Powerlist

Jayadev Misra

Department of Computer Sciences
University of Texas at Austin

Email: misra@cs.utexas.edu
web: http://www.cs.utexas.edu/users/psp

References on Powerlist

1. "Powerlist: A Structure for Parallel Recursion," *ACM Transactions on Programming Languages and Systems,* Vol. 16, No. 6, pp. 1737-1767, November 1994.

2. "Generating–Functions of Interconnection Networks," *Millennial Perspectives in Computer Science: the proceedings of the 1999 Oxford–Microsoft Symposium in honour of Sir Tony Hoare*, St. Catherine's College, Oxford, September 1999.

3. "Derivation of a Parallel String Matching Algorithm," `http://www.cs.utexas.edu/users/psp/StringMatch.ps`

References on Seuss Logic

1. *A Discipline of Multiprogramming*, Jayadev Misra, Springer-Verlag, 2001. A few chapters are available at `http://www.cs.utexas.edu/users/psp/discipline.ps.gz`

2. "A logic for Concurrent Programming (in two parts): Safety and Progress," *Journal of Computer and Software Engineering*, Vol.3, No.2, pp 239-300, 1995. `http://www.cs.utexas.edu/users/psp/SafetyProgress.ps`

Parallel Recursive Algorithms

Fast Fourier Transform
Batcher Sort
Ladner-Fischer Prefix sum
Odd-Even Reductions of tridiagonal Linear Systems
Hypercube Embedding

Recursive Connection Structures

Butterfly Networks, Hypercube
Complete Binary Tree

Powerlists

Powerlist: A list of 2^n items, $n \geq 0$.

Smallest powerlist has a single item, $\langle x \rangle$.

For p, q of the same length:

(tie) $p \mid q$: p concatenated with q,

(zip) $p \bowtie q$: alternate items from p and q, starting with p.

$\langle 0\ 1 \rangle \mid \langle 2\ 3 \rangle$ = $\langle 0\ 1\ 2\ 3 \rangle$, $\langle 0\ 1 \rangle \bowtie \langle 2\ 3 \rangle$ = $\langle 0\ 2\ 1\ 3 \rangle$

Powerlist-length is a power of 2.

Example of a Powerlist Function: Reverse

$\textit{\textbf{rev}}\langle a\ b\ c\ d\rangle = \langle d\ c\ b\ a\rangle$

Definition of Reverse:

$\textit{\textbf{rev}}\langle x\rangle = \langle x\rangle$
$\textit{\textbf{rev}}(p \mid q) = (\textit{\textbf{rev}}\ q) \mid (\textit{\textbf{rev}}\ p)$

Properties:

$\textit{\textbf{rev}}(p \bowtie q) = (\textit{\textbf{rev}}\ q) \bowtie (\textit{\textbf{rev}}\ p)$
$\textit{\textbf{rev}}(\textit{\textbf{rev}}\ p) = p$

Indices in a Powerlist

Base Case: a singleton list

General Case: Deconstruct using $|$ or $\bowtie$

$$\langle \overset{000}{a} \quad \overset{001}{b} \quad \overset{010}{c} \quad \overset{011}{d} \quad \overset{100}{e} \quad \overset{101}{f} \quad \overset{110}{g} \quad \overset{111}{h} \rangle$$

$$\langle \overset{000}{a} \quad \overset{001}{b} \quad \overset{010}{c} \quad \overset{011}{d} \rangle \; | \; \langle \overset{100}{e} \quad \overset{101}{f} \quad \overset{110}{g} \quad \overset{111}{h} \rangle$$

$$\langle \overset{000}{a} \quad \overset{010}{c} \quad \overset{100}{e} \quad \overset{110}{g} \rangle \; \bowtie \; \langle \overset{001}{b} \quad \overset{011}{d} \quad \overset{101}{f} \quad \overset{111}{h} \rangle$$

Scalar Functions

Apply scalar functions pointwise.

$$\neg p = \langle \neg p_0 \;\; \neg p_1 \; ... \rangle$$
$$p + q = \langle p_0 + q_0 \quad p_1 + q_1 \; ... \rangle$$

Note:

$$(p \mid q) + (r \mid s) = (p + r) \mid (q + s)$$
$$(p \bowtie q) + (r \bowtie s) = (p + r) \bowtie (q + s)$$

Laws about powerlists

L0. For singleton powerlists, $\langle x \rangle, \langle y \rangle$**,** $\langle x \rangle \mid \langle y \rangle = \langle x \rangle \bowtie \langle y \rangle$

L1. (Dual Deconstruction)
For non-singleton P**, there are** r, s, u, v **such that**
$P \ = \ r \mid s$ **and** $P \ = \ u \bowtie v$

L2. (Unique Deconstruction)
$(\langle x \rangle = \langle y \rangle) \equiv (x = y)$
$(p \mid q = u \mid v) \equiv (p = u \ \wedge \ q = v)$
$(p \bowtie q = u \bowtie v) \equiv (p = u \ \wedge \ q = v)$

L3. (Commutativity of $\mid$ **and** $\bowtie$**)**
$(p \mid q) \bowtie (u \mid v) \ = \ (p \bowtie u) \mid (q \bowtie v)$

Rotate Right and Rotate Left

$rr\langle a\ b\ c\ d\rangle = \langle d\ a\ b\ c\rangle$
$rl\langle a\ b\ c\ d\rangle = \langle b\ c\ d\ a\rangle$

$rr\langle x\rangle = \langle x\rangle$; $rr(u \bowtie v) = (rr\ v) \bowtie u$
$rl\langle x\rangle = \langle x\rangle$; $rl(u \bowtie v) = v \bowtie (rl\ u)$

Properties:

$rr(rl\ p) = p$
$\textbf{\textit{rev}}(rr(\textbf{\textit{rev}}(rr\ p))) = p$

An Example: The Function *inv*

$$\begin{array}{lccccccccll} & \mathbf{000} & \mathbf{001} & \mathbf{010} & \mathbf{011} & \mathbf{100} & \mathbf{101} & \mathbf{110} & \mathbf{111} & & \\ \textit{\textbf{inv}}\langle & a & b & c & d & e & f & g & h & \rangle & = \\ \langle & a & e & c & g & b & f & d & h & \rangle & \end{array}$$

$\textit{\textbf{inv}}\langle x \rangle = \langle x \rangle$
$\textit{\textbf{inv}}(p \mid q) = (\textit{\textbf{inv}}\; p) \bowtie (\textit{\textbf{inv}}\; q)$

A Duality Property of *inv*

$\textbf{\textit{inv}}(p \bowtie q) = (\textbf{\textit{inv}}\, p) \mid (\textbf{\textit{inv}}\, q)$

Induction : $\{$**Defn.** $\textbf{\textit{inv}}\langle x \rangle = \langle x \rangle$, $\textbf{\textit{inv}}(p \mid q) = (\textbf{\textit{inv}}\, p) \bowtie (\textbf{\textit{inv}}\, q)\}$

$$
\begin{array}{ll}
 & \textbf{\textit{inv}}((r \mid s) \bowtie (u \mid v)) \\
= & \{ \mid,\ \bowtie \ \textbf{commute}\} \\
 & \textbf{\textit{inv}}((r \bowtie u) \mid (s \bowtie v)) \\
= & \{\textbf{definition of } \textbf{\textit{inv}}\} \\
 & \textbf{\textit{inv}}(r \bowtie u) \bowtie \textbf{\textit{inv}}(s \bowtie v) \\
= & \{\textbf{induction}\} \\
 & (\textbf{\textit{inv}}\, r \mid \textbf{\textit{inv}}\, u) \bowtie (\textbf{\textit{inv}}\, s \mid \textbf{\textit{inv}}\, v) \\
= & \{ \mid,\ \bowtie \ \textbf{commute}\} \\
 & (\textbf{\textit{inv}}\, r \bowtie \textbf{\textit{inv}}\, s) \mid (\textbf{\textit{inv}}\, u \bowtie \textbf{\textit{inv}}\, v) \\
= & \{\textbf{definition of } \textbf{\textit{inv}}\}
\end{array}
$$

$\textbf{\textit{inv}}(r \mid s) \mid \textbf{\textit{inv}}(u \mid v)$

Polynomial Evaluation

Evaluate: $p_0 + p_1 \times w + p_2 \times w^2 + p_3 \times w^3$**.**

Evaluate a powerlist $\langle p_0\ p_1\ p_2\ p_3 \rangle$ **at an argument** w**.**

$$\begin{array}{l} \langle x \rangle\ ep\ w = x \\ (p \bowtie q)\ ep\ w = (p\ ep\ w^2) + w \times (q\ ep\ w^2) \end{array}$$

Note: w **could be a powerlist.**

Fast Fourier Transform

Given:

P**:** $\langle ..P_j.. \rangle$ $\quad N$ **items**

Q**:** $\langle ..Q_i.. \rangle$ $\quad N$ **items**

$Q_i = P \; ep \; \omega^i$**;** $\quad \omega = N^{th}$ **principal root of 1.**

$Q = P \; ep \; \langle \omega^0 \omega^1 ... \omega^{N-1} \rangle$

Fast Fourier Transform: Algorithm

Let $P = u \bowtie v$**,** $(U = FFT\ u)$**,** $V = (FFT\ v)$

$Q_i = U_i + \omega^i \times V_i$ **left half of** Q

$Q_{i+len\ u} = U_i - \omega^i \times V_i$ **right half of** Q

$FFT\langle x\rangle = \langle x\rangle$
$FFT(u \bowtie v) = (U + V \times W) \mid (U - V \times W)$
where
$U = FFT\ u$
$V = FFT\ v$
$W = \langle \omega^0 \omega^1 ... \omega^{N/2-1}\rangle$**.**

String Matching Problem

Given: a subject string and a pattern string.

Lengths are powers of 2.

Find: All occurrences of pattern in the subject.

Result has same length as the subject.

> **pattern: "aabb"**
>
> **subject: "aaabbabaaabaaabb"**

Result:

[False,True,False,False,False,False,False,False,

False,False,False,False,True,False,False,False]

String Matching: Simple cases

Subject, Pattern are both singletons.

$$sm\ \langle x\rangle\ \langle y\rangle = \langle x = y\rangle$$

Pattern is a singleton.

$$sm\ \langle x\rangle\ (r \bowtie s) = (sm\ \langle x\rangle\ r)\ \bowtie\ (sm\ \langle x\rangle\ s)$$

Subject is a singleton.

$$sm\ (p \bowtie q)\ \langle y\rangle = \langle \textbf{False} \rangle$$

String Matching: General case

- **Assertion 1:**

  $p \bowtie q$ **matches** $r \bowtie s$ **at some even index** $2k$

  iff p **matches** r **at index** k **and** q **matches** s **at index** k**.**

- **Assertion 2:**

  $p \bowtie q$ **matches** $r \bowtie s$ **at some odd index** $2k+1$

  iff p **matches** s **at index** k**,** q **matches** r **at index** $k+1$**.**

A proof of one part of assertion 2.

$$
\begin{array}{ll}
 & p \bowtie q \textbf{ matches } r \bowtie s \textbf{ at } 2k+1 \\
\equiv & \{\textbf{definition of “matches”}\} \\
 & (\forall k : 0 \leq j < |p \bowtie q| : (p \bowtie q)_j = (r \bowtie s)_{j+2k+1}) \\
\Rightarrow & \{\textbf{consider only the odd indices } 2j+1\} \\
 & (\forall j : 0 \leq 2j+1 < |p \bowtie q| : \\
 & (p \bowtie q)_{2j+1} = (r \bowtie s)_{2j+1+2k+1}) \\
\Rightarrow & \{(p \bowtie q)_{2j+1} = q_j, (r \bowtie s)_{2j+1+2k+1} = r_{j+k+1}\} \\
 & (\forall j : 0 \leq j < |q| : q_j = r_{j+k+1}) \\
\Rightarrow & \{\textbf{definition of “matches”}\} \\
 & q \textbf{ matches } r \textbf{ at } k+1
\end{array}
$$

String Matching Algorithm

$sm\ \langle x\rangle\ \langle y\rangle = \langle x = y\rangle$
$sm\ \langle x\rangle\ (r \bowtie s) = (sm\ \langle x\rangle\ r)\ \bowtie\ (sm\ \langle x\rangle\ s)$
$sm\ (p \bowtie q)\ \langle y\rangle = \langle$**False**$\rangle$
$sm\ (p \bowtie q)\ (r \bowtie s) =$

$$(smpr\ \wedge\ smqs)\ \bowtie\ (sm'qr\ \wedge\ smps)$$

where

$$smpr = sm\ p\ r$$
$$smqs = sm\ q\ s$$
$$sm'qr = ls(sm\ q\ r)$$
$$smps = sm\ p\ s$$

The definition of left shift, ls, is

$ls\ \langle x\rangle = \langle$**False**$\rangle$
$ls\ (u \bowtie v) = v\ \bowtie (ls\ u)$

Calculate sm'

$$
\begin{array}{ll}
 & sm'\ \langle x \rangle\ \langle y \rangle \\
= & \{\textbf{definition of } sm'\} \\
 & ls(sm\ \langle x \rangle\ \langle y \rangle) \\
= & \{\textbf{definition of } sm\ \langle x \rangle\ \langle y \rangle\} \\
 & ls\langle x = y \rangle \\
= & \{\textbf{definition of } ls \textbf{ on a singleton list}\} \\
 & \langle \textbf{False} \rangle
\end{array}
$$

sm' (contd.)

$$
\begin{array}{ll}
 & sm'\ \langle x\rangle\ (r \bowtie s) \\
= & \{\textbf{definition of } sm'\} \\
 & ls(sm\ \langle x\rangle\ (r \bowtie s)) \\
= & \{\textbf{definition of } sm\ \langle x\rangle\ (r \bowtie s)\} \\
 & ls((sm\ \langle x\rangle\ r)\ \bowtie\ (sm\ \langle x\rangle\ s)) \\
= & \{\textbf{definition of } ls\ (u \bowtie v)\ \} \\
 & (sm\ \langle x\rangle\ s)\ \bowtie\ ls(sm\ \langle x\rangle\ r) \\
= & \{\textbf{definition of } sm'\} \\
 & (sm\ \langle x\rangle\ s)\ \bowtie\ (sm'\ \langle x\rangle\ r)
\end{array}
$$

A similar derivation shows that

$$sm'\ (p \bowtie q)\ \langle y\rangle = \langle \textbf{False} \rangle$$

sm' (contd.)

$$
\begin{array}{ll}
 & sm'\ (p \bowtie q)\ (r \bowtie s) \\
= & \{\textbf{definition of } sm'\} \\
 & ls(sm\ (p \bowtie q)\ (r \bowtie s)) \\
= & \{\textbf{definition of } sm\ (p \bowtie q)\ (r \bowtie s)\} \\
 & ls((smpr\ \wedge\ smqs)\ \bowtie\ (sm'qr\ \wedge\ smps)) \\
= & \{\textbf{definition of } ls\} \\
 & (sm'qr\ \wedge\ smps)\ \bowtie\ ls((smpr\ \wedge\ smqs)) \\
= & \{ls\ \textbf{distributes over}\ \wedge\ \textbf{in the second term}\} \\
 & (sm'qr\ \wedge\ smps)\ \bowtie\ (ls(smpr)\ \wedge\ ls(smqs)) \\
= & \{ls(smpr) = sm'pr\ \textbf{and}\ ls(smqs) = sm'qs\} \\
 & (sm'qr\ \wedge\ smps)\ \bowtie\ (sm'pr\ \wedge\ sm'qs)
\end{array}
$$

Putting the Pieces Together

$$
\begin{aligned}
&sm\ \langle x\rangle\ \langle y\rangle = \langle x = y\rangle\\
&sm\ \langle x\rangle\ (r \bowtie s) = (sm\ \langle x\rangle\ r)\ \bowtie\ (sm\ \langle x\rangle\ s)\\
&sm\ (p \bowtie q)\ \langle y\rangle = \langle\textbf{False}\rangle\\
&sm\ (p \bowtie q)\ (r \bowtie s) =\\
&\qquad\qquad (smpr\ \wedge\ smqs)\ \bowtie\ (sm'qr\ \wedge\ smps)\\
&\qquad\qquad \textbf{where}\\
&\qquad\qquad\qquad smpr = sm\ p\ r\\
&\qquad\qquad\qquad smqs = sm\ q\ s\\
&\qquad\qquad\qquad sm'qr = sm'\ q\ r\\
&\qquad\qquad\qquad smps = sm\ p\ s
\end{aligned}
$$

Putting the Pieces Together; contd.

$$
\begin{aligned}
& sm' \ \langle x \rangle \ \langle y \rangle = \langle \textbf{False} \rangle \\
& sm' \ \langle x \rangle \ (r \bowtie s) = (sm \ \langle x \rangle \ s) \ \bowtie \ (sm' \ \langle x \rangle \ r) \\
& sm' \ (p \bowtie q) \ \langle y \rangle = \langle \textbf{False} \rangle \\
& sm' \ (p \bowtie q) \ (r \bowtie s) = \\
& \qquad\qquad (sm'qr \ \wedge \ smps) \ \bowtie \ (sm'pr \ \wedge \ sm'qs) \\
& \qquad\qquad \textbf{where} \\
& \qquad\qquad\qquad sm'qr = sm' \ q \ r \\
& \qquad\qquad\qquad smps = sm \ p \ s \\
& \qquad\qquad\qquad sm'pr = sm' \ p \ r \\
& \qquad\qquad\qquad sm'qs = sm' \ q \ s
\end{aligned}
$$

Sorting

Generic sorting Scheme

$$sort\langle x\rangle = \langle x\rangle$$
$$sort(p \bowtie q) = (sort\ p)\ merge\ (sort\ q)$$

Comparator:

$p:\ \langle 2\ 3\rangle$
$q:\ \langle 4\ 1\rangle$
$p\ \updownarrow\ q = \langle 2\ 4\ 1\ 3\rangle$
$\langle x\rangle \updownarrow \langle y\rangle = \langle x \text{ min } y \quad x \text{ max } y\rangle$
$p \updownarrow q = \langle p \text{ min } q\ \rangle \bowtie \langle p \text{ max } q\rangle$

Batcher Merge

Bitonic: $u\ merge\ v = bi(u \mid (\textbf{rev}\ v))$**, where**

$$bi\langle x\rangle = \langle x\rangle$$
$$bi(p \bowtie q) = (bi\ p) \updownarrow (bi\ q)$$

Batcher Merge:

$$\langle x\rangle\ merge\ \langle y\rangle = \langle x\rangle \updownarrow \langle y\rangle$$
$$(p \bowtie q)\ merge\ (u \bowtie v) = (p\ merge\ v)\ \updownarrow\ (q\ merge\ u)$$

Theorem:: $p\ merge\ q = bi(p \mid (\textbf{rev}\ q))$

Proof, Base case

Theorem:: $p \; merge \; q = bi(p \mid (\textbf{\textit{rev}} \; q))$

Base: Let $p, \; q = \langle x \rangle, \; \langle y \rangle$

$$
\begin{array}{ll}
 & bi(\langle x \rangle \mid rev\langle y \rangle) \\
= & \{\textbf{definition of } rev\} \\
 & bi(\langle x \rangle \mid \langle y \rangle) \\
= & \{(\langle x \rangle \mid \langle y \rangle) = (\langle x \rangle \bowtie \langle y \rangle)\} \\
 & bi(\langle x \rangle \bowtie \langle y \rangle) \\
= & \{\textbf{definition of } bi\} \\
 & \langle x \rangle \updownarrow \langle y \rangle \\
= & \{\textbf{definition of } merge\} \\
 & \langle x \rangle \; merge \; \langle y \rangle
\end{array}
$$

Proof, Inductive case

Theorem:: $p \; merge \; q = bi(p \mid (\textbf{\textit{rev}} \; q))$

Induction: Let $p, \; q = r \bowtie s, \; u \bowtie v$

$$
\begin{aligned}
& bi(p \mid (rev \; q)) \\
= \quad & \{\textbf{expanding } p, q\} \\
& bi((r \bowtie s) \mid rev(u \bowtie v)) \\
= \quad & \{\textbf{definition of } rev\} \\
& bi((r \bowtie s) \mid (rev \; v \bowtie rev \; u)) \\
= \quad & \{ \mid , \bowtie \; \textbf{commute}\} \\
& bi((r \mid rev \; v) \bowtie (s \mid rev \; u))
\end{aligned}
$$

$=$ {**definition of** bi}

$bi(r \mid rev\ v) \updownarrow bi(s \mid rev\ u)$

$=$ {**induction**}

$(r\ merge\ v) \updownarrow (s\ merge\ u)$

$=$ {**definition of** $merge$}

$(r \bowtie s)\ merge\ (u \bowtie v)$

$=$ {**using the definitions of** p, q}

$p\ merge\ q$

Prefix Sum

L: **powerlist of scalars,**

$\oplus$ **: binary, associative operator on that scalar type.**

$(ps\ L)$: **prefix sum of** L **with respect to** $\oplus$ **.**

$(ps\ L)$ **is a list of the same length as** L **given by**

$ps\ \langle x_0, x_1, .., x_i, .., x_N \rangle$

$= \langle x_0, x_0 \oplus x_1, .., x_0 \oplus x_1 \oplus ..x_i, .., x_0 \oplus x_1 \oplus .. \oplus x_N \rangle$

The i^{th} **element of** $(ps\ L)$**:**
apply $\oplus$ **to the first** i **elements of** L**.**

Simple scheme for prefix sum

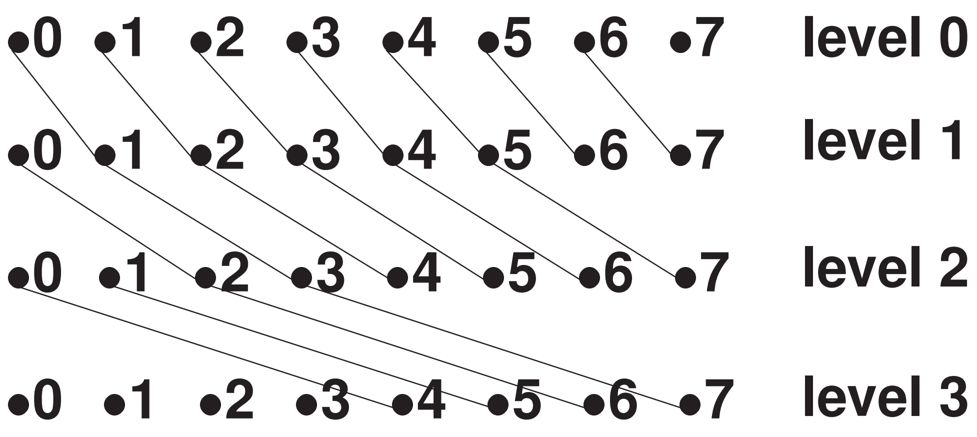

Figure 1: A network for prefix sum.

Ladner and Fischer Scheme

Apply $\oplus$ to adjacent elements x_{2i}, x_{2i+1}.

This computes the list $\langle x_0 \oplus x_1, .. \; x_{2i} \oplus x_{2i+1}, ..\rangle$.

This list has half as many elements as the original.

Its prefix sum is then computed recursively.

Result is $\langle x_0 \oplus x_1, .., x_0 \oplus x_1 \oplus .. \oplus x_{2i} \oplus x_{2i+1}, \ldots\rangle$.

This has half of the elements of the final list.

Missing elements are:

$x_0, x_0 \oplus x_1 \oplus x_2, .., x_0 \oplus x_1 \oplus .. \oplus x_{2i}, ..$.

Add $x_2, x_4, ..,$ appropriately.

Specification of prefix sum

0 **is the left identity element of** $\oplus$ **, i.e.,** $0 \oplus x = x$**.**

p^***: shift** p **to the right by one.**

$\langle a\ b\ c\ d\rangle^* = \langle 0\ a\ b\ c\rangle$**.**

$$\langle x\rangle^* = \langle 0\rangle$$
$$(p \bowtie q)^* = q^* \bowtie p$$

It is easy to show

S1. $(r \oplus s)^* = r^* \oplus s^*$
S2. $(p \bowtie q)^{**} = p^* \bowtie q^*$

Specification, contd.

In (DE), z is unknown, L is a powerlist .

(DE) $z = z^* \oplus L$

This equation has a unique solution in z:

$$
\begin{aligned}
z_0 &= (z^*)_0 \oplus L_0 \\
&= 0 \oplus L_0 \\
&= L_0 \qquad\qquad \textbf{, and} \\
z_{i+1} &= z_i \oplus L_{i+1} \textbf{ ,} \; 0 \le i < (len\ L) - 1
\end{aligned}
$$

For $L = \langle a\ b\ c\ d \rangle$**,**

$z = \langle a \quad a \oplus b \quad a \oplus b \oplus c \quad a \oplus b \oplus c \oplus d \rangle$

This is $(ps\ L)$**, (unique) solution of (DE).**

prefix sum; simple scheme

$sps\ \langle x \rangle = \langle x \rangle$
$sps\ L = (sps\ u) \bowtie (sps\ v)$
where $u \bowtie v = L^* \oplus L$

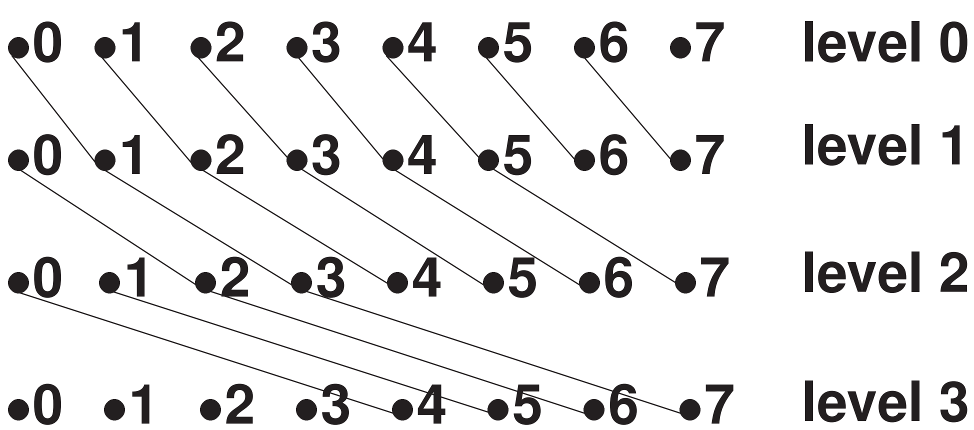

Figure 2: A network for prefix sum.

Explanation of the simple scheme

In the first level, $L^* \oplus L$ is computed.

If $L = \langle x_0, x_1, .., x_i, \ldots \rangle$ then

$L^* \oplus L$ **is** $\langle x_0, x_0 \oplus x_1, .., x_i \oplus x_{i+1} .. \rangle$.

This is the zip of two sublists:

$\langle x_0, x_1 \oplus x_2, .., x_{2i-1} \oplus x_{2i}, .. \rangle$ **and**

$\langle x_0 \oplus x_1, .., x_{2i} \oplus x_{2i+1}, .. \rangle$.

Compute prefix sums of these two lists and zip.

Ladner-Fischer scheme

$$lf\langle x\rangle = \langle x\rangle$$
$$lf(p \bowtie q) = (t^* \oplus p) \bowtie t$$
$$\textbf{where } t = lf(p \oplus q)$$

Derivation of Ladner-Fischer scheme

For a powerlist $p \bowtie q$, what is $ps(p \bowtie q)$?

Let $r \bowtie t = ps(p \bowtie q)$. We solve for r, t.

$$
\begin{array}{ll}
 & r \bowtie t \\
= & \quad \{r \bowtie t = ps\ (p \bowtie q).\ \textbf{Using (DE)}\} \\
 & (r \bowtie t)^* \oplus (p \bowtie q) \\
= & \quad \{(r \bowtie t)^* = t^* \bowtie r\} \\
 & (t^* \bowtie r) \oplus (p \bowtie q) \\
= & \quad \{\ \oplus\ ,\ \bowtie\ \ \textbf{commute}\} \\
 & (t^* \oplus p) \bowtie (r \oplus q)
\end{array}
$$

Ladner-Fischer scheme (contd.)

deconstruct: $r \bowtie t = (t^* \oplus p) \bowtie (r \oplus q)$**,**

LF1. $r = t^* \oplus p$ **, and**

LF2. $t = r \oplus q$

Eliminate r **from (LF2) using (LF1):**

$t = t^* \oplus p \oplus q$**.**

Use (DE) and this equation

LF3. $t = ps(p \oplus q)$

Summary of derivation of the Ladner-Fischer scheme

$$
\begin{array}{ll}
 & ps(p \bowtie q) \\
= & \quad \{\textbf{by definition}\} \\
 & r \bowtie t \\
= & \quad \{\textbf{ Using (LF1) for } r\} \\
 & (t^* \oplus p) \bowtie t
\end{array}
$$

where t is defined by LF3:

LF3. $t = ps(p \oplus q)$

Generating Functions

Typically used on sequences of numbers.
I will apply generating functions to interconnection networks.
I will prove that two families of interconnection networks are isomorphic, using their generating functions.

Example of Interconnection Network

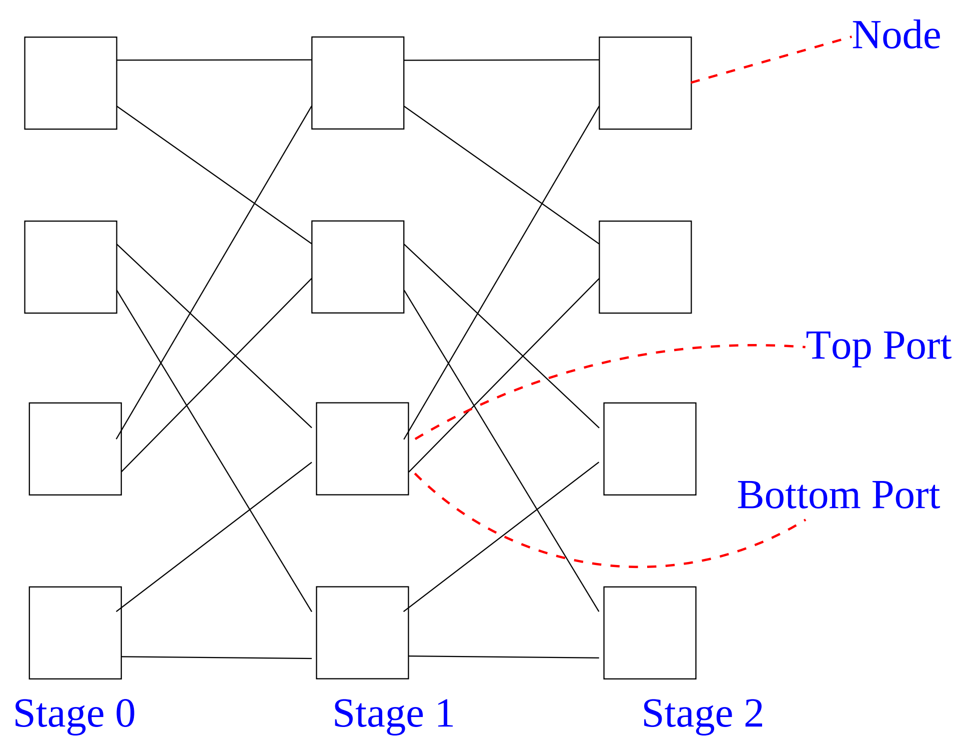

Figure 3: Interconnection Network, $N = 4$

Terminology

Interconnection network of size N, $N = 2^n$, has:

$n + 1$ stages, numbered 0 through n.
Each stage has N nodes.
Each node, except initial and final nodes, has 2 input and output ports. top port, bottom port.
Each output port connected to a distinct input port of the next stage.

Family of Networks

A **family** has a network for each value of N.

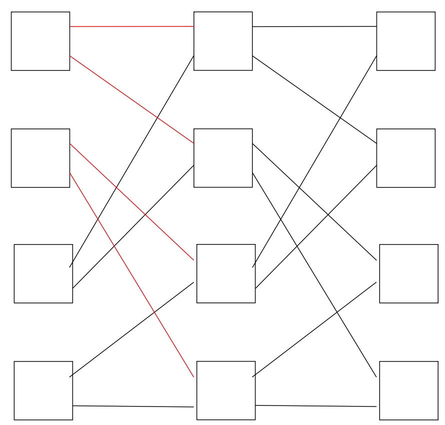

Figure 4: Benes Network, $N = 4$

Top input lines come in order from the upper half of the previous stage.

Bottom input lines come in order from the lower half.

Clos Networks

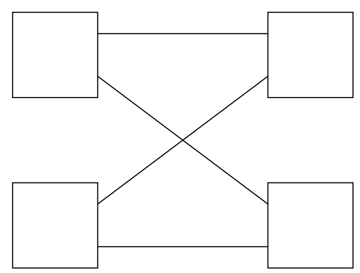

Figure 5: Clos Network for $N = 2$

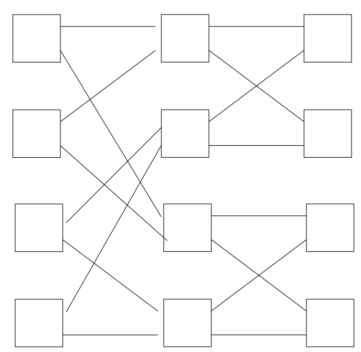

Figure 6: Clos Network, $N = 4$

Clos Network Interconnection

$N = 2$**: shown earlier.**

$N = 2^{n+1}$**: Input lines in the upper/lower half of stage 1 are the top/bottom output lines the stage 0, in order. Then, append two copies of network of size** 2^n**.**

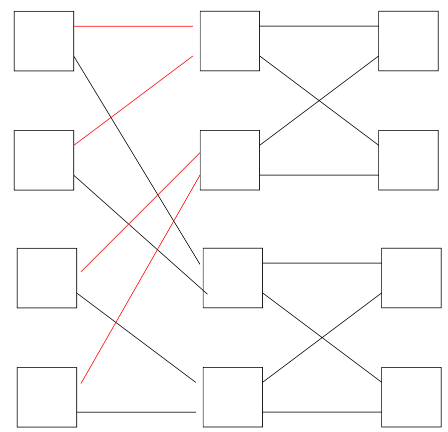

Figure 7: Clos Network, $N = 4$

Butterfly Network and its mirror image

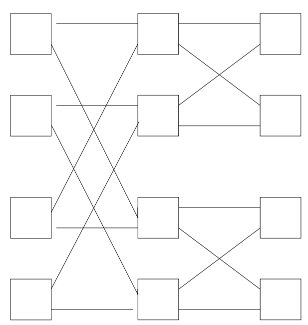

Figure 8: Butterfly Network for $N = 4$

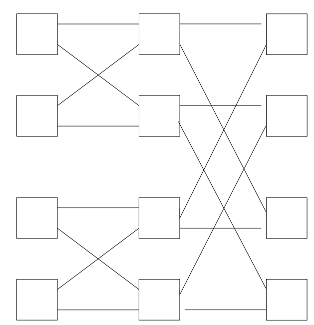

Figure 9: Mirror Image of the Butterfly Network, $N = 4$

Isomorphism

Required to show: two families are isomorphic, that is,

The networks corresponding to N, for each N, in both families are isomorphic.

Strategy: Represent each family by a generating function.

Two families are isomorphic if the corresponding functions are identical (upto a permutation of arguments).

Labelling

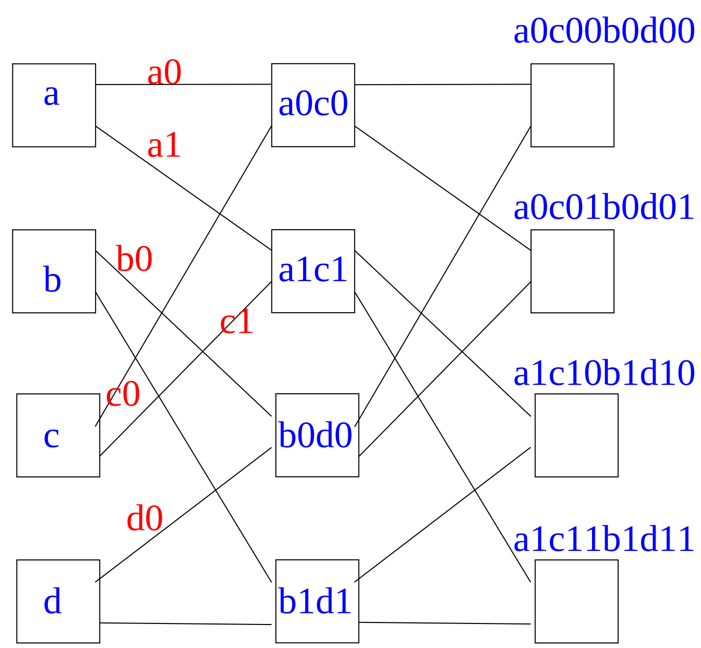

Figure 10: Labelling a Network

A network can be constructed from the set of labels of the final

nodes.

Notation

$\{p\}$: **set of elements of** p**,**

$p0$: **concatenate** 0 **to the end of each element of** p**. Similarly,** $p1$**.**

$p.q$: **concatenate corresponding elements of** p **and** q**.**

- $\{u \bowtie v\} = \{u\} \cup \{v\}$**, and** $\{u \mid v\} = \{u\} \cup \{v\}$**.**
- $(p \bowtie q)0 = (p0 \bowtie q0)$**, and** $(p \mid q)0 = (p0 \mid q0)$**.**
- $(p \bowtie q).(u \bowtie v) = (p.u) \bowtie (q.v)$**, and** $(p \mid q).(u \mid v) = (p.u) \mid (q.v)$**.**
- **For a permuting function** h**,**

$(h\ p)0 = h(p0)$ **and** $h(p.q) = (h\ p).(h\ q)$**.**

Generating-Function for Clos Networks

$$c(\langle x\rangle) = \langle x\rangle$$
$$c(u \bowtie v) = c(u0.v0) \mid c(u1.v1)$$

Note: $\langle a0b0\ c0d0\rangle = \langle a0\ c0\rangle.\langle b0\ d0\rangle = \langle a\ c\rangle 0.\langle b\ d\rangle 0$**.**

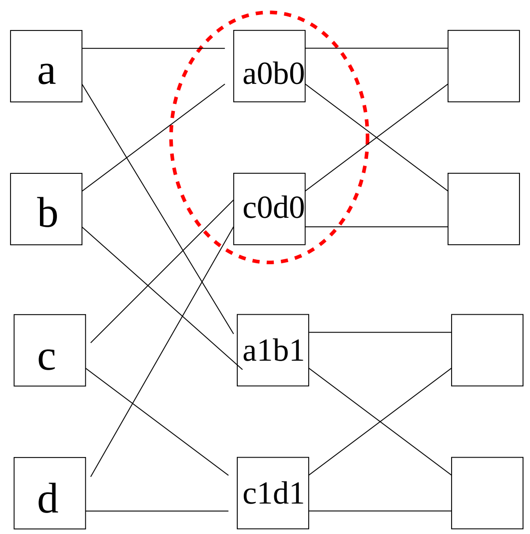

Figure 11: Clos Network, $N = 4$

Generating-Function for Butterfly Network

$f(\langle x\rangle) = \langle x\rangle$
$f(u \mid v) = f(u0.v0) \mid f(u1.v1)$

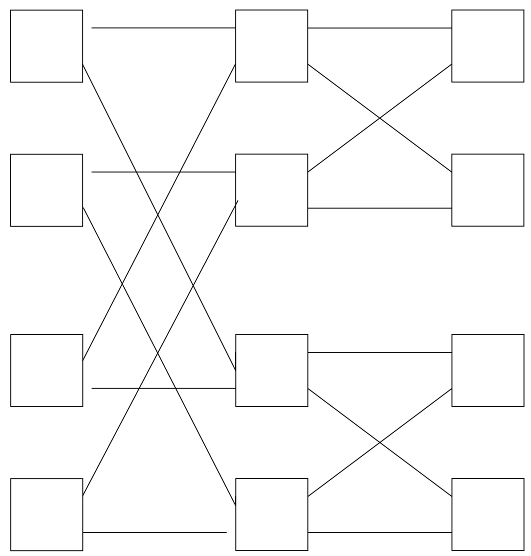

Figure 12: Butterfly Network for $N = 4$

Mirror Image of the Butterfly Network

$m(\langle x \rangle) = \langle x \rangle$
$m(u \bowtie v) = m(u0.v0) \bowtie m(u1.v1)$

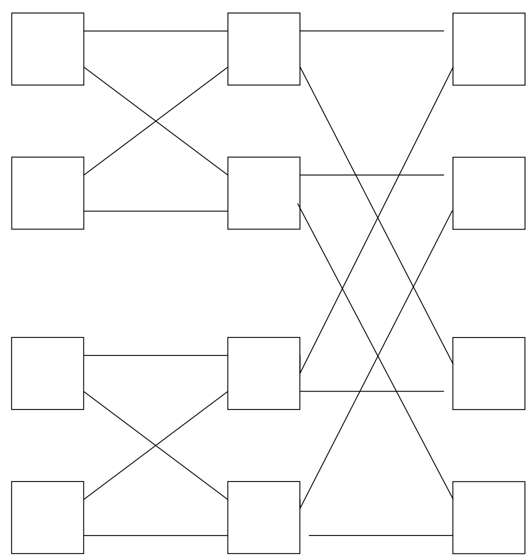

Figure 13: Mirror Image of the Butterfly Network, $N = 4$

Generating-Function for Benes Network

b_k **describes the Benes network with** $k+1$ **stages.**

$b_0\,(p) = \{p\}$
$b_{k+1}(u \mid v) = b_k(u0.v0 \bowtie u1.v1)$**, for** $0 \leq k \leq \overline{u}$**, where** $\overline{u} = \log_2 |u|$

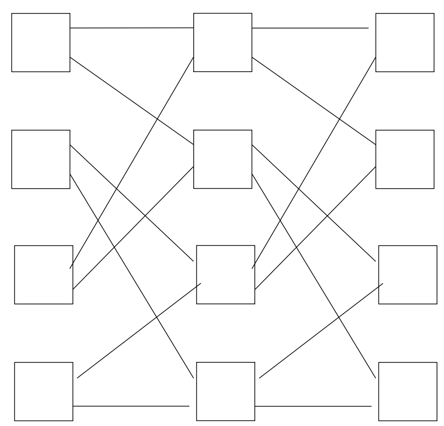

Figure 14: Benes Network, $N = 4$

Definition of Isomorphism

Families with generating functions g and h are isomorphic iff $\{g\} = \{h \circ \pi\}$, for some permutation function π. That is,

$\{g(p)\} = \{h(\pi(p))\}$, for all p.

Isomorphism is an equivalence relation.

The order in the final list does not matter.

Isomorphism: Clos, Butterfly, Mirror Image

Theorem 1:

1. $c \circ inv = f$

2. $c = inv \circ m$

Proof of 2 (Induction): Let the argument powerlist be $u \bowtie v$.

$$
\begin{array}{ll}
 & inv(m(u \bowtie v)) \\
= & \{\textbf{definition of } m\} \\
 & inv[m(u0.v0) \bowtie m(u1.v1)] \\
= & \{\textbf{Property of } inv\} \\
 & inv[m(u0.v0)] \mid inv[m(u1.v1)] \\
= & \{\textbf{induction}\} \\
 & c(u0.v0) \mid c(u1.v1) \\
= & \{\textbf{definition of } c\}
\end{array}
$$

$$c(u \bowtie v)$$

Isomorphism: Benes, Butterfly

Lemma: $b_n(u \bowtie v) = (b_n\ u) \cup (b_n\ v)$**, for all** n**,** $0 \leq n \leq \overline{u}$**.**

Theorem 2: $b_{\overline{u}}(u) = \{f(u)\}$

$$
\begin{array}{ll}
 & b_{\overline{p \mid q}}(p \mid q) \\
= & \{\textbf{definition of } b\} \\
 & b_{\overline{p}}(p0.q0 \bowtie p1.q1) \\
= & \{\textbf{Lemma}\} \\
 & b_{\overline{p}}(p0.q0) \cup b_{\overline{p}}(p1.q1) \\
= & \{\textbf{induction. Note that } \overline{p} = \overline{p0.q0} = \overline{p1.q1}\} \\
 & \{f(p0.q0)\} \cup \{f(p1.q1)\} \\
= & \{\textbf{Observation 1}\} \\
 & \{f(p0.q0) \mid f(p1.q1)\} \\
= & \{\textbf{Definition of } f\} \\
 & \{f(p \mid q)\}
\end{array}
$$

Higher Dimensional Arrays

A matrix of r rows and c columns is a powerlist of c elements: each element is a powerlist of length r storing the items of a column.

Think in terms of array operations rather than operations on nested powerlists.

Introduce construction operators, analogous to $|$ and $\bowtie$, for tie and zip along any specified dimension.

$|', \bowtie'$ for the corresponding operators in dimension 1, $|'', \bowtie''$ for the dimension 2, etc.

Examples of Matrices

$$A = \begin{array}{c} \begin{array}{cc} \wedge & \wedge \end{array} \\ \left\langle \begin{array}{cc} 2 & 4 \\ 3 & 5 \end{array} \right\rangle \\ \begin{array}{cc} \vee & \vee \end{array} \end{array} \qquad B = \begin{array}{c} \begin{array}{cc} \wedge & \wedge \end{array} \\ \left\langle \begin{array}{cc} 0 & 1 \\ 6 & 7 \end{array} \right\rangle \\ \begin{array}{cc} \vee & \vee \end{array} \end{array}$$

$$A \mid B = \begin{array}{c} \begin{array}{cccc} \wedge & \wedge & \wedge & \wedge \end{array} \\ \left\langle \begin{array}{cccc} 2 & 4 & 0 & 1 \\ 3 & 5 & 6 & 7 \end{array} \right\rangle \\ \begin{array}{cccc} \vee & \vee & \vee & \vee \end{array} \end{array} \qquad A \bowtie B = \begin{array}{c} \begin{array}{cccc} \wedge & \wedge & \wedge & \wedge \end{array} \\ \left\langle \begin{array}{cccc} 2 & 0 & 4 & 1 \\ 3 & 6 & 5 & 7 \end{array} \right\rangle \\ \begin{array}{cccc} \vee & \vee & \vee & \vee \end{array} \end{array}$$

$$A \mid' B = \begin{array}{c} \begin{array}{cc} \wedge & \wedge \end{array} \\ \left\langle \begin{array}{cc} 2 & 4 \\ 3 & 5 \\ 0 & 1 \\ 6 & 7 \end{array} \right\rangle \\ \begin{array}{cc} \vee & \vee \end{array} \end{array} \qquad A \bowtie' B = \begin{array}{c} \begin{array}{cc} \wedge & \wedge \end{array} \\ \left\langle \begin{array}{cc} 2 & 4 \\ 0 & 1 \\ 3 & 5 \\ 6 & 7 \end{array} \right\rangle \\ \begin{array}{cc} \vee & \vee \end{array} \end{array}$$

Definition of a matrix

Define a matrix to be either

a singleton matrix $\langle\langle x\rangle\rangle$, or
$p \mid q$ where p, q are (similar) matrices, or
$u \mid' v$ where u, v are (similar) matrices.

Matrix Transposition

$\tau\langle\langle x\rangle\rangle = \langle\langle x\rangle\rangle$
$\tau(p \mid q) = (\tau\ p) \mid' (\tau\ q)$
$\tau(u \mid' v) = (\tau\ u) \mid (\tau\ v)$

Matrix Transposition

$$\sigma\langle\langle x\rangle\rangle = \langle\langle x\rangle\rangle$$
$$\sigma((p \mid q) \mid' (u \mid v)) = ((\sigma\ p) \mid' (\sigma\ q)) \mid ((\sigma\ u) \mid' (\sigma\ v))$$

In the figure $p' = \sigma\ p$, etc.

$$\sigma \begin{array}{|c|c|}\hline p & q \\ \hline u & v \\ \hline\end{array} \quad = \quad \begin{array}{|c|c|}\hline p' & u' \\ \hline q' & v' \\ \hline\end{array}$$

Figure 15: transposition of a square matrix.