"Авитохол"

Упътване за потребителя

Съдържание от:

ИИКТ-БАН

Съставено от: http://www.hpc.acad.bg/documentation

Редактор: Емануил Атанасов

ИИКТ-БАН

За актуализации, моля, проверете http://www.hpc.acad.bg

Съдържание

1.Въведение

През юни 2015 г. суперкомпютър Avitohol беше внедрен в ИИКТ-БАН. Клъстерът се състои от 150 HP Cluster Platform SL250S GEN8 сървъра.

Всеки изчислителен възел има 2 Intel Xeon CPU E5-2650 v2 @ 2,6 GHz (8 ядра всеки), 64 GB RAM и 2 Intel Xeon Phi 7120P копроцесора (с 61 ядра всеки). Така общо има 20700 ядра.

Освен това има 4 входно/изходни възела за обслужване на 96 TB дисково пространство за съхранение на данни. Името на системата, внедрена от HPE, е Avitohol.

Avitohol има следните възможности: теоретична пикова производителност от 412,32 TFlop/s, обща памет 9600 GB и общо дисково пространство с големина 96 TB.

Той е класиран на 389-то място в листа Топ 500 от ноември 2015 г. (http://www.top500.org) с теста LINPACK постигна производителност от 264,2 TFlop/s.

Фигура 1. Система АВИТОХОЛ

2. Архитектура и конфигурация на системата

2.1 Системна конфигурация

Текущата системна конфигурация включва:

- 150 сървъра с двоен сокет HP ProLiant SL250S Gen8
- 1 от тях запазен за влизане/подаване на задачи, тестване и разработване на приложения
- 4 входо/изходни възли HP ProLiant DL380p Gen8 с 2 Intel Xeon CPU E5-2650 v2, 64GB RAM, Fibre Channel карти
- 2 възела за управление HP ProLiant DL380p Gen8 с 2 Intel Xeon CPU E5-2650 v2, 64GB RAM
- 96 TB дисково пространство с онлайн достъп, свързано с Fibre Channel на входо/изходните възли.
- Неблокираща 56Gbps FDR InfiniBand мрежа, свързваща всички възли.

Конфигурация на изчислителните възли:

- Процесори: двоен Intel Xeon 8-core CPU E5-2650 v2 @ 2.6 GHz
- Копроцесори: два копроцесора Intel Xeon Phi 7120P, 16 GB RAM и 61 ядра на всеки един.
- Памет на ускорителите: 16 GB (общо 4.8 TB)
- Основна памет на изчислителните възли: 64 GB (общо 9.6 TB)

Операционна система

Операционната система на Avitohol е Red Hat Enterprise Linux версия 6.7. Има подкълстер с версия 8.

Копроцесорите работят с MPSS версия 3.6-1.

2.2. Файлови системи

Файлова система, която е за четене и запис за всички потребители, е файловата система `/home`, която е от тип Lustre. Групи от потребители могат да поискат да имат споделена директория, така че да могат да споделят данни помежду си.

Част от софтуера, който е достъпен за всички потребители, е достъпен в директорията `/opt`. Тази файлова система се споделя чрез NFS и е само за четене от потребителите.

Файловата система Lustre е оптимизирана за обработка на големи файлове. Това означава, че създаването на голямо количество малки файлове може да бъде относително бавен процес. Има възможност да се разпредели между няколко сървъра за паралелно I/O.

Допълнителна информация

За повече информация относно файловата система Lustre справка с документацията за Lustre,

https://doc.lustre.org/lustre_manual.pdf

Вижте например,
https://wiki.hpdd.intel.com/display/PUB/HPDD+Wiki+Front+Page

Допълнителни детайли

Приложенията с високи изисквания на входно-изходни операции, могат да поискат специално дисково пространство, което може да бъде осигурено чрез други файлови системи.

3. Достъп до системата

3.1 Отдалечен достъп

Входния възел за достъп до Авитохол е gw.avitohol.acad.bg. Ключовете за достъп са:

1024 cd:97:16:41:2c:cd:3d:59:c4:f0:d2:67:ce:1c:5e:62 /etc/ssh/ssh_host_dsa_key.pub (DSA)

2048 b9:e3:6f:f5:a2:ea:8f:95:0a:96:5b:55:03:0c:bf:e9 /etc/ssh/ssh_host_rsa_key.pub (RSA)

От съображения за сигурност е препоръчително потребителите да използват ключове с поне 2048 бита.

За достъп се използва връзка по протокол ssh.

Входния възел има идентична хардуерна и софтуерна конфигурация с възлите за изпълнение. Само този възел може да се използва за стартиране на задания.

След като влезе в клъстера, потребителят може да влезе с ssh към всеки от възлите за изпълнение, където той или тя има работеща задача(`job`), или към ускорителите Xeon Phi, които са свързани с такива възли.

3.2. Използване на изчислителни ресурси на Авитохол

Входният възел **gw.avitohol.acad.bg** е предназначен основно за редактиране, компилиране и пускане на изчислителни задачи. Разрешено е кратки задачи за тестване на паралелни програми. В случай на съмнение потребителите се насърчават да изпратят задачи, изискващи един цял изчислителен възел и да го използват за отстраняване на грешки и разработка. Възлите за изпълнение са с наименования от sl001 до sl150.

3.3. Интерактивно отстраняване на грешки

Ако трябва да дебъгвате вашия програмен код, можете да влезете във възела **gw.avitohol.acad.bg** и да изпълните кода си интерактивно. Този възел има два процесора и два копроцесора Xeon Phi по същия начин както възлите за изпълнение.

Моля, наблюдавайте използването на ресурсите и в случай, че системата се претовари (може да се види от програмата с команда `/top`), спрете стартиранията за отстраняване на грешки и ги изпълнете чрез подаване на задание на възел за изпълнение, където ще получите специален достъп. Потребителите, които причиняват системни сривове на възела или системни сривове и спирания на копроцесорите Xeon Phi, трябва да се опитат да разрешат проблемите с техните кодове, като използват възли за изпълнение, разпределени от система за управление на задачи. Ако проблемите продължават, те се насърчават да се свържат със системните администратори.

Имейлът за поддръжка е: **avitohol-support@parallel.bas.bg.**

3.4. Допълнителна информация

Има и други начини за достъп до паметта за съхранение на системата, които не засягат повечето потребители. Ако вашите нужди за съхранение надхвърлят един терабайт, моля, опишете ги напълно във формуляра за достъп. Въз основа на оценката на тези нужди може да се осигури достъп до различни интерфейси.

Копроцесорите Xeon Phi се виждат като sl001-mic0 до sl150-mic0 и sl001-mic1 до sl150-mic1. От потребителите се очаква да ги използват само след получаване на достъп до съответния възел за изпълнение.

Например: sl040-mic0 е достъпен за потребител използващ sl040.

4. Потребителска среда и програмиране

4.1. Система за изпълнение на задачите

Входния възел **gw.avitohol.acad.bg** на Avitohol е предназначен основно за редактиране и компилиране на паралелни програми.. Интерактивното използване на **mpirun/mpiexec** е разрешено на възела за влизане и неговите копроцесори, въпреки че не би трябвало да е необходимо. Препоръчително е да преминете към използване на специален възел, получен чрез система за изпълнение на задачите, ако подобни паралелни изпълнения отнемат повече от няколко минути или използват много памет.

За да стартирате тестови или продукционни задания, изпратете ги на партидната система `Torque`, която ще намери и разпредели ресурсите, необходими за вашата работа (напр. изчислителните възли, върху които да изпълнявате вашата задача(`job`)).

Кратките тестови задачи (по-кратки от 15 минути) с 2, 4 или 8 ядра ще се изпълняват с кратки времена за изпълнение. Обаче използването на частични възли (т.е. по-малко от 16 ядра от един възел) не се препоръчва. Използването на множество частични възли, например чрез заявяване на 10 възела с 4 ядра, не се препоръчва.

По подразбиране ограничението за изпълнение на задачи е зададено на 24 часа на Avitohol. Ако вашите задачи не могат да се изпълняват независимо една от друга, моля, използвайте стъпките за работа или се свържете с системните администратори, като използвате имейл адрес:

avitohol-support@parallel.bas.bg.

Процесорите на Intel поддържат "Hyperthreading / Simultaneous Multithreading (SMT)" режим, който може да увеличи производителността на вашето приложение с до 20%. Въпреки това, оставяме на потребителите да решат дали да използват Hyperthreading или не. В момента възлите за изпълнение са декларирани, че имат 16 процесорни ядра в torque, въпреки че Linux операционната система вижда 32 (логични) ядра.

Препоръчва се на потребителите да изискват кратни на пълни възли, например чрез добавяне:

```
#PBS -l nodes=100:ppn=16
```

в PBS скрипта можете да поискате 100 изчислителни възела.

С Hyperthreading е възможно да увеличите броя на действителните MPI процеси или OpenMP нишки до 32 вместо 16 за всеки възел.

Въпреки това, най-добре е първо да измерите производителността на реалистичен входни данни, за да видите дали Hyperthreading действително е полезна. И в двата случая, със и без Hyperthreading , заявката за възли към PBS ще бъде същата, но командният ред с mpirun ще се промени.

Например, може да се сравни производителността на една и съща изпълнима тестова програма със и без хипернишка на 100 възела, измервайки времето за изпълнение:

```
mpirun –f hostfile -np 1600 -ppn 16 ./testprogram
```

Спрямо

```
mpirun –f hostfile -np 3200 -ppn 32 ./testprogram
```

Файлът с име hostfile може да съдържа списък на всички възли в задачата. Това може да се постигне чрез:

```
cat $PBS_NODEFILE| sort |uniq > hostfile
```

Моля, имайте предвид, че с 32 MPI процеса на възел всеки процес получава само половината от паметта по подразбиране. В клъстера Авитохол има 150 изчислителни възли с 64 GB памет (някои са заети от операционната система). И така, на тези 150 възела можете да очаквате да имате 2 GB на ядро за MPI задания с Hyperthreading или 4 GB на ядро за MPI процеса без Hyperthreading.

Паралелната среда по подразбиране е Intel MPI. Можете да използвате изпълними файлове, които са създадени с други MPI библиотеки. Например, openMPI също е достъпен за цялата система.

За всеки изпълнителен възел има два асоциирани копроцесора Xeon Phi. Например на възел sl039 копроцесорите са достъпни чрез ssh или за mpiexec с имена sl039-mic0 и sl039-mic1.

Всеки копроцесор работи със собствена операционна система, което означава, че изглежда като отделна машина.

За всеки изпълнителен възел има алтернативни имена, които съответстват на алтернативни мрежови интерфейси, свързани с InfiniBand картите. Например ibsl039 съответства на мрежовия интерфейс IP-over-InfiniBand ib0 на сървър sl039.

Същият тип интерфейси IP-over-InfiniBand са налични на копроцесорите, например ibsl039-mic0 и ibsl039-mic1.

Въпреки че тези интерфейси предлагат по-висока честотна лента от стандартните Ethernet интерфейси, те не са необходими за нормална употреба, тъй като MPI задачи се очаква по подразбиране да използват директно InfiniBand, а не IP-over-InfiniBand.

Файловата система Lustre вече използва InfiniBand свързване и тъй като файловете се споделят между възлите и дори е наличен на копроцесорите, необходимостта от движение на данни между възлите чрез интерфейса IP-over-InfiniBand може да е индикация за неоптимално използване на системата .

InfiniBand свързването се използва също по подразбиране за MPI задачи, когато използвате копроцесорите.

След като даден потребител е получил изключителен достъп до възел за изпълнение, той или тя може също да използва съответните копроцесори Xeon Phi. Вътре в скрипта за задачите може да се прочете съдържанието на файла, посочен от променливата на средата PBS_NODEFILE, и да се получат имената на съответните възли на Xeon Phi. Достъпът до копроцесорите Xeon Phi не се управлява отделно.

На потребителите не е разрешено да влизат с ssh на възли, където нямат изпълнявани задачи. За да определи кои възли са разпределени за дадени задачи, потребителят може да изпълни

```
qstat –na <jobnumber>
```

Не се очаква потребителите да използват копроцесори Xeon Phi, без първо да получат достъп до съответния възел.

Допълнителна информация

- За повече информация относно използването на система torque за управление на задачи, вижте документацията от уебсайта на доставчика на софтуер: **www.adaptivecomputing.com**.

4.2. Компилатори

Компилаторите на Intel за Fortan (ifort) и C/C++ (icc, icpc) са по подразбиране и се предоставят автоматично при влизане (вижте изхода на списъка с модули за подробности относно версията).

За да компилирате и свържете MPI програми с помощта на компилатори на Intel, използвайте съответно командите mpiifort, mpiicc или mpiicpc, съответно.

Колекцията компилатори на GNU (gcc, g++, gfortran) също е налична. Версията по подразбиране идва с операционната система. Версия 4.4.7 е инсталирана на системата Авитохол. По-новите версии могат да бъдат достъпни чрез модули за среда. За да компилирате и свържете MPI кодове с

помощта на GNU компилатори, използвайте съответно `командите mpicc, mpic++/mpicxx, mpif77` или `mpif90.`

Компилацията на програми за Xeon Phi обикновено се извършва на основния възел, т.е. чрез кръст-компилация.

За да заредите средата за разработка за Xeon Phi, можете да изпълните:

```
source /opt/mpss/3.6/environment-setup-k1om-mpss-linux
```

Тогава версията на компилатора и други инструменти за разработка могат да бъдат достъпни като изпълните:

```
k1om-mpss-linux-gcc


k1om-mpss-linux-g++


k1om-mpss-linux-nm
```

Изпълнете командата /`bf`, за да получите общ преглед на всички компилатори и версии, налични на системата Авитохол.

Среда на модулите

- За повече информация относно общото използване на *модули* , моля, вижте Ръководството за най-добри практики на PRACE Generic x86 [http://www.prace-ri.eu/IMG/pdf/Best-Practice-Guide-Generic-x86.pdf].
- За повече информация относно използването на MPSS вижте документацията на Intel [https://software.intel.com/en-us/articles/intel-xeon-phi-coprocessor-developers-quick-start-guide]

4.3. Паралелно програмиране

В клъстера Avitohol HPC са налични два основни метода за паралелизиране

- MPI - Интерфейсът за предаване на съобщения осигурява максимална преносимост за паралелизиране наразпределена памет върху голям брой възли.
- OpenMP - OpenMP е стандартизиран набор от директиви на компилатор за паралелизъм на споделена памет. На Avitohol OpenMP код е ограничен да се изпълнява на един възел (независимо дали възел или копроцесор).

Възможно е да се смесват MPI и OpenMP паралелизъм в рамките на един и същи код, за да се постигне широкомащабен паралелизъм. Поради наличието на ускорители Xeon Phi, те са налични

и на копроцесорите и са налични всички режими на използване на копроцесори - native, symmetric и offload.

Всеки копроцесор Xeon Phi може да изпълнява OpenMP програми в естествен режим. Броят на физическите ядра на един копроцесор е 61. Препоръчително е да запазите едно ядро за използване от операционната система и да използвате най-много 60 ядра. Броят на логическите ядра на копроцесора обаче е 4 пъти по-голям от броя на физическите ядра, следователно 244. Опитът показва, че използването на брой нишки, по-голям от броя на физическите ядра, може да бъде от полза. Потребителите могат да опитат да намерят коя настройка на броя нишки за OpenMP чрез променливата OMP_NUM_THREADS (в естествен режим) или MIC_OMP_NUM_THREADS (в режим на разтоварване) е приемлива. В много случаи 120 изглежда добър компромис.

За програмите, които се изпълняват на основния възел, броят на нишките може да бъде зададен равен на броя на процесорните ядра (16) или броя на логическите ядра (32), ако се използва само OpenMP. Ако се изпълнява хибридна MPI/OpenMP програма, трябва да се вземе предвид колко MPI процеси се изпълняват на един възел. Например, ако на всеки възел се изпълняват два MPI процеса, тогава OMP_NUM_THREADS трябва да бъде зададено на 8, за да се използват всички физически ядра, или на 16, за да се използват всички логически ядра.

4.4. Паралелни MPI приложения

По подразбиране компилаторите на Intel за Fortran/C/C++ са налични на Авитохол.

Изпълнимите файлове за MPI програми са mpiifort, mpiicc и mpiicpc. Тези команди предават включването и връзката на информация за MPI заедно със специфични за компилатора флагове на командния ред до компилатора на Intel.

Паралелната програма може да бъде стартирана с mpiifort. Напомняме, че променливата в среда може да бъде предадена чрез опции.

Например, за да зададете променливата MKL_MIC_ENABLE на 1, като по този начин активирате автоматично разтоварване за всички MPI процеси, можете да добавите

```
-genv MKL_MIC_ENABLE 1
```

към командния ред на mpiexec.

Системата за управление на задачи (Batch jobs)

В производствена среда е необходимо програмата MPI с изчислителни задачи, да се пуска със системата за управление на задачи. Моля, вижте раздел 4.1, .“ Система за изпълнение на задачите ” за допълнителна информация.

Environment variables

Някои променливи на средата може да дават информация за софтуер, който не идва директно от дистрибуциите на операционната система. Примери са някои инструменти за изграждане като ant/maven, които са предоставени.

4.5. Multithreaded Многонишкови (OpenMP или хибридни MPI/OpenMP) приложения

За да компилирате и свържете OpenMP приложения, използвайте опция `-qopenmp` (който замества предишната опция `-openmp`, която вече е остаряла) към компилатора на Intel или `-fopenmp` към компилатора `gcc`.

В някои случаи е необходимо да се увеличи размерът на стека на протекторите по време на изпълнение, например когато нишковидните приложения излизат с грешки в сегментирането. На Avitohol размерът на частния стек на нишката се задава чрез променливата на средата KMP_STACKSIZE. Стойността по подразбиране е 4 мегабайта (4MB). Например, за да поискате размер на стека от 128 мегабайта на Авитохол, задайте KMP_STACKSIZE=128m.

Имайте предвид, че на копроцесора практическото ограничение за брой нишки е 244 (61 хардуерни нишки, умножени по 4 поради Hyperthreading). По този начин използването на големи числа за KMP_STACKSIZE може да получи срив на приложението.

> **Забележка: Понякога приложенията могат да използват памет за размяна. По принцип подобно поведение на приложението е нежелателно и силно не се препоръчва. Предполага се, че е резултат от неправилна конфигурация на приложението или грешка във входните данни. От потребителите се очаква да спират такива задачи, които забелязват и да се опитат да коригират грешките. Ако смятат, че това е нормално поведение, те трябва да обсъдят това със системните администратори.**
>
> За информация относно компилиране на приложения, които използват pthreads, моля, консултирайте се с документацията на компилатора на Intel [http://software.intel.com/en-us/articles/intel-c-composer-xe-documentation/#lin].

4.6. Използване на MKL математически библиотеки на Авитохол

Общ преглед на Intel Math Kernel Library

Математическата библиотека на Intel (MKL) се предоставя на Авитохол. MKL предоставя високо оптимизирани реализации.

- LAPACK/BLAS
- Директни и итеративни решаващи солвъри
- FFT процедури
- ScaLAPACK.

Части от библиотеката поддържат многонишково изпълнение или паралелизъм на разпределена памет.

Чрез задаване на променливата на средата MKL_MIC_ENABLE на 1, напр.

```
MKL_MIC_ENABLE=1
```

в bash shell, може да се активира автоматичното разтоварване на някои рутинни библиотеки към копроцесорите Xeon Phi, които са налични. За използването на тази опция с MPI вижте по-горе.

Забележка

Обширна информация за функциите и използването на MKL се предоставя от официалната документация на Intel MKL [http://software.intel.com/en-us/articles/intel-math-kernel-library-documentation/].

За автоматичното offload можете да намерите информация:

https://software.intel.com/sites/default/files/11MIC42_How_to_Use_MKL_Automatic_Offload_0.pdf

Свързване на програми с MKL

По подразбиране модулът на MKL е зареден. Модулът задава променливите на средата MKL_HOME и MKLROOT в инсталационната директория на MKL. След това тези променливи могат да се използват в makefile и скриптове.

Intel MKL Link Line Advisor често е полезен за получаване на информация как да свържете програми с MKL. Например, за да се свържете статично с многонишковата версия на MKL на Авитохол (Linux, Intel64), използвайки стандартни 32-битови цели числа, нещо подобно трябва да се добави към Makefile при извикване на компилатора на Intel:

```
-Wl,--start-group

$(MKLROOT)/lib/intel64/libmkl_intel_lp64.a
$(MKLROOT)/lib/intel64/libmkl_intel_thread.a
$(MKLROOT)/lib/intel64/libmkl_core.a

-Wl,--end-group -lpthread -lm –qopenmp
```

(на един ред) в makefile. Ако компилирате директно в bash, трябва да използвате ${MKLROOT} вместо $(MKLROOT).

4.7. Числови библиотеки

FFTW

Библиотеките FFTW са инсталирани. Понастоящем версията на FFTW 3, която идва с операционната система, е 3.2.3 и е достъпна на стандартните места за софтуер за разработка.

PETSc

Пакет от структури от данни и процедури за мащабируемо (MPI паралелно) решение на научни приложения, моделирани чрез частични диференциални уравнения. Предлага се като модул.

SLEPc

Библиотека за решаване на широкомащабни задачи с редки собствени стойности на паралелни компютри. Предлага се като модул.

GSL

GNU Scientific Library (GSL) е цифрова библиотека за C и C++ програмисти (инсталиран е интерфейсът FGSL FORTRAN addon). GSL предоставя широк набор от математически процедури като генератори на произволни числа, специални функции и метод на най-малките квадрати. Предлага се като част от дистрибуцията на операционата система.

Python

Версията на Python от дистрибуцията на ОС е от серията 2.6. Последната текуща версия от серия 2.7 се предлага като модул. Модулът tensorflow е наличен като част от инсталацията 2.7.

4.8. Входно/Изходни Библиотеки

NetCDF

NetCDF е набор от софтуерни библиотеки и самоописващи се, машинно-независими формати на данни, които поддържат създаването, достъпа и споделянето на научни данни, ориентирани към масиви. Предлага се като модул (версия 4).

4.9. Разнообразни библиотеки

1. Boost: Библиотеките Boost C++ предоставят много класове и процедури за различни приложения. Версията на boost предоставен от операционата система е доста стар. По-нова версия е достъпна в `/opt/soft/` и е достъпна чрез система за модули.
2. TBB: Intel Threading Building Blocks. TBB позволява на C++ програмиста да интегрира (споделена памет) паралелно възможност в кода.. Предлага се като част от Intel Compiler Suite.
3. IPP: Intel Integrated Performance Primitives (IPP). В IPP съдържа силно оптимизирани примитивни операции, използва се за цифрово филтриране, аудио и обработка на изображения.
4. PAPI: The Performance Application Programming Interface. Интерфейсът за програмиране на приложения за производителност е библиотека за четене на броячи на събития за производителност по преносим начин.

4.10. Друг софтуер

Други популярни пакети са налични за цялата система. Например – R, cmake, gnuplot.

Модули на среда

- Моля, използвайте bf на Авитохол, за да получите изчерпателен списък с модули за среда.

- В повечето случаи само най-новата/текущата производствена версия (към март 2016 г.) на софтуерните пакети се предлагат чрез средата на модула.

Default Fortran data type size Размер на типа данни на Fortran по подразбиране

Някои програмисти на Fortran използват например 32-битови плаващи числа в изходния код (REAL) и ги преобразуват в 64-битови плаващи числа (REAL*8) по време на компилиране. Съответният флаг на компилатора за компилатора на Intel Fortran е `-r8`.

5. Инструменти за програмиране

За достъп до описания по-долу софтуер, моля, използвайте командата (**module avail, module load**).

5.1. Отстраняване на грешки

- Опции на компилатора: Компилаторите обикновено имат някои функции за отстраняване на грешки, които позволяват например да проверяват нарушения на граници на масива. Моля, направете справка с ръчните страници и документацията на компилатора за подробности.

- gdb, GNU отстраняване на грешки.

- Intel Inspector позволява отстраняване на грешки на многонишкови приложения.

- Ако няма да се използват функции за отстраняване на грешки, изпълнимият файл може да бъде лишен от информация за отстраняване на грешки чрез **strip.**

5.2. Профилиране и анализ на ефективността

- Intel VTune/Amplifier е мощен инструмент за анализиране на едноядрената производителност на код.

- gprof, the GNU profiler.

- Intel Trace Analyzer and Collector е инструмент за профилиране на MPI комуникации.
- Scalasca позволява анализ на MPI/OpenMP/hybrid кодове.
- Оптимизирани изпълними файлове могат да бъдат получени като първо създаде профил (опция -prof-gen за Intel, -- fprofile- generate за gcc) и след това стартиране на изпълнимия файл и използване на генерирания профил с опции като.
    - -prof_use -prof_dir ./profdir за Intel
    - '-fprofile-use - за gcc, понякога е необходимо да добавите -fprofile-correction или -flto-partition=none

Допълнителна информация

За повече информация, свързана с раздел 5, „Инструменти за програмиране“, моля, вижте уебсайта на Avitohol:

• http://www.hpc.acad.bg/bg/system-1/

6. Заключение

Системата Авитохол е сравнително нова, но е оборудвана с богат набор от инструменти за разработка, библиотеки и софтуер. Потребителите, които изискват инсталиране на допълнителни инструменти или софтуер или обновяване на съществуващ софтуер, трябва да се свържат с екипа за поддръжка на **avitohol-support@parallel.bas.bg.**

7. Още информация

7.1 TORQUE Resource Manager

Осигурява контрол върху batch задачи и разпределените компютърни ресурси. Това е продукт с отворен код, базиран на оригиналния проект PBS и се разработва от Adaptive Computing, както и общността зад проекта. Постига значителен напредък в областта на мащабируемостта, надеждността и функционалността и в момента се използва в десетки хиляди водещи правителствени, академични и търговски обекти по целия свят.

Полезни команди за подаване на задачи

За изпращане на задачата към опашката на PBS и за поискване на допълнителни ресурси, се използва командата qsub. Състоянието на заданието, което вече е в опашката на PBS, може да се провери с командата qstat. PBS скрипт, съдържащ броя на възлите и процесорите, необходими за задачата, колко време за часовник се изисква, програма, която да се изпълни, може да се използва за опростяване на този процес. Примерни настройки, които могат да се поставят в скрипта на PBS:

За задаване на име на задачата:

```
#PBS -N job_name
```

стандартед изход:

```
#PBS -o stdout_file
```

грешки:

```
#PBS -e stderr_file
```

максимално процесорно време и системно време. Задачата се прекратява ако не завърши в определеното време:

```
#PBS -l cput=hhhh:mm:ss
#PBS -l walltime=hhhh:mm:ss
```

брой на възлите и процеси на всеки от тях:

```
#PBS -l nodes=2:ppn=16
```

максимален размер на необходима физическа памет. Ако се зададе прекалено голям, може да се забави изпълнението прекалено много време докато се освободи достатъчно:

```
#PBS -l mem=4096mb
```

мястото на задачата на опашка и/или сървър:

```
#PBS -q queue@server
```

приоритет на задачата(цяло число от -1024 to 1023). Стандартно е 0:

```
#PBS -p 0
```

Изпращане на задачата:

```
qsub myjob.pbs
```

qsub приема допълнителни параметри за промени на настройките. Ако са необходими повече процесорни ядра например:

```
qsub -l nodes=4:ppn=2 myjob.pbs
```

За проверка на информация:

qstat без аргументи дата информация за всички задачи в системата в момента

За определена задача:

```
qstat jobID
```

За всички задачи на определен потребител:

qstat -u username

За по-пълна информация:

```
qstat -l jobID
```

За проверка на кои възли е била изпратена задачата:

```
qstat –na
```

За да се видят всички задачи на определена опашка:

qstat queue_name

Обощение на статуса на всички опашки:

```
qstat –q
```