SIMULATION OF LARGE NETWORKS ON SMALLER NETWORKS
(Extended Abstract)

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 [2] the notion of network emulation was proposed, to obtain a structure preserving simulation of large networks on smaller networks. We present a detailed analysis of the possible emulations for some important classes of networks.

1. Introduction. Parallel algorithms are normally designed for execution on a suitable network of N processors (viewed as SIMD- or MIMD-machine [6]), 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 this paper we study a notion of simulation, termed: emulation, that was recently proposed by Fishburn and Finkel [2].

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 → V_H such that for every edge (g, g') ∈ E_G : f(g) = f(g') or (f(g), f(g')) ∈ E_H. The function f is called an emulation function or, in short, an emulation of G on H.

Let f be an emulation of G on H. Any processor h ∈ V_H must actively emulate the processors ∈ f^{-1}(h) in G. When g ∈ f^{-1}(h) communicates information to a neighbouring processor g', then h must communicate the corresponding information "internally".

* 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).
when it emulates \( g' \) itself or to a neighbouring processor \( h' = f(g') \) in \( H \) otherwise.

**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 \( G \). When \( |V_G| = |V_H| \), then \( G \) can be emulated on \( H \) if and only if \( G \) is isomorphic to a subgraph of \( H \).

With this observation it is not hard to show that the general **UNIFORM NETWORK EMULATION** problem is NP-complete (cf. [3], reduce from **SUBGRAPH ISOMORPHISM**).

The relevant question is whether (large) networks of some class \( C \) can be uniformly emulated by networks of a smaller size within the same class \( C \). Fishburn and Finkel [2] answered this question affirmatively for the following classes of processor networks: the shuffle-exchange network, the grid-connected network, the \( n \)-dimensional cube, the plus-minus network, the binary lens, and the cube-connected cycles. In this paper we develop a detailed analysis of all possible emulations in selected classes of networks.

2. **Emulations of the shuffle-exchange network.**

2.1. **Preliminaries.** The shuffle-exchange network was proposed initially by Stone [5]. The nodes are given \( n \)-bit addresses in the range \( 0..2^n-1 \), and there is an edge from node \( b \) to node \( c \) if and only if \( b \) can be "shuffled" (move leading bit to tail position) and "exchanged" (flip the tail bit) into \( c \). We use the following notations throughout:

\[
\begin{align*}
\overline{0} & : \text{a bit that can be 0 or 1} \\
\overline{\alpha} & : \text{the complement of bit } \alpha (\overline{0} = 1, \overline{1} = 0) \\
b & : \text{the } n\text{-bit address } b_1..b_n \\
b|_i & : b_1..b_i \text{ (truncation after the } i\text{th bit)} \\
i\overline{b} & : b_i..b_n \text{ (truncation "before" the } i\text{th bit)} \\
(b)_i & : b_i \text{ (the } i\text{th bit)}.
\end{align*}
\]

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 \( V_n = \{(b_1..b_n) | V_1 \leq i \leq n b_i = 0\} \) and \( E_n = \{(b,c) | b,c \in V_n \text{ and } \forall 2i \leq n b_i = c_{i-1}\} \).