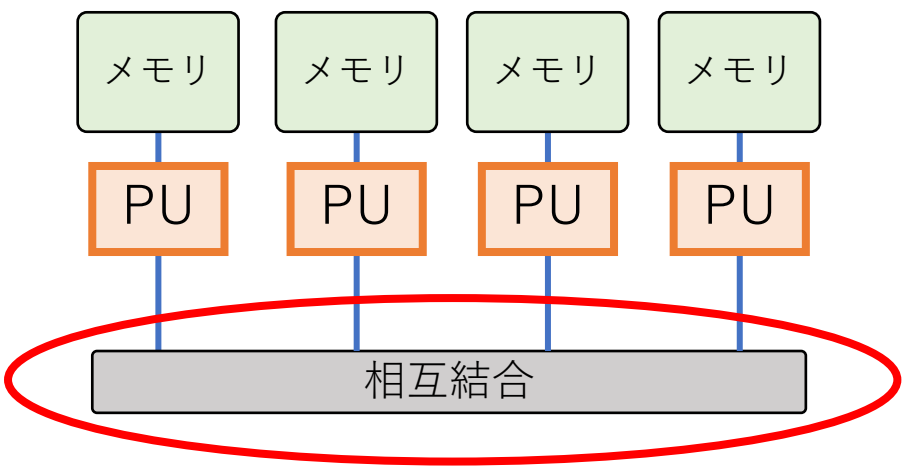
並列分散コンピューティング
(1)相互結合ネットワーク
モデルと経路設定

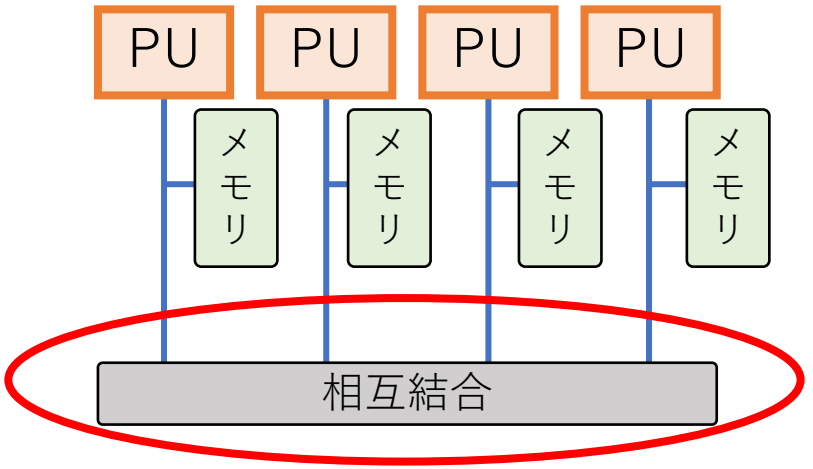
大瀧保広

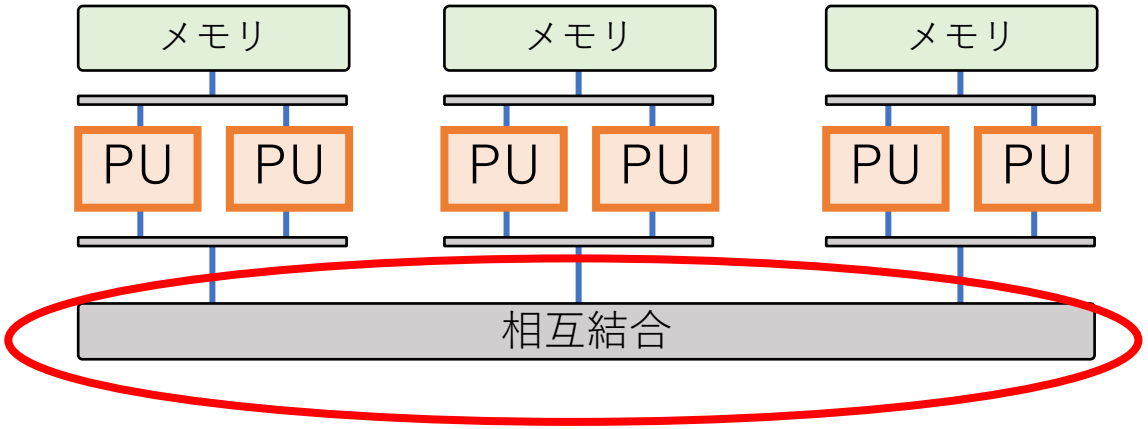
今日の内容

- 相互結合ネットワークモデルと経路設定
  - 完全結合ネットワーク
  - 線型結合ネットワーク
  - 格子結合ネットワーク
  - トーラス結合ネットワーク
  - ２分木ネットワーク
  - 超立方体ネットワーク
  - バタフライネットワーク

様々な並列分散アーキテクチャ







相互結合ネットワーク

複数の演算ユニット（PU)を結合するためのネットワーク

■分散システムでは
通常の一般的なネットワークを使って構築することが多い。

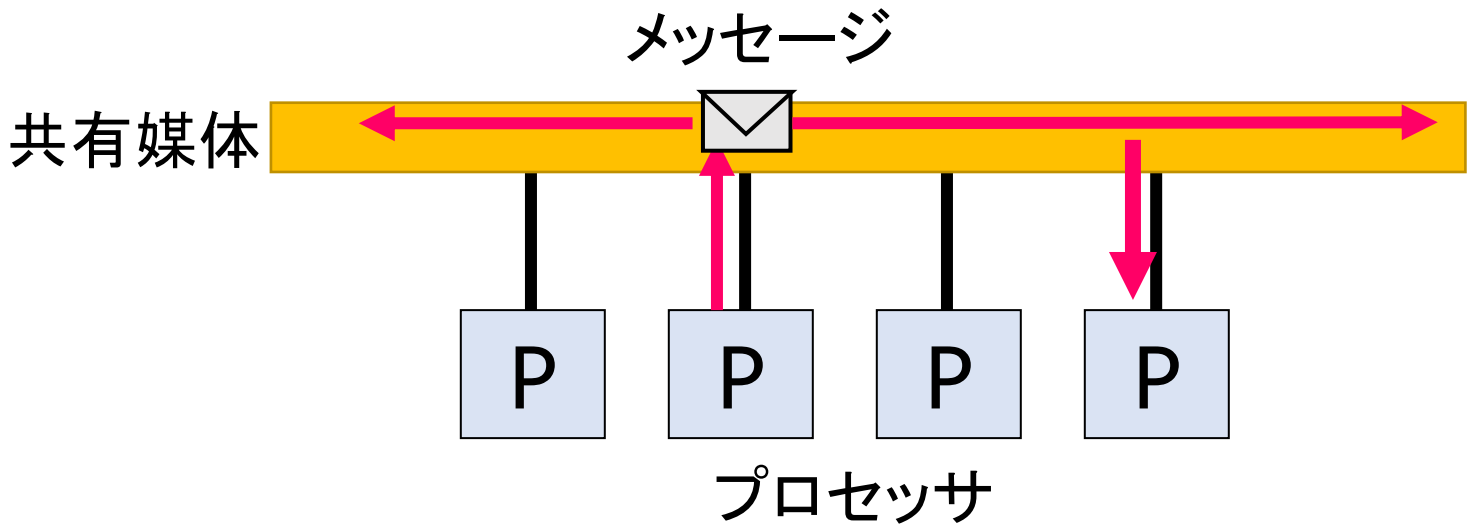
■並列処理を行う(専用)システム
通信の効率向上のために、プロセッサ同士を密に結合する専用配線を使うことが多い。

■相互結合ネットワークの物理的な構成
- ■共有媒体（バス型ネットワーク）
- ■スイッチング装置

物理的構成(1)：共有媒体による結合

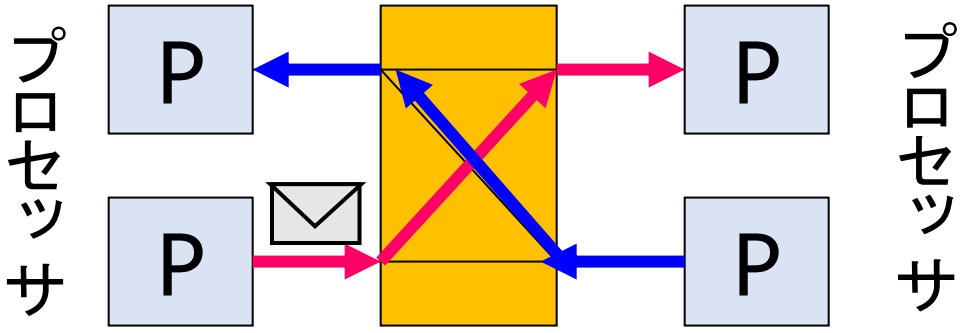
例　Ethernet

- 宛先情報を付加したメッセージに媒体上に送出する。
- 各プロセッサはメッセージが自分宛かどうかを調べ、
  自分宛ならば取り込む。
- 媒体上には、ある瞬間には最大1メッセージしか存在できない。



物理的構成(2)：スイッチング装置

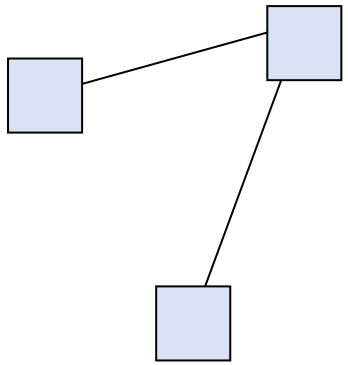
- 各プロセッサはスイッチング装置と接続する。
- プロセッサ対の間で固有の通信路を形成し、1対1のメッセージ通信を行う。
- ぶつからなければ複数の通信路を同時に形成できる。（スイッチング装置の性能による）



相互結合ネットワークモデル

- （プロセッサの）相互結合方法をモデル化した表現。
- 物理的な構成を抽象化し、
  論理的な接続関係（論理ネットワーク）のみで考えることができる。

- 相互結合ネットワークをグラフ $G = (V, E)$ で表現
  - 節点（ノード） $v_i \in V$ ： プロセッサ（またはスイッチ）
  - 辺（枝） $e_i \in E$ ： 通信経路



相互結合ネットワークモデルの例

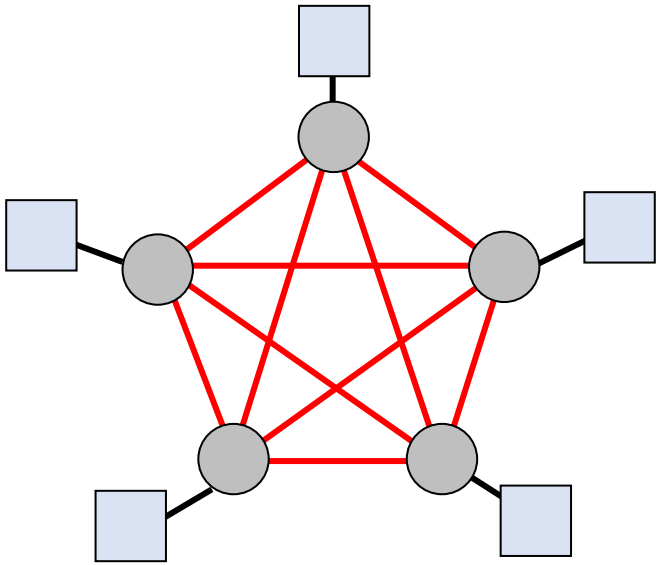
- 完全結合ネットワーク
- 線型結合ネットワーク
- 格子結合ネットワーク
- トーラス結合ネットワーク
- ２分木ネットワーク
- 超立方体ネットワーク
- バタフライネットワーク

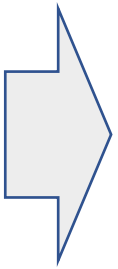
など

完全結合ネットワーク

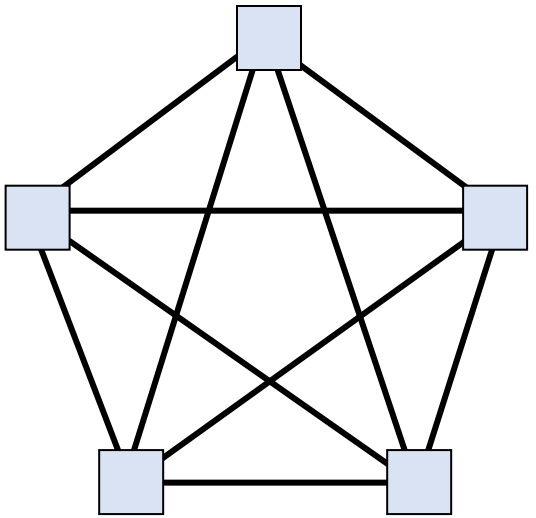
■全プロセッサをスイッチで相互接続

例　ノード数　N=5





簡略化して
下図のように表す。



スイッチ　　プロセッサ

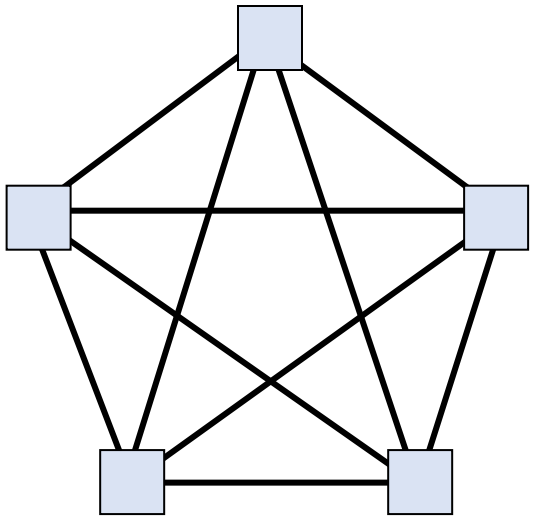
完全結合ネットワーク

- メリット：1ステップで任意のノードと通信できる。
- デメリット：実際に実現するには大量の配線が必要。

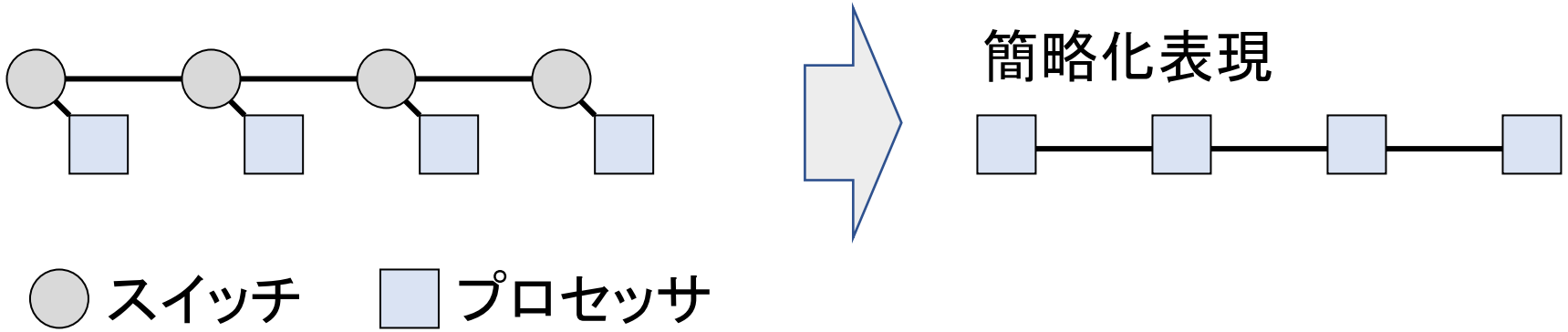
- Nノード完全結合の辺の数

$$ {}_N\mathrm{C}_2 = \frac{N(N-1)}{2} $$

$$ O(N^2) $$



線形結合ネットワーク



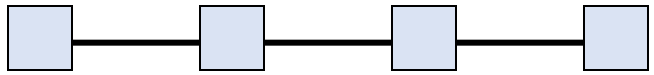
- 配線数が最小の単純なモデル
- メリット：実現するための配線が最小 $N-1$ （であるモデルのひとつ）
  配線数はノード数に比例したオーダー $O(N)$
- デメリット：ノード間の通信時間のばらつきが大きい
  - 最小　？
  - 最大　？
  - 平均　？

=通信経路の長さ

線形結合

■平均
　ノード数Nに比例したオーダーの通信時間

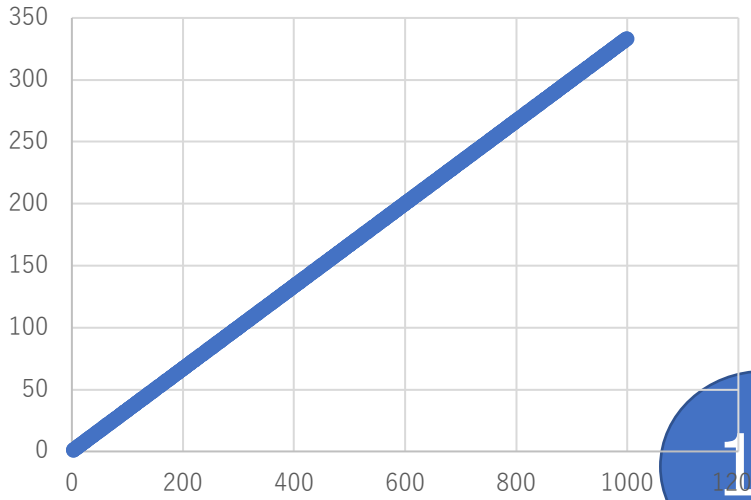
N=4のとき
(1+1+1 + 2+2 + 3)/6 = 10/6



N=10のとき
(1x9+2x8+3x7+ … + 9x1) / 45
= 3.666..

N=100 → 33.66

N=1000 → 333.33



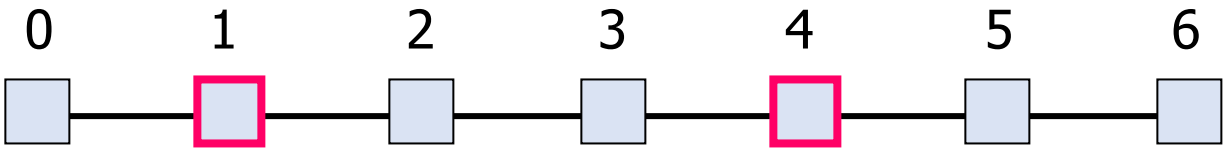
経路設定（ルーティング）

■並列分散システムでは、
演算プロセッサ同士がデータをやり取り
しながら処理を行う。

■完全結合ネットワークならば
すべてのプロセッサ間に直通の通信路がある。
→直接相手に渡せば良い

■それ以外の相互結合ネットワークでは、
指定した相手にデータを渡す経路を見つける必要がある。
  ■具体的には、「中継ノードにおいて、次に誰に渡せば良いか をどう決定するのか」ということ。

線形結合ネットワークの経路設定

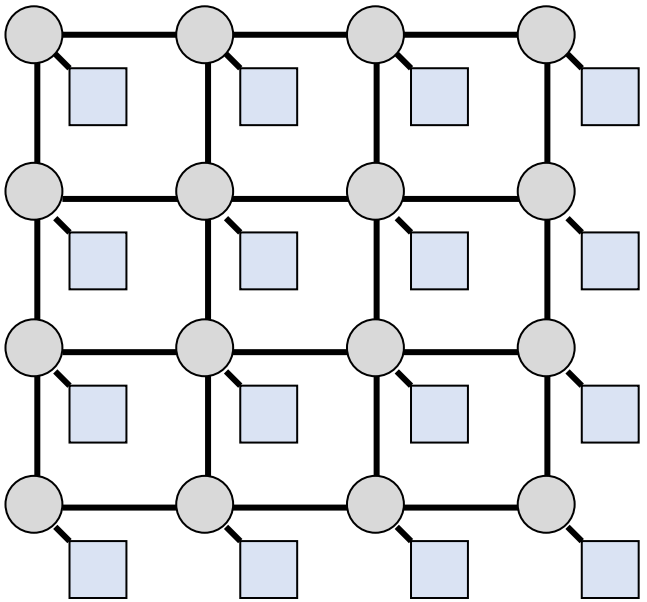


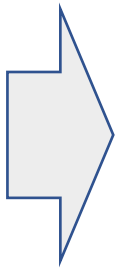
プロセッサ4からプロセッサ1にデータを送りたい。

- 「宛先：1」のデータを隣に渡す。
- 渡す相手：（各プロセッサにおいて）
  自分の番号 i と宛先を比べて、
  宛先が小さければ　プロセッサ i-1 に、
  宛先が大きければ　プロセッサ i+1 に渡す。

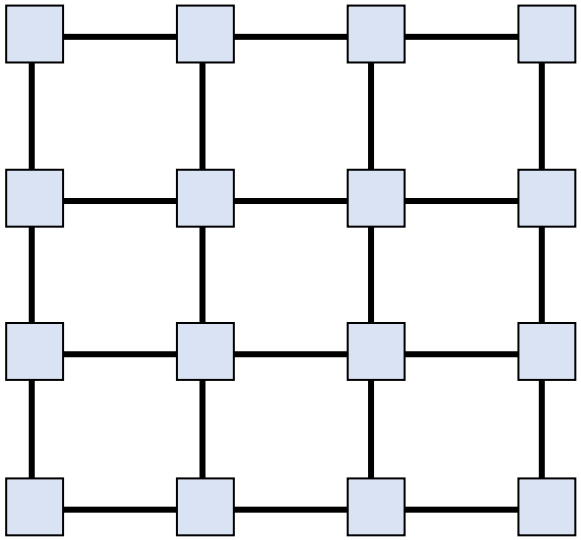
格子結合（2次元メッシュ）

- 最も典型的な結合モデル
  ↑ 2次元配列で表現されたデータと相性がいい
- 応用分野：画像処理、2次元数値計算など





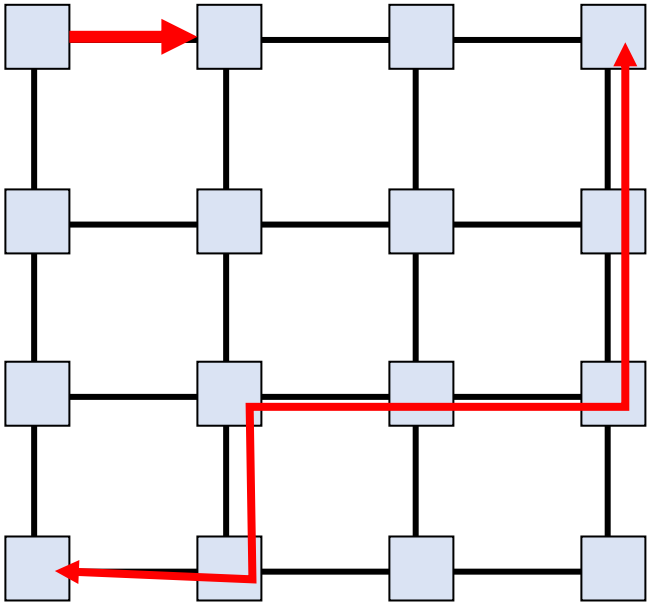
簡略化表現



○ スイッチ　□ プロセッサ

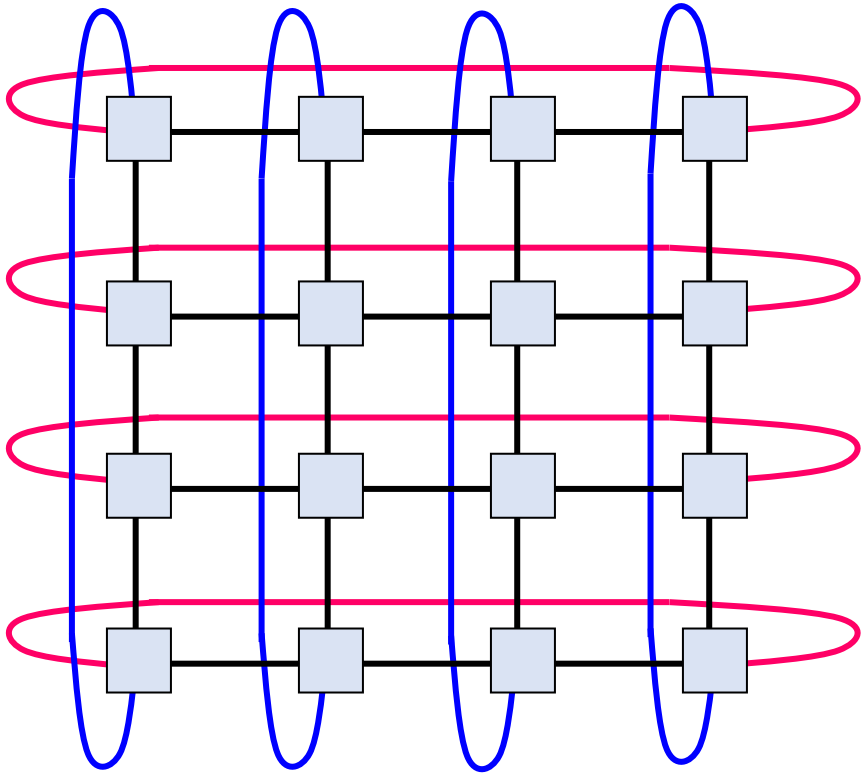
格子結合

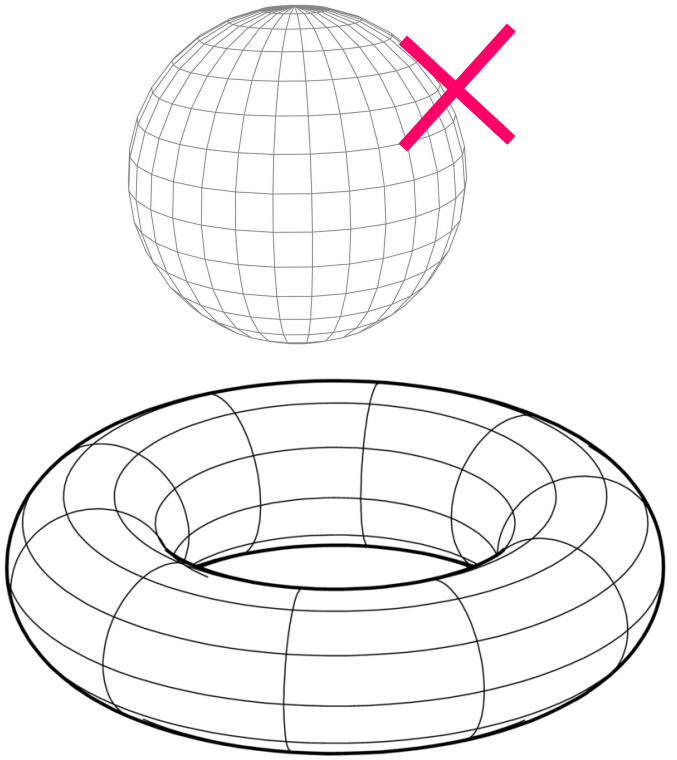
- 特徴
  - 配線：ノード数に比例するオーダー
  - 任意のノード間の通信経路の長さ
    最小　1
    最大



トーラス結合

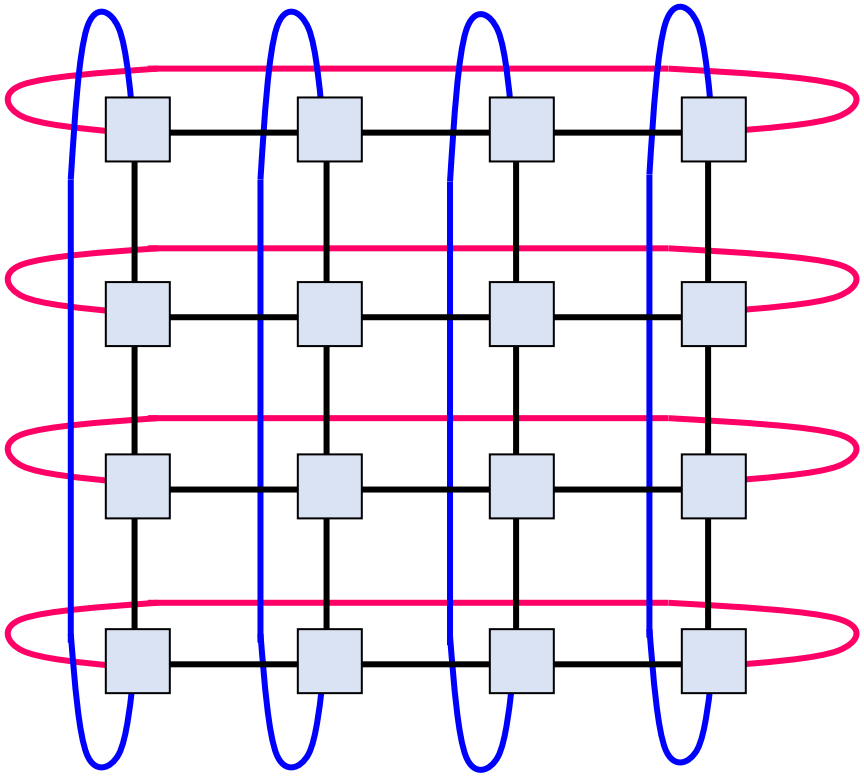
■格子結合の変形モデル
　格子結合の上下端および左右端を接続したもの





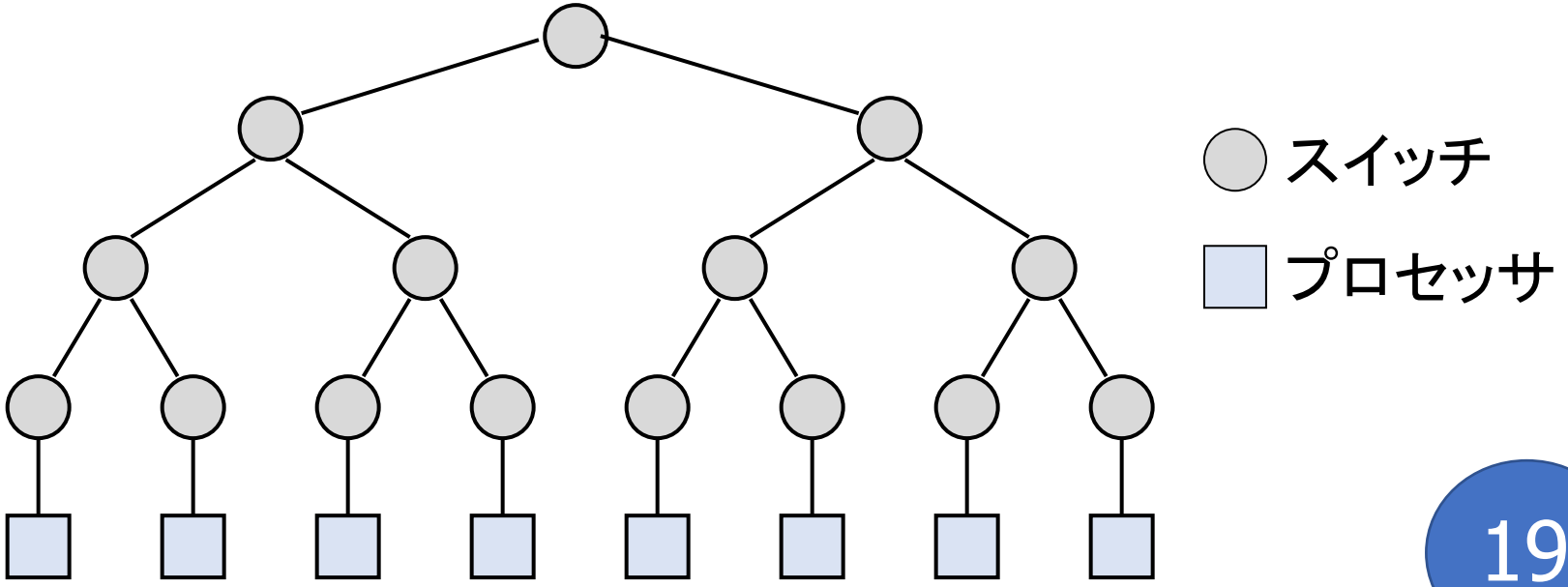
トーラス結合

- 特徴
  - 配線：ノード数に比例
  - 任意のノードとの通信
    最小 1
    最大



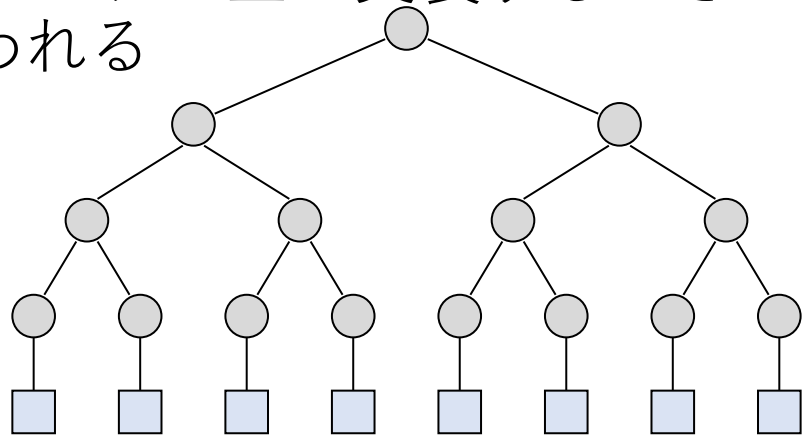
2分木ネットワーク

- 特徴
  - 任意のノードとの通信が高速である。
    通信時間 $O(\log(N))$
  - root ノードに通信の混雑が集中する。
  - 内部ノードにもプロセッサを置く構成もある



2分木ネットワーク

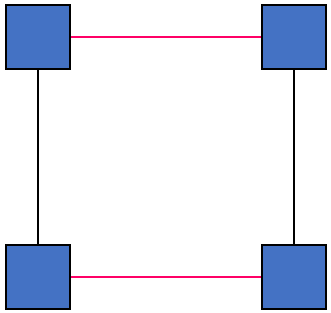
- 2分木は多数のアルゴリズムやデータ構造で使用される。
  - クイックソート、データの探索、分割統治法、グラフィックスの領域処理、他
  - インターネット、分散ネットワーク上でのデータの分散・収集
- これらのアルゴリズムを分散システム上に実装するときの論理ネットワークとしても使われる



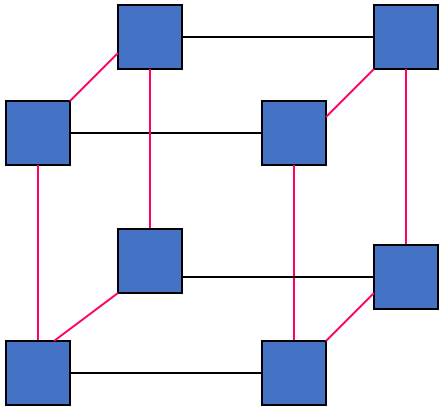
超立方体結合ネットワーク　HyperCube

1次元Cube

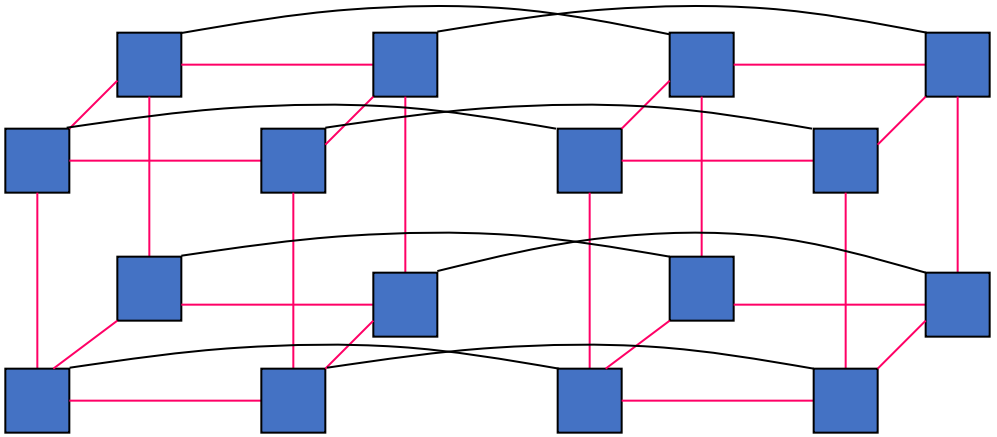
2次元Cube



3次元Cube



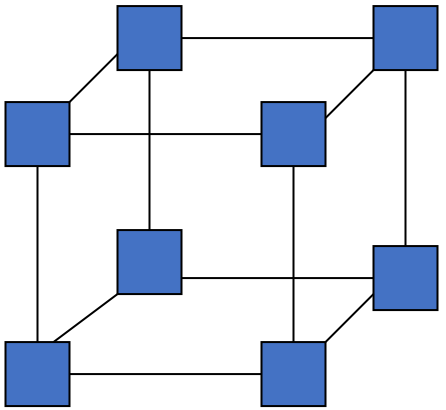
4次元Cube



・・・　*n*次元Cube

超立方体結合ネットワーク

- 特徴
  - 任意のノードとの通信が高速
    通信時間：$O(\log(N))$
  - 2分木、バタフライ、シャッフル
    交換などの構造を含んでいる。
  - 実現のためのハードウェア量は大きい。

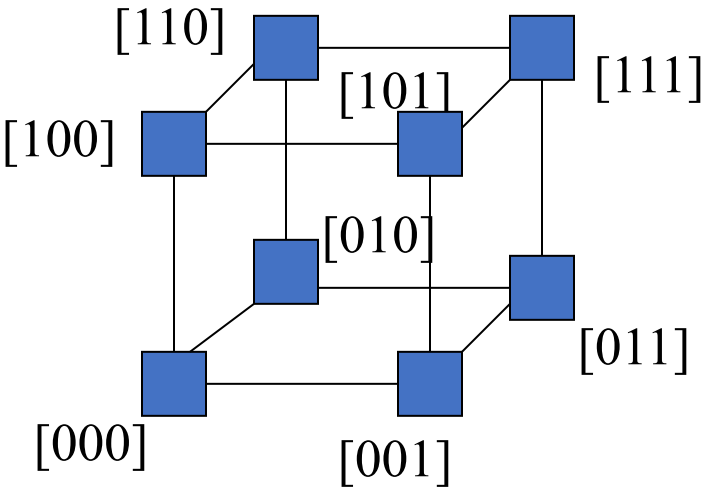


- 2進表現と相性が良いすべての操作に応用できる。
  アルゴリズムとデータ構造の基本

超立方体結合ネットワークの接続方法

- n次元超立方体の
  ノード番号（アドレス）を
  2進表現する。

$$u = [u_{n-1} \cdots u_1 u_0]$$
$$u_i \in \{0, 1\}$$



- ノードの接続の仕方：
  2進表現が1ビットだけ異なる2ノードを接続する。

$$u = [u_{n-1} \cdots u_i \cdots u_1 u_0] \Leftrightarrow u' = [u_{n-1} \cdots \overline{u_i} \cdots u_1 u_0]$$

$\overline{u_i}$:u_i に対する1の補数
(簡単に言えば0と1の反転)

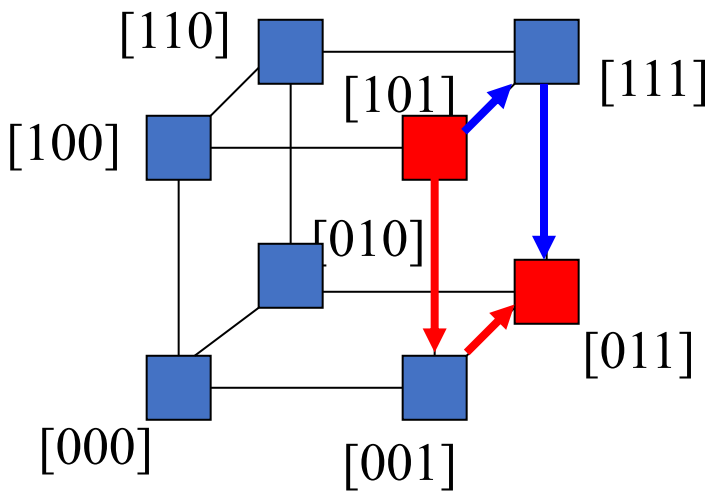
超立方体の経路設定（ルーティング）

例　[101]→[011]の経路：

- 排他的論理和をとると：[101]⊕[011]＝[110]
- アドレスは上位2ビットが異なる。

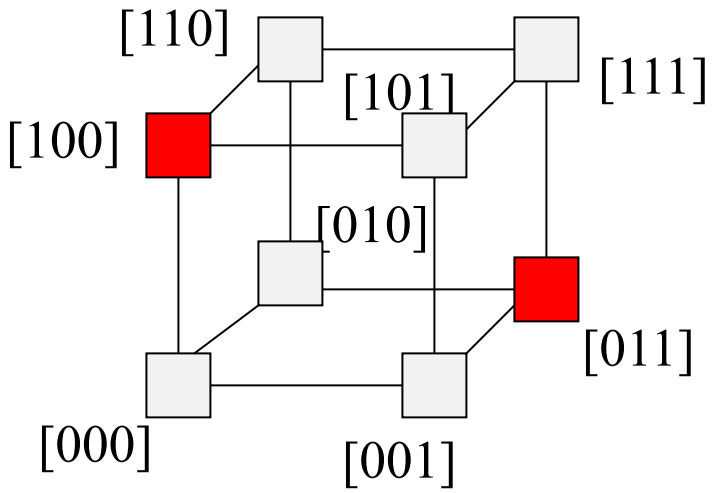
経路1: [101]→[001]→[011]（上位のビットから変更する）

経路2: [101]→[111]→[011]（下位のビットから変更する）



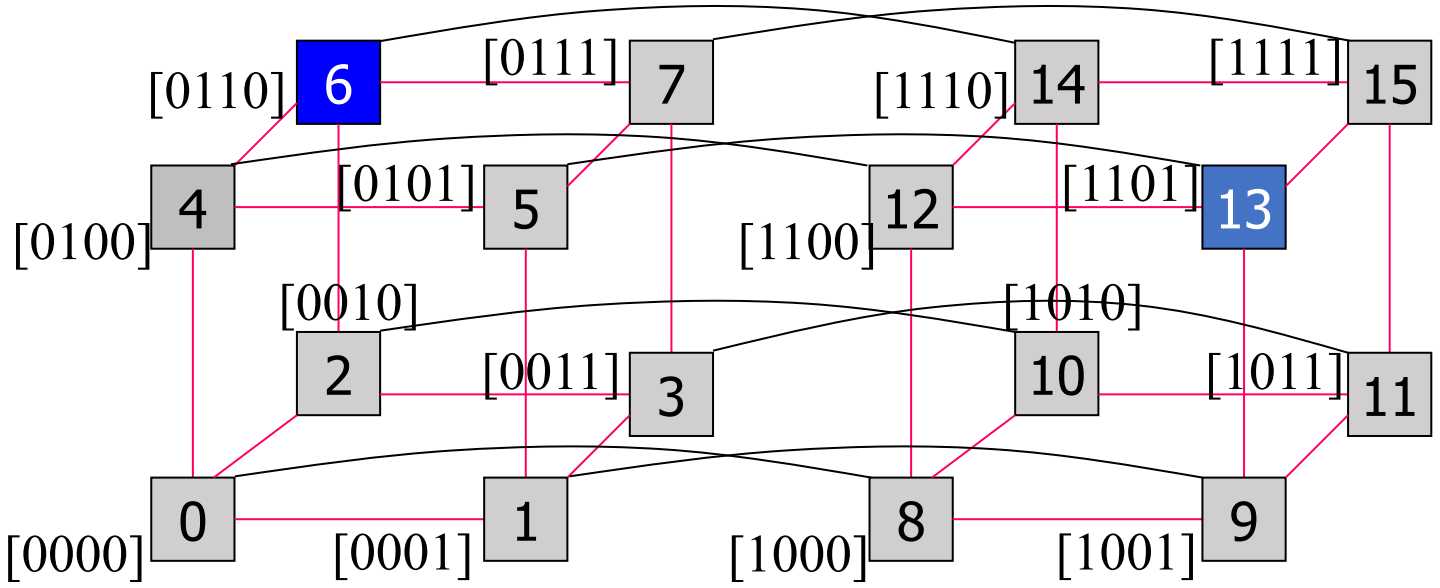
3次元超立方体の経路設定

- ノード4=[100]からノード3=[011]への経路を求めよ。
  - 上位ビットから：
  - 下位ビットから：



4次元超立方体の経路設定

- ノード13=[1101]からノード6=[0110]からへの経路を求めよ
  - 上位ビットから：
  - 下位ビットから：



超立方体ネットワークの拡張(a)

A) k 元 n キューブ

- ノード番号を n次元 k 進表現で表す
- 以下のようにノードを接続する。

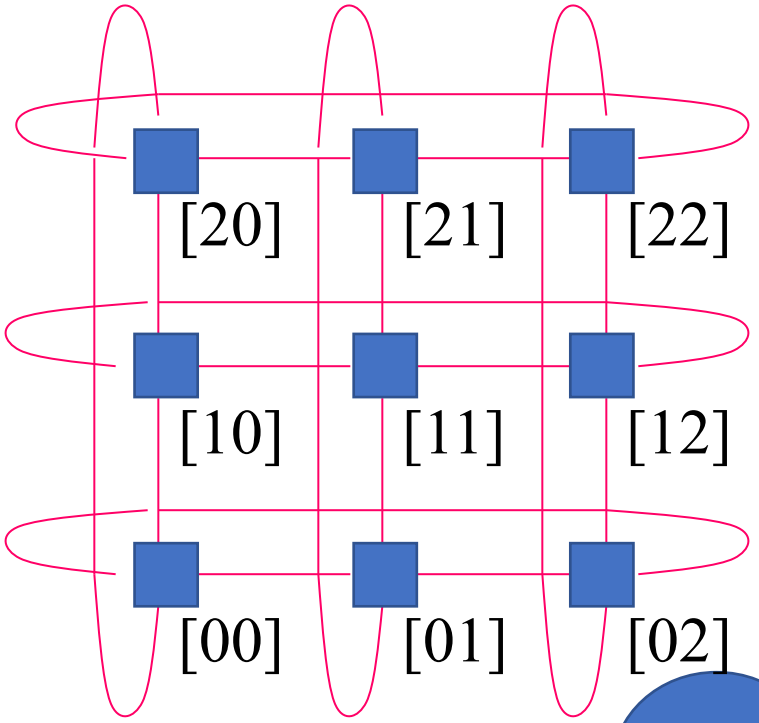
$$u=[u_{n-1}\cdots u_i\cdots u_0],\ u_i\in\{0,1,\cdots,k-1\}$$

↕ 接続

$$u'=[u_{n-1}\cdots((u_i\pm1)\bmod k)\cdots u_0]$$

例　3元2キューブ（n=2, k=3）

$$u=[u_1u_0],\ u_i\in\{0,1,2\}$$



超立方体ネットワークの拡張(b)

B）超直方体

- k元nキューブの各桁の基数b_iが次元ごとに異なるもの。
- 以下のように接続する。

$$u=[u_{n-1}\cdots u_i\cdots u_0],\ u_i\in\{0,1,\cdots,b_i-1\}$$

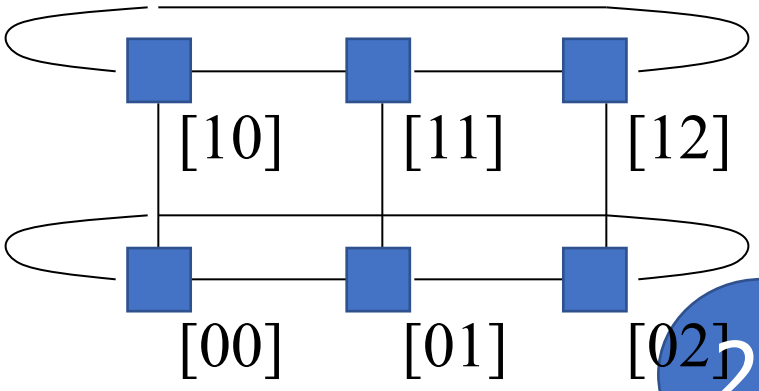
⇕ 接続

$$u'=[u_{n-1}\cdots((u_i\pm1)\bmod b_i)\cdots u_0]$$

例　(2, 3)超直方体

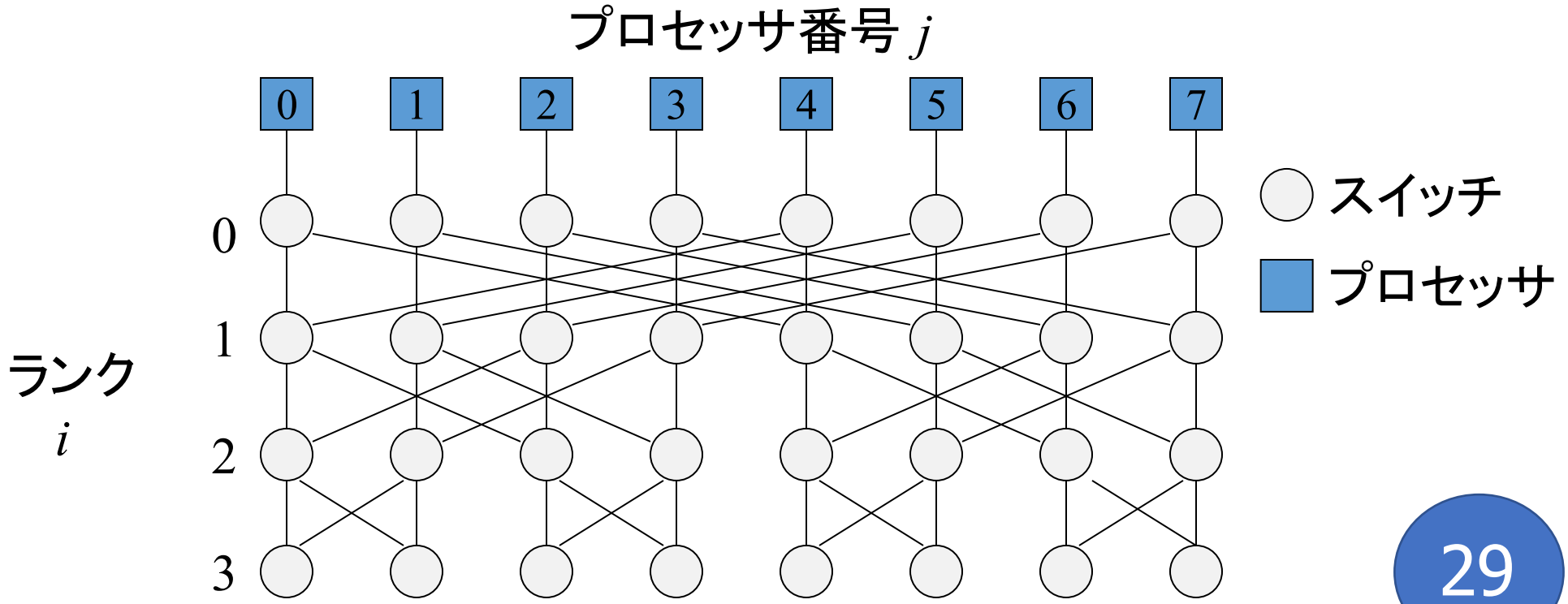
n=2,　$u=[u_1u_0]$

基数 $b_0=3,\ b_1=2$



バタフライネットワーク　（Butterfly：蝶）

- 特徴：高速である。配線のハードウエア量が多い。
- 応用：高速フーリエ変換(FFT)処理など

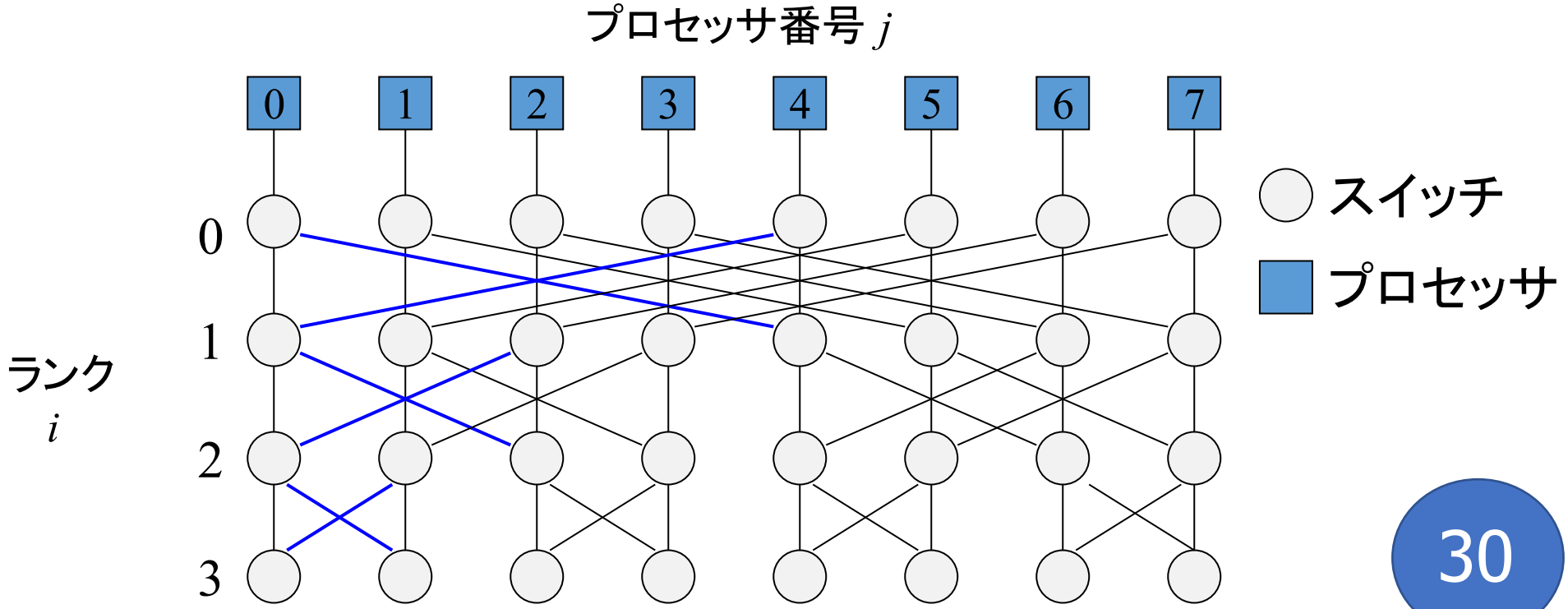


バタフライネットワークの接続方法

■記法：スイッチノード(i, j)：

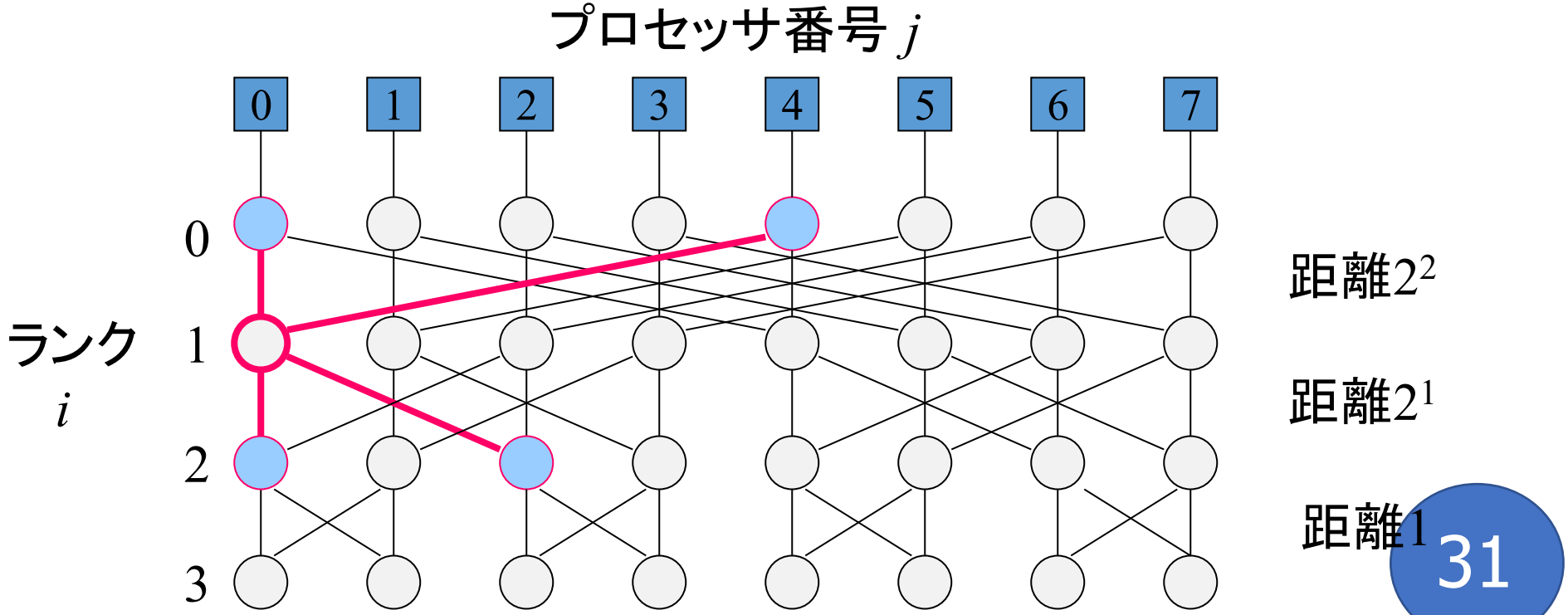
i ：ランク（垂直方向）　$0 \le i \le n$

j ：プロセッサ番号（水平方向）　$0 \le j \le N-1,\ N = 2^n$



バタフライネットワークの接続

- スイッチノードを(i, j)と表す。（i:自然数, j：2進表現）
- 例　n=3, i=1の場合
  $(1, [000]) \Leftrightarrow (0, [000]), (0, [100])$　：第2桁が異なるノードと接続
  $(1, [000]) \Leftrightarrow (2, [000]), (2, [010])$　：第1桁が異なるノードと接続



バタフライネットワークの接続(一般化)

■スイッチノード(i, j)と接続される、
ランク$(i\text{-}1)$とランク$(i\text{+}1)$のスイッチノードは、
次のように表すことができる。

■ランク$(i\text{-}1)$

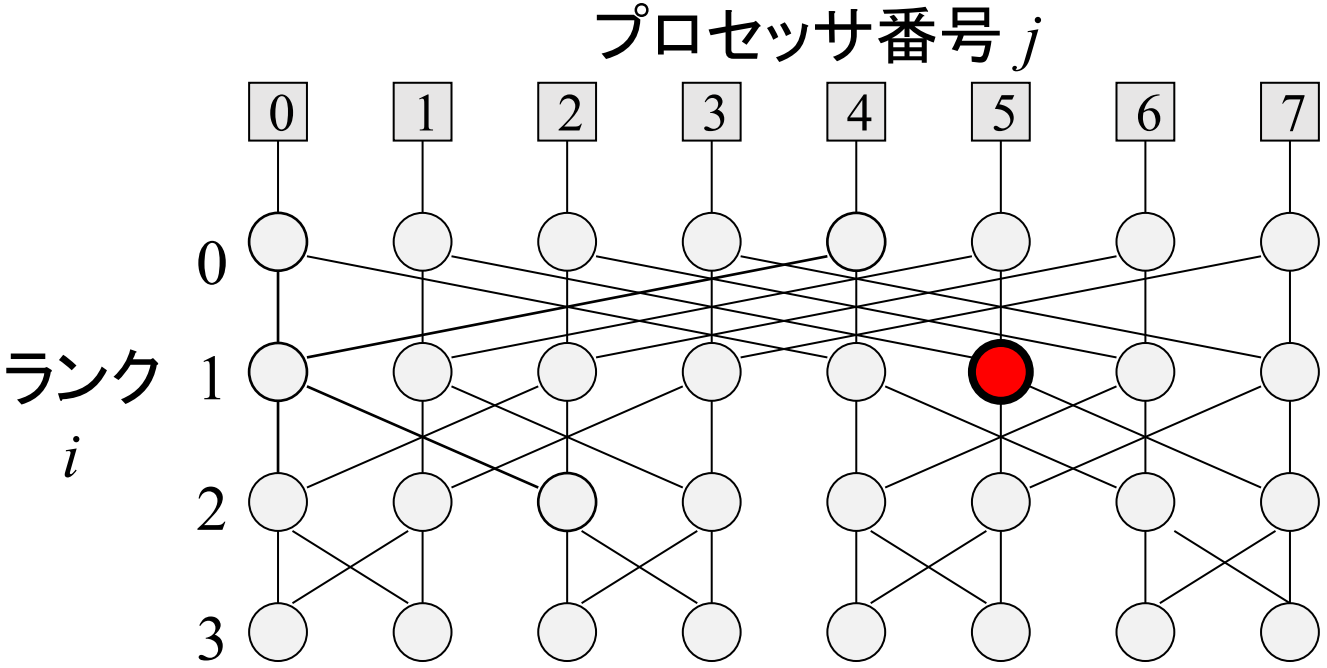
$(i\text{-}1, j)$と $(i\text{-}1, k)$
ただし kはj の2進表現の第$(n\text{-}i)$桁目が異なる値

■ランク$(i\text{+}1)$

$(i\text{+}1, j)$と $(i\text{+}1, m)$
ただし、mはj の2進表現の第$(n\text{-}i\text{-}1)$桁が異なる値

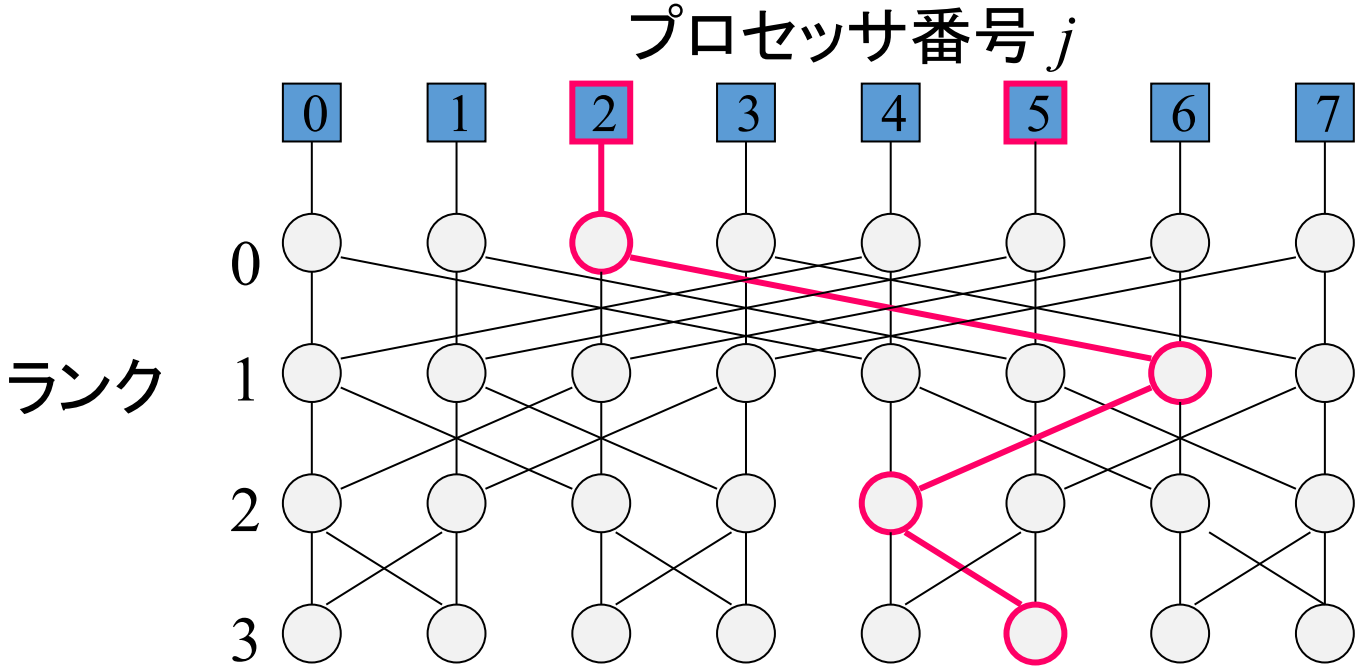
バタフライネットワークの接続

- スイッチノード(1, 5)と接続されるノードを求めよ。



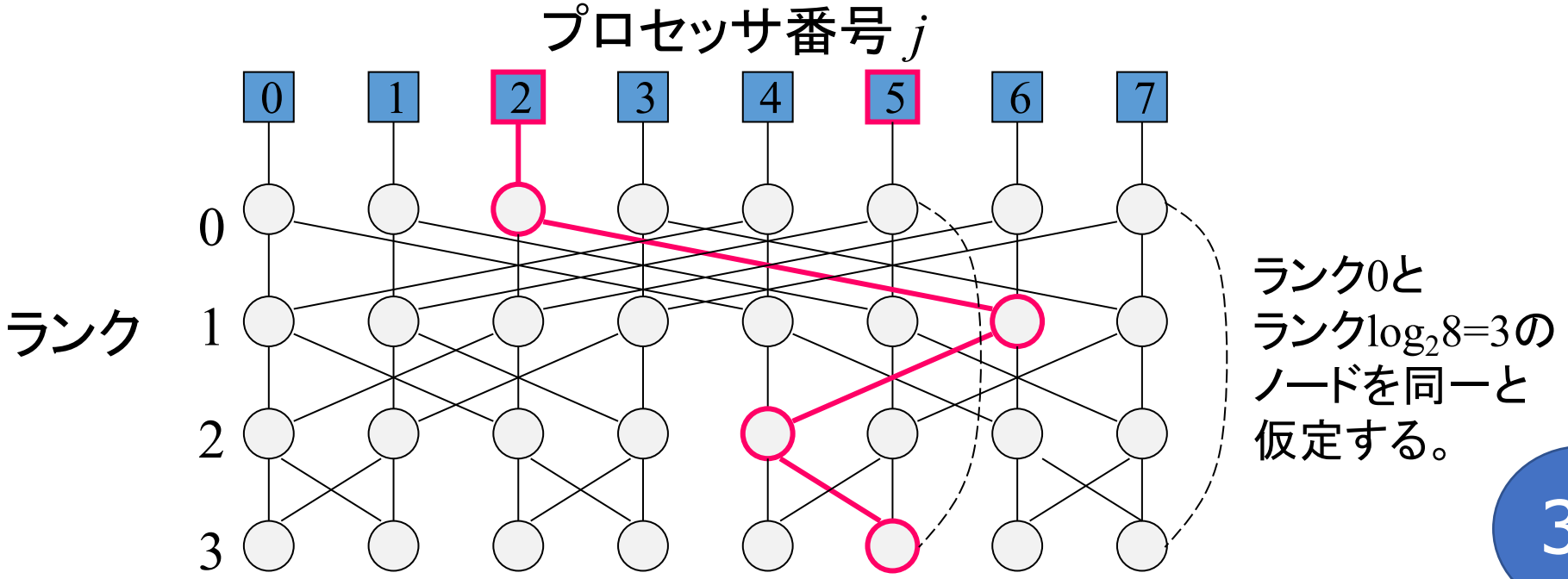
バタフライネットワークの経路設定

■例　プロセッサ2=[010] から プロセッサ5=[101] への経路は次のようにして求まる。
[010]→[110]→[100]→[101]
第2桁目　第1桁目　第0桁目　　∵ [010]⊕[101]=[111]



バタフライネットワークの経路設定

- スイッチノード(3,5)からプロセッサ5への通信
- ランク0とランク($\log_2 N$)-1のスイッチノードが実は同一のものとすれば、プロセッサに接続される。



バタフライネットワークの経路設定

■プロセッサ6=[110]→プロセッサ2=[010]の経路を求めよ。