# UNIFORM EMULATIONS OF THE SHUFFLE-EXCHANGE NETWORK

H.L. Bodlaender and J. van Leeuwen

RUU-CS-84-5

July 1984

Rijksuniversiteit Utrecht

**Vakgroep informatica**

Budapestlaan 6 3584 CD Utrecht
**Corr. adres:** Postbus 80.012 3508 TA Utrecht
Telefoon 030-531454
The Netherlands

# UNIFORM EMULATIONS OF THE SHUFFLE-EXCHANGE NETWORK

H.L.Bodlaender and J. van Leeuwen

Technical Report RUU-CS-84-5
July 1984

Department of Computer Science
University of Utrecht
P.O. Box 80.012, 3508 TA Utrecht
the Netherlands

This paper will appear in the Proceedings of the International Workshop on "Graphtheoretic Concepts in Computer Science" (WG '84), edited by U. Pape, Technische Universität Berlin, June 13-15, 1984.

# UNIFORM EMULATIONS OF THE SHUFFLE-EXCHANGE NETWORK

H.L.Bodlaender* and J. van Leeuwen

Department of Computer Science, University of Utrecht

P.O.Box 80.012, 3508 TA Utrecht, the Netherlands.

Abstract. Parallel algorithms are normally designed for execution on networks of N processors, with N depending on the size of the problem to be solved. In practice there will be a varying problem size but a fixed network size. In [3] the notion of network emulation was proposed, to obtain a structure preserving simulation of large networks on smaller networks. We analyse the concept for the case of the shuffle-exchange network, a common interconnection network underlying many multiprocessor algorithms.

1. Introduction. Parallel algorithms are normally designed for execution on a suitable network of N processors, with N depending on the size of the problem to be solved. In practice N will be large and varying, whereas processor networks will be small and fixed. The resulting disparity between algorithm design and implementation must be resolved by simulating a network of some size N on a fixed and smaller size network of a similar or different kind, in a structure preserving manner. Notions of simulation are well-understood in e.g. automata theory (see [5]), and suitable analogs can be brought to bear on networks of processors. In this paper we study a notion of simulation, termed emulation, proposed by Fishburn and Finkel [3].

* The work of this author was supported by the Foundation for Computer Science (SION) of the Netherlands Organization for the Advancement of Pure Research (ZWO).

Definition. Let $G = (V_G, E_G)$ and $H = (V_H, E_H)$ be networks of processors (graphs). We say that G can be emulated on H if there exists a function $f: V_G \rightarrow V_H$ such that for every edge $(g, g') \in E_G$ : $f(g) = f(g')$ or $(f(g), f(g')) \in E_H$. The function f is called an emulation function or, in short, an emulation of G on H.

Clearly, emulation between networks is transitive. We shall only be interested in emulations f that are "onto".

Let f be an emulation of G on H. Any processor $h \in V_H$ must actively emulate the processors $\in f^{-1}(h)$ in G. When $g \in f^{-1}(h)$ communicates information to a neighboring processor g', then h must communicate the corresponding information "internally", when it emulates g' itself or to a neighboring processor $h' = f(g')$ in H otherwise. If all processors act synchronously in G, then the emulation will be slowed by a factor proportional to $\max_h |f^{-1}(h)|$.

Definition. Let G, H, and f be as above. The emulation f is said to be (computationally) uniform if for all $h, h' \in V_H$ : $|f^{-1}(h)| = |f^{-1}(h')|$.

Every uniform emulation f has associated with it a fixed constant c , called: the computation factor, such that for all $h \in V_H$ : $|f^{-1}(h)| = c$ . It means that every processor of H emulates the same number of processors of G. Again, uniform emulation between networks is transitive. When G can be uniformly emulated on H and H can be uniformly emulated on G, then G and H are necessarily isomorphic. (Thus uniform emulation establishes a partial ordering of networks.) For graphs A, B let A[B] denote the composition of A and B (cf. [4]).

Lemma 1.1 G can be uniformly emulated on H if and only if there exists a graph G' such that G is a spanning subgraph of H[G'].

Proof.

$\Rightarrow$ Let f be a uniform emulation of G on H with computation factor c. The sets $\{f^{-1}(h)\}$, $h \in H$, partition G into blocks of size c. Let G' be any graph on c nodes such that the induced subgraph of every block (in G) is contained in G'. Next observe that for any two nodes $g \in f^{-1}(h)$ and $g' \in f^{-1}(h')$ of G: $(g,g') \in E_G \Rightarrow h = h'$ (and the edge is in G') or $(h,h') \in E_H$. It follows that G is a spanning subgraph of H[G'].

$\Leftarrow$ From the definition of composition (cf. [4]), by projection on H. □

For functions f defined on n-bit numbers b we use :

$f_i(b)$ : $(f(b))_i$ (projection on the $i^{th}$ bit)

We use b, c, .. to denote full addresses and x, y, .. to denote segments of bits. Individual bits are denoted $\alpha$, $\beta$, .. .

Definition. The shuffle-exchange network is the graph $S_n = (V_n, E_n)$ with $V_n = \{ (b_1 .. b_n) \mid \forall_{1 \le i \le n}\ b_i = {}^{o}_{1} \}$ and $E_n = \{(b,c) \mid b, c \in V_n$ and $\forall_{2 \le i \le n}\ b_i = c_{i-1}\}$. The inverse shuffle-exchange network is the graph $\tilde{S}_n = (V_n, \tilde{E}_n)$ with $\tilde{E}_n = \{(b, c) \mid b, c \in V_n$ and $\forall_{2 \le i \le n}\ b_{i-1} = c_i\}$.

It follows that in $S_n$ a node $b_1..b_n$ is connected to $b_2..b_n\,o$ and $b_2..b_n\,1$, in $\tilde{S}_n$ to $ob_1..b_{n-1}$ and $1\,b_1..b_{n-1}$ . The fact that $S_n$ can be (uniformly) emulated on $S_{n-1}$ and, hence, on every $S_{n-k}$ ($k \ge 1$) derives from the following observation, using lemma 1.1 . (Compare [3], theorem 1.) Let $K_2$ denote the complete graph on two nodes.

Lemma 2.1. $S_n$ is a spanning subgraph of $S_{n-1}$ [ $\overline{K}_2$ ], for $n \ge 1$.

Proof.

Consider the mapping $h : S_n \rightarrow S_{n-1}$ [ $\overline{K}_2$ ] defined by $h\,(b_1..b_n) = \langle b_1..b_{n-1}, b_n \rangle$, which clearly is 1-1 and onto on the set of nodes. One easily shows that h is an embedding of $S_n$. □

Lemma 2.2. f is an emulation of $S_n$ on $S_{n-k}$ if and only if for all $x \in ({}^{o}_{1})^{n-1}$, $y \in ({}^{o}_{1})^{n-k-1}$ and $\alpha, \beta \in ({}^{o}_{1})$ : if $f\,(\alpha x) = \beta y$ then $(f(xo) = \beta y \vee f(xo) = y\,{}^{o}_{1})$ and $(f(x1) = \beta y \vee f(x1) = y\,{}^{o}_{1})$.

For a mapping f, define its "companion" $\overline{f}$ by $\overline{f}_i(b) = \overline{f_i(b)}$ for all $1 \le i \le n$.

Lemma 2.3. If f is an emulation of $S_n$ on $S_{n-k}$, then so is $\overline{f}$.

3. Uniform emulations of $S_n$ on $S_{n-1}$. The uniform emulations of $S_n$ on $S_{n-1}$ will be shown to be "step-simulating" in a very precise sense.

Definition. A mapping $g : S_n \rightarrow S_{n-1}$ is called step-simulating (or : a "step-simulation" of $S_n$ on $S_{n-1}$) if and only if for all $x \in ({}^{o}_{1})^{n-1}$, $y \in ({}^{o}_{1})^{n-2}$ and $\alpha, \beta \in {}^{o}_{1}$ : if $g\,(\alpha x) = \beta y$ then $g\,(xo) = y\,{}^{o}_{1}$ and $g\,(x1) = y\,{}^{o}_{1}$ .

$g\ (\gamma x\delta)$ for all $\gamma, x, \delta$. Hence $T^n \circ \Pi^n = id$. Conversely, let h be a step-simulation of $S_{n-1}$ on $S_{n-2}$. Then $\Pi^n \circ T^n(h)\ (\gamma x) = T^n(h)\ (\gamma x o)|_{n-2} = (h\ (\gamma x) .\ h_{n-2}\ (xo))|_{n-2} = h\ (\gamma x)$ for all $\gamma, z$. Hence also $\Pi^n \circ T^n = id$. It follows that $\Pi^n$ and $T^n$ are inverses to one another when considered as operators on step-simulations.

(iv) Let g be a uniform step-simulation of $S_n$ on $S_{n-1}$. Suppose $\Pi^n(g)$ is not uniform. Then there must be a $y \in V_{n-2}$ such that $|\ \Pi^n(g)^{-1}(y)\ | > 2$. Let $x^{(1)}, x^{(2)}, x^{(3)}$ be distinct elements of $\Pi^n(g)^{-1}(y)$. It follows that $g\ (x^{(1)}o), g\ (x^{(2)}o), g\ (x^{(3)}o) \in \{ yo, y1\}$. Because g is step-simulating we have, in fact : $g\ (x^{(1)}o), g\ (x^{(1)}1), g\ (x^{(2)}o), g\ (x^{(2)}1), g\ (x^{(3)}o), g\ (x^{(3)}1) \in \{ yo, y1\}$ and hence $|g^{-1}(yo)| \geq 3$ or $|g^{-1}(y1)| \geq 3$. This contradicts the uniformity of g. □

Theorem 3.3. (i) - (iii) shows that there is a 1-1 onto correspondence between the step-simulations of $S_n$ on $S_{n-1}$ and the step-simulations of $S_{n-1}$ on $S_{n-2}$, for $n \geq 3$. Theorem 3.3. (iv) does not quite show that this correspondence holds for the subclasses of uniform step-simulations, but in the next theorem we will show that it is the case.

<u>Theorem</u> 3.4. For $n \geq 2$,

(i) there are exactly 16 possible step-simulations of $S_n$ on $S_{n-1}$.

(ii) there are exactly 6 possible uniform step-simulations of $S_n$ on $S_{n-1}$ (see table A).

<u>Proof</u>.

(i) By theorem 3.3. (i) - (iii) the number of step-simulations of $S_n$ on $S_{n-1}$ is equal to the number of step-simulations of $S_{n-1}$ on $S_{n-2}$, for $n \geq 3$ (because $\Pi^n$ is bijective). By induction this number is equal to the number of step-simulations of $S_2$ on $S_1$. Clearly every mapping $\in [\ V_2 \rightarrow V_1\ ]$ is step-simulating. There are exactly $2^4 = 16$ mappings in this set.

(ii) There are exactly $\binom{4}{2} = 6$ mappings $\in [\ V_2 \rightarrow V_1\ ]$ that are uniform and step-simulating. By theorem 3.3. (i) - (iv) the number of uniform step-simulations of $S_n$ on $S_{n-1}$ $(n \geq 3)$ is not larger than the number of uniform step-simulations of $S_{n-1}$ on $S_{n-2}$ and thus, by induction, not larger than 6. On the other hand at least 6 uniform step-simulations of $S_n$ on $S_{n-1}$ can be explicitly given, see table A. (The verification of the mappings is immediate from the definition.) □

$f_1$ : $f_1\,(b_1..b_n) = b_1..b_{n-1}$

$\overline{f}_1$ : $\overline{f}_1\,(b_1..b_n) = \overline{b}_1..\overline{b}_{n-1}$

$f_2$ : $f_2\,(b_1..b_n) = b_2..b_n$

$\overline{f}_2$ : $\overline{f}_2\,(b_1..b_n) = \overline{b}_2..\overline{b}_n$

$f_3$ : $f_3\,(b_1..b_n) = c_1..c_{n-1}$ with $c_i = (b_i \equiv b_{i+1})$, $1 \leq i \leq n-1$

$\overline{f}_3$ : $\overline{f}_3\,(b_1..b_n) = \overline{c}_1..\overline{c}_{n-1}$ with $c_i = (b_i \equiv b_{i+1})$, $1 \leq i \leq n-1$

Table A. Listing of the 6 possible uniform step-simulations of the shuffle-exchange network with $2^n$ nodes on the shuffle-exchange network with $2^{n-1}$ nodes.

The remaining problem is to determine whether any other uniform emulations of $S_n$ on $S_{n-1}$ exist. Our main result is the following.

Theorem 3.5. (Characterisation Theorem) Every uniform emulation of $S_n$ on $S_{n-1}$ is step-simulating, and thus equal to one of the mappings listed in table A.

The proof is long and tedious, and given in [1].

4. Uniform emulations of $S_n$ on $S_{n-k}$. We will extend the notion of 'step-simulation' to emulations of $S_n$ on $S_{n-k}$, in order to attempt a characterisation of the uniform emulations in general. We show that the step-simulations of $S_n$ on $S_{n-k}$ (which are not all uniform) can again be characterized in terms of the step-simulations of $S_{k+1}$ on $S_1$ (cf. theorem 3.4). It remains an open question whether a suitable analogue of theorem 3.5 holds for $k > 1$. We show that there are at least $2.2^{2^k} - 2^{2^{k-1}}$ uniform step-simulations of $S_n$ on $S_{n-k}$.

Definition. A mapping $g : S_n \rightarrow S_{n-k}$ is called step-simulating (or: a "step-simulation" of $S_n$ on $S_{n-k}$) if and only if for all $x \in (\frac{0}{1})^{n-1}$, $y \in (\frac{0}{1})^{n-k-1}$, and $\alpha, \beta \in \frac{0}{1}$ : if $g\,(\alpha x) = \beta y$ then $g\,(x0) = y\,\frac{0}{1}$ and $g\,(x1) = y\,\frac{0}{1}$ .

Corollary 4.4. For $n \geq 1$, $S_n$ admits precisely 2 graph-isomorphisms onto itself.

Proof.

Every isomorphism of $S_n$ must be step-simulating. By theorem 4.3 (i) the step-simulations of $S_n$ on $S_n$ are in 1-1 correspondence to the step-simulations of $S_1$ on $S_1$. There are four mappings of this kind and thus precisely four step-simulations of $S_n$ on $S_n$: $g_1(b_1...b_n) = b_1...b_n$, $g_2(b_1...b_n) = \bar{b}_1...\bar{b}_n$, $g_3(b_1...b_n) = o...o$, $g_4(b_1...b_n) = 1...1$. Clearly, only $g_1$ and $g_2$ are isomorphisms. □

The 1-1 correspondence referred to in theorem 4.3 (i) can be made explicit as follows. Given a step-simulation g of $S_n$ on $S_{n-k}$, the uniquely corresponding step-simulation $\tilde{g}$ of $S_{k+1}$ on $S_1$ is defined by the formula $\tilde{g}(b_1..b_{k+1}) = g(b_1..b_{k+1}\ o..o)]_1$. Conversely, given a step-simulation h of $S_{k+1}$ on $S_1$, the uniquely corresponding step-simulation $\underset{\sim}{h}$ of $S_n$ on $S_{n-k}$ is defined by $\underset{\sim}{h}(b_1...b_n) = h(b_1..b_{k+1}).h(b_2..b_{k+2})..h(b_{n-k-1}..b_n)$. While the correspondence $g \to \tilde{g}$ preserves uniformity (cf.theorem 4.2 (iv)), it does not induce a bijection from the uniform step-simulations of $S_n$ on $S_{n-k}$ to the uniform step-simulations of $S_{k+1}$ to $S_1$ for $k > 1$. The existence of such a bijection for $k = 1$ (cf.theorem 3.4 (ii)) was the key to the complete characterisation of the uniform step-simulations of $S_n$ on $S_{n-1}$ and of the uniform emulations of $S_n$ on $S_{n-1}$ (cf.theorem 3.5). A similar characterisation of the uniform step-simulations and of the uniform emulations of $S_n$ on $S_{n-k}$ for $k > 1$ remains an open problem. We can characterize a large class of uniform step-simulations.

Theorem 4.5. Let $n \geq k+1$, and let g be a step-simulation of $S_n$ on $S_{n-k}$.

(i) if $\tilde{g}(b_1..b_{k+1}) = \overline{\tilde{g}(\bar{b}_1\ b_2..b_{k+1})}$ for all $b_1..b_{k+1} \in (\frac{o}{1})^{k-1}$, then g is uniform.

(ii) if $\tilde{g}(b_1..b_{k+1}) = \overline{\tilde{g}(b_1..b_k\ \bar{b}_{k+1})}$ for all $b_1..b_{k+1} \in (\frac{o}{1})^{k+1}$, then g is uniform.

Proof.

We only prove (1) as the proof of (ii) is similar. Induct on n. For $n = k+1$, observe from the assumption that of every pair $b_1..b_{k+1}$, $\bar{b}_1\ b_2..b_{k+1}$ $\tilde{g}$ will map one to $o \in V_1$ and one to $1 \in V_1$. Thus $g=\tilde{g}$ is uniform. Assume it holds up to $n-1 \geq k+1$. Let g be a step-simulation of $S_n$ on $S_{n-k}$ for which the constraint on $\tilde{g}$ is satisfied. Let g' be the uniquely corresponding step-simulation of $S_{n-1}$ on $S_{n-k-1}$ (cf. theorem 4.3 (i)) defined by the formula $g'(b_1..b_{n-1}) = g(b_1..b_{n-1}o)|_{n-1}$. Observe that for all $b_o..b_{n-1} \in \binom{o}{1}^n$ $g(b_o\ b_1..b_{n-1}) = \tilde{g}(b_o\ b_1..b_k).\tilde{g}(b_1..b_{k+1})...\tilde{g}(b_{n-k-2}..b_{n-1})$ and likewise for $g'(b_1..b_{n-1})$, hence $g(b_o\ b_1..b_{n-1}) = \tilde{g}(b_o\ b_1..b_k).g'(b_1..b_{n-1})$. Since $\tilde{g}' = \tilde{g}$, it follows by induction that $g'$ is uniform. Thus for every $c_1..c_{n-k-1} \in \binom{o}{1}^{n-k-1}$: $|(g')^{-1}(c_1..c_{n-k-1})| = 2^k$. Let $b_1..b_{n-1} \in (g')^{-1}(c_1..c_{n-k-1})$. By assumption it follows that of the pair $o\ b_1..b_k$, $1\ b_1..b_k$ $\tilde{g}$ will map one to $o \in V_1$ and one to $1 \in V_1$, and thus $g$ will map one of the strings $o\ b_1..b_{n-1}$, $1\ b_1..b_{n-1}$ to $o\ c_1..c_{n-k-1}$ and the other to $1\ c_1..c_{n-k-1}$. It follows that for all $c_o\ c_1..c_{n-k-1} \in \binom{o}{1}^{n-k}$: $|\ g^{-1}(c_o\ c_1..c_{n-k-1})| = |(g')^{-1}(c_1..c_{n-k-1})\ |$ $= 2^k$, which implies that $g$ is uniform. This completes the inductive argument. □

<u>Theorem</u> 4.6. For $n \geq k+1$, there are at least $2.2^{2^k} - 2^{2^{k-1}}$ uniform step-simulations of $S_n$ on $S_{n-k}$.

<u>Proof</u>.

Use the characterisation from theorem 4.5. By induction on $k$ one easily derives that there exist $2^{2^k}$ functions $\tilde{g}: V_{k+1} \rightarrow V_1$ that satisfy the constraint $\tilde{g}(b_1..b_{k+1}) = \overline{\tilde{g}(b_1\ b_2..b_{k+1})}$, $2^{2^k}$ functions $\tilde{g}: V_{k+1} \rightarrow V_1$ that satisfy the constraint $\tilde{g}(b_1..b_{k+1}) = \overline{\tilde{g}(b_1..b_k\ b_{k+1})}$, and $2^{2^{k-1}}$ functions $\tilde{g}$ that satisfy both constraints simultaneously. Using the unique correspondence of $g$ and $\tilde{g}$, the given bound follows. □

## 5. <u>References</u>.

[1] Bodlaender, H.L., and J. van Leeuwen, Simulation of large networks on smalller networks, Techn.Rep.RUU-CS-84-4, Dept. of Computer Science, University of Utrecht, Utrecht, 1984.

[2] Garey, M.R., and D.S. Johnson, Computers and Intractibility: a Guide to the Theory of NP-completeness, W.H. Freeman, San Francisco, Calif., 1979.

[3] Fishburn, J.F., and R.A. Finkel, Quotient networks, IEEE Trans.Comput., C-31 (1982) 288-295.

[4] Harary, F., Graph theory, Addison-Wesley Publ. Comp., Reading, Mass., 1969.

[5] Herman, G.T., When is a sequential machine the realisation of another, Math.Syst.Th. 5(1971) 115-127.

[6] Stone, H.S., Parallel processing with the perfect shuffle, IEEE Trans. Comput. C-20(1971) 153-161.