Acta Informatica 19, 269–296 (1983)

Acta Informatica

Efficiency of Universal Parallel Computers

Friedhelm Meyer auf der Heide

Johann-Wolfgang-Goethe-Universität Frankfurt, Fachbereich Informatik, D-6000 Frankfurt am Main (Fed. Rep.)

Summary. We consider parallel computers (PC's) with fixed communication network and bounded degree. We deal with the following question: How efficiently can one PC, a so-called universal PC, simulate each PC with n processors? This question is asked in [1] where a universal PC with $O(n)$ processors and time loss $O(\log(n))$ is constructed. We improve this result in two ways by construction two universal PC's which many users can efficiently work with at the same time. The first has the same number of processors and the same time loss as that one above. The second has $O(n^{1+\varepsilon})$ processors for an arbitrary $\varepsilon > 0$ but only time loss $O(\log\log(n))$. Finally we define three types of simulations the most general of which includes all known simulations. We prove non-linear time-processor trade-offs for universal PC's associated with the above types.

Introduction

This paper deals with parallel computers. The model of parallel computation we use is essentially due to Paul and Galil (see [2]). In this sense, a *parallel computer (PC) M* is specified by a finite graph with bounded degree and by processors which are attached to the vertices of the graph. These processors are random access machines (see [3] or [1]).

Such a PC works as follows: In the beginning of a computation, some special processors, the input-processors, contain the input. In one step all processors execute at the same time one of the usual instructions for random access machines or read the content of some fixed register of a processor which is (relative to the graph) one of its neighbours. At the end of the computation, some processors, the output-processors, contain the output.

A multi-purpose PC(MPC) is a PC whose processors are universal random access machines. A *program* of a MPC consists of programs for all its processors. Paul and Galil asked in [1] the following question which will be the subject of this paper:

How efficiently can one MPC simulate all other PC's? We measure the efficiency of such a simulation by the number of processors of the MPC and the number of steps the MPC needs, relative to the PC being simulated. A MPC which can simulated all PC's from a certain set H of PC's is called *universal for* H. The exact definitions of the above terms can be found in Chap. 1, a discussion of our model of parallel computation in [2].

Paul and Galil constructed a universal MPC for all PC's with n processors which itself has n processors, too, and which has a time-loss $O(\log(n))$, that means, which is by a factor $O(\log(n))$ slower than the PC being simulated.

If several users want to work with this MPC, that means, if several small PC's shall be simulated, the time-loss remains $O(\log(n))$ although these PC's have much less than n processors.

The MPC used in [2] are the Cube-Connected Cycles which Preparata and Vuillemin introduced in [4].

In Chap. 2 we generalize this MPC in such a way that we obtain a universal MPC which can for each $n' \leqq n$ simulate $\left\lfloor \frac{n}{n'} \right\rfloor$ PC's with n' processors each with a time-loss $O(\log(n'))$.

In Chap. 3 we construct a universal MPC which can simulate any set of PC's which together only have n processors. The simulation of a PC with $n' \leqq n$ processors only has a time-loss of $O(\log\log(n'))$ and this universal MPC has $O(n^{1+\varepsilon})$ processors for an arbitrary $\varepsilon > 0$.

The remaining three chapters contain lower bounds for the efficiency of universal MPC's.

For this purpose we define three types of simulations. We suppose that at each step of a simulation of a PC M by a universal MPC M_0 each processor of M is simulated by at least one processor of M_0, its representant(s). The communication between processors is simulated by transporting the corresponding informations along pathes in M_0. The time-loss then depends on the lengthes of such pathes.

In Chap. 4 we present the following types of simulations:

Type 1. Each processor of M is simulated by one processor of M_0.

The universal MPC of Chap. 2 is of this type. Also the simulation of M by M_0 (M and M_0 are described in Fig. 1) is of type 1, if for each $i \in \{1, 2, 3\}$ P_i is simulated by Q_{2i}.

The time-loss of this simulation is 2.

Now let for $i \in \{1, 2, 3\}$ P_i be simulated by Q_i and Q_{i+3}. Then obviously we get a simulation of M by M_0 with time loss 1, because the neighbours of each representant of some P_i are representants of the neighbours of P_i. Generalizing this kind of simulation we obtain:

Type 2. Each processor of M is simulated by at least one processor of M_0.

The simulations which are used by the universal MPC of Chap. 3 do not belong to one of the defined types. Those simulations allow that the representants of some processor of M may vary dependent on the number of steps of M being already simulated. The following type of simulations also includes the universal MPC of Chap. 3:

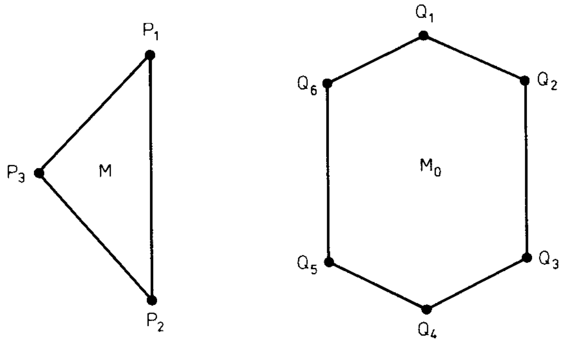

Fig. 1

Type 3. Each processor of M is at each time of the simulation simulated by at least one processor of M_0.

In what follows, a universal MPC which only uses simulations of type i, $i \in \{1, 2, 3\}$ is called universal of type i. Let M_0 be universal for all PC's with n processors. M_0 has m processors and the maximal time-loss of some simulation M_0 executes let be k. In Chap. 5 we consider a family of graphs which we call uniform distributors. The graph of the universal MPC from Chap. 3 belongs to this family. We prove:

If M_0 is universal of type 3 and its graph is a uniform distributor, then $k = \Omega(\log\log(n))$.

This bound is proved to be asymptotically tied in Chap. 3. In Chap. 6 we prove time-processor trade-offs for universal parallel computers.

- If M_0 is universal of type 1, than $k = \Omega(\log(n))$ or $m = n^{\Omega(n)}$.
- If M_0 is universal of type 2, than $m \cdot k = \Omega(n \log(n))$.
- If M_0 is universal of type 3, then $m \cdot k = \Omega(n \log(n)/\log\log(n))$.

The first trade-off tells us that a universal MPC of type 1 which is asymptotically faster than thatone constructed in Chap. 2 must have an exponential size.

The second trade-off is proved in Chap. 2 to be asymptotically tied.

The third trade-off is not proved to be tied but it shows that also with the help of the very general type 3 of simulations it is impossible to construct a universal MPC without a significant loss of efficiency.

Chapter 1: Definitions

A parallel computer (PC) M is specified by a tupel (G, I, C, PS).

G is a finite graph, I and C are two injective sequences of vertices of G, PS is a set of processors which contains a processor P_i for each vertex i of G. P_i and P_j are neighbours, if the vertices i and j of G are neighbours in G. A processor is a random access machine with the following modifications:

– Let i be the j'th member of I, $j\in[0, \#I-1]$[1]. Then P_i is the j'th input-processor and has an input- but no output-tape.
– Let i be the j'th member of $\mathcal{O}$, $j\in[0, \#\mathcal{O}-1]$.
Then P_i is the j'th *output-processor* of M and has an output- but no input-tape.
– If i is neither a member of I nor of $\mathcal{O}$, then P_i has neither an input- nor an output-tape.
– Each processor has a special register, its *communication register*, and is able to read in one step the content of the communication register of one of its neighbours.

M works as follows: In the beginning of the computation, each input-tape contains a tupel from $\mathbb{N}^*(:=\bigcup_{n\geqq 0}\mathbb{N}^n)$. (The input-processors are able to read one integer from their input-tapes in one step.)

In one step of the computation, all processors execute at the same time one instruction of their programs. M stops, if all output-processors have stopped.

Suppose M has n input- and m output-processors. If in the beginning of the computation, $x_j\in\mathbb{N}^*$ is written on the j'th input-tape, $j\in[0,n-1]$, and in the end, $y_j\in\mathbb{N}^*$, $j\in[0,m-1]$, is written on the j'th output-tape, then we say, M started with $\bar{x}=(x_0,\dots,x_{n-1})$ computes $\bar{y}=(y_0,\dots,y_{m-1})$. $\bar{x}\in(\mathbb{N}^*)^n$ is an input, $\bar{y}\in(\mathbb{N}^*)^m$ an output for M.

The measure for the time-complexity of M started with $\bar{x}$ is the number of steps, M executes. The size of M is the number of processors of M. The degree of M is the degree of the graph G of M. We only consider families of PC's, whose degree does not grow with its size.

A multi-purpose PC (MPC) is a PC whose processors are universal random access machines. A program for such a MPC is given by programs for each of its processors. As the set of processors of a MPC is completely defined by this property, we denote it by the tupel $(G, I, \mathcal{O})$ for short.

The task we want to set to a MPC $M_0=(G,I,\mathcal{O})$ is the following: r users $B_1,\dots,B_r$ want to let M_0 simulate their PC's $M_1,\dots,M_r$ at the same time. Each of them gets a sequence of relative to I and $\mathcal{O}$ consecutive input- and output-processors. Each user B_i writes a coding of his PC M_i on the input-tapes of his input-processors. With the help of these codings, M_0 becomes initialized such that the following holds:

If each user B_i writes an input $\bar{x}_i$ for his PC M_i on his input-tapes, then M_0 stops with the output on the output-tapes for B_i, which is computed by M_i started with $\bar{x}_i$.

If M_0 can fullfil this task, we say, M_0 can simulate $(M_1,\dots,M_r)$. Let H be a set of finite tupels of PC's such that M_0 can simulate each tupel from H. *Then* M_0 is called *universal for H*. We are not interested in the time which is needed for initializing M_0 but only in the time which is necessary after the initialization to compute the output. More exactly, we denote by the *simulation-time* of M_0 for some PC M started with some input $\bar{x}$ the maximal number of steps which M_0 needs to compute the output of M started with $\bar{x}$ if

[1] For a sequence (set) A, $\#A$ denotes the length (cardinality) of A. $\mathbb{N}$ denotes the set of all non-negative integers. For some $a,b\in\mathbb{N}$, $a\leqq b$, the intervall $[a,b]$ is defined as $\{x\in\mathbb{N},\ a\leqq x\leqq b\}$

– M_0 is initialized for some tupel from H which contains M,
– the user B who wants to let M_0 simulate M writes the input $\bar{x}$ on his input-tapes, and
– all other users write any inputs for their PC's on their input-tapes.

Chapter 2

In this chapter, we shall for each $n=2^{2^k}\cdot 2^k$ for some $k\in\mathbb{N}$ construct a universal MPC which can for every $n'\leqq n$ be used to simulate $\left\lfloor\frac{n}{n'}\right\rfloor^2$ PC's of size n'. The simulation-time of some PC M is by a factor $O(\log(n'))$ slower than the time M itself needs. This MPC has $3n$ processors and is an *n-permuter*.

By an *n-permuter* we mean a MPC $M=(G, I, \mathcal{O})$ with the following properties:
– $I=\mathcal{O}$. This sequence is called the *base B of M*. B has the length n.
– For each permutation π on $[0, n-1]$ M can be initialized such that M started with $(x_0, \ldots, x_{n-1})\in\mathbb{N}^n$ computes $(x_{\pi(0)}, \ldots, x_{\pi(n-1)})$. We say, *M permutes* $(x_0, \ldots, x_{n-1})$ *according to* π. We specify M by (G, B). For some $n'\leqq n$ let $m:=\left\lfloor\frac{n}{n'}\right\rfloor$ and $\pi_1, \ldots, \pi_m$ be permutations on $[0, n'-1]$. Let π be the following permutation on $[0, n-1]$:

For $i\in[1, m]$, $j\in[(i-1)n', in'-1]$, $\pi(j):=\pi_i(j-(i-1)n')+(i-1)n'$.

For $j\in[n'm, n-1]$, $\pi(j):=j$.

If M can permute $(x_0, \ldots, x_{n-1})$ according to π, then we say, M can permute $(x_0, \ldots, x_{n-1})$ according to $(\pi_1, \ldots, \pi_m)$.

Now let $n:=2^{2^k}\cdot 2^k$ for some $k\in\mathbb{N}$. We shall construct an n-permuter which can for every $n'\leqq n$ permute $(x_0, \ldots, x_{n-1})\in\mathbb{N}^n$ according to $\left\lfloor\frac{n}{n'}\right\rfloor$ arbitrary permutations on $[0, n'-1]$ in $O(\log(n'))$ steps. This construction is based on the MPC which was introduced by Preparata and Vuillemin in [4]: the Cube-Connected Cycles.

2.1. *Definition* (Preparata, Vuillemin). The graph $C_k^1=(V, E)$ is defined as follows:

$$V:=[0,1]^{2^k}\times[0, 2^k-1],$$

$$E:=E_1\cup E_2.$$

A pair

$$e=\{(a_0, \ldots, a_{2^k-1}, p), (b_0, \ldots, b_{2^k-1}, q)\}\subset V$$

is in E_1, iff $(a_0, \ldots, a_{2^k-1})=(b_0, \ldots, b_{2^k-1})$ and $q=(p+1)\bmod(2^k)$ or $q=(p-1)\bmod(2^k)$.

e lies in E_2, iff $p=q$ and $(a_0, \ldots, a_{2^k-1})$, $(b_0, \ldots, b_{2^k-1})$ differ exactly at the p'th position.

[2] For some real number x, $\lfloor x\rfloor$ ($\lceil x\rceil$) denotes the largest (smallest) integer less (greater) or equal to x

2.2. *Remark.* C_k^1 has $n=2^{2^k}\cdot 2^k$ vertices and degree 3. For some fixed $(a_0,\ldots,a_{2^k-1})\in[0,1]^{2^k}$, the vertices $\{(a_0,\ldots,a_{2^k-1})\}\times[0,1]^{2^k}$ form a cycle of length 2^k with the help of edges from E_1. The edges from E_2 join these circles in such a way, that a cube is built whose vertices are the cycles. Let B be any injective sequence consisting of all elements of V.

2.3. *Definition* (Preparata, Vuillemin). The MPC $V_k^1=(C_k^1, B)$ is called *the Cube-Connected Cycles*. In [4] it is proved:

2.4. Theorem. *V_k^1 is an n-permuter. For every permutation π on $[0,n-1]$, it can permute n numbers according to π in $O(\log(n))$ steps.*

Now consider the following graph:

2.5. *Definition.* C_k^2 is the graph which results from C_k^1 by inducing the following additional edges:

If $\bar{a}\in[0,1]^{2^k}$, $p,p'\in[0,2^k-1]$ and $p+p'\in\{2^{k-1}-1, 3\cdot 2^{k-1}-1\}$, then $\{(\bar{a},p),(\bar{a},p')\}$ is an edge in C_k^2. A subgraph of C_k^2 which is induced by the vertices $\{\bar{a}\}\times[0,2^k-1]$ is shown in Fig. 2.

For some $\bar{b}=(b_0,\ldots,b_{2^k-1})\in[0,1]^{2^k}$ let $\bar{b}^1:=(b_0,\ldots,b_{2^{k-1}-1})$ and $\bar{b}^2:=(b_{2^{k-1}},\ldots,b_{2^k-1})$.

Now consider for some $\bar{a}\in[0,1]^{2^{k-1}}$ the following subsets of V:

$$D_{\bar{a}}^1:=\{(\bar{b},p)\in V\,|\,\bar{a}=\bar{b}^1,\ p\in[2^{k-1},2^k-1]\},$$
$$D_{\bar{a}}^2:=\{(\bar{b},p)\in V\,|\,\bar{a}=\bar{b}^2,\ p\in[0,2^{k-1}-1]\}.$$

We define the following mappings

$$f_1:\ D_{\bar{a}}^1\to[0,1]^{2^{k-1}}\times[0,2^{k-1}-1]$$

and

$$f_2:\ D_{\bar{a}}^2\to[0,1]^{2^{k-1}}\times[0,2^{k-1}-1].$$

Let for $\bar{b}\in[0,1]^{2^k}$

$$h_1(\bar{b}):=(b_{2^{k-1}+2^{k-3}},\ldots,b_{2^k-1},b_{2^{k-1}},\ldots,b_{2^{k-1}+2^{k-3}-1}),$$

and

$$h_2(\bar{b}):=(b_{2^{k-3}},\ldots,b_{2^{k-1}-1},b_0,\ldots,b_{2^{k-3}-1}),$$

then,

$$f_1(\bar{b},p):=(h_1(\bar{b}),(p+2^{k-3})\bmod(2^{k-1}))$$

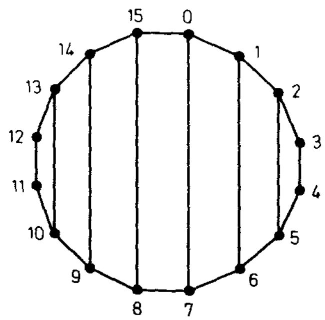

Fig. 2. A modified cycle in C_4^2

and

$$f_2(\bar{b}, p) := (h_2(\bar{b}), (p + 2^{k-3}) \bmod (2^{k-1})).$$

It can easily be verified that $f_1[f_2]$ is an isomorphism between the subgraph of C_k^2 being induced by $D_{\bar{a}}^1[D_{\bar{a}}^2]$ and C_{k-1}^2. As the sets $D_{\bar{a}}^1, D_{\bar{a}}^2$, $a \in [0,1]^{2k-1}$ form a disjoint partition of the vertex set V of C_k^2, we obtain inductively:

2.6. Lemma. *For each $r \leqq k$, C_k^2 can be partitioned in $n/2^{2r}2^r$ pairwise disjoint graphs which are isomorphic to C_r^2.*

In order to construct an n-permuter with graph C_k^2, we recursively define a base B_k^2 for it in which the vertices which belong to one subgraph being isomorphic to some C_r^2 are consecutive.

$$B_0^2 = ((0,0), (1,0)).$$

Let $k > 0$.

Then we may assume that for each of the subgraphs $G_1, \ldots, G_m$, $m = n/2^{2k-1} \cdot 2^{k-1}$, which form the disjoint partition of G from Lemma 2.6, such a base is already defined. Let them be called $B(G_1), \ldots, B(G_m)$. (The order of these subgraphs may be arbitrary.) Then $B_k^2 := (B(G_1), \ldots, B(G_m))$. With the help of Theorem 2.4 we now get:

2.7. Lemma. *The MPC $V_k^2 := (C_k^2, B_k^2)$ is an n-permuter. For every $r \leqq k$ and $n' := 2^{2r} \cdot 2^r$ it can permute n arbitrary numbers according to $\frac{n}{n'}$ arbitrary permutations on $[0, n'-1]$ in $O(\log(n'))$ steps.*

Now we shall construct a MPC V_k for which the statement of the lemma holds even if n' is not of the form $2^{2r} \cdot 2^r$.

2.8. Definition. Let (C_1^*, B_1^*), (C_2^*, B_2^*) be two examplaries of V_k^2, and let $B_k := (d_0, \ldots, d_{n-1})$ and $a \in \mathbb{N}$, $a \leqq n$.

Then $C_{k,a}$ is the graph which results from C_1^*, C_2^* and B_k by adding edges for every $i \in [0, n-1]$ from d_i to the i'th element of B_1^* and to the $((i+a) \bmod (n))$'th element of B_2^*. $V_{k,a}$ is the MPC $(C_{k,a}, B_k)$.

We now shall find an a such that for any $n' \leqq n$ holds: the neighbours of n' consecutive elements of the base B_k of $V_{k,a}$ either in C_1^* or in C_2^* are completely contained in a subgraph being isomorphic to a "small" C_q^2. "small" means that $\log(n') = O(\log(2^{2^q} 2^q))$.

For some $p, x \in \mathbb{N}$, let the intervall $[xp, (x+1)p-1]$ be called a p-intervall. For some set $E \subset \mathbb{N}$ and some $s, a \in \mathbb{N}$ let $E + a((E+a) \bmod (s))$ be the set $\{y + a, y \in E\}$ $((y+a) \bmod (s), y \in E)$.

2.9. Lemma. *Let $b_0, \ldots, b_k \in \mathbb{N}$, $b_0 = 1$, such that for each $i \in [0, k-1]$ it holds: $3 \leqq d_i := \frac{b_{i+1}}{b_i} \in \mathbb{N}$. Let $a := \sum_{i=0}^{k-1} b_i$, then for every $p \in [0, b_k - 1]$, $q \in [1, k]$ with $b_{q-1} < p \leqq b_q$ and each interval $E \subset [0, b_k - 1]$ with length p it holds: E or $(E+a) \bmod (b_k)$ is completely contained in a b_{q+1}-intervall.*

Proof. Each $x \in [0, b_k - 1]$ can be represented as $\sum_{i=0}^{k-1} c_i b_i$ with $c_i \in [0, d_i - 1]$ for $i \in [0, k-1]$.

Let for $i \in [0, k-1]$ $c_i, c_i^1, c_i^2, c_i^3, c_i^4 \in [0, d_i - 1]$. Let p, q, E be as defined in the lemma. If E is completely contained in a b_{q+1}-intervall, we are ready. Otherwise E contains an x with the representation $x = \sum_{i=q+1}^{k-1} c_i b_i$. Let $x_1 := x - b_q$, $x_2 := x + b_q$. Then $E \subset [x_1, x_2 - 1]$ and $x_1 \geqq 0$, $x_2 \leqq b_k$, because otherwise E would be contained in a b_{q+1}-intervall.

Obviously x_1 and x_2 have the following representations:

$$x_1 = \sum_{i=q}^{k-1} c_i^1 b_i \quad \text{with } c_q^1 = d_q - 1$$

and

$$x_2 = \sum_{i=q}^{k-1} c_i^2 b_i \quad \text{with } c_q^2 = 1.$$

Therefore, each $y \in [x_1, x_2 - 1]$ has the representation

$$y = \sum_{i=0}^{k-1} c_i^3 b_i \quad \text{with } c_q^3 \in \{0, d_q - 1\}.$$

Claim. $y + a$ can not be divided by b_{q+1}.

Proof. Suppose $y + a$ can be divided by b_{q+1}, then $c_0^3 = d_0 - 1$, $c_i^3 = d_i - 2$ for all $i \in [1, q]$.

Especially, $c_q^3 = d_q - 2 \notin [0, d_q - 1]$, because $d_q \geqq 3$. This contradicts the above representation of y.

Now we have that $[x_1, x_2 - 1] + a$ is completely contained in a b_{q+1}-intervall. As $E \subset [x_1, x_2 - 1]$, it follows that $E + a$ and therefore $(E + a) \bmod (b_k)$ is completely contained in a b_{q+1}-intervall. □

Now choose $b_0 = 1$, $b_i = 2^{2^i} \cdot 2^i$ for $i \in [1, k]$ and $a = \sum_{i=0}^{k-1} b_i$. From Lemma 2.9 it follows that the neighbours of the elements of each intervall E of consecutive elements of the base B_k of $V_{k,a}$ either in C_1^* or C_2^* are completely contained in a MPC V_{q+1}^2, where q is defined by $b_{q-1} < \# E \leqq b_q$.

As $\log(\# E) = O(\log(b_{q+1}))$, we obtain:

2.10. Theorem. $V_k := V_{k,a}$ *is a n-permuter with size* $3n$ *and degree* 5, *which can for every* $n' \leqq n$ *permute n numbers according to* $\left\lfloor \frac{n}{n'} \right\rfloor$ *arbitrary permutations on* $[0, n'-1]$ *in* $O(\log(n'))$ *steps.* (The claims about the size and the degree of V_k follow from Remark 2.2 and the Definitions 2.5 and 2.8.)

Now the same algorithm as Paul and Galil used in [2] to construct a universal MPC for all PC's with n processors and fixed degree c (independent on n) yields the following theorem.

2.11. Theorem. *Let* $\bar{H}_n$ *be the set of all tupels of PC's with equal size and degree c, which together have at most n processors. Then* V_k *is universal for* H_n.

Let M be a PC which appears in at least one tupel of $\bar{H}_n$ *and has the size* $n' \leqq n$. *If M started with some input* $\bar{x}$ *executes t steps to compute the output, the simulation-time of* V_k *for M started with* $\bar{x}$ *is* $O(t \cdot \log(n'))$.

Thus we have achieved that the simulation time for a PC M does no longer depend on the size of the universal PC but only on thatone of M itself.

Chapter 3

In this chapter we construct for each $n \in \mathbb{N}$ a universal MPC which can simulate all tupels of PC's with degree c, which together have at most n processors. The simulation-time of this MPC for a PC of size $n' \leqq n$ with degree c is only by a factor $O((\log\log(n'))$ slower than the PC itself. The MPC has size $O(n^{1+\varepsilon})$ for some arbitrary $\varepsilon > 0$ and is a n-distributor.

An *n-distributor* is a MPC $M_0 = (G, I, \mathcal{O})$ with $I = \mathcal{O} = (0, 1, \ldots, n-1)$. This sequence is called the base B of M_0. For arbitrary, pairwise disjoint subsets $A_0, \ldots, A_{n-1}$ of $[0, n-1]$, (some of them may be empty,) it is possible to initialize M_0 in such a way that M_0 started with some tupel $(x_0, \ldots, x_{n-1}) \in (\mathbb{N}^*)^n$ (each x_i may be a tupel of integers!) computes a tupel $(y_0, \ldots, y_{n-1})$ such that $y_i = x_j$, if $i \in A_j$, $i, j \in [0, n-1]$. Thus, for $j \in [0, n-1]$, x_j is transported to all processors of A_j. We then say, M_0 *distributes* $(x_0, \ldots, x_{n-1})$ *according to* $A_0, \ldots, A_{n-1}$.

We specify M_0 by (G, B). (If each A_i has one element, M_0 permutes the input.) We now shall construct a family $\{W_n, n \in \mathbb{N}\}$ of n-distributors such that W_n can distribute some input $(x_0, \ldots, x_{n-1}) \in (\mathbb{N}^*)^n$ according to arbitrary sets $A_0, \ldots, A_{n-1}$ in $O(\log(n) + s)$ steps, if each x_i has the length at most s. Furthermore, for arbitrary numbers $r, n_1, \ldots, n_r \in \mathbb{N}$, $\sum_{i=1}^{r} n_i \leqq n$, W_n can be disjointly partitioned into r distributors $W_{n_1}, \ldots, W_{n_r}$. Thus, r users can use W_n at the same time as n_i-distributors W_{n_i}, $i \in [1, r]$.

The construction of these MPC's is based on so called Waksman-permutation-networks, which are introduced by Waksman in [5].

3.1. Definition (Waksman). The family $(G_k^*, k \in \mathbb{N})$ of graphs is defined as follows:

– G_1^* consists of two isolated vertices 0 and 1. $(0, 1)$ is the base and the sequence of tops of G_1^*.

– For $k > 1$, G_k^* results from two exemplaries $\bar{G}_1$ and $\bar{G}_2$ of G_{k-1}^* and a sequence of 2^k new tops.

For $i \in [0, 2^{k-1} - 1]$, the i'th of these tops is joint with the i'th top of $\bar{G}_1$ and the $(i + 2^{k-1})$'th top of $\bar{G}_2$. For $i \in [2^{k-1}, 2^k - 1]$, the i'th of these tops is joint with the $(i - 2^{k-1})$'th top of $\bar{G}_1$ and the i'th top of $\bar{G}_2$.

The concatination of the bases of $\bar{G}_1$ and $\bar{G}_2$ form the base of G_k^*. The graph G_k^1 which is constructed in [5] is the following: G_k^1 consists of two exemplaries H_1 and H_2 of G_k^*. Additionally, there are edges for every $i \in [0, n-1]$ between the i-th vertices of the bases of H_1 and H_2 and the i'th vertex of $H_1(H_2)$ and the $(i+1)$'th vertex of $H_2(H_1)$, if i is even (odd). The tops of H_1 are the sources, those of H_2 the sinks of G_k^1. The two exemplaries of G_{k-1}^* in H_1 are called B_1 and B_2, those in H_2 B_3 and B_4.

The construction of G_k^1 is illustrated in Fig. 3.

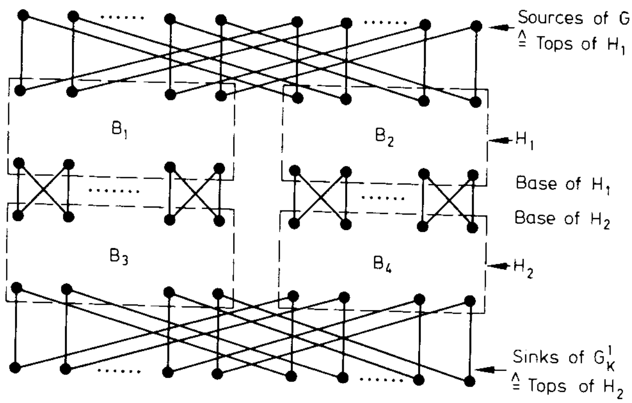

Fig. 3. The graph G_k^1

In the following construction, the base of H_1 will become the base of a distributor. Waksman proved in [3]:

3.2. Theorem (Waksman). *For each $k \in \mathbb{N}$, G_k^1 is a 2^k-permutation-network, i.e. G_k^1 contains for each permutation π on $[0, 2^k - 1]$ 2^k pairwise disjoint pathes of length $2k-1$, such that the i'th of these pathes join the i'th source and the $\pi(i)$'th sink of G_k^1, $i \in [0, 2^k - 1]$.*

This theorem shows that the MPC with graph G_k^1 whose input-processors belong to the sources and whose output-processors to the sinks of G_k^1 can be initialized for each permutation π on $[0, 2^k - 1]$, such that this MPC started with $(x_0, \ldots, x_{2^k-1}) \in (\mathbb{N}^*)^{2^k}$ computes $(x_{\pi(0)}, \ldots, x_{\pi(2^k-1)})$. If s is the maximal length of the x_i's, G_k^1 needs $O(k+s)$ steps. Now let us consider the following graph:

3.3. Definition. Let $n \in [2^{k-1}+1, 2^k]$. Then G_n^2 is the subgraph of G_k^1 which is induced by the first n sources of G_k^1 and by B_1 (compare Fig. 3). Additionally, it contains edges for each $i \in [0, 2^{k-1}-2]$ between the i'th and the $(i+1)$'th vertex of the base of B_1.

This graph is illustrated in Fig. 4.

Now let π be a permutation on $[0, n-1]$ and $R'_0, \ldots, R'_{n-1}$ the first n of the 2^k pathes in G_k^1, given by Theorem 3.2 for a permutation on $[0, 2^k - 1]$ whose restriction on $[0, n-1]$ is π.

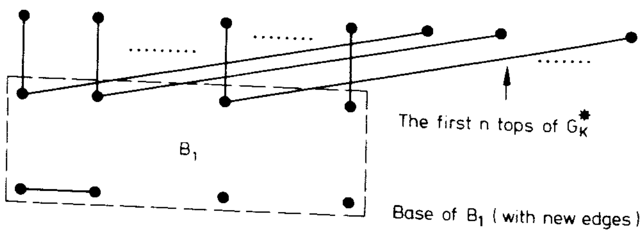

Fig. 4. The graph G_n^2

Now we subdivide for each $i\in[0,n-1]$ the path R_i' into four parts a_i, S_i', T_i', b_i.

a_i is the first vertex of R_i', S_i' the part which lies in H_1, T_i' thatone in H_2 and b_i is the last vertex of R_i'. Now we draw the pathes S_i', T_i' (which are pathes in an exemplary of B_{k-1}^*!) into B_1 and call them S_i and T_i. Now it is obvious that there is a path R_i for each $i\in[0,n-1]$ in G_n^2, which joins the i'th and the $\pi(i)$'th source of G_n^2 and which only uses vertices from S_i and T_i in B_1. (The new edges in the base of G_n^2 play the role of the edges between H_1 and H_2 in G_k^1.) As the pathes $R_0',\ldots,R_{n-1}'$ are pairwise disjoint each vertex of G_n^2 is contained in at most 4 pathes from $R_0,\ldots,R_{n-1}$.

Thus we obtain:

3.4. Lemma. *For each permutation π on $[0,n-1]$ there are pathes $R_0,\ldots,R_{n-1}$ in G_n^2 such that for each $i\in[0,n-1]$ the path R_i joins the i'th and the $\pi(i)$'th source of G_n^2. Each vertex of G_n^2 belongs to at most 4 of these pathes. Each of these pathes has the length at most $2\lceil\log(n)\rceil-1$.*

Now consider the following graph:

3.5. *Definition.* G_n is the graph with vertex set $V=\{c_{ij},\ i\in[0,n-1], j\in[0,\lceil\log(n)\rceil-1]\}$. $\{c_{ij},c_{i'j'}\}\subset V$, $j\leqq j'$, is an edge of G_n, if $j'=j+1$ and either $i=i'$ or $|i-i'|=2^{j'}$, or if $j=j'=0$ and $|i-i'|=1$. The MPC W_n is specified by (G_n,B_n) where $B_n:=(c_{i0},\ i\in[0,n-1])$. Figure 5 illustrates this graph.

3.6. *Remark.* G_n^2 is a subgraph of G_n in which for $i\in[0,n-1]$ $c_{i,\lceil\log(n)\rceil-1}$ is the i'th source of G_n^2.

As we have chosen the first n vertices of the base of H_1 (see Fig. 3) instead of its sources as the base of G_n^2, we have achieved that the subgraph $G_n^{a,b}$ of G_n^2 which is for some $0\leqq a\leqq b\leqq n-1$ induced by the vertices $\{c_{ij},\ i\in[a,b], j\in[0,\lceil\log(b-a+1)\rceil-1]\}$ is isomorphic to G_{b-a+1}^2 in such a way that its base $B_n^{a,b}:=(c_{i0},\ i\in[a,b])$ is isomorphic to the base of G_{b-a-1}^2 and consists of consecutive vertices of the base of G_n^2. Now we are ready to prove:

3.7. Theorem. *W_n is an n-distributor with the following properties:*

E1: W_n has $n\lceil\log(n)\rceil$ vertices and degree 6.

E2: For $a,b\in[0,n-1]$, $a\leqq b$, the MPC $(G_n^{a,b},B_n^{a,b})$ is an exemplary of W_{b-a-1}, where $B_n^{a,b}$ consists of consecutive vertices of B_n.

E3: Let $A_0,\ldots,A_{n-1}$ be pairwise disjoint subsets of $[0,n-1]$ and let $(x_0,\ldots,x_{n-1})\in(\mathbb{N}^*)^n$ such that each x_i has the length at most s, then W_n can distribute $(x_0,\ldots,x_{n-1})$ according to $A_0,\ldots,A_{n-1}$ in $O(\log(n)+s)$ steps.

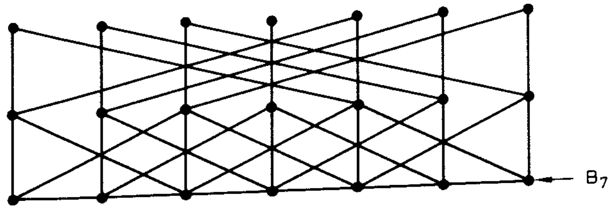

Fig. 5. The graph G_7

Proof. *E1* and *E2* follow from the Definition 3.5. In order to prove *E3*, let $A_0, \ldots, A_{n-1}$ be fixed and let $i_1 < \ldots < i_p$ chosen such that $\{i_1, \ldots, i_p\} = \{j \in [0, n-1], A_j \neq \emptyset\}$.

Let $s_1 := 0$ and $s_j = \sum_{l=1}^{j-1} \# A_{i_l}$ for $j \in [2, p+1]$.

Now we consider two permutations π and π' on $[0, n-1]$ with the following properties:

$$\pi(i_j) = s_j \quad \text{for } j \in [1, p],$$

$$\pi'([s_j, s_{j+1} - 1]) = A_{i_j} \quad \text{for all } j \in [1, p].$$

As $[s_j, s_{j+1} - 1] = s_{j+1} - s_j = \# A_{i_j}$, such a π' exists.

Now let $R_0, \ldots, R_{n-1}$ and $R'_0, \ldots, R'_{n-1}$ be the pathes given by Lemma 3.4 for π and π'.

Now an input $(x_0, \ldots, x_{n-1}) \in (\mathbb{N}^*)^n$ is distributed as follows:

1) For all $i \in [0, n-1]$, transport x_i from the i'th vertex of the base to the i'th source.

2) For all $i \in [0, n-1]$, transport x_i from the i'th source to the $\pi(i)$'th source along the pathes $R_0, \ldots, R_{n-1}$.

Remark. For each $j \in [1, p]$, x_{i_j} is contained in the s_j'th source of G_n.

3) For all $j \in [1, p]$ transport x_{i_j} from the s_j'th source to all l'th sources with $l \in [s_j, s_{j+1} - 1]$.

Remark. This transport can obviously be executed along trees in G_n.

(4) For all $j \in [1, p]$ and all $l \in [s_j, s_{j+1} - 1]$, transport x_{i_j} from the l'th to the $\pi(l)$'th source along the pathes $R'_0, \ldots, R'_{n-1}$.

Remark. Now for each $j \in [1, p]$, x_{i_j} is contained in every l'th source with $l \in A_{i_j}$.

5) For every $j \in [1, p]$ and every $l \in A_{i_j}$, transport x_{i_j} from the l'th source to the l'th vertex of the base.

Thereby, we have distributed $(x_0, \ldots, x_{n-1})$ according to $A_0, \ldots, A_{n-1}$.

The Lemmas 3.4 and 3.6 imply that for the parts 2 and 4, $O(\log(n) + s)$ steps are needed, if each x_i, $i \in [0, n-1]$, has length at most s. Obviously the same holds for the other parts and thereby the theorem is proved. □

Now we shall show that W_n is a universal MPC. Let M be a PC with size n and degree c. $P_0, \ldots, P_{n-1}$ are the processors of M.

For some processor P_i of M let K_i be a configuration of P_i. $K := (K_0, \ldots, K_{n-1})$ then is called *a configuration of* M. If M has the configuration K, executes r steps, and has afterwards the configuration K', then K' is called the *r'th successor configuration of* K.

For some $i \in [0, n-1]$, Info(M, P_i, K, r) denotes the sequence of contents of communication registers of neighbours of P_i which are read by P_i during the r steps of the computation of M started in configuration K.

If A is a subset of the vertices of the graph G of M, then the PC which is defined by the subgraph of G being induced by A and the processors of M belonging to vertices of A is called *the restriction* M' *of* M *on* A.

If P is a processor of M' and Q a neighbour of P in M which doesn't belong to A, then the instruction "read the content of the communication register of Q" is replaced by "read a zero".

Now let A be a subset of the vertices of G which contains all vertices from the r-environment of some processor P_i of M. Let M' be the restriction of M on A, $K=(K_0, \ldots, K_{n-1})$ a configuration of M and L the configuration (K_i, P_i is processor of M').

Then, obviously the following lemma holds:

3.8. Lemma. *Suppose M and M' are started with the configurations K and L. Then the configurations of P_i in the r'th successor configurations of K and L in M and M' are the same. Furthermore,* $\text{Info}(M, P_i, K, r) = \text{Info}(M', P_i, L, r)$.

Now we shall define a simulation of M by the m-distributor W_m, where m is chosen in a suitable way. For some suitable $r \in \mathbb{N}$ let M_i, $i \in [0, n-1]$, be the restriction of M to the r-environment of P_i. The idea of the simulation is the following: In order to simulate M, we recursively simulate all M_i's at the same time by "small" $W_{m'}$'s which are contained in W_m by Theorem 3.7.

By Lemma 3.8 we know that after the simulation of r steps of the M_i's there is for every $i \in [0, n-1]$ at least one processor of M which simulates P_i "correctly" relative to M. This will be called the main-representant of P_i. The other processors, which simulate P_i but perhaps go wrong sometimes, are called potential representants of P_i. After the simulation of these r steps we use that W_m is a m-distributor to transport the information about the "true" configuration of every P_i from its main representant to all its potential representants.

They execute an "updating" to compute the "true" configuration of P_i, too.

Now we describe the parameters m, r and the size of M_i dependent on n.

The size of every M_i is $f(n)$, the length m of the base of W_m is $g(n)$, the number r of steps, for which we simulate the M_i's, is $h(n)$.

The simulation is illustrated in Fig. 6.

$W_{g(n)}$					
1 $W_{g(f(n))}$	2 $W_{g(f(n))}$	…	i $W_{g(f(n))}$ Here M_i is simulated.	…	n $W_{g(f(n))}$

Fig. 6

3.9. Definition. Let $p \in \mathbb{N}$, $p > 1$. Then f, g, h: $\mathbb{N} \to \mathbb{N}$ are defined as follows:

- For $n < c^p$ (c is the degree of M), $f(n) = h(n) = 1$.
- For $n > c^p$ let $k \in \mathbb{N}$ be chosen such that $c^{p^k} \leq n < c^{p^{k+1}}$. Then $f(n) = c^{p^{k-1}}$ and $h(n) = p^{k-1}$.
- $g(n) = \lfloor n^{1 + \frac{1}{p-1}} \rfloor$.

Thus $f(n) \approx n^{\frac{1}{p}}$ and $h(n) \approx \log(n)$.

3.10. Remark. f, g, h have the following properties:

a) $f(n) < n$ for all $n > 1$.

b) If G is a graph of degree c and x a vertex of G, then the $h(n)$-environment of x has at most n vertices.

c) $g(n) \geqq n \cdot g(f(n))$ for all $n \in \mathbb{N}$.

By 3.10b), there is for every $i \in [0, n-1]$ a set D_i of processors of M which contains the $h(f(n))$-environment of P_i and has $f(n)$ elements. The restriction of M on D_i let be called M_i.

Because of 3.10c) the MPC's

$$\bar{W}_i := W_{g(n)}^{ig(f(n)),(i+1)g(f(n))-1} \qquad \text{(compare Theorem 3.7)}$$

exist for every $i \in [0, n-1]$.

By Theorem 3.7 we know that each of these $\bar{W}_i$'s is a $g(f(n))$-distributor $W_{g(f(n))}$.

Now we define inductively *the potential and main representants of the processors of M in* $W_{g(n)}$.

Let $n < c^p$, then for every $i \in [0, n-1]$, the i'th vertex of the base of $W_{g(n)}$ is the (only) potential and the main representant of P_i.

Let $n \geqq c^p$. Because of 3.10a) we may inductively assume that for every $j \in [0, n-1]$ the potential representants and the main representant of every processor of M_j in $\bar{W}_j$ are already defined, because M_j has $f(n)$ processors and $\bar{W}_j$ is the $g(f(n))$-distributor $W_{g(f(n))}$.

For each $i \in [0, n-1]$, a vertex Q of the base of $W_{g(n)}$ is a potential representant of P_i in $W_{g(n)}$, iff there is a $j \in [0, n-1]$ such that P_i is a processor of M_j and Q is a potential representant of P_i (the processor of M_j) in $\bar{W}_j$.

The main representant of P_i in $W_{g(n)}$ is the main representant of P_i (the processor of M_i) in $\bar{W}_i$.

Now we shall describe how $h(n)$ steps of M can be simulated by $W_{g(n)}$.

Let $K = (K_0, \ldots, K_{n-1})$ be a configuration of M.

We say, $W_{g(n)}$ is prepared for K, if for every $i \in [0, n-1]$, every potential representant of P_i in $W_{g(n)}$ has stored K_i.

Let $\bar{K} = (\bar{K}_0, \ldots, \bar{K}_{n-1})$ be the $h(n)$'th successor configuration of K. We say, $W_{g(n)}$ *can compute* $\bar{K}$ *from* K, if it is possible to initialize $W_{g(n)}$ in such a way that the following holds: If $W_{g(n)}$ is prepared for K and starts its computation, then afterwards $W_{g(n)}$ is prepared for K and $\bar{K}$ and for every $i \in [0, n-1]$ the main representant of P_i has stored $\text{Info}(M, P_i, K, h(n))$.

3.11. Lemma. *If* $W_{g(n)}$ *is prepared for* K *then it can compute* $\bar{K}$ *from* K *in* $O(\log(n) \log\log(n))$ *steps.*

As $h(n) = O(\log(n))$, $W_{g(n)}$ *needs* $O(\log\log(n))$ *steps on an average to simulate one step of M.*

Proof of Lemma 3.11. We describe a recursive program for $W_{g(n)}$ which computes $\bar{K}$ from K.

If $n < c^p$ (compare Definition 3.9), it is obviously possible to compute $\bar{K}$ from K with the help of the techniques used in the proof of Theorem 2.11 in a constant (i.e. only dependent on c and p) number of steps.

Now let $n \geqq c^p$. Then we denote the $h(f(n))$'th successor configuration of K by $K' = (K'_0, \ldots, K'_{n-1})$. For $i \in [0, n-1]$ let L^i be the configuration (K_j, P_j is processor of M_i) for M_i and $\bar{L}^i = (\bar{L}^i_j$, P_j is processor of M_i) its $h(f(n))$'th successor configuration. As $W_{g(n)}$ is prepared for K, it follows from the definition of the potential and main representants that for every $i \in [0, n-1]$, W_i is prepared for L^i.

Our simulation begins as follows:

Part 1. For every $i \in [0, n-1]$, $\bar{W}_i$ computes $\bar{L}^i$ from L^i. This is done recursively.

Now we know that for $i \in [0, n-1]$ the main representant Q of P_i (the processor of M_i) in $\bar{W}_i$ has stored Info$(M_i, P_i, L^i, h(f(n))$. By Lemma 3.8 it follows that this tupel is Info $(M, P_i, K, h(f(n))$, as Q is the main representant of P_i (the processor of M) in $W_{g(n)}$, too.

Now let $j \in [0, g(n)-1]$, $i \in [0, n-1]$ such that the j'th vertex of the base of $W_{g(n)}$ is the main representant of P_i. Then $x_j := \text{Info}(M, P_i, K, h(f(n))$ and B_j is the set of all potential representants of P_i. All other x_j's and B_j's, $j \in [0, g(n)-1]$, are empty.

Part 2. Distribute $(x_0, \ldots, x_{g(n)-1})$ according to $(B_0, \ldots, B_{g(n)-1})$.

Now, for every $i \in [0, n-1]$, every potential representant of P_i has stored K_i and Info$(M, P_i, K, h(f(n)))$. Therefore the following program can be executed:

Part 3. For every $i \in [0, n-1]$, each potential representant of P_i computes K'_i.

After having executed these three parts, $W_{g(n)}$ is prepared for K' and K and for every $i \in [0, n-1]$, the main representant of P_i in $W_{g(n)}$ has stored Info$(M, P_i, K, h(f(n))$.

Now we execute these three parts $\dfrac{h(n)}{h(f(n))}$ times. Thereby, $W_{g(n)}$ computes $\bar{K}$ from K. We have to count the number of steps, this program needs.

Let $T(n)$ be the maximal number of steps which $W_{g(n)}$ needs to compute the $h(n)$'th successor configuration of some configuration of some PC with size n and degree c.

Then, part 1 needs $T(f(n))$ steps. As Info$(M, K, P_i, h(f(n)))$ has at most length $h(f(n))$ for every $i \in [0, n-1]$, we know from Theorem 3.7 that part 2 needs $O(\log(g(n)) + h(f(n))$ steps. Obviously, part 3 needs $O(h(f(n)))$ steps.

As each part is executed $\dfrac{h(n)}{h(f(n))}$ times we obtain:

$$T(n) \leqq \frac{h(n)}{h(f(n))}(T(f(n)) + O(\log(g(n)) + h(f(n))))$$

for $n > c^p$, $T(n) = \text{constant}$ otherwise.

By Definition 3.9 we know that $\dfrac{h(n)}{h(f(n))} = p$, $\log(g(n)) = O(\log(n))$, $h(f(n)) = O(\log(n))$ and $f(n) \leqq \lfloor n^{\frac{1}{p}} \rfloor$.

Therefore we get:

$T(n) \leqq p(T(\lfloor n^{\frac{1}{p}} \rfloor) + O(\log(n)))$ for $n \geqq c^p$ and therefore $T(n) = O(\log(n) \log\log(n))$, which proves the lemma. □

In order to simulate M started with some input $(x_0, \ldots, x_{q-1}) \in (\mathbb{N}^*)^q$, $W_{g(n)}$ first has to transport for each $i \in [0, q-1]$ the input from the i'th element of its base to all potential representants of the i'th input processor of M. As $W_{g(n)}$ is a distributor, this can be done in time $O(\log(g(n)) + s)$, where s is the maximal length of the x_i's. Afterwards it simulates M as described in Lemma 3.11, until M has stopped. Then the output is transported from the main representants of the output processors of M to the corresponding vertices of the base of $W_{g(n)}$. This needs $O(\log(g(n)) + s')$ steps, if every element of the output tupel has length at most s'. Obviously, $s, s' \leqq t$, if t is the number of steps M executes started with $(x_0, \ldots, x_{q-1})$. Therefore we obtain that the simulation time of M started with $(x_0, \ldots, x_{q-1})$ is $O(t \log\log(n) + \log(n))$.

Now let H_n be the set of all tupels of PC's $M_1, \ldots, M_r$ with degree c for which the following holds: if n_i is the size of M_i, $i \in [1, r]$, then $\sum_{i=1}^{r} g(n_i) \leqq g(n)$. ($H_n$ contains among other tupels all tupels which consist of PC's with degree c, which together have at most n processors.)

With the help of the above and Theorem 3.7, $E2$ we obtain:

3.12. Theorem. *Let $p \in \mathbb{N}$, $g(n) := \lfloor n^{1 + \frac{1}{p-1}} \rfloor$ for every $n \in \mathbb{N}$. Then $W_{g(n)}$ is universal for H_n. Let M be a PC with size $n' \leqq n$ and degree c, which needs t steps, if it is started with some input $\bar{x}$. Then the simulation time of M started with $\bar{x}$ in $W_{g(n)}$ is $O(t \log\log(n') + \log(n'))$.*

Chapter 4: *Types of Simulations.*

In this chapter we present three types of simulations of a PC M by a MPC M_0. In the two following chapters we prove lower bound for the efficiency of universal MPC's which can simulate all PC's with n processors and degree c, and which only use simulations of one of the above types.

The efficiency of such a universal MPC M_0 we measure by its size and its time-loss, i.e. the maximal factor by which the time some PC M with size n and degree c needs started with some input $\bar{x}$ and the simulation time of M_0 for M started with $\bar{x}$ differ.

We assume that each processor of the PC M being simulated by M_0 is at every time of the simulation simulated by at least one processor of M_0, its *representant*. Furthermore, each representant of some processor P of M shall at each time be capable to communicate with representants of all the neighbours of P. This communication is executed along transport pathes. Their lengthes determine the time-loss of the simulation.

As these kinds of simulations only depend on the graph of the PC's being simulated, we define the types of simulations in a graphtheoretical way.

Let $G(n, c)$ denote the set of all graphs with n vertices and degree c.

Let M_0 be an arbitrary graph with vertex set V_0 and $G \in G(n, c)$ a graph with vertex set $[1, n]$.

The easiest type of simulations we present here is the following: M_0 simulates G by attaching one representant i.e. one vertex of M_0 to each vertex of G. The communication between neighbours in G is simulated by transporting the corresponding information along transport pathes between their representants. These simulations we will call "of type 1".

In the introduction of this paper we have seen an example which shows that it can be reasonable to attach several representants to every processor of G. Such simulations are of type 2.

4.1. Definition. A simulation of G by M_0 of type 2 is specified by n pairwise disjoint, nonempty subsets $B_1, \ldots, B_n$ of V_0, and a set W of pathes in M_0.

For $i \in [1, n]$, B_i is the set of representants of the vertex i of G. W contains for all neighbouring vertices i and j of G and every representant of i a path from this representant to some representant of j. These pathes are called transport pathes. The time-loss of such a simulation is the length of a longest transport path. The number of representants is $\sum_{i=1}^{n} \# B_i$. M_0 is k-universal ((h, k)-universal) for $G(n, c)$ of type 2, if for every graph $G \in G(n, c)$ there is a simulation of G by M_0 of type 2 with time loss k (which uses at most h representants).

4.2. Definition. M_0 is k-universal for $G(n, c)$ of type 1 if it is (n, k)-universal for $G(n, c)$ of type 2.

In Chap. 2 we have got to know a $O(\log(n))$-universal MPC for $G(n, c)$ of type 1.

The simulations executed by the MPC from Chap. 3 are not of type 1 or 2.

In these simulations, the sets of representants vary dependent on time. For example sometimes only the main representants are representants, i.e. simulate "correctly", some times all potential representants are representants in the sense described in the beginning of this chapter.

Also the transport pathes vary dependent on time and the number of steps necessary to simulate one step of G, too. We now shall define a third type of simulations which includes the simulations shown in Chap. 3.

4.3. Definition. A simulation of T steps of G by M_0 of type 3 is specified by n pairwise disjoint, nonempty subsets $B_{1,t}, \ldots, B_{n,t}$ of V_0 for every $t \in [0, T]$, and by sets W_t of pathes of M_0 for every $t \in [1, T]$.

For $t \in [0, T]$, $j \in [1, n]$, $B_{i,t}$ is the set of representants of i at time t. For $t \in [1, T]$, W_t contains for every pair i, j of vertices of G which are neighbours or which fullfil $i = j$ and for every representant of i at time t a path to some representant of j at time $(t-1)$. Such a path is a t-transport path and the length of a longest t-transport path is the t-time-loss k_t. The time-loss of the simulation is $\frac{1}{T} \sum_{i=1}^{T} k_t$. The number of representants of this simulation is

$$\max\left\{\sum_{i=1}^{n} \# B_{i,t}, t \in [0, T]\right\}.$$

M_0 is k-universal ((h, k)-universal) for $G(n, c)$ of type 3, if for every $G \in G(n, c)$ and every $T \in \mathbb{N}$ there is a simulation of T steps of G by M_0 of type 3 which has a time-loss of at most k (and uses at most h representants).

If x is a representant at time $t\in[1, T]$ of a vertex i of G in M_0, then we call each representant at time $(t-1)$ which is connected to x by a t-transport path G *a predecessor of* x *(on level* $(t-1)$*)*.

Inductively, we call for $j>1$ every predecessor of a predecessor of x on level $t-(j-1)$ *a predecessor of* x *on level* $t-j$, if $j\leqq t$. We finish this chapter with the following observation:

4.4. Remark. If $t\geqq j$ and i and i' are vertices of G, such that i' is contained in the j-environment of i, then for every representant x of i at time t there is a predecessor of x on level $(t-j)$ which is a representant of i' at time $(t-j)$.

Chapter 5

In this chapter we prove a lower bound for the time-loss of a universal MPC for $G(n, c)$ of type 3 whose graph is a so-called uniform distributor. These graphs are interesting, because the graph of the universal MPC from Chap. 3 belongs to them. We shall show that this MPC has an asymptotically minimal time-loss among all universal MPC's for $G(n, c)$ of type 3 whose graph is a uniform distributor.

5.1. Definition. A *uniform n-distributor* is a graph which has n distinguished vertices $b_1, \ldots, b_n$, the base B of the graph, with the property: For all $b_p, b_q\in B$, $d(b_p, b_q)=\Omega(\log(|p-q|)$.[3]

Obviously the graphs of the distributors of chapter 3 are uniform distributors.

Let D be a balanced, binary tree with n vertices.

5.2. Theorem. *Let* $m\in\mathbb{N}$, M_0 *a uniform m-distributor, and* $T\geqq\lfloor(\log(n)\rfloor$. *Then every simulation of* T *steps of* D *by* M_0 *of type* 3, *which only uses vertices of its base as representants, has a time-loss of* $\Omega(\log\log(n))$.

Especially, M_0 is $\Omega(\log\log(n))$*-universal for* $G(n, c)$ *of type* 3, *if it only uses vertices of its base as representants.*

Proof. Let $T\geqq\lfloor\log(n)\rfloor$. We consider a simulation of T steps of D by M_0 of type 3, which only uses vertices of its base as representants. Let $a>0$ chosen such that $d(b_p, b_q)\geqq a\cdot\log(|p-q|)$ for $p, q\in[1, n]$. Let b_p be a representant of a vertex of D at time t, $t\in[0, T]$. Let the t-time-loss of this simulation be k_t. If b_q is a predecessor of b_p, we have $d(b_p, b_q)\leqq k_t$. Therefore, $a\cdot\log(|p-q|)\leqq k_t$ which implies, $|p-q|\leqq 2^{\frac{k_t}{a}}$. Induction guarantees that for each predecessor b_r of b_p on level $(t-j)$, $j\leqq t$, it holds that $|r-p|\leqq \sum_{i=0}^{j-1} 2^{\frac{k_{t-i}}{a}}$.

Therefore it follows:

5.3. Lemma. b_p *has at most*

$$2\cdot\sum_{i=0}^{j-1} 2^{\frac{k_{t-i}}{a}} \qquad \textit{predecessors on level } (t-j).$$

---

[3] $d(b_p, b_q)$ is the length of a shortest path in the graph between b_p and b_q

Now let $t\in[\lfloor\log(n)\rfloor, T]$, $j\in[1,\lceil\log(n)\rceil-1]$, and b_q a representant of the root of D at time t. Then by Remark 4.4 every vertex of the j-environment of this root has a representant at time $(t-j)$ which is a predecessor of b_p on level $(t-j)$. As these representants are pairwise disjoint and the j-environment of this root has $2^{j+1}-1$ elements ($j\leqq\lfloor\log(n)\rfloor-1$!) it follows by Lemma 5.3:

$$2^{j+1}-1\leqq 2\cdot\sum_{i=0}^{j-1} 2^{\frac{k_{t-i}}{a}}.$$

Therefore, there is a $i_0\in[0,j-1]$, such that

$$2^{j+1}-1\leqq 2\cdot j\cdot 2^{\frac{k_{t-i_0}}{a}}.$$

Let $t_0=t-i_0$, then we obtain:

$$k_{t_0}\geqq a\cdot\log\left(\frac{2^{j+1}-1}{2j}\right)\geqq a'\cdot j \qquad \text{for some suitable } a'>0.$$

Thus we have proved:

5.4. Lemma. *There is $a'\geqq 0$ such that for every $t\in[\lfloor\log(n)\rfloor, T]$ and every $j\in[1,\lfloor\log(n)\rfloor-1]$, there is a $t_0\in[t-j+1,t]$, such that $k_{t_0}\geqq a'j$.*

Now let $h:=\lfloor\log(\lfloor\log(n)\rfloor-1)\rfloor$. Then $2^h\leqq\lfloor\log(n)\rfloor-1$. We partition for some $r\in[0,h]$ the intervall $[t-2^h+1,t]$ in 2^{h-r} pairwise disjoint intervalls of length 2^r. By Lemma 5.4, each intervall contains a t_0 with $k_{t_0}\geqq a'2^r$.

Let C_r denote the set of numbers $t_1\in[t-2^h+1,t]$ with $k_{t_1}\geqq a'2^r$, then we may conclude that $\#C_r\geqq 2^{h-r}$ and $C_r\subset C_{r-1}\subset\ldots\subset C_0$. Obviously, for every $r\in[0,h]$ there is a subset C'_r of C_r with $\#C'_r\geqq 2^{h-r-1}$ such that $C'_0,\ldots,C'_r$ are pairwise disjoint. Thus we obtain:

$$\begin{aligned}\sum_{i=0}^{2^h-1} k_{t-i}&\geqq\sum_{r=0}^{h}\sum_{t'\in C'_r} k_{t'}\\ &\geqq\sum_{r=0}^{h}\sum_{t'\in C'_r} a'2^r\geqq a'\sum_{r=0}^{h} 2^{h-r-1}2^r=a'\cdot h\cdot 2^{h-1}\\ &\geqq\bar{a}\cdot 2^h\cdot\log\log(n) \qquad \text{for some suitable } \bar{a}>0.\end{aligned}$$

Let $s:=\left\lfloor\frac{T}{2^h}\right\rfloor$ ($\geqq 1$, because $T\geqq\lfloor\log(n)\rfloor$).

Now we partition the intervall $[1,T]$ in $(s+1)$ (not neccecarily pairwise disjoint) intervalls $I_1,\ldots,I_{s+1}$ of length 2^h.

Obviously this can be done in such a way that every element of $[1,T]$ is contained in at most two of these intervalls.

Therefore we get:

$$\sum_{i=1}^{T} k_t\geqq\frac{1}{2}\sum_{j=1}^{s+1}\sum_{t\in I_j} k_t.$$

As $\sum_{t\in I_j} k_t$ is proved to be at least

$$\bar{a}\cdot 2^h\cdot \log\log(n) \qquad \text{for every } j\in[1,s+1],$$

and as $s+1\geqq \frac{T}{2^h}$ it follows:

$$\begin{aligned}\sum_{i=1}^{T} k_t &\geqq \frac{1}{2}\sum_{j=1}^{s+1} \bar{a}\cdot 2^h\cdot\log\log(n)\\ &=\tfrac{1}{2}\cdot\bar{a}\cdot(s+1)\cdot 2^h\cdot\log\log(n)\\ &\geqq\frac{1}{2}\cdot\bar{a}\frac{T}{2^h}2^h\log\log(n)=\frac{1}{2}T\bar{a}\log\log(n).\end{aligned}$$

Thus the time-loss $\frac{1}{T}\sum_{i=1} k_t$ is bounded from below by $\frac{1}{2}\cdot\bar{a}\cdot\log\log(n)$ which proves Theorem 5.2. □

In order to finish this chapter we notice that for a k-universal uniform m-distributor of type 2 which only uses vertices of its base as representants it follows that $k=\Omega(\log(n))$.

This follows by evaluating Lemma 5.4 for $j=\lfloor\log(n)\rfloor-1$, and by taking into consideration that a simulation of type 2 is a simulation of type 3 in which (among other things) all t-time-losses are equal.

Chapter 6

In this chapter we prove that universal MPC's for $G(n,c)$ of type 1, 2, or 3 can not have a "small" time-loss and a "small" number of processors at the some time.

In what follows let c, $d\geqq 3$.

6.1. Theorem. *Let $M_0\in G(m,d)$ be (h,k)-universal for $G(n,c)$ of type 2 then $h\cdot k=\Omega(n\log(n))$ or $m=n^{\Omega(n^2/h)}$.*

6.2. Corollary. *Let $M_0\in G(m,d)$ be k-universal for $G(n,c)$ of type 1, then $k=\Omega(\log(n))$ or $m=n^{\Omega(n)}$.*

Thus a universal MPC of type 1 which has an asymptotically smaller time-loss than that one from Chap. 2 has an exponential size.

6.3. Corollary. *Let $M_0\in G(m,d)$ be k-universal for $G(n,c)$ of type 2. Then $m\cdot k=\Omega(n\log(n))$.*

This bound is asymptotically achieved by the universal MPC of Chap. 2.

6.4. Theorem. *Let $M_0\in G(m,d)$ be (h,k)-universal for $G(n,c)$ of type 3. Then $h\cdot k=\Omega(n\log(n)/\log\log(n))$ or $m=n^{\Omega\left(\frac{n\log(n)}{h}\right)}$.*

6.5. Corollary. *Let $M_0 \in G(m, d)$ be k-universal for $G(n, c)$ of type 3. Then* $m \cdot k = \Omega(n \log(n)/\log\log(n))$.

The corollaries are easy conclusions of the theorems.

The rest of this paper is devoted to the proofs of the Theorems 6.1 and 6.4.

These proofs follow this pattern:

Let $M_0 \in G(m, d)$.

To each simulation S of a graph by M_0 of type 2 or 3 with time-loss k, which uses at most h representants, we attach a fragment, i.e. an object from which we can still recognize the graph being simulated. Therefore the number of these fragments is an upper bound for the number of graphs for which there are such simulations by M_0.

In order to get better estimentions, we only consider the number Y of fragments, which belong to graphs from a certain subset $G'(n, c)$ of $G(n, c)$.

On the other hand we bound $\# G'(n, c)$ from below. As M_0 is (h, k) universal for $G(n, c)$, therefore for $G'(n, c)$, too, we obtain that $\# G'(n, c) \leqq Y$. This unequality proves the respective theorem.

First we note some estimations which we need in the proofs.

6.6. Lemma.

a) *For all* $k, n \in \mathbb{N}$, $1 \leqq k \leqq n$, $\binom{n}{k} \leqq n^k$.

b) $\# \{(a_1, \ldots, a_n) \in (\mathbb{N} \setminus \{0\})^n \mid \sum_{i=1}^{n} a_i \leqq h\} \leqq 2^h$.

c) *Let* $(a_1, \ldots, a_n), (b_1, \ldots, b_n) \in (\mathbb{N} \setminus \{0\})^n$.

Let $p \in \mathbb{N}$ *such that* $p \cdot a_i \geqq b_i$ *for every* $i \in [1, n]$, *and* $\sum_{i=1}^{n} a_i \leqq h$. $\sum_{i=1}^{n} b_i \leqq h$. *Then.*

$$\prod_{i=1}^{n} \binom{p \cdot a_i}{b_i} \leqq e^{2h} \cdot p^h.$$

The stimation in a) is very ruff. We only shall estimate more carefully if it is necessary. Otherwise we use such ruff estimations in order to get simpler terms.

Proof of Lemma 6.6. a) and b) are standard estimations.

For the proof of c) we use the well known unequality:

For $n, k \in \mathbb{N}$, $1 \leqq k \leqq n$, $\binom{n}{k} \leqq \left(\frac{ne}{k}\right)^k$.

Thereby we obtain:

$$\prod_{i=1}^{n} \binom{p \cdot a_i}{b_i} \leqq \prod_{i=1}^{n} \left(\frac{p \cdot a_i \cdot e}{b_i}\right)^{b_i}$$
$$\leqq p^h \cdot e^h \cdot \prod_{i=1}^{n} \left(\frac{a_i}{b_i}\right)^{b_i}.$$

It remains to prove: $\prod_{i=1}^{n} \left(\frac{a_i}{b_i}\right)^{b_i} \leqq e^h$.

We need the following, well known unequality:

For every $n \in \mathbb{N}$ and every real number $x \geqq 0$: $\left(1+\frac{x}{n}\right)^n \leqq e^x$.

Now we get:

$$\prod_{i=1}^{n} \binom{a_i}{b_i}^{b_i} \leqq \prod_{\substack{i \in [1,n] \\ a_i > b_i}} \binom{a_i}{b_i}^{b_i}$$

$$= \prod_{\substack{i \in [1,n] \\ a_i > b_i}} \left(1+\frac{a_i-b_i}{b_i}\right)^{b_i} \leqq \prod_{\substack{i \in [1,n] \\ a_i > b_i}} e^{a_i-b_i}.$$

$$\leqq \prod_{i \in [1,n]} e^{a_i} \leqq e^h. \quad \square$$

Now let $c' < c$ and $G_0 \in G(n, c')$. Let $G(n, c, G_0)$ be the set of all graphs from $G(n, c)$ which contain the subgraph G_0. We now shall bound the cardinality of $G(n, c, G_0)$.

Let a graph be called r-colorable if there is a mapping from its edges to $[1, r]$ such that neighbouring edges have different images.

6.7. Lemma. *Let $c' < c$ and let $G_0 \in (n, c')$ be a r-colorable graph. Then $\# G(n, c, G_0) \geqq n^{\frac{c-r}{2}} \cdot 2^{-a_1 n}$ for some suitable $a_1 > 0$.*

Proof. Let $c'' := c - r$ and let $G_1, \ldots, G_{c''} \in G(n, 1)$. A (n, c'')-multigraph with marked edges is defined by the vertices $[1, n]$, such that two vertices $a, b \in [1, n]$ are joint by an edge marked with i, if $\{a, b\}$ is an edge of G_i, $i \in [1, c'']$. (Maybe there a serval edges with different marks between two vertices).

Let B be the set of all (n, c'')-multigraphs with marked edges.

Claim 1. $\# B \geqq n^{\frac{c''}{2} n} \cdot 2^{-an}$ for some suitable $a > 0$.

Proof. Obviously, $\# G(n, 1) \geqq \binom{n}{2}!$

By Stirlings Formula we obtain that $\# G(n, 1) \geqq n^{\frac{n}{2}} \cdot 2^{-a'n}$ for some suitable $a' > 0$.

Therefore, $\# B = (\# G(n, 1))^{c''} \geqq n^{c'' \frac{n}{2}} \cdot 2^{-c'' a' n}$ which proves Claim 1.

Now let B be the set of all (n, c)-multigraphs whose edges marked with $[c - r + 1, c]$ form the graph G_0. As G_0 is r-colorable, G_0 can be formed in this way.

Let g: $B' \to G(n, c, G_0)$ be the mapping which attaches to some $H \in B'$ that graph which has an edge between exactly those pairs of vertices between which there is at least one arbitrarily marked edge in H. Obviously, g is well defined and surjective.

Claim 2. For every $G \in G(n, c, G_0)$, $\# g^{-1}(G) \leqq (c \cdot 2^c \cdot c^c)^n$.

Proof. Let $G \in G(n, c, G_0)$, x a vertex of G and $x_1, \ldots, x_b$, $b \leqq c$ the neighbours of x.

Then we know about every multigraph $H \in g^{-1}(G)$:

1) The number b' of neighbours of x in H is at most c.

2) There is a tupel $(a_1, \ldots, a_b) \in (\mathbb{N} \setminus \{0\})^b$ with $\sum_{i=1} a_i = b'$, such that exactly a_i edges are between x in x_i in H.

3) The edges in H are marked with numbers from $[1, c]$.

Therefore, there are at most c possible choices of b. By Lemma 6.6.b, there are at most 2^c possible choices of $a_1, \ldots, a_b$. The number of possible choices of the marks of the edges starting from x is $c^{b'} \leqq c^c$.

Thus, there are at most $c \cdot 2^c \cdot c^c$ possibilities to fix the edges and their marks between x and $x_1, \ldots, x_b$ in H.

As every element of $g^{-1}(G)$ is specified by fixing these edges and marks for all of its vertices, we obtain:

$$\# g^{-1}(G) \leqq (c \cdot 2^c \cdot c^c)^n.$$

Obviously, $\# B' \geqq \# B$.

Therefore we may conclude from claim 1 and 2 that

$$\# G(n, c, G_0) \geqq (\# B') \cdot (c \cdot 2^c \cdot c^c)^{-n} \geqq n^{\frac{c-r}{2} n} \cdot 2^{-a_1 n}$$

for some suitable $a_1 > 0$. □

Proof of Theorem 6.1. Let $M_0 \in G(m, d)$ be (h, k)-universal for $G(n, c)$ of type 2. We assume that the vertex set of M_0 is $[1, m]$ and that one of some graph from $G(n, c)$ $[1, n]$.

Let $C_0 \in G(n, 1)$ be the cycle with n vertices and edges $\{i, i+1\}$ for every $i \in [1, n-1]$ and the edge $\{n, 1\}$.

Let $G \in G(n, c, C_0)$. As $G(n, c, C_0) \subset G(n, c)$, there is a simulation of G by M_0 of type 2 with time-loss at most k which uses at most h representants.

Let it be specified by the representant sets $B_1, \ldots, B_n$ and the set W of transport pathes. Then $(B_1, \ldots, B_n, W)$ let be called a *(h, k)-strategy for G.*

Let $S := \{(x, y) \in [1, m]^2 \mid x = \min(B_i)$ for some $i \in [1, n]$, and there is a transport path from x to y in $W\}$.

Then $(B_1, \ldots, B_n, S)$ is called a *fragment of* $(B_1, \ldots, B_n, W)$.

This fragment doesn't specify any longer how the strategy simulates, but it still specifies which graph is simulated.

Let R be the number of graphs from $G(n, c, C_0)$ for which there is a (h, k)-strategy, and Y the number of fragments of (h, k)-strategies of graphs from $G(n, c, C_0)$.

Then it follows:

6.8. Proposition. $R \leqq Y$. *Now we shall bound Y from above.*

6.9. Proposition. $Y \leqq m^{\frac{h}{n}} \cdot e^{3h} \cdot d^{(k+1)h} \cdot d^{(k+1)cn} \cdot c^n$.

Proof. Let Y_1 be the number of possible choices of representant sets $B_1, \ldots, B_n$ in fragments of (h, k)-strategies of graphs from $G(n, c, C_0)$.

Claim 1. $Y_1 \leqq m^{\frac{h}{n}} \cdot e^{3h} \cdot d^{(k+1)h}$.

Proof. Let $(h_1, \ldots, h_n) \in (\mathbb{N} \setminus \{0\})^n$, $\sum_{i=1}^{n} h_i \leqq h$. First we bound the number Z of possible choices of $B_1, \ldots, B_n$ with $\# B_i = h_i$, $i \in [1, n]$.

Let i_1 be chosen such that h_{i_1} is minimal among $h_1, \ldots, h_n$, and let $(i_1, \ldots, i_n) := (i_1, i_1+1, \ldots, n, 1, \ldots, i_1-1)$.

Thus for every $j \in [1, n-1]$, $\{i_j, i_{j+1}\}$ is an edge in C_0.

Let the k-environment of some subset B of the vertices of M_0 be denoted by $U_k(B)$.

Then the following holds:

- There are $\binom{m}{h_{i_1}}$ possible choices of B_{i_1}.
- Let $1 \leqq p < n$, and suppose that $B_{i_1}, \ldots, B_{i_p}$ are already fixed.
- As $\{i_p, i_{p+1}\}$ is an edge of every graph from $G(n, c, C_0)$, each representant of i_{p+1}, i.e. each element of $B_{i_{p+1}}$, is joint to an element of B_{i_p} by a transport path of length at most k.

Therefore $B_{i_{p+1}} \subset U_k(B_{i_p})$.

As $\# U_k(B_{i_p}) \leqq d^{k+1} \cdot h_{i_p}$, there are at most $\binom{d^{k+1} h_{i_p}}{h_{i_{p+1}}}$ possible choices for $B_{i_{p+1}}$.

Thus we obtain:

$$Z \leqq \binom{m}{h_{i_1}} \prod_{p=1}^{n-1} \binom{d^{k+1} h_{i_p}}{h_{i_{p+1}}}.$$

As h_{i_1} is chosen minimally among $h_1, \ldots, h_n$, $h_{i_1} \leqq \frac{h}{n}$ and therefore, $\binom{m}{h_{i_1}} \leqq \binom{m}{h/n}$.

Applying Lemma 6.6a) and c) we obtain: $Z \leqq m^{\frac{h}{n}} \cdot d^{(k+1)h} \cdot e^{2h}$.

By Lemma 6.6b we know that there are at most 2^h possible choices for $h_1, \ldots, h_n$. Therefore $Y_1 \leqq Z \cdot 2^h$ which proves claim 1.

Now let $B_1, \ldots, B_n$ be fixed. We now bound the number Y_2 of fragments with representant sets $B_1, \ldots, B_n$.

Claim 2. $Y_2 \leqq d^{(k+1)cn} \cdot c^n$.

Proof. For $i \in [1, n]$ let $b_i = \min(B_i)$ and $(B_1, \ldots, B_n, S)$ a fragment of $a(h, k)$-strategy.

Then it holds for S:

- for every $i \in [1, n]$ the number c_i of vertices Y of M_0 with $(b_i, y) \in S$ is at most c.
- Every pair from S has the form (b_i, y) for some $i \in [1, n]$ such that $y \in U_k(b_i)$.

Therefore in follows:

In order to specify S, there are for every $i \in [1, n]$ at most c possible choices for c_i and

$$\binom{\# U_k(b_i)}{c_i} \leqq \binom{\# U_k(b_i)}{c}$$

possibilities to fix the c_i pairs (b_i, y).

Therefore $Y_2 \leqq c^n \cdot \binom{d^{k+1}}{c}^n$.

With the help of Lemma 6.6a), Claim 2 follows. Proposition 6.9 is now proved by Claim 1 and 2 because $Y \leqq Y_1 \cdot Y_2$. □

Now we are able to prove Theorem 6.1.

W.l.o.g. we may assume that n is even. Then C_0 is 2-colorable and Lemma 6.7 shows that

$$\# G(n, c, C_0) \geqq n^{\frac{c-2}{2}n} \cdot 2^{-a_1 n}.$$

Therefore we obtain with the help of Proposition 6.8 and 6.9:

$$\begin{aligned} n^{\frac{c-2}{2}n} \cdot 2^{-a_1 n} &\leqq \# G(n, c, C_0) \leqq R \leqq Y \\ &\leqq m^{\frac{h}{n}} \cdot e^{3h} \cdot d^{(k+1)h} \cdot d^{(k+1)cn} \cdot c^n. \end{aligned} \qquad (*)$$

Now let $a_2, a_3 > 0$ be chosen such that

$$\frac{c-2}{2} > a_3 = a_2(\log(d)(c+1) + 3\log(e)) \quad \text{and} \quad h \cdot (k+1) \leqq a_2 n \log(n).$$

The theorem is proved, if we now can show that $m = n^{\Omega(n^2/h)}$. This is done by manipulating the unequality (*).

$$\begin{aligned} m &\geqq 2^{\frac{n}{h}\left(\frac{c-2}{2} n\log(n) - 3h\log(e) - (k+1)h\log(d)(c+1) - O(n)\right)} \\ &\geqq 2^{\frac{n}{h}\left(\left(\frac{c-2}{2} - a_3\right) n\log(n) - O(n)\right)} = n^{\Omega(n^2/h)}. \end{aligned}$$

Proof of Theorem 6.4. Let M_0 be (h,k)-universal for $G(n,c)$ of type 3. Again let the vertex sets we consider be $[1,m]$ and $[1,n]$.

Let $D \in G(n,3)$ be a balanced, binary tree. D has depth $\lfloor \log(n) \rfloor$. Now let $A \in \mathbb{N}$ be fixed, $A \leqq n$. A will be specified later.

Let $r \in \mathbb{N}$ and $V_1, \ldots, V_r$ be r subsets of $[1,n]$ of cardinality A which cover $[1,n]$ such that for every $i \in [1,r]$, the subgraph of D induced by V_i is a balanced, binary tree of depth $\lfloor \log(A) \rfloor$. Obviously, $V_1, \ldots, V_r$ can be chosen such that $r \leqq \frac{2n}{A}$ and every $i \in [1,n]$ is contained in at most two of the V_i's. Now we consider a graph $G \in G(n,c,D)$ and a $T \in \mathbb{N}$. As $G(n,c,D) \subset G(n,c)$, there is a simulation of T steps of G by M_0 of type 3 with time-loss at most k which uses at most h representants. Let it be specified by the representant sets $B_{1,t}, \ldots, B_{n,t}$ for every $t \in [0,t]$ and the sets W_t of t-transport pathes for every $t \in [1,T]$. We assume that $T \geqq 2\lfloor \log(A) \rfloor + 1$ and call $(B_{1,t}, \ldots, B_{n,t}, W_t)_{t \leqq T}$ *a (h,k)-strategy for* G. For $t \in [1,T]$ let k_t be the t-time-loss of the strategy.

We count the number of graphs for which there is a (h,k)-strategy as follows: For some t_0 we count the number of possible choices of $B_1, \ldots, B_n = B_1^{t_0}, \ldots, B_n^{t_0}$ in a strategy. Afterwards we estimate the number of possible choices of sets S of edges of graphs which can be simulated by a strategy with the above representants at time t_0 and (t_0+1)-time-loss k_{t_0+1}. Unfortunately, this method, i.e. the choice of $(B_1, \ldots, B_n, S)$ as fragments, is to weak for our purpose, because there are too many choices for $B_1, \ldots, B_n$. Therefore we first fix the representants $B'_1, \ldots, B'_r$ of r suitably chosen vertices of G – one from each V_i – at time $t_0 - 2\lfloor\log(A)\rfloor$. Their number is not too large if t_0 is chosen reasonably. As all considered graphs contain a balanced binary tree, after having fixed $B'_1 \ldots B'_r$ the number of choices of $B_1, \ldots, B_n$ decreases considerably.

Formally a fragment is defined as follows:

Let $t_0 \in [2\lfloor\log(A)\rfloor, T-1]$ be chosed such that $\sum_{t=t_0-2\lfloor\log(A)\rfloor+1}^{t_0+1} k_t)$ is minimal relative to the choice of t_0. This sum is called R_0.

Now a *fragment of the strategy* $(B_{1,t}, \ldots, B_{n,t}, W_t)_{t \leqq T}$ is specified by a tupel $(B_1, \ldots, B_n, B'_1, \ldots, B'_r, S)$ as follows:

$$(B_1, \ldots, B_n) = (B_{1,t_0}, \ldots, B_{n,t_0}).$$

If $j \in [1, r]$ and $i_j \in V_j$ such that $B_{i_j, t_0 - 2\lfloor\log(A)\rfloor}$ has a minimal cardinality relative to the coice of i_j, then $B'_j = B_{i_j, t_0 - 2\lfloor\log(A)\rfloor}$.

$S := \{(x, y) \in [1, m]^2 / x \in B_{i,t_0}$ and there is an $i \in [1, n]$, such that there are two (t_0+1)-transport pathes in W_{t_0+1} which join x and y to the minimal element of $B_{i,t_0+1}\}$.

Let R' be the number of graphs from $G(n, c, D)$ for which there is a (h, k)-strategy in M_0, and Y' the number of fragments of (h, k)-strategies for graphs from $G(n, c, D)$.

Obviously a fragment still specifies the graph being simulated. Therefore, the following holds:

6.10. Proposition. $R' \leqq Y'$.
Before we bound Y', we state some properties of the fragment described above.

6.11. Proposition. a) $K_{t_0+1} \leqq R_0 \leqq 2k(2\lfloor\log(A)\rfloor + 1)$.

b) $\sum_{i=1}^{r} \# B'_1 \leqq \frac{2h}{A}$.

c) *For every* $j \in [1, r]$ *and every* $i \in V_j$, $B_i \subset U_{R_0}(B'_j)$.

Proof. a) and b) follow easily from the definition of the fragment. c) follows from Remark 4.4. □

Now we bound Y'.

6.12. Proposition. $Y' \leqq m^{\frac{2h}{A}} \cdot d^{(h+2cn)(5k\log(A))} \cdot e^{4h} \cdot \left(\frac{h}{n}\right)^n$.

Proof. First we bound the number Y'_1 of all tupels $(B_1, \ldots, B_n, B'_1, \ldots, B'_r)$, which belong to a fragment of a (h, k)-strategy of a graph from $G(n, c, D)$.

Claim 1. $Y'_1 \leqq m^{\frac{2h}{A}} \cdot e^{4h} \cdot d^{h(R_0+1)}$.

Proof. Let the cardinalities $h_1, \dots, h_n, h'_1, \dots, h'_r$ of $B_1, \dots, B_n, B'_1, \dots, B'_r$ be fixed.
– By Lemma 6.6b), there are at most 2^{2h} possible choices of $h_1, \dots, h_n, h'_1, \dots, h'_r$.
– There are at most $\prod_{i=1}^{r} \binom{m}{h'_i}$ possible choices of $B'_1, \dots, B'_r$.
– For $j \in [1, r]$ let $V'_j \subset V_j$ chosen such that $V'_1, \dots, V'_r$ form a disjoint partition of $[1, n]$. By Proposition 6.11c) follows for every $j \in [1, r]$ and every $i \in V'_j$: There are at most $\binom{h'_j \cdot d^{R_0+1}}{h_i}$ possible choices for B_i. Therefore we obtain:

$$Y'_1 \leqq 2^{2h} \cdot \prod_{q=1}^{r} \binom{m}{h'_q} \prod_{j=1}^{r} \prod_{i \in V'_j} \binom{h'_j \cdot d^{R_0+1}}{h_i}.$$

Applying Lemma 6.6a) and c) we obtain

$$Y'_1 \leqq 2^{2h} \cdot m^{\sum_{i=1}^{r} h'_i} \cdot d^{h(R_0+1)} \cdot e^{2h}.$$

By Lemma 6.11b), $\sum_{i=1}^{r} h'_i \leqq \frac{2h}{A}$, which proves Claim 1.

Now we bound for some fixed sets $B_1, \dots, B_n, B'_1, \dots, B'_r$ the number Y'_2 of fragments of (h, k)-strategies which can be formed by these sets.

Claim 2. $Y'_2 \leqq \left(\frac{h}{n}\right)^n \cdot d^{2(k_{t_0+1}+1)cn}$.

Proof. If $(B_1, \dots, B_n, B'_1, \dots, B'_r, S)$ is a fragment of a (h, k)-strategy, it follows for S:
– There are at most n different first components of pairs occuring in S, one in each B_i, $i \in [1, n]$.
– At most c second components belong to each first component x. They are contained in $U_{2(k_{t_0+1})}(x)$.

For $i \in [1, n]$ let $h_i = \# B_i$. Then there are at most $\prod_{i=1}^{n} h_i \leqq \left(\frac{h}{n}\right)^n$ possible choices for the n first components of the pairs of S.

In order to fix the second components for some first component x, there are at most $\binom{d^{2k_{t_0+1}+1}}{c}$ possible choices. Therefore it follows by Lemma 6.6a):

$$Y'_2 \leqq \left(\frac{h}{n}\right)^n \cdot \binom{d^{2k_{t_0+1}+1}}{c}^n$$
$$\leqq \left(\frac{h}{n}\right)^n \cdot d^{(2k_{t_0+1}+1)cn}.$$

As $Y' \leqq Y'_1 \cdot Y'_2$ the Proposition is proved by Claim 1 and 2 and the bounds for R_0 and k_{t_0+1} from 6.11a). □

Now we are ably to prove Theorem 6.4. As D is 3-colorable, it follows from Lemma 6.7 that

$$\# G(n,c,D) \geqq n^{\frac{c-3}{2}n} 2^{-a_1 n}.$$

W.l.o.g. we may assume that $c \geqq 4$. An analogous argument as used in the proof of Theorem 6.1 guarantees with the help of the Proposition 6.10 and 6.12 that

$$n^{\frac{c-3}{2}n} \cdot 2^{-a_1 n} \leqq 2^{\frac{2h}{A}} \cdot e^{4h} \cdot d^{(h+2cn)(5k\log(A))} \cdot \binom{h}{-\atop n}^n.$$

Therefore,

$$m \geqq 2^{\frac{A}{2h}\left(\frac{c-3}{2} n\log(n) - a_1 n - 4h\log(e) - \log(d)(h+2cn)(5k\log(A)) - \log\binom{h}{-\atop n} n\right)}.$$

Let a_4, $a_5 > 0$ be chosen such that

$$\frac{c-3}{2} > a_5 = a_4(4\log(e) + 5(2c+1)\log(d))$$

and let $h \cdot k \cdot \log(A) \leqq a_4 n \log(n)$.

Then $\log\binom{h}{-\atop n} \leqq \log\log(a_4 n)$ and it follows:

$$\begin{aligned} m &\geqq 2^{\frac{A}{2h}\left(\frac{c-3}{2} - a_5\right) n\log(n) - a_1 n - n\log\log(a_4 n))} \\ &\geqq n^{\Omega\left(\frac{An}{h}\right)}. \end{aligned}$$

Now we choose $A = \lfloor \log(n) \rfloor$ and obtain Theorem 6.4.

References

1. Aho, A.V., Hopcroft, J.E., Ullman, J.D.: The Design and Analysis of Computer Algorithms. Reading, MA: Addison-Wesley, 1974
2. Galil, Z., Paul, W.J.: A Theory of Complexity of Parallel Computation. Proc. 13th Ann. ACM Symp. Theory Comput. Milwankee, pp. 247–262, 1981
3. Paul, W.J.: Komplexitätstheorie. Stuttgart: Teubner Verlag, 1978
4. Preparata, F.P., Vuillemin, J.: The Cube-Connected Cycles: A Versatile Network for Parallel Computation. Proc. 20th Ann. IEEE Symp. Found. Comput. Sci., Puerto Rico, pp. 140–147, 1979
5. Waksman, A.: A Permutation Network. J. ACM **15**, 159–163 (1968)

Received May 10, 1982/November 30, 1982