Результаты масштабирования бенчмарка NPB UA на тысячи ядер суперкомпьютера Blue Gene/P с помощью PGAS-расширения OpenMP

Антон Корж

anton@korzh.ru

anton@exascale.ru

План

- Описание бенчмарка NPB UA
- Описание библиотеки SHMEM
- Детали SHMEM-версии бенчмарка
- Результаты

Библиотека SHMEM

SHMEM

- shmem_init, shmem_finalize
- my_pe, num_pes
- shmem_alloc

- shmem_double_put, shmem_double_get, shmem_double_add,shmem_double_p
- *То же для типов long и int*
- shmem_sync_gets
- shmem_barrier_all

SHMEM

- У каждого процесса есть локальная память, и есть удаленная.
- Как указать адрес в памяти другого узла?
- Ответ (идея Cray T3E):

  симметричный указатель

- shmem_alloc() – коллективный вызов

SHMEM vs OMP

- Программист должен явно распределить данные и выделить обращения к удаленным данным

- Самый простой стиль программирования – глобальные фазы вычислений и коммуникаций разделенные барьером

Где есть SHMEM ?

- Все машины Cray
- Машины с сетью Quadrics Elan
- SGI Altix
- Макеты с сетью НИЦЭВТ
- Макет МВС-Экспресс (А.О.Лацис)
- Скиф-Аврора
- **IBM Blue Gene/P + MPI2**

SHMEM версия

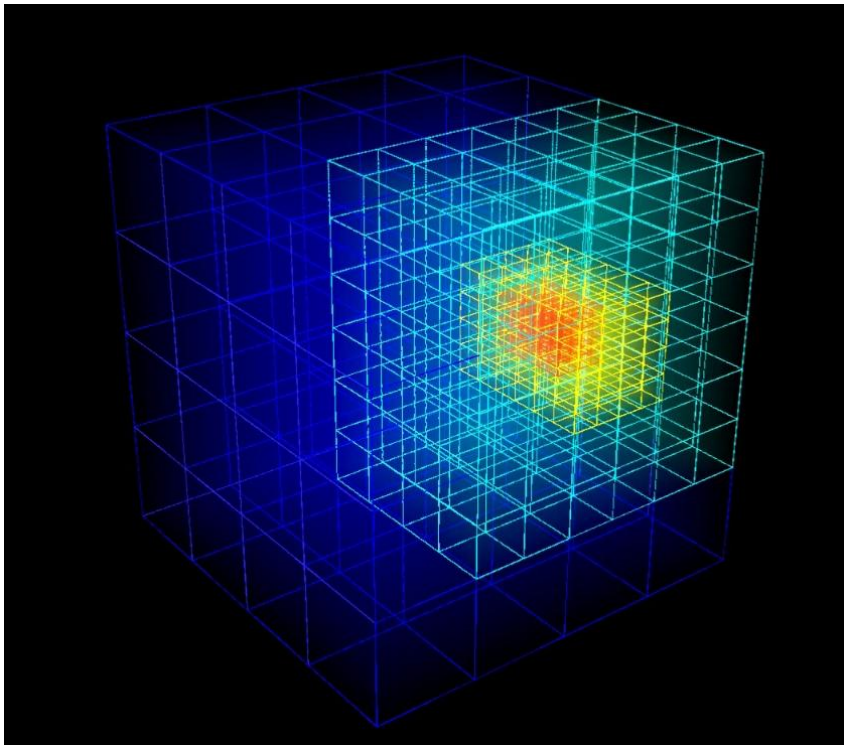
- Поверх MPI-2 one-sided
  - MPI_Win_fence
  - MPI_Put, MPI_Get, MPI_Accumulate
  - MPI_Allreduce, MPI_Allgatherv

- DCMF Deep Computing Messaging Framework
  - DCMF_GlobalBarrier, DCMF_Messager_advance
  - DCMF_Put, DCMF_Get, DCMF_Send
  - DCMF_GlobalBcast, DCMF_GlobalAllreduce

Задача:
Unstructured Adaptive

Описание бенчмарка NPB UA

- Уравнение теплопереноса в кубической области (конвекция + диффузия)
- Источник тепла –
- Движущийся шар
- Нерегулярная сетка
- Каждый 5й шаг – сетка меняется
- Используется параллельный 3D Mortar method



Основной цикл

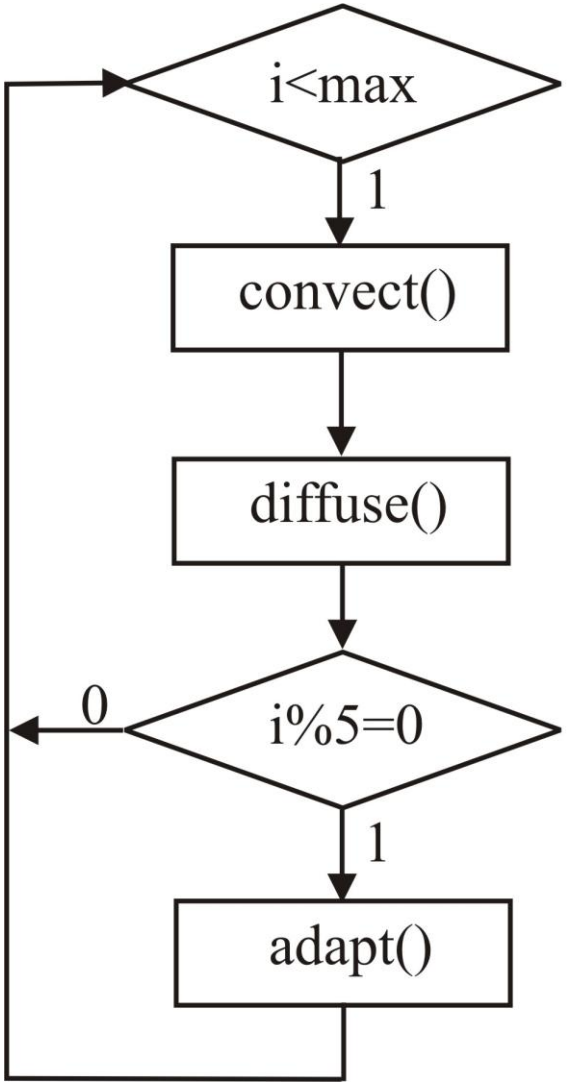
- convect – продвижение во времени конвекционного члена исходного уравнения

$$\hat{T}^{n+1} = \mathrm{RK}_4\left(-\mathbf{v}\nabla T^n + S(\mathbf{x},t)\right)$$

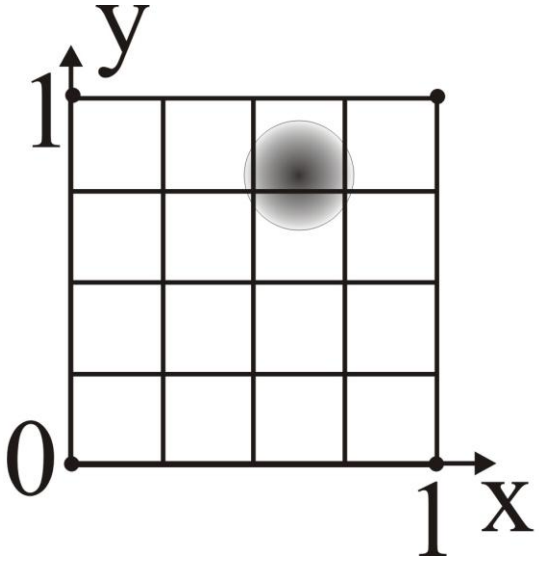
- diffuse – продвижение вперёд диффузионного члена уравнения методом сходящегося градиента

$$\frac{T^{n+1} - \hat{T}^{n+1}}{\Delta t} = \varepsilon \nabla^2 T^{n+1}$$

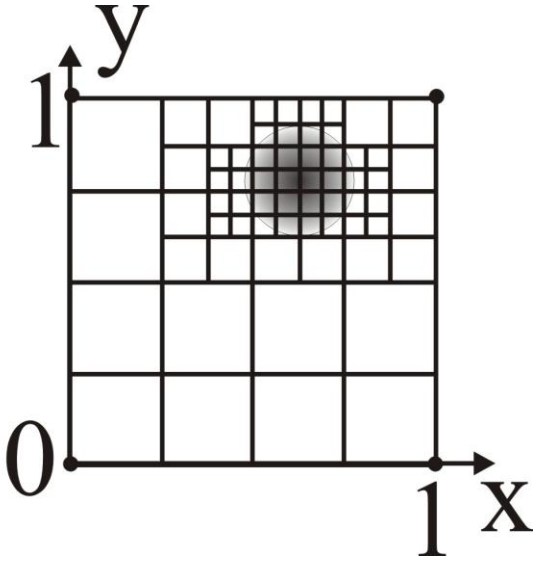
- adapt – изменение сетки



Нерегулярность сетки



равномерная сетка

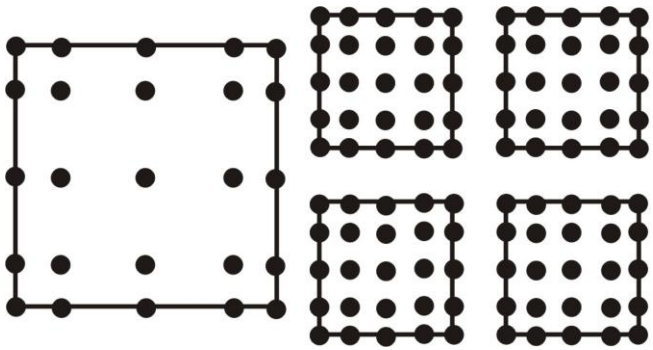


нерегулярная сетка

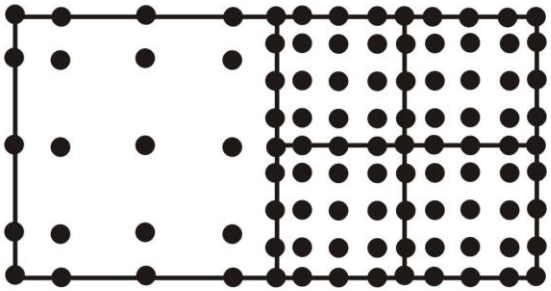
Пространственная дискретизация

Spectral Element Method – SEM:

- Каждый элемент сетки покрывается декартовым произведением $(N+1)^3$ точек коллокации, где N=4



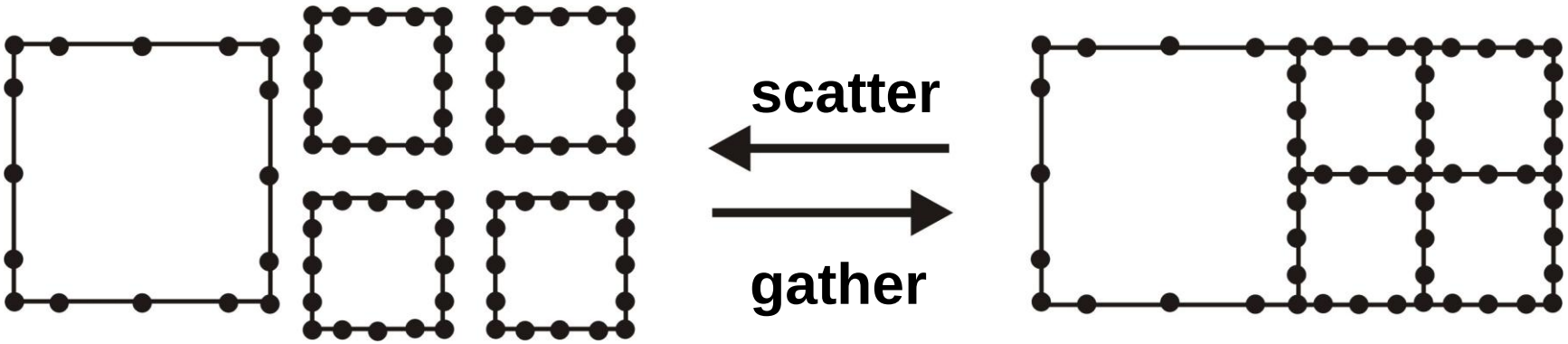
Точки коллокации



Точки сетки

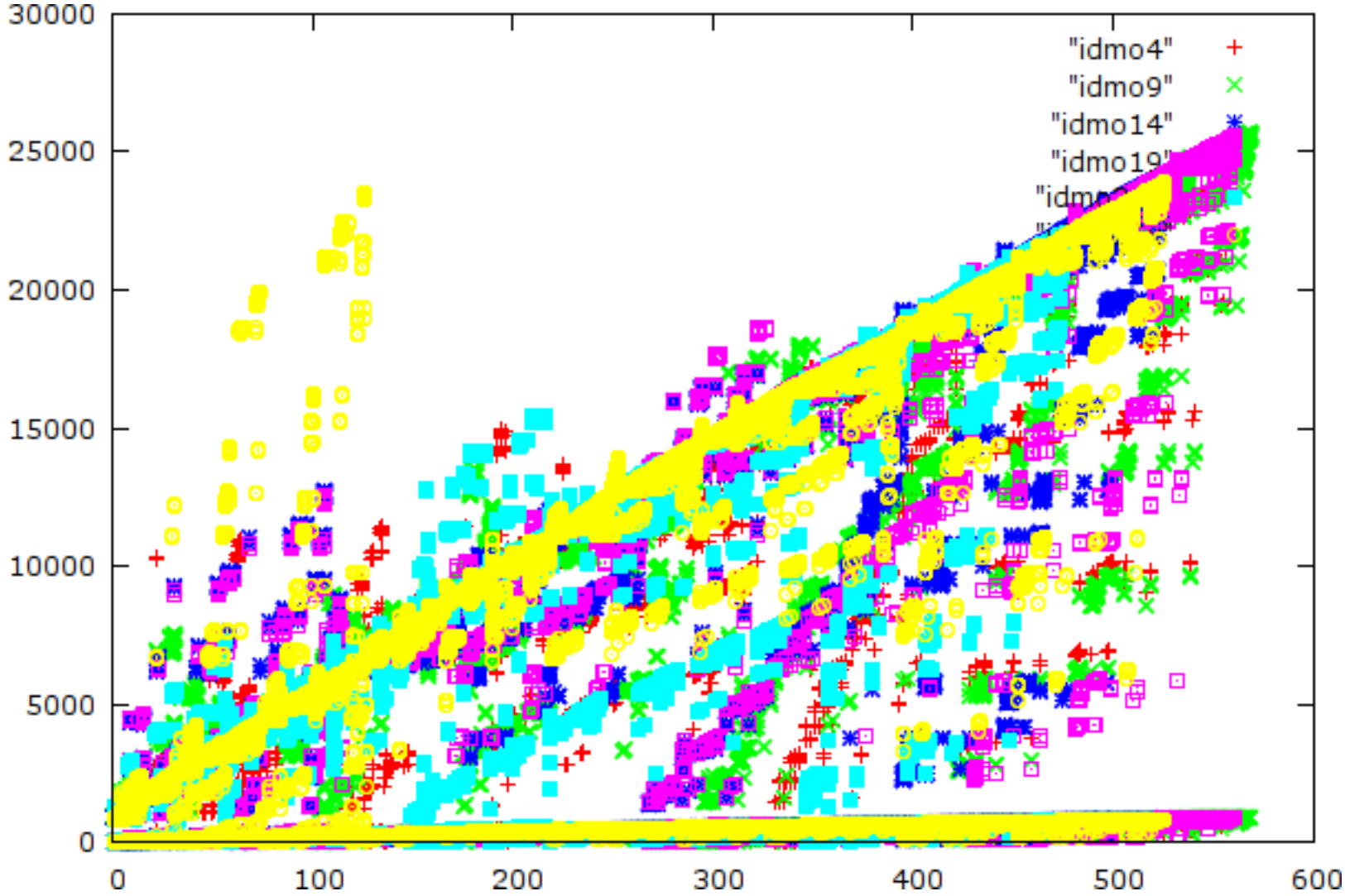
- Положение точек коллокации внутри элемента соответствует квадратурной формуле Гаусса-Лобатто-Лежандра

Согласование решения



- Метод Mortar Element Method (MEM) основывается на введении пространства согласования, состоящее из точек сетки, на границе элементов

Шаблон доступа в операциях Scatter/Gather



Номер элемента

В итоге: факты об NPB UA

- Нерегулярный шаблон доступа к памяти
- Динамически изменяемая сетка

- Нет MPI версии
- OMP версия так себе масштабируется
- На кластере не запустить вообще
- 8000 строк кода на Fortran77
- Класс D не везде запустить (> 8 GB)

Что было сделано?

- OMP версия трансформирована в SHMEM+OMP
  - Данные вручную распределены по узлам
  - Итерации OMP циклов распределены по узлам
  - Зависимости по данным заменены на вызовы SHMEM
  - Инкрементальное распараллеливание (часть кода все еще последовательная)

- SHMEM реализован на Blue Gene/P
  - Через библиотеку MPI-2
  - Через DCMF (низкоуровенная комм. библиотека BG/P)
  - Добавлены active messages в SHMEM (DCMF только)

Распределяем массивы блочно

- Циклы полностью параллельные по данным

```
do iel=1,nelt
 call laplacian(tx(..,..,..,iel)
end do
```

трансформируем в

```
do iel=startelt,endelt
 call laplacian(tx(..,..,..,iel)
end do
```

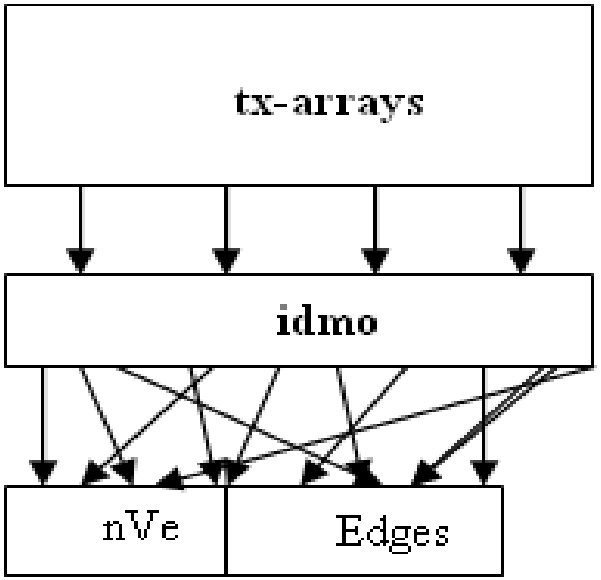
Не все OMP циклы такие…

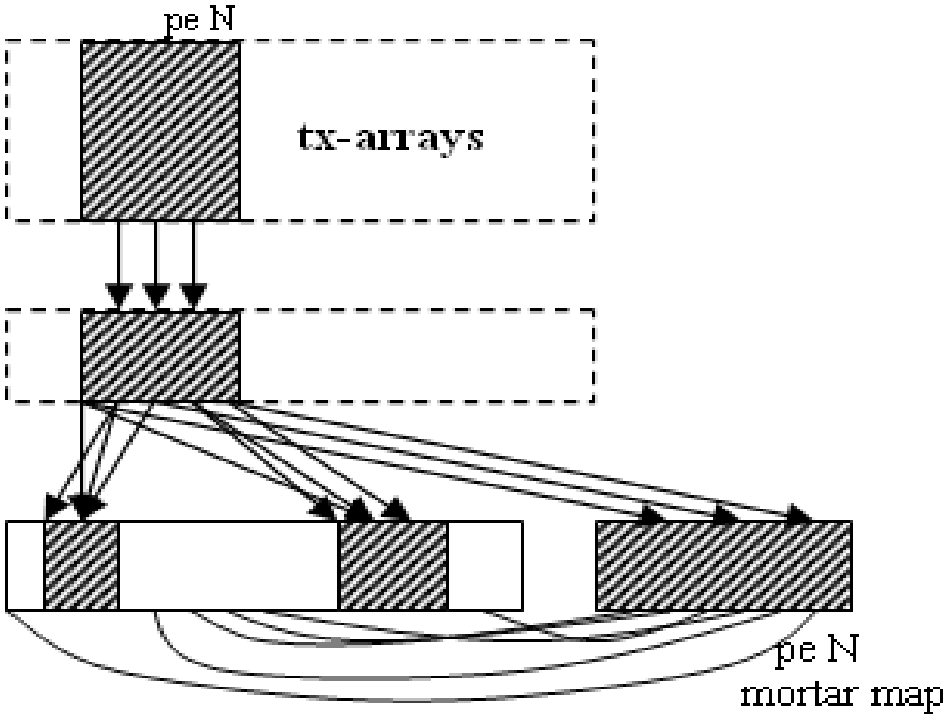
- Чтения только локальных элементов, записи могут быть удаленные, между итерациями нет зависимостей
- Есть небольшое число нелокальных чтений (может быть большой блок – граница)
- Большое число удаленных чтений

Gather

```
c$OMP PARALLEL DO
do iel=1,nelt
    do j=1,25
     tx (j,iel) =a(j,1)*tmor(idmo(f(j,1),iel))
&                +a(j,2)*tmor(idmo(g(j,1),iel))+...
    end do
end do
c$OMP END PARALLEL DO
```

Модификация индекс-вектора





serial

distributed

Распределение не только блочное

- Циклическое
- Блочно-циклическое итп

- Любое заданное пользователем

Задать функции

makedistribution
islocal_elt
pe_elt
local_elt

Gather

```
call shmem_barrier_all
 do i = 1,mormap_size
    pe = pemor(mormap(i))
    call shmem_double_g (morloc(i),
&                                  tmor(localmor(mormap(i))), pe)
 end do
call shmem_sync_gets
c$OMP PARALLEL DO
    do iel=startelt, endelt
      do j=1,25
        tx(j,iel) =  a(1,j) * readmor ( xidmo(f(j,1),iel) ) +
 &                  +b(2,j) * readmor ( xidmo(g(j,1),iel) )+ ….
      end do
    end do
c$OMP END PARALLEL DO
```

PUT вместо GET

```
do i=0,numpes-1
   j = 1 + pe * (deltamor+1)
   do while(mormap2(j).gt.0)
         k = IBITS(mormap2(j),0,32)
         ii = ISHFT(mormap2(j)-k,-32)
         call shmem_double_p(morloc(ii),tmor(k),pe)
         j = j + 1
   end do
end do
call shmem_barrier_all
```

mormap2 – обратная карта доступа

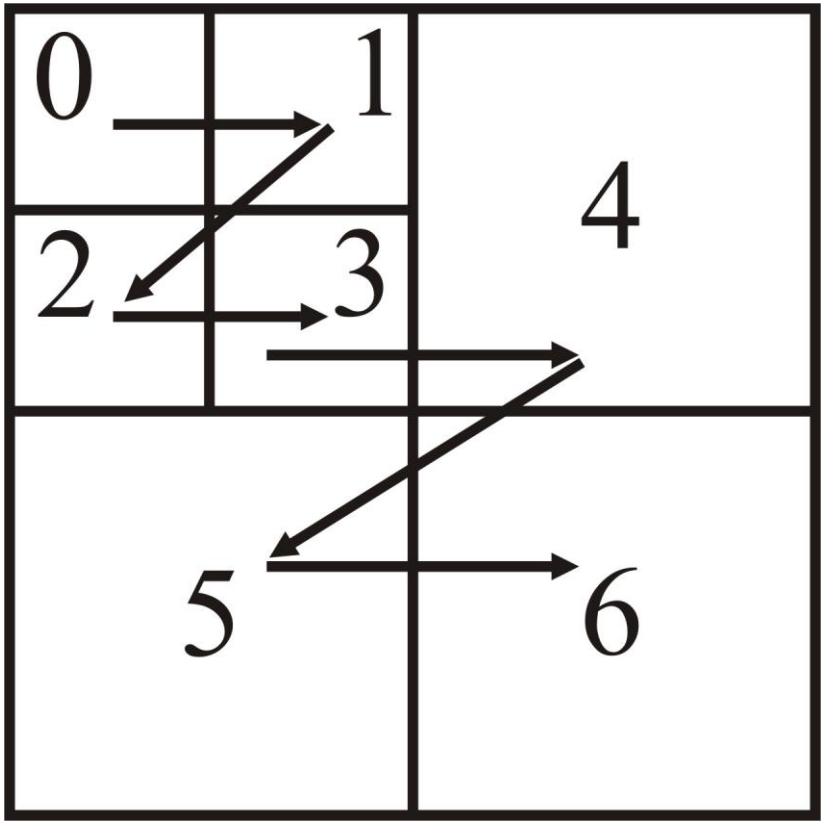
Трюки в стиле PGAS

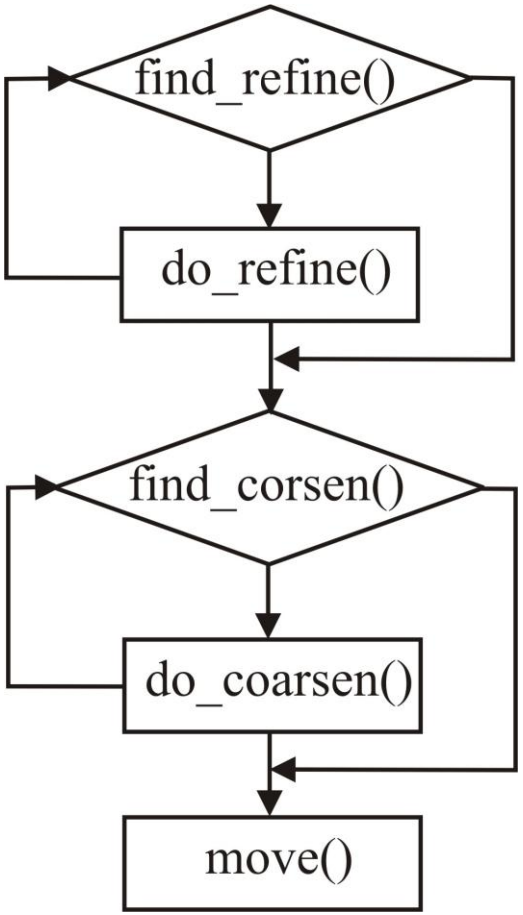
- Вставить цикл prefetch перед циклом с множеством удаленных запросов (DAE)
- Заменить GET на PUT

- Группировать атомарные локально
- Посылать вычисления к данным (active messages или RPC)

Особенности процедуры адаптации

- Нумерация (обход) по Мортону – Z--нумерация



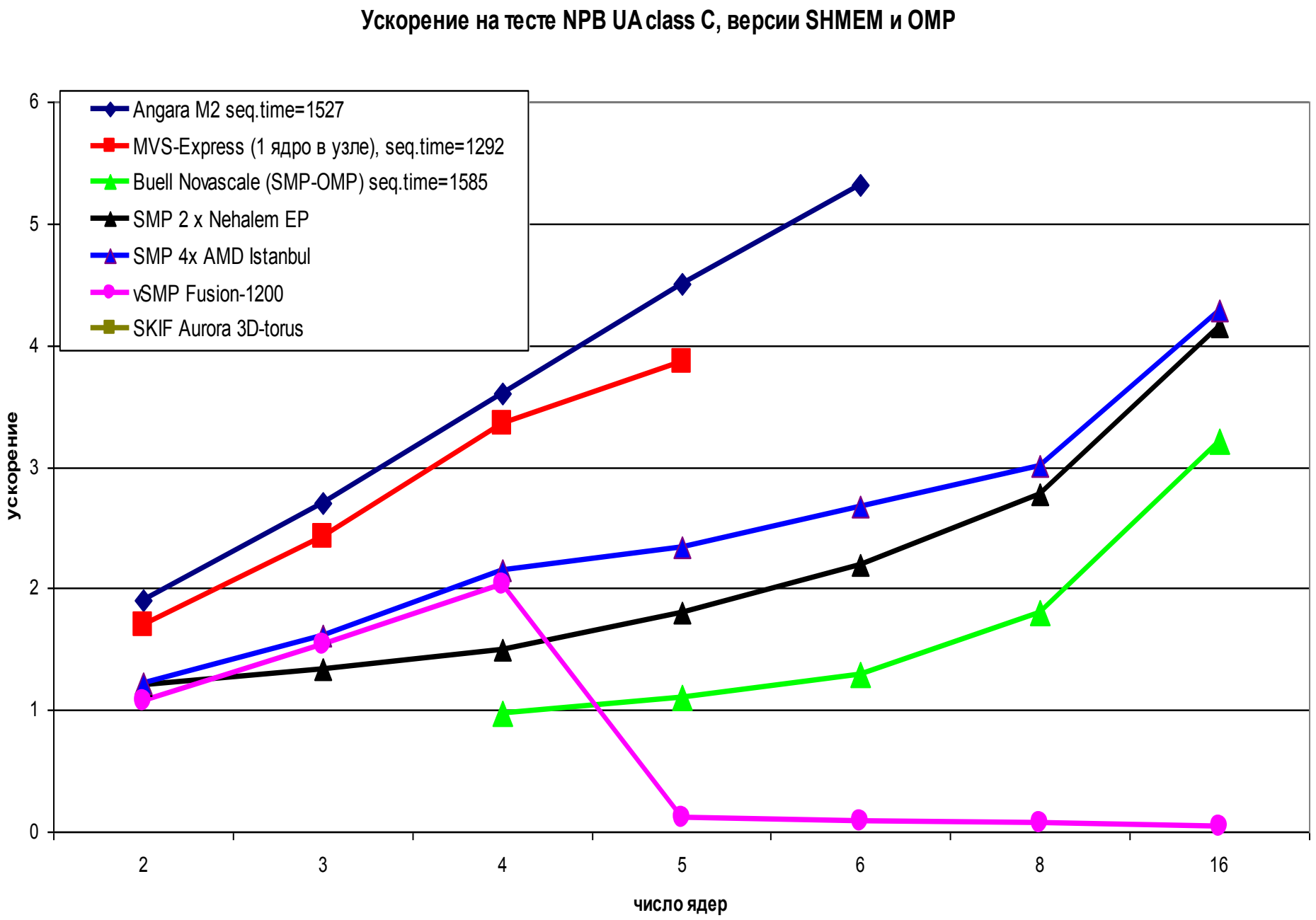


Инкрементальное распараллеливание

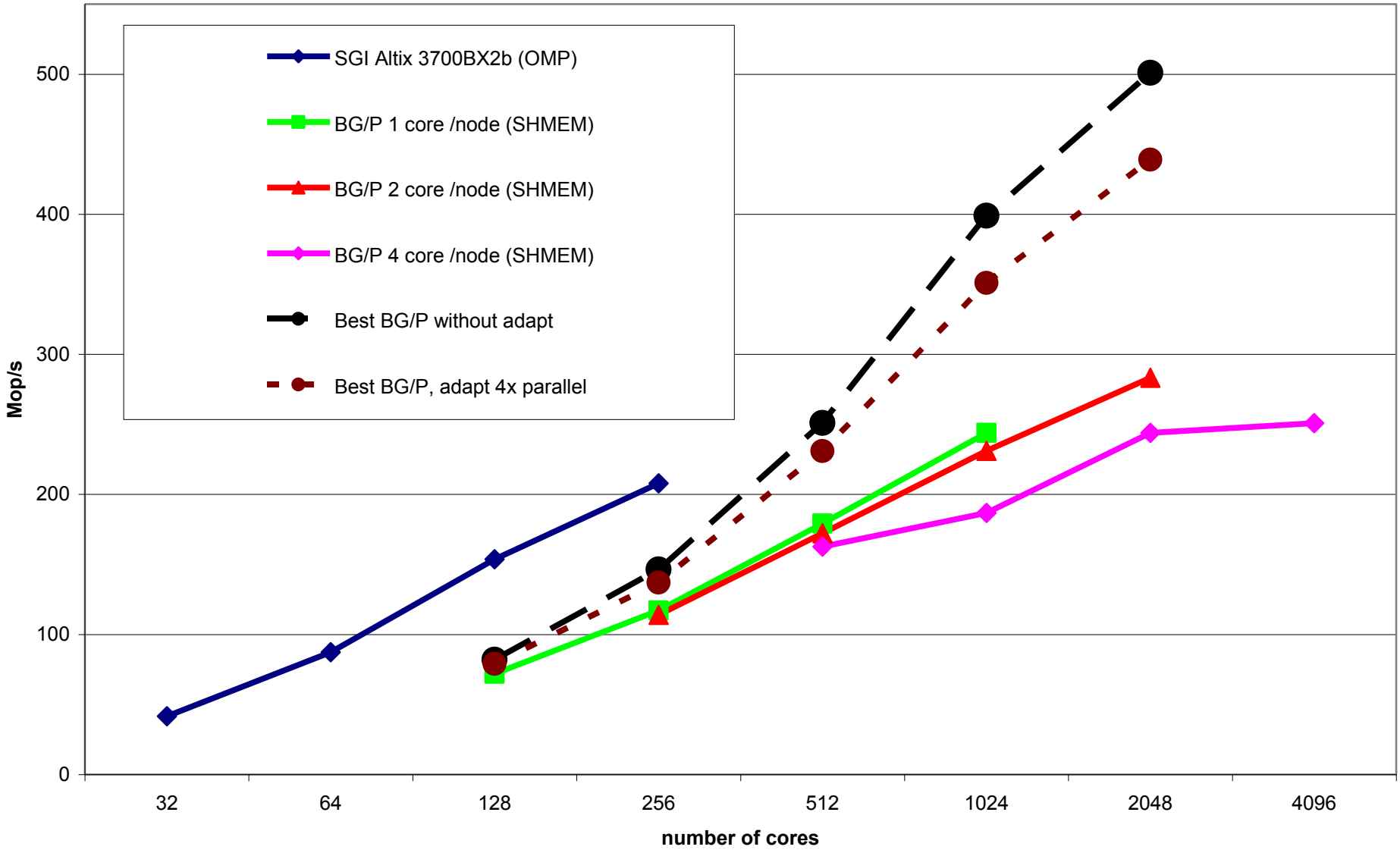
```
do step=0,nstep
c..........parallel code
      call convect_parallel
      call diffuse_parallel

      call shmem_allgatherv(t, t_big,starts,lens)
c……….serial code
      call adapt_serial
end do
```

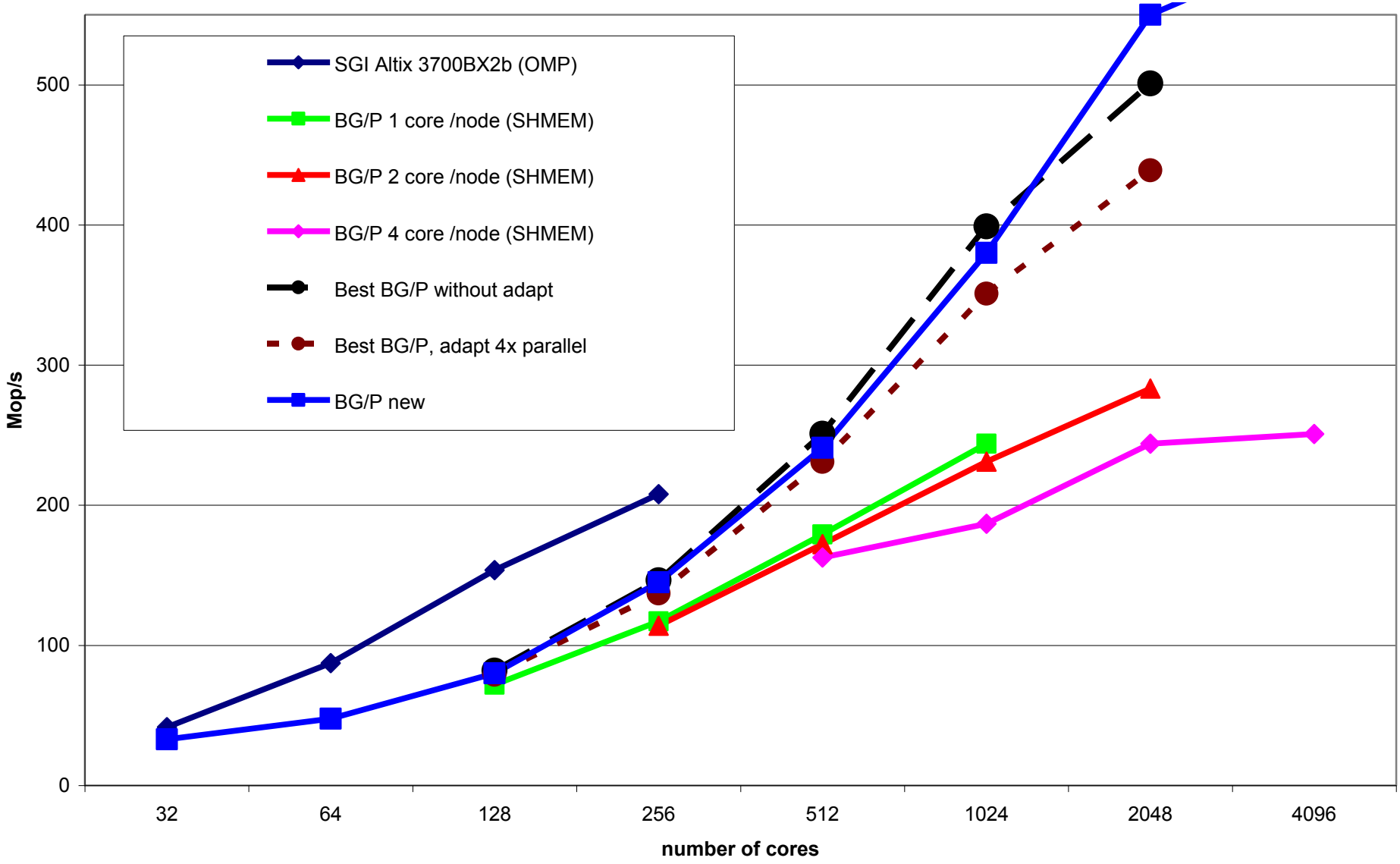
Результаты


Ускорение на тесте NPB UA class C, версии SHMEM и OMP
Angara M2 seq.time=1527
MVS-Express (1 ядро в узле), seq.time=1292
Buell Novascale (SMP-OMP) seq.time=1585
SMP 2 x Nehalem EP
SMP 4x AMD Istanbul
vSMP Fusion-1200
SKIF Aurora 3D-torus
ускорение
0
1
2
3
4
5
6
2
3
4
5
6
8
16
число ядер

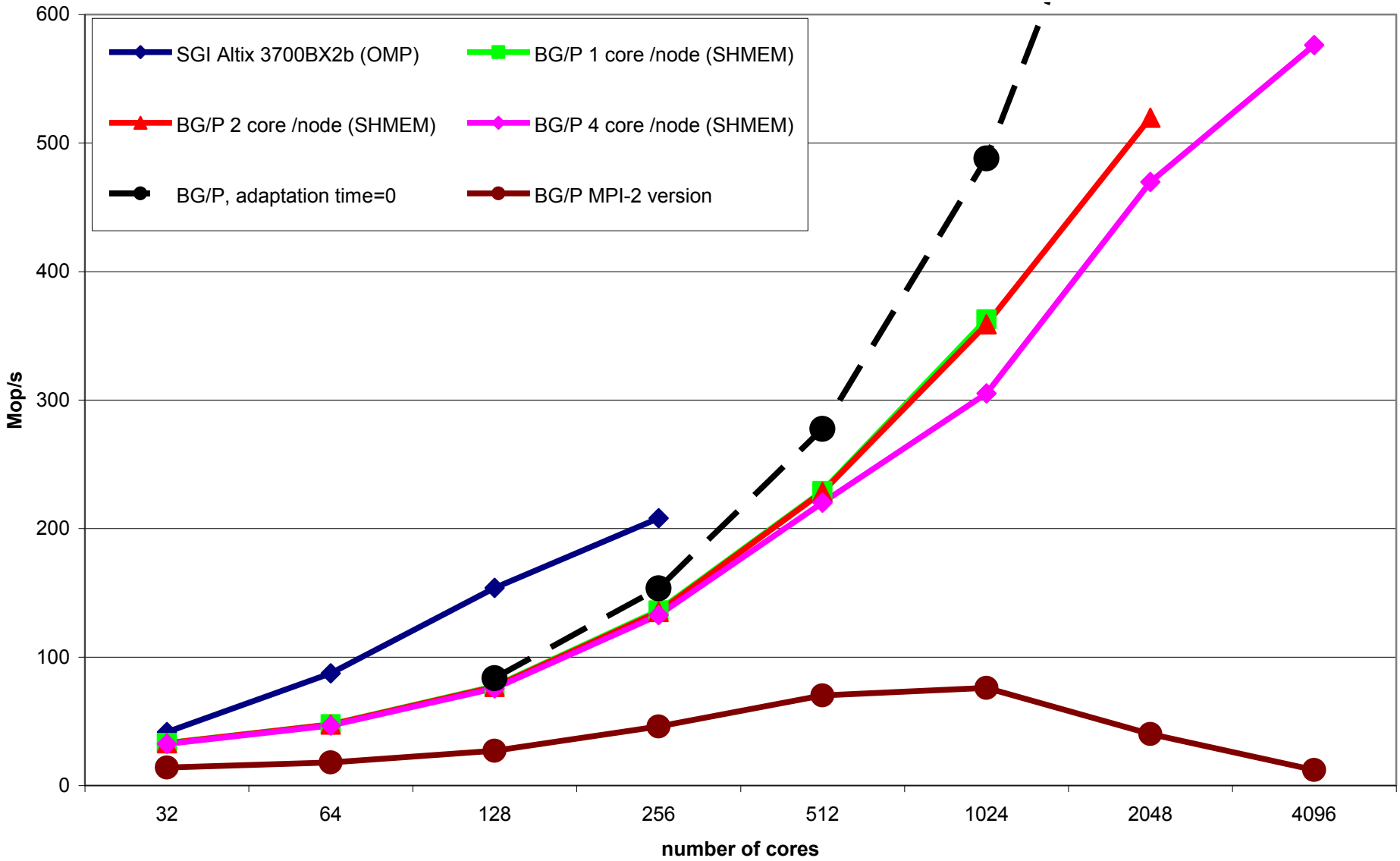
Если изменение сетки последовательно



Если изменение сетки частично распараллелено



Сравнение с MPI-2



Производительность на классе C, миллионы точек согласования / с

	1	32	64	128	256	512	1K	2K	4K
Altix* Itanium	5.9	42	87	154	207				
BG/P DCMF	2.5	33	48	80	142	241	385	557	617
BG/P MPI-2	2.5	14	18	27	46	70	76	40	12

* Данные по SGI Altix предоставлены Jin. Haoqiang из NASA NAS Division

Анализ масштабирования

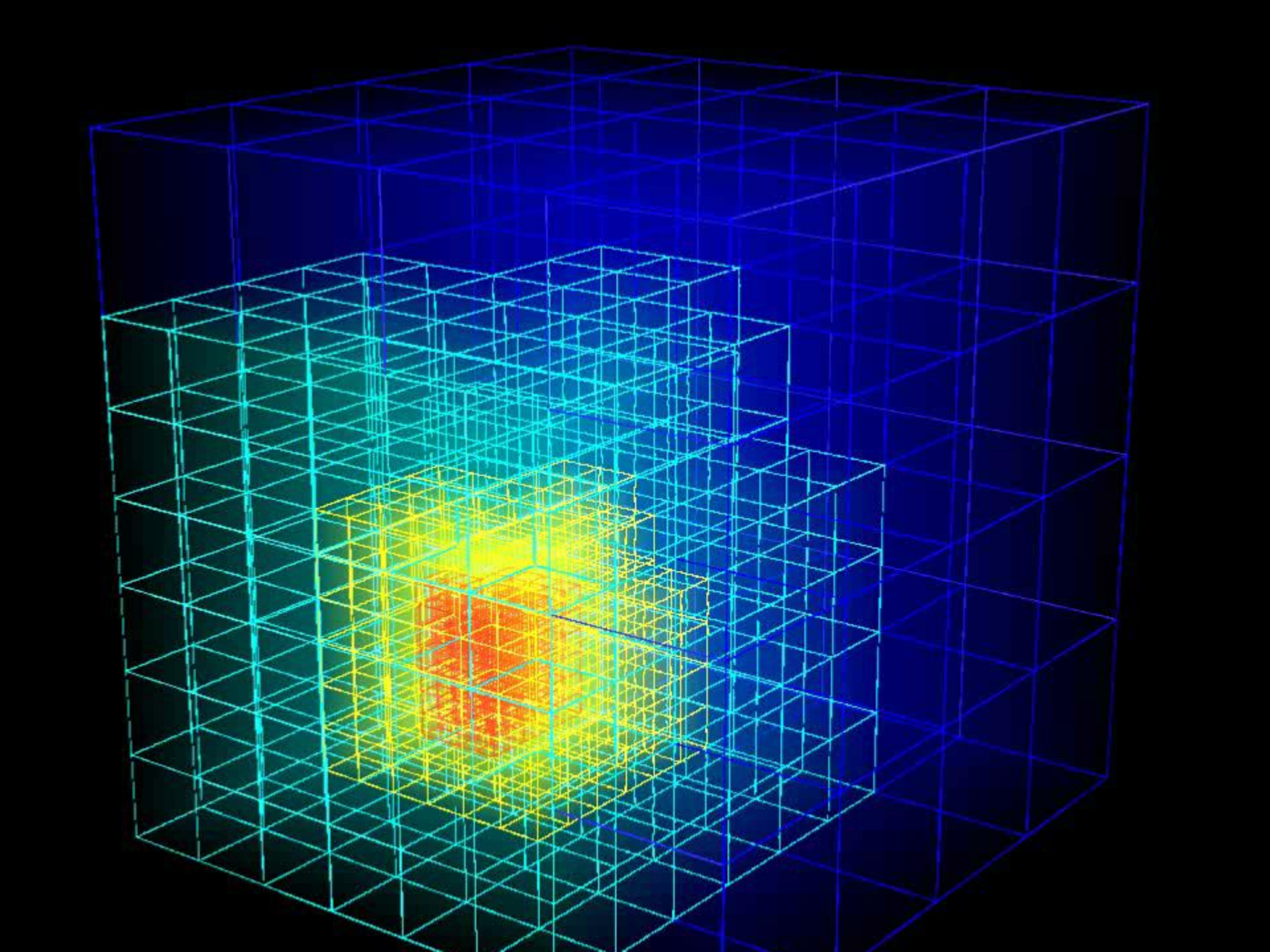
	serial	1	2	4	8	16	32	64	128	256	512	1 K	2 K*	4 K*
Total time, seconds	**3480**	**3784**	**2440**	**1200**	**700**	**410**	**266**	**185**	**113**	**65**	**38**	**24**	**17**	**14**
Mesh Adaptation + Overhead, %	0.4	2	2	3	3	4	4	5	7	11	16	25	35	43
Scatter time, %	12	14	18	20	22	26	30	34	36	37	35	33	29	21
Gather time, %	9	15	19	21	24	27	31	34	34	32	31	27	24	28
Rest computations time, %	78	69	61	56	51	43	35	27	23	20	18	15	13	8
Parallel efficiency, %	100	92	71	73	62	53	41	29	31	21	18	14	11	6

Class D

- 64 ядра клона Altix (Itanium 1.6) : **32** MOp/s
  значит на классе C был false scaling
- Большое потребление памяти SHMEM-версии,
  => 2 ядра в узле для Blue Gene/P с 2GB / node

	64	128	256	512	1024	2048*
BG/P SHMEM	79	115	167	290	490	730

Результаты & Планы

- Мировой рекорд на классе C (617 MOp/s на 4096 ядрах) и классе D (729 MOp/s на 2048 ядрах) Использовалась только одна стойка BG/P !!!
- Готовы к будущим системам с поддержкой PGAS
- MPI-2 RMA не масштабируется
- В Shmem не хватает Active messages для экстремального масштаьирования
- Message Rate очень важна

- Закончить параллелизацию адаптации сетки
- Уменьшить потребление памяти
- Попробовать аггрегацию и другие приемы
- Полуавтоматизировать перевод OMP -> SHMEM