# Data Transmission in Processor Networks

**Andreas Jakoby**⋆

**Rüdiger Reischuk**

Med. Universität Lübeck⋆⋆

**Abstract.** We investigate the communication capacity and optimal data transmission schedules for processor networks connected by communication links, for example Transputer clusters. Each link allows the two processors at its endpoints to exchange data with a given fixed transmission rate $\tau_d$. The communication itself is done in a blocking mode, that means the two processors have to synchronize before starting to exchange data and at any time each processor cannot communicate with more than one other processor.

Our efficiency analysis will be more realistic by also taking into account the setup time for a communication, which will be assumed to be a fixed constant $\tau_s > 0$. Thus, a large amount of data can be sent from one processor to a neighbour faster by a single long communication step than by a bunch of small data exchange steps: sending $p$ data units in one step takes time $\tau_s + p \cdot \tau_d$. However, there is a tradeoff since the receiver has to wait until it has received the complete set of data before it can forward pieces to other processors.

The following prototype task called *scattering* will be considered: At the beginning one processor called the *source* possesses a set of unit size data packets, one for each processor in the network. The goal is to distribute the packets in minimal time to all recipients.

Our results concerning the complexity of this problem in arbitrary processor networks are as follows: For the general case, we give lower bounds on the minimal schedule length and show that to determine the length precisely is $\mathcal{NP}$–complete. Special classes of simple strategies are investigated in more detail. For certain networks they turn out to yield optimal schedules.

Finally, we investigate optimal schedules that can be computed efficiently and good approximation algorithms for specific regular networks like hypercubes and multidimensional grids.

## 1 Introduction

### 1.1 Motivation

When designing efficient algorithms for distributed networks one often faces a serious bottleneck due to the data distribution overhead. For example, consider

⋆ supported by DFG Research Grant Re 672-2

⋆⋆ Institut für Theoretische Informatik, Wallstraße 40, 23560 Lübeck, Germany
email: jakoby / reischuk @ informatik.mu-luebeck.de

parallelizing matrix multiplication by distributing the computation of the scalar products among the processors. At the beginning the data typically resides at a single location like an I/O-processor. Then, often about the same amount of time is spent to distribute the matrix elements among the processors by message passing as on the numerical computation itself. Thus, it is important to perform the data transmission as fast as possible. In previous work it was generally assumed that the transmission delay over a single link is proportional to the amount of data. Under this assumption a simple receive-and-forward strategy of small data chunks is able to generate optimal schedules. Making the additional assumption that a processor can simultaneously forward a data packet and receive the next one the problem becomes trivial.

In real systems, however, like Transputer based networks the set-up time for a data transmission cannot be neglected. Establishing a connection requires some overhead. Therefore, it is faster to send a few large data chunks than a large number of small packets. Our model defined below will take this property of real systems into account. For this more realistic model we develop optimal or nearly optimal schedules for arbitrary networks and also for several specific topologies that have been proposed as efficient networks.

In a similar way, the scheduling of multiprocessor systems has been extended to a more realistic scenario by including the communication delays between processors. If the predecessor of a task is executed on a different processor the time it will be available is given by the length of the task plus the time to transmit the task. As one would expect, constructing optimal schedules in this generalized model is $\mathcal{NP}$–hard in general. This already holds for very simple networks. On the other hand, efficient approximation schemes guaranteeing a factor 2 have been obtained. These results can be found in [PY88, JKS89, JR92].

The data distribution problem considered here called *scattering* is different from the *broadcasting* and *gossiping* problem, which have been studied extensively, see [SCH81, HHL88, LP88, HJM90, BHLP92, JRS94]. For scattering there is a different piece of information that has to be transmitted to each processor. Scattering has been studied for example in [SS89, JH89, FMR90, BBP93] for different computational models, either neglecting the set–up delay or allowing multiport communication of nodes.

### 1.2 The Communication Model

**Definition 1** *A network of processors with two-way communication links is modelled by an undirected connected graph $G = (V, E)$. The two parameters $\tau_s$, the set-up delay, and $\tau_d$, the transmission rate of a link, describe the communication capacity of the network. For simplicity, we may choose the time scale such that both parameters are integer values. The data transmitted within the network is modelled by a set $\mathcal{W}$ of packets. Packets have identical length and require link transmission time $\tau_d$. Thus, a single data exchange of $m$ packets over a link takes time*

$$t(m) \;=\; \tau_s + m \cdot \tau_d \;.$$

*We assume that processors exchange data in synchronous rounds in a blocking mode. Such a schedule will be specified by a set $S$ of triples*

$$((u,v),w,t) \in E \times 2^{\mathcal{W}} \times \mathbb{N} \ ,$$

*meaning that processors $u,v$ exchange data $w$, a subset of packets in $\mathcal{W}$, starting at round $t$. For each pair of triples $((u,v),w,t) \neq ((u',v'),w',t')$ in $S$ with $\{u,v\} \cap \{u',v'\} \neq \emptyset$ and $t \leq t'$ the following inequality has to hold:*

$$t' \ \geq \ t + t(|w|) \ = \ t + \tau_s + |w| \cdot \tau_d \ ,$$

*that is, different communication events may not interfere. The length of a schedule $S$ is given by*

$$\texttt{length}(S) \ := \ \max_{(e,w,t)\in S} t + t(|w|) \ .$$

*For a schedule $S$ and round $t$ the set of packets $\boldsymbol{\mathcal{R}_v(t)}$ received by processor $v$ is given by*

$$\mathcal{R}_v(t) \ := \ \begin{cases} \emptyset & \text{if } t = 0, \\ \mathcal{R}_v(t-1) \ \cup \ w & \text{if } ((u,v),w,t-t(|w|)) \in S, \text{ and} \\ \mathcal{R}_v(t-1) & \text{else .} \end{cases}$$

*Naturally, for each tuple $((v,u),w,t) \in S$ it is required that $v$ has already received all packets in $w$, that is $w \subseteq \mathcal{R}_v(t)$,*
*One distinguished processor $\bar{v} \in V$ serves as the* **data source**. *For the specific data transmission problem considered here the set $\mathcal{W}$ contains exactly one packet $p_v$ for each processor $v$. Thus the starting configuration at round $0$ can be described by setting $\mathcal{R}_{\bar{v}}(0) := \{p_v | v \in V\}$.*
*A schedule $S$ solves the data distribution problem for $G$ iff after termination of $S$ each processor has received its packet: $p_v \in \mathcal{R}_v(\texttt{length}(S))$ for all $v \in G$. Let us call the minimal length of a schedule for $G$ the* **data distribution time** *of $G$, denoted by $\texttt{DDTime}(G, \tau_s, \tau_d)$, where $\tau_s, \tau_d$ are the delay and capacity parameters.*
*A schedule $S$ is called* **canonical** *if*

- *each processor $v$ receives each packet at most once,*
- *if $v$ receives a packet $p_u \neq p_v$ it will forward $p_u$ to a neighbour,*
- *and each $v$ does not receive any more packets after it has received its own packet.*

Note that each schedule $S$ can easily be transformed into a canonical schedule $S'$ with $\texttt{length}(S') \leq \texttt{length}(S)$ .

## 2 Bounds for Arbitrary Graphs

We start by proving a general lower bound for arbitrary networks of size $n$, which we call the **delay-capacity bound** for the data distribution problem. It says that the set-up delay multiplied with the minimal graph theoretic radius for graphs of outdegree 2, which is about $\log_2 n$, *plus* the communication capacity of the source for the $n-1$ packets to be sent to the other processors, which is $(n-1)\cdot\tau_d$, cannot be beaten. The proof is based on the following combinatorial lemma. Define $\log n \;:=\; \lceil \log_2 n\rceil$ and $\overline{n} \;:=\; 2^{\log n}$.

**Lemma 1** *Given a sequence $a_1,\ldots,a_l$ of $l<\log n$ positive natural numbers that add up to at least $n-1$, then there exists an $i$ such that*

$$a_i \;+\; \sum_{j=1}^{i} a_j \;>\; n \qquad \text{and} \qquad \lceil \log a_i\rceil \geq \lceil \log n\rceil - i\ .$$

*Proof of Lemma 1:* Let $2^{k-1}<n\leq 2^k$ then $2^i\cdot\overline{a_i}\geq\overline{n}$ implies $a_i>2^{k-i-1}$. Assume that for all $i\in\{1,\ldots,l\}$ with $l<k$ it holds $a_i\leq 2^{k-i-1}$ then

$$\sum_{i=1}^{l} a_i < 2^{k-1} < n\ .$$

We conclude that there exists at least one $i$ with $a_i>2^{k-i-1}$. Choose a maximal $i_0$ with this property, then $\sum_{i=i_0+1}^{l} a_i < 2^{k-i_0-1}$ . Assume that $a_{i_0}+\sum_{i=1}^{i_0} a_i\leq n$ then $\sum_{i=1}^{i_0} a_i \leq n-a_{i_0} < n-2^{k-i_0-1}$ and therefore

$$\sum_{i=1}^{l} a_i = \sum_{i=1}^{i_0} a_i + \sum_{i=i_0+1}^{l} a_i < n-2^{k-i_0-1}+2^{k-i_0-1}-1 = n-1\ .$$

∎

In a similar way the following Lemma can easily be derived.

**Lemma 2** *Given a sequence $a_1,\ldots,a_k$ of positive natural numbers adding up to $2^k-1$ with the property that for each $i$ it holds $a_i \;+\; \sum_{j=1}^{i} a_j \;\leq\; 2^k$ and $a_i \;\leq\; 2^{k-i}$, then $a_i \;=\; 2^{k-i}$.*

**Theorem 1** *Any schedule for the data distribution problem on an $n$-processor network has length at least the delay-capacity bound*

$$\mathtt{DCB}(n,\tau_s,\tau_d) \;\;:=\;\; \log n\cdot\tau_s + (n-1)\cdot\tau_d\ .$$

*Proof:* Let $G=(V,E)$ be an $n$-processor network with parameters $\tau_s,\tau_d$. By the remark above it suffices to consider canonical schedules $S$ for $G$. For a node $v$ let

$$S_{\mathtt{send}}(v) \;=\; ((v,u_1),w_1,t_1),((v,u_2),w_2,t_2),\ldots$$

be all communication events with sender $v$ in chronological order at rounds $t_i$ relative to the round in which $w_1$ was sent, that means $0=t_1<t_2<\ldots$. Similarly for $w\subseteq\mathcal{W}$

$$S_{\mathtt{send}}(v,w) \;=\; ((v,u_1'),w_1',t_1'),((v,u_2'),w_2',t_2'),\ldots$$

denotes the subsequence of all events with $w \cap w'_i \neq \emptyset$, that means messages from $v$ containing packets in $w$. Note that for each $i$

$$t_i \geq \sum_{j=1}^{i-1} t(|w_j|) \;=\; (i-1)\cdot\tau_s + \sum_{j=1}^{i-1} |w_j|\cdot\tau_d \ ,$$

$$t'_i \geq \sum_{j=1}^{i-1} t(|w'_j|) \ ,$$

$$\mathtt{length}(S) \geq \sum_{j=1}^{i} t(|w_j|) \;+\; \mathtt{length}(S_{\mathtt{send}}(u_i, w_i)) \ .$$

We will show that the assumption

$$\mathtt{length}(S) \;<\; \log n\cdot\tau_s + (n-1)\cdot\tau_d$$

yields a contraction. Since

$$\mathtt{length}(S) \;\geq\; |S_{\mathtt{send}}(\bar{v})|\cdot\tau_s + (n-1)\cdot\tau_d$$

it must hold $|S_{\mathtt{send}}(\bar{v})| < \log n$. Lemma 1 applied to

$$S_{\mathtt{send}}(\bar{v}) \;=\; ((\bar{v}, u_1), w_1, t_1), \ldots \ ,$$

that is $n-1 = |\mathcal{W} - \{p_{\bar{v}}\}|$ and $a_i = |w_i|$ implies that there exists a tuple $((\bar{v}, u_i), w_i, t_i) \in S_{\mathtt{send}}(\bar{v})$ with

$$\log n\cdot\tau_s + (n-1)\cdot\tau_d \;<\; \log(2^i|w_i|)\cdot\tau_s + (\sum_{j=1}^{i}|w_j| + (|w_i|-1))\cdot\tau_d \ .$$

Combining this inequality with the ones for $\mathtt{length}(S)$ yields

$$\mathtt{length}(S_{\mathtt{send}}(v_i, w_i)) \;<\; \log(|w_1|)\cdot\tau_s + (|w_i|-1)\cdot\tau_d \ .$$

Since $S$ is a canonical schedule and

$$\mathtt{length}(S_{\mathtt{send}}(v_i, w_i)) \;\geq\; |S_{\mathtt{send}}(v_i, w_i)|\cdot\tau_s + (|w_i|-1)\cdot\tau_d$$

it holds $|S_{\mathtt{send}}(v_i, w_i)| < \log(|w_i|)$. Therefore, we can repeat this argument for the subschedule $S_{\mathtt{send}}(v_i, w_i)$ instead of the complete schedule $S$ to obtain a subschedule for a neighbour of $v_i$ that starts after message $w_i$ has been transmitted from $\bar{v}$ to $u_i$ and so on. But this process cannot go forever since $S$ is finite. ∎

Consider a 1-dimensional network of size $n$ with the source as one of the two endpoints. The obvious strategy sending each packet individually from the source to its destination beginning with the packet that has to travel the longest distance takes time exactly $(2n-3)\cdot(\tau_s+\tau_d)$. This network seems to be the worst topology for data distribution. It is not hard to show that any other network of the same size can achieve this transmission time.

**Lemma 3** *Every $n$-node network has a schedule of length $(2n-3)\cdot(\tau_s+\tau_d)$. Such a schedule can be constructed efficiently.*

For a 1-dimensional network with the source in the middle one can alternate the strategy described above between the two directions to achieve a time bound of $n\cdot(\tau_s+\tau_d)$. Thus, a more favourable placement of the source gives an improvement by a factor about 2.

In both cases, however, there is a better method by sending packets for neighbouring processors in pairs. The source at the endpoint starts by combining the two packets to the two most distant processors $v_{n-1}, v_n$ into a single message that is relayed to $v_{n-1}$, then the packets for $v_{n-3}, v_{n-2}$, and so on. Processor $v_{n-1}$ takes its packet and forwards the other packet to its neighbour $v_n$. This strategy yields a better time bound $(n-1)\cdot\tau_s+(2n-3)\cdot\tau_d$. It can be shown that this bound is the best one can achieve for such a chain. For 1-dimensional networks sending messages with an even larger number of packets does not give further improvements, on the contrary we get a degradation. If the source lies in the middle the time bound $\lfloor(n+1)/2\rfloor\cdot\tau_s+(n-1)\cdot\tau_d$ can be achieved.

**Theorem 2** *Let $L_n$ and $C_n$ denote a 1-dimensional processor chain of length $n$ with the source at one endpoint, resp. a circle of length $n$. Then,*

$$\begin{aligned}\mathtt{DDTime}(L_n,\tau_s,\tau_d) &= (n-1)\cdot\tau_s+(2n-3)\cdot\tau_d\ ,\\ \mathtt{DDTime}(C_n,\tau_s,\tau_d) &= \left\lfloor\frac{n+1}{2}\right\rfloor\cdot\tau_s+(n-1)\cdot\tau_d\ .\end{aligned}$$

Thus this simple example already indicates that a variety of strategies might be suitable for optimal data distribution and it is not that easy to determine the optimal ones.

The broadcast properties of a network provide another way to obtain a lower bound on its data transmission time. The broadcast problem is simpler than the data distribution problem consider here, it requires only that a single packed located at the source reaches every processor in the network. Thus there is only one packet and a processor may duplicate it to forward it to different neighbours, one at a time.

**Definition 2** *A broadcast schedule $B$ for a network $G=(V,E)$ with source $\bar{v}$ is a sequence of subsets of edges $E_1,E_2,\ldots,E_f$ with associated subset of processors $V_0=\{\bar{v}\}$ and $V_f=V$, where for $i>0$ $V_i\ :=\ V_{i-1}\cup\{\,v\mid(u,v)\in E_i$ and $u\in V_{i-1}\,\}$. $V_i$ are those processors that have received the broadcast information by round $i$. It is required that at each round a processor can forward this information to only one neighbour, that is $E_i\subseteq\{\,(u,v)\in E\mid u\in V_{i-1}\,\}$* and *$\forall u\in V_{i-1}$ : $|\,E_i\cap(\{u\}\times V)|\ \leq\ 1$ .*
*Define* $\mathtt{BCTime}(G)$ *as the minimal length $f$ of a broadcast schedule for $G$.*

When a data transmission schedule terminates each node of the network has received at least one packet from the source. Thus, from a data transmission schedule one can easily obtain a broadcast schedule. A message containing packets takes time at least $\tau_s + \tau_d$ for transmission over a link in the data distribution model, whereas the time scale for the broadcast problem are units. Therefore,

**Theorem 3** *For every network $G$ holds:*

$$\texttt{DDTime}(G, \tau_s, \tau_d) \;\geq\; \texttt{BCTime}(G) \cdot (\tau_s + \tau_d) \; .$$

There is another interesting difference between broadcasting a single piece of information and distributing data with set-up delays in a blocking mode. For every network $G$ one can achieve the minimal broadcasting time by choosing a suitable spanning tree of $G$. For the data distribution problem, however, a spanning tree may not be sufficient to achieve the optimal schedule length (see figure 1).

## 3 Networks Matching the Delay-Capacity Bound

In this section we will consider specific network topologies $G$ that are best suited for the data distribution problem with set-up delays. Such a $G$ has the property that

$$\texttt{DDTime}(G, \tau_s, \tau_d) = \texttt{DCB}(|G|, \tau_s, \tau_d) \; .$$

It turns out that a special family of trees are essential, the binomial trees [OW93].

**Definition 3** *Binomial trees are defined inductively as follows. The graph $\texttt{BT}(0)$ consists of a single node. For level $k > 0$*

$$\texttt{BT}(k) \;\; := \;\; (V_l \cup V_r, \{\{\rho_l, \rho_r\}\} \cup E_l \cup E_r) \; ,$$

*where $T_l = (V_l, E_l)$ and $T_r = (V_r, E_r)$ are two $\texttt{BT}(k-1)$–trees with roots $\rho_l$, resp. $\rho_r$. One of the two roots will serve as the root of $\texttt{BT}(k)$, let us choose the right one $\rho_r$.*

$\texttt{BT}(k)$ has $2^k$ nodes. The root, which serves as the source, has outdegree $k$.

**Theorem 4** *Binomial trees achieve the delay capacity bound, that is:*

$$\texttt{DDTime}(\texttt{BT}(k), \tau_s, \tau_d) \;=\; \texttt{DCB}(2^k, \tau_s, \tau_d) \;=\; k \cdot \tau_s + (2^k - 1) \cdot \tau_d \; .$$

*An optimal schedule can be constructed in linear time.*

*Proof:* The claim obviously holds for $k = 0, 1$. Let $T$ be a $\texttt{BT}(k+1)$–tree with source $\bar{v}$ consisting of $\texttt{BT}(k)$–subtrees $T_l = (V_l, E_l)$, $T_r = (V_r, E_r)$ with roots $\rho_l$, resp. $\rho_r = \bar{v}$. Let $S_l$ and $S_r$ be optimal schedules for these subtrees and $\mathcal{W}_l$ be the set of packets for $V_l$. Then

$$S \;\; := \;\; \{ \; ((\rho_r, \rho_l), \mathcal{W}_l, 0)\} \;\cup\; \{ \; (e, w, t + t(|\mathcal{W}_l|)) \mid (e, w, t) \in S_l \cup S_r \; \}$$

is a schedule for $T$ of

$$\texttt{length}(S) \;=\; \tau_s + |\mathcal{W}_l| \cdot \tau_d + k \cdot \tau_s + (2^k - 1) \cdot \tau_d \;=\; (k+1) \cdot \tau_s + (2^{k+1} - 1) \cdot \tau_d \ .$$

The lower bound of Theorem 1 implies that this schedule is optimal. This can also easily be seen directly because the root has to send $k+1$ messages to its $k+1$ neighbours, yielding a delay of $(k+1) \cdot \tau_s$. ∎

These trees have to occur as spanning trees in networks that achieve data distribution in optimal time.

**Theorem 5** *A network $G$ of size $2^k$ with source $\bar{v}$ achieves the delay-capacity bound iff $\texttt{BT}(k)$ can be embedded into $G$ such that its root is mapped onto $\bar{v}$.*

*Proof:* If such a tree spans $G$ it is obvious that the bound can be achieved. On the other hand, let $S$ be a schedule for $G$ with source $\bar{v}$ of length $k \cdot \tau_s + (2^k - 1) \cdot \tau_d$. Then $|S_{\texttt{send}}(\bar{v})| \leq k$. Let $S_{\texttt{send}}(\bar{v}) \;=\; ((\bar{v}, v_1), w_1, t_1), \ldots, ((\bar{v}, v_a), w_a, t_a)$, with $t_i < t_{i+1}$. Note that $w_1, \ldots, w_a$ is a partition of $\mathcal{W} \setminus \{p_{\bar{v}}\}$. Thus from Lemma 1 follows $|S_{\texttt{send}}(\bar{v})| = k$ and from Lemma 2 $|w_i| = 2^{k-i}$.
Since $t_i \geq (i-1) \cdot \tau_s + (2^k - 2^{k-i+1}) \cdot \tau_d$ the set $S_{\texttt{send}}(v_i, w_i)$ has to distribute a set of $2^{k-i}$ packets within

$$\texttt{length}(S_{\texttt{send}}(v_i)) \;\leq\; (k-i) \cdot \tau_s + (2^{k-1} - i) \cdot \tau_d$$

many steps. This is only possible if $p_{v_i} \in w_i$. Therefore, the claim follows inductively. ∎
It is not hard to see that this family of trees can be embedded into hypercubes. Thus,

**Corollary 1** *For the $d$–dimensional hypercube $H_d$ of size $2^d$ holds*

$$\texttt{DDTime}(H_d, \tau_s, \tau_d) \;=\; \texttt{DCB}(|H_d|, \tau_s, \tau_d) \;=\; d \cdot \tau_s + (2^d - 1) \cdot \tau_d \ .$$

**Definition 4** *Define a $l_1 \times l_2 \times \ldots \times l_d$–grid as the induced subgraph of the infinite $d$-dimensional integer grid restricted to nodes with $i$-th coordinate in the range $[0, l_i - 1]$. Nodes of degree $2d$ are called inner nodes.*

The embedding property still holds if one enlarges a hypercube in each dimension by a factor 2.

**Corollary 2** *Let $G_d$ be a $4 \times 4 \times \ldots \times 4$–grid of dimension $d$ of size $4^d$ with the source being an inner node. Then*

$$\texttt{DDTime}(G_d, \tau_s, \tau_d) \;=\; \texttt{DCB}(|G_d|, \tau_s, \tau_d) \;=\; 2d \cdot \tau_s + (4^d - 1) \cdot \tau_d \ .$$

## 4 $\mathcal{NP}$-Completeness Result

Next, we will investigate the complexity of the data distribution problem with set-up delays for arbitrary graphs. For the reduction one can use the 3–dimensional matching problem.

**Definition 5 3–Dimensional Matching (3DM)**
*Given three disjoint sets $A, B, C$ of equal size and a set $M \subseteq A \times B \times C$ of triples, decide whether $M$ contains a matching, i.e. whether there exists a subset $M' \subseteq M$ of size $|A|$ such that no two elements of $M'$ agree in any coordinate?*

The 3DM problem is known to be $\mathcal{NP}$–complete even if the cardinality of $A$ is a power of 2 and each element of $A, B, C$ occurs in at most three triples.
For a given instance $A, B, C, M$ of the 3DM problem we generate the following graph $G = (V, A)$:

- For each element $\alpha_i \in A$ let $A_i = (V_{\alpha_i}, E_{\alpha_i})$ be a $\mathtt{BT}(4)$–tree with root $\alpha_i \in V_{\alpha_i}$ and let $A = (V_A, E_A) = (\bigcup_{\alpha_i \in A} V_{\alpha_i}, \bigcup_{\alpha_i \in A} E_{\alpha_i})$.
- For each element $\beta_i \in B$ let $\beta_i'''$ be the root of a $\mathtt{BT}(1)$–tree $T_{\beta_i'''} = (V_{\beta_i'''}, E_{\beta_i'''})$ and let $T_{\beta_i} = (V_{\beta_i}, E_{\beta_i})$ be a $\mathtt{BT}(2)$–tree with root $r_{\beta_i}$ then define $B_i := (\{\beta_i, \beta_i', \beta_i''\} \cup V_{\beta_i'''} \cup V_{\beta_i}, \{\{\beta_i, r_{\beta_i}\}\} \cup E_{\beta_i'''} \cup E_{\beta_i})$ and let $B = (V_B, E_B) := (\bigcup_{\beta_i \in B} V_{\beta_i}, \bigcup_{\beta_i \in B} E_{\beta_i})$.
- For each element $\gamma_i \in C$ let $C_i = (V_{\gamma_i}, E_{\gamma_i})$ be a $\mathtt{BT}(2)$–tree with root $\gamma_i \in V_{\gamma_i}$ and let $C = (V_C, E_C) = (\bigcup_{\gamma_i \in C} V_{\gamma_i}, \bigcup_{\gamma_i \in C} E_{\gamma_i})$.
- For each element $\mu_i = (\alpha_i, \beta_i, \gamma_i) \in M$ let $M_i := (V_{M_i}, E_{M_i})$ with $V_{M_i} := V_{\alpha_i} \cup V_{\beta_i} \cup V_{\gamma_i} \cup \{\mu_i\}$ and

  $$E_{M_i} := E_{\alpha_i} \cup E_{\beta_i} \cup E_{\gamma_i} \cup \{\{\mu_i, \alpha_i\}, \{\mu_i, \beta_i\}, \{\mu_i, \beta_i'''\}, \{\mu_i, \gamma_i\}\} .$$

  For each element $\beta_i \in B$ let $M_{\beta_i} := (\mu_i, \mu_j, \mu_k) \subseteq M$ be the set of triples where $\beta_i$ occurs with $i < j < k$ then let $M := (V_M, E_M)$ with $V_M := \bigcup_{\mu_i \in M} V_{\mu_i}$ and

  $$E_M = \bigcup_{\mu_i \in M} E_{\mu_i} \cup \bigcup_{\beta_i \in B} \{\{\mu_i, \beta_i'\}, \{\mu_j, \beta_i'\}, \{\mu_k, \beta_i''\} \mid M_{\beta_i} = \{\mu_i, \mu_j, \mu_k\}\}$$

  Note that each node $\alpha_i$ has degree 7, each node $\mu_i$ and $\gamma_i$ has degree 5, and each node $\beta_i$ has degree 4.
- Let $V_T = \{\alpha_1', \ldots, \alpha_{|A|}'\}$, let $A_{\alpha_i'} = (V_{\alpha_i'}, E_{\alpha_i'})$ be a $\mathtt{BT}(5)$–tree with root $\alpha_i'$ and let $T_{\alpha_i'} = (V_{T_i'}, E_{T_i'})$ be a $\mathtt{BT}(6)$–tree with root $r_{\alpha_i'}$ for each $\alpha_i' \in V_T$. Since $|A| = 2^k$ there exists a $\mathtt{BT}(k)$–tree $A_T = (V_T, E_T)$ with root $\alpha_1'$. So let $G(A, B, C, M) = (V, E)$ with

  $$\begin{aligned} V &:= V_M \cup V_T \cup \bigcup_{i \in \{1, \ldots, |A|\}} \left(V_{\alpha_i'} \cup V_{T_i'}\right) \quad \text{and} \\ E &:= E_M \cup E_T \cup \bigcup_{i \in \{1, \ldots, |A|\}} \left(\{\{\alpha_i, \alpha_i'\}\} \cup E_{\alpha_i'} \cup \left\{\{\alpha_i', r_{\alpha_i'}\}\right\} \cup E_{T_i'}\right) . \end{aligned}$$

**Lemma 4** *The* $\mathtt{BT}(k+7)$*–tree with root* $\alpha_1$ *can be embedded into* $G(A,B,C,M)$ *iff* $M$ *contains a matching.*

**Theorem 6** *To decide whether an arbitrary network* $G$ *achieves the delay-capacity bound for the data distribution problem is* $\mathcal{NP}$*–complete.*

The proof follows from theorem 5 and the fact that $|V_M| = 16 \cdot |A| + 9 \cdot |B| + 4 \cdot |C| + |M| + 96 \cdot |A| = 125 \cdot |A| + 3 \cdot |A| = 2^7 \cdot 2^k = 2^{k+7}$.

**Corollary 3** *The problem to compute* $\mathtt{DDTime}(G, \tau_s, \tau_d)$ *for arbitrary networks* $G$ *is* $\mathcal{NP}$*-hard.*

## 5 Approximations

We have seen that a special family of trees achieves the delay-capacity bound. The complexity of the data distribution problem for arbitrary trees is unknown. However, we can at least give reasonable approximations to the optimal schedule length. For the purpose, two important subclasses of schedules will be considered in more detail.

### 5.1 One Block Schedule

**Definition 6** *We call a schedule a* **one–block schedule** *if each node receives at most one message.*

Note that the optimal schedule for the special trees described above is a one–block schedule. The same holds for hypercubes and the d–dimensional grid of size $4^d$. Thus networks that achieve the delay-capacity bound always have an optimal schedule with the one-block property. One–block schedules are easy to analyze. Let us first generalize the notion of binomial trees.

**Definition 7** *For* $d \in \mathbb{N}$, $d \geq 2$, *the family of* $\mathtt{BT}^{[\mathtt{d}]}$*–trees is defined as follows:*

$$\mathtt{BT}^{[\mathtt{d}]}(k) \;:=\; \begin{cases} (\{\rho\}, \{\}) & \text{if } k = 0, \\ (\{\rho\} \cup \bigcup_{j=0}^{k-1} V_j, \bigcup_{j=0}^{k-1} (E_j \cup \{\{\rho, \rho_j\}\}) & \text{if } k \leq d, \\ (\{\rho\} \cup \bigcup_{j=k-d}^{k-1} V_j, \bigcup_{j=k-d}^{k-1} (E_j \cup \{\{\rho, \rho_j\}\}) & \text{else,} \end{cases}$$

*where* $\rho$ *is the root of the* $\mathtt{BT}^{[\mathtt{d}]}(k)$*–tree and the* $(V_j, E_j)$ *are* $\mathtt{BT}^{[\mathtt{d}]}(j)$*–trees with root* $\rho_j$*.*

**Theorem 7** *For a* $\mathtt{BT}^{[\mathtt{d}]}(k)$*–tree holds*

$$\mathtt{DDTime}(T, \tau_s, \tau_d) \;\leq\; k \cdot \tau_s + \left( \left(1 + 2^{-\lfloor (d-1)/2 \rfloor}\right) \cdot |T| - \left\lceil \frac{k}{d} \right\rceil \right) \cdot \tau_d \;.$$

*Such a schedule can be constructed in linear time.*

Since $\log|T| \geq (k-2)/2$ this bound is at most twice the delay-capacity bound. Hence, this schedule approximates an optimal schedule for $\mathtt{BT}^{[\mathtt{d}]}(k)$–trees with a factor at most 2.

*Proof of Theorem 7:* For a $\mathtt{BT}^{[\mathtt{d}]}(k)$–tree $T = (V, E)$ with root $\rho$ let

$$T_{k-1}, T_{k-2}, \ldots, T_a$$

(with $a = \max\{k-d, 0\}$) be the $\mathtt{BT}^{[\mathtt{d}]}(j)$-subtrees of $T$ rooted at the sons $\rho_j$ of $\rho$. Let $\mathcal{W}_j$ be the subset of packets for $T_j$ and $S$ be the schedule

$$S_{\mathtt{send}}(\rho) \;=\; \{((\rho, \rho_j), \mathcal{W}_j, t_j)\}$$

with $t_j \;=\; \sum_{l=0}^{j-1} t(|\mathcal{W}_l|)$. The subschedules for these subtrees are defined recursively the same way. After $d \cdot \tau_s + (|V|-1) \cdot \tau_d$ steps only processors in subtrees of $\mathtt{BT}^{[\mathtt{d}]}(k-d)$–trees are still lacking their packet. Thus, $\mathtt{length}(S) = L(k)$ with

$$L(k) \;=\; \begin{cases} d \cdot \tau_s + (|T_k| - 1) \cdot \tau_d + L(k-d) & \text{if } k \geq d, \\ k \cdot \tau_s + (|T_k| - 1) \cdot \tau_d & \text{else,} \end{cases}$$

where $T_k$ is a $\mathtt{BT}^{[\mathtt{d}]}(k)$–tree. Since

$$|T_k| - 1 \;=\; \begin{cases} \sum_{l=0}^{d-1} |T_{k-d+l}| & \text{if } k \geq d, \\ \sum_{l=0}^{k-1} |T_l| & \text{else,} \end{cases}$$

we get

$$\begin{aligned} L(k) = k \cdot \tau_s + \sum_{j=0}^{k-1} |T_j| \cdot \tau_d \;&\leq\; k \cdot \tau_s \;+ \sum_{j=0}^{\lceil k/d \rceil - 1} (|T_{k-j\cdot d}| - 1) \cdot \tau_d \\ &\leq k \cdot \tau_s + \left( (|T_k| + |T_{k-d+1}|) - \left\lceil \frac{k}{d} \right\rceil \right) \cdot \tau_d \\ &\leq k \cdot \tau_s + \left( \left(1 + 2^{-\lfloor (d-1)/2 \rfloor}\right) \cdot |T_k| - \left\lceil \frac{k}{d} \right\rceil \right) \cdot \tau_d \;. \end{aligned}$$

∎

Next let us consider balanced binary trees and generalizations of such trees.

**Definition 8** *Let* $\mathtt{leaves}(T)$ *denote the number of leaves of a tree* $T$ *and* $0 < \alpha < 1$ *be a rational number. A* $\mathtt{BB}(\alpha)$*–tree is a binary rooted tree* $T$ *such for each internal node* $v$ *its left and right subtrees* $T_l, T_r$ *are of about the same size, that is:*

$$\begin{aligned} \alpha \cdot (1 + |\mathtt{leaves}(T_l)| + |\mathtt{leaves}(T_r)|) \;&\leq\; |\mathtt{leaves}(T_l)| \\ &\leq\; (1-\alpha) \cdot (1 + |\mathtt{leaves}(T_l)| + |\mathtt{leaves}(T_r)|) \;. \end{aligned}$$

$T$ *is a* $\mathtt{BV}(d, \alpha)$*–tree if it is either a single node, or its root has* $k \leq d$ *subtrees* $T_1, \ldots, T_k$ *and for each* $T_i$ *holds:* $T_i$ *is a* $\mathtt{BV}(d, \alpha)$*–tree with* $|T_i| \leq (1-\alpha) \cdot |T|$.

For balanced trees there are good approximations for the data distribution problem.

**Theorem 8** *For* $\texttt{BB}(\alpha)$*–trees one can efficiently construct a schedule of length at most*

$$2 \cdot \log_{\frac{1}{1-\alpha}} \left( \frac{|V|+1}{2} \right) \cdot \tau_s + \left( \frac{|V|-1}{\alpha} - 2 \cdot \log_{\frac{1}{1-\alpha}} \left( \frac{|V|+1}{2} \right) \right) \cdot \tau_d \ .$$

**Corollary 4** *A complete or almost complete binary tree of size* $n$ *has a simple schedule for the data distribution problem of length at most*

$$2 \cdot (\log(n+1) - 1) \cdot \tau_s + (2n - 2\log(n+1)) \cdot \tau_d \ \leq \ \texttt{DCB}(n, \tau_s, \tau_d) \ .$$

**Theorem 9** *A* $\texttt{BV}(d, \alpha)$*–tree of size* $n$ *has a schedule constructable in polynomial time of length at most*

$$d \cdot \log_{\frac{1}{1-\alpha}} n \cdot \tau_s + \left( \frac{n-1}{\alpha} \ - \ \log_{\frac{1}{1-\alpha}} n \right) \cdot \tau_d \ .$$

We can, however, show that not all graphs have an optimal schedule that is a one–block schedule. For example, a 1-dimensional chain of length $n \geq 4$ with the source being one of its endpoints requires length $(n-1) \cdot \tau_s + \frac{n \cdot (n-1)}{2} \cdot \tau_d$ . Again, one can show that this topology exhibits the worst case for one–block schedules.

**Theorem 10** *An arbitrary connected network* $G$ *of size* $n$ *has a one–block schedule of length*

$$(n-1) \cdot \tau_s + \frac{n \cdot (n-1)}{2} \cdot \tau_d \ .$$

### 5.2 OLBS–Schedule

How can one-block schedules be generalized to allow more flexible and efficient strategies? Considering again the example of 1-dimensional chains we notice that a processor forwards a bunch of messages each containing more than one packet and that at most the last message is split into smaller pieces. The other messages are relayed without any change. Let us formalize this observation as follows.

**Definition 9** *A schedule* $S$ *has the* **only last block split property** *iff* $S$ *is a canonical schedule and for each node* $v$ *excluding the source holds:* $((v,u),w,t) \in S$ *implies that* $S$ *contains a tuple* $((u',v),w',t')$ *such that either* $w' = w$ *or that this tuple is the last receive event of* $v$*. We call such a schedule an* **olbs–schedule**.

An $\boldsymbol{l_1 \times l_2 \times \ldots \times l_d}$**–torus** is obtained from an $l_1 \times l_2 \times \ldots \times l_d$–grid by connecting the corresponding outer boundaries in each dimension.

**Theorem 11** *A* $l_1 \times l_2 \times \ldots \times l_d$ $d$*–dimensional torus* $T_d$ *has a lower bound*

$$\texttt{DDTime}(T_d, \tau_s, \tau_d) \ \geq \ \max \left\{ (|T_d| - 1) \cdot \tau_d, \ \sum_{i=1}^{d} \left\lfloor \frac{l_i}{2} \right\rfloor \cdot \tau_s \right\} .$$

*Proof:* The lower bound $(|T_d| - 1) \cdot \tau_d$ is trivial. The second bound follows from the observation that each node of $T_d$ has other nodes in distance $\sum_{i=1}^{d} \lfloor \frac{l_i}{2} \rfloor$. ∎
Now we will prove an almost matching upper bound even for grids without wrap around. The bound is basically the sum of the two arguments of the max operator, and thus at most twice as large.

**Theorem 12** *The data distribution problem on an $l_1 \times l_2 \times \ldots \times l_d$–grid $G_d$ with source in the center, that means radius $\sum_{i=1}^{d} \lfloor \frac{l_i}{2} \rfloor$, can be solved in time*

$$(|G_d| - 1) \cdot \tau_d \; + \; \sum_{i=1}^{d} \left\lceil \frac{l_i}{2} \right\rceil \cdot \tau_s \; .$$

However, such a strategy is not always sufficient, see figure 2.

**Theorem 13** *There exists trees such that olbs–schedules do not achieve optimal time.*

## 6 Conclusions

We have defined and analysed a more realistic model for the distribution of data in processor networks with message passing in a blocking mode. It takes into account the set-up delay for a communication event between two neighbouring processors. This problem is trivial if one neglects the set-up delays and allows nonblocking information exchange.
Even synchronizing a network without set-up delays, but which operates in a blocking mode, turns out to be a nontrivial task. In [J95] we have analysed this model and determined the optimal schedule length. Optimal schedules can be constructed for arbitrary such networks in polynomial time, but this seems to require a detailed analysis of the network topology.

## References


[BBP93] S. N. Bhatt, G. Bilardi, G. Pucci, A. Ranade, A. L. Rosenberg and E. J. Schwabe, *On Bufferless Routing of Variable Length Messages in Leveled Networks*, 1st European Symp. Algorithms, 1993.

[BPRR93] S. N. Bhatt, G. Pucci, A. Ranade and A. L. Rosenberg, *Scattering and Gathering Messages in Networks of Processors*, IEEE Trans. Computers 42, 1993, 938-949.

[BHLP92] J.-C. Bermond, P. Hell, A. Liestman, and J. Peters, *Broadcasting in Bounded Degree Graphs*, SIAM J. Disc. Math. 5, 1992, 10-24.

[FMR90] P. Fraigniaud, S. Miguet and Y. Robert, *Complexity of Scattering on a Ring of Processors*, Parallel Computing 13, 1990, 377-383.

[HHL88] S. Hedetniemi, S. Hedetniemi, and A. Liestman, *A Survey of Gossiping and Broadcasting in Communication Networks*, Networks 18, 1988, 319-349.

[HJM90] J. Hromkovič, C.–D. Jeschke, and B. Monien, *Optimal Algorithms for Dissemination of Information in Some Interconnection Networks*, Proc. 15th MFCS, 1990, 337-346.

[J95] A. Jakoby, *Optimal Data Distribution in Blocking Networks*, Technical Report Universität Lübeck, in preparation.

[JH89] S. L. Johnsson and C.-T. Ho, *Optimal Broadcasting and Personalized Communication in Hypercubes*, IEEE Trans. Computers 38, 1989, 1249-1268.

[JKS89] H. Jung, L. Kirousis, P. Spirakis, *Lower Bounds and Efficient Algorithms for Multiprocessor Scheduling of DAGs with Communication Delays*, Proc. 1st SPAA, 1989, 254 - 264.

[JR92] A. Jakoby, R. Reischuk, *The Complexity of Scheduling Problems with Communication Delays for Trees*, Proc. 3rd SWAT, 1992, 165-177.

[JRS94] A. Jakoby, R. Reischuk, C. Schindelhauer, *The Complexity of Broadcasting in Planar and Decomposable Graphs,* Proc. 14th International Workshop on Graph-Theoretic Concepts in Computer Science, Herrsching, June 1994.

[LP88] A. Liestman and J. Peters, *Broadcast Networks of Bounded Degree*, SIAM J. Disc. Math. 4, 1988, 531-540.

[OW93] T. Ottmann and P. Widmayer, *Algorithmen und Datenstrukturen*, BI Wissenschaftsverlag, 1993.

[PY88] C. Papadimitriou and M. Yannakakis, *Towards an Architecture-Independent Analysis of Parallel Algorithms*, Proc. 20th STOC, 1988, 510-513, see also SIAM J. Comput. 19, 1990, 322-328.

[SCH81] P. Slater, E. Cockayne, and S. Hedetniemi, *Information Dissemination in Trees*, SIAM J. Comput. 10, 1981, 692-701.

[SS89] Y. Saad and M. H. Schultz, *Data Communication in Parallel Architectures*, Parallel Computing 11, 1989, 131-150.

## Figures

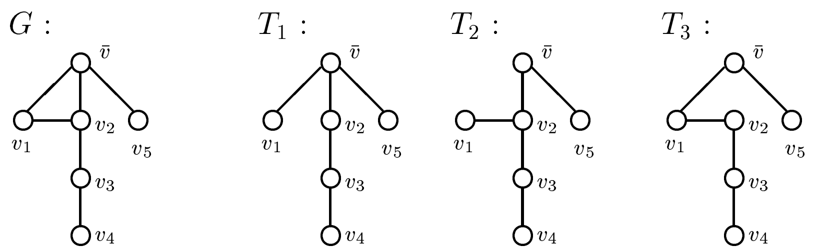


**Fig. 1.** A network $G$ for which the data cannot be distributed in minimal time in a tree-like fashon, that means restricting to one of its spanning trees $T_i$.

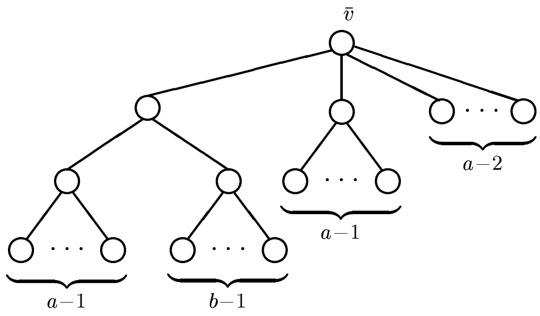


**Fig. 2.** The tree $T_{a,b}$: for $a = 6$, $b = 4$, $\tau_s = 4$ and $\tau_d = 2$ there is no OLBS–schedule that achieves minimal length.

This article was processed using the LaTeX macro package with LLNCS style