Relations between Communication Complexity Classes
(extended abstract)

Bernd Halstenberg and Rüdiger Reischuk*

Institut für Theoretische Informatik, Technische Hochschule Darmstadt
Alexanderstraße 24, D - 6100 Darmstadt, West-Germany
REISCHUK@DDADV1.EARN

ABSTRACT: We study the complexity of communication between two processors in terms of complexity classes as introduced in [1], where Babai et al. showed some analogies between Turing machine classes like \( P, NP, \Sigma_k \) etc. and the corresponding communication complexity classes \( C-P, C-NP, C-E_k \) etc. We enlarge this correspondence by showing that \( C-\Sigma_k \equiv C^k+1 \) and giving an equivalent characterization of \( C-PP \) by polylogarithmic probabilistic protocols with moderately bounded error (thus solving a problem suggested in [1]). In contrast to the case of TM complexity, we are able to show some proper inclusions like \( C-PP \subset C-\#P \). The nonuniformity of communication protocols allows us to show that the Boolean communication hierarchy does not collapse.

For completeness an overview on communication complexity classes is added with proofs of some properties already observed by other authors.

1. Introduction

In [2] we have studied different measures for the communication complexity of discrete functions \( f : X_0 \times X_1 \to Y \) when computed by two processors, \( P_0 \) and \( P_1 \), where initially \( P_i \) knows the value of \( z_i \in X_i \). In this paper we will examine communication complexity from a different point of view, namely we consider communication complexity classes of languages \( L \) corresponding to sequences \( (f_n)_{n \in \mathbb{N}} \) of functions \( f_n : \{0,1\}^n \times \{0,1\}^n \to \{0,1\} \). Notice that the notion of "language" in this context is quite different from that in the sequential case since we do not require the languages \( L \) to be decidable. Neither do we require the sequence of algorithms solving the problem \("(x,y) \in L ?\)" to be constructable, although, for some natural languages the upper bounds are obtained by uniform families of algorithms.

For convenience let denote
\[ \{0,1\}^* := \{(x,y) \mid x, y \in \{0,1\}^n, \ |x| = |y| \} \]
the set of all pairs of 0/1-strings of equal length and denote its powerset by
\[ 2\{0,1\}^* \]
We identify a set in \( 2\{0,1\}^* \) with its characteristic function
\[ f : 2\{0,1\}^* \to \{0,1\} \]
and consider \( f \) as a sequence \( (f_n)_{n \in \mathbb{N}} \) of functions \( f_n : \{0,1\}^n \times \{0,1\}^n \to \{0,1\} \), where \( f_n(x,y) = f(x,y) \) for all \( x, y \in \{0,1\}^n \). In the following, \( n \) means the length of the inputs if not specified otherwise.

The computational model can briefly be described as follows. For more details the reader is referred to [2].

Two processors, \( P_0 \) and \( P_1 \), get inputs \( x_0 \in X_0 \) and \( x_1 \in X_1 \), respectively. In order to compute \( f \), they alternately exchange messages according to some (deterministic, nondeterministic, or probabilistic) algorithm \( A \); finally, one of them computes the result depending only on its input and the messages that have been exchanged.

The messages exchanged during a computation of \( A \) are strings \( m_1, \ldots, m_r \in \{0,1\}^* \), where \( r \) is the number of rounds of that computation. The messages \( m_i \) with odd (even) \( i \) are those sent by \( P_0 (P_1) \). We require the algorithm \( A \) to generate "self delimiting" messages, i.e., for any two computations \( (m_1, \ldots, m_r) \) and \( (m'_1, \ldots, m'_r) \) (possibly padded with empty strings to have the same number of rounds) and every \( i \in \{1, \ldots, r\} \), the concatenation \( m_1 \ldots m_i \) must not be a proper prefix of \( m'_1 \ldots m'_i \).

* Research supported by DFG grant RE 672/1-1
A deterministic algorithm, also called protocol, can be specified by two transmission functions \( \phi_i : X_i \times \{0,1\}^* \to \{0,1\} \) and the partial output functions \( a_i : X_i \times \{0,1\}^* \to Y, i \in \{0,1\} \). For \( k := k \mod 2 \), \( \phi_k(x_k, w_1 \ldots w_k) \) is the message sent by processor \( P_k \) if the messages \( w_1, \ldots, w_k \) have already been exchanged. If for some \( k \geq 0 \), \( a_k(x_k, w_1 \ldots w_k) \) is defined this is the result of the computation. To guarantee prefix freeness it may be necessary that in this case one additional round is needed to inform \( P_{k+1} \) about termination.

For a deterministic protocol \( A \), let \( l(A) \) denote its length or communication complexity, that is the maximal number of bits exchanged during a computation of \( A \) for any pair of inputs. \( A \) computes the function \( f \) if for all \( (x_0, z_1) \in X_0 \times X_1 \) the computation of \( A \) on \( (x_0, z_1) \) is finite and the result equals \( f(x_0, z_1) \). Then the deterministic communication complexity of \( f \) is defined as

\[
C_{\text{det}}(f) := \min \{ l(A) \mid \text{A computes } f \}.
\]

Probabilistic protocols may be defined differently. Yao used transmission functions \( \phi_i : X_i \times \{0,1\}^* \times \{0,1\}^* \to \{0,1\} \) and output functions \( a_i : X_i \times \{0,1\}^* \times Y \to \{0,1\} \) to denote probability distributions on the set of possible messages, resp. possible decision values ([3], [4]). \( \phi_k(x_k, w_1 \ldots w_k, w) \) is the probability that message \( w \) will be sent by \( P_k \) if the messages \( w_1, \ldots, w_k \) have already been exchanged. \( a_k(x_k, w_1 \ldots w_k, y) \) (if defined) is the probability that the computation will stop with result \( y \) after exchanging messages \( w_1, \ldots, w_k \). Then the probability that the result on input \( (x_0, z_1) \) equals \( y \) is

\[
\sum_{k \in \mathbb{N}} \sum_{w_1, \ldots, w_k \in \{0,1\}^*} a_k(x_k, w_1 \ldots w_k, y) \cdot \prod_{j=0}^{k-1} \phi_j(x_j, w_1 \ldots w_j, w_{j+1}).
\]

For a probabilistic algorithm \( A \) let \( l(A) \) denote the maximal length of the computations of \( A \) on input \( (x_0, z_1) \), maximized over all inputs \( (x_0, z_1) \in X_0 \times X_1 \). \( \epsilon(A, f) \) (or \( \epsilon(A) \) if \( f \) is fixed) denotes the error probability of \( A \) with respect to \( f \) for the worst input, that is

\[
\epsilon(A, f) = \max_{(x_0, z_1)} P \left( A \text{ on } (x_0, z_1) \text{ yields some } y \neq f(x_0, z_1) \right).
\]

Then, for \( 0 < \epsilon < \frac{1}{2} \), we call

\[
C_\epsilon(f) := \min \{ l(A) \mid A \text{ is a probabilistic algorithm with } \epsilon(A, f) < \epsilon \}
\]

the (worst case) probabilistic communication complexity of \( f \) for error probability less than \( \epsilon \).

For boolean-valued functions \( f \), we also consider nondeterministic algorithms \( A \) accepting the language defined by \( f \), \( L(f) := f^{-1}(\{1\}) \). Nondeterministic algorithms accepting \( L(f) \) are like deterministic ones, but with different alternatives how to continue a computation. There exists an accepting computation (output 1) of \( A \) on input \( (x_0, z_1) \) if and only if \( (x_0, z_1) \in L(f) \); let \( l'(f) \) denote the minimum length of such an accepting computation, maximized over all inputs \( (x_0, z_1) \in L(f) \). We call

\[
C_{\text{Ldet}}(f) := \min \{ l'(A) \mid A \text{ accepts } L(f) \}
\]

the nondeterministic communication complexity of the language defined by \( f \), and we denote the complexity of its complement by

\[
C_{\text{Lndet}}(f) := C_{\text{Ldet}}(\bar{f})
\]

where \( \bar{f} := 1 - f \) is the complement function of \( f \).

For all the models defined above the corresponding \( r \)-round communication complexities are defined as the complexities of algorithms that use at most \( r \) rounds of communication; these complexities will be denoted by \( C_{r,\text{det}}, C_{r,\text{det},t}, \) etc.

As a generalization of the nondeterministic complexities \( C_{L\text{ndet}}, \) and \( C_{L\text{ndet}} \) Babai et al. ([1]) have defined the alternating communication complexity for boolean-valued functions \( f \in 2^{\{0,1\}^*}. \) One possible formal definition is as follows.

A \( k \)-alternating algorithm \( A \) is given by a sequence \( l_1, \ldots, l_k \) of integers for each \( n, \sum_{i=1}^k l_i := l(A) \), and two decision functions \( \phi, \psi : \{0,1\}^n \times \{0,1\}^{l(A)} \to \{0,1\} \). \( A \) computes \( f \) if

\[
f(x, y) = 1 \iff \exists u_1 \in \{0,1\}^{l_1} \forall u_2 \in \{0,1\}^{l_2} \ldots \forall u_k \in \{0,1\}^{l_k} : \phi(x, u) \circ \psi(u, y),
\]

where \( u = u_1 \ldots u_k \), and \( Q = \emptyset \) and \( Q = \emptyset \) if \( k \) is even, \( Q = \emptyset \) and \( Q = \emptyset \) otherwise. Then

\[
C_{\text{alt},k}(f) := \min \{ l(A) \mid A \text{ is a } k\text{-alternating algorithm which computes } f \}.
\]
2. Communication Complexity Classes

Definition: For any function \( T : \mathbb{N} \rightarrow \mathbb{N} \) we define the following communication complexity classes. \( f \) will always be (the characteristic function of) a subset of \([0,1]^*\).

\[
DComm(T) := \{ f | C_{det}(f_n) \leq T(n) \}
\]

For any function \( k : \mathbb{N} \rightarrow \mathbb{N} \),

\[
A_kComm(T) := \{ f | C_{alt,k}(f_n) \leq T(n) \}
\]

and

\[
coA_kComm(T) := \{ f | \neg C_{alt,k}(f_n) \leq T(n) \}
\]

\[
NComm(T) := A_1Comm(T)
\]

\[
coNComm(T) := coA_1Comm(T)
\]

\[
C \in \mathbb{N}
\]

\[
R_kComm(T) := \{ f | C_{(n)}(f_n) \leq T(n) \}
\]

Most important will be the following sets of error probabilities

\[
E_{const} := \{ \epsilon | \epsilon(n) = \epsilon \text{ for some } 0 < \epsilon < \frac{1}{2} \}
\]

\[
E_{plog} := \{ \epsilon | \epsilon(n) = \frac{1}{2} - 2^{-\alpha n} \text{ for some } \alpha \in \mathbb{N} \}
\]

\[
E_{eq} := \{ \epsilon | \epsilon(n) < \frac{1}{2} \}
\]

which yield

\[
R_kComm(T) := \bigcup_{\epsilon \in E_{const}} R_kComm(T),
\]

\[
RMComm(T) := \bigcup_{\epsilon \in E_{plog}} R_kComm(T),
\]

\[
RUComm(T) := \bigcup_{\epsilon \in E_{eq}} R_kComm(T).
\]

For any sequence \( g \) of functions \( g_n : \mathbb{N} \times \mathbb{N} \rightarrow \{0,1\} \),

\[
\#RelComm(g,T) := \{ f | \exists \text{ nondeterministic algorithms } A_n \text{ of length } \leq T(n) : 
\]

\[
f_n(x,y) = g_n(a_A(x,y),r_A(x,y)),
\]

where \( a_A(x,y) \) resp. \( r_A(x,y) \) is the number of accepting resp. rejecting computations of \( A_n \) on input \((x,y)\) and the "length" of \( A_n \) means the maximal number of bits exchanged in any computation on any input of length \( n \).

Since, for any \( f \), the deterministic complexity is bounded by \( C_{det}(f) \leq n \), all the classes defined above are equal to \( 2^{(0,1)^*} \) if \( T(n) \geq n \). Hence, of interest are only sublinear bounds on \( T \). In particular we are interested in polylogarithmic communication complexity classes as defined in [1]. This may be viewed as an analogue to the classical polynomial-exponential time hierarchy. In that case the problem size is measured by the length of the input and all \( NP \)-complete problems are solvable by exhaustive search with exponential effort. Here the problem size is the amount of information to code a single arbitrary input bit. Exhaustive search then corresponds to a complete transfer of all bits.

Definition: Let

\[
PLOG := \{ T : \mathbb{N} \rightarrow \mathbb{N} | \exists a \in \mathbb{N} : T(n) \leq O(\log^a n) \}
\]

We define the following communication complexity classes.

\[
C-P := DComm(PLOG).
\]

\[
C-NP := NComm(PLOG).
\]

\[
C-coNP := coNComm(PLOG).
\]

\[
C-BPP := RBComm(PLOG).
\]

\[
C-MA := RMComm(PLOG).
\]

\[
C-UP := RUComm(PLOG).
\]

For any fixed \( k \in \mathbb{N} \),

\[
C_{\Sigma_k} := A_kComm(PLOG)
\]

and

\[
C-\Pi_k := coA_kComm(PLOG).
\]

\[
C-PSpace := A(\log n)Comm(PLOG).
\]

\[
C-PP := \#RelComm(gt, PLOG),
\]

where \( gt(a,r) = 1 \) iff \( a > r \).

\[
C-#P := \#RelComm(ev, PLOG),
\]

where \( ev(a,r) = 1 \) iff \( a \) is even.

Remark: One may give another characterization of \( C_{\Sigma_k} \) and \( C-\Pi_k \) which seems to be more natural and is equivalent to the previous one for \( k \geq 1 \): Let \( C_{\Sigma_0} := C-P \). A language \( L \) is in \( C_{\Sigma_k} \) if there exists some polylogarithmic function \( f : \mathbb{N} \rightarrow \mathbb{N} \) and a function \( L' : \{0,1\}^* \rightarrow C-\Pi_{k-1} \) such that

\[
(x_0,x_1) \in L \iff \exists u \in \{0,1\}^{f(n)} : (x_0,x_1) \in L'(u).
\]

Following [1], we define \( C-\#P \) as the set of all functions \( f : \{0,1\}^{(0,1)*} \rightarrow \mathbb{N} \), for which there exists some nondeterministic communication protocol \( A \), such that the number of bits exchanged during any computation of \( A \) on inputs of length \( n \) is at most polylogarithmic in
In this chapter we study closure properties of communication complexity classes under different operations.

Clearly, for any function $T : \mathbb{N} \to \mathbb{N}$, $DComm(T)$ is closed under complementation. The same holds for $RComm(T)$ for any function $\epsilon : \mathbb{N} \to \{0,1\}$. This implies that $C-P$, $C-BPP$, $C-MPP$, $C-P^{C-#P}$, and $C-UPP$ are closed under complementation. Also, it is easy to see that $C-P$, $C-\Sigma_2P$, and $C-\Sigma_2P$ are closed under complementation.

On the other hand, it is well known that $C-NP = C-L_1$ and $C-coNP = C-P_1$ are not closed. The $n$-bit identification function $id_n$ has nondeterministic complexity $C_{L_n}det(id_n) = n$ while $C_{coL_n}det(id_n) \leq [\log n] + 1$.

The question remains open whether $C_k \Sigma_k$ is closed under complementation for $k \geq 2$, that means whether $C_k \Sigma_k = C_k \Pi_k$.

Next consider intersection and union of languages. Of course, for complexity classes $C$ which are closed under complementation, we only have to consider one of these operations.

**Theorem 1:**

1. $L_0, L_1 \in DComm(T) \implies L_0 \cap L_1, L_0 \cup L_1 \in DComm(2T + 1).

   \begin{align*}
   i) & \quad L_0, L_1 \in DComm(T) \implies L_0 \cap L_1, L_0 \cup L_1 \in DComm(2T + 1). \\
   ii) & \quad L_0, L_1 \in RBCOMM(T) \implies L_0 \cap L_1, L_0 \cup L_1 \in RBCOMM(2T). 
   \end{align*}

   **Corollary:** $C-P$ and $C-BPP$ are closed under intersection and union.

   We will see later that also $C-P^{C-#P}$ is closed under intersection and union. The question whether $C-P$ and $C-UPP$ are closed under $\cap$ and $\cup$ remains open.

**Theorem 2:** For any function $k : \mathbb{N} \to \mathbb{N}$, $L_0, L_1 \in A_k Comm(T) \implies L_0 \cup L_1 \in A_k Comm(2T + 1)$ and $L_0 \cap L_1 \in A_k Comm(4T + 1)$.

   **Corollary:** $C_k \Sigma_k$, $C_k \Pi_k$, and $C-kPSpace$ are closed under intersection and union.

**Theorem 3:** $C-\Sigma_2P$ is closed under intersection and union.

Another interesting operation is “joining” two languages such that either of them can be selected by a separate part of the input.

**Definition:** For any function $T : \mathbb{N} \to \mathbb{N}$ and any set $C$ of languages or functions let $DComm^C(T)$ denote the class of languages $L'$ for which there exists some oracle $L \in C$ such that there exists some oracle protocol with complexity at most $T$. Also, let $C-P^C = \bigcup_{T \in PLOG} DComm^C(T)$. If $C = \{L\}$ we also write $DComm^L(T)$ and $C-P^L$, respectively.

### 3. Closure Properties

In this chapter we study closure properties of communication complexity classes under different operations.
Corollary: $C-P^C#P$ is closed under intersection and union.

Proof: A deterministic algorithm which accepts $L_0 \cap L_1$ for $L_0, L_1 \in C-P^C#P$ may use the $C#$P-oracle for $L_0 \& L_1$ to determine both results and then deterministically computes the logical "and" resp. "or".

The last operation we will consider is taking the symmetrical difference $L \triangle L' := (L - L') \cup (L' - L)$ of two languages $L$ and $L'$ (corresponding to addition mod 2 of their characteristic functions).

Lemma 2: $C-P$, $C-P$Space, $C-BPP$, $C-P^C#P$, and $C-#2P$ are closed under $\triangle$.

Proof: The mentioned classes are closed under complementation, intersection, and union.

Lemma 3: $C-NP$ and $C-coNP$ are not closed under taking the symmetrical difference of two languages.

Proof: Any class $C$ which contains the trivial language $\{0,1\}^\ast$ and is closed under taking the symmetrical difference of two languages is also closed under complementation.

Lemma 4: $C-MPP$ and $C-UPP$ are closed under $\triangle$.

Proof: Let $C \in \{C-MPP, C-UPP\}$ and $f_0, f_1 \in C$. In order to compute $f_0 \oplus f_1$, $P_0$ and $P_1$ compute both functions and add the results mod 2. Let $\epsilon_i(x, y)$ be the error probability for computing $f_i(x, y)$ and $\epsilon(x, y)$ the error probability for computing $f_0 \oplus f_1(x, y)$ this way. Then

$$
\epsilon(x, y) = \epsilon_0(x, y) \cdot (1 - \epsilon_1(x, y)) + (1 - \epsilon_0(x, y)) \cdot \epsilon_1(x, y) \\
= \frac{1}{2} - 2 \cdot \left( \frac{1}{2} - \epsilon_0(x, y) \right) \cdot \left( \frac{1}{2} - \epsilon_1(x, y) \right).
$$

Hence, $f_0 \oplus f_1 \in C$.

We summarize the above results in the following table where closure is indicated by "✓" and non-closure by "✗". A "?” has been placed where the result is not known.

<table>
<thead>
<tr>
<th>Class \ Operation</th>
<th>compl.</th>
<th>$\cap$</th>
<th>$\cup$</th>
<th>$\triangle$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C-NP \cap C-coNP$</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>$C-NP$, $C-coNP$</td>
<td></td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>$C-NP \cup C-coNP$</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>$C-\Sigma_k \cap C-\Pi_k$</td>
<td>✗</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>$C-\Sigma_k$, $C-\Pi_k$</td>
<td></td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>$C-\Sigma_k \cup C-\Pi_k$</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
</tr>
<tr>
<td>$C-PSpace$</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>$C-BPP$</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>$C-PF$, $C-#2P$</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

1) equal to $C$ (see Theorem 4)  
2) $k \geq 2$  
3) see Theorem 12

Remark: It is easy to see that the closure results for polylogarithmic complexity classes under union and intersection may be extended to polylogarithmically bounded union and intersection, that is: For $C \in \{C-P, C-\Sigma_k, C-\Pi_k, C-PSpace, C-BPP, C-P^C#P, C-#2P\}$, any sequence of languages $\{L_1, L_2, \ldots\} \subseteq C$, and any polylogarithmically bounded function $p$, we also have

$$
\bigcup_{n \in \mathbb{N}} \left( \{0,1\}^n \times \{0,1\}^n \cap \bigcup_{i \leq p(n)} L_i \right) \subseteq C,
$$

and

$$
\bigcup_{n \in \mathbb{N}} \left( \{0,1\}^n \times \{0,1\}^n \cap \bigcap_{i \leq p(n)} L_i \right) \subseteq C.
$$

4. Communication Hierarchies

In this chapter we will study the properties of two communication hierarchies which correspond to the polynomial time hierarchy, resp. the Boolean hierarchy in TM complexity. While for the latter we can show proper inclusion at each level, we don't know whether the polynomial communication hierarchy collapses.

The following result relates $C-P$ with the first level of the polynomial communication hierarchy. Clearly,

$$
DComm(T) \subseteq NComm(T) \cap coNComm(T).
$$

Furthermore, from [5] and the improvement in [2] follows
Theorem 4: \( NComm(T_1) \cap coNComm(T_2) \subseteq DComm(T_1^{-1} \cdot T_2 \cdot (1 + o(1))) \).

Corollary: \( C \mathcal{P} = C \mathcal{NP} \cap C \text{coNP} \).

From the definition it is clear that
\[
A_kComm(T) \cup \text{co}A_kComm(T) \subseteq A_{k+1}Comm(T) \cap \text{co}A_{k+1}Comm(T).
\]
and hence
\[
C\Sigma_k \cup C\Pi_k \subseteq C\Sigma_{k+1} \cap C\Pi_{k+1} \subseteq C\text{PSpace}. \]

Definition: Babai, Frankl, and Simon defined a (polynomial-time) rectangular reduction from a language \( L \) to another, \( L' \), to be a pair of functions \((f, g)\), \( f, g : \{0, 1\}^n \rightarrow \{0, 1\}^{|x|}\), such that \( \log |x| < O(\log |y|) \) for some constant \( a \) and \( (x, y) \in L \iff (f(x), g(y)) \in L' \). If there exists a rectangular reduction from \( L \) to \( L' \) we denote this property by \( L \leq \text{L} \).

A language \( L \) is \( \Phi \)-hard for a class \( @ \) of languages if for every \( L' \in @ \), \( L \leq L' \). \( L \) is \( \Phi \)-complete if \( L \) is \( \Phi \)-hard and \( L \in @ \).

Babai et al. have constructed complete languages for \( C\Sigma_k \), \( C\Pi_k \), and \( C\text{PSpace} \). They observed that, just as for Turing machine classes, if a \( C\Pi_k \)-complete language belongs to \( C\Sigma_k \) then
\[
C\Sigma_k \cap C\Pi_k \subseteq C\Sigma_{k+1} \cap C\Pi_{k+1} \subseteq \ldots
\]

Another simple observation in this context is ("C" always denotes proper inclusion of sets)

Theorem 5: If, for some \( k \in \mathbb{N} \), \( C\Sigma_k \neq C\Pi_k \) then \( C\Sigma_k \cap C\Pi_k \subseteq C\Sigma_{k+1} \cap C\Pi_{k+1} \).

Proof: Choose a \( C\Sigma_k \)-complete language \( L \). If \( C\Sigma_k \neq C\Pi_k \), then \( L \notin C\Pi_k \) and \( L \subseteq C\Sigma_k \cup C\Pi_k \). But \( L, \overline{L} \in C\Sigma_{k+1} \cap C\Pi_{k+1} \) and, since \( C\Sigma_{k+1} \) and \( C\Pi_{k+1} \) are closed under \( \cup \), also \( L \cup \overline{L} \in C\Sigma_{k+1} \cap C\Pi_{k+1} \).

It should be clear that the corresponding result holds for TM classes. But for communication complexity we know that the condition of Theorem 5 is satisfied for \( k = 1 \). So we have the following separations.

Corollary: \( C\mathcal{NP} \cup C\text{coNP} \subseteq C\Sigma_2 \cap C\Pi_2 \).

Another analogy to sequential complexity classes is the characterization of the polynomial hierarchy by nondeterministic oracle protocols which may be defined analogously to the deterministic case. Hence the oracle queries of a nondeterministic protocol are always a function of the local input and the messages exchanged before. In this sense, oracle queries are deterministic. (If we would allow to guess the oracle queries without communication we would have \( C\mathcal{NP}^{id} = 2^{\{0,1\}^\ast} \).)

Theorem 6: For all \( k \geq 1 \)
\[
C\mathcal{NP}^{C\Sigma_k} = C\mathcal{NP}^{C\Pi_k} = C\Sigma_{k+1},
\]
where \( C\mathcal{NP}^{L[m]} \) resp. \( C\mathcal{NP}^{L[m]}^+ \) denotes the class of languages recognizable by nondeterministic protocols with oracle \( L \) where only \( m \) oracle queries are allowed and along any accepting path only negative resp. positive answers are given by the oracle.

Proof: From the definition follows immediately
\[
C\Sigma_{k+1} \subseteq C\mathcal{NP}^{C\Pi_k} = C\Sigma_{k+1},
\]
and it is not hard to see that
\[
C\mathcal{NP}^{C\Pi_k} = C\Sigma_{k+1}.
\]
The only thing left to prove is that \( C\mathcal{NP}^{C\Sigma_k} \subseteq C\mathcal{NP}^{C\Pi_k} \).

Let \( L \in C\mathcal{NP}^{C\Sigma_k} \) and choose some \( C\Sigma_k \)-complete language \( L' \) as an oracle. As in the case of simple nondeterministic computations we still may assume nondeterministic oracle protocols to be one-way: Processor \( P_0 \) may guess the whole computation (including the oracle's answers) in advance and send this string to \( P_1 \). Together they check the oracle's answers and finally \( P_1 \) accepts if the guess was right.

Now we modify those queries where the guessed answer is 1: Instead of asking "\( (f(x_0), g(x_1)) \in L' \)?", that is
\[
\exists u_1 \forall u_2 \ldots Q u_k : \phi(f(x_0), u) \circ \psi(g(x_1), u) ?
\]

\( P_0 \) guesses some \( u_1 \) and the new query takes the form
\[
\exists u_2 \forall u_3 \ldots Q' u_k : \neg \left( \phi(f(x_0), u) \circ \psi(g(x_1), u) \right) ?
\]
where the answer should be 0 for accepting computations. These modified queries correspond to an oracle \( L'' \in C\Sigma_{k+1} \) and hence may be reduced to \( L' \). Since the number of queries is at most polylogarithmic and \( C\Sigma_k \) is closed under polylogarithmically bounded union, it can be decided by a single query whether all answers are 0. Hence, \( C\mathcal{NP}^{C\Sigma_k} \subseteq C\mathcal{NP}^{C\Pi_k} \).
As we already mentioned before $DComm(n) = 2^{O(1)^\ast}$. But since we do not require languages to be decidable in our notion, a simple counting argument leads to the following result.

**Theorem 7:** For any function $k : \mathbb{N} \rightarrow \mathbb{N}$, there are languages $L$ such that

$$L \not\subseteq A_kComm(n - 2).$$

**Corollary:** $C^P\text{Space} \neq 2^{O(1)^\ast}$.

Another analogy to Turing machine complexity is
the **Boolean communication hierarchy**. Its levels are defined as follows

\[
\begin{align*}
\text{C-NP}(1) & := CNP \\
\text{C-NP}(2) & := \{L_1 \cap L_2 | L_i \in \text{C-NP}\} \\
\text{C-NP}(3) & := \{(L_1 \cap L_2) \cup L_3 | L_i \in \text{C-NP}\} \\
\text{C-NP}(4) & := \{((L_1 \cap L_2) \cap L_3) \cap L_4 | L_i \in \text{C-NP}\} \\
\end{align*}
\]

Its complementary classes are denoted by

$$C^{\text{coNP}}(k) := \{\overline{L} | L \in \text{C-NP}(k)\}.$$  

For each $k$ a C-NP(k)-complete language $L_k$ can be defined in terms of the C-NP-complete set intersection problem $SI := \{(x,y) | x_i = y_i = 1 \text{ for some } i \in \{1, \ldots, n\}\}$ and its complement $\overline{SI}$. (In the following we assume $|x_1| = |y_1| = |x_2| = |y_2| = \ldots$)

$$L_1 := SI$$

$$L_2 := SI \setminus \overline{SI}$$

$$L_3 := SI \setminus \overline{SI}$$

$$L_4 := (SI \setminus \overline{SI}) \setminus SI$$

Similarly the C-coNP(k)-complete languages $L_{-k}$ are defined as

- $L_{-1} := \overline{SI}$
- $L_{-2} := SI \setminus \overline{SI}$
- $L_{-3} := (SI \setminus \overline{SI}) \setminus SI$
- $L_{-4} := (SI \setminus \overline{SI}) \setminus SI$

Since $C-NP$ and $C^{\text{coNP}}$ are closed under union, respectively intersection it follows that $C-NP(k) = C^{\text{coNP}}(k) = C^{\text{coNP}}(k + 1) = C^{\text{coNP}}(k + 1) = \ldots$ if $L_k \in C^{\text{coNP}}(k)$. Also it is easy to see that $C^{\text{coNP}}(k) \cup C^{\text{coNP}}(k) \subseteq C-NP(k + 1) \cap C^{\text{coNP}}(k + 1)$ and $C^{\text{coNP}}(k) \subseteq C^{\text{coNP(k)}},$ which is the $k$th level of the constant query bounded hierarchy.

In contrast to the sequential case we are able to show that the Boolean communication hierarchy does not collapse. This extends the result of the corollary to Theorem 5.

**Theorem 8:** For all $k \in \mathbb{N}$ $C-NP(k) \neq C^{\text{coNP}}(k)$.

**Proof:** The proof is based on a result of Kadin, who showed in [6] that the existence of a polynomial time reduction of $NP(k)$ to $coNP(k)$ with a sparse oracle $S_k$ implies the existence of a polynomial time reduction of $NP(k-1)$ to $coNP(k-1)$ with a sparse oracle $S_{k-1}$. Because of the nonuniformity of communication protocols sparse oracles are of no use in communication complexity. Hence, from $C-NP \neq C^{\text{coNP}}$ we can conclude that $C-NP(k) \neq C^{\text{coNP}}(k)$ for all $k \in \mathbb{N}$. The adaption of Kadin's result to communication complexity may be formulated as follows.

If, for some $k > 1$, there exists a rectangular reduction $(f_k, g_k)$ of $L_k$ to $L_{-k}$ then there exists a rectangular reduction $(f_{k-1}, g_{k-1})$ of $L_{k-1}$ to $L_{-(k-1)}$.

Suppose $(f_k, g_k)$ reduces $L_k$ to $L_{-k}$. A pair $(A, B)$ of strings of length $n$ is $(f_k, g_k)$-easy if there exist pairs $(A_1, B_1), \ldots, (A_{k-1}, B_{k-1})$ of strings of length $n$ such that

$$f_k(A_1 \ldots A_{k-1}A) := (C_1 \ldots C_{k-1}C)$$

$$g_k(B_1 \ldots B_{k-1}B) := (D_1 \ldots D_{k-1}D)$$

and $(C, D) \in SI$.

$(A, B)$ is $(f_k, g_k)$-hard if $(A, B) \not\in \overline{SI}$ and $(A, B)$ is not $(f_k, g_k)$-easy.
Let $A, B$ be strings of length $n$. Suppose that $(A, B)$ is $(f_k, g_k)$-easy. Then $(A, B)$ must be in $SI$. For example, if $k$ is even, $(C, D) \in SI$ implies
\[(C_1 \ldots C_{k-1}, D_1 \ldots D_{k-1}) \in L_{-k} \]
Thus since $(f_k, g_k)$ is a rectangular reduction, $(A_1 \ldots A_{k-1}, B_1 \ldots B_{k-1}) \in L_k$, but this implies $(A, B) \in SI$. If $k$ is odd, the same reasoning holds with $L_k$ and $L_{-k}$ resp. $SI$ and $SI$ interchanged.

For each length $n$ we distinguish two cases.

**Case 1:** Suppose that all pairs $(A, B) \in SI$ of strings of length $n$ are $(f_k, g_k)$-easy. Then there exists a $CNP$-algorithm $A_k$ which accepts $SI \cap \{0, 1\}^n \times \{0, 1\}^n$.

On input $(A, B)$ processor $P_0$ guesses $A_1 \ldots A_{k-1}$ and computes $f_k(A_1 \ldots A_{k-1} A)$, processor $P_1$ guesses $B_1 \ldots B_{k-1}$ and computes $g_k(B_1 \ldots B_{k-1} B)$. $(A, B)$ is accepted if the last components of the results are in $SI$.

Thus there exists a $CNP$-algorithm $A_k$ which accepts $SI \cap \{0, 1\}^n \times \{0, 1\}^n$ to $SI$ and thus $L_{k-1}$ to $L_{-(k-1)}$.

**Case 2:** Suppose there exists a $(f_k, g_k)$-hard pair $(A, B)$ of strings of length $n$. If $k$ is even, then for all $(k-1)$-tuples $(A_1, B_1), \ldots, (A_{k-1}, B_{k-1})$ of pairs of strings of length $n$
\[(A_1, \ldots, A_{k-1}, B_1, \ldots, B_{k-1}) \in L_{k-1} \]
\[\iff (A_1, \ldots, A_{k-1} A, B_1, \ldots, B_{k-1} B) \in L_k \]
Since $(A, B) \in SI$. Since $(f_k, g_k)$ is a reduction,
\[(A_1, \ldots, A_{k-1} A, B_1, \ldots, B_{k-1} B) \in L_k \]
\[\iff (C_1, \ldots, C_{k-1} C, D_1, \ldots, D_{k-1} D) \in L_{-k} \]
Since $(A, B)$ is $(f_k, g_k)$-hard, $(C, D) \not\in SI$, thus
\[(C_1, \ldots, C_{k-1} C, D_1, \ldots, D_{k-1} D) \in L_{-k} \]
\[\iff (C_1, \ldots, C_{k-1} C, D_1, \ldots, D_{k-1} D) \in L_{-(k-1)} \]
If $k$ is odd, the same reasoning holds with $L_{i}$ and $L_{-i}$ interchanged.

Thus there exists a rectangular reduction of $L_{k-1}$ to $L_{-(k-1)}$.

5. **Probabilistic Classes, C-#P and related Classes**

From the definition follows immediately

**Lemma 5:** $C\#P \subseteq C\#P P \subseteq C\#P P \subseteq C\#P P$.

Babai et al. mentioned the following analogy to sequential complexity classes.

**Theorem 9:** $C\text{-BPP} \subseteq C\text{-NP} \cap C\text{-HP}$.  

**Proof:** The proof is an easy adaption of the Laute-man proof [7] to communication complexity. The only nontrivial part of the adaption is a result from [2] (Theorems 8 and 9): For any $C\text{-BPP}$-algorithm $A$ one can construct a $C\text{-BPP}$-algorithm $A'$ such that in each round all messages of $A'$ are sent with probability either 0 or $p(n)$, where $\log \frac{1}{p(n)}$ is polylogarithmic in $n$. Hence a $C\text{-BPP}$-algorithm may be regarded as a deterministic algorithm with an additional input of $k(n) \cdot \log \frac{1}{p(n)}$ uniform distributed random bits, where $k(n)$ is the maximum number of rounds on inputs of length $n$ (and hence is also at most polylogarithmic in $n$).

**Theorem 10:** $C\text{-BPP} = (C\text{-NP} \cup C\text{-coNP}) \neq \emptyset$.

**Proof:** The $n$-bit ordering function $g_n$ has probabilistic complexity $C_\text{i}(g_n) \leq O(\log^2 n)$ (cf. [4]) and nondeterministic complexities $C_\text{nondet}(g_n) = \text{Cout}_{\text{nondet}}(g_n) = n$ (since at least $2^k - 1$ monochromatic rectangles are needed to cover either 0's or 1's).

**Theorem 11:** $C\text{-NP} \neq C\text{-BPP}$.

**Proof:** The set intersection function $s_n$ has nondeterministic complexity $C_\text{nondet}(s_n) = \log n + 1$ and probabilistic complexity $C_\text{i}(s_n) \geq \Omega(\sqrt{n})$ (cf. [1]). Recently it was shown in [8] that even $C_\text{i}(s_n) \geq \Omega(n)$ holds.

As observed by Babai et al. (without proof), the following inclusions hold.

**Lemma 6:** $C\text{-PP} \subseteq C\text{-P} \subseteq C\text{-P} \subseteq C\text{-P} \subseteq C\text{-Space}$.

Another result mentioned in [1] is the following obvious

**Lemma 7:** $C\text{-#P} \subseteq C\text{-#P} \subseteq C\text{-#P}$.

Babai et al. have observed that for each language $L \subseteq C\text{-PP}$ there is a probabilistic polylogarithmic communication algorithm with moderately bounded error probability (at most $2^{-\log^2 n}$), hence $C\text{-PP} \subseteq C\text{-MP}$.

We will now show that this boundedness condition suffices to put a language in $C\text{-PP}$.

**Theorem 12:** $C\text{-PP} = C\text{-MP}$.
Proof: In order to show that \( \text{C-PP} \subseteq \text{C-MPP} \), let \( L \in \text{C-PP} \) and let \( A \) be a \( \text{C-MPP} \)-algorithm such that \((x_0, x_1) \in L\) if and only if more than half of the computations of \( A \) on input \((x_0, x_1)\) are accepting. Let \( N \leq 2^{(\log n)^x} \) be the total number of computations of \( A \). A probabilistic algorithm \( A' \) may now simulate \( A \) by sending at random some computation of \( A \) on input \((x_0, x_1)\). \( A' \) accepts with probability \( 1 - \frac{1}{2^{n/2}} \) if \( A \) accepts and rejects otherwise. Then the error probability of \( A' \) is at most \( \frac{1}{2} \). Hence \( L \in \text{C-MPP} \).

Now, let \( L \in \text{C-MPP} \) and let \( f \) be its characteristic function. Then \( C_1(f_n) \leq (\log n)^x \) with \( \epsilon \leq \frac{1}{2} - 2^{-2(\log n)^x} \) for some \( c > 0 \). By Theorem 7 of [2],

\[
C_1(f_n) + 1 = (\log n)^x + 1,
\]

where

\[
\frac{1}{2} - \epsilon = \left(\frac{1}{2} - \epsilon\right) \cdot 2^{-(C_1(f_n) + 1)} \geq 2^{-2(\log n)^x - 1}.
\]

Without loss of generality we may assume one-way algorithms to have the same fixed length for all computations. Hence, by Theorem 9 of [2], there exists a probabilistic one-way algorithm \( A \) which computes \( f_n \) with error probability less than

\[
\epsilon'' = \frac{1}{2} - \frac{1}{2} \cdot \left(\frac{1}{2} - \epsilon\right) \leq \frac{1}{2} - 2^{-2(\log n)^x - 2}
\]

and at most

\[
2C_1(f_n) + \log \frac{4}{\frac{1}{2} - \epsilon} \leq 4 \cdot (\log n)^x + 5
\]

bits of communication such that any message is sent with probability either 0 or \( p \), where

\[
\frac{1}{2} \leq \frac{4 \cdot 2C_1(f_n)}{\frac{1}{2} - \epsilon} \leq 2^{2(\log n)^x + 2}.
\]

Let \( M \) be the set of messages of \( A \). For \( \alpha \in M \), let \( p_0(x_0, \alpha) \in \{0, 1\} \) be the probability that message \( \alpha \) will be sent on input \( x_0 \in \{0, 1\}^n \), and let \( p_1(x_1, \alpha) \in [0, 1] \) be the probability that, on input \( x_1 \in \{0, 1\}^n \), message \( \alpha \) leads to output 1. Let \( a = \lfloor 2(\log n)^x \rfloor \). Then we define a probabilistic one-way algorithm \( A' \) with messages from \( M \times \{0, 1\}^a \) as follows: On input \( x_0 \), processor \( P_0 \) sends some message \( (\alpha, i) \) with probability \( \frac{1}{2} \). On input \( x_1 \), processor \( P_1 \) accepts iff it receives some message \( (\alpha, i) \) with \( i \leq a \cdot p_1(x_1, \alpha) \).

Obviously, the probability of accepting has not increased and has decreased at most by a factor of \( \frac{a+1}{a} \).

So, the error probability of \( A' \) is at most

\[
1 - (1 - \epsilon'') \cdot \frac{a+1}{a} \leq \frac{1}{2} - 2^{-2(\log n)^x - 2} + \left(\frac{1}{2} + 2^{-2(\log n)^x - 2}\right) \cdot 2^{-3(\log n)^x} < \frac{1}{2}
\]

But now, each message has probability either 0 or \( \frac{1}{a} \) and \( P_2 \) accepts or rejects deterministically. So \( A' \) may be viewed as a nondeterministic algorithm and, for each input, more than half of the possible computations of \( A' \) give the right result. Furthermore each computation has only polylogarithmic length. Therefore, \( L \in \text{C-PP} \).

The next result extends the lower bound in [1] on the probabilistic communication complexity of \( IP_2 \), the inner product mod 2, for (strictly) bounded error to error probabilities which are not even moderately bounded. This allows us to separate \( \text{C-PP} \) from \( \text{C-P}_c' \) and strengthens the conjecture of [9] that \( IP_2 \not\subseteq \text{C-PP} \).

Theorem 13: Let \( ip_2 \) be the inner product mod 2 function on bitstrings of length \( n \). Then, for any \( \epsilon \leq \frac{1}{2} - \frac{1}{2} \cdot 2^{-\frac{n}{2}} \),

\[
C_1(ip_2) \geq \frac{n}{2}
\]

Proof: The proof is based on the following lemma due to J. H. Lindsey as cited in [1].

**Lemma 8:** Let \( H \) be a \( m \times m \) Hadamard matrix and \( T \) an arbitrary \( a \times b \) submatrix of \( H \). Then the difference between the number of \( +1 \)'s and \( -1 \)'s in \( T \) is at most \( \sqrt{ab} \).

**Proof:** Omitted (cf. [1]).

Let \( IP_2 \) be the 0/1-matrix corresponding to \( ip_2 \). An erroneous deterministic algorithm which computes \( ip_2 \) induces a covering of \( IP_2 \) by disjoint rectangles \( R_1, \ldots, R_d \), where \( d \) is the number of different possible computations. For any rectangle \( R_t \), the algorithm outputs the same result for all entries in \( R_t \).

Since \( IP_2 \) corresponds to a Hadamard matrix of size \( 2^n \times 2^n \), each rectangle \( R_t \) of size \( r_t \) in \( IP_2 \) includes at least

\[
\frac{r_t}{2} - \sqrt{r_t} \cdot 2^n / 2
\]
errors. Furthermore, 
\[ \sum_{i=1}^{d} r_i = 2^{2n} \]
because the rectangles form a partition of \( IP_2 \). So the fraction of errors in this covering is at least 
\[ \frac{1}{2} - \frac{1}{2} \cdot 2^{-\frac{s}{n}} \sum_{i=1}^{d} \sqrt{r_i} \]
\[ \geq \frac{1}{2} - \frac{1}{2} \cdot 2^{-\frac{s}{n}} \cdot d \cdot \sqrt{2^{2n}} = \frac{1}{2} - \frac{1}{2} \cdot 2^{-\frac{s}{n}} \cdot \sqrt{d}. \]
So, if \( d \leq 2^s \) then the fraction of errors is at least \( \frac{1}{2} - \frac{1}{2} \cdot 2^{-\frac{s}{n}} \). Hence Lemma 2 of [2] and Yao's bound on deterministic communication complexity [3] imply 
\[ C_i(i\overline{p2n}) \geq n \]
for any \( \epsilon \leq \frac{1}{2} - \frac{1}{2} \cdot 2^{-\frac{s}{n}} \).

Corollary: \( C-\#P \not\subseteq C-PP \), hence \( C-PP \subseteq C-P^{\#P} \).

Corollary: Undirected graph reachability, planarity, bipartiteness, and 2-CNF-satisfiability for sparse graphs are not in \( C-PP \).

Proof: The mentioned problems are \( C-\text{PSPACE}-\text{hard} \) (cf. [1]).

As an extension of Theorem 5 we have the following

Theorem 14: \( C-NP \cup C-coNP \cup C-BPP \subseteq C_{2^2} \cap C-H \cap C-PP \).

Proof: Let \( S \) be the language corresponding to the set intersection function. We have already seen that \( S \notin C-BPP \) and that \( S \) is \( C-NP \)-complete. Hence \( S \notin C-NP \cup C-coNP \cup C-BPP \). Since \( S \in C-NP \subseteq C_{2^2} \cap C-H \cap C-PP \) and \( C_{2^2} \cap C-H \cap C-PP \) are closed under joining we also have \( S \notin C-\text{PSPACE} \).

Lemma 9: \( RUC\text{omm}(n - 6) \neq 2^{(0,1)^*} \).

Proof: Almost all boolean valued functions on pairs of \( n \)-bit strings have unbounded error probabilistic complexity at least \( n - 5 \) (cf. [10]).

References


