Electronic Colloquium on Computational Complexity, Report No. 33 (2002)

A very simple function that requires exponential size nondeterministic graph-driven read-once branching programs

Beate Bollig*

FB Informatik, LS2, Univ. Dortmund,
44221 Dortmund, Germany
bollig@ls2.cs.uni-dortmund.de

Abstract. Branching programs are a well-established computation model for boolean functions, especially read-once branching programs (BP1s) have been studied intensively. A very simple function f in n^2 variables is exhibited such that both the function f and its negation $\neg f$ can be computed by Σ_p^3-circuits, the function f has nondeterministic BP1s (with one nondeterministic node) of linear size and $\neg f$ has size $O(n^4)$ for oblivious nondeterministic BP1s but f requires nondeterministic graph-driven BP1s of size $2^{\Omega(n)}$. This answers an open question stated by Jukna, Razborov, Savický, and Wegener [13].

1 Introduction

Besides boolean circuits and formulae branching programs (BPs), sometimes also called binary decision diagrams (BDDs), are one of the standard representations for boolean functions. (For a history of results on branching programs see, e.g., the monograph of Wegener [18]).

Definition 1. *A* branching program (BP) *or binary decision diagram* (BDD) *on the variable set $X_n = \{x_1, \ldots, x_n\}$ is a directed acyclic graph with one source and two sinks labeled by the constants 0 and 1. Each non-sink node* (*or decision node*) *is labeled by a boolean variable and has two outgoing edges, one labeled by 0 and the other by 1. A* nondeterministic branching program ($\vee$-BP *for short*) *is a branching program with some additional unlabeled nodes, called nondeterministic nodes, which have out-degree 2.*

An input $a \in \{0,1\}^n$ activates all edges consistent with a, i.e., the edges labeled by a_i which leave nodes labeled by x_i and all unlabeled edges. A computation path for an input a in a BP *G is a path of edges activated by the input a that leads from the source to a sink. A computation path for an input a that leads to the 1-sink is called accepting path for a.*

Let B_n denote the set of all boolean functions $f : \{0,1\}^n \to \{0,1\}$. The BP *$G$ represents a function $f \in B_n$ for which $f(a) = 1$ iff there exists an accepting path for the input a.*

A parity branching program (*or* $\oplus$-BP *for short*) *is syntactically a nondeterministic branching program but instead of the usual existential nondeterminism the parity acceptance mode is used. An input a is accepted iff the number of its accepting paths is odd.*

The size of a branching program G is the number of its nodes and is denoted by $|G|$. The branching program size of a boolean function f is the size of the smallest BP *representing f. The length of a branching program is the maximum length of a path.*

* Supported in part by DFG We 1066/9.

ISSN 1433-8092

In order to learn more about the power of branching programs, various restricted models have been investigated intensively and several interesting restricted types of BPs could be analyzed quite successfully (for the latest breakthrough for semantic superlinear length BPs see [1],[2], and [3], where using a subtle combinatorial reasoning superpolynomial lower bounds were obtained).

Besides this complexity theoretical viewpoint people have used branching programs in applications. Bryant [9] introduced ordered binary decision diagrams (OBDDs) which are up to now the most popular representation for formal circuit verification.

Definition 2. *An* OBDD *is a branching program with a* variable ordering *given by a permutation π on the variable set. On each path from the source to the sinks, the variables at the nodes have to appear in the order prescribed by π (where some variables may be left out). A* π-OBDD *is an* OBDD *ordered according to π.*

Unfortunately, several important and also quite simple functions have exponential OBDD size. Therefore, more general representations with good algorithmic behavior are necessary. Generalizing the concept of variable orderings to graph orderings Gergov and Meinel [10, 11] and Sieling and Wegener [16] have shown independently how deterministic read-once branching programs can be used for verification.

Definition 3. *A* graph ordering *is a branching program with a single sink, where on each path from the source to the sink all variables appear exactly once. Let $\omega \in \{\vee, \oplus\}$. An* ω-nondeterministic graph-driven BP1 *is an ω-nondeterministic* BP1 *G for which there exists a graph ordering G_0 with the following property: If for an input a, a variable x_i appears on a computation path of a in G before the variable x_j, then x_i also appears on the unique computation path of a in G_0 before x_j.*

In the following if nothing else is mentioned nondeterministic graph-driven BP1s means $\vee$-nondeterministic graph-driven BP1s .

The main idea is that in graph-driven BP1s according to a fixed graph ordering, for each input the variables are tested in the same ordering, whereas (different from OBDDs) for different inputs different orderings may be used. It is easy to see that any deterministic BP1 is in fact a BP1 for a suitably chosen graph ordering. For nondeterministic BP1s graph orderings do not exist in general. If we generalize OBDDs to nondeterministic OBDDs we gain the possibility of nondeterministic guesses, but the variable ordering remains the same on all computation paths. For read-once branching programs the situation is different. It is possible to guess nondeterministically and moreover, for each input arbitrary orderings of the variables are allowed.

The concept of graph-ordered branching programs has turned out to be also useful in other settings, see e.g. [14] and [17]. Gergov and Meinel [10] were the first ones who suggested parity graph-driven BP1s as a data structure for boolean functions. Another reason for investigating parity graph-driven BP1s is that until now exponential lower bounds on the size of parity read-once branching programs for explicitly defined boolean functions are unknown. One step towards the proof of such bounds might be to investigate BP models *inbetween* deterministic and parity BP1s. Nondeterministic and parity graph-driven BP1s have been investigated more intensively in [4], [8], and [7].

Definition 4. *Σ_p^d, Π_p^d be the classes of functions that can be computed by polynomial size depth-d circuits over the de Morgan basis $\{\wedge, \vee, \neg\}$ (negations are allowed only at*

the input variables and do not contribute to the depth) that have $\vee$ *(respectively,* $\wedge$*) as output gate.*

The following general question has been widely studied for various computational models:

Suppose that both a computational problem f *and its complement* $\neg f$ *posses an efficient nondeterministic computation in some model. Does this imply that* f *can also be computed efficiently and deterministically in the same model?*

This question is often called the P *versus* NP∩co-NP *question* and for boolean (non-uniform) complexity, polynomial size instead of polynomial time is investigated.

In [13] an explicitly defined boolean function is presented that is in Σ_p^3 and in NP∩co-NP for read-once branching programs but has exponential deterministic read-once branching program size. Another function belongs to the smaller class $\Sigma_p^3 \cap \Pi_p^3$, but the separation is only quasipolynomial. Jukna, Razborov, Savický, and Wegener (1999) asked whether the class $\Sigma_p^3 \cap \Pi_p^3$ contains a function separating NP∩co-NP from quasipolynomial size in the context of read-once branching programs. Here we answer the question in the affirmative. Moreover, we prove that not only the deterministic BP1 size but even the nondeterministic graph-driven BP1s size of the selected function is exponential. In [5] a lower bound method for nondeterministic graph-driven read-once branching programs is presented and the first exponential lower bound for an explicitly defined boolean function has been proved. Since the lower bound technique is very general we cannot apply the method directly to our function. Therefore, we have to improve the lower bound method.

2 The separation result

The theory of communication complexity is a powerful tool for proving lower bounds on the size of restricted nondeterministic oblivious BPs (a BP is called *oblivious* if the nodes can be partitioned into levels such that edges point only from lower to higher levels and all internal decision nodes of one level are labeled by the same variable). (See, e.g., [12] and [15] for the theory of communication complexity.) In [7] and [5] it has been shown how this tool can be used for proving large lower bounds on the size of so-called well-structured ω-nondeterministic BP1s, $\omega \in \{\vee, \oplus\}$, repectively, of nondeterministic graph-driven BP1s.

For the ease of notations we assume w.l.o.g. that n is an even number. The function f_n is defined on $n \times n$ boolean matrices X on the variable set $X_n = \{x_{1,1}, \ldots, x_{n,n}\}$ and outputs 1 iff the matrix X contains exactly one 1-entry in each row or exactly $n-1$ columns with exactly one 1-entry and $n-1$ 1-entries altogether. The technique described in [5] is very general and cannot be applied directly in order to prove an exponential lower bound on the size of nondeterministic BP1s representing f_n. Therefore, we have to improve the method. First, we need some further notation.

Definition 5. *Let* f *be a boolean function defined on the variables in* $X_n = \{x_1, \ldots, x_n\}$*. A set* $A(f) = \{(\alpha_1, \beta_1), (\alpha_2, \beta_2), \ldots, (\alpha_k, \beta_k)\}$*,* $\alpha_i \in \{0,1\}^{n'}$ *and* $\beta_i \in \{0,1\}^{n-n'}$*, is called a* strong 1-fooling set *if*

i) $f(\alpha_i, \beta_i) = 1$ *for all* $i \in \{1, \ldots, k\}$*, and*
ii) $i, j \in \{1, \ldots, k\}$ *and* $i \neq j$ *implies that* $f(\alpha_i, \beta_j\} = 0$ *and* $f(\alpha_j, \beta_i\} = 0$*.*

For a subset $Z \subseteq X_n$ we denote by $\mathcal{A}(Z)$ the set of all possible assigments to the variables in Z. Let G_0 be a graph ordering and v a node in G_0. Let X_v be the set of variables tested on a path from the source to v (excluding the variable which is the label of v) and $\mathcal{A}_v \subseteq \mathcal{A}(X_v)$ a set of partial assignments which lead in G_0 from the source to v. Using well-known facts from communication complexity the following is easy to prove. Let $v_1, \ldots, v_k$ be nodes in G_0, where $X_{v_1} = \ldots = X_{v_k}$, $\mathcal{A}_v = \cup_{1 \leq j \leq k} \mathcal{A}_{v_j}$, and $X_v := X_{v_1}$. If there exists a strong 1-fooling set $A(f) = \{(\alpha_1, \beta_1), (\alpha_2, \beta_2), \ldots, (\alpha_{|\mathcal{A}_v|}, \beta_{|\mathcal{A}_v|})\}$, $\alpha_i \in \mathcal{A}_v$, and $\beta_i \in \mathcal{A}(X_n \setminus X_v)$, then any ω-nondeterministic BP1 representing f according to G_0 has a size of at least $|\mathcal{A}_v|$.

Now our proof idea is the following one. We use the fact that in a nondeterministic BP for a 1-input there has to be at least one accepting path and for 0-inputs accepting paths are not allowed. This property does not hold for parity BPs and until now promising methods for proving large lower bounds on the size of (usual) parity graph-driven BP1s are unknown.

Let G be a nondeterministic graph-driven BP1 representing f_n and let G_0 be a graph ordering such that G is ordered according to G_0. We choose a large set of subpaths in G_0 and define V as the set of nodes $v_1, \ldots, v_l$ which are reached by at least one of the chosen subpaths. If there is large number of subpaths which lead to the nodes $v_{i_1}, \ldots, v_{i_k}$ in V, where $X_{v_{i_1}} = \ldots = X_{v_{i_k}}$, then we can construct a large strong 1-fooling set. Otherwise, there are many nodes $v_{j_1}, \ldots, v_{j_m}$, where $X_{v_{j_l}} \neq X_{v_{j_{l'}}}$, $1 \leq l < l' \leq m$. Let S_w be the union of all variables tested on a path from w to a sink (including the label of w). Let $x_{i',j'}$ be a variable that is not contained in $X_{v_{j_1}} \cup \ldots \cup X_{v_{j_m}}$. We can prove that for each set $X_{v_{j_i}}$ there exists a node w in G such that $S_w \subseteq X_n \setminus \{X_{v_{j_i}} \cup \{x_{i',j'}\}\}$. Therefore, the size of G is at least m.

Theorem 1. *The function f_n is in $\Sigma_p^3 \cap \Pi_p^3$ and can be represented by nondeterministic* BP1*s with one nondeterministic node in linear size and the function $\neg f_n$ has size $O(n^4)$ for nondeterministic* OBDD*s but its nondeterministic graph-driven* BP1 *size is $2^{\Omega(n)}$.*

Proof.

The upper bounds:

The proof of the upper bounds are not difficult to prove.

- We can test whether the matrix X has exactly one 1-entry in row i by

$$(x_{i,1} \vee \ldots \vee x_{i,n}) \wedge \bigwedge_{1 \leq j < k \leq n} (\overline{x}_{i,j} \vee \overline{x}_{i,k}).$$

  Therefore, the test whether the matrix X has exactly one 1-entry in each row can be computed by Π_p^2-circuits of size $O(n^3)$. Similarly, the test whether the matrix X contains no 1-entry in the jth column and exactly one 1-entry in each of the other columns can be computed by Π_p^2-circuits of size $O(n^3)$. Since there are n possibilities to choose the column without a 1-entry, we can conclude that the function f_n can be computed by Σ_p^3-circuits of size $O(n^4)$.
- The output of the function $\neg f_n$ is 1 iff there exist a row and two columns such that the row and the columns do not contain exactly one 1-entry each or there exist a row without exactly one 1-entry and a column with at least two 1-entries. The test

whether the matrix X contains in row i not exactly one 1-entry can be computed by the Π_p^2-circuit

$$\bigwedge_{1\leq j\leq n} (x_{i,1} \vee \ldots \vee x_{i,j-1} \vee \overline{x}_{i,j} \vee x_{i,j+1} \vee \ldots \vee x_{i,n})$$

in size $O(n)$. Simlilarly the test whether the matrix X contains neither in column j_1 nor in column j_2 exactly one 1-entry can be performed. The test whether the matrix X contains in a chosen column at least two 1-entries can be performed in the same way in size $O(n)$. Since there are $O(n^3)$ possibilities to choose a row and two columns and $O(n^2)$ possibilities to choose a row and a column, $\neg f_n$ can be computed by Σ_p^3-circuits of size $O(n^4)$. Therefore, f_n can be computed in size $O(n^4)$ by Π_p^3-circuits.

– Now we construct two (deterministic) OBDDs G_1 and G_2, where G_1 accepts exactly the satisfying inputs with exactly one 1-entry in each row and G_2 accepts exactly the satisfying inputs with exactly $n-1$ columns with exactly one 1-entry and $n-1$ 1-entries in X. The BP1 G_1 (G_2) uses a rowwise (columnwise) variable ordering which means that all variables of one row (column) are tested one after another. Moreover, in G_2 we store the information whether there has been one column without a 1-entry. Obviously, $G_1 \vee G_2$ is a nondeterministic BP1 with one nondeterministic node for f_n and – because any satisfying input contains either exactly n ones or $n-1$ ones – also a valid parity BP1. The size of G_i, $i \in \{1,2\}$, is bounded above by $O(n^2)$. (Note, that n^2 is the number of variables in X.)
– For $\neg f_n$ we choose an arbitrary variable ordering and guess a row and two columns such that the number of 1-entries is not exactly 1 in the row and for each of the chosen columns. Since there are $O(n^3)$ possibilities to choose a row and two columns and $O(n^2)$ possibilities to choose a row with not exactly one 1-entry and a column with at least two 1-entries, the size of the nondeterministic OBDD (for an arbitrary variable ordering) is $O(n^4)$.

The lower bound:

For the lower bound the following observation is helpful. A partial assignment a is *crucial* if it is possible to complete a in two different ways a_r and a_c such that for the first complete assignment (a, a_r) there exist in each row of the matrix X exactly one 1-entry and for the second one (a, a_c) there exist exactly $n-1$ columns with exactly one 1-entry and $n-1$ 1-entries altogether.

Let G be a nondeterministic graph-driven BP1 representing f_n and G_0 be a graph ordering such that G is ordered according to G_0. We consider all paths in G_0 that correspond to permutation matrices which means that there exist n variables $x_{i_1,j_1}, \ldots, x_{i_n,j_n}$ which are set to 1, where $i_l \neq i_{l'}$ and $j_l \neq j_{l'}$ if $l \neq l'$, and all other variables are set to 0. The number of these paths is $n!$ Next, we define a cut through all these paths after exactly $n/2+1$ variables are set to 1. For each crucial assignment a corresponding to a chosen path to the cut there exist $(n/2-1)!$ different possibilities to complete a to a permutation matrix. Therefore, there exist $\frac{n!}{(n/2-1)!}$ different paths from the source to the cut. Let R_p (C_p) be the set of indices i for which a variable $x_{i,\cdot}$ ($x_{\cdot,i}$) is set to 1 on p. If $(n/2+1)$ rows and columns have been chosen, there are $(n/2+1)!$ possibilities to map the indices of the rows to the indices of the columns. Therefore, there is a set P of different paths, $|P| \geq \binom{n}{n/2+1}$, such that for two different paths p and p' it is $R_p \neq R_{p'}$

or $C_p \neq C_{p'}$. Using the pigeonhole principle we can conclude that there exists a variable $x_{i,j}$ such that for at least $|P|/n^2$ paths p in P the variable $x_{i,j}$ is the last variable tested on p. Let $P' \subseteq P$ be the set of these paths. The set V consists of all nodes v on a path from P' labeled by $x_{i,j}$. Now we consider all subpaths from the paths in P' to a node $v \in V$. Let P'' be the set of these paths. Obviously, $|P''| = |P'|$. By the definition of P and the fact that $x_{i,j}$ is set to 1 on all paths in P', we know that $R_p \neq R_{p'}$ or $C_p \neq C_{p'}$ for two different paths $p, p' \in P''$.

In the following let a_p be the corresponding (partial) assignment of the variables tested on a path p. Let v_p be a node that is reached by a path $p \in P''$. We distinguish two cases.

1) There are at least $|P''|^{2/3}$ paths in P'' such that for any two of them $X_{v_p} = X_{v_{p'}}$ or
2) there are at least $|P''|^{1/3}$ paths in P'' such that for any two of them p, p', where $p \neq p'$, $X_{v_p} \neq X_{v_{p'}}$.

– We consider the first case. One of the following properties is true. There are at least $|P''|^{1/3}$ paths in P'' such that for any two of them $R_p \neq R_{p'}$ or at least $|P''|^{1/3}$ paths such that $C_p \neq C_{p'}$ for two different paths $p, p' \in P''$.
  - If there are at least $|P''|^{1/3}$ paths in P'' such that for any two of them $R_p \neq R_{p'}$, let $\mathcal{A}_v$ be the set of the partial assignments which correspond to these paths. The variable $x_{i,j}$ is the label of the nodes reached by these paths. We choose for each $a_p \in \mathcal{A}_v$ a partial assignment a_p^R of the variables in $X_n \setminus X_{v_p}$ such that (a_p, a_p^R) corresponds to a permutation matrix and $x_{i,j} = 1$. By the definition of P'' such an assignment a_p^R exists. The function value $f(a_p, a_p^R)$ is 1 since in each row there exists exactly one 1-entry. For $a_{p'} \in \mathcal{A}_v$, let $a_{p'}^R$ be a partial assignment such that $(a_{p'}, a_{p'}^R)$ corresponds to a permutation matrix and $x_{i,j} = 1$. For $(a_{p'}, a_p^R)$, $p' \neq p$, there exists at least one row without a variable set to 1. Since the number of ones in X is n, $f(a_{p'}, a_p^R) = 0$. With the same arguments $f(a_p, a_{p'}^R) = 0$.
  - If there are at least $|P''|^{1/3}$ paths in P'' such that for any two of them $C_p \neq C_{p'}$, let $\mathcal{A}_v$ be the set of the partial assignments which correspond to these paths. Again $x_{i,j}$ is the label of the nodes reached by these paths. We choose for each a_p a partial assignment a_p^C which resembles a_p^R but with the exception that $x_{i,j}$ is set to 0. Since there are exactly $n-1$ columns with one 1-entry and the number of 1-entries is $n-1$ altogether, the function value $f(a_p, a_p^C)$ is 1. For $(a_{p'}, a_p^C)$, $p' \in P''$ and $p' \neq p$, there exist at least two columns without a variable set to 1. Since the number of 1-entries in X is $n-1$, we can conclude $f(a_{p'}, a_p^C) = 0$. With the same arguments $f(a_p, a_{p'}^C) = 0$.

  Altogether, we have proved that there exists a strong 1-fooling set of size at least $|P''|^{1/3}$.
– Now we consider the case that for at least $|P''|^{1/3}$ paths in P'' for any two of them $X_{v_p} \neq X_{v_{p'}}$, $p \neq p'$. For each path p we consider a partial assignments a_p^C of the variables in $X_n \setminus X_{v_p}$. As a first step we consider a partial assignment a_p^R such that (a_p, a_p^R) corresponds to a permutation matrix and $x_{i,j} = 1$. By the definition of P'' such an assignment a_p^R exists. Now the partial assignment a_p^C resembles a_p^R but with the exception that $x_{i,j}$ is set to 0. Obviously, $f(a_p, a_p^C) = 1$. For each of the chosen assignments, we consider one accepting path in G. Each of these accepting paths has

length of at least n^2-1 and the variable $x_{i,j}$ is the only variable that can be left out. We define a cut in G through all these accepting paths after exactly $n/2$ variables are set to 1. The variable $x_{i,j}$ cannot be tested on any of these paths, because $x_{i,j}$ is set to 0 and after the test of $x_{i,j}$ only $n/2-1$ variables are set to 1. Let W be the set of nodes reached for one of these accepting paths. If p is one of the chosen paths in P'' and v is the node in G_0 that is reached by p, then there exist a node $w \in W$ such that $X_w = X_{v_p}$ and $S_w = X_n \setminus X_{v_p}$ or $S_w = X_n \setminus (X_{v_p} \cup \{x_{i,j}\})$. By assumption the sets X_{v_p} for the chosen paths in P'' are all different, therefore, there are at least $|P''|^{1/3}$ nodes in W.

Summarizing we can conclude that the nondeterministic BP1 complexity is at least $|P''|^{1/3} = (|P|/n^2)^{1/3} = \left(\binom{n}{n/2+1}/n^2\right)^{1/3} = 2^{\Omega(n)}$.

□

Acknowledgement

The author would like to thank Martin Sauerhoff and Ingo Wegener for proofreading and fruitful discussions on the subject of the paper.

References

1. Ajtai, M. (1999). A non-linear time lower bound for Boolean branching programs. Proc. of 40th FOCS, 60–70.
2. Beame, P., Saks, M., Sun, X., and Vee, E. (2000). Super-linear time-space tradeoff lower bounds for randomized computation. Proc. of 41st FOCS, 169–179.
3. Beame, P. and Vee, E. (2002). Time-space trade-offs, multiparty communication complexity, and nearest neighbor problems. Proc. of 34th STOC, 688–697.
4. Bollig, B. (2001). Restricted nondeterministic read-once branching programs and an exponential lower bound for integer multiplication. RAIRO Theoretical Informatics and Applications, 35:149–162.
5. Bollig, B. (2002). Complexity theoretical results on nondeterministic graph-driven read-once branching programs. Forschungsbericht Universität Dortmund Nr. 772.
6. Bollig, B., Waack, St., and Woelfel, P. (2002). Parity graph-driven read-once branching programs and an exponential lower bound for integer multiplication. To appear in Proc. of 2nd IFIP International Conference on Theoretical Computer Science.
7. Bollig, B. and Woelfel, P. (2002). A Lower Bound Technique for nondeterministic graph-driven read-once branching programs and its applications. To appear in Proc. of MFCS 2002.
8. Brosenne, H., Homeister, M., and Waack, St. (2001). Graph-driven free parity BDDs: algorithms and lower bounds. Proc. of MFCS, 212–223.
9. Bryant, R. E. (1986). Graph-based algorithms for Boolean manipulation. IEEE Trans. on Computers 35, 677–691.
10. Gergov, J. and Meinel, C. (1993). Frontiers of feasible and probabilistic feasible boolean manipulation with branching programs. Proc. of STACS, LNCS 665, 576–585.
11. Gergov, J. and Meinel, C. (1994). Efficient Boolean manipulation with OBDDs can be extended to FBDDs. IEEE Trans. on Computers 43, 1197–1209.
12. Hromkovič, J. (1997). *Communication Complexity and Parallel Computing*. Springer.
13. Jukna, S. Razborov, A., Savický, P., and Wegener, I. (1999). On P versus NP∩CO-NP for decision trees and read-once branching programs. Computational Complexity 8, 357–370.
14. Krause, M. (2002). BDD-based cryptanalysis of keystream generators. To appear in Proc. of EUROCRYPT and ECCC Report TR 01-078.
15. Kushilevitz, E. and Nisan, N. (1997). *Communication Complexity*. Cambridge University Press.
16. Sieling, D. and Wegener, I. (1995). Graph driven BDDs – a new data structure for Boolean functions. Theoretical Computer Science 141, 283–310.
17. Sieling, D. and Wegener, I. (2001). A comparison of free BDDs and transformed BDDs. Formal Methods in System Design 19, 223–236.
18. Wegener, I. (2000). *Branching Programs and Binary Decision Diagrams – Theory and Applications*. SIAM Monographs on Discrete Mathematics and Applications.

ECCC ISSN 1433-8092
http://www.eccc.uni-trier.de/eccc
ftp://ftp.eccc.uni-trier.de/pub/eccc
ftpmail@ftp.eccc.uni-trier.de, subject 'help eccc'