Ръководството съставя годишен доклад за дейността на Сдружение „Национален център за суперкомпютърни приложения" в съответствие с изискванията на чл. 40 ал. 3 от Закона за юридическите лица с нестопанска цел. Настоящият доклад на ръководството съдържа изложение за дейността на Сдружение "Национален център за суперкомпютърни приложения" през 2016 година, както и целите за постигане през 2017 г.

Годишният финансов отчет към 31 декември 2016 г. е изготвен в съответствие с Националните счетоводни стандарти (НСС) приети в Република България. Финансовият отчет е одитиран от регистрирания одитор доц. д-р Али Вейсел.

**Регистрация и основни дейности**

Сдружение "Национален център за суперкомпютърни приложения" (Сдружение НЦСП) е учредено на 19.03.2009 г. и е регистрирано от Фирмено отделение с решение по ф.д. 11620/2009 г. от 29 април 2009 г. на Софийски градски съд. Сдружението е организация, регистрирана по Закона за юридическите лица с нестопанска цел и работеща в обществена полза.

НЦСП е създаден за извършване на дейност в обществена полза, като предметът на дейност включва: осигуряване на достъп на български дипломанти, магистри и докторанти, научни колективи, фирми, организации и академичната общност до европейски и световни високо производителни компютърни ресурси, организиране и провеждане на образователни курсове, семинари и обучения, стимулиране развитието и организиране представянето и практическо приложение на високо производителните компютри, други незабранени дейности, свързани с развитието на информационните и комуникационните технологии.

**Основен капитал**

За Сдружението не съществува изискване, наложено от закон, за регистриране на основен капитал. По решение на Общото събрание членовете на Сдружението внасят учредителни вноски за покриване на разходите по учредяване. Общият размер на определените учредителни вноски през 2016 година възлиза на 225 лева (2015: 225 лева), които са изцяло внесени.

**Управление**

Сдружението се управлява от Управителен съвет и от Председателя на УС, който ръководи чрез представляването на Сдружението и администрирането на дейността. Общото събрание избира Председателя на Управителния съвет.

**Оценка на дейността през отчетния период**

Периодът на дейност през 2016 година беше период на развитие, очертано основно от главните цели на ръководството да допринесе за постигане на обществени ползи.

В условията на неблагоприятната бизнес-среда през последните няколко години, която може да се характеризира като период на стагнация в икономическата криза като цяло, ръководството на Сдружението набеляза следните съществени дейности, изразходваните за тях средства, връзката им с целите и програмите на НЦСП и постигнатите резултати:

**Дейности, извършени в изпълнение на проект PRACE-4IP – Fourth Implementation Phase Project в съответствие с договор RI - 653838 от 01.02.2015г., сключен с Европейската комисия по Седма рамкова програма, за периода 01.01.2016 –31.12.2016г.**

**I.** Изпълнение на работните задачи по **Работен пакет №6 (РП6): Оперативни услуги за високопроизводителната екосистема**

**Услуги и мидълуър за свързване към PRACE като Tier-1**

Основните дейности в изпълнение на задача 6.1: Operation and coordination of the comprehensive common PRACE operational services (Опериране и координация на цялостните общи оперативни услуги на PRACE) включват:

- Разполагане и поддръжка за услугите за данни (GridFTP)
- Услуги за управление на ресурсите (напр. UNICORE и GLOBUS GRAM)
- Хармонизирани процедури за получаване на автентикация и оторизация (контрол на достъпа)
- Поддръжка за инфраструктурата с публични и частни ключове (PKI), администриране на потребителите, поддръжка за GSI-SSH
- Поддръжка за потребители и приложения
- Мониторинг на операциите
- Участие в регулярните дежурства HoD и OoD, създаване и проследяване на trouble tickets.

Резюме на извършената работа през периода по задача 6.1:

Продължава наблюдението за състоянието на физическите мрежови връзки, поддържащи достъпа до новата Tier-1 система чрез криптираната връзка на PRACE. Мониторинг-тестовете, които се извършват по проекта, показват положителен резултат.

През 2016 година се инсталираха и тестваха серия от услуги, изискващи се по проекта, а именно GridFTP, GLOBUS – GRAM и GSI-SSH. Разположени са сървърите, които да поддържат тези услуги, осигурен е достъп до тях по криптираната мрежа на PRACE. Осъществен е достъп от тези сървъри до общата файлова система Lustre и системата за обработка на задачите Moab.

Българският екип се включва в регулярните дежурства HoD (Helpdesk on Duty) и OoD (Operator on Duty), по време на които се наблюдава състоянието на цялата инфраструктура на PRACE, създават се и се проследяват trouble tickets и се координира поддръжката на потребителите.

През 2016 година, съобразно с хармонизираните политики и процедури за достъп и за допустима употреба (Acceptable use) се предоставя достъп на потребители, рабощи по други пакети на PRACE. Създават се потребителски акауити, осъществява се поддръжка на потребителите се инсталира софтуер, необходим за изпълнението на задачите им, продължава издаването и обновяване на X509 сертификати. Извършват са регулярни обновявания на базисния и приложен софтуер и наблюдение за сигурността.

GEANT4 и GEANTV на хетерогенни архитектури с ко-процесори Intel XeonPhi

През 2016 г. продължи разработването на пилотната услуга GEANT4@MIC. GEANT4 [http://geant4.cern.ch/] и GEANTV [http://geant.cern.ch/] (GEometryANdTracking) са Монте-Карло пакети за симулиране на преминаването на йонизиращи частици през веществото. Това е важна задача за няколко различни научни области, като се започне от фундаментални изследвания във физика на

елементарните частици и астрофизика и се стигне до чисто приложни области, като радиационна защита, космически изследвания, биология и медицина (изследване на ДНК, адронна терапия, медицинска физика и образна диагностика).

GEANT4, както и GEANTV са комплексни софтуерни библиотеки, предвидени да бъдат свързвани към специфичен потребителски код. От своя страна, GEANT се нуждае от допълнителни библиотеки и пакети, инсталирани и конфигурирани по специфичен начин. Двата пакета (GEANT4 и GEANTV) бяха инсталирани и конфигурирани за работа както на основните процесори, така и на ко-процесори от типа Intel Xeon Phi [http://www.intel.com/content/www/us/en/processors/xeon/xeon-phi-detail.html]. Към настоящия момент GEANTV не е в състояние да замени GEANT4, а представлява по-скоро прототип и затова не е интегриран в разработената от нас GEANT4 услуга.

GEANT4 е базиран на обектно-ориентирана технология, която позволява да се постигне максимална прозрачност при имплементацията на физичните процеси, като и възможност за лесно разширяване и еволюция. GEANT4 съдържа богат набор от компоненти за детекторна симулация, като например моделиране на геометрията на детекторите, отклика им, управление на отделните събития, трасиране, визуализация и потребителски интерфейс. Мултидисциплинарната приложимост на GEANT4 се гарантира от богатия набор имплементирани физични процеси, валидни в различни енергийни диапазони.

GEANT4 е наследник на FORTRAN-базираните GEANT пакети, които датират от далечната 1978 г. GEANT4 е първата версия, разработена на базата на обектно-ориентирана технология и е имплементиран на езика C++. GEANT4 може да се ползва, като самостоятелно приложение или като библиотека. Например в приложения, като CMSSW (ползва се във физиката на елементарните частици) и GATE (от областта на медицинската физика), GEANT4 се използва индиректно, посредством комплексни софтуерни рамки. HPC подходът, който стандартно се прилага, е базиран на GRID технология. Най-големите научни комплекси в областта на физиката на елементарните частици, използват Worldwide LHC ComputingGrid (WLCG) инфраструктура, за да извършват подобни симулации. При този подход на паралелизиране различни компютри симулират различни събития. Събитията са изцяло независими, затова не се налага трансфер на данни между различните компютри. Могат да се отличат обаче два проблема:

- С увеличаване на обема данни дори ресурсите на инфраструктури, като WLCG, могат да се окажат недостатъчни;
- Новите хетерогенни архитектури с ко-процесори Intel XeonPhi, не се използват ефективно при тези изследвания;

Нашата цел бе да разработим и предоставим услуга, позволяваща изпълняването на GEANT4 базирани симулации на платформа с множество Intel XeonPhi копроцесори.

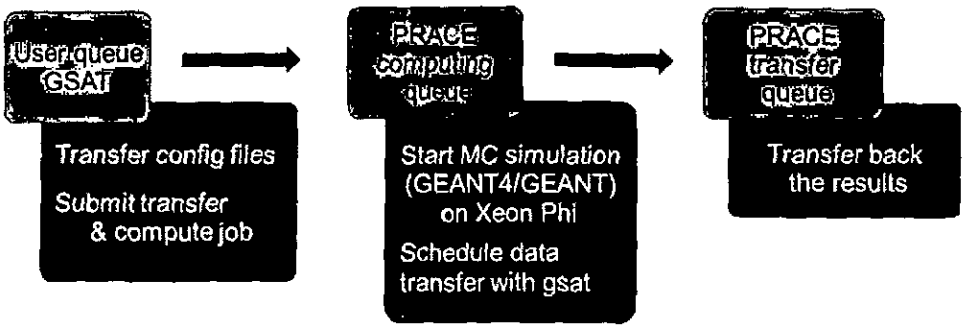
Дизайн на GEANT4@MIC услугата

Изградената GEANT4 услуга използва Gsatellite [https://github.com/fr4nk5ch31n3r/gsatellite/wiki/Gsatellite-ex-plained] за планировчик (scheduler) на трансфера на данни. Повечето от приложенията на Geant4 изискват големи изчислителни мощности и генерират голям обем от симулирани данни. Затова е важно симулираните данни да бъдат съхранявани и транспортирани по надежден начин. В допълнение, обработката на конфиденциални данни (например данни за пациент на център за адронна терапия) изисква спазването на стриктни правила и политика за сигурност, което допълнително затруднява работния процес.Gsatellite (https://github.com/fr4nk5ch31n3r/gsatellite/wiki/Gsatellite-explained) е обещаващ кандидат за планировчик на задачите. Gsatellite предоставя голяма функционалност, и едновременно с това е лесен за усвояване от потребителите. Базиран е на BASH , което го прави лесен за поддръжка и съвместим на практика с всяка операционна система. Gsatellite предлага някои много полезни възможности, които не се поддържат от повечето планировъчни програми. Например, някои задачи може да се прехвърлят в изчаквателен режим (използвайки командитеgqhold и gqrls на Gsatellite). Gsatellite се оказа полезен за осъществяването на планиран трансфер на данни с помощта на инструментите gtransfer или xrdcp. Като недостатъци на Gsatellite бяха идентифицирани липсата на възможност за задаване на приоритети на задачите и поставяне на лимит по време.

В текущата имплементация на услугата Gsatellite е отговорен за следните дейности (фиг. 1):

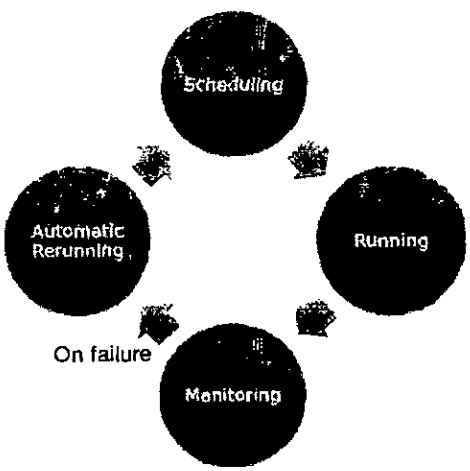
- Трансфер на входящите данни от потребителя до изчислителния център

- Планиране на задача
- Мониториране на задачата
- Трансфер на резултата от изчисленията до потребителя на услугата



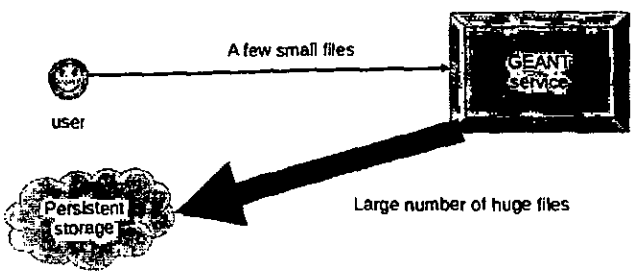

Фигура 1. *Използване на Gsatellite в работната последователност на услугата.*

Надеждността на услугата изисква неизпълнените вследствие на възникнал проблем задачи да бъдат обработени по подходящ начин. Ако проблемът се дължи на неправилна конфигурация, зададена от потребителя на услугата, единственото, което може да се направи, е да се изпрати съобщение за грешка. Ако обаче проблемът е вследствие на вътрешна грешка на услугата, задачата трябва да се стартира автоматично наново (Фиг. 2):




Фигура 2. *Обработка на проблемна задача.*

GEANT4 е Монте Карло пакет за симулиране на преминаването на йонизиращо лъчени през веществото. Входните данни, като правило се състоят от конфигурационен файл, докато резултатът от

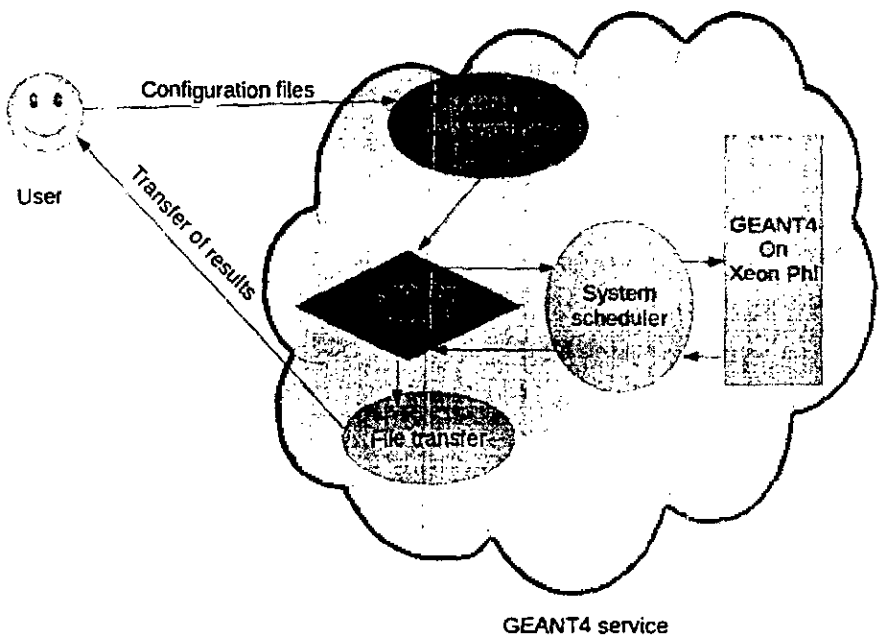


симулацията може да бъде голям файл с физически събития (Фиг. 3).

Фигура 3. *Асиметричен двупосочен трансфер: от потребителя към услугата се трансферират само малки конфигурационни файлове, докато от услугата към потребителя се трансформират големи файлове със симулационни данни.*

Дизайнът на услугата е вдъхновен от Облачните технологии, т.е. след изпращане на заявка от потребителя, задачата се изпълнява автоматично. В нашия случай заявката се осъществява, чрез качването на конфигурационни файлове от потребителя в специализирана директория на услугата. Услугата регистрира автоматично наличието на нови конфигурационни файлове и стартира автоматично изпълнението на задачата, като накрая автоматично трансферира резултата (Фиг. 4). Това се осъществява посредством специално разработена програма, която проверява за наличието на нови файлове в служебната директория. Услугата очаква два конфигурационни файла – един конфигурационен файл за GEANT4 и един служебен конфигурационен файл, който дава инструкции на услугата. Конфигурационния файл за GEANT4 трябва да съблюдава синтаксиса описан в инструкциите в ръководството на GEANT4. Служебният файл предоставя необходима информация на услугата, а именно информация необходима за осъществяване на автоматичен файлов трансфер и по-точно:

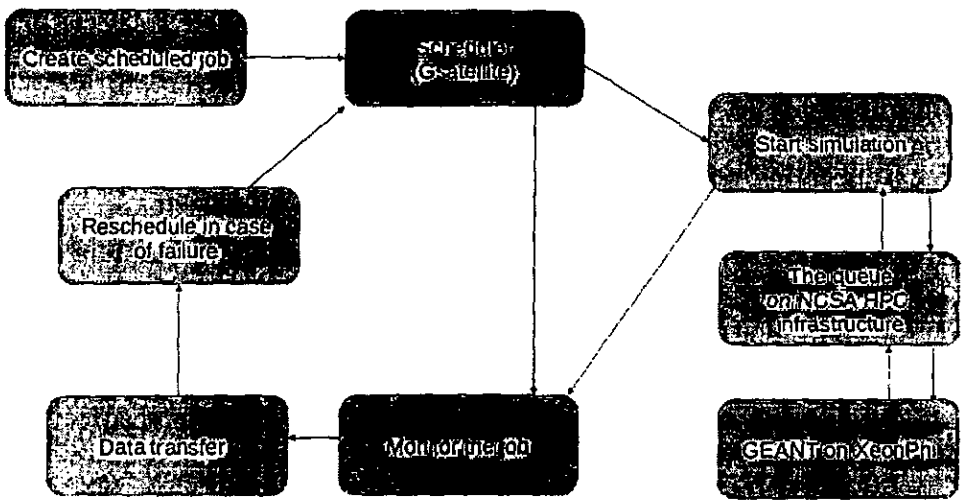
- протоколът за трансфер на файлове. Ако не бъде специфициран такъв протокол, не се извършва автоматичен трансфер и тогава потребителя трябва ръчно да се погрижи за трансфера на файловете.
- пътят (адресът) на хранилището, където трябва да бъде трансфериран автоматично резултата.
- e-mail адрес за нотификация (за приключване на задачата и осъществяване на трансфера)




Фигура 4. *Използване на Gstallite в реализирането на услугата GEANT4@MIC.*

Когато двата конфигурационни файла бъдат детектирани, започва автоматичната процедура на услугата (фиг. 4). Автоматично се създава задача и нейното изпълнение се планира от Gsatellite. Детектирането на конфигурационните файлове и автоматичното създаване на задача се осъществява от

специален скрипт написан на Python и работещ във фонов режим на фронтенда. Gsatellite от своя страна изпраща изчислителна задача на системния планировчик (т.е. Gsatellite се явява планировчик на услугата и комуникира със системния планировчик на Авитохол). Системният планировчик и инсталиран на суперкомпютъра и е отговорен за споделянето на ресурсите между потребителите и услугите. Към настоящия момент GEANT4 услугата не се предвижда да заобикаля системния планоровчик и да достъпва директно изчислителните модули – достъпът до изчислителните модули е само през системния планировчик. След като приключи изчислителната задача, контролът отново се поема от Gsattelite, който планира и изпълнява файлов трансфер. Детайлна диаграма на работния цикъл е представена на фиг. 5. Gsatellite стартира симулацията, мониторира нейния прогрес и се грижи за трансфера на данните. В случай, че възникне проблем изчислителната задача автоматично се пуска наново.




Фигура 5. *Работна диаграма за ползването на Gsatellite, като планировчик на услугата.*

Към настоящия момент се поддържат само два протокола за трансфер -SFTP и gtransfer. В случай на SCP, ключовете могат да бъдат настроени за постоянно още със създаването на потребителския акаунт, което в последствие позволява напълно автоматичен трансфер на файлове. В случай на gtransfer се използва гридпрокси, което има ограничена във времето валидност и се налага това прокси да се създава наново от потребителя преди трансфер на данни, като за целта се използва командата grid-proxy-init от пакета Globus. Тази команда изисква ръчно въвеждане на парола. Ако поради някаква причина (например изтекло гридпрокси) трансферът не се осъществи, файловете се запазват докато не бъдат безопасно трансферирани на по-късен етап. Успешно трансферираните файлове се изтриват с цел оптимизация на дисковото пространство.

Препоръчителния трансфер е чрез gtransfer , който от своя страна се поддържа от PRACE. SCP е добавен, за да направи услугата достъпна за потребители, които нямат с GRID и не разполагат с GRID инфраструктура. Пакетът Globus, gtransfer, UberFTP и TGFTP са инсталирани на фронт-end машината на услугата и са достъпни за потребителите.

**Инсталнране на GEANTV**

Програмния пакет GEANT4 е наследник на GEANT3 и не е оптимизиран за паралелни изчисления. В момента се разработва така наречения GEANTV, който по идея е оптимизиран за паралелни изчисления. Функционалността на GEANTV е все още прототипна и този продукт все още не се използва за физически изследвания. GEANTV се нуждае от ред допълнителни пакети при своята инсталация. GEANTV , като и допълнителните пакета бяха компилирани и инсатлирани на Авитохол. Необходимите пакети зависят един от друг и това налага да бъдат конфигурирани и инсталирани, като

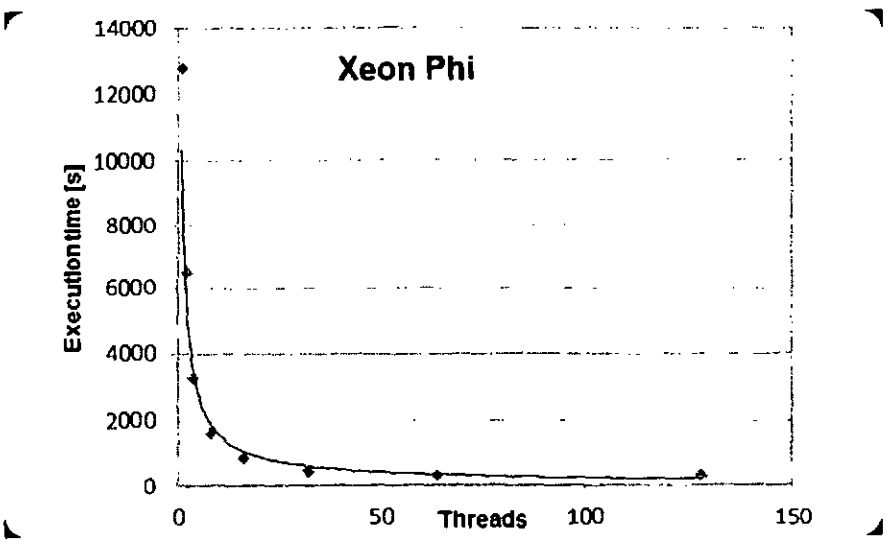
се спазва специфична последователност. Системният компилатор на Red Hat Enterprise Linux версия 6.7 не позволява компилирането на GEANTV и това налага дори и GNU компилатора да бъде компилиран и инсталиран, както и някои системни библиотеки на компилатора. Вересии на компилатора по-високи от 4.8.0 позволяват компилация на GEANTV. Конкретно на Авитохол бе инсталирана и тествана версия 4.8.5 на компилатора.

Следните допълнителни пакети трябва да бъдат конфигурирани и инсталирани преди инсталирането на GEANTV:

- **ROOT6**
- **Vc**
- **VecGeom**
- **Pythia8**
- **HepMC3**
- **Xerces-c**
- **GEANT4**

**Изследване на производителността**

При тестовете използваме симулация на сноп частици с енергия 1 TeV, който се насочва към фиксирана оловна мишена, като се проследяват взаимодействията на първичните и вторичните частици през експерименталната установка. Тя е съставена от 5 детекторни диска. Оловната мишена е дебела 5 cm, а детекторите представляват дебели 20 cm йонизационни камери, запълнени с ксенон. Разстоянието между центъра на мишената и първата йонизационна камера е 80 cm, а разстоянието между отделните йонизационни камери е също 80 cm. Околното пространство е запълнено с въздух. На Фиг. 6 е показано ускорението в зависимост от броя нишки на XeonPhi в нативен режим, при симулиране на мюонен сноп.




Фигура 6. *Зависимост на изчислително време от броя нишки (128к събития).*

Разгледани са два случая:

- протонен сноп

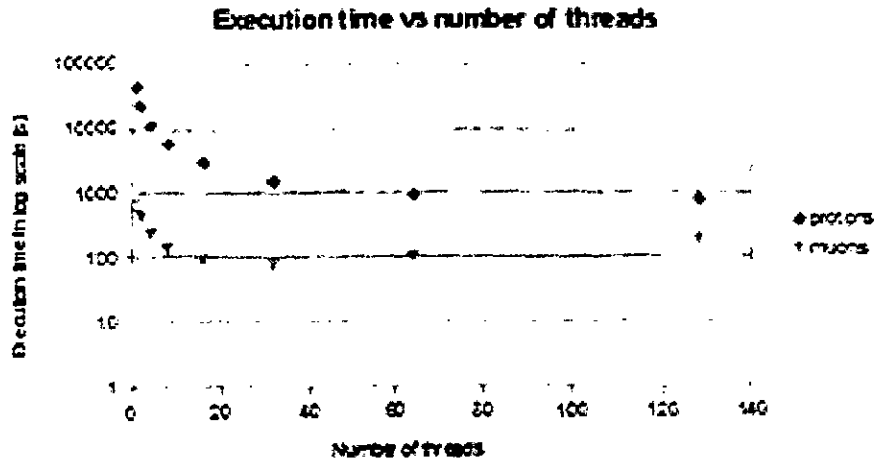
- сноп мюони (положително заредени)

Характерното е, че двата вида частици (протони и мюони) взаимодействат по различен начин с веществото, като мюоните пораждат малък брой вторични частици, а протоните пораждат адронни лавини с голям брой вторични частици.

Резултатите за ускорението са представени на Фиг. 7 и напълно потвърждават предварителните очаквания. За случая с мюоните времето за изпълнение на програмата достига минимум при 32 нишки и след това отново се увеличава. За случая с протоните се наблюдава бързо намаляване на времето за изпълнение на програмата до 32 нишки, но при по-голям брой нишки времето за изпълнение продължава да намалява, макар и доста по-плавно. Естественият извод е, че за мюонния случай времето за настройка на програмата доминира над времето за реални изчисления в случай на голям брой нишки. Обаче в случай на голям брой събития (Фиг. 7, горе), кривата с мюоните не достига минимум.

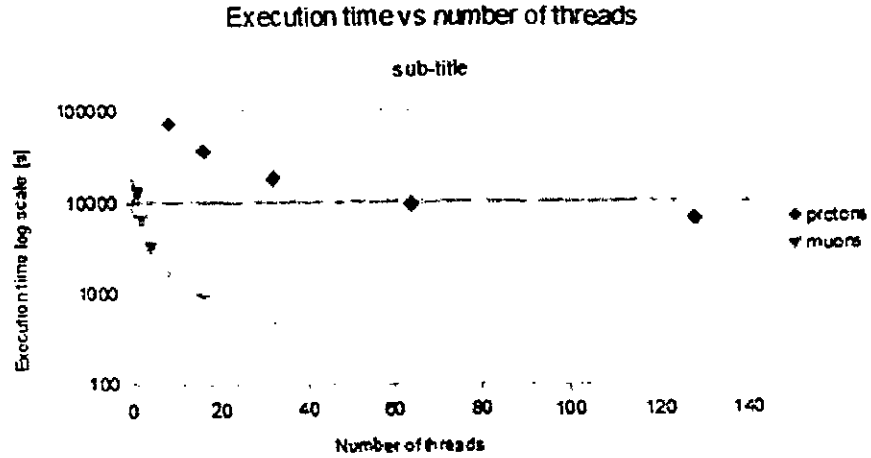
Резултатите показват значително ускорение с увеличаване на броя на нишките, особено в случаите на голям брой симулирани събития. Оказва се, че производителността зависи от конкретната физическа задача. Предимството на многонишковите изчисления се забелязва най-много при случаите с голям брой частици, като това дали частиците са първични или вторични, не оказва съществено влияние. Поведението на скалируемостта може да се обясни, чрез различните свойства на симулираните частици. Препоръчваме потребителите да използват голям брой нишки, ако симулираните частици се очаква да породат голям брой вторични частици. В случай на малък брой вторични частици, увеличаване на производителността може да се осъществи по два начина:

- да се определи оптималния брой нишки за определен тип симулация и да се ползва този брой




нишки

- да се




яка от нишките

Фигура 7. *Изчислително време в зависимост от броя нишки, като са симулирани два различни типа частици (протони и мюони) и различен брой събития – 10000 (горе) и 128000 (долу).*

Чисто формално, копроцесорът няма по-висока производителност от централния процесор, но за случаите на голям брой събития (а точно този случай е най-важен за практиката) производителността е сравнима и това оправдава използването на Intel XeonPhi за GEANT4 базирани изчисление. Паралелната версия на GEANT, а именно GEANTV, ще позволи увеличаване на производителността, включително и върху Intel XeonPhi базирана платформа. Отчитайки и енергийната ефективност на пресмятанията, подобна хетерогенна архитектура е обещаващ кандидат за провеждане на тези важни и ресурсоемки симулации.

Разработката на услугата GEANT@MIC и изследванията на производителността на пакета на хетерогенна архитектура, както и общите проблеми на пренос на данни между мащабни научни инфраструктури и HPC-екосистемата са отразени в три публикации:

(1) E. Atanassov, M. Barth, M. Byckling, V. Codreanu, N. Ilieva, T. Karasek, J. Rodriguez, S. Saarinen, O.W. Saastad, M. Schliephake, M. Stachon, J. Strassburg, V. Weinberg (Ed.)

*BestPracticeGuide – Intel XeonPhi, January 2017*
http://www.prace-ri.eu/best-practice-guide-intel-xeon-phi-january-2017/

(2) N. Ilieva, Z. Kiss, B. Pavlov, G. Szigeti

*UsingGsatellitefordataintensiveproceduresbetweenlarge-scalescientificinstrumentsand HPC*
PRACE Whitepaper WP247 (to appear)

(3) N.Ilieva, B.Pavlov, P.Petkov, andL.Litov

*GEANT4 onLarge Intel XeonPhiClusters: Service ImplementationandPerformance*
PRACE Whitepaper WP257 (to appear)

**II.** Изпълнение на работни задачи по **Работен пакет 7:** ***Създаване и поддръжка на приложения (ApplicationEnablingandSupport)***

### Инсталиране изследване на програмни пакети за молекулна динамика на високопроизводителната компютърна система АВИТОХОЛ

**NAMD и LAMMPS**

При компилирането на програмния пакет NAMD за хибридна архитектура Intel Xeon - Intel Xeon Phi е използваната информацията, която се намира на страницата https://software.intel.com/en-us/articles/lammps-for-intel-xeon-phi-coprocessor

Изтегля се архива на програмния код на NAMD версия 2.11 от страницата
http://www.ks.uiuc.edu/Development/Download/download.cgi?PackageName=NAMD
След това се разархивира:

*tar xf../NAMD_2.11_Source.tar.gz*

Програмния код се намира в директория NAMD_2.11_Source и всички следващи команди се изпълняват в нея (след като се смени директорията с командата "cd NAMD_2.11_Source"). Ако в системните библиотеки няма инсталирани библиотеките fftw и tcl, то от страницата http://www.ks.uiuc.edu/Research/namd/libraries/ се изтеглят файловете fftw-linux-x86_64.tar.gz и tcl8.5.9-linux-x86_64.tar.gz и след това се разархивират в директории, наименувани fftw и tcl, съответно.

*tar -xf fftw-linux-x86_64.tar.gz*
*mv linux-x86_64 fftw*
*tar xf tcl8.5.9-linux-x86_64.tar.gz*
*mv tcl8.5.9-linux-x86_64 tcl*

Преди да се компилира програмния патек NAMD и се свърже с библиотеките е необходимо да се компилира машинно независимата платформа за паралелно програмиране Charm++. Архив на програмния код се намира в директорията с програмния код на NAMD, а именно charm-6.7.0.tar. Charm++ трябва да бъде компилиран съобразно архитектурата и компилаторите , налични на компютърна система. Разархивирането и компилирането, като се отчита спецификата на комуникационната мрежа с протоколите Mellanox Infiniband Non Blocking Networks и компилаторите на Intel, се прави по следния начин.

Задават се променливите на обвивката, които определят пътищата до изпълнимите файлове и библиотеките на желания компилатор. В конкретният случай:

*source /opt/intel/composer_xe_2015.0.090/bin/compilervars.sh intel64*

Разархивира се програмния код
*tar xf charm-6.7.0.tar*
и се променя директорията
*cd charm-6.7.0*
И в тази директорията, се изпълнява програмата build, която конфигурира и компилира системата Charm++
*./build charm++ verbs-linux-x86_64 smp iccstatic --with-production*
След това в "главната" директория на програмния код на NAMD - NAMD_2.11_Source
*cd ..*
се извърша конфигуриране на програмния код на NAMD като се посочи, че копроцесора Intel Xeon Phi да бъде използван в "offload" режим, а начина на комуникиране и компилатора се задават неявно, чрез указване на вече компилирана паралелна платформа Charm++.
*./config Linux-x86_64-icc --with-mic --charm-arch verbs-linux-x86_64-smp-iccstatic*
След като са налични всички необходими библиотеки за свързването на изпълнимия файл на NAMD, чрез програмата make се пристъпва към компилиране и свързване като е необходимо да бъде сменена текущата директория преди това

*cd Linux-x86_64-icc*
*make*

Изпълнимите файлове, нужни за стартиране на моделирането на молекулярната динамика с NAMD, а именно namd2 и charmrun , се намират в текущата директория (Linux-x86_64-icc, която е поддиректория на главната директория на програмния код).

За провеждането на тестове на производителността на NAMD2.11 при използването за различен брой изчислителни възли бе разработен и използван следния bash скрипт за стартиране на задачи, чрез процедурата за отложен старт ( подреждане на задачите за изпълнение в опашка):

---

*#!/bin/bash*
*# броят на използваните изчислителни възли се задава като първи аргумент*
*arg_nodes=$1*

*#директорията, където се намират входните данни*
*inputdir=/home/ppetkov/ribo-vac*

*#входен файл с параметри на МД симулацията за namd2*
*infilename=ribo-box.namd*

```
#задаване на работна директория
wd=$(pwd)/test-2mic-$arg_nodes
mkdir -p $wd

#копиране на параметричния файл в работната директория
cp $inputdir/$infilename $wd

#зареждане на задачата в опашката ( за отложен старт)
qsub -d $wd <<ENDSUB

#PBS -N namd2-2mic-$arg_nodes
#PBS -l nodes=${arg_nodes}:ppn=16

#Създаване на файл, описващ изчислителните възли на които ще бъде изпълнена програмата
nodefile=tmpnodefile
#записване имената на изчислителните възли , на които ще се смята
cat \$PBS_NODEFILE|sort -u > \$nodefile
nodes=\$(wc -l < \$nodefile)

NODES=\$(cat \${nodefile})
# създаване на файл в който се описват избраните изчислителни възли във формат за NAMD
NODELIST=namd2.nodelist
echo "group main" > \$NODELIST
for node in \$NODES
do
  echo "host \$node" >> \$NODELIST
done

#задаване на пътя до библиотеките на intel компилатора
source /opt/intel/composer_xe_2015.0.090/bin/compilervars.sh intel64

#абсолютен път до изпълнимия файл charmrun
CHARMRUN=/home/ppetkov/bin/charmrun

#абсолютен път до изпълнимия файл namd2
NAMD2=/home/ppetkov/bin/namd2

#задаване на броя нишки , които се изпълняват  на всеки изчислителен възел
PPN=31

#стартиране на симулацията като се използват  PPN нишки и два копроцесора на всеки
изчислителен възел

\$CHARMRUN ++verbose ++p \$[\$nodes*\$PPN] ++ppn \$PPN ++nodelist namd2.nodelist \$NAMD2
ribo-box.namd \${infilename} +isomalloc_sync +pemap 1-\$PPN +commap 0 +devices 0,1

ENDSUB
```

---

LAMMPS

При компилирането на програмния пакет LAMMPS за хибридна архитектура Intel Xeon - Intel Xeon Phi са използвани на част от инструкциите, публикувани в статията , която се намира на страницата

https://software.intel.com/en-us/articles/lammps-for-intel-xeon-phi-coprocessor

Задават се променливите на обвивката, които определят пътищата до изпълнимите файлове и библиотеките на желания компилатор. В конкретният случай:
*source /opt/intel/compilers_and_libraries_2016.1.150/linux/bin/compilervars.sh intel64*

Програмния код на пакета LAMPPS еизтеглен от Git-хранилището на ( relice7 Apr 2016).

*git clone git://git.lammps.org/lammps-ro.git mylammps*

Променя се директорята

*cd mylammps/src*

и се конфигурира програмният код

*make yes-asphere yes-class2 yes-kspace yes-manybody yes-misc yes-molecule*
*make yes-mpiio yes-opt yes-replica yes-rigid*
*make yes-user-omp yes-user-intel*

След това се компилира

*make intel_phi*

Името на изпълнимия файл е файл е lmp_intel_phi.

За да се тества на производителността на LAMMPS при използването на различен брой изчислителни възли бе разработен bash скрипт за стартиране на задачи, чрез системата за отложен старт ( подреждане на задачите на опашка):

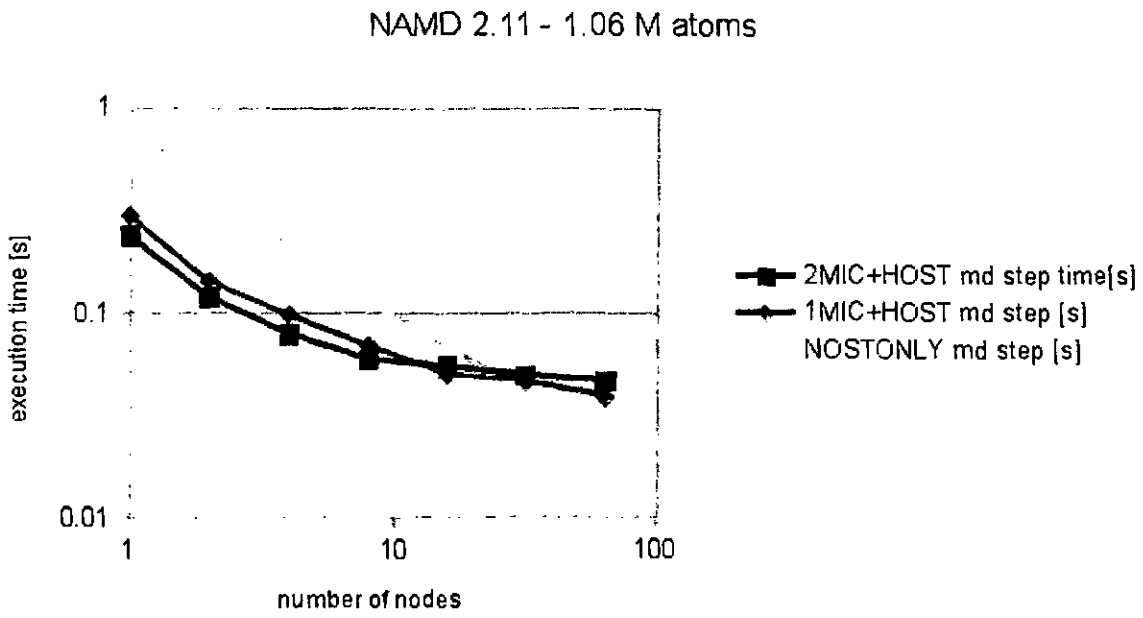
---

```
#!/bin/bash
# броят на използваните изчислителни възли се задава като първи аргумент
arg_nodes=$1

#директорията, където се намират входните данни
inputdir=/home/ppetkov/lammps-test

#входен файл с параметри на МД симулацията за LAMMPS
infilename=in-2mic.rhodo

#задаване на работна директория
wd=$(pwd)/test-2mic-$arg_nodes
mkdir -p $wd

#копиране на параметричния файл в работната директория
cp $inputdir/$infilename $wd

#зареждане на задачата в опашката (за отложен старт)
qsub -d $wd <<ENDSUB
#!/bin/bash

#PBS -N lmp-2mic-$arg_nodes
#PBS -l nodes=${arg_nodes}:ppn=16
```

```
nodefile=tmpnodefile
cat \$PBS_NODEFILE|sort -u > \$nodefile
nodes=\$(wc -l < \$nodefile)
# създаване на файл в който се описват избраните изчислителни възли
rm -f lmpshosts
for n in \$(cat \$nodefile)
do
  echo \$n >> lmpshosts
echo \$n >> lmpshosts
done

nodes=\$(wc -l < lmpshosts)

#абсолютен път до изпълнимия файл charmrun
LAMMPS=/home/ppetkov/bin/lmp_intel_phi

#параметри определящи броя и подреждането на нишките касаещи изпълнението на CPU
export OMP_NUM_THREADS=1
export KMP_AFFINITY=compact

#параметри определящи броя и подреждането на нишките касаещи изпълнението на копроцесора
export MIC_ENV_PREFFIX=PHI
export PHI_OMP_NUM_THREADS=240
export PHI_KMP_AFFINITY=compact

#стартиране изпълнението на LAMMPS
mpiexec.hydra -env I_MPI_FABRICS shm:dapl -n \$nodes -machinefile lmpshosts \$LAMMPS -in
$infilename

ENDSUB
```

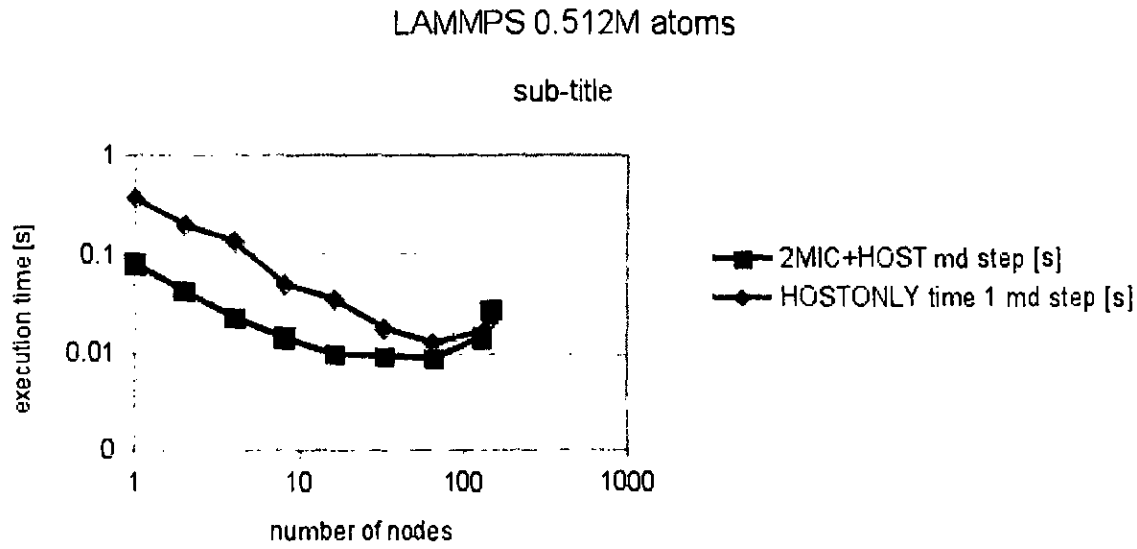
На високопризводителната система "Авитохол" с хибридна архитектура Intel Xeon – Intel Xeon Phi бяха инсталирани програмните пакети NAMD (http://www.ks.uiuc.edu/Research/namd/) и LAMMPS (http://lammps.sandia.gov/). И двете инсталирани програми използват копроцесорите в помощен режим (*offload mode*) като се използва хибриден модел на MPI/OpenMP паралелизиране на изчисленията. Бяха направени тестове на производителността на програмните пакети със атомни системи съдържащи около два милиона частици. Основната величина, която беше измерена, а именно средното време за изпълнение на една итеграциоона стъпка от алгоритъма за интегриране. В конкретният случай алгоритъм на Верле със скорости. Като тестови системи бяха използвани модели на протеини във воден разтвор за изпитанията с LAMMPS и модел на рибозоман единица в случая с NAMD. Бяха измерени времената за изпълнение при използването само на централните процесори и при използването на копроцесори. На Фигура 1 и Фигура 3 са изчертани за времената за изпълнение на една интеграционна стъпка при използването и без използването на копроцесори на копроцесори за LAMMPS и NAMD, съответно.

*Фигура 4: Време за изпълнение на една интеграционна стъпка на програмния пакет NAMD, молекулно динамична-симулация на система, състояща се от 1.06 милиона атома, при използването само на централните процесори (жълта линия), времето за изпълнение при използването на един копроцесора (Intel Xeon Phi) на изчислителен възел (червена линия) и времето за изпълнение при използването на два копроцесора (Intel Xeon Phi) на изчислителен възел в зависимост от броя на изчислителните възли (синя линия) зависимост от броя на изчислителните възли (синя линия).*

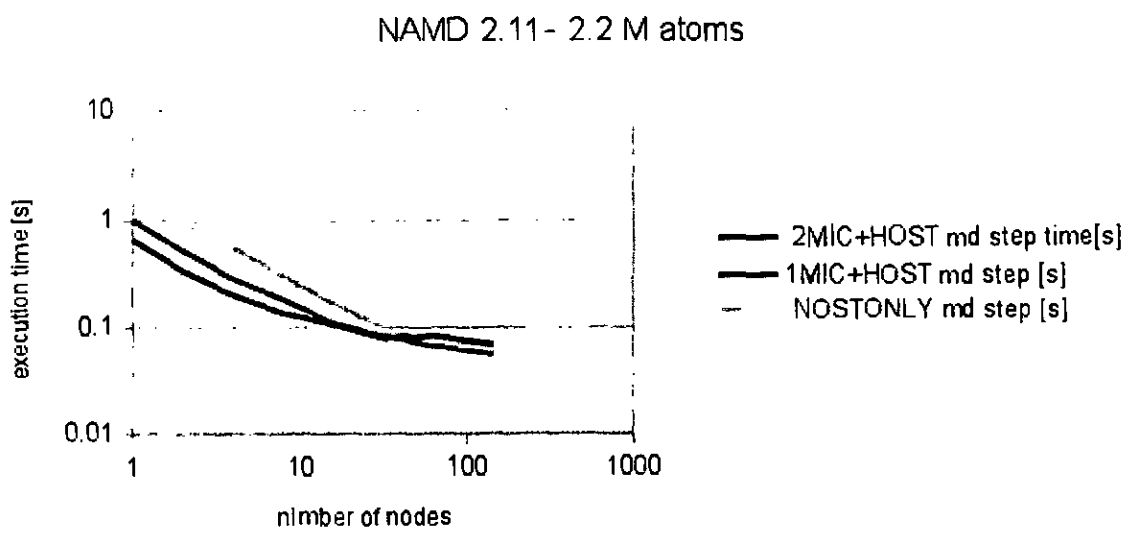



Както се вижда на споменатите графики мащабирането на производителността се влошава при използване на копроцесори. Това може да бъде обяснено с въвеждане на допълнителна латентност в следствие на комуникациите между централните процесори и компроцесорите. Последното се омаловажава от факта, че използването на копроцесорите, ускорява няколко пъти изчисленията. На Фигура 9 и Фигура 11 са показани ускоренията на двата програмни пакета при използването на копроцесори. Както се вижда на Фигура 9, ускорението на производителността намалява с увеличаване на броя на изчислителните възли, но даже и при най-големия брой (146) изчислителни възли производителността с използване на копроцесори е 1.5 пъти за LAMMPS. Поведението на NAMD (виж Фигура 11), обаче е по-лошо и полза от използване на копроцесори при повече от 32 възела не е оправдана за конкретната задача. Максимална производителност се постига при използването на едит копроцесор на възел.

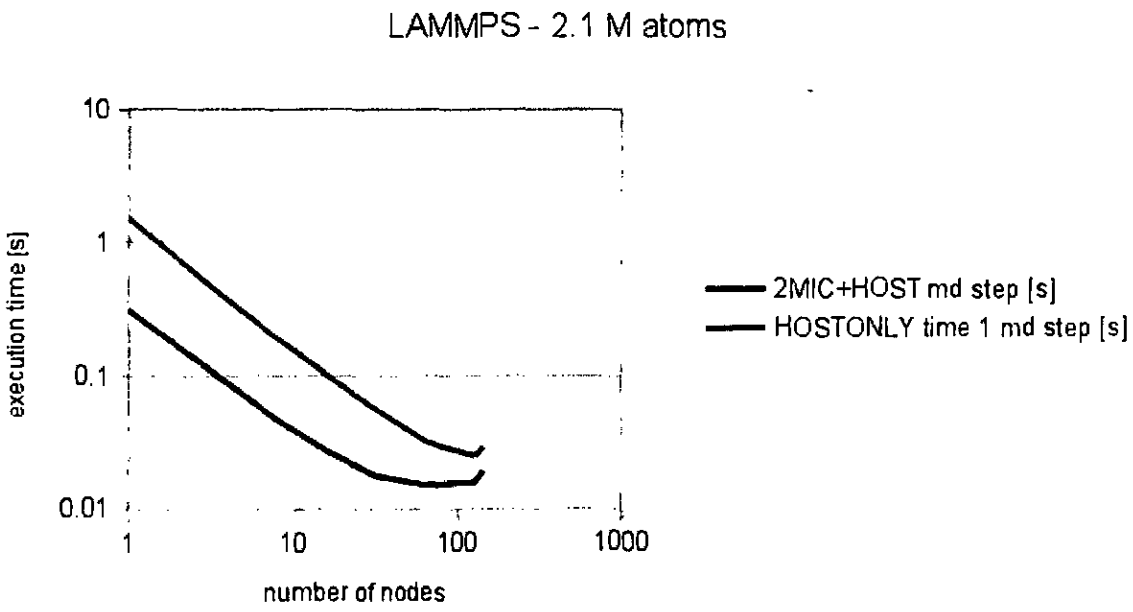
*Фигура 2: Време за изпълнение на една интеграционна стъпка на програмния пакет LAMMPS, молекулно динамична-симулация на система, състояща се от 0.512 милина атома, при използването само на централните процесори (червена линия) и времето за изпълнение при използването на два копроцесора (Intel Xeon Phi) на изчислителен възел в зависимост от броя на изчислителните възли (синя линия).*




*Фигура 3: Време за изпълнение на една интеграционна стъпка на програмния пакет NAMD, молекулно динамична-симулация на система, състояща се от 2.2 милиона атома, при използването само на централните процесори (жълта линия), времето за изпълнение при използването на един копроцесора (Intel Xeon Phi) на изчислителен възел (червена линия) и времето за изпълнение при използването на два копроцесора (Intel Xeon Phi) на изчислителен възел в зависимост от броя на изчислителните възли (синя линия) зависимост от броя на изчислителните възли (синя линия).*

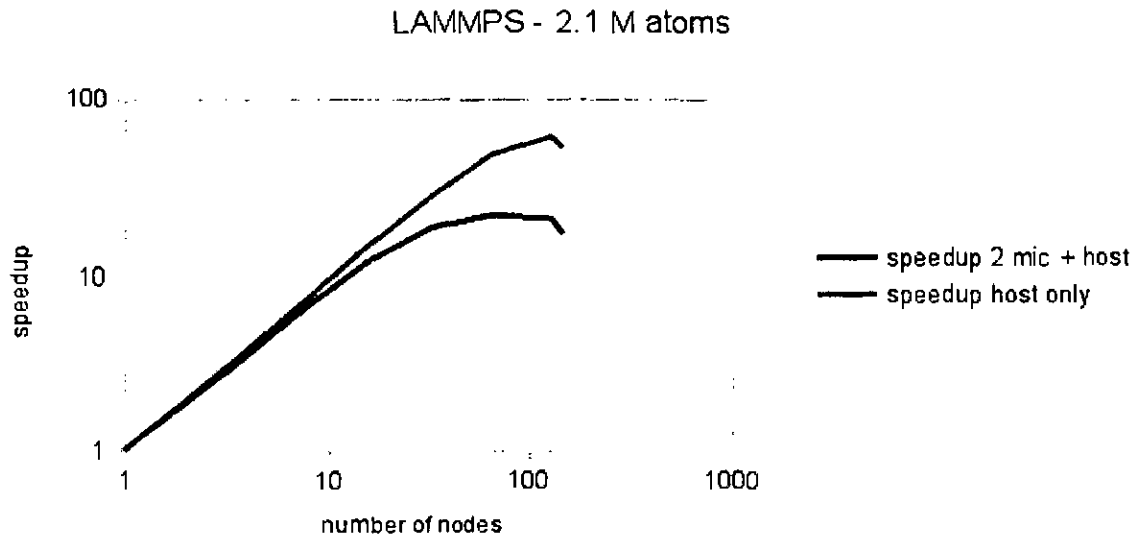


*Фигура 1: Време за изпълнение на една интеграционна стъпка на програмния пакет LAMMPS, молекулно динамична-симулация на система, състояща се от 2.1 милина атома, при използването само на централните процесори (червена линия) и времето за изпълнение при използването на два копроцесора (Intel Xeon Phi) на изчислителен възел в зависимост от броя на изчислителните възли (синя линия).*

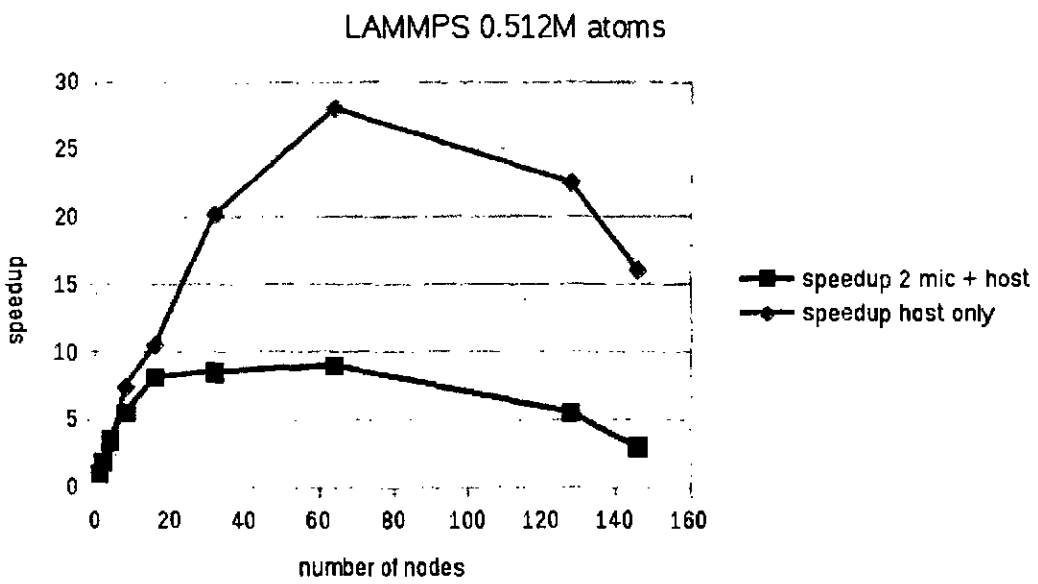



Мащабирането на производителността също е важна характеристика, даваща информация не само за качествата на паралелната компютърна система, но и за комбинацията между хардуерната система и използвания софтуерен продукт. Този параметър показва как се увеличава производителността с увеличаване на използваните изчислителни мощности. Фигура 5 и Фигура 7 е показано как се мащабира производителността на LAMMPS и NAMD, съответно при увеличаване на броя на изчислителните възли в сравнение с тази на един изчислителен възел.

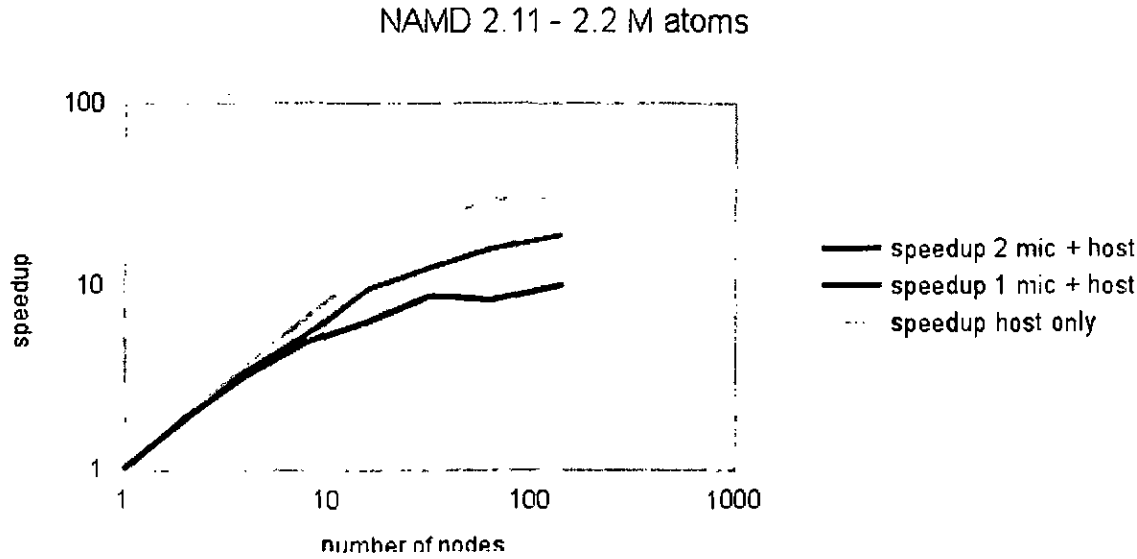
*Фигура 5: Мащабиране на производителността на програмния пакет LAMMPS, молекулно динамична-симулация на система, състояща се от 2.1 милиона атома, при използването само на централните процесори (червена линия) и ускорението при използването на два копроцесора (Intel Xeon Phi) на изчислителен възел в зависимост от броя на изчислителните възли (синя линия).*



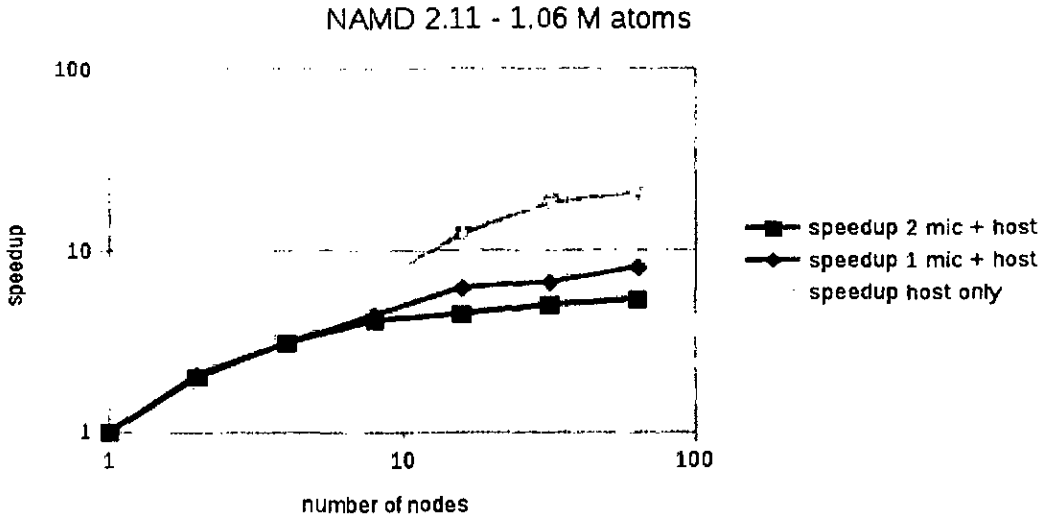

*Фигура 6: Мащабиране на производителността на програмния пакет LAMMPS, молекулно динамична-симулация на система, състояща се от 0.512 милиона атома, при използването само на централните процесори (червена линия) и ускорението при използването на два копроцесора (Intel Xeon Phi) на изчислителен възел в зависимост от броя на изчислителните възли (синя линия).*



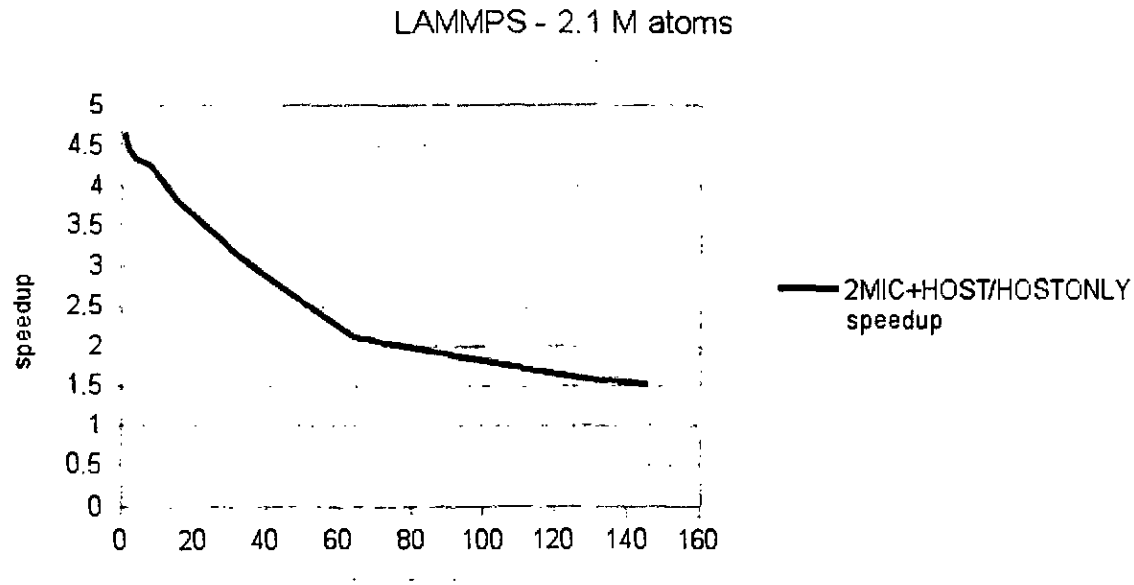
*Фигура 7: Мащабиране на производителността на програмния пакет NAMD, молекулно динамична-симулация на система, състояща се от 2.2 милиона атома, при използването само на централните процесори (жълта линия), ускорението при използването на един копроцесор (Intel Xeon Phi) на изчислителен възел(червена линия) и ускорението при използването на два копроцесора (Intel Xeon Phi) на изчислителен възел в зависимост от броя на изчислителните възли (синя линия).*



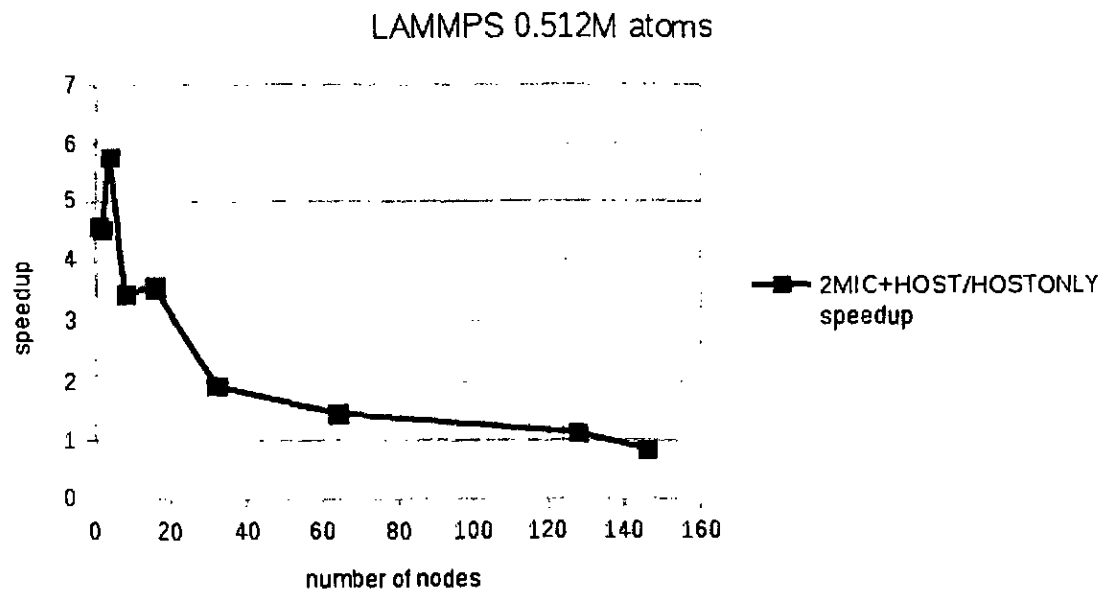

*Фигура 8: Мащабиране на производителността на програмния пакет NAMD, молекулно динамична-симулация на система, състояща се от 1.06M милиона атома, при използването само на централните процесори (жълта линия), ускорението при използването на един копроцесор (Intel Xeon Phi) на изчислителен възел(червена линия) и ускорението при използването на два копроцесора (Intel Xeon Phi) на изчислителен възел в зависимост от броя на изчислителните възли (синя линия).*



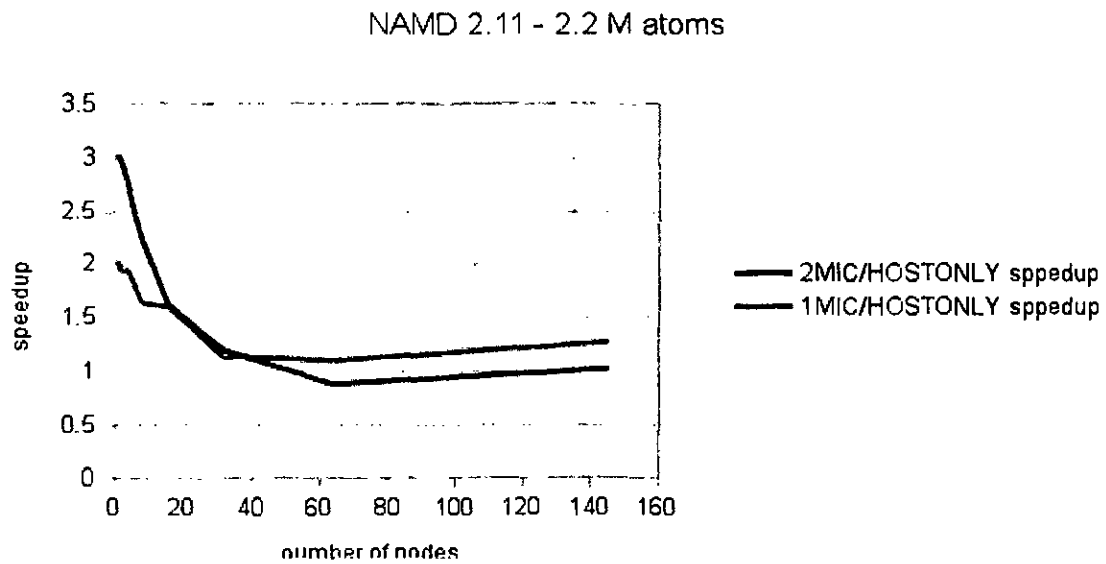
*Фигура 9: Отношение на производителностите при използването на два копроцесора на изчислителен възел и при използването само на централните процесори.*



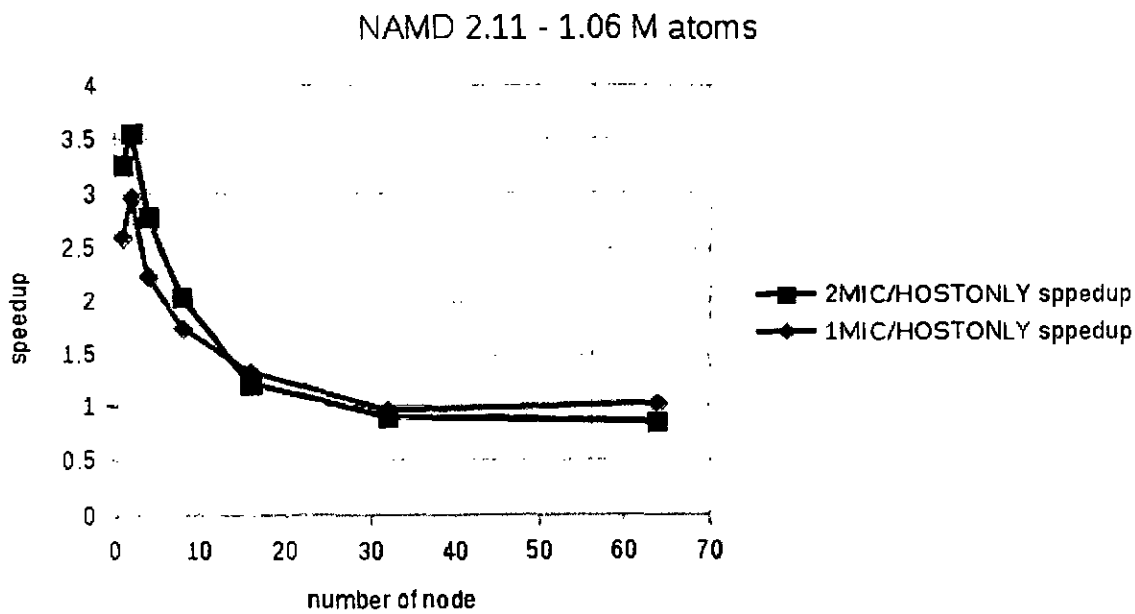

*Фигура 10: Отношение на производителностите при използването на два копроцесора на изчислителен възел и при използването само на централните процесори.*



*Фигура 11: Отношение на производителностите при използването на два копроцесора на изчислителен възел и при използването само на централните процесори (синя линия). Отношение на производителностите при използването на един копроцесор на изчислителен възел и при използването само на централните процесори (червена линия).*




*Фигура 12: Отношение на производителностите при използването на два копроцесора на изчислителен възел и при използването само на централните процесори (синя линия). Отношение на производителностите при използването на един копроцесор на изчислителен възел и при използването само на централните процесори (червена линия).*




**NWChem**

Беше инсталиран програмния пакет NWChem 6.6 ( http://www.nwchem-sw.org/index.php/Main_Page).Бяха добевени необходимнте за инсталацията на пакета библиотеки.

Бяха дадени подхдящите за пространството параметри, за да могат да се използват Xeon Phi ко-процесори. Бяха дадени специфичните потребителски функции и алиаси за да може пакетът да се използва. Софтуерния пакет беше успешно инсталиран и тестван.
Необходими параметри за инсталацията на пакета:

```
export NWCHEM_TOP=/home/YOURHOME/nwchem-6.6 (your directory)
export NWCHEM_TARGET=LINUX64
export ARMCI_NETWORK=OPENIB
export ARMCI_DEFAULT_SHMMAX_UBOUND=65536
export USE_MPI=y
export NWCHEM_MODULES=all python
export USE_MPIF=y
export USE_MPIF4=y
export MPI_HOME=$I_MPI_ROOT/intel64
export MPI_INCLUDE="$MPI_HOME"/include
export MPI_LIB="$MPI_HOME"/lib
export LIBMPI="-lmpi -lmpifort -lmpigi -lrt -lpthread -ldl"
export MKLROOT=/opt/intel/compilers_and_libraries/linux/mkl
export SCALAPACK_LIB=" -mkl -openmp -lmkl_scalapack_ilp64 -lmkl_blacs_intelmpi_ilp64 -lpthread -
lm"
export SCALAPACK="$SCALAPACK_LIB"
export LAPACK_LIB="-mkl -openmp  -lpthread -lm"
export BLAS_LIB="$LAPACK_LIB"
export BLASOPT="$LAPACK_LIB"
export USE_SCALAPACK=y
export SCALAPACK_SIZE=8
export BLAS_SIZE=8
export LAPACK_SIZE=8
export PYTHONHOME=/usr
export PYTHONVERSION=2.6
export PYTHONLIBTYPE=so
export USE_PYTHON64=y
export USE_CPPRESERVE=y
export USE_NOFSCHECK=y
# Xeon Phi
export USE_OPENMP=1
export USE_OFFLOAD=1
#make FC=ifort
# Run variables
export MIC_USE_2MB_BUFFER=16k
export ARMCI_OPENIB_DEVICE=mlx4_0
```

Download new patches for NWChem 6.6:

```
cd $NWCHEM_TOP
wget http://www.nwchem-sw.org/images/Tddft_mxvec20.patch.gz
gzip -d Tddft_mxvec20.patch.gz
patch -p0 < Tddft_mxvec20.patch
```

Команди за компилиране:

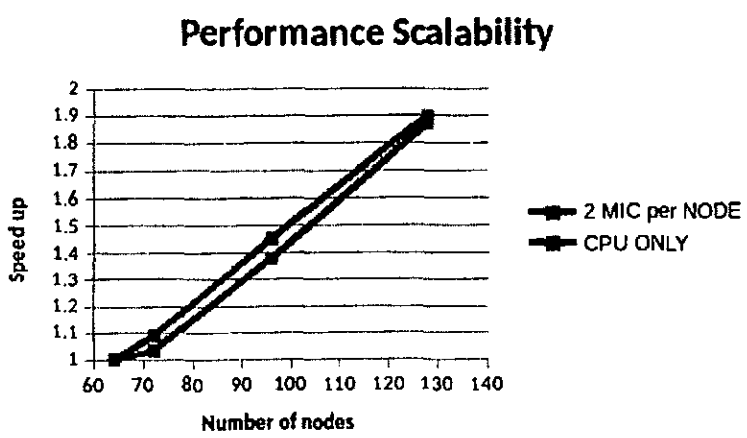
```
cd $NWCHEM_TOP/src
make realclean
make FC=ifort CC=icc AR=xiar
```

Put in the file .bashrc the following paths:

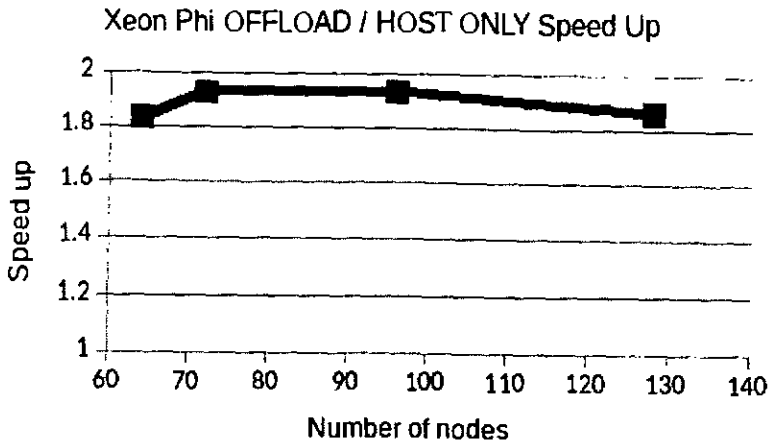
```
# Source global definitions
if [ -f /etc/bashrc ]; then
    . /etc/bashrc
fi

# Специфични потребителски функции:
export I_MPI_LIBRARY_KIND=release
[ -e /opt/intel/compilers_and_libraries_2016.2.181/linux/mpi/intel64/bin/mpivars.sh ] && \
    source /opt/intel/compilers_and_libraries_2016.2.181/linux/mpi/intel64/bin/mpivars.sh
[ -e /opt/intel/compilers_and_libraries_2016.2.181/linux/bin/compilervars.sh ] && \
 source /opt/intel/compilers_and_libraries_2016.2.181/linux/bin/compilervars.sh intel64
export I_MPI_USE_DYNAMIC_CONNECTIONS=1
export I_MPI_DYNAMIC_CONNECTION_MODE=dapl
export I_MPI_FABRICS=shm:dapl
export I_MPI_MIC_POSTFIX=.mic
export I_MPI_DAPL_UD=enable
```

На фигури 13 и 14 са показани съответно мащабиране на производителността и ускорението при използване на копроцесор Xeon Phi на системата за високопроизводителни изчисления АВИТОХОЛ.

*Фигура 13: Ускорение при симулиране на пентацен с метода CCD(T) при използване само на централните процесори (червена линия) и при използване на копроцесори (синя линия)*



*Фигура 14: Отношение на производителността с и без използване на копроцесори в зависимост броя на изчислителните възли.*




**Инсталиране и оптимизиране на програмния пакетOpenFOAM на високопроизводителната компютърна система АВИТОХОЛ**

Целта на дейността беше да се приспособи OpenFOAM софтуерът (http://openfoam.org/), така че да може да се използва на ускорители от тип Xeon Phi. За тачи цел използвахме PRACE системата Авитохол, която се състои от 150 сървъри, всеки снабден с двойна Intel Xeon E5-2640 v2 процесори, 64 GB памет, два Intel Xeon Phi 7120P съ-процесори с 16 GB RAM всеки, всички свързани с неблокираща връзка InfiniBand с ниска латентност.

Тъй като само компилаторът Intel поддържа векторизиране на Xeon Phi, решихме да правим изследвания само с този компилатор.

Първоначално започнахме с немодифициран изходен код и се опитахме да го компилираме за Xeon Phi. Открихме, че флаговете по подразбиране създават два различни проблеми. На първо място, използването на флага

-fp-trap = common

води до segmentation fault. Това не е нещо специфично за OpenFOAM. По този начин този флаг, който е полезен за обикновено CPU, се оказва пречка. Освен това, използването на

-fp-model precise

довежда до това да не се използва векторизация, а да се използват инструкции x87. Такъв тип операции с плаваща запетая не са силната страна на Xeon Phi, затова решихме да премахнем този флаг и да се опитаме да следим за проблеми във фазата на валидиране.

Тъй като OpenFOAM кодът поддържа само MPI (а не OpenMP), извършихме опити само в native и симетричен режим.

Нашите първоначални сравнителни тестове на кода върху MIC с кода на процесора с помощта на стандартни тестови задачи (кухина) показа, че извършеното от компилатора автоматично векторизиране не е от полза. Като цяло едно ядро на Xeon Phi извършва изчисленията около 6 пъти по-бавно от едно ядро на процесора. При провеждане на паралелни тестове, ситуацията се подобрява за Xeon Phi, тъй като картата Xeon Phi в native режим става конкурентоспособна спрямо процесора. Заради нашето наблюдение, че векторизирането не се използва ефективно, започнахме да разгледаме най-важните части от кода, които допринасят за общото време за изпълнение и какво може да се направи, за да се подобрят резултатите. Ще отбележим, че в

https://www.nersc.gov/assets/Uploads/2aBoF-ISC2015-OpenFOAM.pdf

се разглежда въпросът за оптимизиране на функцията symGaussSeidel. Ето защо ние се концентрирахме върху други важни части, като PCG с DIC preconditioner.

Открихме циклите, които допринасят най-много за времето за изпълнение.

Отделно от изходните файлове DICPreconditioner.C и PCG.C също така е възможно да се оптимизират операциите blduMatrixATmul.C например.
Във всички случаи, това, което направихме беше да разгледаме най-важните цикли и оценка на резултатите от добавянето на някои pragma и други трикове. Пример за оптимална комбинация имаме в DICpreconditioner.C:

```
#pragma unroll(16)
#pragma noprefetch
 for (cell=0; cell<nCells-48; cell++)
   {
      _mm_prefetch((char*)&rAPtr[cell+48],_MM_HINT_T1);
      _mm_prefetch((char*)&rDPtr[cell+48],_MM_HINT_T1);
      wAPtr[cell] = rDPtr[cell]*rAPtr[cell];
      _mm_prefetch((char*)&rAPtr[cell+16],_MM_HINT_T0);
      _mm_prefetch((char*)&rDPtr[cell+16],_MM_HINT_T0);
   }
```

Този цикъл изглежда праволинеен, но оптимизацията по подразбиране не беше много добра. В много случаи наблюдавахме, че добавянето на
#pragma SIMD
доведе до различни резултати, така че решихме да добавим
#pragma novector
с цел да осигурим коректно изпълнение. Така например, в lduMatrixATmul.C имаме следните цикли:

```
#pragma unroll(32)
 #pragma novector
   for (label cell=0; cell<nCells; cell++)
   {
      ApsiPtr[cell] = diagPtr[cell]*psiPtr[cell];
   }
.....
   #pragma unroll(32)
   #pragma novector
   for (label face=0; face<nFaces; face++)
   {
      ApsiPtr[lPtr[face]] += upperPtr[face]*psiPtr[uPtr[face]];
      ApsiPtr[uPtr[face]] += lowerPtr[face]*psiPtr[lPtr[face]];
   }
```

Чрез използването на този подход стана възможно да се постигне подобрение повече от 30% по някои показатели. В края на работата си достигнахме до следния подход: можем да заменим тези статични ръчно настроени редове макроси, което зависят от входни променливи от включен (include) файл. След това можем да стартираме нелинеен оптимизатор, който може да се опита и да намерите най-добрите входове за макросите.
Пример на резултати от 20 стъпки на теста за кухина с 200x200x200 дискретизиране, с малка стъпка по времето 0.00001, така че PCG с DIC preconditioner доминира времето за изпълнение:

16 ядра (две CPU) - ExecutionTime = 1136.52 S

240 ядра (два Xeon Phi, използвайки HyperThreading) - ExecutionTime = 611.61 S.

Опциите за MPI, използвани на Xeon Phi са важни:

I_MPI_PIN_ORDER=scatter

I_MPI_PIN_DOMAIN=core
I_MPI_PIN=1
I_MPI_DAPL_UD = 1
I_MPI_ADJUST_ALLREDUCE=5
I_MPI_DAPL_SCALABLE_PROGRESS=1
I_MPI_FABRICS=shm:dapl
I_MPI_DYNAMIC_CONNECTION=1
I_MPI_SPIN_COUNT=1
Например, промяна на I_MPI_ADJUST_ALLREDUCE с някаква друга стойност води до значителен спад в производителността.
Някои резултати от тестове, като използват два възела (4 Xeon Phi):
Непроменениият код - 20 стъпки, използвайки 4 Xeon Phi карти (общо 480 MPI процеса, 2x HyperThreading) 548.73s
Подобрен код - 20 стъпки, използвайки 4 Xeon Phi карти (общо 480 MPI процеса, 2x HyperThreading) 459.44s
Открихме, че в тези конфигурации 2x HyperThreading (120 нишки на всяка карта) даде най-добри резултати.
OpenFOAM работи успешно и в симетричен режим (CPU+XeonPhi). Чрез decomposeParDict може да посочите различни тегла за процесорите, при използване на scotch за разлагането. В момента, тъй като все още не сме направили ръчна векторизация, ядрата на Xeon Phi извършват изчисления при около 1/6 от скоростта на ядрата на процесора. Когато работи в паралел, е необходимо да се заложи още по-голяма разлика в теглото, около 1/8.
В момента софтуерът за OpenFOAM е инсталиран в системата Авитохол и може да се използва от потребителите.

**III.** Изпълнение на работните задачи по **Работен пакет №4 (РП4): Обучение**

През 2016г., сдружение Национален Център за Суперкомпютърни приложения организира два обучителни курса в рамките на работния пакет:

1. **Code Modernisation for INTEL Multi Core and Xeon Phi Architectures – 25-28 Април 2016 (Модернизация на кодове за архитектури с INTEL Multi Core and Xeon Phi)**

*Събитие:* Курсът беше организиран от НЦСП с помощта на INTEL (UK) и Science and Technologies Facilities Council (STFC, UK). Училището се проведе в Института по Информационни и комуникационни технологии към БАН.

*Цели:* Училището беше насочено към студенти, Магистъри, докторанти и млади изследователи със специалности информатика и изчислителни науки, които имат интереси в прилагането на нови суперкомпютърни технологии в техните научни изследвания.

*Съдържание:*
Курсът представи иновативни софтурни подходи, нужни за следващото поколение суперкомпютри с високо съдържание на микропоцесори, хомогенни (INTEL Xeon) и хибридни, с ускорителни ко-процесори (INTEL Xeon Phi). Програматасъдържаше лекции и упражнения, които показаха важни аспекти за създаването на нови суперкомпютерни програми, както и в ефективното реформатиране на вече съществуващи софтуерни продукти. INTEL предостави двама лектори,както итренировъчни лаптопи с два INTEL XeonPhi сървъра и мрежо впревключвател.

*Преподаватели:*

- Стивън Блеър-Чапел (Stephen Blair-Chappell INTEL, Великобритания)
- Виктор Гамаюнов (Victor Gamayonov INTEL, Великобритания)
- Д-р. Алин-Марин Елена (Dr. Alin-Marin Elena, STFC Daresbury Laboratory, Великобритания )
- Д-р Майкъл Сийтън (Dr. Michael Seaton, STFC Daresbury Laboratory, Великобритания)
- Д-р Илиян Тодоров (Dr. IlianTodorov, STFC Daresbury Laboratory, Великобритания )
- Д-р Пейчо Петков НЦСП, България

*Ресултат:*Представените методики на софтуерно инжинерство за високопродуктивни езици бяха съпътствани от традиционни лекции за паралелно програмиране,за да позволят и затвърдят приложението, развитието чрез постояннат амодернизация на научните софтуерни пакети,които се нуждаят от постоянна поддръжка на комплексните и бързоразвиващите се суперкомпютърни архитектури.)

Обуителният курс завършиха над 30 научни работници, преподаватели и студенти от БАН, ТУ-София, Факултета по Физика – СУ. Лекциите и упражненията бяха предавани по видео връзка и на 25 студента отТехническия Университет в Пловдив.Имаше присъствие и от софтуерни инжинери от индустрията (Rila Solutions, България).

Училището се оказа много популярно и получи висок рейтинг в анкетата, проведена след края на курса.Тя показа необходимост от последващ курс за затвърждаване на научените методики, тяхното усъвършенстване и разширяване на професионалния кръгозор, чрез по-доброто разбиране на нуждите на софтуерните инжинери в институтите и университетите.

## 2. Enabling Software Scalability and Performance on INTEL Xeon and Xeon Phi Platforms – 15-17Декември 2016

3.

*Събитие:* Курстът беше организиран отново от НЦСП със съдействието на INTEL (UK), Bayncore (UK), Science and Technologies Facilities Council (STFC, UK) и Rila Solutions (България) и се проведе в Института по Информационни и Комуникационни Технологии към БАН.

*Цели:* Курсът беше създаден специално за обучение на програмисти от индистрията и изследователския сектор с интереси в прилагането на нови суперкомпютърни технологии в научните си изследвания.

*Съдържание:* Курсът беше насочен към нуждите на софтуерните инжинери за специфични познания за компилатори, библиотеки, инструменти и съпътстващи пособия, с цел извичане на максимална производителност и скалирне на техните проекти с програмни пакети.
Програмата следваше познатия от преди формат от лекции и упражнения. INTEL (UK), Bayncore (UK) и Rila Solutions (България) предоставиха по един преподавател, като INTEL (UK) също предостави и трениривъчни лаптопи с два INTEL XeonPhi сървъра и мрежов превключвател.

*Преподаватели:*

- Стивън Блеър-Чапел (Stephen Blair-Chappell INTEL, Великобритания)
- Виктор Гамаюнов (Victor Gamayonov INTEL, Великобритания)
- Д-р. Алин-Марин Елена (Dr. Alin-Marin Elena, STFC Daresbury Laboratory, Великобритания )
- Д-р Илиян Тодоров (Dr. IlianTodorov, STFC Daresbury Laboratory, Великобритания )
- Д-р Валентин Павлов Рила Солюшънс, България

*Резултати:* Бяха представени редица специални методики, техники, библиотеки, инструменти и др. за INTEL XeonPhi архитектури.
Обуителният курс завършиха над 30 научни работници, преподаватели и студенти от БАН, ТУ-София, ТУ-Пловдив, Факултет по Физика – СУ и един участник от. Македонската Академия на

Науките, както и от софтуерни инжинери от индустрията (Rila Solutions, България).

Училището отново се оказа много сполучливо и получи високи оценки в анкетата, проведена сред участниците след края на занаятията.Тя показа необходимост от последващо училище, където да се покажат нагледно методиките, приложени към софтуерни проекти в научните изследвания.

**IV.** Изпълнение на работни задачи по **Работен пакет №3 (РП3): „Разпространение на резултатите по проекта и комуникация"**

Екип от НЦСП участва в дейностите на WP3 web team, отговарящ за поддръжката и информационното обслужване на web портала на PRACE-RI(www.prace-ri.eu). Извършените дейности включват:

- Поддържане на активни контакти и кореспонденции с членовете на екипа.
- Участие в редовните телеконферентни работни срещи и дискусии относно планираните дейности и тяхното изпълнение.
- Поддържане на актуалността на публикуваното информационно съдържание в web порталана PRACE-RI.
    - Архивиране и премахване на остаряло информационно съдържание
    - Премахване на информация за дублицирани PRACE събития в порталите на PRACE-RI и PRACE TrainingPortal(http://www.prace-ri.eu/prace-training-events/), получена в резултат на установена неправилна работа на feed модула на CMS WordPress платформата обслужваща портала на PRACE, грижещ се за координацията между нея и платформата, обслужваща PRACE Trainingпортала.
        - Прегледани и премахнати над 3000 дублицирани събития.
- Редактиране на съществуващи и публикуване на нови информационни материали под формата на web обяви за различни PRACE инициативи и събития в *Announcements, Partners Trainings, HPC related events, Job vacancies, .*
- Категоризиране, подготовка за web и публикуване на White papers (http://www.prace-ri.eu/white-papers/):
    - WP223 Enabling Space Filling Curves parallel mesh partitioning in Alya
    - WP226 Data Centre Infrastructure Monitoring
    - WP228 Hybrid iterative-direct solution strategy for improving the scalability of nuclear waste management simulations
    - WP229 High order finite element schemes and domain decomposition solvers for large-scale simulations in electromagnetics
    - WP230 Reducing latency and bandwidth costs in parallel sparse linear solvers
    - WP231 Performance portability of OpenCL with application to Neural Networks
    - WP232 GPU Simulations of Violent Flows with Smooth Particle Hydrodynamics (SPH) Method
    - WP233 Stellar Atmosphere Simulation code Bifrost on Intel Xeon Phi Knights Landing
    - WP234 Study of Xeon Phi Performance of a Molecular Dynamics Proxy Application
- Подготовка за web и публикуване на Best Practice Guides(http://www.prace-ri.eu/best-practice-guides/)

- Изработване на графични файлове с логота, банери и др. визуална информация, предназначена за публикуване в PRACE портала.

- Публикуване на информация и ръководства за използване на изчислителите ресурси на високопроизводителна хетерогенна изчислителна система "Avitohol" чрез портала на НЦСА (http://scc.acad.bg/ncsa/index.php/bg/hpc/2016-08-09-11-32-53)

- Обновяване и допълване на ресурсите на библиотека "Суперкомпютрите в науката, високите технологии и практиката" (http://scc.acad.bg/ncsa/index.php/bg/80-libraries-and-supercomputer-applications/214-lib-supercomputers-in-science)

Участие в общи дейности по проекта PRACE-4IP

През изминалата година членове на сдружението работиха съвместно с колеги от редица европейски суперкомпютърни центрове и научни институти – участници в проекта PRACE-4IP по изпълнението на общата работна програма на проекта. Във финалните документи, представени на Европейската Комисия, е отчетено участието ни в изготвянето на

- Deliverable D2.4: Stakeholder Management in PRACE 2.0

- Deliverable D6.4: Deployment of Prototypal new Services

Също така, на член на сдружението беше възложено вътрешното рецензиране на

- Deliverable 7.2: Final Report on Applications Enabling

В заключителна фаза е работата по

- Deliverable 2.3: Management Processes and Tools,

отново с участието на представител на НЦСП.

Всичко това доказва авторитета, с който е ползват сдружението и неговите членове сред експертите по високопроизводителни пресмятания в Европа.

## Резултати от работата на Националния център за суперкомпютърни приложения и цели за постигане през 2017 година

Националният център за суперкомпютърни приложения (НЦСП) улеснява и насърчава използването на съвременни методи за паралелна обработка на информацията. Изтъкнатите учени и специалисти от няколко университета, софтуерни фирми и институти на Българската Академия на Науките участват в изпълнението на проектите PRACE 4IP.

Специално внимание се отделя на приложенията на суперкомпютрите в изчислителната химия, в изследването на наноструктури, молекулярната и клетъчна биология, фармацията, медицината, моделирането на динамиката на флуиди, опазването на околната среда и климатичните промени, индустрията, финансите и др.

Голяма част от усилията на НЦСП са да осъществи мост между научните и научно-приложните суперкомпютърни разработки към директните приложения в индустрията и да бъде място за обсъждане и намиране на отговор на множеството въпроси свързани с развитието и приложенията на високопроизводителните компютърни архитектури и съответния софтуер за тях, от една страна, и приложенията при решаване на задачи от практиката, от друга страна.

Ресурсите на Българския суперкомпютърен център се използват в момента за изследвания в областта на математиката, физиката и химията. Някои от конкретните теми, по които се работи, са:

В съответствие с изискванията на чл.40 ал.2 от Закона за юридическите лица с нестопанска цел, ръководството е задължено да посочи в годишния доклад размера на безвъзмездно полученото имущество, вида, размера, стойността и целите на получените и предоставени дарения, както и данни за дарителите. В унисон с целите, набелязани при учредяването, в годината на създаването си Сдружението е получило за безвъзмездно използване суперкомпютър, собственост на българската държава. Няма други получени или предоставени дарения, които следва да бъдат посочени в годишния доклад на Сдружението.

В годишния финансов отчет на НЦСП са посочени реализираните приходи общо за 365 хил. лева (2015: 179 хил. лева): от осъществената стопанска дейност в размер на 0 хил. лева (2015: 0 хил. лева) и от нестопанска дейност в размер на 365 хил. лева (2015: 179 хил. лева). Извършените във връзка с обезпечаването на тезн дейности разходи са общо за 365 хил. лева (2015: 179 хил. лева), от които 0 хил. лева (2015: 0 хил. лева) за стопанската дейност и съответно 365 хил. лева (2015: 179 хил. лева) за нестопанската дейност. Реализираният финансов резултат е загуба 0 хил. лева (2015: загуба 0 хил. лева). Източник на загубата за текущия период са начислените разходи за амортизации на дълготрайните материални активи. В съответствие с Устава на Сдружението, резултатът следва да се отнесе в резерви.

**Управление на финансовия риск**

В условията на действаща по време на отчетния период световна финансова криза дейностите на Сдружението предполагат редица финансови рискове: пазарен риск (включително лихвен риск), кредитен риск, ликвиден риск и др. Общата програма на Сдружението за управление на риска се фокусира върху контрола за събиране на вземанията от контрагенти и непредвидимостта на финансовите пазари, като цели да сведе до минимум потенциалния неблагоприятен ефект върху финансовото представяне на Сдружението. Управлението на риска се извършва, като се идентифицират и оценяват финансовите рискове в т.ч. и такива рискове, покриващи специфични области, като лихвен риск, кредитен риск, използване на различни финансови инструменти, както и инвестиране на свободните парични средства.

**Отговорност на ръководството**

Задължение на Ръководството на Сдружението е в съответствие с българското законодателство да подготви финансов отчет, който да представя вярно и честно финансовото състояние, финансовите резултати и паричните потоци на Сдружението в съответствие с Националните счетоводни стандарти (НСС).

Ръководството на Сдружението потвърждава, че сегашният финансов отчет, изработен в съответствие с НСС, е изготвен според счетоводните политики на Сдружението, нормативните и правните изисквания и принципите на действащото предприятие и последователност. Всички начисления и провизии се извършват в резултат на консервативна оценка, реално представяне и последователност.

Ръководството на Сдружението потвърждава, че са съблюдавани всички изисквания на приложимите счетоводни стандарти при подготовката на финансовия отчет.

Ръководството е отговорно за представянето на резултатите, запазването на собствеността и интересите на Сдружението, както и за предприемането на необходимите мерки за избягване и откриване на евентуални злоупотреби и други нередности.

Вярваме в успеха на дейността на Сдружението и използваме възможността да изразим лоялността си към нашите партньори и персонал. Очакваме да получим още по-окуражаващи резултати в бъдеще.

СДРУЖЕНИЕ
НАЦИОНАЛЕН ЦЕНТЪР
ЗА СУПЕРКОМПЮТЪРНИ
ПРИЛОЖЕНИЯ

.....

проф. дтн Стоян Марков
Председател на Управителен съвет
Сдружение „Национален център за суперкомпютърни приложения"

24 февруари 2017 г.,
София