Дополнительное распараллеливание MPI программ с помощью системы SAPFOR

Н.А. Катаев, А.С. Колганов

Институт прикладной математики им. М.В. Келдыша РАН

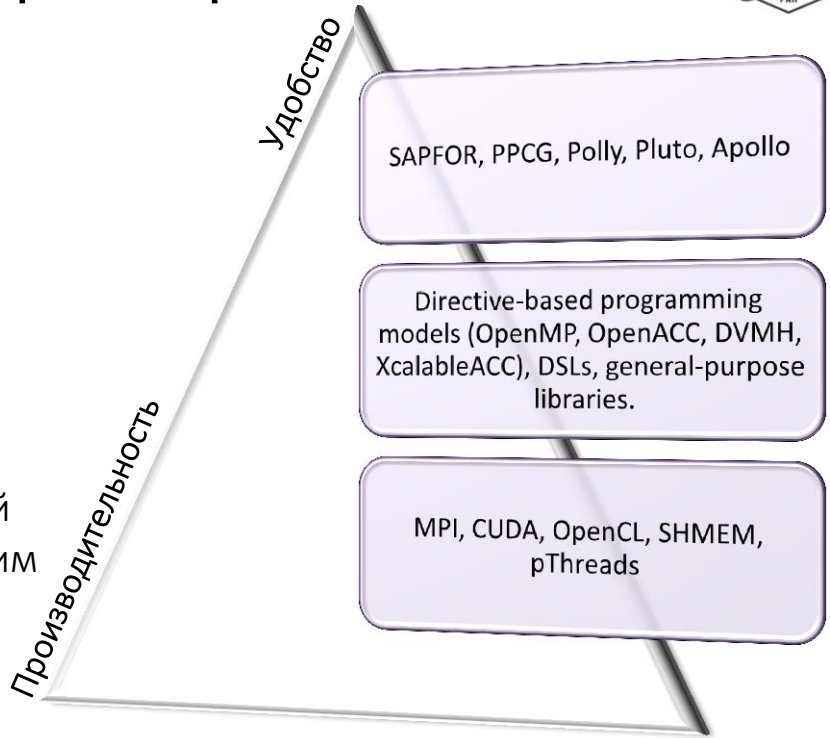
28 сентября, 2021 | Москва

Инструменты параллельного программирования

Автоматически распараллеливающие компиляторы создают параллельный код для входной программы (но не всегда оптимальный).

Директивные языки упрощают программирование и повышают удобство сопровождения ПО, обеспечивая при этом высокую производительность.

Низкоуровневые модели дают программистам точный контроль над выполнением программы и позволяют им добиться наилучшей производительности.



Наш подход к использованию параллелизма в существующих MPI-программах (C и Fortran):

- использование директивных моделей Fortran-DVMH и CDVMH;
- использование системы автоматизации SAPFOR;
- участие пользователя в процессе распараллеливания.

DVMH-модель

Язык параллельного программирования высокого уровня.
Модель программирования на основе директив.

DVMH

CUDA

OpenMP

MPI

Модель программирования на основе директив, которая направлена на создание параллельных программ для гетерогенных вычислительных кластеров (GPU NVidia, Intel Xeon Phi, многоядерные процессоры).

Модель включает в себя два языка программирования, которые являются расширениями стандартных языков C и Fortran спецификациями параллелизма: CDVMH и Fortran-DVMH

Параллельная программа разрабатывается в терминах последовательной.

Fortran-DVMH Program with Data Distribution

```
program jacoby_dvmh
  parameter (l=4096, itmax=100)
  real a(l,l), b(l,l), eps
!DVM$ DISTRIBUTE(BLOCK, BLOCK) :: A
!DVM$ ALIGN B(I,J) WITH A(I,J)
  ...
  do it = 1, itmax
    eps = 0.
!DVM$ REGION
!DVM$ PARALLEL (J,I) ON  A(I,  J), REDUCTION(MAX(EPS))
    do  j = 2, l-1
      do  i = 2, l-1
        eps = max(eps, abs(B(i, j) - A(i, j)))
        a(i, j) = b(i, j)
      enddo
    enddo
!DVM$ PARALLEL (J,I) ON B(I,  J), SHADOW_RENEW(A)
    do  j = 2, l-1
      do  i = 2, l-1
        b(i, j) = (a(i-1, j) + a(i, j-1) + a(i+1, j) + a(i, j+1)) / 4
      enddo
    enddo
!DVM$ END REGION
  enddo
!DVM$ GET_ACTUAL(B)
  print *, b
end
```

- распределение данных,
- вычислительные области и спецификации перемещения данных между CPU и GPU,
- распределение вычислений,
- свойства переменных и удаленных данных.

Fortran-DVMH Program without Data Distribution

```
program jacoby_dvmh
  parameter (l=4096, itmax=100)
  real a(l,l), b(l,l), eps


  ...
  do it = 1, itmax
    eps = 0.
!DVM$ REGION
!DVM$ PARALLEL (J,I), REDUCTION(MAX(EPS)), TIE(A(I,J), B(I, J))
    do  j = 2, l-1
      do  i = 2, l-1
        eps = max(eps, abs(B(i, j) - A(i, j)))
        a(i, j) = b(i, j)
      enddo
    enddo
!DVM$ PARALLEL (J,I), TIE(A(I,J), B(I, J))
    do  j = 2, l-1
      do  i = 2, l-1
        b(i, j) = (a(i-1, j) + a(i, j-1) + a(i+1, j) + a(i, j+1)) / 4
      enddo
    enddo
!DVM$ END REGION
  enddo
!DVM$ GET_ACTUAL(B)
  print *, b
end
```

- вычислительные области и спецификации перемещения данных между CPU и GPU,
- распределение вычислений,
- свойства переменных и удаленных данных.

Распараллеливание MPI программ с использованием DVMH

Система поддержки времени выполнения DVMH не участвует в межпроцессорном взаимодействии и работает локально в каждом MPI процессе.

Спецификация tie используется для задания соответствия между циклами в гнезде параллельных циклов и измерениями массивов.

Возможности системы DVM позволяют:

- использовать параллелизм на общей памяти с использованием процессорных ядер (потоков OpenMP) или графических ускорителей;
- выполнять автоматические преобразования данных на графических процессорах и использовать упрощенное управление перемещением данных между памятью ЦП и графических процессоров;
- выбирать оптимизационные параметры системы runtime DVMH;
- использовать инструменты для отладки и анализа производительности полученных параллельных программ.

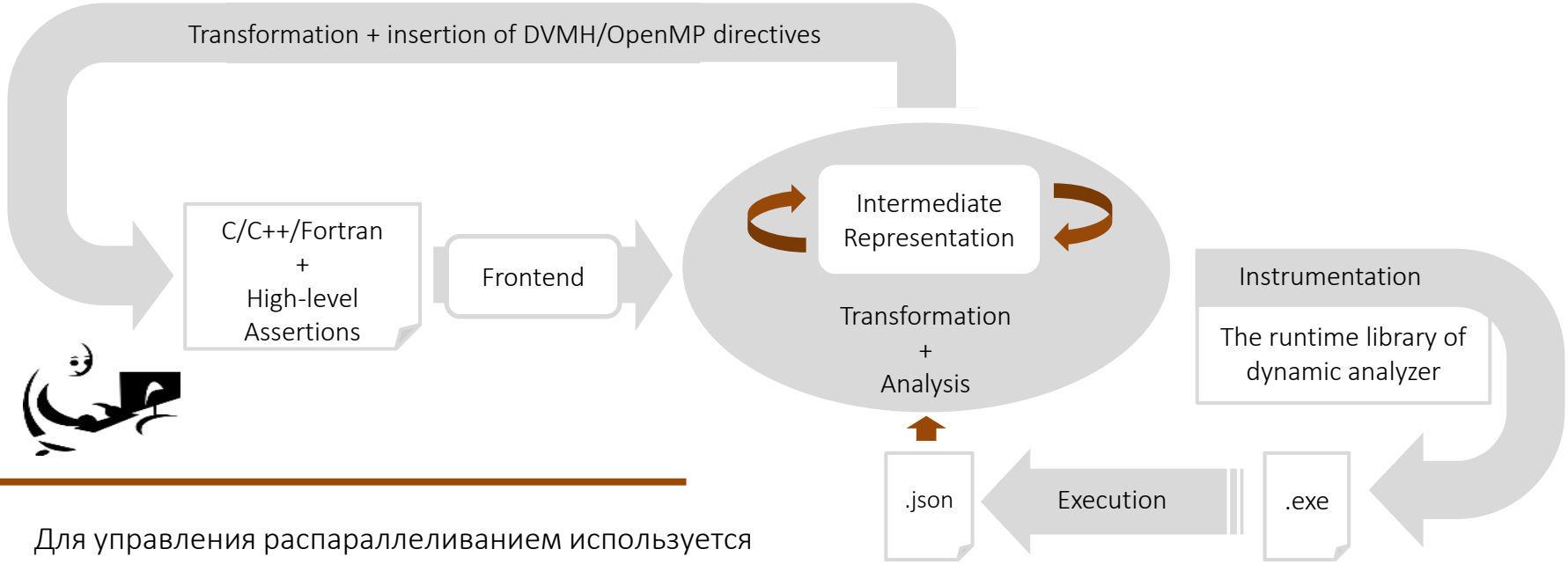
Система SAPFOR

SAPFOR (System For Automate Parallelization) является системой для разработки параллельных программ, которая ориентирована на снижение затрат на ручное распараллеливание программ.

Главные цели разработки системы SAPFOR :

- Исследование последовательных программ (анализ и профилирование программ).
- Автоматическое распараллеливание (в соответствии с моделями DVMH или OpenMP) потенциально параллельной программы, для которой программист максимизирует параллелизм на уровне алгоритма и/или добавляет высокоуровневые спецкомментарии свойств программы.
- Полуавтоматическое преобразование программы для получения потенциальной последовательной версии исходной программы.

Архитектура системы SAPFOR



- Для управления распараллеливанием используется графический пользовательский интерфейс.
- Инструменты автоматизации сборки, такие как Make, также можно использовать для выполнения анализа программ.

Автоматическое распараллеливание с использованием DVMH модели

Для распараллеливание на вычислительные устройства с общей памятью (многоядерный процессор или ускоритель) требуется, чтобы в исходный код были вставлены три вида директив:

- спецификации для циклов, которые могут выполняться параллельно, а также спецификации приватных и редукционных переменных, а также шаблон доступа к массиву,
- спецификация вычислительных областей, которые могут быть выполнены на ускорителях, каждая область может содержать один или несколько параллельных циклов,
- высокоуровневые спецификации передачи данных между памятью центрального процессора и памятью ускорителя (директивы актуализации данных).

Для каждого цикла рассматриваются следующие ограничения:

- безопасность потока управления (отсутствие операций ввода-вывода, побочных эффектов и т.д.),
- безопасность доступа к памяти (отсутствие зависимостей по данным в циклах и «захваченных» указателей),
- направление использования данных (входные, выходные и локальные данные),
- каноническая форма цикла в соответствии со стандартом OpenMP,
- возможность выразить свойство переменной с использованием спецификаций DVMH языков,
- возможность объединения итерационных пространств вложенных циклов в одно большее итерационное пространство.

Анализ приложений NAS Parallel Benchmarks 3.3.1

Автоматическая подстановка на уровне IR:

- определение вызовов функций, которые ухудшают анализ, и выполнение их подстановки,
- не влияет на исходный код программы.

Расширен анализ приватизируемых, редукционных и индуктивных скаляров, участвующих в адресной арифметике:

- локальное изменение IR-кода для разрыва явной связи между переменными и MPI функциями,

```
 double sum = 0.0E0;

 for (int j = 1; j <= lastcol - firstcol + 1; ++j)
   sum = sum + r[j - 1] * r[j - 1];

MPI_Send(&sum, 1, dp_type, reduce_exch_proc[i - 1],
         i, MPI_COMM_WORLD);
```

⇨

```
double sum = 0.0E0;
double sum_promoted = sum;
for (j = 1; j <= lastcol - firstcol + 1; ++j)
  sum_promoted = sum_promoted + r[j - 1] * r[j - 1];
sum = sum_promoted;
MPI_Send(&sum, 1, dp_type, reduce_exch_proc[i - 1],
         i, MPI_COMM_WORLD);
```

- Анализ обращений к памяти на основе LLVM, которые считаются ссылками на регистры.

Предварительное задание in/out для встроенных процедур языка Fortran и Си, а также **MPI функций** для расстановки директив актуализации данных между ЦПУ и ГПУ.

Динамический анализ для выявления приватизируемых массивов, и ручная спецификация параметров анализа, чтобы указать, что выражение индекса не выходит за пределы выделенной памяти.

Ручное преобразование приложений EP и BT

Замена редукционного массива набором редукционных скаляров (EP):

```
q[l] = q[l] + 1.0;
```

⇨

```
switch (l) {
  case 0: q0 = q0 + 1.0; break;
  case 1: q1 = q1 + 1.0; break;
   ...
}
```

Устранение большого приватизируемого массива для уменьшения использования памяти на ГПУ (EP, BT):

```
for (i = 0; i < NK; i++) {
  ...
  x[i] = r46 * (*x4);
}

for (i = 0; i < NK; i++) {
  x1 = 2.0 * x[2 * i] - 1.0;
  x2 = 2.0 * x[2 * i + 1] - 1.0;
  ...
}
```

⇨

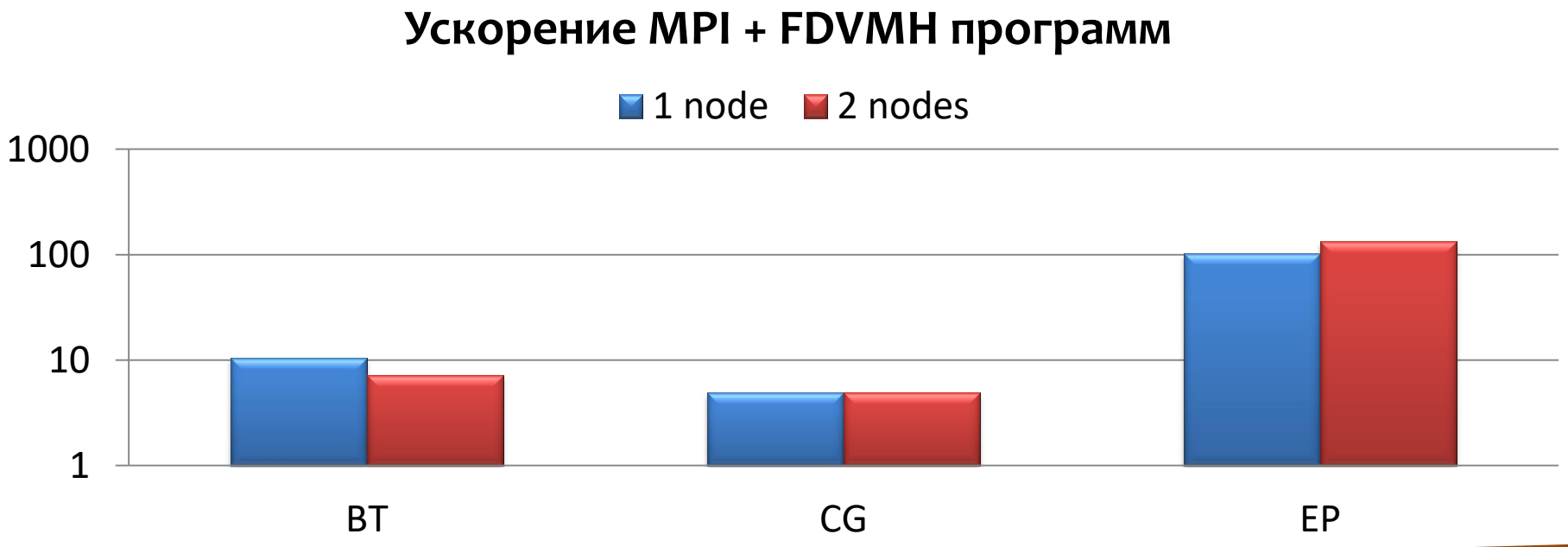
```
for (i = 0; i < NK; i++) {
  double x_2i, x_2i1;
  ...
  x_2i = r46 * (*x4);
  ...
  x_2i1 = r46 * (*x4);

  x1 = 2.0 * x_2i - 1.0;
  x2 = 2.0 * x_2i1 - 1.0;
  ...
}
```

Распараллеливание NAS Parallel Benchmarks 3.3.1 (Fortran)

Time (sec)

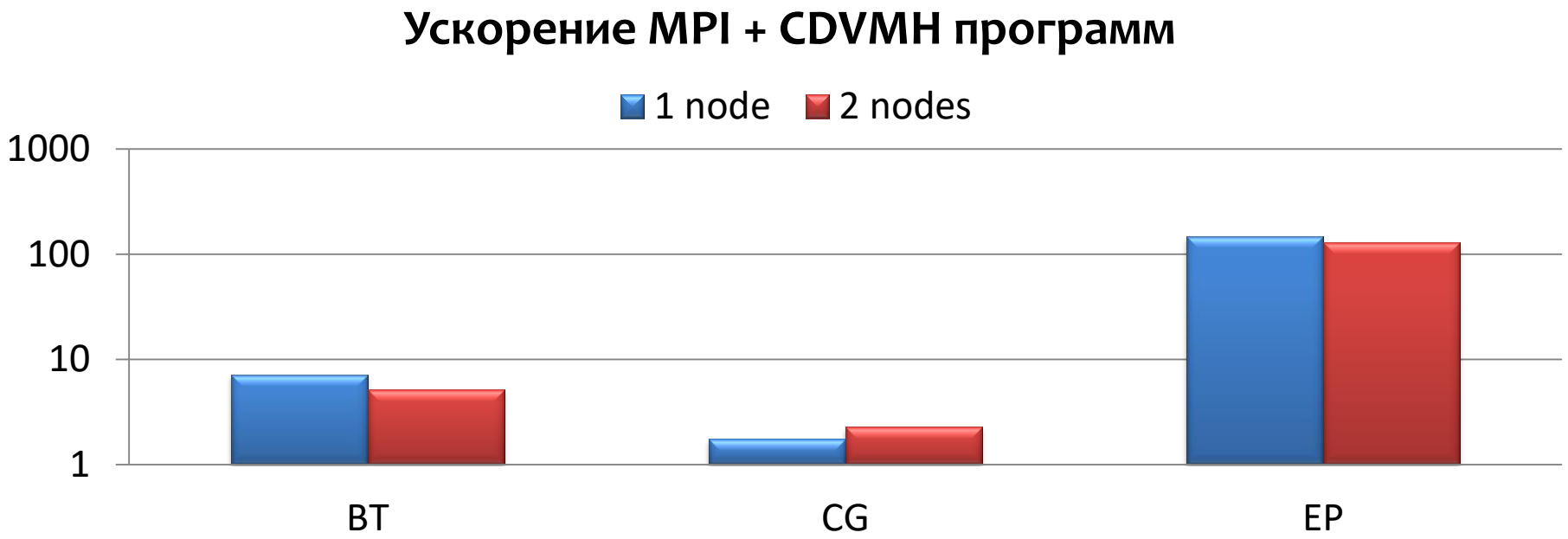
	BT	CG	EP	BT (MPI+DVMH)	CG (MPI+DVMH)	EP (MPI+DVMH)
1 node	665.1	397.5	93.6	63.3	80.99	0.62
2 nodes	361.6	209.6	46.5	50.3	42.6	0.38



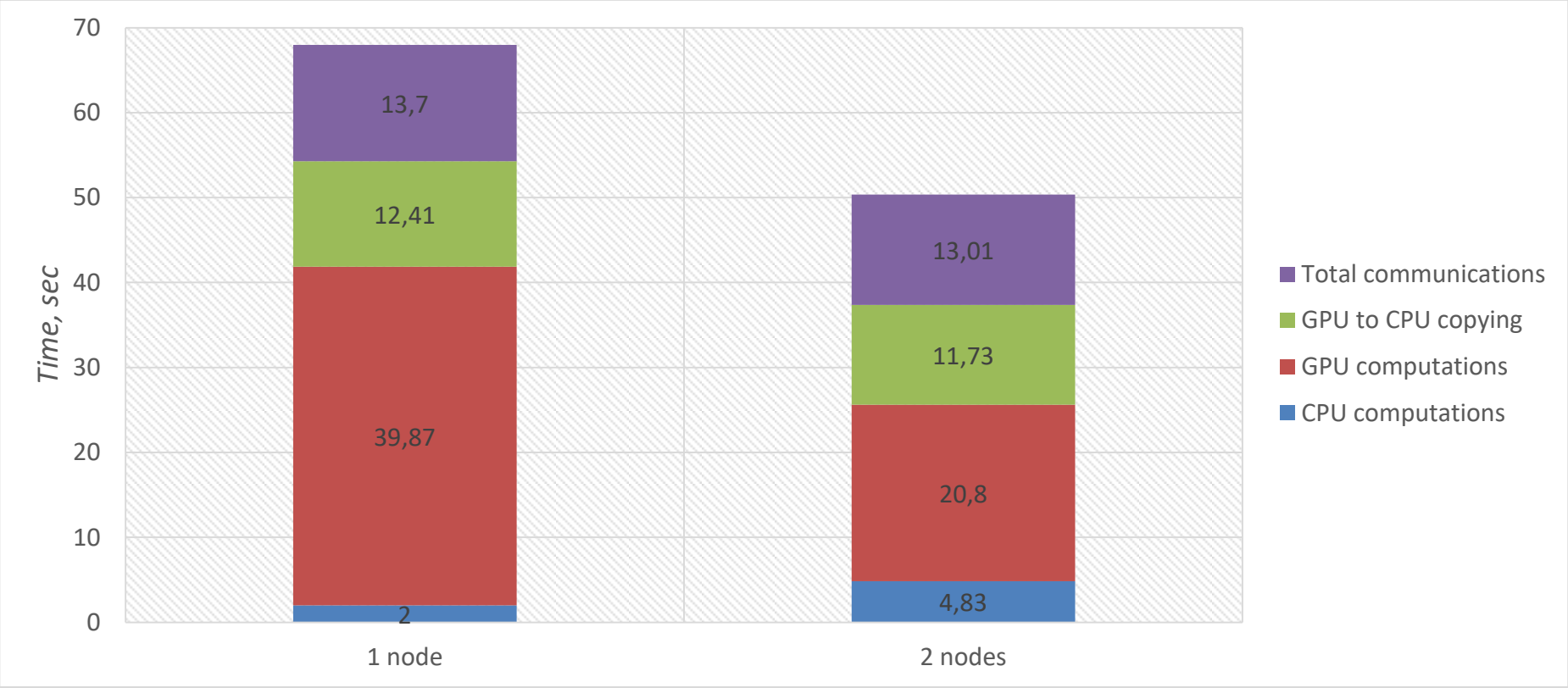
Распараллеливание NAS Parallel Benchmarks 3.3.1 (C)

Time (sec)

	BT	CG	EP	BT (MPI+DVMH)	CG (MPI+DVMH)	EP (MPI+DVMH)
1 node	694.6	326.1	98.4	97.7	186.12	0.67
2 nodes	386	218.9	49.2	75.7	96.75	0.38



Соотношение времен вычислений и коммуникаций на примере приложения BT (Fortran)



Заключение

SAPFOR полагается на новые функции системы DVM, которые позволяют отображать параллельные циклы на графические процессоры в MPI-программах.

В системе SAPFOR реализован автоматически распараллеливающий компилятор, который подходит для распараллеливания потенциально параллельных программ без участия пользователя.

Пользователь может добавлять свойства программы или определять последовательность необходимых преобразований.

Для повышения производительности параллельных программ система SAPFOR может полагаться на различные оптимизации, реализованные в компиляторе и в runtime системе DVMH. Система DVM предоставляет инструменты анализа производительности, которые работают в понятных для пользователя терминах.

Системы SAPFOR и DVM могут значительно снизить затраты на использование внутриузлового параллелизма в существующих MPI-программам, а также позволяют использовать все доступные устройства в узле, такие как многоядерные процессоры или графические процессоры.

Спасибо за внимание!

Russian Supercomputing Days

http://dvm-system.org

KIAM RAS DVM SYSTEM

ОРДЕНА ЛЕНИНА ИНСТИТУТ ПРИКЛАДНОЙ МАТЕМАТИКИ им.М.В.КЕЛДЫША РАН