Parallel Sorting on Recursive Dual-Nets

Yamin Li, Department of Computer Science, Hosei University, yamin@hosei.ac.jp
Shietung Peng, Department of Computer Science, Hosei University, speng@hosei.ac.jp
Wanming Chu, Department of Computer Hardware, University of Aizu, w-chu@u-aizu.ac.jp

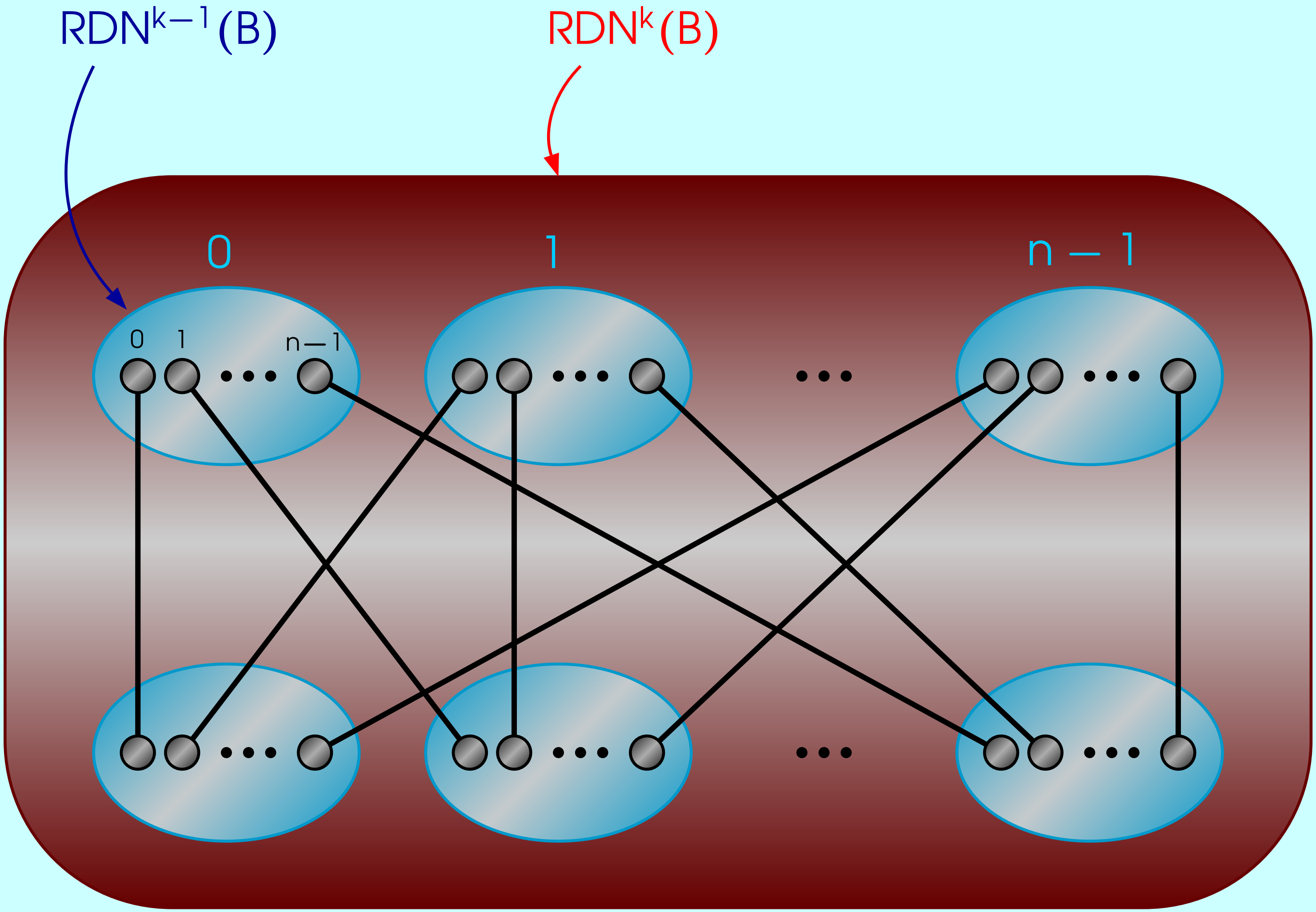
Main Contents

- Introduce a recursive interconnection network
  - RDN: Recursive Dual-Net
    - Can build a very large system
    - Low node degree
    - Short diameter
- Implement a parallel sorting algorithm on RDN
  - Efficient on
    - Computation time and
    - Communication time

Interconnection Networks

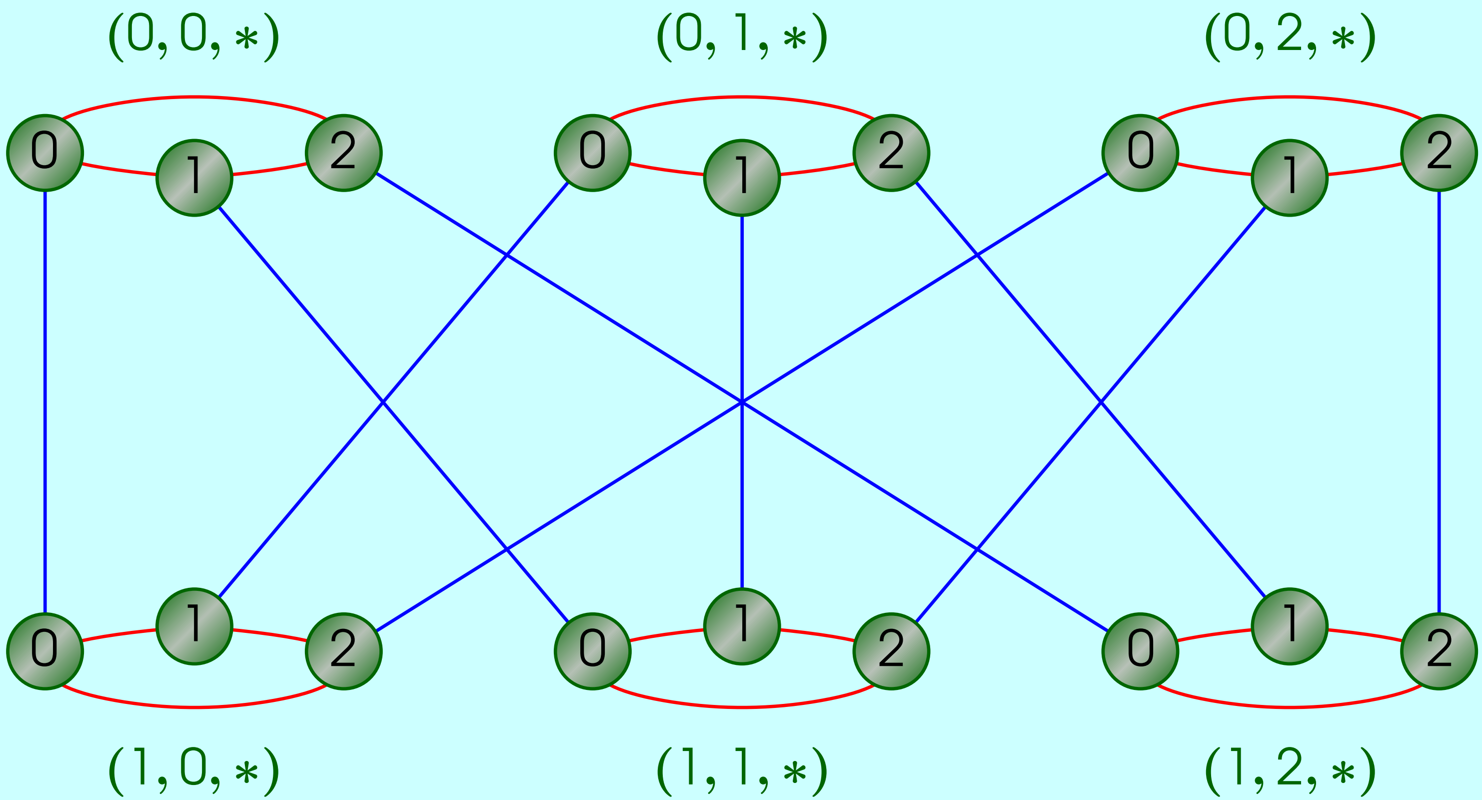
- The modern high-performance supercomputers consist of hundreds of thousands of CPUs
- In the near future, the number of CPUs in supercomputers will reach several millions
- How to connect these extremely large number of CPUs is an important issue for achieving high performance of the supercomputers
- A "good" interconnection network should use a small number of links and meanwhile keep the diameter as shorter as possible
- The symmetric structure and efficient routing should also be considered

The Recursive Dual-Construction



$$\text{if } |RDN^{k-1}(B)| = n, \; |RDN^{k}(B)| = 2n^2$$

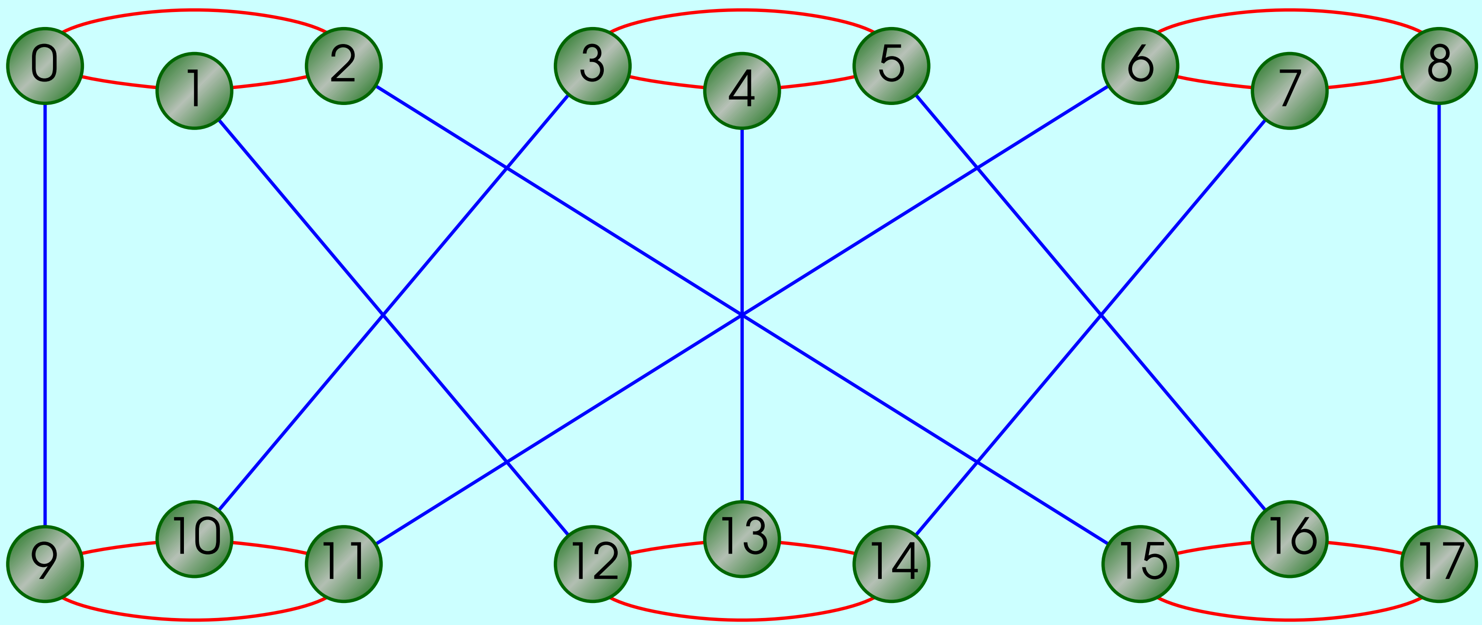
The Recursive Dual-Net $RDN^1(B(3))$



Base network $RDN^0(B)$ example: 3-node Ring

A link connects nodes (0,X,Y) and (1,Y,X)

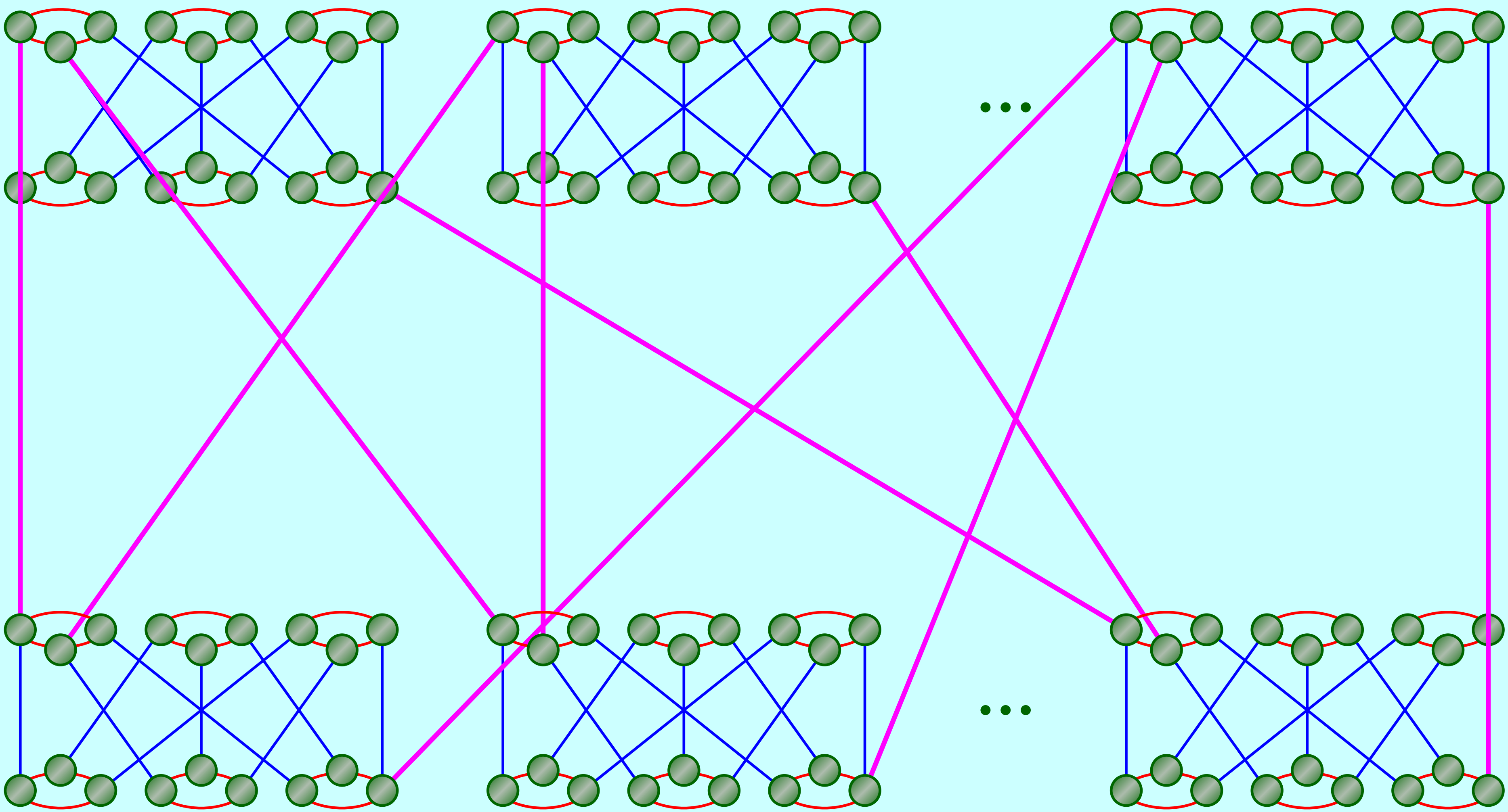
The Recursive Dual-Net $RDN^1(B(3))$



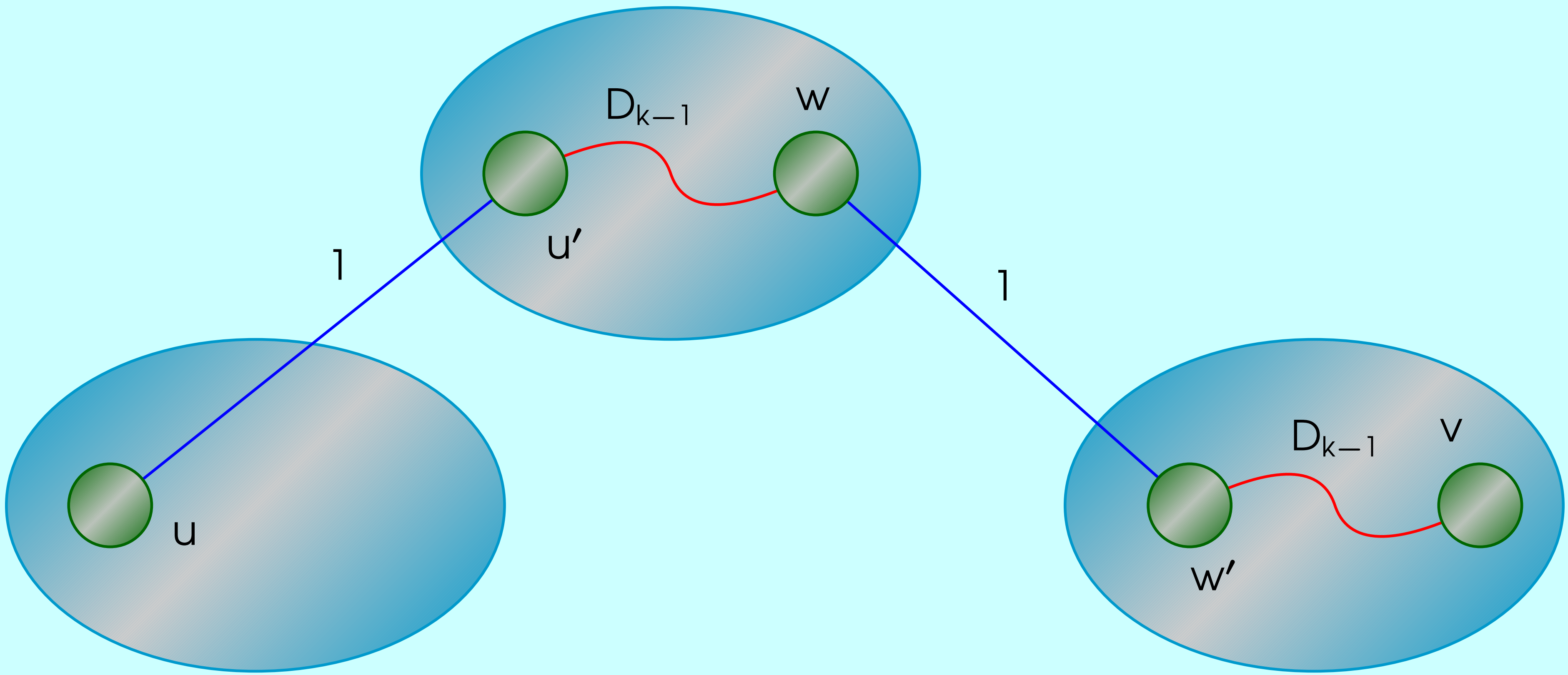
A node can be presented with an integer

Ex. the ID of node $(1, 1, 2)$ is $1 * 3^2 + 1 * 3 + 2 = 14$

The Recursive Dual-Net $RDN^2(B(3))$



The Diameter of the RDN

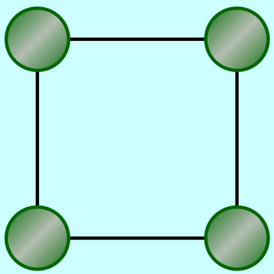


$$D_k = 2D_{k-1} + 2 = 2^k * D_0 + 2^{k+1} - 2,$$

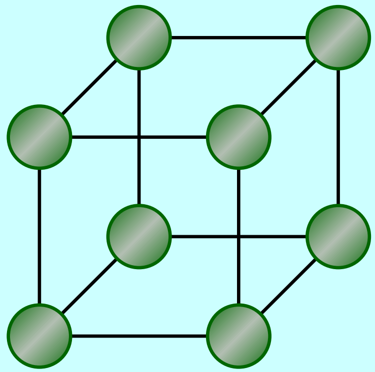
where D_0 is the diameter of the base network

Examples of the Base Network: $RDN^0(B)$

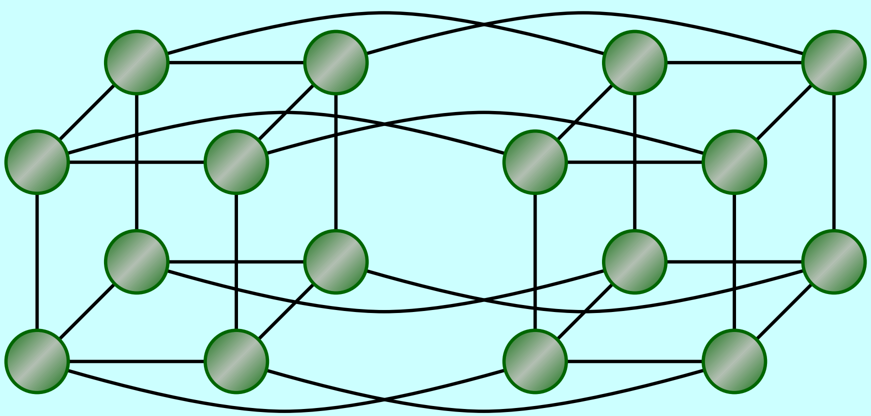
Hypercubes



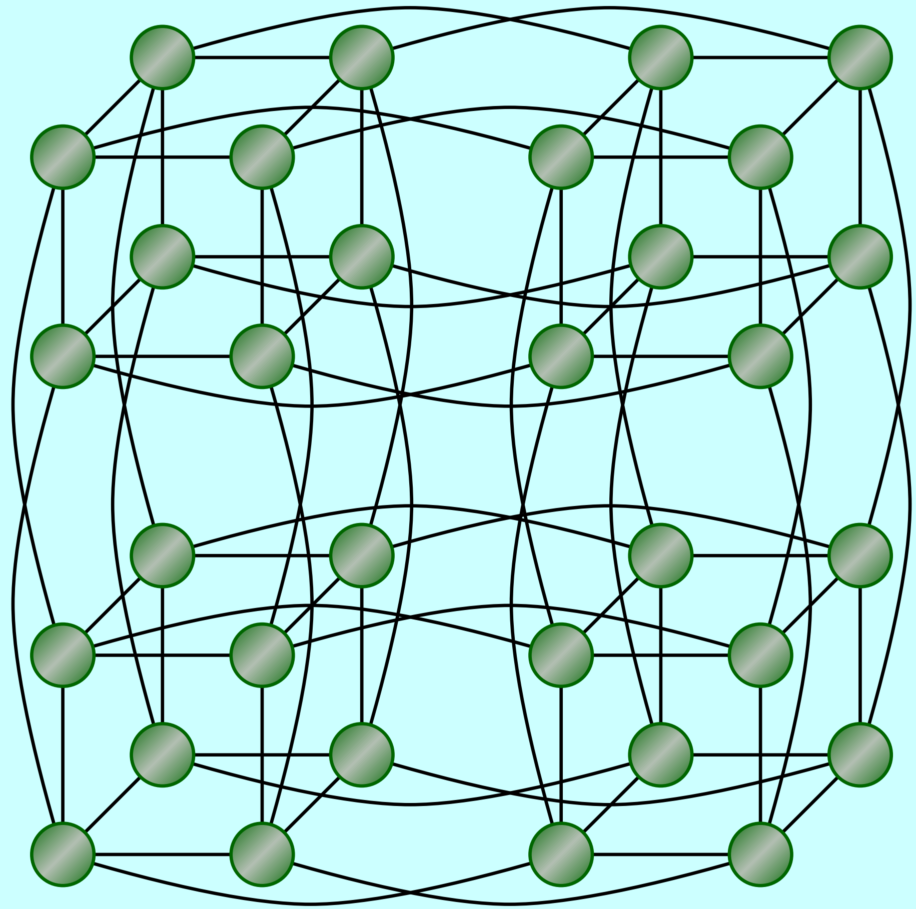
2-Cube



3-Cube



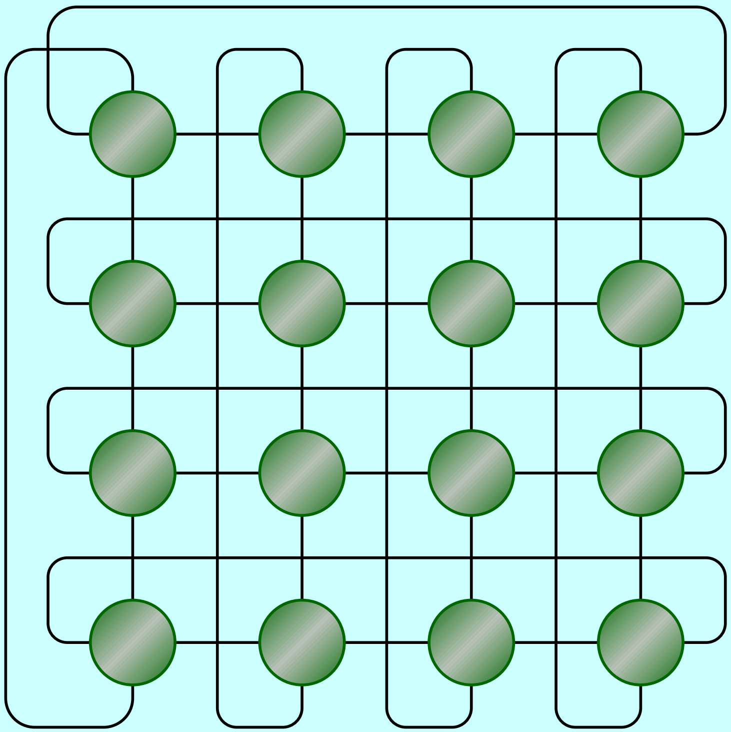
4-Cube



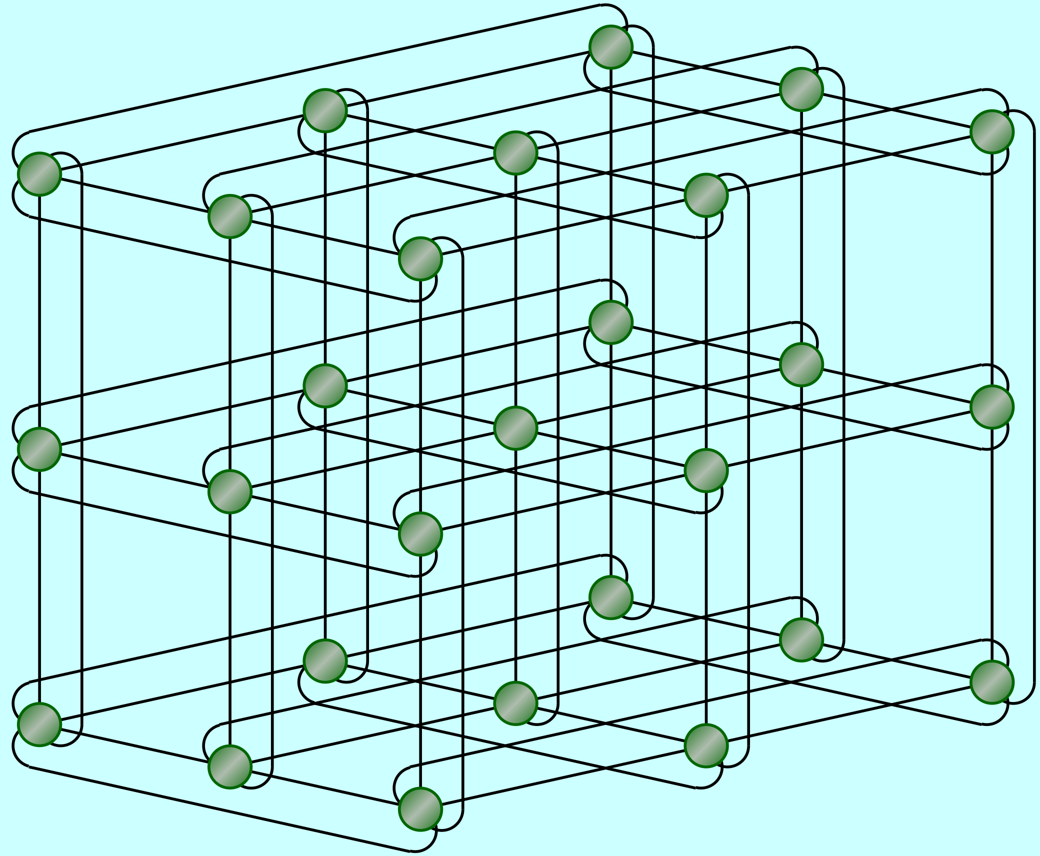
5-Cube

Examples of the Base Network: $RDN^0(B)$

Torus



2D Torus



3D Torus

Topological Properties

Networks	#Nodes $\log_2 \|(G)\|$	Node-degree $d(G)$	Diameter $D(G)$
3D Torus	$x * y * z$	6	$(x + y + z)/2$
n-cube	2^n	n	n
CCC(n)	$n * 2^n$	3	$2n + \lfloor n/2 \rfloor - 2$
WK(n, t)	n^t	n	$2^t - 1$
DC(n)	2^{2n-1}	n	$2n$
$RDN^k(B(m))$	$(2m)^{2^k}/2$	$d_0 + k$	$2^k * D_0 + 2^{k+1} - 2$

$$\text{Cost Ratio}: \; CR(G) = \frac{d(G) + D(G)}{\log_2 |(G)|}$$

Cost Ratio

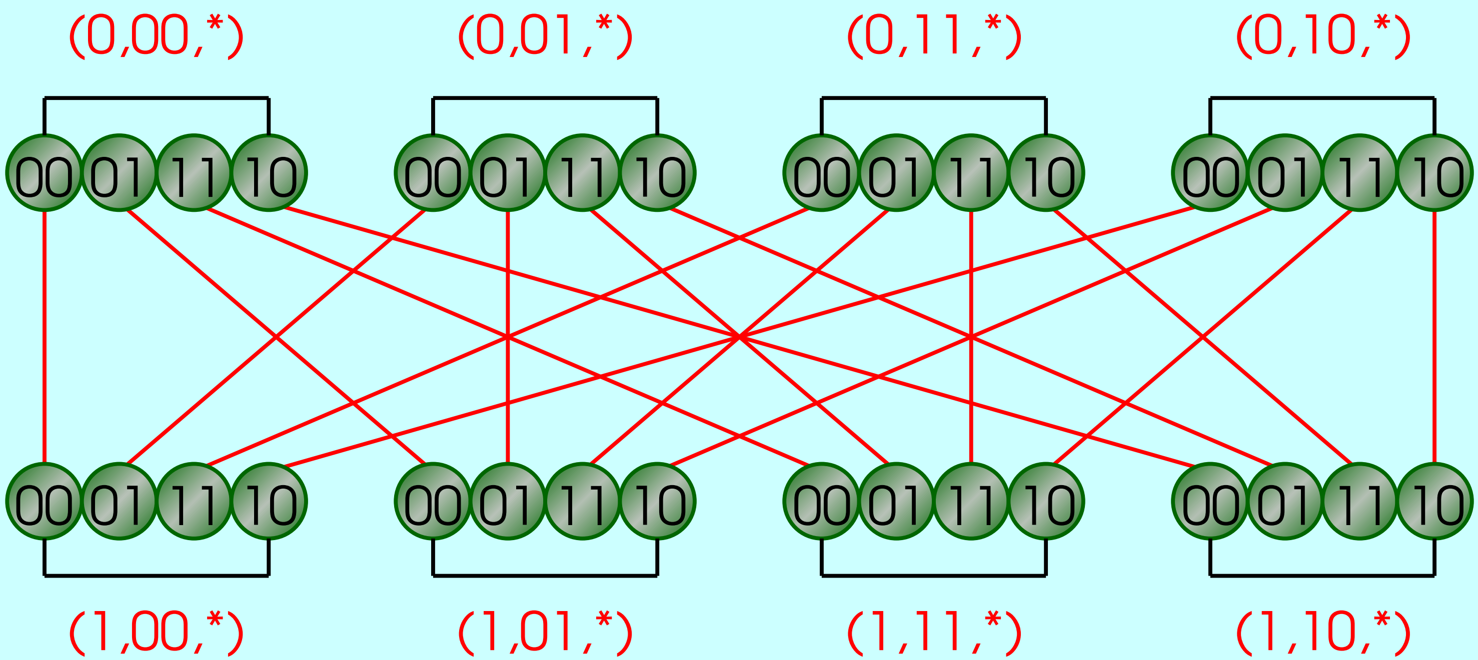
Network	Nodes	Degree	Diameter	CR
10-cube	1,024	10	10	2.00
$RDN^1(B(25))$	1,250	5	10	1.46
$RDN^1(B(27))$	1,458	7	8	1.43
3D-Torus(10)	1,000	6	15	2.11
22-cube	4,194,304	22	22	2.00
$RDN^2(B(25))$	3,125,000	6	22	1.30
$RDN^2(B(27))$	4,251,528	8	18	1.18
3D-Torus(160)	4,096,000	6	240	11.20

$RDN^2(Q(m))$ — Base Network Q(m): m-Cube

Address format:

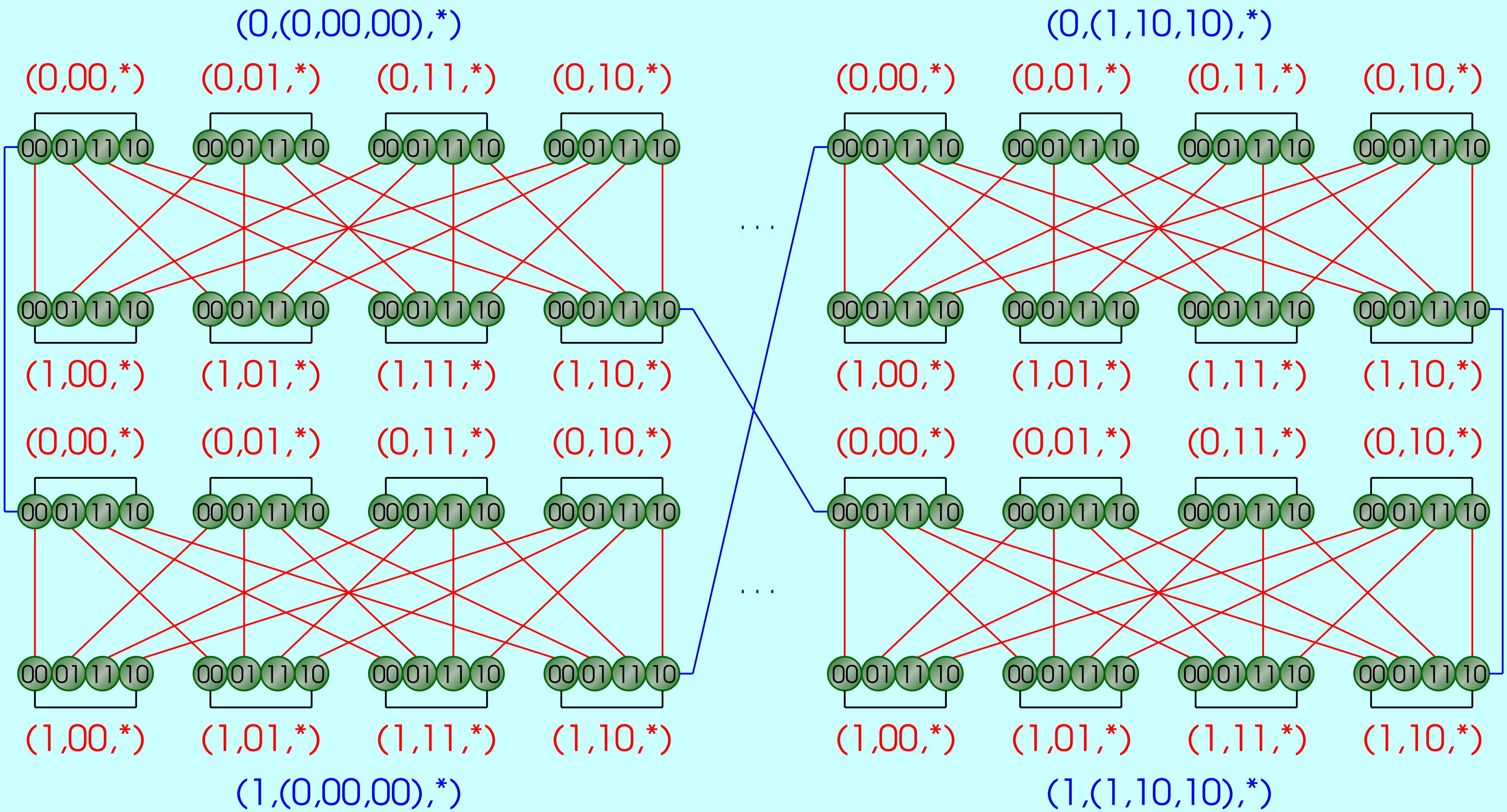
1	1	m	m	1	m	m
c_2	c_1	$b^3_{m-1} \cdots b^3_0$	$b^2_{m-1} \cdots b^2_0$	c_0	$b^1_{m-1} \cdots b^1_0$	$b^0_{m-1} \cdots b^0_0$

$RDN^1(Q(2))$:



RDN^2(Q(m)) — Base Network Q(m): m-Cube

RDN^2(Q(2)):



Routing Path Example

0) 0 0 00 00 0 00 00 $\longrightarrow$ 0 0 00 00 0 00 01

1) 0 0 00 00 0 00 00 $\longrightarrow$ 0 0 00 00 0 00 10

2) 0 0 00 00 0 00 00 $\longrightarrow$ 0 0 00 00 1 00 00 $\longrightarrow$
0 0 00 00 1 00 01 $\longrightarrow$ 0 0 00 00 0 01 00

3) 0 0 00 00 0 00 00 $\longrightarrow$ 0 0 00 00 1 00 00 $\longrightarrow$
0 0 00 00 1 00 10 $\longrightarrow$ 0 0 00 00 0 10 00

4) 0 0 00 00 0 00 00 $\longrightarrow$ 0 0 00 00 1 00 00

5) 0 0 00 00 0 00 00 $\longrightarrow$ 1 0 00 00 0 00 00 $\longrightarrow$
1 0 00 00 0 00 01 $\longrightarrow$ 0 0 00 01 0 00 00

6) 0 0 00 00 0 00 00 $\longrightarrow$ 1 0 00 00 0 00 00 $\longrightarrow$
1 0 00 00 0 00 10 $\longrightarrow$ 0 0 00 10 0 00 00

Routing Path Example

7) 0 0 00 00 0 00 00 $\longrightarrow$ 1 0 00 00 0 00 00 $\longrightarrow$
1 0 00 00 1 00 00 $\longrightarrow$ 1 0 00 00 1 00 01 $\longrightarrow$
1 0 00 00 0 01 00 $\longrightarrow$ 0 0 01 00 0 00 00

8) 0 0 00 00 0 00 00 $\longrightarrow$ 1 0 00 00 0 00 00 $\longrightarrow$
1 0 00 00 1 00 00 $\longrightarrow$ 1 0 00 00 1 00 10 $\longrightarrow$
1 0 00 00 0 10 00 $\longrightarrow$ 0 0 10 00 0 00 00

9) 0 0 00 00 0 00 00 $\longrightarrow$ 1 0 00 00 0 00 00 $\longrightarrow$
1 0 00 00 1 00 00 $\longrightarrow$ 0 1 00 00 0 00 00

10) 0 0 00 00 0 00 00 $\longrightarrow$ 1 0 00 00 0 00 00

Theorem 1

In the bidirectional channel and 1-port communication model, the communication between nodes u and $u^{(i)}$ in $RDN^k(Q_m)$, where the addresses of u and $u^{(i)}$ differ in ith bit position for $0 \leq i < 2^k m + 2^k - 1$ takes at most

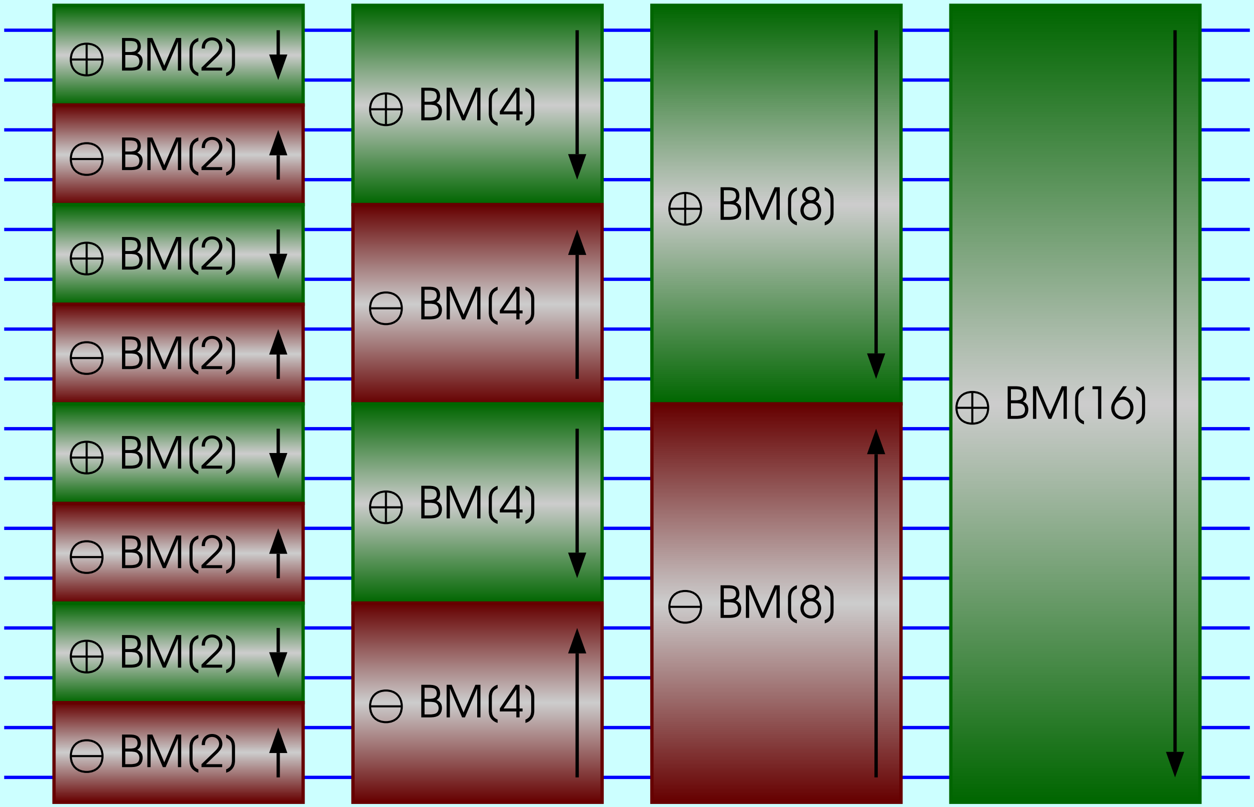
$$t_k = 2k + 1$$

steps.

Bitonic Sorting on Hypercube

- Bitonic sort is based on repeatedly merging two bitonic sequences to form a larger bitonic sequence.
- A bitonic sequence is a sequence of values $(a_0, a_1, \ldots, a_{n-1})$ with the property that either
  1. (1) there exists an index i, where $0 \leq i \leq n-1$, such that $(a_0 \ldots, a_i)$ is monotonically increasing and $(a_{i+1}, \ldots, a_{n-1})$ is monotonically decreasing, or
  2. (2) there exists a cyclic shift of indices so that (1) is satisfied.
- For example, $(2, 3, 8, 13, 15, 14, 7, 0)$ is a bitonic sequence because it first increases and then decreases.

Bitonic Sorting Network of Size 16

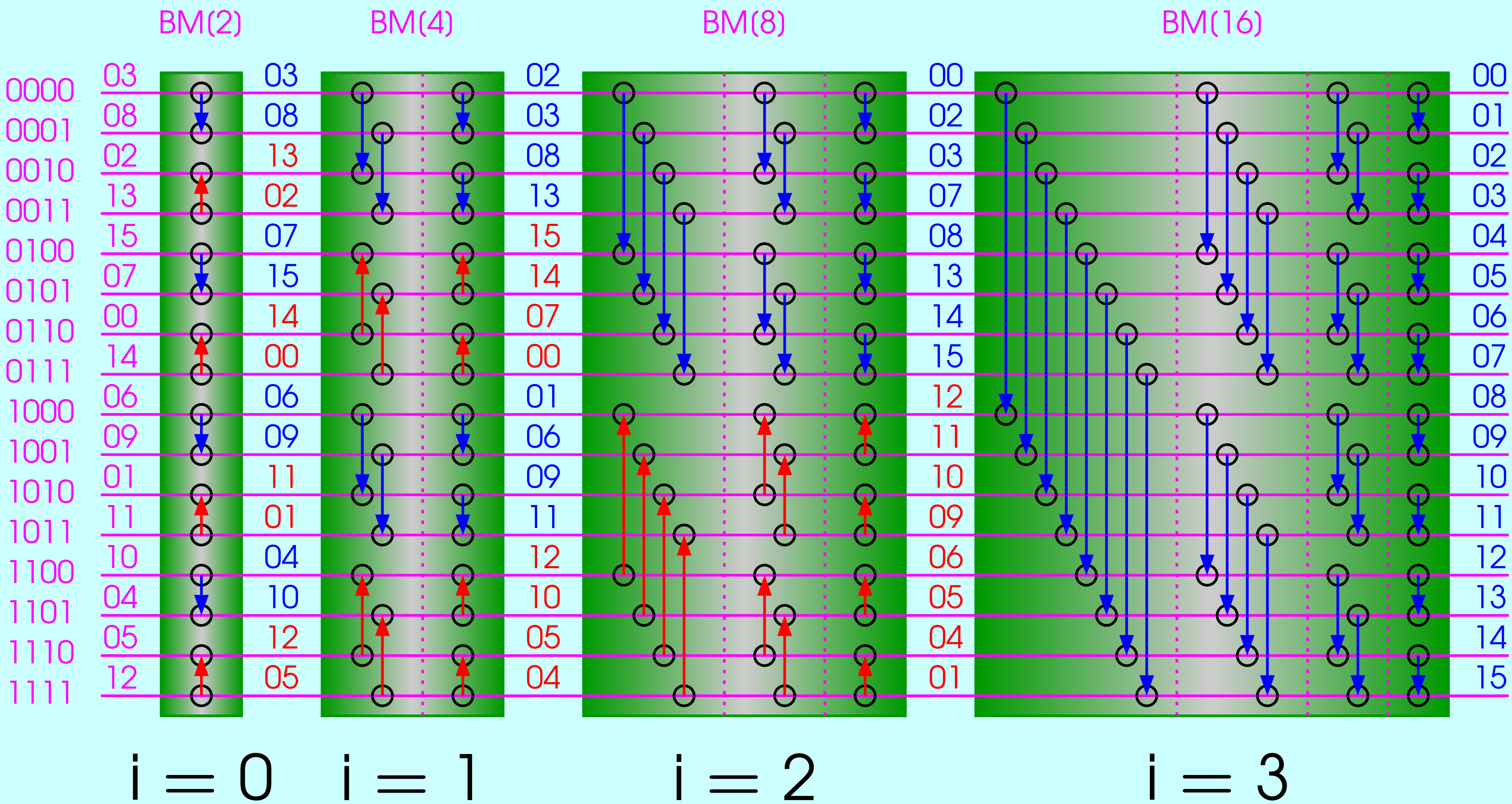


- ⊕BM(k): increasing bitonic merging networks of size k;
- ⊖BM(k): decreasing bitonic merging networks of size k;
- The last merging network (⊕BM(16)) sorts the input.

Bitonic Sorting Algorithm on n-Cube

```
Bitonic_sort_hypercube (my_id, my_number, n, result)
begin
    result ← my_number;
    for i ← 0 to n − 1 do
        for j ← i downto 0 do
            partner ← my_id XOR 2^j;
            send result to partner;
            receive number from partner;
            if (my_id AND 2^(i+1) ≠ my_id AND 2^j)                /* max */
                if (number > result)
                    result ← number;
                endif
            else                                                  /* min */
                if (number < result)
                    result ← number;
                endif
            endif
        endfor
    endfor
end
```

Bitonic Sorting on 4-Cube



Bitonic Sorting Algorithm on $RDN^2(Q(m))$

Algorithm 1 Bitonic_Sort_RDN (*my_id, my_number, m, result*)

begin

$result \leftarrow my\_number$;

for $i \leftarrow 0$ **to** $4m + 2$ **do**

for $j \leftarrow i$ **downto** 0 **do**

if $m - 1 \geq j \geq 0$ **then** $t = j$; *path* $=$

$(c_2, c_1, b^3_{m-1} \cdots b^3_0, b^2_{m-1} \cdots b^2_0, c_0, b^1_{m-1} \cdots b^1_0, b^0_{m-1} \cdots b^0_t \cdots b^0_0) \longrightarrow$

$(c_2, c_1, b^3_{m-1} \cdots b^3_0, b^2_{m-1} \cdots b^2_0, c_0, b^1_{m-1} \cdots b^1_0, b^0_{m-1} \cdots \overline{b^0_t} \cdots b^0_0)$;

else if $2m - 1 \geq j \geq m$ **then** $t = j - m$; *path* $=$

$(c_2, c_1, b^3_{m-1} \cdots b^3_0, b^2_{m-1} \cdots b^2_0, c_0, b^1_{m-1} \cdots b^1_t \cdots b^1_0, b^0_{m-1} \cdots b^0_0) \longrightarrow$

$(c_2, c_1, b^3_{m-1} \cdots b^3_0, b^2_{m-1} \cdots b^2_0, \overline{c_0}, b^0_{m-1} \cdots b^0_0, b^1_{m-1} \cdots b^1_t \cdots b^1_0) \longrightarrow$

$(c_2, c_1, b^3_{m-1} \cdots b^3_0, b^2_{m-1} \cdots b^2_0, \overline{c_0}, b^0_{m-1} \cdots b^0_0, b^1_{m-1} \cdots \overline{b^1_t} \cdots b^1_0) \longrightarrow$

$(c_2, c_1, b^3_{m-1} \cdots b^3_0, b^2_{m-1} \cdots b^2_0, c_0, b^1_{m-1} \cdots \overline{b^1_t} \cdots b^1_0, b^0_{m-1} \cdots b^0_0)$;

else if $j = 2m$ **then** *path* $=$

$(c_2, c_1, b^3_{m-1} \cdots b^3_0, b^2_{m-1} \cdots b^2_0, c_0, b^1_{m-1} \cdots b^1_0, b^0_{m-1} \cdots b^0_0) \longrightarrow$

$(c_2, c_1, b^3_{m-1} \cdots b^3_0, b^2_{m-1} \cdots b^2_0, \overline{c_0}, b^1_{m-1} \cdots b^1_0, b^0_{m-1} \cdots b^0_0)$;

Bitonic Sorting Algorithm on $RDN^2(Q(m))$

else if $3m \geq j \geq 2m+1$ then $t = j-(2m+1)$; *path* =

$(c_2, c_1, b^3_{m-1} \cdots b^3_0, b^2_{m-1} \cdots b^2_t \cdots b^2_0, c_0, b^1_{m-1} \cdots b^1_0, b^0_{m-1} \cdots b^0_0) \longrightarrow$
$(\overline{c_2}, c_0, b^1_{m-1} \cdots b^1_0, b^0_{m-1} \cdots b^0_0, c_1, b^3_{m-1} \cdots b^3_0, b^2_{m-1} \cdots b^2_t \cdots b^2_0) \longrightarrow$
$(\overline{c_2}, c_0, b^1_{m-1} \cdots b^1_0, b^0_{m-1} \cdots b^0_0, c_1, b^3_{m-1} \cdots b^3_0, b^2_{m-1} \cdots \overline{b^2_t} \cdots b^2_0) \longrightarrow$
$(c_2, c_1, b^3_{m-1} \cdots b^3_0, b^2_{m-1} \cdots \overline{b^2_t} \cdots b^2_0, c_0, b^1_{m-1} \cdots b^1_0, b^0_{m-1} \cdots b^0_0)$;

else if $4m \geq j \geq 3m+1$ then $t = j-(3m+1)$; *path* =

$(c_2, c_1, b^3_{m-1} \cdots b^2_t \cdots b^3_0, b^2_{m-1} \cdots b^2_0, c_0, b^1_{m-1} \cdots b^1_0, b^0_{m-1} \cdots b^0_0) \longrightarrow$
$(\overline{c_2}, c_0, b^1_{m-1} \cdots b^1_0, b^0_{m-1} \cdots b^0_0, c_1, b^3_{m-1} \cdots b^2_t \cdots b^3_0, b^2_{m-1} \cdots b^2_0) \longrightarrow$
$(\overline{c_2}, c_0, b^1_{m-1} \cdots b^1_0, b^0_{m-1} \cdots b^0_0, \overline{c_1}, b^2_{m-1} \cdots b^2_0, b^3_{m-1} \cdots b^2_t \cdots b^3_0) \longrightarrow$
$(\overline{c_2}, c_0, b^1_{m-1} \cdots b^1_0, b^0_{m-1} \cdots b^0_0, \overline{c_1}, b^2_{m-1} \cdots b^2_0, b^3_{m-1} \cdots \overline{b^2_t} \cdots b^3_0) \longrightarrow$
$(\overline{c_2}, c_0, b^1_{m-1} \cdots b^1_0, b^0_{m-1} \cdots b^0_0, c_1, b^3_{m-1} \cdots \overline{b^2_t} \cdots b^3_0, b^2_{m-1} \cdots b^2_0) \longrightarrow$
$(c_2, c_1, b^3_{m-1} \cdots \overline{b^2_t} \cdots b^3_0, b^2_{m-1} \cdots b^2_0, c_0, b^1_{m-1} \cdots b^1_0, b^0_{m-1} \cdots b^0_0)$;

else if $j = 4m+1$ then *path* =

$(c_2, c_1, b^3_{m-1} \cdots b^3_0, b^2_{m-1} \cdots b^2_0, c_0, b^1_{m-1} \cdots b^1_0, b^0_{m-1} \cdots b^0_0) \longrightarrow$
$(c_2, \overline{c_1}, b^3_{m-1} \cdots b^3_0, b^2_{m-1} \cdots b^2_0, c_0, b^1_{m-1} \cdots b^1_0, b^0_{m-1} \cdots b^0_0)$;

Bitonic Sorting Algorithm on $RDN^2(Q(m))$

```
        else if j = 4m + 2 then path =
            (c_2, c_1, b^3_{m-1} ··· b^3_0, b^2_{m-1} ··· b^2_0, c_0, b^1_{m-1} ··· b^1_0, b^0_{m-1} ··· b^0_0) →
            (~c_2, c_1, b^3_{m-1} ··· b^3_0, b^2_{m-1} ··· b^2_0, c_0, b^1_{m-1} ··· b^1_0, b^0_{m-1} ··· b^0_0);
        endif
        send result along with path;
        receive number along with path;
        if (my_id AND 2^{i+1} ≠ my_id AND 2^j)                         /* max */
            if (number > result)
                result ← number;
            endif
        else                                                          /* min */
            if (number < result)
                result ← number;
            endif
        endif
    endfor
  endfor
end
```

In the path expressions: $(c_2, c_1, b^3_{m-1} \cdots b^3_0, b^2_{m-1} \cdots b^2_0, c_0, b^1_{m-1} \cdots b^1_0, b^0_{m-1} \cdots b^0_0) \rightarrow (\overline{c_2}, c_1, b^3_{m-1} \cdots b^3_0, b^2_{m-1} \cdots b^2_0, c_0, b^1_{m-1} \cdots b^1_0, b^0_{m-1} \cdots b^0_0)$; condition: $my\_id \text{ AND } 2^{i+1} \neq my\_id \text{ AND } 2^j$

Theorem 2

In the bidirectional channel and 1-port communication model, bitonic sorting on an $RDN^k(Q_m)$ with $N = 2^{2^k m + 2^k - 1}$ nodes can be done in

$$O((m2^k)^2)$$

computation steps and

$$O((mk2^k)^2)$$

communication steps, respectively.

Conclusions

- The recursive dual-net, RDN, has many attractive properties including
  - Small and flexible node-degree
  - Short diameter
  - Recursive structure
  - Efficient routing and broadcasting algorithms
- We showed that a parallel sorting algorithm can be implemented efficiently in RDN
- The RDN can be used as an efficient interconnection network of a supercomputer of the future generation

THX