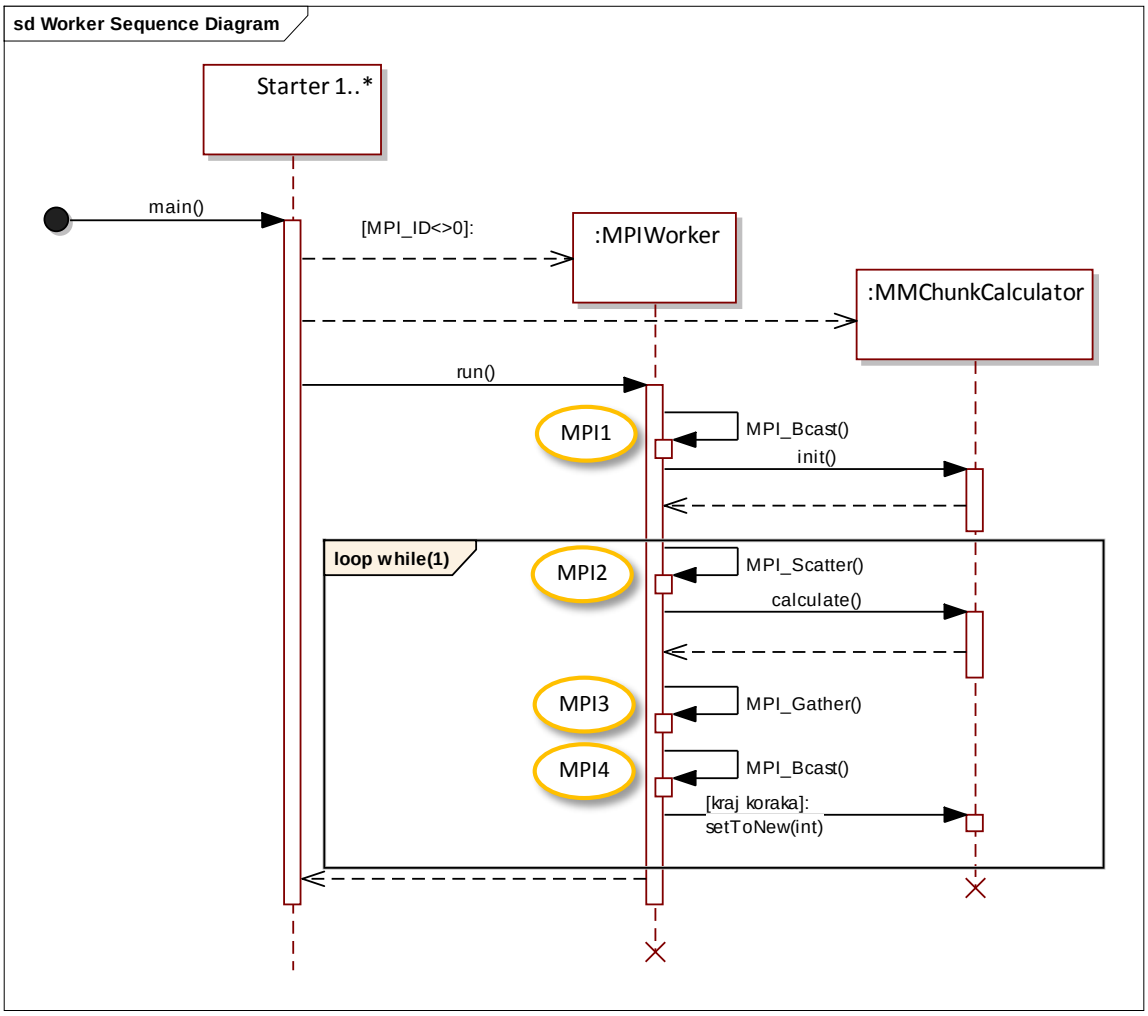
Активности Руководиоца почињу креирањем примерка класе *MPIManager* и покретањем његовог метода *run()* (Слика 6.13, **порука М1**). Тада *MPIManager* креира примерак класе *FEM* и иницира покретање двоскалне симулације слањем поруке *run()* (Слика 6.13, **порука М2**). При позиву тог метода, *MPIManager* се пријављује да, као QP*ointCaluclatorI*, одговара на захтеве из макро слоја.



Слика 6.13 Дијаграм секвенци процеса Руководиоц

Након учитавања података о макро и микромоделу и постављања КЕ мреже, *FEM* шаље податаке о броју и врстама материјалних модела *MPIManager*-у. Ови подаци су неопходни да се би се извршила иницијализација *MMChunkCalcualtorI*-а и *Scheduler*-а (Слика 6.13, **порука M3**). Дакле, у том кораку *Scheduler* одређује декомпозицију домена материјалних модела и креира ПМИ мапу. Затим се подаци о величинама порција и врстама материјалних модела деле учесничким процесима (Слика 6.13, **порука MPI1**), након чега сваки од њих иницијализује на одговарајући начин свој *MMChunkCalcualtorI*.

По обављеним иницијализацијама почиње извршавање итеративне шеме (Слика 4.13). У свакој итерацији КЕ симулације *FEM* шаље *calculate()* поруку *MPIManager*-у у чијем аргументу задаје деформације (Слика 6.13, **порука M4**). *MPIManager* се обраћа *Scheduler*-у да препакује захтеве према ПМИ мапи и да их подели учесничким процесима (Слика 6.13, **порука MPI3**). По завршетку прорачуна *MMChunkCalcualtorI*-а, чекају се одговори свих учесничких процеса (Слика 6.13, **порука MPI3**) и враћају се *FEM*-у препаковани у оригинални распоред. На крају сваке итерације поруком *saveAndContinue() FEM* обавештава Руководиоца о томе да ли је симулациони алгоритам стигао до краја, тј. има ли потребе за даљим услугама Руководиоца и Радника. Додатно, истом поруком се шаље и обавештење о томе да ли се дошло до краја временског корака, тј. има ли потребе поставити почетна стања материјалних модела за следећи времс (Слика 6.13, **порука M5**). Добијене информације *MPIManager* прослеђује учесничким процесима (Слика 6.13, **порука MPI4**).



Слика 6.14 Дијаграм секвенци процеса Радник

Активности Радника почињу креирањем примерка класе *MPIWorker* и позивањем његовог метода *run()*. Први корак Радника је да сачека податке потребне за иницијализацију *MMChunkCalcualtorI*-а (Слика 6.14, **порука MPI1**). Затим у бесконачној петљи понавља следећа три корака:

- Чека своју порцију задатака (Слика 6.14, **порука MPI2**), па их прослеђује *MMChunkCalcualtorI*-у;
- Одговор *MMChunkCalcualtorI* шаље процесу Руководиоцу (Слика 6.14, **порука MPI3**);
- Чека информацију о наставку рада и ажурирању стања материјалних модела (Слика 6.14, **порука MPI4**). Поруку о ажурирању прослеђује *MMChunkCalcualtorI*-у, а уколико је добио информацију о крају симулације, завршава са радом.

6.6 Детаљи имплементације решења

Комплетан код развијеног софтвера написан је у програмском језику C++. Као платформа током развоја софтвера коришћен је *GNU C++* компајлер, верзија 4.4.7. Коришћено је и неколико спољних библиотека и алата, и то:

- Као имплементације MPI стандарда библиотека *OpenMPI,* верзија 1.6.5;
- За CUDA кодове је коришћен *nvcc*, NVIDIA-ин сет компајлерских алата, верзија 7.0.27;
- За аутоматизацију поступка превођења и повезивања је коришћен вишеплатформски алат *CMAKE*, верзија 3.0.0;
- Документација је аутоматски генерисана алатом *Doxygen*, верзија 1.8.10;
- За дефинисање КЕ мреже као и извођење појединих делова КЕ алгоритма је коришћена *Deal.II* бибилотека, верзија 8.2;
- За униформисану манипулацију конфигурационим фајлом је коришћена библиотека *Libconfig*, верзија 1.3.2;
- Током развоја CUDA имплементације микромодела су тестиране и верзије имплементације које за паралелно извршавање векторских операције користи *Thrust C++* библиотеку.

7

Употреба феноменолошког модела у оптимизацији распоређивања

Статичка декомпозиција домена материјалних модела је техника усмерена ка обезбеђивању скалабилности развијеног решења. Њом се остварује драстично смањење мрежног саобраћаја у односу на сценарио у коме се стања материјалних модела у свакој итерацији КЕ симулације преносе процесима који их обрађују. Одлуке донете током иницијалне и једине расподеле послова имају много већи утицај на укупни учинак паралелног решења у односу на ситуацију када се врши њихово динамичко распоређивање. Да би распорела била што ефикаснија, потребно је добити што више информација о домену који се декомпонује. Модели реалних објеката, по правилу, поседују комплексну структуру сачињену од материјала различитих карактеристика који се у различитим условима различито понашају. Постојање и тачност информација о рачунској комплексности сваког дела модела могу значајно утицати на оптималност распореда који се на основу тих информација може направити. За такве информације је потребно поседовање одговарајућег знања о моделу.

У овој глави је представљена методологија стицања доменског знања о двоскалном моделу мишића, са циљем добијања података о рачунској сложености Хаксли микромодела којима

је описана микроструктура мишића [75]. Дефинисана методологија је имплементирана као самосталан алат, назван *PhenoTip*.

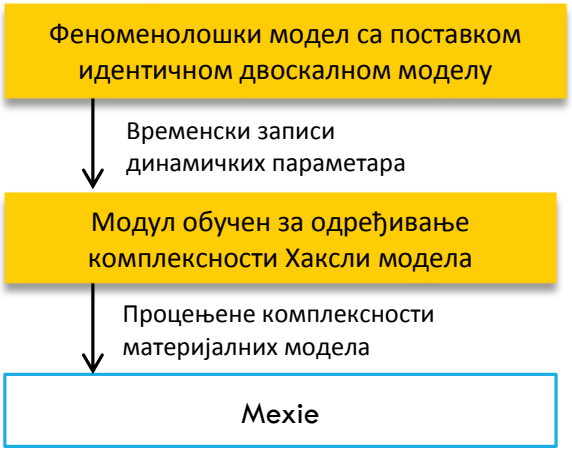
7.1 Полазне претпоставке методологије

Сложеност материјалних модела којима се описује микроструктура мишића је, како је већ речено, сведена на сложеност Хаксли микромодела који се користе за одређивање механичког одговора активних делова мишића. Она је изражена бројем итерација у симулацији микромодела и зависи од вредности параметара самог модела. Додатно, комплексност зависи и од улазних параметара симулације микромодела, чије вредности нису унапред доступне, већ се мењају током двоскалне симулације (у даљем тексту ће бити називани **динамичким параметрима**). То значи да процену комплексности модела треба вршити и на основу параметара који нису доступни пре извођења саме симулације, што је у раскораку са политиком статичког распоређивања послова. Да би се одредиле комплексности материјалних модела потребно је одредити временске записа (*time history*) динамичких параметара микро симулација. За тачно одређивање временских записа је потребно извршити целу двоскалну сималуцију. То, свакако, није решење, већ је потребно дефинисати начин на који се они могу приближно одредити.

Полаз у развоју читавог *PhenoTip* алата јесу биле претпоставке:

- да је могуће приближно одредити рачунску сложеност Хаксли микромодела на основу временских записа појединих динамичких параматара,
- да се временски записи тих параметара могу добити употребом много једноставнијег феноменолошког модела истог реалног објекта са поставком идентичном двоскалој.

План употребе алата је подразумевао обучавање алата да на основу вредности значајних параметара процени комплексност Хаксли микромодела. Пре почетка сваке двоскалне симулације, употребом феноменолошког модела се генеришу временски записи динамичких параметара, који се онда прослеђују модулу обученом за процену (Слика 7.1).



Слика 7.1 Концепт рада *PhenoTip* алата

7.2 Кораци у развоју методологије

Развој методологије коју *PhenoTip* примењује се састојао из неколико корака:

1. утврђивање који динамички параметри микромодела имају утицаја на комплексност, тј. одређивање **значајних параметара**,
2. дефинисање начина употребе феноменолошког модела мишића у одређивању временских записа значајних динамичких параметара,
3. одређивање методологије којом би се алат обучио да на основу вредности значајних параметара процени комплексност Хаксли микромодела.

7.2.1 Одређивање значајних параметара

Као прво, анализа броја итерација које један исти примерак Хаксли микромодела направи при сваком позиву појединачно се мења током двоскалне симулације. Међу вредностима које Хаксли модел користи у израчунавањима, а које се разликују при различитим позивима, налазе се следеће варијабле:

- оне које описују стање модела, вектори X и N,
- **брзина деформације**, V,
- **степен активације мишићног влакна**.

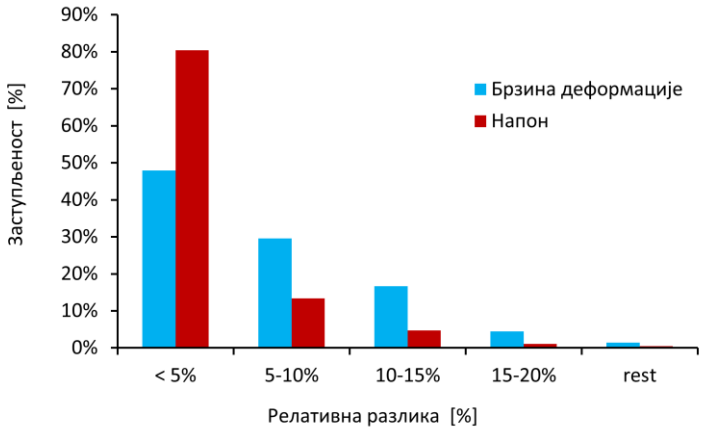
Како векторе стања модела није могуће проценити, сами по себи они нису интересантни за анализу утицаја. Зато је у анализи утицаја као фактор који осликава тренутно стање модела узет **напон** који у суштини агрегира информације ова два вектора у скалар који је лакши за анализу, а доступан је за процењивање, независно од микромодела.

Иницијална *PCA* анализа је показала да степен активације, брзина деформације и тренутни напон јесу **значајни динамички параметри**, тј. динамички параметри који имају значајан утицај на сложеност Хаксли микро симулације.

7.2.2 Употреба феноменошког модела у креирању временских записа значајних динамичких параметара

Модел који је коришћен у формирању временских записа значајних параметара на исти начин описује мишић на макро скали, употребом метода КЕ, као и описани двоскални модел, а за одређивање тренутног одговора материјала користи једноставан Хилов феноменолошки модел. Поступак одређивања временских записа параметара се састоји у томе да се из симулације КЕ-Хил модела са подешавањима идентичним КЕ-Хаксли моделу формирају временски записи напона и брзине деформације, који се бележе на крају сваког временског корака. Подаци о активацијама мишићних влакана су улазни подаци за обе симулације, па се не разматрају посебно.

Претпоставка да КЕ-Хил модел даје временске записе у довољној мери сличне онима које даје КЕ-Хаксли модел са идентичном поставком је тестирана на реалном моделу, чији је детаљан опис дат у Одељку 8.2. Слика 7.2 приказује хистограм релативне разлике одговарајућих вредности у временским записима добијених КЕ-Хил и КЕ-Хаксли моделом у односу на референтне КЕ-Хаксли вредности. Анализа је показала да је релативна грешка предвиђања вредности у 94.2% случајева мања од 10% од референтних вредности. Када је у питању брзина деформације, КЕ-Хил у 94.2% прави релативну грешку мању од 15%. Вредности релативних грешака указују на то да добијене апроксимације у великој мери прате реалне вредности, те да се могу сматрати довољно информативним за процењивање понашања микро модела.



Слика 7.2 Хистограм релативне грешке Хилових у односу на КЕ-Хаксли временске записе значајних параметара

7.2.3 Обучавање *PhenoTip* алата за процену комплексности Хаксли модела

Оспособљавање алата за процену рачунске сложености Хаксли микромодела изведено је применом једне од стандардних техника машинског учења, метод К-најближих суседа (КНС) [76,77]. Тренинг скуп су чинили подаци који су добијени извршавањем симулације над реалним моделом (Одељак 8.2). Комплетан скуп података добијен том симулацијом је садржао 1,165,056 записа облика

$$<V,\sigma,\alpha,\mathrm{I}>,$$

где је V брзина деформације добијена на основу израчунате деформације на крају временског корака и деформације из претходног временског корака, σ напон из претходног временског корака, α степен активације мишићног влакна у датој интеграционој тачки, I број итерација микромодела. Од укупног скупа записа 75% је ушло у тренинг скуп, а остатак је искоришћен као контролни скуп.

Контролни скуп је, као прво, био употребљен за одређивање оптималног броја суседа. **Укупна процентуална грешка** у процени броја итерација над целим контролним скупом је дефинисана као

$$Err[\%] = \frac{\sum_{i=1}^{n} \left| \mathrm{I}(i) - \mathrm{I}_{knn}(i) \right|}{\sum_{i=1}^{n} \mathrm{I}(i)} * 100, \tag{7.1}$$

где је $\mathrm{I}(i)$ број итерација у i-том узорку из контролног скупа, $\mathrm{I}_{knn}(i)$ број итерација које је научени алат предвидео за i-ти узорак, а n укупан број узорака у контролном скупу. Табела 7.1 приказује укупне релативне грешке у процени за различит број суседа. Уочава се да је процена на контролном скупу довољно добра за сваки од тестираних бројева суседа, јер су све грешке мање од 1%. У *PhenoTip* алату, број суседа је постављен на 3.

Табела 7.1 Укупна грешка процене за различит број суседа

Број суседа	**2**	**3**	**4**	**5**	**10**	**15**
Грешка (%)	0.530	**0.528**	0.580	0.608	0.795	0.93

Квалитет процена добијених применом КНС метода потврђује Табела 7.2. КНС за већину контролних записа даје исправне процењује исправно. У случају нетачних процена, разлика је у само 1% случајева већа од 2. Имајући у виду да је просечан број итерација на целом контролном скупу 27.8, то значи да се грешке у процени броја итерација веће од 10% дешавају у сваком стотом узорку, па се метод процене може сматрати врло успешним.

Табела 7.2 Укупна грешка процене за различит број суседа

Грешка у процењеном броју итерација	**<2**	**2**	**3**	**4**	**5**	**>5**
Густина (%)	97.50	0.97	0.48	0.30	0.20	0.03

7.3 Детаљи имплементације *PhenoTip* алата

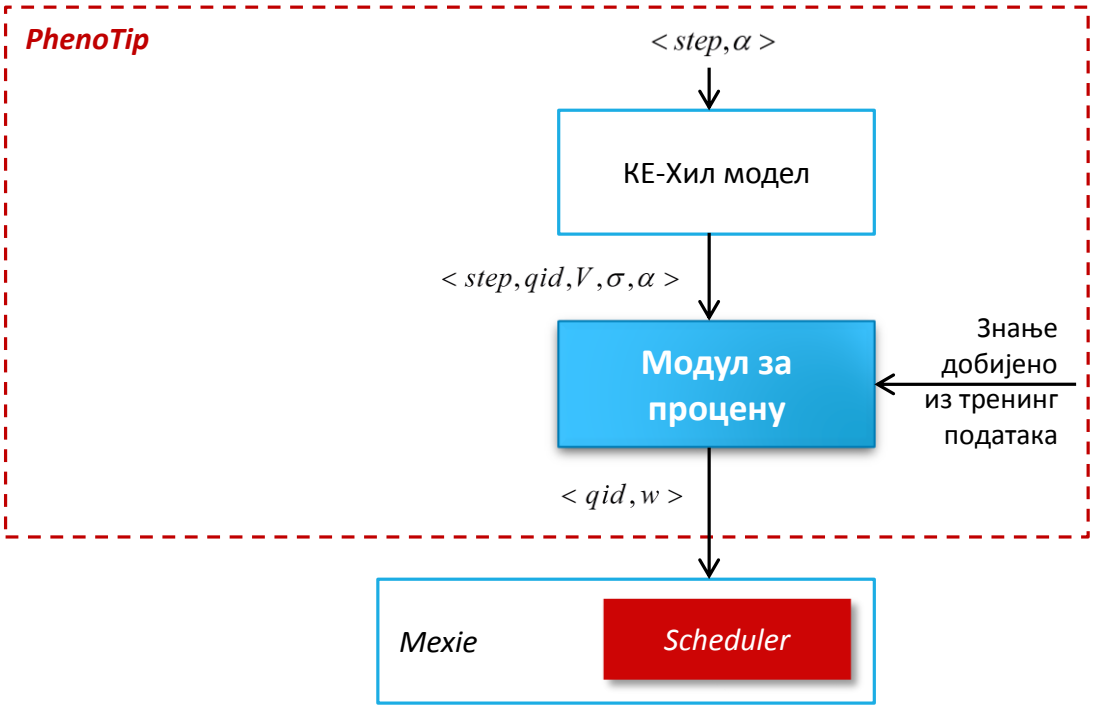
У контексту *Mexie* система, *PhenoTip* има претпроцесорску улогу. Пре покретања двоскалне симулације алат обавља свој задатак у неколико корака (Слика 7.3).

Прво поставља КЕ-Хил модел објекта који ће касније бити моделиран КЕ-Хаксли техником, а затим покреће симулацију са подешавањима која ће користити и за двоскалну симулацију. Током КЕ-Хил симулације се за сваки материјални модел, на крају сваког времеског корака креирају записи облика

$$< step, qid, V, \sigma, \alpha >,$$

где је $step$ ознака временског корака, qid ознака интеграционе тачке којој је материјални модел придружен, V брзина деформације добијена на основу израчунате деформације на

крају временског корака и деформације из претходног времеског корака, σ напон из претходног временског корака, α степен активације мишићног влакна у датој интеграционој тачки.



Слика 7.3 Шема функционисања *PhenoTip* алата

Скуп свих записа представља улаз за Модул за процену, део алата који на основу тако спакованих временских записа брзине деформације, напона и активације, за сваки материјални модел појединачно сумира процењени број итерација у сваком временском кораку.

Битно је још једном нагласити да излазни подаци не представљају процену укупног броја итерација микромодела које ће се обавити у свим итерацијама свих корака симулације КЕ модела, већ само процену колико ће се итерција микромодела обавити у последњој итерацији сваког симулационог корака КЕ модела. Зато ови подаци служе, пре свега, за оцену односа између степена комплексности које ће микромодели испољити током целокупне двоскалне симулације. Из тог разлога се као излаз из алата шаљу записи облика

$$< qid, w >,$$

где је тежина w добијена нормализацијом вредности процењеног броја итерација у свакој од интеграционих тачака. Добијене податке користи компонента Распоређивача при формирању ПМИ мапе.

Прорачун КЕ-Хил модела је обављен употребом алата за анализу методом коначних елемената PAK-S [78]. Модул за процену је реализован као скрипта коју извршава алат за статистичку обраду података R, верзија 3.2.2 [79]. Његово обучавање је спроведено употребом R-ових стандардних библиотека.

8

Резултати и дискусија

Један од главних исхода ове дисертације јесте софтверско решење које омогућава извршавање двоскалних модела скелетних мишића у хетерогеним ГПЈ/ЦПЈ окружењима. У првом делу овог поглавља приказани су резултати емпиријске анализе којом су испитиване ефикасност и скалабилност развијеног решења у различитим конфигурацијама извршног окружења. Тестирања су спроведена на идеализованим вештачким моделима који имају хомогену структуру и готово униформно понашање елемената модела током симулације.

Добијени резултати не би имали довољно велики значај за употребу решења на реалним моделима уколико би се статичком распоређивачу ускратиле информације о природи самог модела. У ту сврху је развијен *PhenoTip* алат. Прецизност алата, као и ефекат употребе података који он даје на успешност Mexie система, су анализирани и приказани у другом делу овог поглавља. Анализе су спроведене у оквиру студије случаја, где је као пример узета симулација деформације људског језика током гутања.

У свим тестирањима спроведним у сврху поменутих анализа, као хардверска платформа, је коришћен кластер који се састоји из 22 нода, опремљена са *dual Intel Xeon E5-2670@2.6GHz 8-core* процесором и 48GB меморије, са *InfiniBand QDR* мрежном инфраструктуром. Четири нода су опремљена са по једном *Tesla M2090* картицом, које располажу са по 6 GB GDDR5 меморије. Референтна јединица за сва одређивања убрзања је време потребно да се симулација која се разматра изврши у секвенцијалном маниру на *Xeon E5-2670* процесору.

8.1 Анализа перформанси платформе идеализованим моделима хомогене структуре

У овом одељку се приказују резултати анализа којима се дају одговори на следећа питања:

1. Каква је скалабилност система на хомогеним платформама?
2. Какви су комуникацијски трошкови?
3. Колико је успешно изведено прилагођавање расподеле задатака хетерогеном окружењу?

За потребе ових анализа коришћени су вештачки примери који представљају 2Д правоугаоне моделе мишића различитих густина мрежа. Коришћена су 3 тестна модела изграђена од четворочворних изопараметарских КЕ, где сваки има по четири интеграционе тачке. Карактеристике употребљених модела су следеће:

- **Модел 1** - 250 КЕ, што значи да модел подразумева 1000 примерака микромодела,
- **Модел 2** - 1000 КЕ са 4000 микромодела,
- **Модел 3** - 2500 КЕ са 10000 микромодела.

Сва три модела имају хомогену структуру, тј. структуру која у свим својим деловима има поптпуно идентичне микромоделе. Симулирано је скупљање мишића у трајању од 0.5 s, при чему су сва мишићна влакна постављена у правцу *x* осе и током читаве симулације су у пуној активацији. За извршавање ових модела није коришћен *PhenoTip* алат, већ су тежине свих материјалних модела сматране једнаким, па је Рапоређивач при расподели одлучивао само о броју модела који се додељују процесима. Таква расподела ће у даљем тексту бити називана **униформном расподелом**.

Пре тестирања перформанси развијене платформе, изведена су неопходна мерења брзина ПЕ. Као што је већ напоменуто, брзине се изражавају бројем итерација микромодела које ПЕ може да обави у јединици времена. За њихово одређивање су извођене симулације над Моделом 1 у трајању од 0.5 s. За сваки ПЕ, симулација је извршавана на једном процесу одговарајућег типа (ЦПЈ или ГП-подржан). Брзине су одређене као однос укупног времена које протекне у извршавању *calculateChunk()* метода (Слика 6.13) и укупног броја итерација Хаксли модела које се обаве током целе симулације. Као коначне, узете су вредности које представљају просек брзина добијених у десет независних извођења симулације.

Након одређивања тако изражених величина, због једноставности каснијих анализа, самом распоређивачу су достављане **релативне брзине ПЕ**. Релативна брзина ПЕ је одређена као однос брзине ПЕ и брзине ЦП језгра *Xeon E5-2670*, која је узета за референтну.

Када је у питању конкретан ГП уређај, *Tesla M2090*, показало се да је, извршавајући *HuxleyCalculatorCudaCustom* имплементацију Хаксли модела, способан да изврши око 22 пута више Хаксли итерација у поређењу са референтним *Xeon* језгром које извршава ЦПЈ имплементацију Хаксли модела. Такође, извршена су и мерења брзине извођења Хаксли симулација на истој *Tesla* картици употребом *HuxleyCalculatorCudaThrust* имплементације. У том случају, ГПЈ

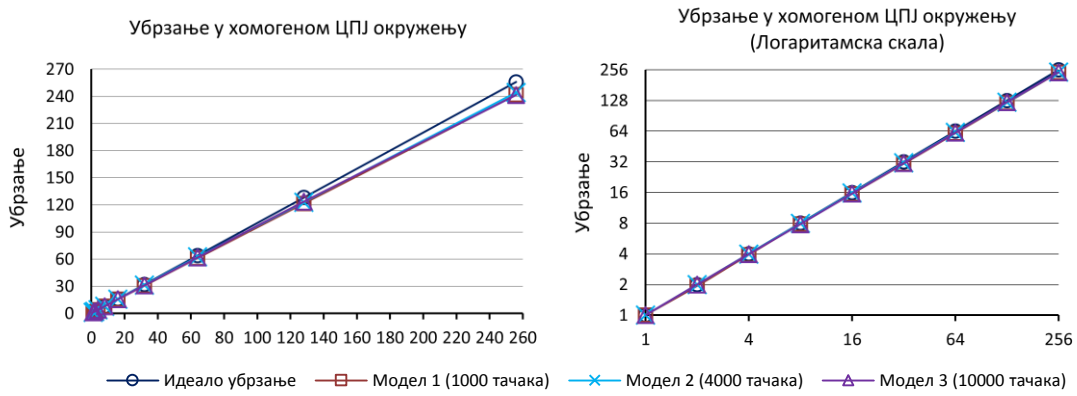
је била око 6 пута бржа од референтне ЦПЈ. Како се показала мање ефикасном, Хаксли имплементација која користи функције *Thrust* библиотеке је изостављена из даљих тестова. У свим тестовима перформанси је за релативну брзину ГП уређаја, тј. ГП-подржаног процеса који се изводи на поменутој *Tesla* картици, узета апроксимирана вредност 22.0.

Додатно, мерен је утицај пропусног опсега мреже на укупно време извођења двоскалне симулације. У случају највећег модела који је коришћен у тестовима спроведеним током истраживања, употреба стандардне гигабитне инфраструктуре у односу на *InfiniBand* је утицала на повећање укупног времена извршења у опсегу од 0.13% до 0.53%. Како је смањење пропусног опсега за 2 реда величине, што се десило у случају замене *InfiniBand* гигабитном инфраструктуром, узроковало увећање укупног времена извршења за неколико промила, даљи тестови над гигабитном мрежом нису спроведени.

8.1.1 Перформансе решења у хомогеном окружењу

Прва група тестова је изведена са циљем одређивања убрзања која се добијају на појединим врстама ПЕ, као и скалабилности решења на свакој врсти ПЕ појединачно. Анализе су рађене на сва три модела са циљем испитивања утицаја величине модела на резултате паралелизације.

Слика 8.1 показује убрзања добијена извршавањем симулације на 1, 2, 4, 8, 16, 32, 64, 128 и 256 ЦПЈ процеса. Како се може видети систем добро скалира за сва три модела. На графику десно, где је приказано убрзање на логаритамској скали, се може видети да ЦПЈ сценарио скоро идеално скалира.



Слика 8.1 Убрзања добијена извршавањем симулација над тестним моделима у хомогеном ЦПЈ окружењу

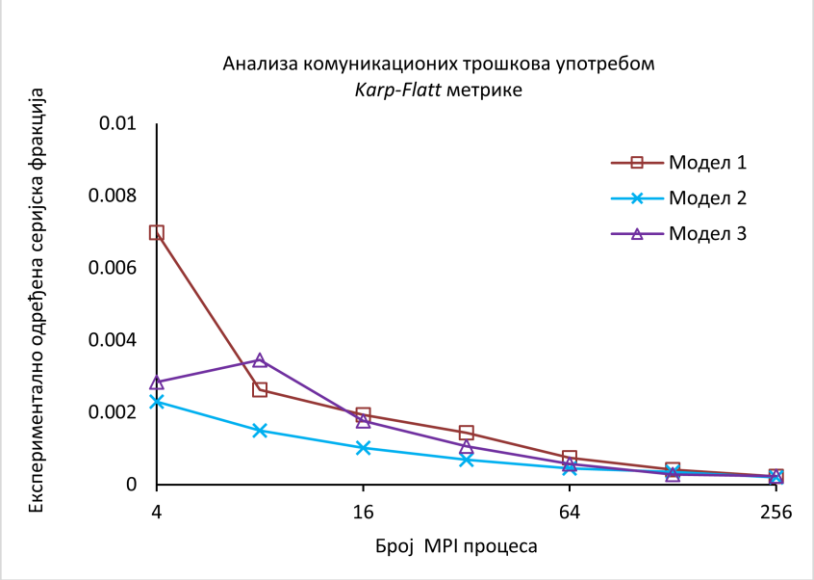
Табела 8.1 приказује времена извршавања симулација над сва три модела када је ангажован само 1 ЦП процес, као и постигнута времена и убрзања на 256 ЦПЈ. Додатно, приказане су горње границе комуникацијских трошкова у свим изведеним симулацијама, изражене у про-

центима у односу на укупно време паралелног извршења. Комуникацијски трошкови су мерени временом које је процес Руководилац провео у чекању да се изрше операције слања захтева и прикупљања података.

Табела 8.1 Резултати анализе перформанси платформе добијени извођењем симулација у хомогеном ЦПЈ окружењу

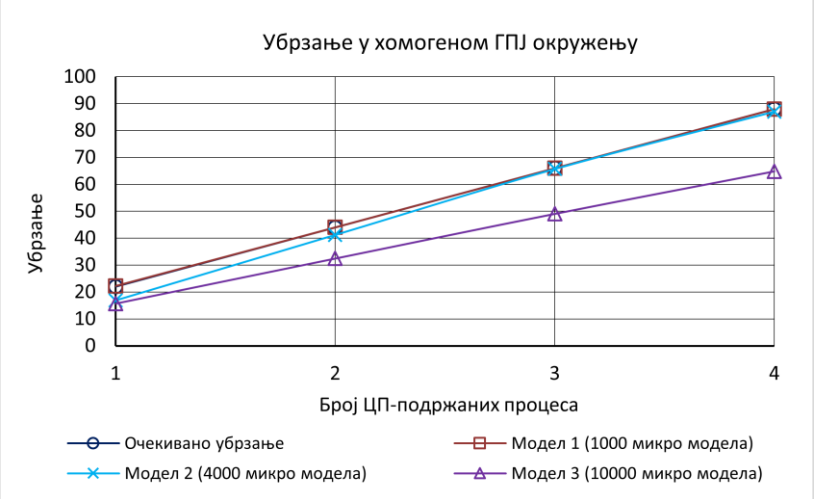
Модел	Секвенцијално време (s)	Постигнуто убрзање на 256 ЦПЈ процеса	Најбоље време (s)	Комуникацијски торшкови (%)
Модел 1	80824.2	241.80	334.25	< 0.62
Модел 2	347209.8	243.31	1427.11	< 0.81
Модел 3	880343.78	241.35	3647.58	< 0.93

Слика 8.2 приказује дијаграм експериментално одређене фракције e еграфик промене вредности серијске фракције e у зависности од броја ЦП процеса, за сва три модела. У сва три случаја вредност фракције је релативно мала и од неког броја процеса скоро константна, што говори у прилог оцени да комуникациони трошкови током извршавања симулација немају битног утицаја на опште перформансе развијене софтверске платформе.



Слика 8.2 Вредности серијских фракција при паралелном извршавању симулација над тестним моделима у хомогеном ЦПЈ окружењу

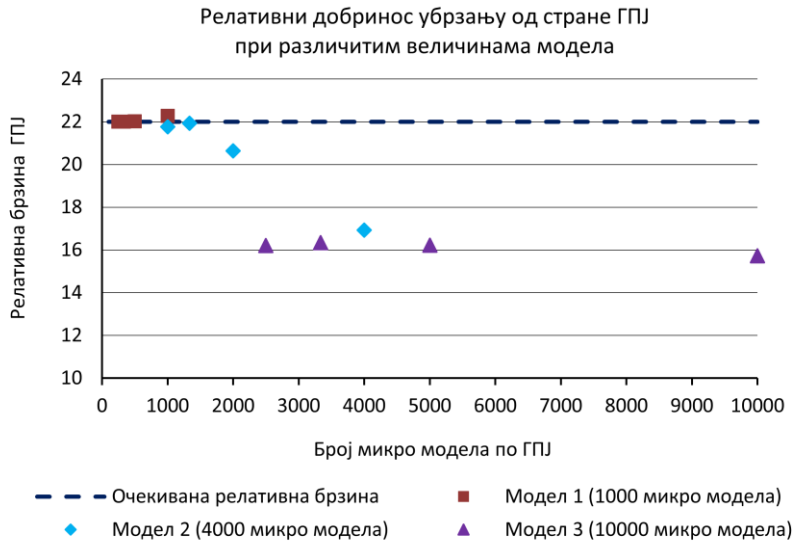
Слика 8.3 приказује убрзања која се добијају извршавањем симулација над тестним моделима када се извршавају од стране 1, 2, 3 и 4 ГП-подржана процеса. Убрзања која су добијена су поређена са вредностима које би требало очекивати имајући у виду раније наведен података о релативној брзини извршавања ГП-подржаног процеса од 22. Тако се за извршавање симулације употребом само једне ГП јединице, у идеалном случају, очекује убрзање од 22 пута у односу на секвенцијално извршавање на једном референтном процесору, а за 4 употребљене ГП јединице, 88 пута боље време.



Слика 8.3 Убрзања добијена извршавањем симулација над тестним моделима у хомогеном ГПЈ окружењу

Показано је да је убрзање за Модел 1, са 1000 Хаксли микромодела, у складу са очекиваним. Међутим, за Модел 2, са 4000 Хаскли микромодела, постоји евидентно одступање, које је нешто веће у случају када се користи само једна картица. Када је у питању модел 3, одступања су још очигледнија, као и спорији тренд раста.

Како је утицај величине модела на убрзање осетан у случају ГП-подржаних процеса, извршене су додатне анализе. Слика 8.4 приказује зависност између броја микромодела над којима ГП извршава операције и његовог ефективног доприноса укупном убрзању, где је ефективни допринос одређен као количник постигнутог убрзања са бројем ангажованих ГП јединица. На графику се може уочити да, почевши од 1000 додељених микромодела, ефективни допринос ГП почиње да пада, и то на нелинеаран начин. Док за оптерећења до 1000 тачака може рачунати на брзину од око 22, а између 1000 и 2000 брзина је око 20 са тенденцијом пада, преко 2000 нагло пада на око 16.



Слика 8.4 Релативна брзина ГПЈ у односу на ЦПЈ у завиности од броја додељених микро модела

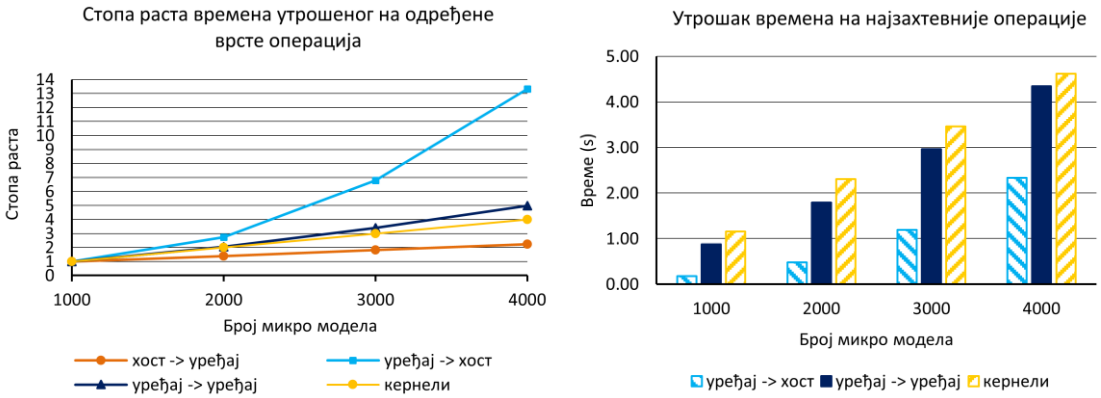
Додатним анализама и профилисањем ГПЈ Хаксли имплементације добијени су резултати који откривају узрок пада перформанси ГПЈ са повећањем броја микромодела. Број микромодела увећава количину посла коју уређај треба да обави, али се према алгоритму у једном тренутку извршавају калкулације на само једном микромоделу. Одатле би требало да следи да време проведено у кернелским функцијама треба линеарно да расте са броје микромодела, тј. да релативна брзина не би требало да опада. Међутим, у овој дискусији се занемарује утицај операција трансфера између уређаја и хоста, а како показује Табела 8.2, управо се ту налази и главни узрок пада перформанси.

Табела 8.2 Број позива операција са меморијом уређаја током извођења симулација над 1000, 2000, 3000 и 4000 микро модела

	1000 тачака	2000 тачака	3000 тачака	4000 тачака
Хост -> Уређај	1	1	1	1
Уређај -> Хост	40002	80002	120002	160002
Уређај -> Уређај	102003	204003	306003	408003

Табела 8.2 показује број позива свих врста операција са меморијом уређаја када уређај оперише са 1000, 2000, 3000 и 4000 микромодела. Може се уочити да бројност операција копирања са уређаја на уређај и са уређаја на хост линеарно расте са бројем модела. Разлог оволиком броју копирања са уређаја на хост је што у њих спадају не само финална копирања вектора напона и крутости (којих увек има исти број), већ и копирања која се одвијају у оквиру ажурирања вредности вектора *X* и *N* (Одељак 6.3.2), чији број расте са бројем микромодела. Иако број ових операција линеарно расте са порастом броја микромодела, времена утрошена на те исте операције не расту уједначеним темпом (Слика 8.5). На графику лево се може видети да операције копирања са уређаја на хост имају тенденцију значајног нелинеарног раста

са повећањем броја микромодела. За 4000 микромодела, време утрошено на њихово извршење је 14 пута веће од времена утрошеног на те исте операције у раду са 1000 микромодела. Очигледно, узрок промене перформанси уређаја при чешћим преносима података са уређаја на хост морају имати везе се специфичностима самог ГП уређаја. Један од могућих разлога за посебно понашање може бити тај што се током ажурирања вредности X и N врши неколико копирања малих порција података. Позиви су дискретни и не могу се спојити у мањи број позива за веће порције података, па CUDA компајлер нема механизам оптимизације за такве позиве. Са повећањем њиховог броја, повећава се и негативни утицај на опште перформансе. Операције копирања између уређаја и хоста су временски скупе, неоптимизоване и очигледно смањују опште перформансе уређаја.



Слика 8.5 Анализа раста времена утрошеног на најзахтевније операције ГПЈ Хаксли имплементације са растом броја Хаксли микро модела. График лево показује стопу раста у односу на времена трајања одговарајућих операција на 1000 микро модела. График десно показује времена утрошена на најзахтевније операције током извођења по једне симулације у трајању од 5ms над сваким микро моделом.

Табела 8.3 приказује најбоља времена извршавања симулације постигнута на четири ГП-подржана процеса, као и горње границе комуникацијских трошкова у свим изведеним симулацијама.

Табела 8.3 Резултати добијени извођењем симулација у хомогеном ГП окружењу

Модел	Секвенцијално време (s)	Постигнуто убрзање на 256 ЦПЈ процеса	Најбоље време (s)	Комуникацијски торшкови (%)
Модел 1	80824.2	88.00	918.41	< 0.003
Модел 2	347209.8	87.08	3986.92	< 0.3
Модел 3	880343.78	64.84	13576.71	< 1.2

Без обзира на додатна разматрања везана за ГП-подржане процесе, на основу показаних резултатата може се закључити да у хомогеним окружењима софтверско решење чврсто скалира и даје скоро идеална убрзања. Овде треба имати у виду да се идеална убрзања ПЕ пореде са очекивањима одређеним на основу њихове релативне брзине.

Снажно скалирање у хомогеним окружењима значи да би са исправно постављеном политиком расподеле послова у хетерогеним окружењима резултати требали да буду подједнако добри.

8.1.2 Перформансе платформе у хетерогеном окружењу

У овом одељку су приказани резултати мерења убрзања која су постигнута извођењем симулација у хетерогеном ЦПЈ/ГПЈ окружењу. Циљ изведених тестова је провера успешности оптимизације оптерећења учесничких процеса у складу са релативним брзинама и меморијским ограничењима ПЕ на којима се извршавају.

Анализа перформанси на уређајима различитих брзина

У идеалном случају непостојања меморијских ограничења, оптимизација расподеле задатака се сматра успешном уколико сваки ПЕ може да замени онолико референтних ЦПЈ колика му је релативна брзина, а да се при томе постигне једнако убрзање.

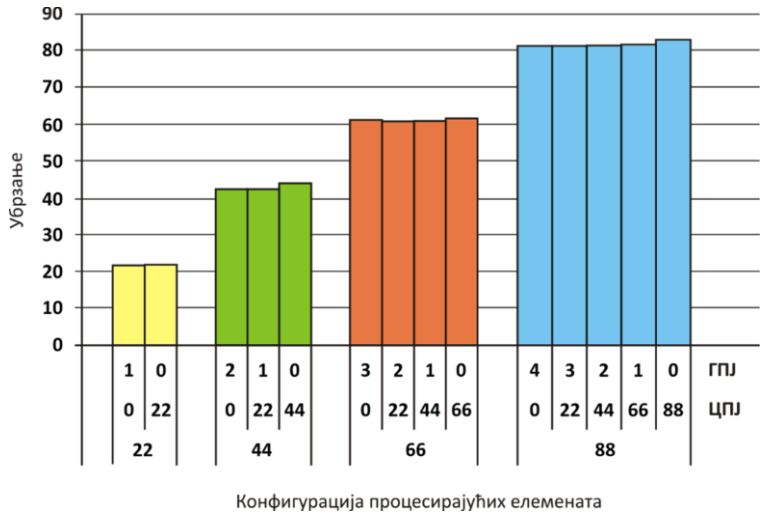
Анализа је изведена тако што су мерена убрзања симулације над **Моделом 1** на различитим конфигурацијама, а затим поређена убрзања добијена на конфигурацијама које имају једнаке укупне брзине ПЕ. Дефинисане су групе конфигурација чије су брзине еквивалентне брзинама 22, 44, 66, односно 88 референтних ЦПЈ. Конфигурације по групама су:

- Група 22: 1Г/0Ц и 0Г/22Ц,
- Група 44: 2Г/0Ц, 1Г/22Ц и 0Г/44Ц,
- Група 66: 3Г/0Ц, 2Г/22Ц, 1Г/44Ц и 0Г/66Ц,
- Група 88: 4Г/0Ц, 3Г/22Ц, 2Г/44Ц, 1Г/66Ц и 0Г/88Ц,

где запис *n*Г/*m*Ц означава да је у тестној конфигурацији употребљено *n* ГПЈ и *m* ЦПЈ.

Слика 8.6 приказује вредности добијених убрзања. Секвенцијално време извршавања тестног примера износи 80824.2s. Укупна времена извршавања за различите конфигурације се крећу у опсегу од 993.579 s за конфигурацију 0Г/88Ц до 3693.79 s за конфигурацију 1Г/0Ц.

На графику се може видети да су убрзања остварена конфигурацијама које припадају истом рачунском капацитету приближно иста, па се може закључити да је да сам алгоритам расподеле добро усклађује оптерећења процеса спрам брзина ПЕ на којима се извршавају.

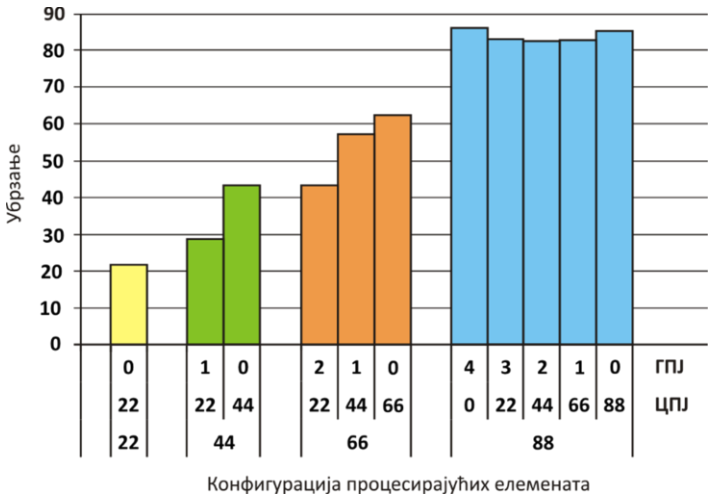


Слика 8.6 Убрзања добијена извођењем симулације у трајању од 0.5s над Моделом 1 (1000 интеграционих тачака/микро модела)

Анализа перформанси на уређајима различитих брзина и меморијских капацитета

Меморијска ограничења се изражавају бројем Хаксли микромодела које један ПЕ може да смести у припадајућу меморију. Сваки ПЕ има на располагању коначан меморијски простор. У зависности од врсте уређаја и природе проблема, попуњавање меморијских капацитета је спорије или брже остварив. Са друге стране, од користи може бити и вештачко наметање меморисјких ограничења, рецимо у случају када неки ПЕ елемент показује значајан пад у перформансама у тренутку када број додељених микромодела пређе одређену границу.

Како је раније објашњено (Одељак 6.4), алгоритам распоређивања је дефинисан тако да, уколико очекивано оптерећење неког ПЕ превазилази његове меморијске капацитете, изврши корекцију релативне брзине ПЕ и коригује оптерећења у складу са тим. У сврху анализе резултата расподеле која узима у обзир меморијска ограничења, извршена су тестирања у ЦПЈ/ГПЈ окружењу где је за ГПЈ дефинисано ограничење од 1000 микромодела, што је оправдано ако се жели максимално искоришћење ГПЈ. Слика 8.7 приказује вредности убрзања добијених на Моделу 2 (4000 интеграционих тачака). Секвенцијално време извршавања тестног примера износи 347209.83 s. Тестови су спроведени на конфигурацијама које су описане у претходном одељку. Због наметнутих ограничења меморијског капацитета ГПЈ, из теста су изостављене конфигурације 1Г/0Ц, 2Г/0Ц и 3Г/0Ц. Укупна времена извршавања за различите конфигурације се крећу у опсегу од 3936.92 s за конфигурацију 0Г/88Ц до 15788.25 s за конфигурацију 0Г/22Ц.



Слика 8.7 Убрзања добијена извођењем симулације у трајању од 0.5s над Моделом 2 (4000 интеграционих тачака/микро модела)

На графику (Слика 8.7) се може уочити да у конфигурацијама које одговарају рачунским капацитетима 44 и 66 допринос ГПЈ није приближно еквивалентан доприносу 22 ЦПЈ. У случају 2Г/22Ц убрзање је приближно једнако убрзању добијеном на конфигурацији 0Г/44Ц, што значи да је допринос ГПЈ приближно једнак доприносу 11 ЦПЈ. Такав однос је оправдан чињеницом да је 1000 Хаксли микромодела 11 пута веће од броја који се у конфурацији 0Г/44Ц додељује ЦПЈ процесу. То даље говори у прилог претпоставци да алгоритам распоређивања успешно балансира оптерећења ПЕ и када постоје ограничења у меморијским капацитетима.

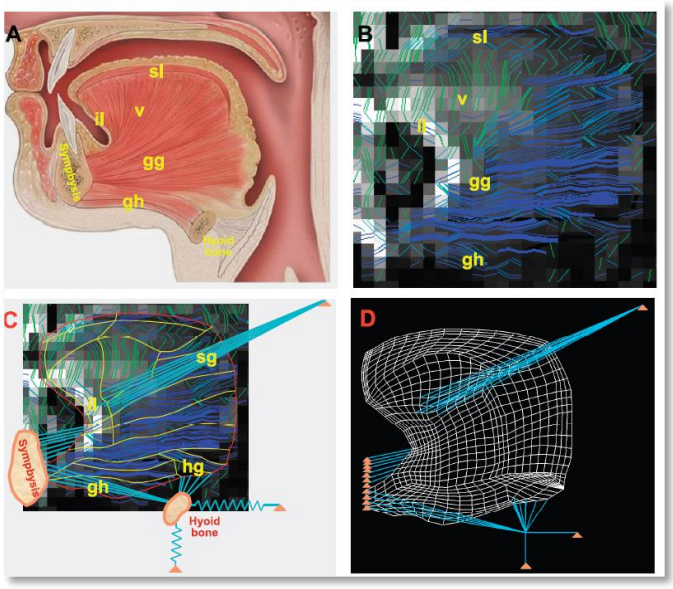
8.2 Студија случаја на моделу деформације језика - анализа успешности *PhenoTip* методологије и њен значај за реалне перформансе *Mexie* платформе

До сада приказани резулатати су део емпиријске студије извршене са циљем одређивања могућности *Mexie* система када се примењује на моделе хомогене структуре, тј. моделе у којима је комплексност микромодела једнака.

У овом одељку ће бити приказани резултати анализе перформанси система испитаних у студији случаја, за коју је одабрана симулација деформације људског језика током гутања. Језик као орган састављен од осам мишићних група има сложену структуру и способност за велики број варијација облика и грчења током говора и гутања [80]. Имајући у виду те карактеристике, модел је погодан за тестирање перформанси *Mexie* система са и без употребе *PhenoTip* алата.

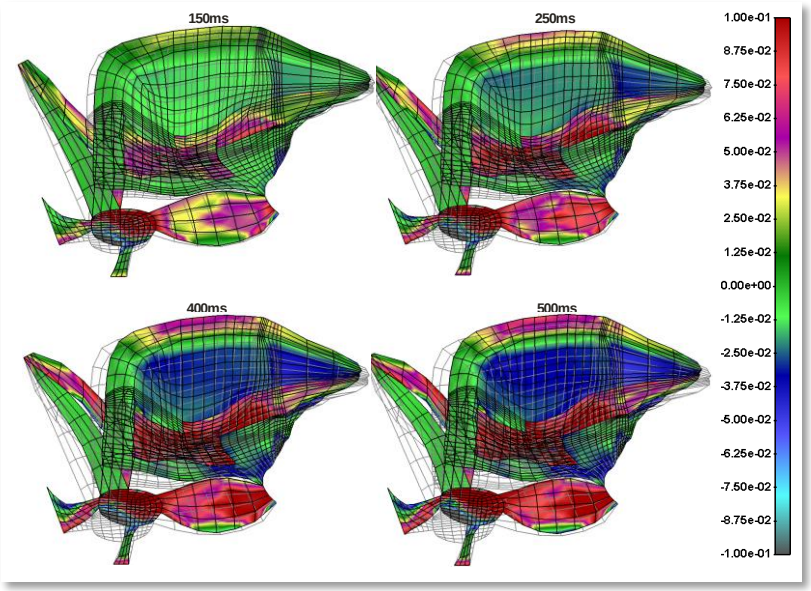
Модел језика

Језик представља мишићни орган сложене структуре. Његову микроархитектуру чини низ различито постављених и испреплетаних унутрашњих и спољашњих мишића. За студију случаја дефинисана је дводимензиона КЕ мрежа којом је апроксимирана анатомија језика. Мрежа је изграђена на основу трактографске слике извдене из DTI снимка, где су тродимензиони правци мишићних влакана пројектовани на раван којој припрада средишњи вертикални пресек језика (Слика 8.8). Захваљујући снимцима, одређене су групе једнако оријентисаних снопова мишићних влакана [40], а тиме је омогућено и везивање физиолошких атрибута појединачних и везаних миоцита, као и читавих миоцитних регија током симулације гутања. Процес гутања је симулиран употребом различитих активационих функција на различитим групама КЕ.



Слика 8.8 Развој мреже КЕ на основу дифузионих MRI снимака људског језика

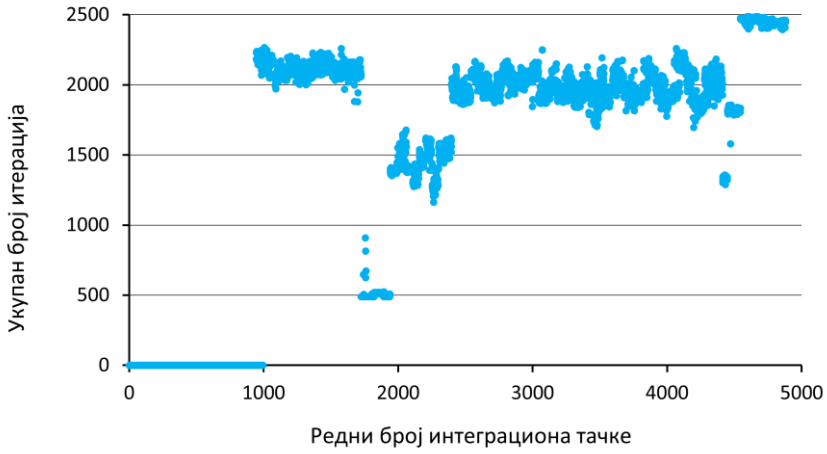
Модел се састоји из 1238 коначних елемената, са укупно 4952 интеграционе тачке. Симулирана је промена облика језика која се јавља при контакту врха језика са тврдим непцем током гутања, у трајању од 0.5 s подељених у 50 временских корака (Слика 8.9).



Слика 8.9 Симулација деформације језика при контакту са тврдим непцем током гутања.

Анализа реалне комплексности микромодела

Одређивањем укупног броја Хаксли итерација које обави сваки од микромодела придружених интеграционим тачкама током целе симулације добијен је велики распон вредности. Вредности се крећу од 0 у случају елемената који се не активирају током целе симулације, или припадају делу језика у коме се искључиво налази везивно ткиво, до око 2500 итерација. На графику (Слика 8.10) се издваја шест већих група микромодела према показаној укупној рачунској копмлексности.

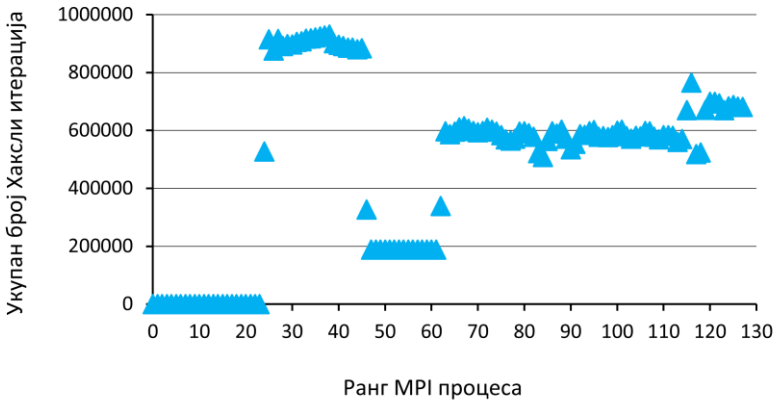


Слика 8.10 Укупан број итерација по интеграционим тачкама

Резултат декомпозиције домена по униформном алгоритму

У одсуству знања о сложености микромодела, Распоређивач ради по униформном алгоритму, што резултира неравноменим оптерећењем учесничких процеса. У сличају расподеле на 128

ЦПЈ истих брзина и меморијских капацитета, од укупно 61977063 Хаксли итерација које се обаве током читаве симулације, 24 процеса не изврши ни једну (Слика 8.11). Међу процесима који имају активности Хаксли модела, постоји изражена неравнотежа, па тако најоптерећенији процеси обављају до пет пута више послова од најмање оптерећених.

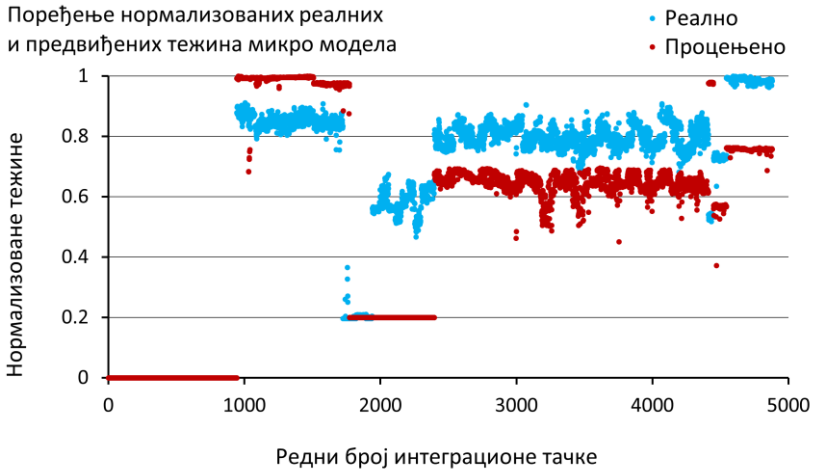


Слика 8.11 Оптерећења 128 ЦПЈ процеса при униформној расподели

Анализа прецизности *PhenoTip* алата

Како је већ детаљно описано у седмом поглављу, из КЕ-Хил симулације се добијају временски записи напона и брзине деформације за сваку интеграциону тачку. На основу тих података Модул за процену даје процену укупног броја итерација које ће се у сваком микромоделу обавити.

Слика 8.12 приказује распоред нормализованих вредности укупног броја итерација које су се заиста обавиле током симулације и броја који предвиђа *PhenoTip* алат по свакој интеграционој тачки/микромоделу.



Слика 8.12 Поређење нормализованих вредности укупног броја итерација које су се заиста обавиле током симулације и броја који предвиђа *PhenoTip* алат по свакој интеграционој тачки/микро моделу

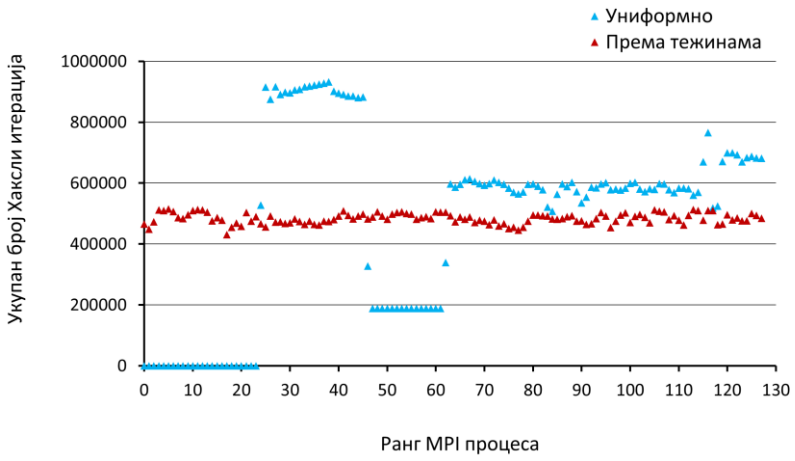
Може се уочити да предвиђене рачунске комплексности делимично прате распореде оптерећења. Одступања реалних од процењених вредности које алат даје су очигледна, иако је на тестном скупу КНС метод који Модул за процену користи показао велику прецизност (Одељак 7.2.3). Узрок одступањима лежи у двема чињеницама:

- *PhenoTip* даје процену рачунске комплексности микро модела у интеграционим тачкама само на основу предвиђања броја итерација које ће се у сваком микро моделу обавити на крају КЕ временског корака. Да би се постигла већа прецизност Модул за процену би морао добити одговоре и на следећа питања:
  - Колико пута у сваком временском кораку КЕ алгоритам итерише да би стигао до испуњења услова конвергенције?
  - Које су вредности брзина деформација и напона у свакој итерацији КЕ алгортима?
- Брзине деформација и напони у интеграционим тачкама добијени из КЕ-Хил симулација су приближно, а не у потпуности, једнаке вредностима које се добијају КЕ-Хаксли симулацијом.

Упркос непрецизностима, резултати показују да се *PhenoTip* алатом релативно добро одвајају веће групе тачака према комплексности.

Резултати декомпозиције изведене на основу предвиђених комплексности

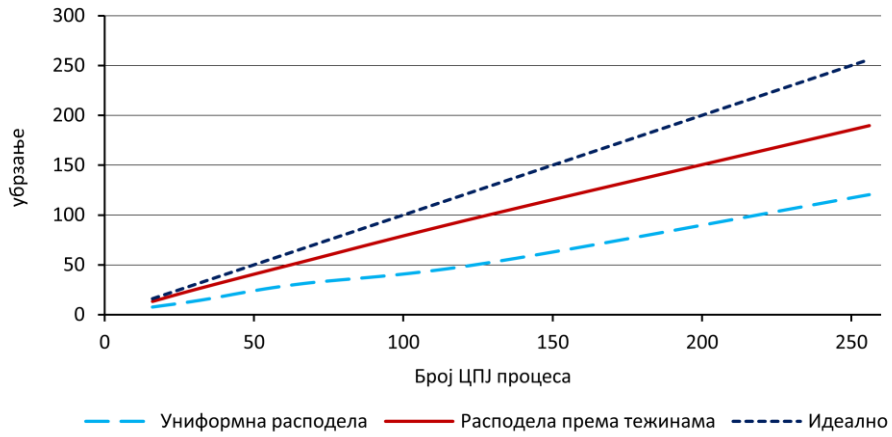
График (Слика 8.13) приказује упоредо резлултате униформне и "тежинске" расподеле на 128 ЦПЈ истих брзина и меморијских капацитета. Очигледно, знање о домену декомпозиције добијено од *PhenoTip* алата предато Распоређивачу резултира драстично уравнотеженијим оптерећењем учесничких процеса. Број итерација које обави један процес при таквој расподели се налази у интервалу од 430 610 до 514 810. Просечан број обављених итерација је приближно једнак 484 196, што је 0.7% веће од броја итерација које би требао да обави сваки од процеса при иделно равномерној расподели 480 487.



Слика 8.13 Поређења опетерећења 128 ЦПЈ процеса при униформној и при расподели која узима у обзир предвиђања *PhenoTip* алата

Анализа убрзања

Слика 8.13 приказује упоредо убрзања добијена извршавањем симулације на 16, 32, 64, 128, 256 ЦПЈ, при униформној и "тежинској" расподели микромодела. Показало се да употреба на описан начин добијеног доменског знања разултира 40% повећаним убрзањем. Укупно време трајања симулације извршаване у секвенцијалном маниру је приближно 65.5 сати, време извршења на 256 ГПЈ при униформној расподели је приближно 32 минута, а при "тежинској" око 20 минута.



Слика 8.14 Дијаграм убрзања

8.3 Дискусија постигнутих резултата

На основу приказаних резултата се може закључити да примењена методологија статичке декомпозиције домена микромодела омогућава достизање убрзања од два реда величине у односу на секвенцијално извођење двоскалне симулације. То заправо значи да се симулација која би на једној ЦПЈ трајала 10 дана може извести за непун сат. Приказани резултати недвосмислено показују да је време извршења комплексног мишићног модела могуће свести у оквире који га чине употребљивим за реално корисне анализе. Поред тога, убрзања која се постижу употребом ГПЈ за извршавање прорачуна Хаксли микромодела резултирају значајним унапређењем односа цена/перформансе, што је велики корак ка увођењу вишескалних модела мишића у софтвере намењене свакодневној клиничкој пракси.

Специјализовани алгоритам расподеле је показао да добро усклађује расподелу послова са хетерогеном конфигурацијом хардверског окружења. Да би се расподелом постигли приказани резултати, потребно је да пре извођења симулације буду познате:

- конфигурација хардверског окружења, тј. колико којих ПЕ елемената може бити употребљено,
- карактеристике ПЕ (брзина и меморијски капацитет).

Од доступности података о конфигурацији и прецизности информација о брзинама зависи ефикасност целог система. У случају да такви подаци не постоје, извршно окружење се сматра хомогеним, брзине јединичним, а сви процеси се покрећу као процеси ЦПЈ типа, што у хетерогеним окружењима не може дати оптимално искоришћење ресурса.

Иако је на вештачим примерима платформа показала велику ефикасност, добијена убрзања није могуће у потпуности достићи употребом у симулацијама реалних модела. Наиме, када су реални модели у питању, показало се да је за потпуно искоришћење могућности система неопходно знање о домену микромодела. Анализа алата за екстракцију потребног знања је са једне стране показала његове несавршености. Ипак, његовом употребом је било могуће извести неку врсту кластеризације, издвајањем већих група микромодела приближно истих комплексности. Таква информација о релативним односима комплексности микромодела је била довољна да се постигну значајна побољшања перформанси целог решења.

9

Закључна разматрања о употребним вредностима и могућим побољшањима

Свеобухватан и ефикасан модел скелетних мишића је моћан алат у истраживачким активностима. *In silico* анализе омогућавају јефтиније и брже тестирање и евалуацију хипотеза. Њима се могу идентификовати важни аспекти или корелације које захтевају даља испитивања. Детаљан модел мишића мора обухватити процесе и структурне елементе на више скала. Ограничавајући фактор у развоју и употреби вишескалних модела мишића јесте њихова велика сложеност. Њихова практична употреба је могућа само ако се симулације извршавају применом техника рачунарства високих перформанси и у одговајућем окружењу.

До тренутка писања овог текста, најдетаљнији модел скелетних мишића, описан у [53], контролисано уводи биомеханичке микромоделе, и користећи технику хомогенизације, одређује утицај процеса са микро скале на одзив мишића на макро скали. Да би обезбедили практичну употребљивост, за дефинисање модела и извођење симулација аутори су користили софтвески оквир *OpenCMISS* и његову подршку дистрибуираном извршавању уз употребу MPI стандарда. Паралелизовали су извршавање прорачуна домена микромодела на који су применили статичку декомпозицију. Резултати анализе убрзања на 1, 2 и 4 процесора делују

обећавајуће у смислу скалабилности. И под претпоставком да је добро скалирање глобални тренд, овакав вид убрзавања прорачуна није довољан за ширу употребу описаног модела.

Да би вишескалне симулације мишића могле да пруже употребљиве резултате у временском оквиру прихватљивом за апликације које би се користиле у клиничкој пракси, убрзања морају бити веома висока. Додатно, велику улогу има и однос цена/перфомансе, који мора бити разуман. Скупа опрема великих супер-рачунара није увек доступна, а однос утрошене енергије и учинка није довољно низак.

Главни циљ ове дисертације је дефинисање методологије којом се у великој мери решава проблем утицаја рапидног увећања сложености модела на његову ефикасност и употребљивост. Неколико основних идеја представљају основу за достизање постављеног циља:

1. Детаљан модел мишића захтева масивне прорачуне великог броја микромодела. Убрзавање тог дела вишескалне симулације може обезбедити убрзање читаве симулације за два реда величине.

2. Имајући у виду да су нумеричке симулације физиолошких процеса по својој природи погодне за приступ паралелизма података, реално је очекивати да ће употреба ГПЈ у убрзавању прорачуна уз стандардне ЦПЈ резултирати значајним убрзањем и унапређењем скалабилности.

3. Развој хибридног програмског модела за извршавање симулација у хетерогеном рачунарском окружењу омогућава флексибилност и могућност већег искоришћења расположивих рачунарских ресурса.

4. Оптимизација расподеле послова у складу са брзинама и меморијским ограничењима игра пресудну улогу у ефикасности софтверских платформи које подразумевају хетерогена извршна окружења.

5. Униформна расподела послова по процесима не даје добре резулатате када су микромодели у питању. Проценом комплексности сваког микромодела појединачно се могу постићи значајна побољшања у декомпозицији домена микромодела, па последично и бољи резултати у убрзавању.

6. Екстракција знања о конкретном домену микромодела је могућа употребом техника машинског учења и једноставнег, рачунски далеко мање захтевног, феноменолошког модела мишића.

Пратећи наведене идеје, развијена је методологија, а према њој и софтверска платформа за дистрибуирано и паралелно извршавање двоскалних симулација динамике мишића. Методологија је дефинисана и тестирана на двоскалном моделу скелетних мишића, где је на макро скали мишић моделиран методом коначних елемената, а механичке карактеристике материјала у интеграционим тачкама се одређују молекуларним моделом мишићне контракције.

9.1 Постигнути резултати

Главни исход дисертације је софтверска палатформа, названа ***Mexie***, која представља проширив и скалабилно софтверско решење изграђено на MPI и CUDA програмским моделима. У овом тренутку је постављена да подржи извршавање симулација на произвољном броју ЦП и ГП јединица, али је, захваљујући модуларној архитектури, отворена за једноставно проширивање модулима који би омогућили употребу додатних програмских модела и рачунарских ресурса високих перформанси.

Кључне новине представљене платформе у односу на постојећа решења паралелизације вишескалних модела мишића су:

1. Употреба ГПЈ у прорачунима микромодела и дефинисање хибридног програмског модела за ефикасно извођење симулација у окружењу са произвољним броју ГПЈ/ЦПЈ;

2. Оптимизован алгоритам статичке расподеле задатака прилагођен за примену на хетерогеним извршним окружењима и двоскалним моделима нехомогене струкуре, у смислу комплексности израчунавања микромодела.

Још један, важан исход овог рада јесте *PhenoTip* алат, који примењује потпуно нову методологију за процену комплексности микромодела. Кључну новину у методологији представља формирање временских записа динамичких параметара микромодела употребом Хиловог феноменолошког модела. Сам алат на основу добијених записа и знања о микромоделима створеног техникама машинског учења даје процену комплексности израчунавања у свакој интеграционој тачки макромодела.

Софтверска платформа је у тестовима показала чврсто скалирање и убрзања од чак два реда величине. Са таквим карактеристикама представља одличну основу и окружење за даљи развој вишескалних модела мишића и истраживања у области физиологије мишића, што је и био примарни циљ рада у оквиру ове дисертације. Упошљавањем ГП јединица је обезбеђено веће убразање, али и драстично поправљање односа цена/перформансе, али и утрошена снага/перформансе. Тиме је отворен пут ка развоју софтверских решења намењених клиничкој пракси.

9.2 Смернице за даља истраживања

Иако су постављени циљеви дисертације испуњени, простора за даља истраживања и унапређење система паралелизације вишескалних модела мишића свакако има, и то у неколико праваца:

- Алгоритам расподеле би могао да се измени тако да брзину ПЕ прихвати као функцију меморијских оптерећења, што би у случају приказане ГП имплементације решило проблем неусклађености између очекиване и постигнуте брзине.
- ГП имплементација Хаксли модела се може боље оптимизовати додатним смањењем обима трансфера између меморија уређаја и хоста.

- Проширивање решења новим програмским моделима и имплементацијама микромодела на уређајима као што су ФПГА.
- Прорачуни на макро скали могу бити паралелизовани. Метод КЕ је често коришћен и већина бибилиотека које омогућавају поједностављену употребу тог метода већ имају уграђене механизме паралелизације.
- Процена комплексности микромодела на нивоу целе симулације и статичка декомпозиција на основу тих информација не дају оптималну расподелу послова током целе симулације. Једна од могућности унапређења јесте примена квази-статичке расподеле, где би се на крају сваког временског корака симулације КЕ вршило ажурирање процене. У случају великих одступања у проценама би се вршила проновна декомпозиција, уз евентуалну минимизацију комуникације између учесничких процеса приликом размене стања микромодела.

Што се самог двоскалног модела тиче, могуће је његово даље развијање проширивањем новим врстама микромодела, као и дефинисањем сложенијег модела и увођењем више процеса на макро скали.

Не мање важне су и активности којима би се добијени резулатати и развијане имплементације модела сместиле у контекст међународних истраживачких иницијатива везаних за вишескално моделирање у биолошким системима како што је *VPH/Physiome* и понудиле у форматима већ дефинисаних стандарда. Додатно, један од планова јесте испитивање могућности повезивања развијеног софтверског решења са оптшим оквиром за дистрибуирано извршавање вишескалних модела као што је *MUSCLE* 2. Представљено решење је оптимизовано за извршавање модела мишића. Уколико би мишићни модел био испитиван у контексту сложеног система симулирања различитих процеса у целом организму, морао би да буде смештен у контекст робуснијих система који омогућавају извршавање симулација у извршном окружењу великих капацитета.

Библиографија

[1] J. Southern, J. Pitt-Francis, J. Whiteley, D. Stokeley, H. Kobashi, R. Nobes, et al. *Multi-scale computational modelling in biology and physiology*. Prog. Biophys. Mol. Biol. 96 (1-3) :60–89, 2008.

[2] I.G. Kevrekidis, C.W. Gear, J.M. Hyman, P.G. Kevrekidid, O. Runborg, C. Theodoropoulos. *Equation-Free, Coarse-Grained Multiscale Computation: Enabling Mocroscopic Simulators to Perform System-Level Analysis.* Commun. Math. Sci. 1 (4) :715–762, 2003.

[3] W. E, B. Engquist. *The Heterognous Multiscale Methods.* Commun. Math. Sci. 1 (1) :87–132, 2003.

[4] A. Brandt. *Multi-Level Adaptive Solutions to Boundary-Value.* Math. Comput. 31 (138) :333–390, 1977.

[5] A. Brandt. *Multiscale scientific computation: review 2000*. in: T.J. Barth, T. Chan, R. Haimes (Eds.), Multiscale and Multiresolution Methods, pp. 1–96. Springer Berlin Heidelberg, Berlin, Heidelberg, 2002.

[6] A.G. Hoekstra, J.-L. Falcone. A. Caiazzo, *Cellular Automata*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.

[7] J. Borgdorff, E. Lorenz, A.G. Hoekstra, J.-L. Falcone, B. Chopard. *A Principled Approach to Distributed Multiscale Computing, from Formalization to Execution*. 2011 IEEE Seventh Int. Conf. E-Science Work. 97–104, 2011.

[8] The COAST project, Complex automata Simulation Techique. `http://http://www.complex-automata.org/`

[9] VPH/Physiome project, `http://physiomeproject.org/`

[10] OpenCMISS, Physiome Project. http://physiomeproject.org/software/opencmiss

[11] E. Winsberg. Computer Simulations in Science. Stanford Encycl. Philos. (Fall 2014 Edition), Edward N. Zalta (ed.), http://plato.stanford.edu/archives/fall2014/entries/simulations-science/

[12] M. Kojić, R. Slavković, M. Živković, N. Grujović. *Metod konačnih elemenata I (linearna analiza)*. Mašinski fakultet, Univerzitet u Kragujevcu, Kragujevac, 2010.

[13] B. Stojanović. *Generalizacija fenomenološkog Hilovog modela u cilju izučavanja zamora mišića*. PhD thesis, Univerzitet u Kragujevcu, 2007.

[14] K.J. Bathe. *Finite element procedures in Engineering Analisys*. Prentice-Hall, Englewood Cliffs, NJ, 1982.

[15] M. Lister. *The numerical solution of hyperbolic partial differential equations by the method of characteristics*. Math. Methods Digit. Comput. 1:165–179, 1960.

[16] H.E. Twizell. *Computational Methods For Partial Differential Equations*. Ellis Horwood Limited, Chichester, 1984.

[17] M.D. Binder, N. Hirokawa, U. Windhorst, eds. *Encyclopedia of Neuroscience*. Springer-Verlag GmbH Berlin Heidelberg, 2009.

[18] A.F. Huxley, R. Niedergerke. *Structural Changes in Muscle During Contraction: Interference Miscroscopy of Living Muscle Fibers*. Nature. 173 (4412) :971–973, 1954.

[19] H. Huxley, J. Hanson. *Changes in the Cross-Striations of Musle during COntraction and Stretch and their Structural interpretation*. Nature. 173 (4412) 4412 :973–976, 1954.

[20] C.L. Stafield. *Principles of Human Physiology* 5th ed. Pearson Education, Glenview, IL, USA, 2012.

[21] M. Blix. *Die Lange und die Spannung des Muskels I.* Skand Arch Physiol. (1891) 3 :295–318.

[22] M. Blix. *Die Lange und die Spannung des Muskels II.* Skand Arch Physiol. (1893) 4 :399–409.

[23] M. Blix. *Die Lange und die Spannung des Muskels III.* Skand Arch Physiol. (1894) 5 :149–206.

[24] A. V. Hill. *The Heat of Shortening and the Dynamic Constants of Muscle*. Proc. R. Soc. B Biol. Sci. 126 (843) :136–195, 1938.

[25] A.M. Gordon, A.F. Huxley, F.J. Julian. *The variation in isometric tension with sarcomere length in vertebrate muscle fibres*. J. Physiol. 184 (1) 170–192, 1966.

[26] F.E. Zajac. *Muscle and tendon: properties, models, scaling, and application to biomechanics and motor control.* Crit. Rev. Biomed. Eng. 17 (4) :359–411, 1989.

[27] M. Kojic, S. Mijailovic, N. Zdravkovic, Modelling of muscle behaviour by the finite element method using Hill's three-element model, Int. J. Numer. Methods Eng. 43 (5) :941–953, 1998.

[28] B. Stojanovic, M. Kojic, M. Rosic, C.P. Tsui, C.Y. Tang. *An extension of Hill's three-component model to include different fibre types in finite element modelling of muscle*. Int. J. Numer. Methods Eng. 71 (7) :801–817, 2007.

[29] C.Y. Tang, B. Stojanovic, C.P. Tsui, M. Kojic, Modeling of muscle fatigue using Hill's model., Biomed. Mater. Eng. 15 (5) :341–8, 2005.

[30] M. Kojić, N. Filipović, B. Stojanović, N. Kojić. *Computer modeling in Bioengineering*. J Wiley and Sons, Chichester, 2008.

[31] C.Y. Tang, C.P. Tsui, B. Stojanovic, M. Kojic. *Finite element modelling of skeletal muscles coupled with fatigue*. Int. J. Mech. Sci. 49 (10) :1179–1191, 2007.

[32] A.F. Huxley. *Muscle structure and theories of contraction*. Prog. Biophys. Biophys. Chem. 7:255–318, 1957

[33] T.A. McMahon. *Muscles, reflexes and locomotion*. Princeton University Press, New Jersey, 1984.

[34] A.F. Huxley. *A Note Suggesting that the Cross-Bridge Attachment during Muscle Contraction may Take Place in Two Stages*. Proc. R. Soc. B Biol. Sci. 183 (1070) 1070 :83–86, 1973.

[35] R.W. Lymn, E.W. Taylor. *Mechanism of adenosine triphosphate hydrolysis by actomyosin.* Biochemistry. 10 (25) :4617–4624, 1971.

[36] B. Brenner, E. Eisenberg. *The mechanism of muscle contraction. Biochemical, mechanical, and structural approaches to elucidate cross-bridge action in muscle.* Basic Res. Cardiol. 82 Suppl 2 :3–16, 1987.

[37] E. Eisenberg, T.L. Hill, Y. Chen. *Cross-bridge model of muscle contraction. Quantitative analysis.*, Biophys. J. 29 (2) :195–227, 1980.

[38] H.E. Huxley, A. Stewart, H. Sosa, T. Irving. *X-ray diffraction measurements of the extensibility of actin and myosin filaments in contracting muscle.* Biophys. J. 67 (6) :2411–21, 1994.

[39] K. Wakabayashi, Y. Sugimoto, H. Tanaka, Y. Ueno, Y. Takezawa, Y. Amemiya. *X-ray diffraction evidence for the extensibility of actin and myosin filaments during muscle contraction*. Biophys. J. 67 (6) :2422–2435, 1994.

[40] S.M. Mijailovich, B. Stojanovic, M. Kojic, A. Liang, V.J. Wedeen, R.J. Gilbert. *Derivation of a finite-element model of lingual deformation during swallowing from the mechanics of mesoscale myofiber tracts obtained by MRI*. J. Appl. Physiol. 109 (5) :1500–14, 2010.

[41] T.L. Daniel, A.C. Trimble, P. Bryant Chase. *Compliant Realignment of Binding Sites in Muscle: Transient Behavior and Mechanical Tuning*. Biophys. J. 74 (4) :1611–1621, 1998.

[42] M. V Razumova, A.E. Bukatina, K.B. Campbell. *Stiffness-distortion sarcomere model for muscle simulation.* J. Appl. Physiol. 87 (5) :1861–1876, 1999.

[43] H. El Makssoud, D. Guiraud, P. Poignet, M. Hayashibe, P.-B. Wieber, K. Yoshida, et al. *Multiscale modeling of skeletal muscle properties and experimental validations in isometric conditions.* Biol. Cybern. 105 (2) :121–38, 2011.

[44] J. Bestel. *Modèle différentiel de la contraction musculaire con- trolée. Application au système cardio-vasculaire*. PhD thesis, University Paris IX Dauphine, FR, 2000.

[45] J. Bestel, M. Sorine. *A differential model of muscle contraction and applications.* In: schloessmann Seminar on mathematical models in biology, chemistry and physics. University Paris IX Dauphine, FR, 2000.

[46] M. Hayashibe, D. Guiraud. *Voluntary EMG-to-force estimation with a multi-scale physiological muscle model.* Biomed. Eng. Online. 12:86, 2013.

[47] J.W. Fernandez, M.L. Buist, D.P. Nickerson, P.J. Hunter. *Modelling the passive and nerve activated response of the rectus femoris muscle to a flexion loading: a finite element framework.* Med. Eng. Phys. 27 (10) :862–70, 2005.

[48] P.J. Hunter. *Myocardial constitutive laws for continuum mechanics models of the heart*. Adv Exp Med Biol. 382 :303–318, 1995.

[49] M. Böl, R. Weikert, C. Weichert. *A coupled electromechanical model for the excitation-dependent contraction of skeletal muscle*. J. Mech. Behav. Biomed. Mater. 4 (7) :1299–310, 2011.

[50] O. Röhrle, J.B. Davidson, A.J. Pullan. *Bridging Scales: A Three-Dimensional Electromechanical Finite Element Model of Skeletal Muscle.* SIAM J. Sci. Comput. 30 (6) :2882–2904, 2008.

[51] O. Röhrle, J.B. Davidson, A.J. Pullan. *A physiologically based, multi-scale model of skeletal muscle structure and function.* Front. Physiol. 3 (September) :358, 2012.

[52] P.R. Shorten, P. O'Callaghan, J.B. Davidson, T.K. Soboleva. *A mathematical model of fatigue in skeletal muscle force contraction.* J. Muscle Res. Cell Motil. 28 (6) :293–313, 2007.

[53] C. Bradley, A. Bowery, R. Britten, V. Budelmann, O. Camara, R. Christie, et al. *OpenCMISS: a multi-physics & multi-scale computational infrastructure for the VPH/Physiome project*. Prog. Biophys. Mol. Biol. 107 (1) :32–47, 2011.

[54] R.G. Christie, P.M.F. Nielsen, S.A. Blackett, C.P. Bradley, P.J. Hunter. *FieldML: concepts and implementation*. Philos. Trans. A. Math. Phys. Eng. Sci. 367 (1895) :1869–1884, 2009

[55] B. Stojanović, M. Svičević, A. Kaplarević-Mališić, M. Ivanović, D. Nedić, N. Filipović, et al. *Multiscale Muscle Modelling*, BIBE 2015, 15th Int. Conf. Bioinforma. Bioeng. 2-4th Novemb. 2015, Belgrade, Serbia, :74., 2015.

[56] M. Kojic, K.-J. Bathe. *Inelastic analysis of solids and structures*. Springer, 2005.

[57] S.M. Mijailovich, J.J. Fredberg, J.P. Butler. *On the theory of muscle contraction: filament extensibility and the development of isometric force and stiffness*. Biophys. J. 71 (3) :1475–84, 1996.

[58] Top 500 lists. http://www.top500.org/

[59] D. Culler, J.P. Singh, A. Gupta. *Parallel Computer Architecture: A Hardware/Software Approach.* Morgan Kaufmann, San Franciscko, CA, USA, 1997.

[60] M. Quinn. *Parallel Programming in C with MPI and OpenMP*. 1st ed., McGraw-Hill, New York, USA, 2003.

[61] M.J. Flynn. *Some Computer Organizations and Their Effectiveness*. IEEE Trans. Comput. C-21 (9) :948–960, 1972.

[62] D. Kirk, W. Hwu. *Programming Massively Parallel Processors: A Hands-on Approach*. 2nd ed, Morgan Kaufmann, San Franciscko, CA, USA, 2012.

[63] B. Barney. *POSIX Threads Programming*. Lawrence Livermore Natl. Lab., https://computing.llnl.gov/tutorials/pthreads/

[64] *OpenMP*. http://openmp.org/

[65] *MPI, Message Passing Interface Forum*. http://www.mpi-forum.org/

[66] *NVIDIA, NVIDIA CUDA Compute Unified Device Architecture Programming Guide*. http://docs.nvidia.com/cuda/cuda-c-programming-guide/

[67] OpenCL - The open standard for parallel programming of heterogeneous systems, Khronos Gr., https://www.khronos.org/opencl/

[68] M. Ivanović. *Glatka čestična hidrodinamika - paralelizacija algoritama i primena u dinamici fluida*. PhD thesis, Prirodno-matematički fakultet, Univerzitet u Kragujevcu, 2010.

[69] *MPICH*. https://www.mpich.org/

[70] *OpenMPI*. http://www.open-mpi.org/

[71] I. Foster. *Designing and building parellel programs: Concepts and Tools for Parallel Software Engineering*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, 1995.

[72] G.M. Amdahl. *Validity of the Single Processor Approach to Achieving Large Scale Computing Capabilities, Reprinted from the AFIPS Conference Proceedings.* Vol. 30 (Atlantic City, N.J., Apr. 18–20), AFIPS Press, Reston, Va., 1967, pp. 483–485, when Dr. Amdahl was at Inte, IEEE Solid-State Circuits Newsl. 12 (3) :19–20, 2007.

[73] T. Heidlauf, O. Röhrle. *Modeling the chemoelectromechanical behavior of skeletal muscle using the parallel open-source software library OpenCMISS.* Comput. Math. Methods Med. 2013 :517287, 2013.

[74] M. Ivanović, B. Stojanović, A. Kaplarević-Mališić, R. Gilbert, S. Mijailovich. *Distributed multi-scale muscle simulation in a hybrid MPI-CUDA computational environment*. Simulation: Transactions of the Society for Modeling and Simulation International, 92 (1) :19-31, 2016.

[75] A. Kaplarević-Mališić, M. Ivanović, B. Stojanović, M. Svičević, D. Antoniojević. *Employing Phenomenological Model in Load-balancing Optimization of Parallel Multi-scale Muscle Simulations*. BIBE 2015, 15th Int. Conf. Bioinforma. Bioeng. 2-4th Novemb. 2015, Belgrade, Serbia, :73, 2015.

[76] T. Cover, P. Hart. *Nearest neighbor pattern classification*. Inf. Theory, IEEE Trans. 13 (1) :21–27, 1967.

[77] N.S. Altman. *An Introduction to Kernel and Nearest-Neighbor Nonparametric Regression*. Am. Stat. 46 (3) :175–185, 1992.

[78] M. Kojić, R. Slavković, M. Živković, N. Grujović. *PAK – Finite Element Program for Linear and Non-linear Structural Analysis*. Faculty of Mechanical Engineering, University of Kragujevac, 1996.

[79] *The R Project for Statistical Computing*. https://www.r-project.org/

[80] R.J. Gilbert, V.J. Napadow, T.A. Gaige, V.J. Wedeen. *Anatomical basis of lingual hydrostatic deformation*. J. Exp. Biol. 210 (23) :4069–82, 1992.

[81] *OpenACC*. http://www.openacc.org/

[82] J.O. Dada, P. Mendes. *Multi-scale modelling and simulation in systems biology.* Integr. Biol. (Camb). 3 (2) :86–96, 2011.

[83] S.I. Fox. *Human Physiology*. 12th ed. McGraw-Hill, New York, USA, 2011.

Биографија

Ана Капларевић-Малишић је рођена 1974. године у Крагујевцу. Основну школу и гимназију завршила Крагујевцу као носилац Вукове дипломе. Природно-математички факултет у Крагујевцу, група математика, смер рачунарство, уписала 1992/93, а дипломирала 1996. године са просечном оценом 8.86. Од 1997. до 2003. године била је ангажована као асистент-приправник за групу предмета из програмирања на Природно-математичком факултету у Крагујевцу. Од 2003. године је ангажована, као асистент за ужу научну област информациони системи, на истом факултету.

Последипломске-магистарске студије на групи математика, смер рачунарство, уписала је школске 1996/97. године и све предмете предвиђене планом и програмом положила са просечном оценом 10. Магистарску тезу под насловом "Интернет технологије и интелигентни системи у образовању" одбранила 2003. године и тиме стекла звање магистра информатичких наука.

До сада је била ангажована на реализацији вежби на неколико студијских програма и на неколико предмета и то:

- На студијском програму за стицање звања Дипломирани математичар-информатичар: Базе података, Оперативни системи и конкурентно програмирање, Пројектовање програма и информационих система, Методика наставе математике и информатике;
- На предметима студијског програма за стицање стручног/академског звања Информатичар и Дипломирани информатичар - мастер: Рачунарски системи, Структуре података и алгоритми 1, Базе података 1, Оперативни системи 1, Објектно-оријентисано програмирање, Интернет технологије.

Учествовала је у реализацији 3 научна, 2 техничко-технолошка као и 2 стручна пројекта. Била је учесник 2 ТЕМПУС пројекта, оба везана за развој курикулума. Активно је учествовала у раду семинара из Логике на Математичком институту Српске академије наука и уметности.

Активни је члан Друштва математичара Србије и секретар Подружнице Крагујевца. Од 2003. члан је редакције у уредник сталне рубрике у часопису *Тангента* - јединоm националном часопису за ученике средњих школа из области математике и рачунарства. У периоду од 2005. до 2007. била је технички уредник *Математичког листа* - часописа за математику и рачунарство намењеног ученицима основне школе у издању Друштва математичара Србије.

Покретач је и активни предавач у оквиру Математичке радионице младих, која од 1999. године ради са младим талентима у Крагујевцу. Од 2007. члан је Републичке комисије за Такмичења из програмирања за ученике основних школа, 2007. била је лидер тима Републике Србије на Јуниорској балканској олимпијади из информатике. У периоду од 2003. до 2005. била је ментор ученицима основних школа на изради радова у оквиру активности Регионалног центра за таленте.

СПИСАК ОБЈАВЉЕНИХ РАДОВА

1. D. Parezanović, D. Stefanović, A. Kaplarević, *In Internet Environment Business and Educational Activities of Universities*, Management, 17-18, 2000 (44-51)
2. D. Stefanović, A. Kaplarević-Mališić, *Information Technologies in University Education Activities*, Kragujevac Journal of Mathematics, 23, 2001 (131-153)
3. B. Radenković, D. Stefanović, A. Kaplarević-Mališić, A. Савић, *An application of distance learning as support for traditional education at the University*, INFO M, 2003 (21 – 27)
4. M. Mosurović, T. Stojanović, A. Kaplarević-Mališić, *Reasoning in Basic Description Logics and Description Logics with Modal Operators*, Зборник радова Logic in Computer Science, Математички институт SANU, 2009 (113-158) ISBN 978-86-80593-40-1
5. T. Stojanović, A. Kaplarević-Mališić, Z. Ognjanović, *An extension of the probability logic* LPP_2, Kragujevac J. Math. 33, 2010 (113-158) ISSN 1450-9628
6. D. Stefanović, I. Radojević, Lj. Čomić, A. Ostojić, M. Topuzović, A. Kaplarevic-Mališić, Management Information System of Lakes and Reservoirs, Water Resources, 39 (4)488-495, 2012. ISSN: 0097-8078
7. M. Ivanović, A. Kaplarevic-Mališić, V. Simić, B. Stojanović, *Design and comparison of two web service based frameworks for parallel evaluation of the population in genetic algorithms,* Facta Universitatis, Series: Mathematics and Informatics, 29(2):155–171, 2014. ISSN: 0352-9665
8. M. Ivanović, V. Simić, B. Stojanović, A. Kaplarević-Mališić, B. Marović, *Elastic grid resource provisioning with WoBinGO: A parallel framework for genetic algorithm based optimization*, Future Generation Computer Systems, 42(0):44 – 54, 2015. ISSN: 0167-739X
9. M. Ivanović, B. Stojanović, A. Kaplarević-Mališić, R. Gilbert, S. Mijailovich, *Distributed multi-scale muscle simulation in a hybrid MPI-CUDA computational environment*, SIMULATION: Transactions of The Society for Modeling and Simulation International, 92(1) 19-31, 2016. ISSN: 0037-5497

РАДОВИ ШТАМПАНИ НА СРПСКОМ ЈЕЗИКУ У ЧАСОПИСУ СА РЕЦЕНЗИЈОМ

10. А. Капларевић, С. Милошевић, Б. Раденковић, В. Вујин, *Интернет технологије у образовном процесу*, Info Science, 1-2, 2000 (26-32)

САОПШТЕЊА НА КОНФЕРЕНЦИЈАМА ШТАМПАНА У ИЗВОДУ ИЛИ У ПОТПУНОСТИ

11. А. Капларевић, Д. Парезановић, *An Application of the Internet Technologies in Activities of the Faculty of Science in Kragujevac*, EURO XVII, 17th European Conference on Operational Research, Budapest, Hungary, July 16-19, 2000
12. Стефановић Д., Радисављевић С., Тимотијевић Т., Капларевић А., ТЕХНИЧКИ ИНФОРМАЦИОНИ СИСТЕМ РЕЗЕРВНИХ ДЕЛОВА ВОЗИЛА, Info-Teh 2001, XV научно-стручни скуп, Зборник радова, 316-319, Врњачка Бања, 18-22 јун 2001.
13. Б. Раденковић, Д. Стефановић, А. Капларевић-Малишић, А. Савић, *An application of distance learning as support for traditional education at the University*, 6th International Conference on Telecommunications in Modern Satellite, Cable and Broadcasting Service, TELSIKS 2003., Volume 2, 2003 (746 – 752)
14. B. Stojanović, V. Simić, M. Ivanović, A. Kaplarević-Mališić, A. Stanojević, *WCF Platform for Distributed Evaluation in Evolutionary Algorithms*, Proceedings of the 4th International Conference "Science and Higher Education in Function of Sustainable Development" SED 2011, Uzice, Serbia, 7-8 October 2011, pp. 2:8-13, ISBN 978-86-83573-22-6
15. A. Kaplarević-Mališić, M. Ivanović, B. Stojanović, M. Svičević, D. Antonijević, *Employing Phenomenological Model in Load-balancing Optimization of Parallel Multi-scale Muscle Simulations*, 15th International Conference on Bioinformatics and Bioengineering, Nov 02--04, 2015, Belgrade, Serbia, ISBN: 978-1-4673-7982-3
16. B. Stojanović, M. Svičević, A. Kaplarević-Mališić, M. Ivanović, Đorđe M. Nedić, Nenad D. Filipovic and Srboljub M. Mijailovich, *Coupling Finite Element and Huxley Models in Multiscale Muscle Modeling*, 15th International Conference on Bioinformatics and Bioengineering, Nov 02-04, 2015, Belgrade, Serbia ISBN: 978-1-4673-7982-3

УЧЕШЋЕ НА ПРОЈЕКТИМА

1. 1997-2001 – 10М04 Рачунарство
2. 2002-2005 - 101379 Методе математичке логике за подршку закључивању у реалим ситуацијама
3. 01.04.2008.-31.03.2010. ТР – 22001 Развој и имплементација SeLaR информационог система
4. 2010 - ТР – 37013 Развој система за подршку оптималном одржавању високих брана у Србији
5. Tempus project N°JEP-CD-16156-2001: Computer Science Curricula Founding and Upgrading

6. Tempus Project N°CD-JEP-40053-2005: Science Teacher Education Revision and Upgrading STERU

Стручни пројекти

7. Развој, имплементација и одржавање информационог система студентске службе Природно-математичког факултета у Крагујевцу
8. CD каталога резервних делова Застава камиони ДОО 2000.

Simulation

Simulation: Transactions of the Society for Modeling and Simulation International
1–13

DOI: 10.1177/0037549715620299
sim.sagepub.com
SAGE SCS

Distributed multi-scale muscle simulation in a hybrid MPI–CUDA computational environment

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

Abstract

We present Mexie, an extensible and scalable software solution for distributed multi-scale muscle simulations in a hybrid MPI–CUDA environment. Since muscle contraction relies on the integration of physical and biochemical properties across multiple length and time scales, these models are highly processor and memory intensive. Existing parallelization efforts for accelerating multi-scale muscle simulations imply the usage of expensive large-scale computational resources, which produces overwhelming costs for the everyday practical application of such models. In order to improve the computational speed within a reasonable budget, we introduce the concept of distributed calculations of multi-scale muscle models in a mixed CPU–GPU environment. The concept is applied to a two-scale muscle model, in which a finite element macro model is coupled with the microscopic Huxley kinetics model. Finite element calculations of a continuum macroscopic model take place strictly on the CPU, while numerical solutions of the partial differential equations of Huxley's cross-bridge kinetics are calculated on both CPUs and GPUs. We present a modular architecture of the solution, along with an internal organization and a specific load balancer that is aware of memory boundaries in such a heterogeneous environment. Solution was verified on both benchmark and real-world examples, showing high utilization of involved processing units, ensuring high scalability. Speed-up results show a boost of two orders of magnitude over any previously reported distributed multi-scale muscle models. This major improvement in computational feasibility of multi-scale muscle models paves the way for new discoveries in the field of muscle modeling and future clinical applications.

Keywords

multi-scale, muscle modeling, GPU, heterogeneous computing, scheduling policy

1. Introduction

Investigating musculoskeletal disorders, as well as characterizing and prognosing neuromuscular diseases requires deep understanding of the structural and functional properties of muscles. The 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.[1,2] The expansion of knowledge regarding the complex mechanisms underlying muscle physiology solely leaning on *in vivo* and *in vitro* experiments is quite slow and limited. *In silico* analysis of detailed muscle models enables less-expensive and time-consuming hypothesis testing and evaluation, therefore providing a valuable tool in research activities.[3,4] Comprehensive and efficient computer models also play a key role in the development of the future software solutions which may be used in everyday clinical practice.

Generally speaking, existing computational muscle models fall into two classes: (1) biophysical, which investigates the ability of contractile proteins to generate force and movement at the cellular level and (2) phenomenological, which evaluates the performance of the whole muscle. Most biophysical models evolve from the hypothesized cross-bridge kinetic concepts originally formulated by

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

Corresponding author:
Boban Stojanović, Faculty of Science, University of Kragujevac, Radoja Domanovica 12, 34000 Kragujevac, Serbia
Email: bobi@kg.ac.rs

Employing Phenomenological Model in Load-balancing Optimization of Parallel Multi-scale Muscle Simulations

Ana M. Kaplarević-Mališić, Miloš R. Ivanović, Boban S. Stojanović, Marina R. Svičević, and Darko B. Antonijević

***Abstract*— Since multi-scale models of muscles rely on the integration of physical and biochemical properties across multiple length and time scales, these models are highly CPU consuming and memory intensive. Therefore, their practical implementation and usage in real-world applications is limited by their high requirements for computational power. There are various reported solutions to the problems of the distributed computation of the complex systems that could also be applied to the multi-scale muscle simulations. In this paper, we present a novel load balancing method for parallel multi-scale muscle simulations on distributed computing resources. The method uses data obtained from simple Hill phenomenological model in order to predict computational weights of the integration points within the multi-scale model. Using obtained weights it is possible to improve domain decomposition prior to multi-scale simulation run and consequently significantly reduce computational time. The method is applied to two-scale muscle model where a finite element (FE) macro model is coupled with Huxley's model of cross-bridge kinetics on the microscopic level. The massive parallel solution is based on decomposition of micro model domain and static scheduling policy. It was verified on real-world example, showing high utilization of all involved CPUs and ensuring high scalability, thanks to the novel scheduling approach. Performance analysis clearly shown that inclusion of complexities prediction in reducing the execution time of parallel run by about 40% compared to the same model with scheduler that assumes equal complexities of all micro models.**

I. Introduction

THE 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 ([1]–[3]) 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 [4]. Simulations of these kinetic processes, in the context of whole muscle models are tremendously computationally intensive and require simplifications of geometry, composition, and activation [5]. 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.

Regarding the fact that multi-scale muscle models are more informative and realistic than phenomenological models, their practical implementation and usage in real-world applications is limited by their requirements for computational power. Feasible usage of these models can be reached only by employing parallelization techniques and high performance computing (HPC) environments. Existing solutions for accelerating multi-scale muscle simulations lean on general parallelization techniques and frameworks [11]. Better utilization of general purpose tools in such context can be improved only by exploiting domain knowledge, i.e. muscle system specifics. This paper considers a new scheduling methodology based on extracting and using domain knowledge about muscle model in order to improve multi-scale muscle simulation efficiency.

Multi-scale muscle simulations use a large set of data to capture complex system state and its dynamics. In this kind of parallel computing problems, the domain decomposition is usually static, due to the fact that the network transfer is too expensive for that amount of data. Thus, the decisions made in initial decomposition have much greater impact on overall performance than in case of dynamic scheduling. Moreover, most realistic models consist of complex geometries with different materials with various characteristics, subjected to various conditions. Computational complexity of each individual model part significantly affects the optimal schedule. In the absence of any knowledge regarding these differences, all of them are

This work is supported by the Ministry of Science in Serbia, Grants III41007, OI174028, TR37013, and National Institutes of Health AR048776 and DC011528.

The authors are with Faculty of Science, University of Kragujevac, Radoja Domanovica 12, Kragujevac, Serbia (e-mail: ana@kg.ac.rs; mivanovic@kg.ac.rs; bobi@kg.ac.rs; marina.svicevic@kg.ac.rs; dantonijevic.kg@gmail.com)