Коллективные алгоритмы межпроцессорного взаимодействия

Николай Игоревич Хохлов

МФТИ, Долгопрудный

15 февраля 2017 г.

Задача интегрирования

Постановка задачи

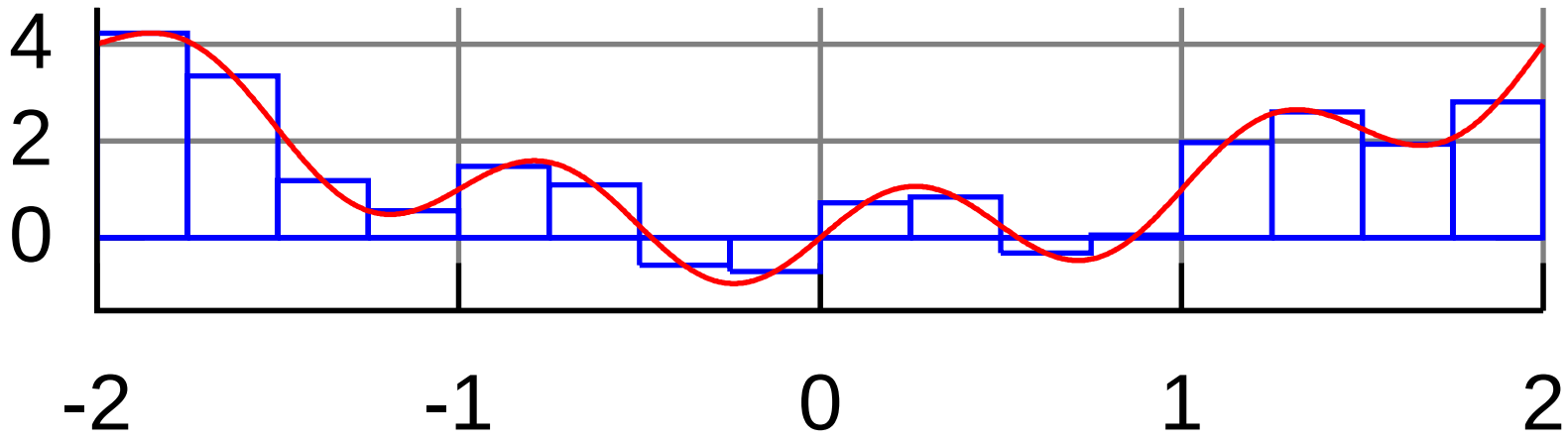
$$I = \int_b^a f(x)\,dx,$$

правило прямоугольника

$$\int_a^b f(x)\,dx \approx (b-a)\,f\left(\frac{a+b}{2}\right).$$

Задача интегрирования

Разбиение отрезка интегрирования на интервалы



Число интервалов N, размер интервала $h = (b - a)/N$,

$$I \approx \sum_i hf(x_i),$$

где $x_i = a + ih/2$, $i = 0 \ldots N - 1$.

Последовательный алгоритм

Algorithm 1 Последовательный алгоритм численного интегрирования

$I = 0$
for $i = 0 \ldots N - 1$ **do**
  $x_i = a + i * h/2$
  $I = I + h * f(x_i)$
end for
print I

Параллельный алгоритм

Число процессов P, номер текущего процесса k.

Algorithm 2 Параллельный алгоритм численного интегрирования

$I_k = 0$
for $i = k; i < N - 1; i+ = P$ **do**
  $x_i = a + i * h/2$
  $I_k = I_k + h * f(x_i)$
end for
if $k = 0$ **then**
  $I = I_k$
  for $i = 1...P - 1$ **do**
    recv I_i *from* i
    $I = I + I_i$
  end for
else
  send I_k *to* 0
end if

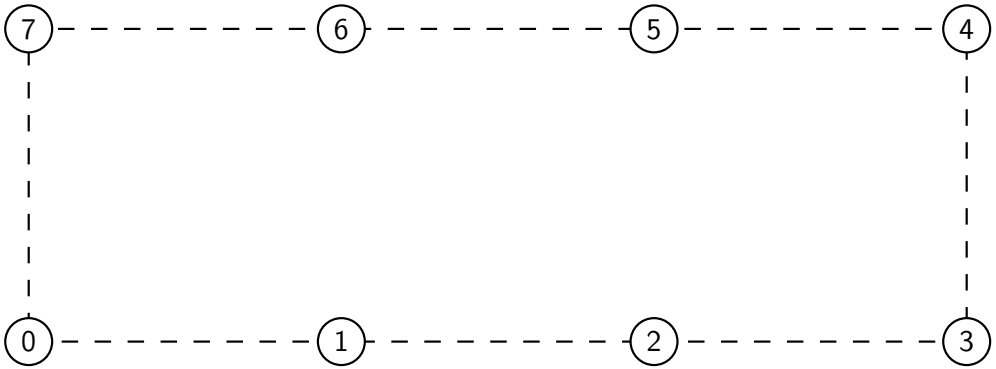
Параллельный алгоритм

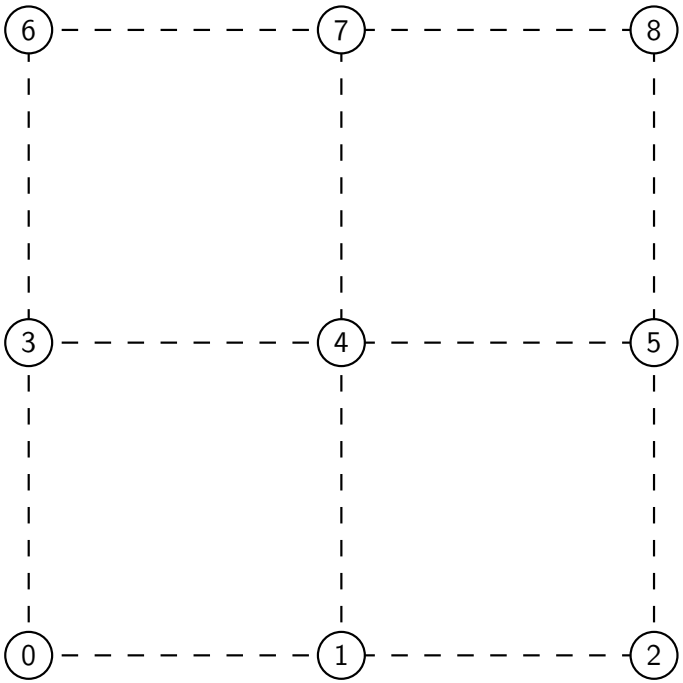
Сложность сбора данных у одного процесса есть $O(P)$.

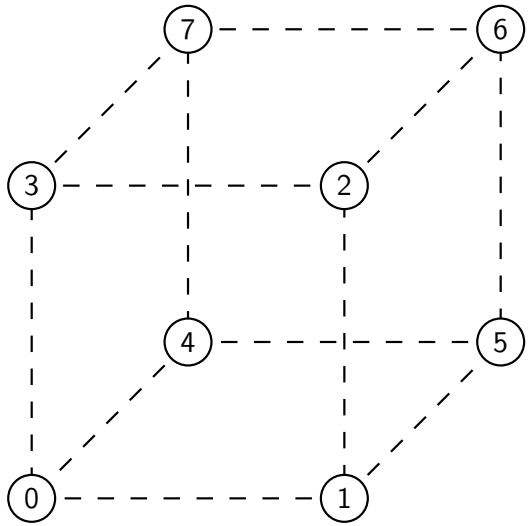
Сбор может быть оптимизирован используя алгоритмы коллективного взаимодействия процессов.

Топологии

кольцо (ring); **сеть** (mesh); **гиперкуб** (hypercube).



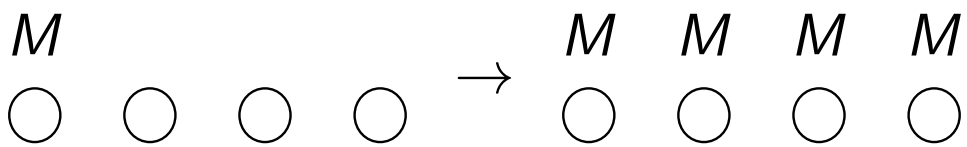




Линейная модель взаимодействия процессов

- t_s – латентность сети;
- t_w – время на передачу одной единицы информации (слово);
- время взаимодействич топа точка-точка (p-t-p) $t_s + t_w m$;
- m – размер сообщения (в словах);
- двусторонние линки между процессами;
- каждый узел может одновременно принимать и отправлять.

One-to-all broadcast



Входные данные:

- Сообщение M хранится локально у процесса root.

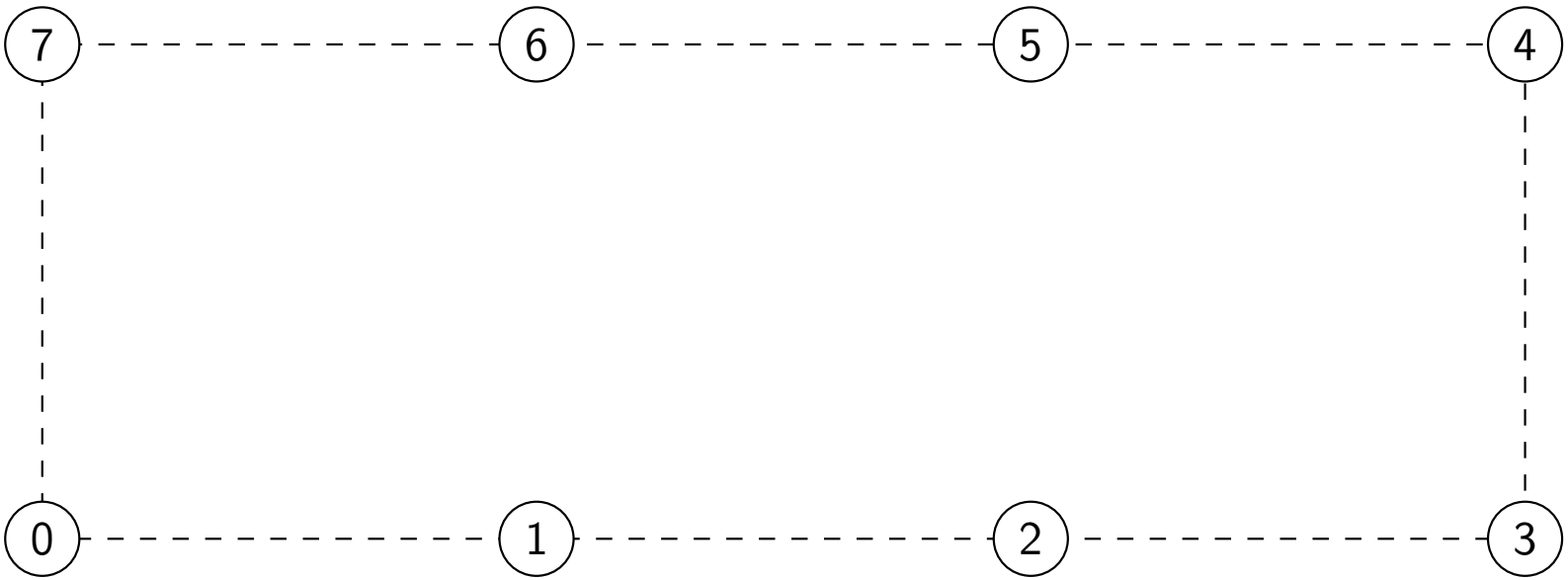
Выходные данные:

- Сообщение M хранится локально на каждом процессе.

One-to-all broadcast

Кольцо

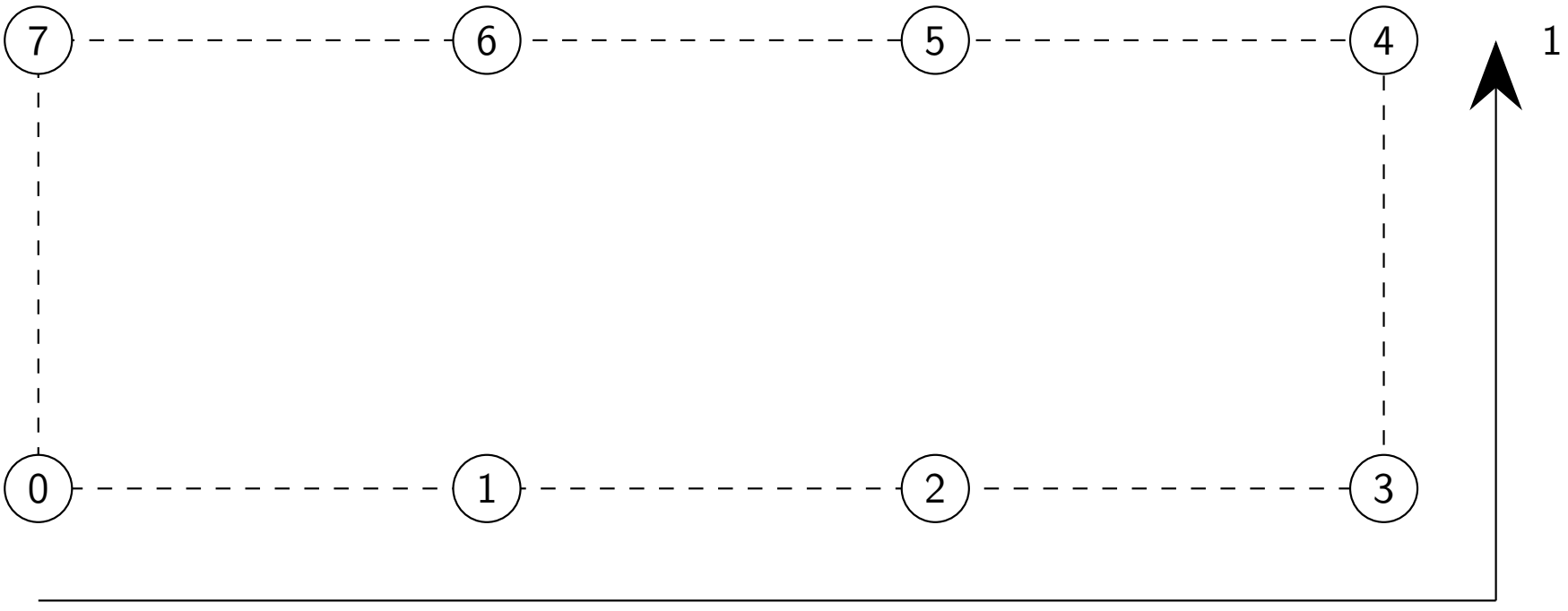
- Рекурсивное удвоение.
- Число активных процессов удваивается каждый шаг.



One-to-all broadcast

Кольцо

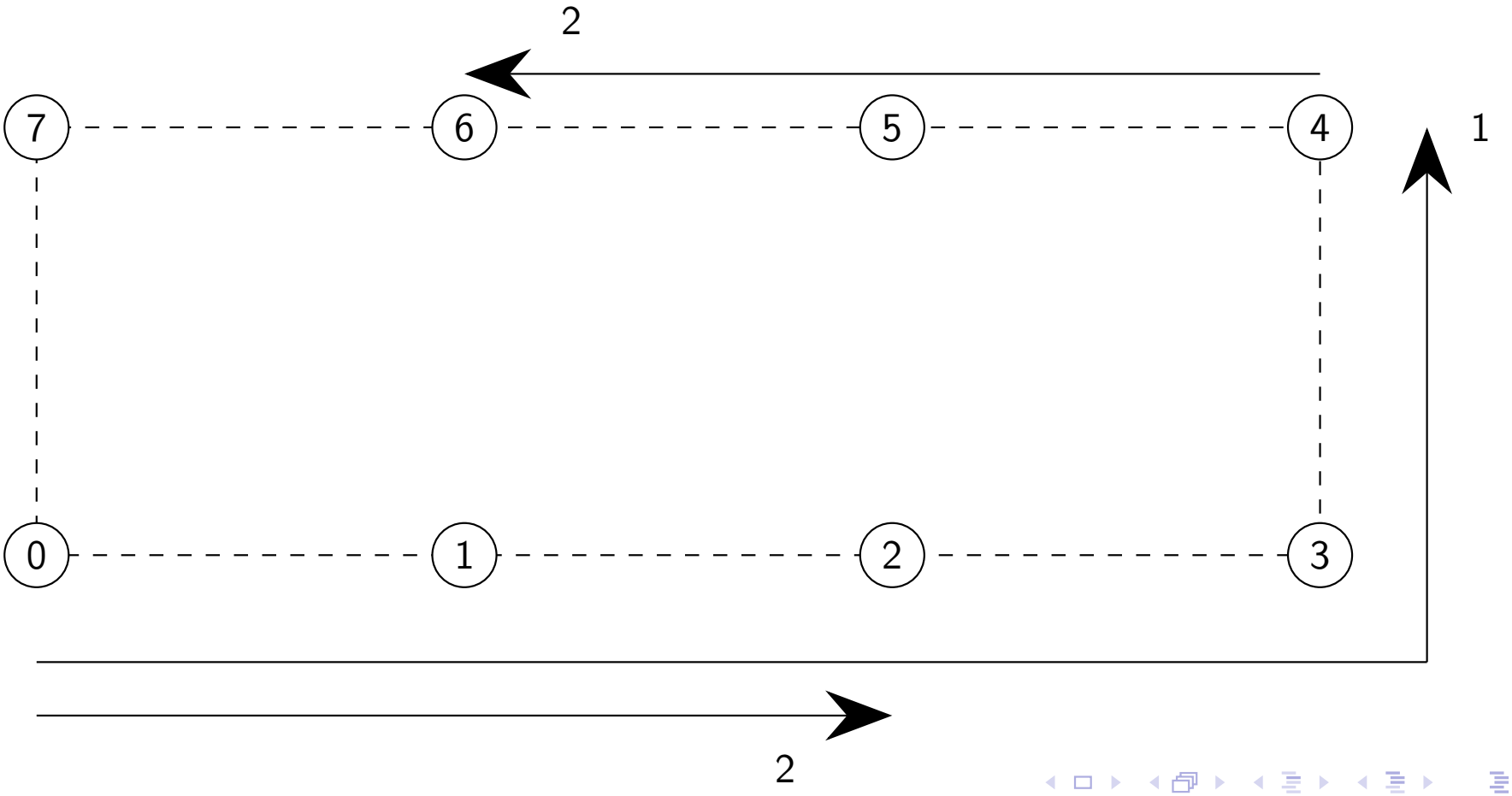
- Рекурсивное удвоение.
- Число активных процессов удваивается каждый шаг.



One-to-all broadcast

Кольцо

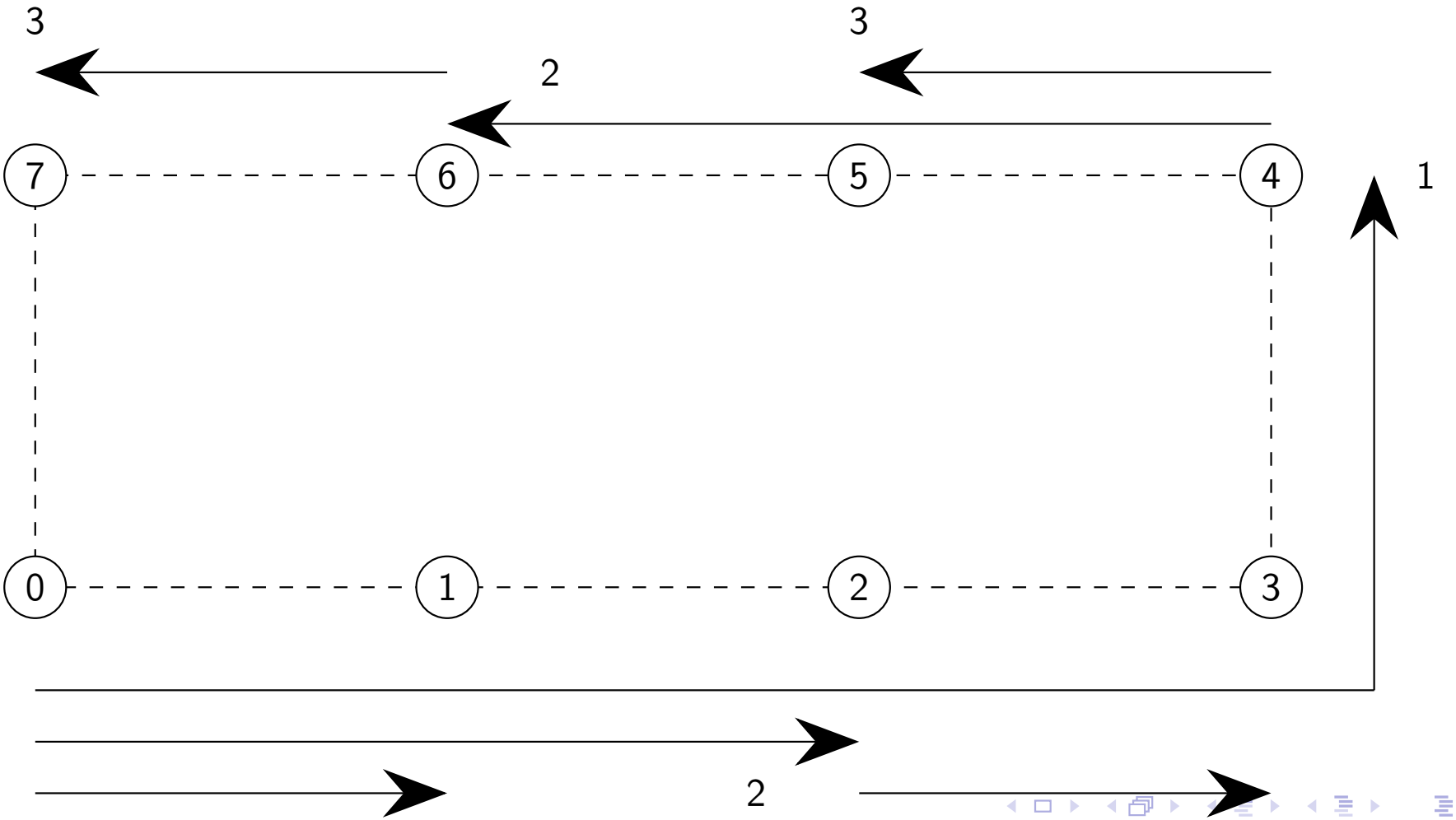
- Рекурсивное удвоение.
- Число активных процессов удваивается каждый шаг.



One-to-all broadcast

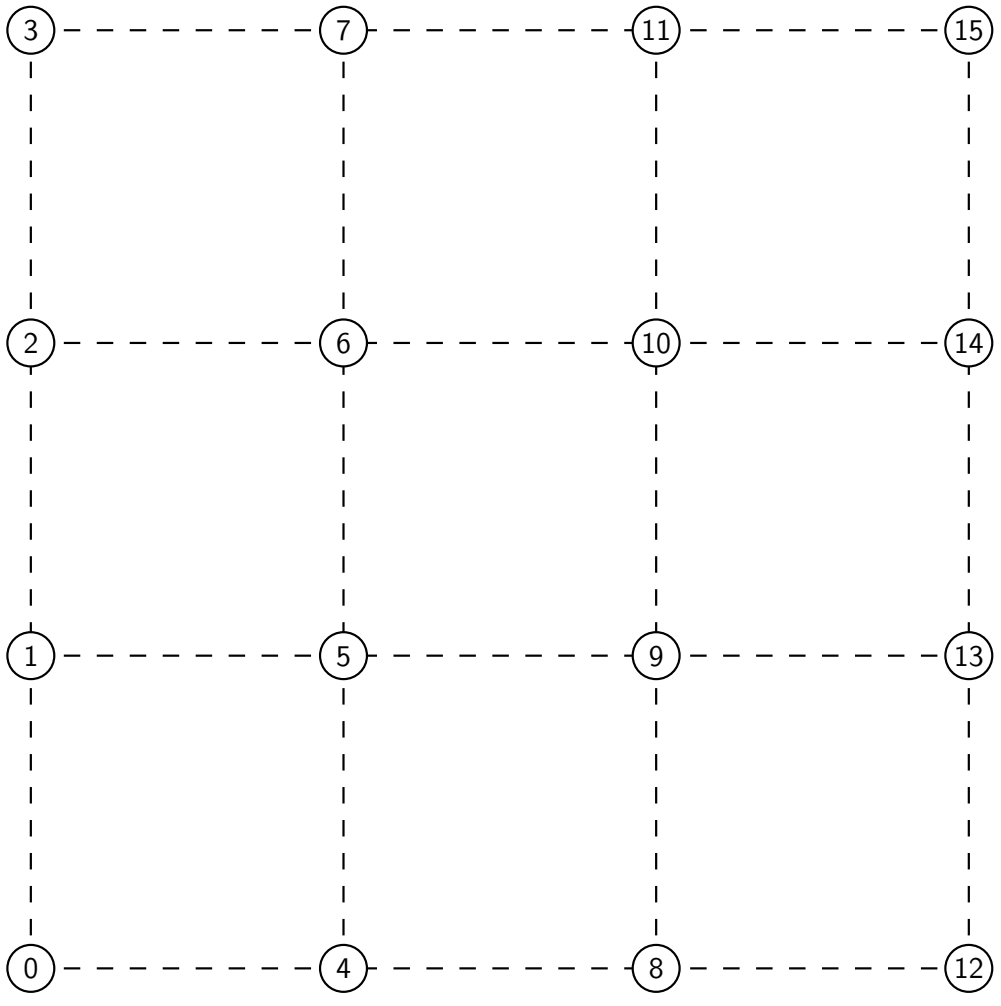
Кольцо

- Рекурсивное удвоение.
- Число активных процессов удваивается каждый шаг.



One-to-all broadcast

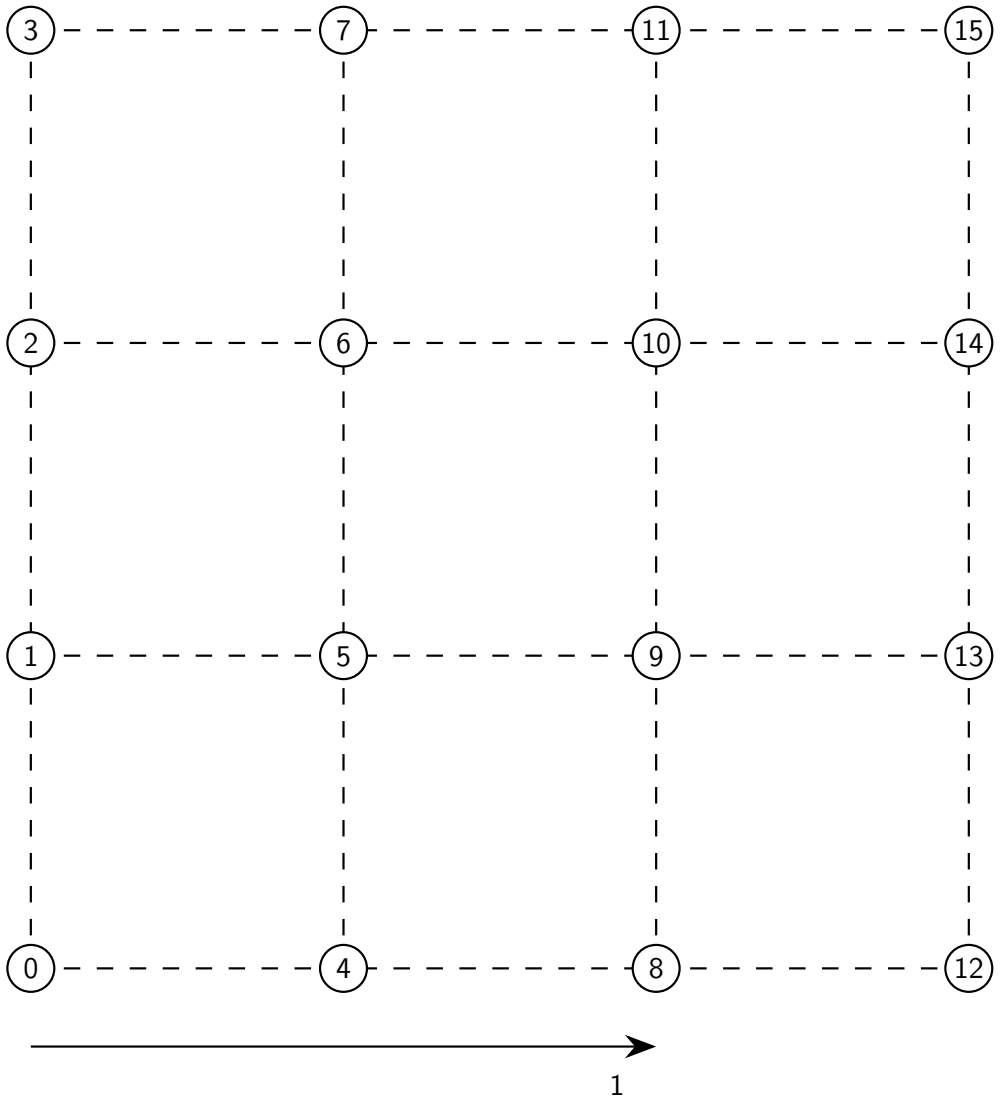
Сеть



- Используется алгоритм для кольца для строки процесса root.
- используется алгоритм кольца для всех колонок параллельно.

One-to-all broadcast

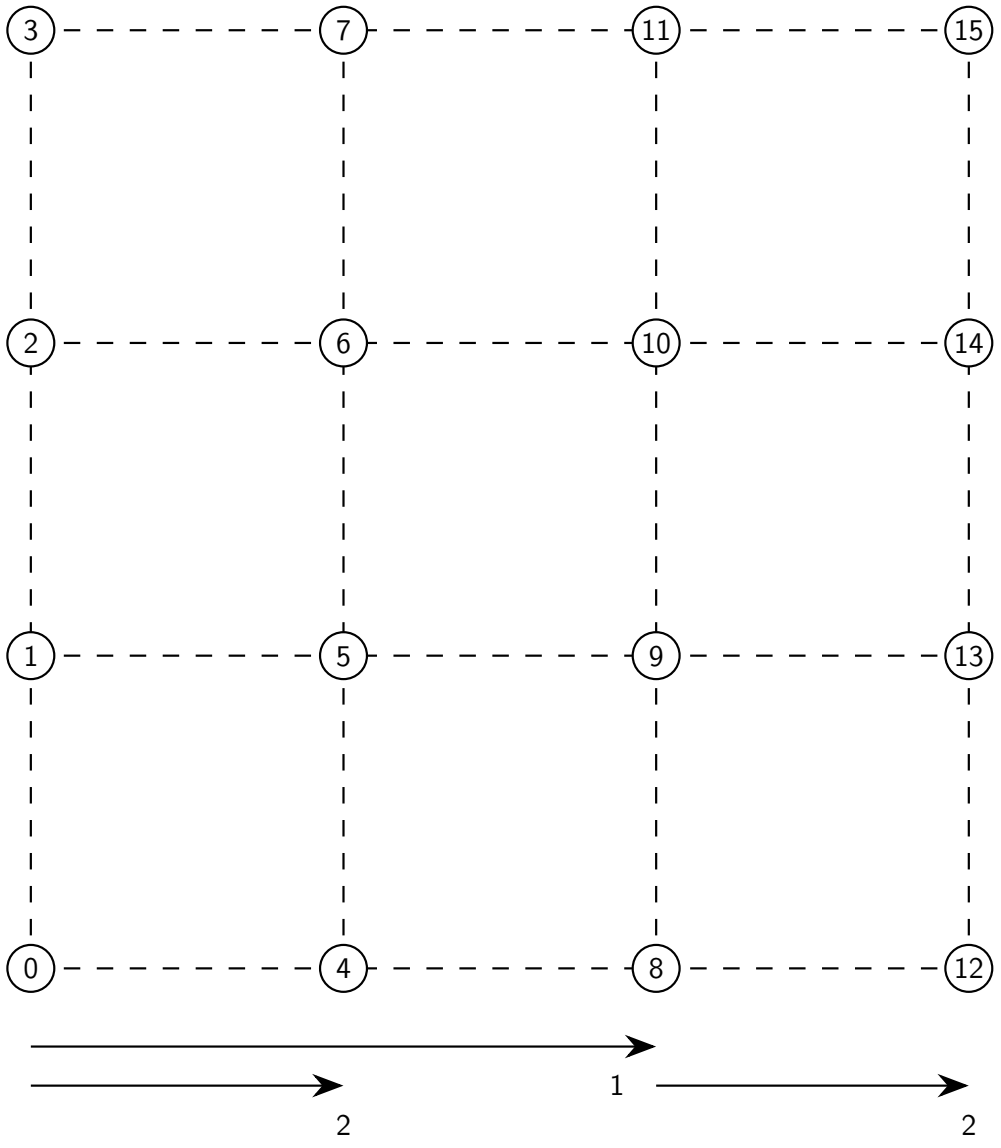
Сеть



- Используется алгоритм для кольца для строки процесса root.
- используется алгоритм кольца для всех колонок параллельно.

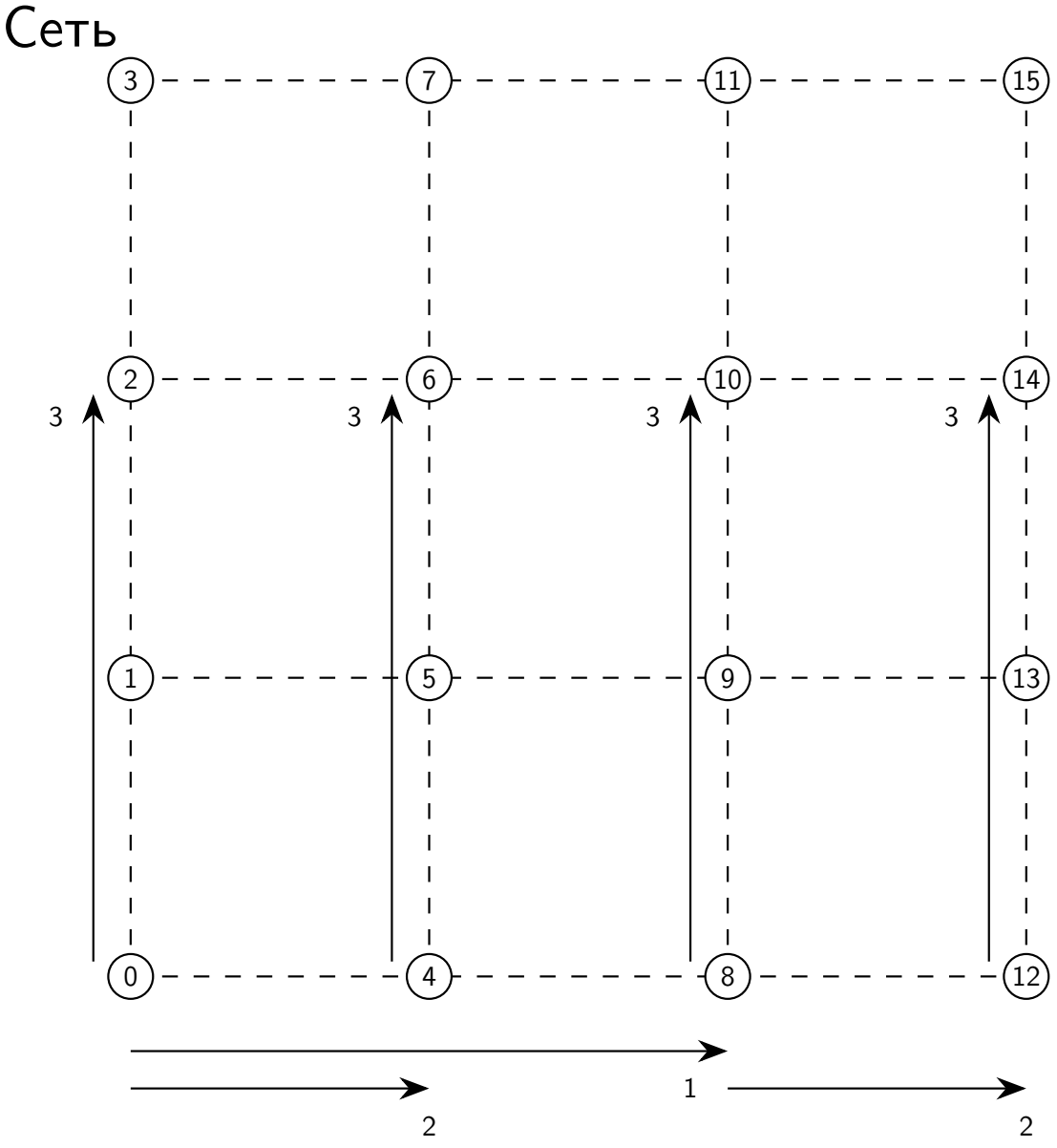
One-to-all broadcast

Сеть



- Используется алгоритм для кольца для строки процесса root.
- используется алгоритм кольца для всех колонок параллельно.

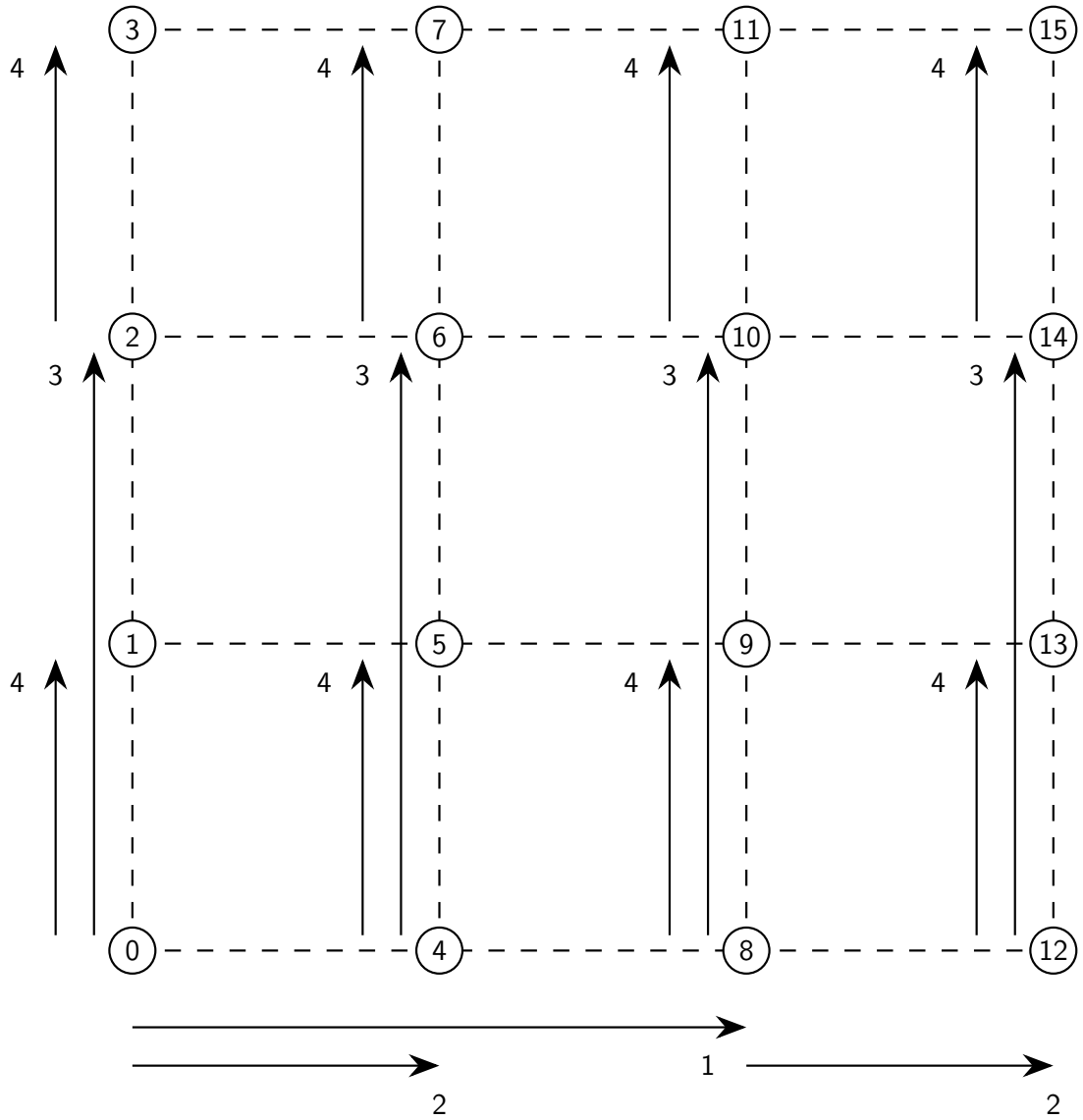
One-to-all broadcast



- Используется алгоритм для кольца для строки процесса root.
- используется алгоритм кольца для всех колонок параллельно.

One-to-all broadcast

Сеть

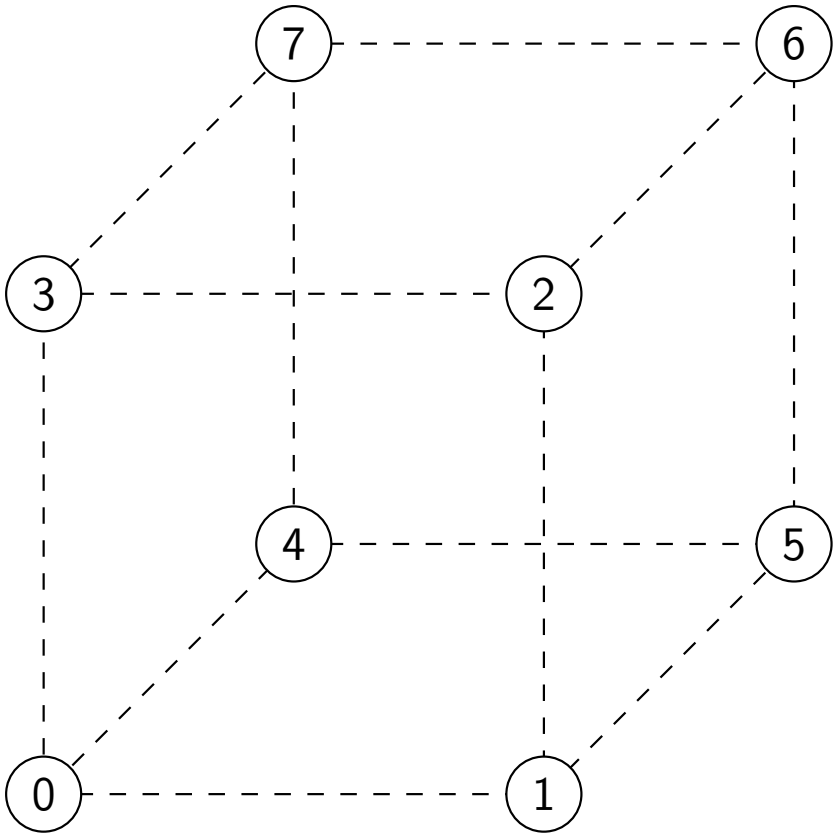


- Используется алгоритм для кольца для строки процесса root.
- используется алгоритм кольца для всех колонок параллельно.

One-to-all broadcast

Гиперкуб

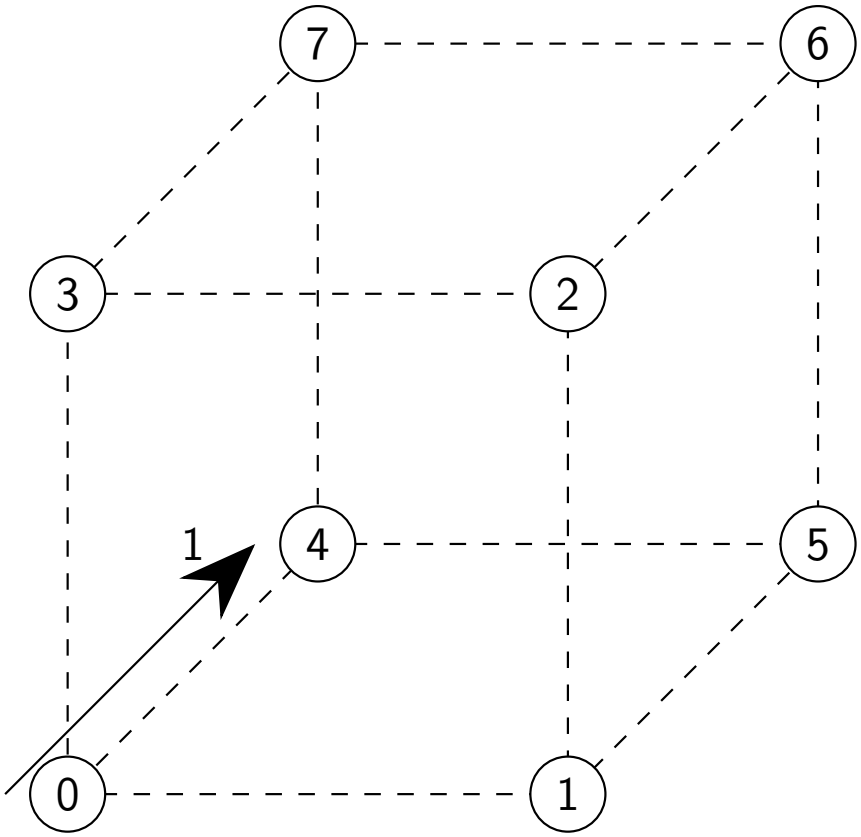
- Обобщенние алгоритма сети на пространство размерности d.



One-to-all broadcast

Гиперкуб

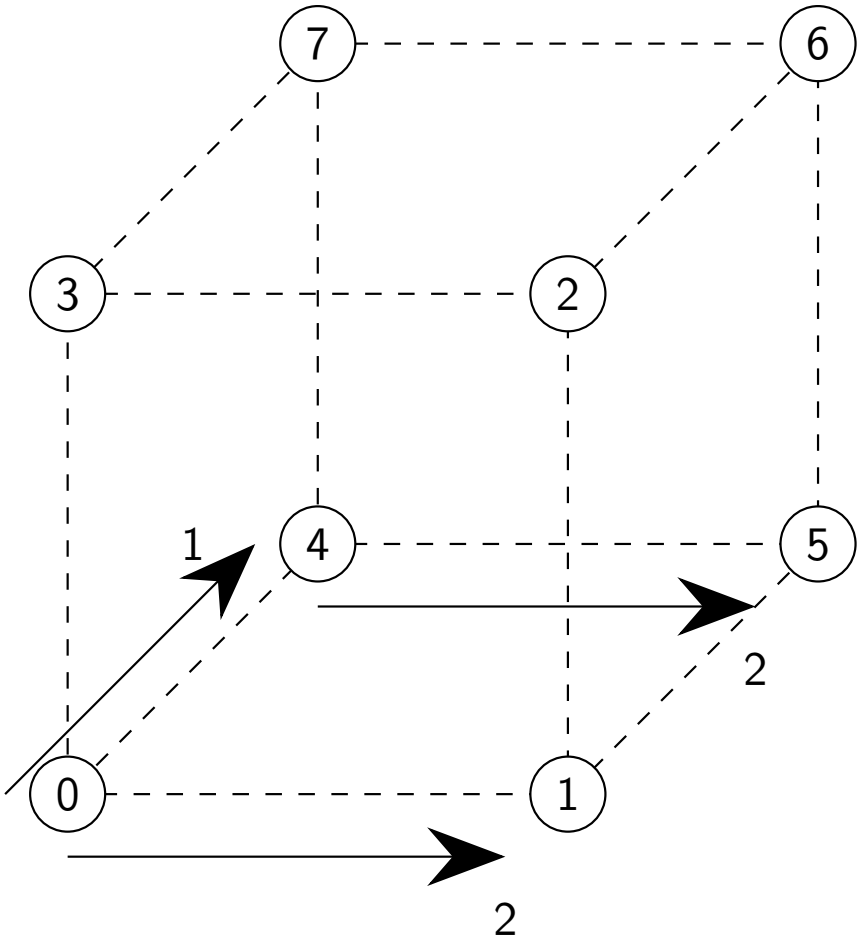
- Обобщенние алгоритма сети на пространство размерности d.



One-to-all broadcast

Гиперкуб

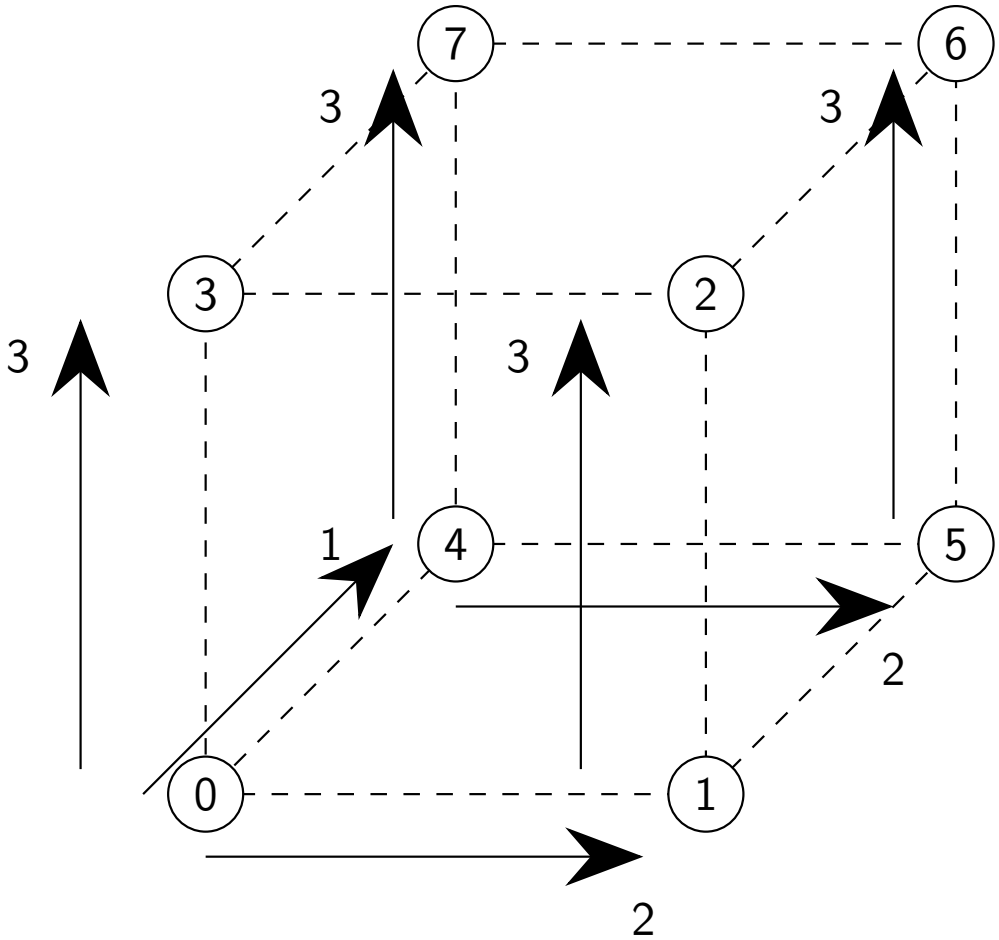
- Обобщенние алгоритма сети на пространство размерности d.



One-to-all broadcast

Гиперкуб

- Обобщенние алгоритма сети на пространство размерности d.



Логические операции

AND (&)

a	b	a AND b
1	1	1
1	0	0
0	1	0
0	0	0

OR (|)

a	b	a OR b
1	1	1
1	0	1
0	1	1
0	0	0

XOR (^)

a	b	a XOR b
1	1	1
1	0	1
0	1	1
0	0	0

Алгоритм

Алгоритм одинаков для всех трех топологий. Номер текущего процесса k.

Algorithm 3 Алгоритм One-to-all broadcast

Пусть $p = 2^d$
$mask = 2^d - 1$ *(выставить все биты)*
for $i = d - 1, ..., 0$ **do**
  $mask = mask\ XOR\ 2^i$ *(очистить бит i)*
  if $k\ AND\ mask = 0$ **then**
    $partner = k\ XOR\ 2^i$ *(у партнера i-й бит другой)*
    if $k\ AND\ 2^i != 0$ **then**
      send M to partner
    else
      recv M from partner
    end if
  end if
end for

Дополнения алгоритма

- Что делать если число процессов не степень двойки ($p \neq 2^d$)?
  - Принять $d = \lceil log_2(p) \rceil$
  - Не взаимодействовать с процессом если $partner \geq p$
- Что делать если процесс root не 0?
  - Перенумеровать процессы по правилу $k = k\ XOR\ root$

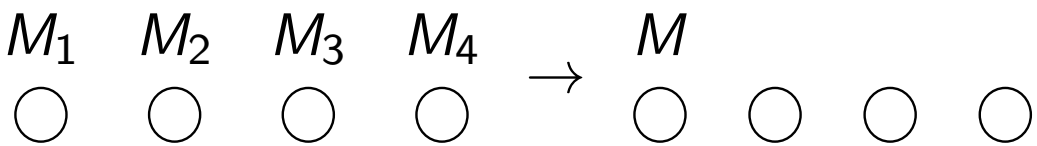
One-to-all broadcast

- Число итераций: $d = log_2 p$
- Время одного взаимодействия: $t_s + t_w m$
- Общее время: $(t_s + t_w m) log_2 p$
- На практике стоит заметить, что взаимодействие p^2 процессов занимает в два раза больше времени, чем p процессов ($log_2 p^2 = 2 log_2 p$)

All-to-one reduce

- All-to-one reduction
- Соответствует вызову MPI_Reduce

All-to-one reduce



$$M = M_1 \oplus M_2 \oplus M_3 \oplus M_4$$

Входные данные:

- Всего p процессов
- Всего p сообщений M_k, где $k = 0, 1, ..., p-1$
- Сообщение M_k хранится локально у процесса с номером k
- Ассоциативная операция редукции $\oplus$ (например $+, \times, min, max$ и т. д.)

Выходные данные:

- Результат операции $M = M_1 \oplus ... \oplus M_{p-1}$ локально на одном процессе с номером root

Алгоритм

Algorithm 4 Алгоритм all-to-one reduction

Пусть $p = 2^d$
$mask = 0$, $sum = M$
for $i = 0, ..., d - 1$ **do**
  if k *AND* $mask = 0$ **then**
    $partner = k$ *XOR* 2^i
    if k *AND* $2^i != 0$ **then**
      send sum to partner
    else
      recv M from partner
      $sum = sum \oplus M$
    end if
  end if
  $mask = mask$ *XOR* 2^i
end for

Дополнения алгоритма

- Что делать если число процессов не степень двойки ($p \neq 2^d$)?
  - Принять $d = \lceil log_2(p) \rceil$
  - Не взаимодействовать с процессом если $partner \geq p$
- Что делать если процесс root не 0?
  - Перенумеровать процессы по правилу $k = k\ XOR\ root$

Задача 1

- Реализовать задачу численного интегрирования, используя схему сборки данных типа гиперкуб.
- Отметка за задачу 1 балл.