Recursive Decomposition

A technique used to describe networks.

Analogous to recursion in computer programs.

Need a *scalable network description* to do this.

Scalable Network Description

A *scalable network description* is a one-to-one mapping from the positive integers to network descriptions.

In words: an infinite family of networks, each assigned an integer, starting with one.

Any kind of network description will do, for example, five-tuples and LGMs.

Example, the 2×2-cell omega networks:

$$(n, 2, 2^{n-1}, \mathrm{S}, \mathrm{T})_n.$$

To Describe a Network Using Recursive Decomposition

- Start with a scalable network description.
- Mark certain cells as *recursive*, the rest are *atomic* as follows:

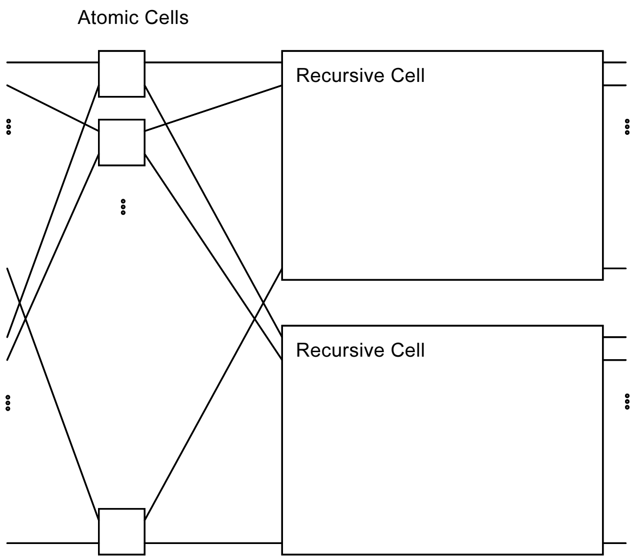
  A cell may (or may not) be marked recursive if the cell has a inputs and b outputs and their exists a scalable network with a inputs and b outputs.

  Otherwise, mark the cell atomic.

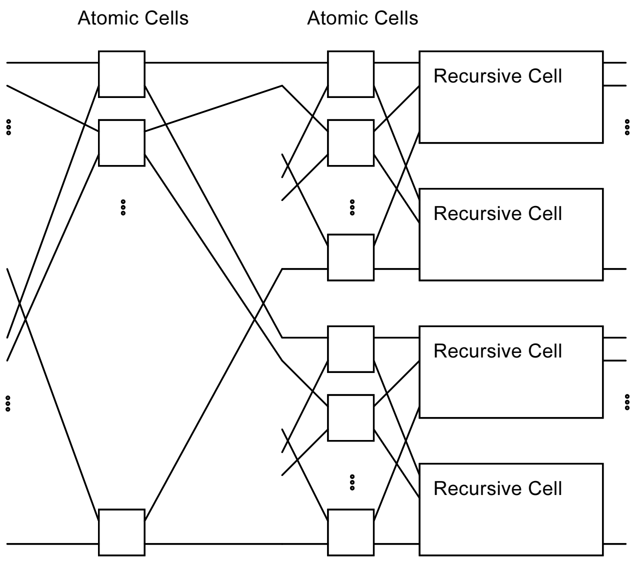
Recursive Decomposition Example, Recursive Omega Redux

Scalable network:

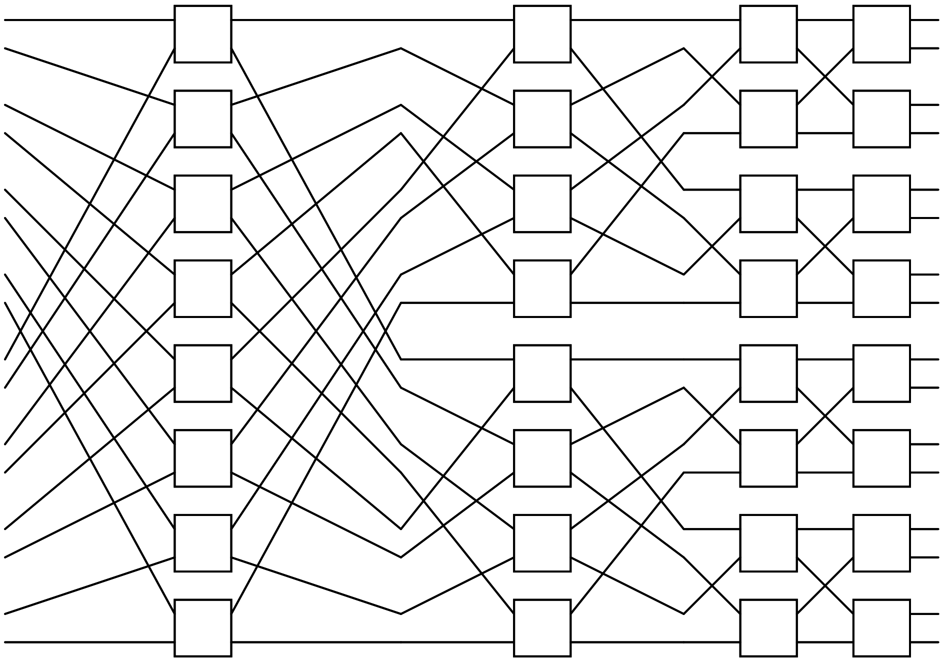
$$\begin{cases} (2,(m,m^{n-1}),(m^{n-1},m),\mathrm{S},\mathrm{T})_n & \text{if } n>1; \\ (1,m,1,\mathrm{T},\mathrm{T}) & \text{if } n=1. \end{cases}$$



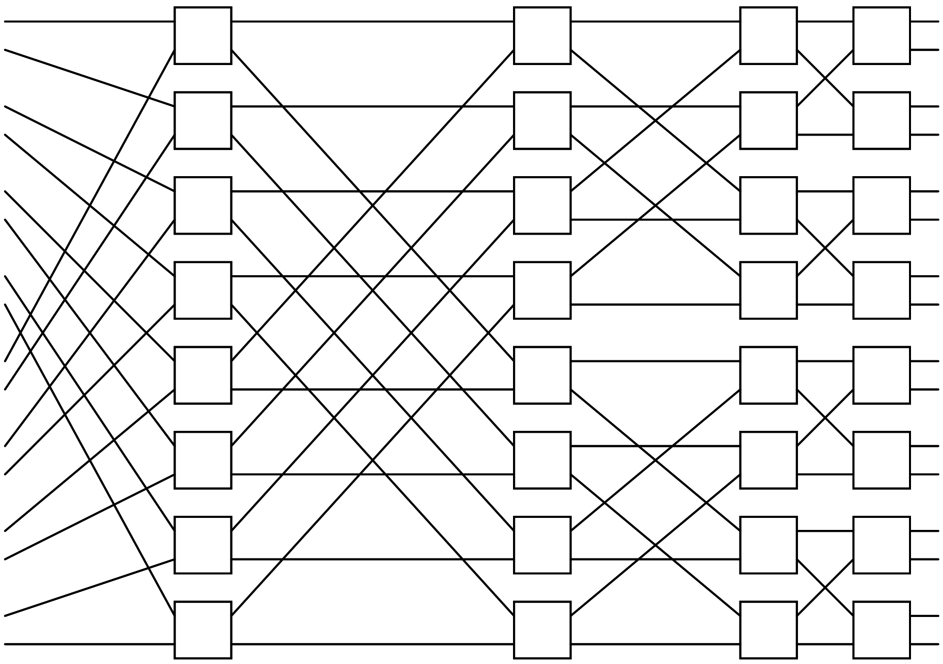
After replacing recursive cells by smaller networks



The completed network of size $n = 4$:



When the link patterns are redrawn, this network appears as the recursive omega presented earlier.



Introducing Butterfly Networks

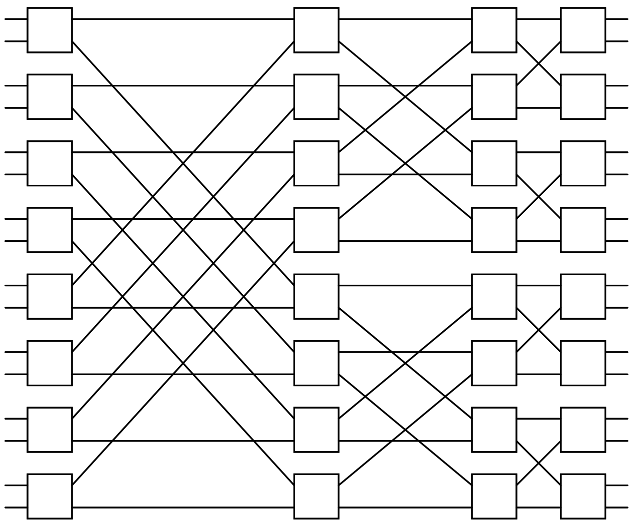
Structure of an n-stage $m \times m$-cell butterfly network:

- Each stage has m^{n-1} cells.
- Network inputs connect to like-numbered stage-0 cell inputs.
- Stage-x cell outputs connect to stage-$(x+1)$ cell inputs with a $n-x-1$ butterfly, for $0 \le x < n-1$.
- Stage-$(n-1)$ cell outputs connect to like-numbered network outputs.

Routing in butterfly networks:

At stage x request (a, α) uses cell-output $\alpha_{(n-1-x)}$.

A 16-input butterfly network:



Inverse Butterfly Network

An *inverse butterfly network* is the mirror image of the butterfly.

Extended Shuffle Function

The *extended shuffle function,* $\sigma \mid \mathbf{Z} \to \mathbf{Z}$, is similar to the shuffle function, except that it can operate on any non-negative integer.

(Notation $\mathbf{Z}$ is the set of all non-negative integers.)

Used to shuffle least significant digits of a number.

For example:

$$\sigma_{2,2^2}(11001) = 11010$$

$$\sigma_{2^2,2}(11001) = 11100$$

$$\sigma_{2,2^3}(11001) = 10011, \text{ and}$$

$$\sigma_{10,10^3}(12345) = 13452.$$

Formally:

Let m, k, and i be positive integers. The *extended shuffle function* $\sigma_{m,k} \mid \mathbf{Z} \to \mathbf{Z}$ is defined to be

$$\sigma_{m,k}(i) = i_{(\mathrm{H})} + \left(m \left(i_{(\mathrm{L})} \right) + \left\lfloor \frac{i_{(\mathrm{L})}}{k} \right\rfloor \bmod mk \right),$$

where $i_{(\mathrm{L})} = i \bmod mk$ and $i_{(\mathrm{H})} = \left\lfloor \frac{i}{mk} \right\rfloor mk$.

The extended shuffle function can also be defined for sequences of symbols.

Introducing Baseline Networks

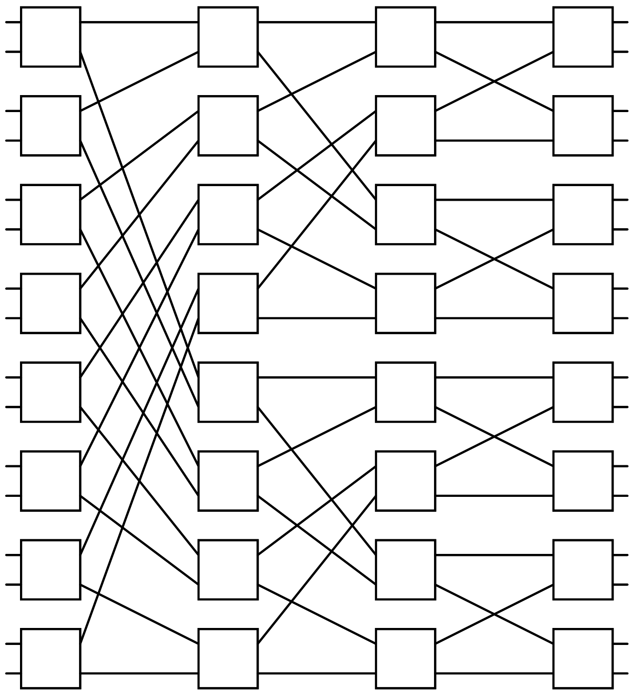
Structure of an n-stage $m \times m$-cell baseline network:

- Each stage has m^{n-1} cells.
- Network inputs connect to like-numbered stage-0 cell inputs.
- Stage-x cell outputs connect to stage-$(x+1)$ cell inputs with a m^{n-x-1}, m extended shuffle, for $0 \leq x < n-1$.
- Stage-$(n-1)$ cell outputs connect to like-numbered network outputs.

Routing in baseline networks:

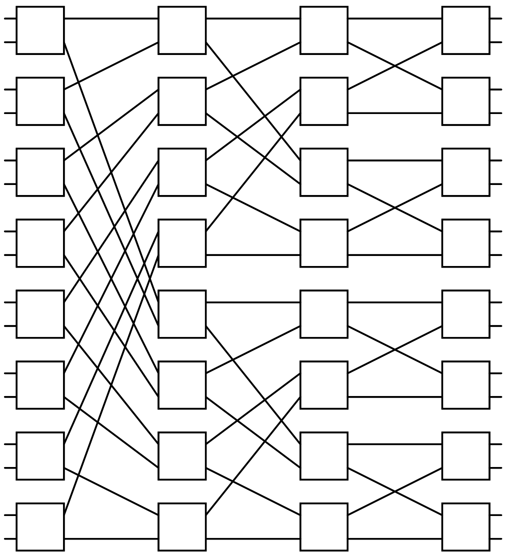
At stage x request (a, α) uses cell-output $\alpha_{(n-1-x)}$.

A 16-input baseline network:



 7-7

Baseline, Butterfly, and Omega Comparison



Similarity:

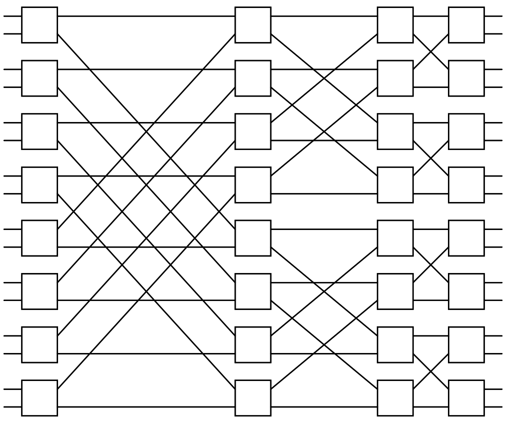
Routing from MSD to LSD of destination number.

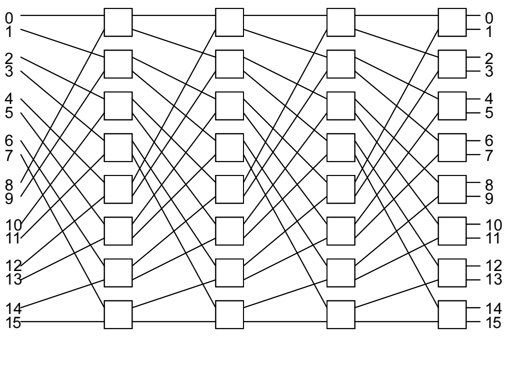
Difference:

Not TE to each other.

Significance of Difference

A connection assignment on one may not be realized by the other.





  7-7

Banyan Network Family

A multistage network is a *banyan network* if…

…there exists exactly one path between…

…each input/output pair.

Banyan Network Examples

Omega, butterfly, inverse omega.

Not Banyan Networks

Modified omega network

Delta Network Family

A multistage network is a *delta network* if

- There is a path between every input/output pair.
- The cell output needed by request (a, α)...

  ...is **uniquely** determined only by...

    ...the cell stage...

    ...and α.

In other words, the one-and-only routing tag...

...is determined by the output only.

Delta Network Property

Lemma: All delta networks are banyan networks.

Proof: directly from the definition...

...by definition, deltas have paths between all inputs and outputs...

..."unique" cell output equivalent to unique path.

A Non-Delta Network

A modified omega network is not a delta network because. . .

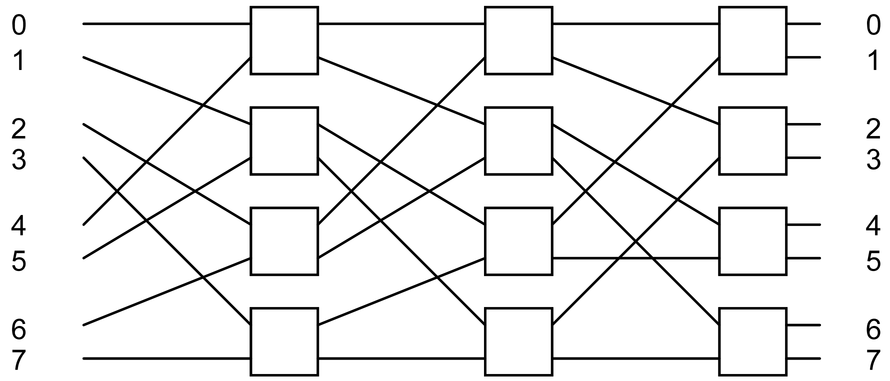
. . .for requests with multiple routing tags. . .

. . .any of several cell outputs could be used.

Delta Network Examples

The following are delta networks: omega, inverse omega, butterfly, inverse butterfly.

The network below is not a delta network, but is a banyan network:



Definition: The part of the routing tag used to reach a cell is called the *path descriptor*.

Theorem 1: The path descriptors for all requests passing through a cell in a delta network are identical.

Proof: (By contradiction.)

Consider requests, (a, α) and (b, β).

Suppose these pass through the same cell, u.

Suppose they have different path descriptors.

Construct a request that starts at a, passes through u and goes to β.

If it's impossible to construct such a request, the network is not a delta network.

If it is possible to construct such a request there are two different routing tags to β, so the network is not a delta network.

Recursive Decomposability of Delta Networks

Theorem 2: All delta networks are recursively decomposable.

Proof outline:

Consider the second stage of cells.

Partition the cells into subsets…

…based on the output number of cells in the previous stage to which they connect.

All cells in each subset will have the same path descriptor.

Expand the subsets by including cells in the following stages to which they connect.

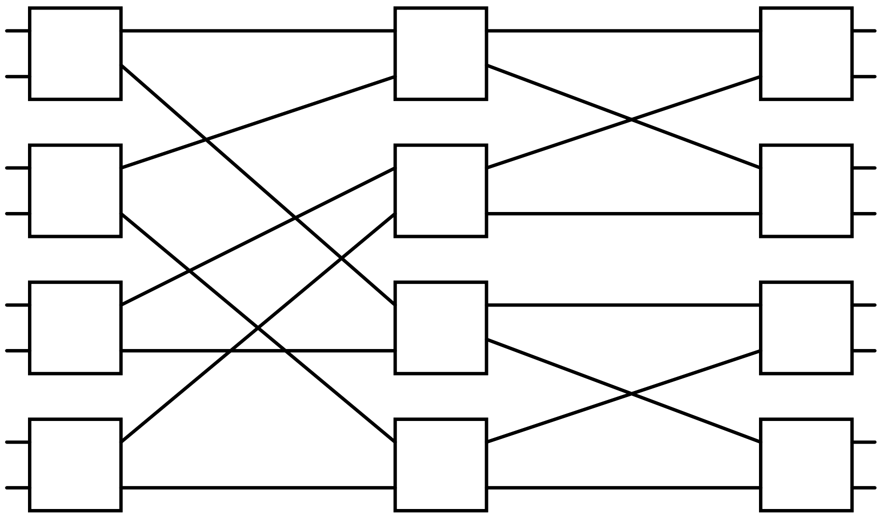
Since each output can be reached using one tag, the subsets remain disjoint.

Bidelta Networks

A network is a *bidelta network* if it is a delta network and its inverse is a delta network.

The omega, baseline, butterfly, and their inverses are all bidelta networks.

The following is a delta network but not a bidelta network:



Definition: A bidelta network cell's *dual path descriptor* is the sequence obtained by joining its path descriptor with the path descriptor of the corresponding cell in the network's inverse.

Theorem 3: The dual path descriptor for a cell in a stage of a bidelta network is unique.

Proof:

Choose two requests, (a, α) and (b, β), so they pass through stage-x cells with the same dual path descriptor.

Call the stage-x cell that (a, α) passes through u.

Call the stage-x cell that (b, β) passes through v.

Call the dual path descriptor $l\,r$.

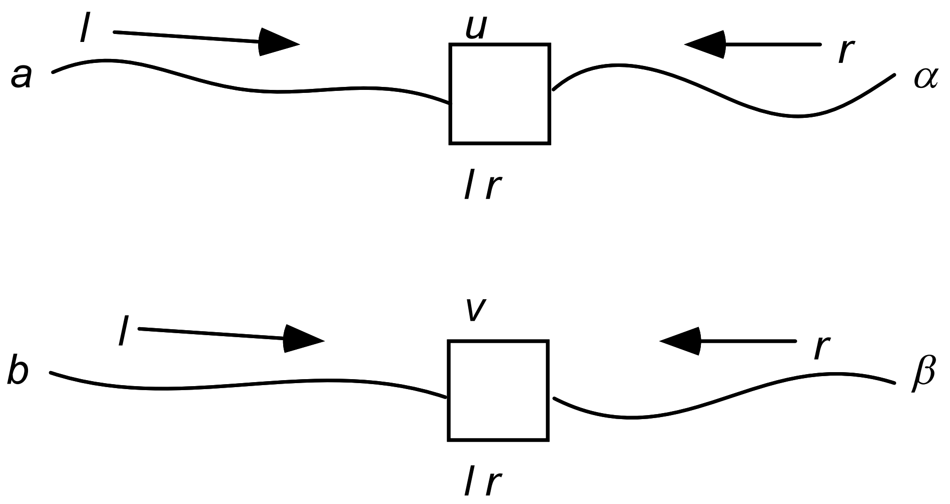
Next, consider request (a, β).

The path must pass through u since l is part of the routing tag to β.

Consider the reverse path from β to a.

The routing tag will start with r, since that's on the path to a.

This leads to cell v, so u and v must be the same cells since these are bidelta networks.



Bidelta Network Properties

Theorem 4: All bidelta networks are topologically equivalent.

Proof outline:

Consider network X.

Will map node labels so that network mapped to baseline.

Step 1: Create new labels for each output in X using the routing tags needed to reach the outputs.

For example, in an omega network

Output 7 in an 4-stage omega is labeled 0 1 1 1.

Output i in an n-stage omega is $i_{(n-1)}i_{(n-2)}\cdots i_{(0)}$.

A new output number is created by treating routing tag as an n-digit, mixed radix number, with the radices determined by the cell size.

The element used for routing in the first stage is used as the most-significant digit.

Step 2: Similarly, create new labels for each input in X using the routing tags needed to reach the inputs.

Create an input number by treating the routing tag as a mixed radix number.

The element used for routing in the first stage is the least-significant digit.

Label the cells using the dual path descriptor.

The networks can be shown to be identical to the baseline by comparing labels of adjacent cells.