

# Transferred Cash Grows in Size

David Chaum  
CWI  
The Netherlands

Torben Pryds Pedersen\*  
Aarhus University  
Denmark

## Abstract

All known methods for transferring electronic money have the disadvantages that the number of bits needed to represent the money after each payment increases, and that a payer can recognize his money if he sees it later in the chain of payments (forward traceability). This paper shows that it is impossible to construct an electronic money system providing transferability without the property that the money grows when transferred. Furthermore it is argued that an unlimited powerful user can always recognize his money later. Finally, the lower bounds on the size of transferred electronic money are discussed in terms of secret sharing schemes.

## 1 Introduction

Transferability of electronic cash means that the payee in one payment transaction can spend the received money in a later payment to a third person without contacting the bank or another central authority between the two transactions. As on-line electronic payment systems require communication with a central authority during the payment transaction, transferability is only an issue for off-line systems. Although the ability to transfer "normal" money (coins, notes) is very important in our daily life, this property has only received very little attention in relation to electronic money. To the knowledge of the authors, transferability of electronic money has only been described in [vA90], [OO90] and [OO92].

This paper first sketches the (generic) method for transferring electronic cash proposed in [vA90]. At a first glance this method is not ideal, because extra bits are appended to the transferred coins, and a person, whose coin has been transferred a number of times, can always recognize this coin if he sees it later (this property will be referred to as forward traceability).

Intuitively, it is not surprising that the size of transferred money increases, because it must be possible for the bank to identify people, who spends a coin twice. Hence, a transferred coin must contain some information about every person, who has spent it.

This paper formalizes this argument, as it gives lower bounds on the number of bits needed to represent transferred money. These lower bounds depend on whether the systems provide unconditional untraceability or computational untraceability. In

---

\*Research done while visiting CWI

particular it is shown that in case of unconditional untraceability the size of a transferred coin must increase by the number of bits needed to identify the payer, whereas a computational untraceable coin grows with approximately half this number of bits.

It is furthermore argued that a payer with unlimited computing power can always recognize his own money, if he sees it later in the chain of payments.

All statements in this paper are with respect to coin-systems, but it is not hard to see that the results are valid for electronic checks as well. The first section describes a general model of off-line electronic coins and presents the notation which will be used in this paper. In Section 3 it is shown how to add transferability to all known payments systems, and Section 4 and 5 give lower bounds on the size of transferred electronic coins. Section 4 considers payment systems providing unconditional payer untraceability, and Section 5 gives a lower bound for computationally untraceable money. In Section 6 it is argued that these lower bounds are optimal, and Section 7 concludes the paper.

## 2 The Model

This section presents a basic model for off-line electronic cash which will be used in the following.

The results in this paper are independent of whether the payments system provides a protocol for refunding unspent parts of the money. For simplicity, we will therefore assume that the payer always spends his electronic money for its total value (coins). Hence, we consider an off-line electronic payment system involving a bank ( $B$ ) and  $K$  individuals ( $p_1, \dots, p_K$ ) providing protocols for:

1. Withdrawal of money from the bank;
2. Payment transactions from one individual to another; and
3. Deposit (at  $B$ ) of received money.

The system is said to provide transferability, if the payee in on payment transaction can use the received money as a payer in a later payment transaction without talking with the bank (or anybody else) between these two transactions.

The "life-cycle" of an electronic coin in such a system looks like:

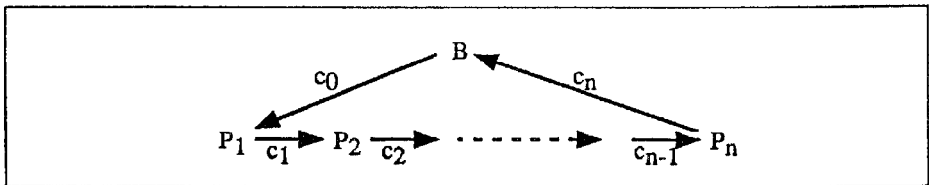


Figure 1: Life-cycle of a coin

Figure 1 illustrates that a person,  $p_1$ , first withdraws  $c_0$  from  $B$  and then spends the coin in a payment transaction. During this transaction  $c_0$  is changed to  $c_1$ . In general, for  $i = 2, 3, \dots, n - 1$ ,  $p_i$  receives  $c_{i-1}$ , and when he later spends it, it is transformed into  $c_i$ . Finally,  $p_n$  deposits the coin at the bank, who receives  $c_n$ .

Unconditional payer untraceability means that even an unlimited powerful bank cannot identify any of  $p_1, p_2, \dots, p_{n-1}$  from  $c_n$ . Computational untraceability means that given  $c_n$ , the bank cannot identify the payers unless it can make a computation, which is thought to be infeasible.

To prevent a payer from using a coin twice, it must be possible for the bank (with very high probability) to discover the identity of such a double-spender. This must be possible even if the coins have been transferred a number of times after the double-spending occurred.

### 3 How to Transfer Electronic Money

This section gives a brief description of the method for transferring electronic money presented in [vA90]. As this method is generic in the sense that it works for a large class of electronic payment systems, it will only be described in general terms here. First a general coin-system is sketched, and then it is extended with transferability (for more details see [CFN90] and [vA90]). It is not difficult to apply this method to electronic checks as well (see [vA90]).

Let the bank have two secret keys  $S_0$  and  $S_1$  with corresponding public keys  $P_0$  and  $P_1$ . A signature with secret key  $S_1$  is worth a fixed amount (say \$1), whereas a signature with  $S_0$  is worth nothing. Both signature schemes must have the property that it is possible to make blind signatures: A user can get a signature  $S_i(m)$  on the message  $m$ , but the signer gets no information about  $m$  (for  $i = 0, 1$ ). See [Cha83] and [Cha84] for examples of such signature schemes.

An electronic coin system can now be constructed as follows.

#### Withdrawal

1. User  $P$  constructs a message  $m_P$  of a special form (see later), and proves that  $m_P$  is constructed correctly (without giving the bank any information about  $m_P$ ).
2. The bank makes a blind signature on  $m_P$  and withdraws \$1 from  $P$ 's account.
3.  $P$  recovers the signature on  $m_P$ .

$P$  can later pay another person,  $R$ , one dollar using the following protocol

#### Payment

1.  $P$  sends  $m_P$  and  $S_1(m_P)$  to  $R$ .
2.  $R$  verifies that  $S_1(m_P)$  is a signature on  $m_P$ , chooses a random challenge,  $c_P$ , and sends it to  $P$ .
3.  $P$  sends back an answer  $r_P$ .
4.  $R$  verifies that  $r_P$  is correct (using  $c_P$  and  $m_P$ ).

In order to prevent double-spending,  $m_P$  must be constructed such that if  $P$  can send correct answers corresponding to two different challenges, then the bank can find  $P$ 's identity from these two answers and  $m_P$ . However, a single correct answer must not give the bank any (Shannon) information about  $P$ 's identity (see [CFN90] and [vA90] for details about how  $P$  can construct  $m_P$  and prove to the bank that it was constructed correctly).

The receiver,  $R$ , can at any time after the payment deposit the electronic coin at the bank (and get \$1):

### Deposit

1.  $R$  sends  $m_P$ ,  $S_1(m_P)$ ,  $c_P$  and  $r_P$  to the bank.
2. The bank verifies the signature and that  $r_P$  is a correct response to the challenge  $c_P$ . Then the bank increases  $B$ 's account with the amount \$1.
3. Finally, the bank searches through its database to see if  $m_P$  has been deposited previously, and in that case it finds the identity of  $P$  (provided the challenges in the two payments are different).

In [CFN90] it is discussed how it can be ensured that  $P$  will always get different challenges, if she tries to spend a coin more than once.

In order to add transferability to this scheme the signature scheme given by  $(S_0, P_0)$  is needed. Furthermore, a one-way function,  $f$ , is required.

Before  $R$  acts as a payee in a payment, he goes to the bank and performs a protocol corresponding to the withdrawal except that the bank gives  $R$  a signature  $S_0(m_R)$ , where  $m_R$  has the same properties as  $m_P$  (in practice,  $R$  would get signatures on many different messages,  $m_R$ , in an initial transaction).

When  $R$  receives the coin given by  $S_1(m_P)$  from  $P$ , he does not choose the challenge at random, but as

$$c_P = f(m_R, \rho_R)$$

where  $\rho_R$  is formed in a special way (to ensure that  $R$  can later deposit the money if he wants to, and to ensure that  $P$  gets different challenges, if he tries to spend the same coin twice — even if  $P$  and  $R$  cooperate).

Later  $R$  can pay the received coin to a third person,  $S$ , without contacting the bank between the two payments:

### Payment of a transferred coin

1.  $R$  sends  $m_P$ ,  $S_1(m_P)$ ,  $r_P$ ,  $m_R$ ,  $S_0(m_R)$  and  $\rho_R$  to  $S$ .
2.  $S$  verifies the two signatures.  
 $S$  verifies, that  $r_P$  is a correct answer to the challenge  $f(m_R, \rho_R)$ .
3.  $S$  computes a challenge,  $c_R$ , and sends it to  $R$ .
4.  $R$  sends an answer  $r_R$ .
5.  $S$  verifies that  $r_R$  is correct (using  $c_R$  and  $m_R$ ).

$S$  can in particular compute the challenge  $c_S$  such that she can later transfer the coin. This method has the advantage, that it is easy to implement in known electronic cash systems, but it has two drawbacks:

1. The money grows in size when transferred (because transcripts of the previous payments are appended and must be verified in each payment).
2. If a payer sees his coin later in the chain of payments, he can recognize it.

It is intuitively clear that the bank can identify double-spenders even if the coins has been transferred a number of times, but it is outside the scope of this paper to state and prove this property formally.

## 4 Unconditionally Untraceable Money

In this section it is shown that the size of electronic money has to grow each time it is transferred, if unconditional payer untraceability is provided.

Consider the tree of payments constructed as follows. A payer,  $p_1$ , withdraws a coin,  $c'_0$ , from the bank and uses it to pay the user  $p_2$ . During this transaction  $c'_0$  is changed to  $c'_1$ . Later  $p_2$  pays this coin to  $p_3$ , and after this transaction the coin has been changed to  $c'_2$ . In general, we consider  $n$  such payments ( $n \in \mathbb{N}$ ), in which  $p_i$  receives the coin  $c'_{i-1}$ , and when he spends it again later, it is transformed into  $c'_i$ . In the following it will be assumed that the prescribed payment protocol is executed *correctly* in all these transactions.

Assume furthermore that each  $p_i$  spends  $c'_{i-1}$  again in another *correct* execution of the payment protocol completely *independent* of  $p_i$ 's first payment. The resulting coin (the payee's output from this transaction) is denoted  $c_i$ . In the following we are going to look at these  $c_i$ 's and forget about the  $c'_i$ 's except  $c'_n$  which we will also denote by  $c_{n+1}$ . Figure 2 shows the relation between the payers and the coins.

The  $c_i$ 's and  $p_i$ 's depend on the random choices in the transactions and the choices of payees. Let therefore  $C_i$  be a random variable whose value is  $c_i$  for  $i = 1, 2, \dots, n, n+1$ , and let  $P_i$  be a random variable whose value is  $p_i$  for  $i = 1, 2, \dots, n$ .

In this section, a lower bound will be given for the entropies of the random variables representing coins. The results are based on elementary information theory as presented in [Wel88] for example. In the following  $H(X)$  denotes the entropy of the finite random variable,  $X$ :

$$H(X) = - \sum_x \text{Prob}(X = x) \log(\text{Prob}(X = x))$$

(all logarithms are with the base 2).

Let  $U$ ,  $V$  and  $W$  be three vectors of finite, random variables. The following rules will be used repeatedly:

$$H(U) \geq 0 \tag{1}$$

$$H(U, V) = H(U | V) + H(V) \tag{2}$$

$$H(U, V | W) = H(U | V, W) + H(V | W) \tag{3}$$

$$H(U | V, W) \leq H(U | V) \tag{4}$$

The following lemma will also be used several times.

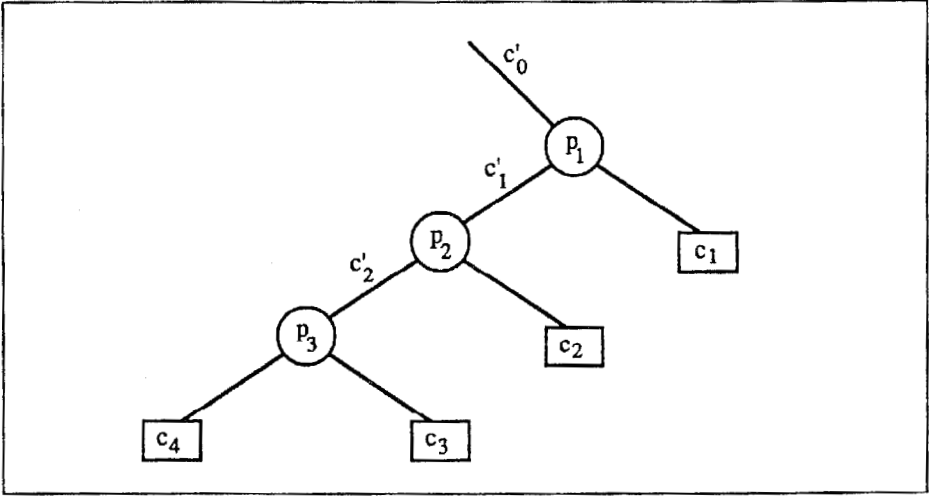


Figure 2: Payment tree for  $n = 3$  — unconditional untraceability

**Lemma 4.1**

Let  $\epsilon > 0$ , and let  $U, V, W$  and  $Z$  be four vectors of finite, random variables. If

$$H(V | W, Z) \leq \epsilon$$

then

$$H(U | W, Z) \leq H(U | V, Z) + \epsilon.$$

**Proof**

$$\begin{aligned} H(U | V, Z) &\geq H(U | V, W, Z) && \text{by (4)} \\ &= H(U, V | W, Z) - H(V | W, Z) && \text{by (3)} \\ &\geq H(U, V | W, Z) - \epsilon \\ &= H(V | U, W, Z) + H(U | W, Z) - \epsilon \\ &\geq H(U | W, Z) - \epsilon && \text{by (1)} \end{aligned}$$

If the payment system provides unconditional payer untraceability then the conditional entropy of  $P_i$  given  $C_i$  equals  $H(P_i)$ :

$$H(P_i | C_i) = H(P_i) \quad \text{for } i = 1, 2, \dots, n.$$

This property can be further strengthened to

$$H(P_i | C_1, C_2, \dots, C_i) = H(P_i) \quad \text{for } i = 1, 2, \dots, n,$$

because of the independence of the payment transactions.

The fact that the bank can identify double-spenders with probability at least  $1-p$ , where  $p$  is the probability that a double-spender is not detected, implies that

$$H(P_i | C_i, C_j) \leq p \log K \quad \text{for } 1 \leq i < j \leq n+1,$$

where  $K$  is the number of possible payers. Let

$$\epsilon = p \log K.$$

In practice  $p$  must be very small (negligible as a function of a security parameter) and then  $\epsilon$  is very small as well.

**Theorem 4.2**

$$H(C_{n+1}) \geq H(P_1) + H(P_2) + \dots + H(P_n) - 2n\epsilon.$$

**Proof**

Claim: For  $1 \leq i \leq n$ :

$$H(C_{n+1} | C_1, C_2, \dots, C_i) \geq H(C_{n+1} | C_1, C_2, \dots, C_i, C_{i+1}) + H(P_i) - 2\epsilon.$$

From this claim it follows by simple induction that

$$\begin{aligned} H(C_{n+1}) &\geq H(C_{n+1} | C_1) \\ &\geq H(C_{n+1} | C_1, C_2) + H(P_1) - 2\epsilon \\ &\dots \\ &\geq H(C_{n+1} | C_1, C_2, \dots, C_i) + \sum_{j=1}^{i-1} H(P_j) - 2\epsilon(i-1) \\ &\dots \\ &\geq H(C_{n+1} | C_1, C_2, \dots, C_n) + \sum_{j=1}^n H(P_j) - 2\epsilon n \\ &= \sum_{j=1}^n H(P_j) - 2\epsilon n \end{aligned}$$

In order to prove the claim, let  $i \in \{1, 2, \dots, n\}$  be given, and let  $A_i$  denote the vector  $(C_1, C_2, \dots, C_i)$ . Then

$$\begin{aligned} H(C_{n+1} | A_i) &= H(P_i, C_{n+1} | A_i) - H(P_i | C_{n+1}, A_i) \quad \text{by (3)} \\ &\geq H(P_i, C_{n+1} | A_i) - \epsilon \\ &= H(C_{n+1} | P_i, A_i) + H(P_i | A_i) - \epsilon \quad \text{by (3)} \\ &= H(C_{n+1} | P_i, A_i) + H(P_i) - \epsilon \\ &\geq H(C_{n+1} | C_{i+1}, A_i) + H(P_i) - 2\epsilon \end{aligned}$$

where last inequality follows from Lemma 4.1 and the fact that  $H(P_i | C_{i+1}, A_i) \leq \epsilon$ . ■

As the entropy is a measure of the number of bits in optimal encodings, Theorem 4.2 implies that the number of bits needed to represent an electronic coin grows

each time the coin is transferred. Furthermore, the increase is the number of bits needed to identify the new payer. In particular, if  $H(P_i) = k$  for every  $i$  (the bank needs  $k$  bits of information to identify each payer) we see that

$$H(C_{n+1}) \geq kn - 2n\epsilon,$$

and by the symmetry of  $C_n$  and  $C_{n+1}$  we get

$$H(C_n) \geq n(k - 2\epsilon).$$

As  $\epsilon \ll k$  the coin grows with the number of bits needed to identify the payer each time the coin is transferred.

We conclude this section with a short remark on forward traceability. If all secret keys of the bank are uniquely determined by the bank's public key, then a payer with unlimited computing power can always determine, given a transferred coin, if he has previously had this coin in his possession:

1. Simulate another payment of his original coin.
2. Compute the secret key of the bank (using the unlimited power).
3. Determine the identity of the double-spender (as the bank would have done).

## 5 Computationally Untraceable Money

In the previous section we saw that unconditionally untraceable, electronic money must grow in size when transferred. In this section a similar result is proven for computationally untraceable money.

The tree considered in the previous section is not sufficient to give an interesting lower bound on the size of computational untraceable money. This is due to the fact that in such a system each  $c_i$  could, in principle, contain all information needed to identify  $p_i$ , but no additional information about the previous payers.

The proof in this section is therefore based on a tree of payments constructed as follows. First,  $p_0^1$  receives a coin,  $c$ , from the bank, and then he chooses two payees,  $p_0^2$  and  $p_1^2$ , at random among all individuals in the system and pays both of them in *correct and independent* executions of the payment protocol. As the money system provides transferability  $p_0^2$  and  $p_1^2$  can later, independently of each other, spend the received coin twice in a similar way. In general, for  $j \geq 1$  and  $0 \leq i < 2^{j-1}$ ,  $p_i^j$  transfers a received coin to  $p_{2i}^{j+1}$  and  $p_{2i+1}^{j+1}$ . After some time, the original coin has been changed to  $c_0, c_1, \dots, c_{2^n-1}$ , where  $p_i^n$  is the payer of  $c_{2i}$  and  $c_{2i+1}$  for  $i = 0, 1, \dots, 2^{n-1} - 1$ .

Let  $C_i$  be a random variable with value  $c_i$ , and let  $P_i^j$  be a random variable whose value is the identity of  $p_i^j$ . Figure 3 shows how  $C_i$  is related to each  $P_i^j$  for  $n = 3$ .

For any  $i = 0, 1, \dots, 2^n - 1$ , let  $P_{i,1}, P_{i,2}, \dots, P_{i,n}$  be the path from the root to  $C_i$  in the tree of payments ( $P_{i,1} = P_0^1$  for all  $i$ ). Hence,  $P_i^j$  ( $0 \leq i < 2^{j-1}$ ) denotes the  $i$ 'th payer (from left) at depth  $j - 1$ , whereas  $P_{i,j}$  ( $0 \leq i < 2^n$ ) denotes then  $j$ 'th payer of the coin which is finally transferred to  $C_i$ . In both cases  $1 \leq j \leq n$ .

Furthermore, for any pair  $(i, j)$  where  $0 \leq i < 2^n$  and  $1 \leq j \leq n$ , let  $T_{i,j}$  be the sub-tree of height  $n - (j - 1)$  having  $P_{i,j}$  as root. The leaves in this subtree can be numbered from 0 to  $2^{n+1-j} - 1$  from left to right. Let  $C_i$  have number  $k$  in this



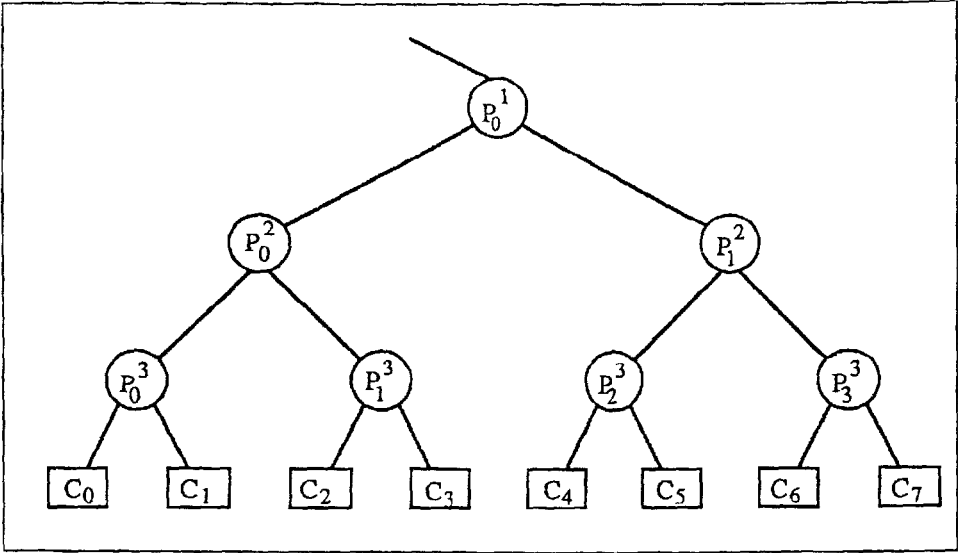


Figure 3: Payment tree for random variables ( $n = 3$ ) — computational untraceability

enumeration. Then  $f(i, j)$  is defined as the index of the leaf with number  $2^{n+1-j} - 1 - k$  in this enumeration ( $C_{f(i,j)}$  is “symmetric” to  $C_i$  in the sub-tree  $T_{i,j}$ ). For example for  $n = 3$

$$\begin{aligned} f(0, 1) &= 7 \\ f(2, 1) &= 5 \\ f(4, 2) &= 7 \\ f(0, 3) &= 1 \end{aligned}$$

The function  $i \mapsto f(i, j)$  is a bijection for every  $j \in \{1, 2, \dots, n\}$  because

$$f(f(i, j), j) = i.$$

Let  $p$  be the maximal probability with which a user can spend a coin twice without being identified, and let

$$\epsilon = p \log K,$$

where  $K$  is the number of possible participants. The property that the bank can identify double-spenders can be expressed in terms of entropies as follows. Given two leaves  $C_i$  and  $C_k$  ( $i \neq k$ ), let  $j$  be maximal such that  $C_k$  is a leaf in  $T_{i,j}$ . Then  $C_i$  and  $C_k$  are both (transferred) results of double-spending by  $p_{i,j}$ . Hence

$$H(P_{i,j} | C_i, C_k) \leq \epsilon \quad (*).$$

In particular this implies that

$$H(P_{i,j} | C_i, C_{f(i,j)}) \leq \epsilon.$$

Now consider the subtree,  $\bar{T}_{i,j}$ , defined as the entire tree, but with the tree  $T_{i,j}$  removed ( $\bar{T}_{2,3}$  for  $n = 3$  is shown in figure 4).

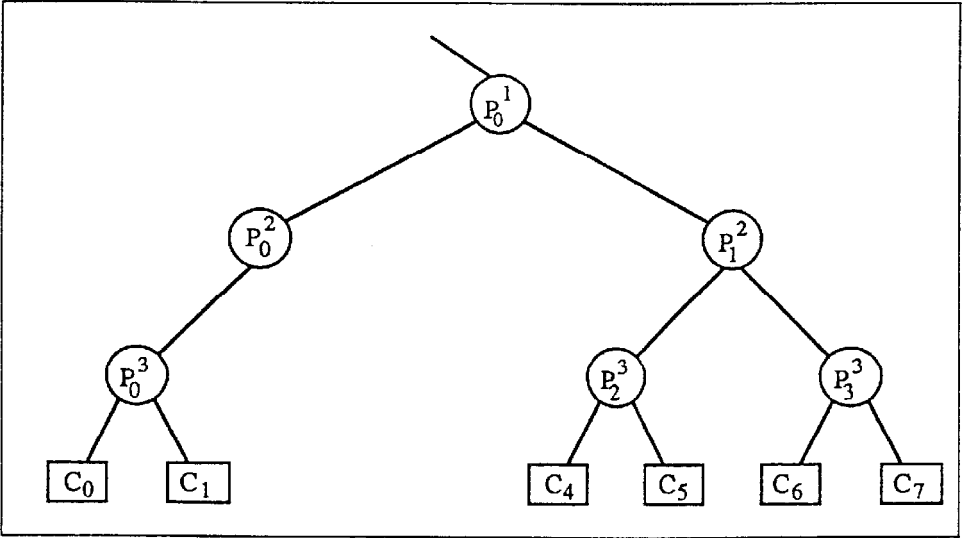


Figure 4:  $\bar{T}_{2,3}$  for  $n = 3$

In the construction of the tree it was required that each payer spends the received coin twice in two independent payments to two independently chosen payees. This means that for any vector,  $V$ , of random variables in  $\bar{T}_{i,j}$  (nodes and leaves):

$$H(P_{i,j} | V) = H(P_{i,j})$$

(the independence property). As this section considers payment systems, which do not (necessarily) offer unconditional untraceability, the proof of the following theorem is based on (\*) and the independence property.

**Theorem 5.1**

In a tree of depth  $n$ :

$$\sum_{i=0}^{2^n-1} H(C_i) \geq \sum_{j=1}^n [ \sum_{i=0}^{2^{j-1}-1} 2^{n-j} H(P_i^j) ] - 2^{n-1} 3n\epsilon$$

As entropies are always positive this shows that electronic coins must grow in size when transferred. To be more concrete, consider the case where the uncertainty about each payer is  $k$  ( $k$  bits of information are needed to uniquely identify a payer). Then the theorem implies that

$$\begin{aligned} \sum_{i=0}^{2^n-1} H(C_i) &\geq \sum_{j=1}^n [ \sum_{i=0}^{2^{j-1}-1} 2^{n-j} k ] - 2^n n \frac{3}{2} \epsilon \\ &= n 2^n \left( \frac{k - 3\epsilon}{2} \right) \end{aligned}$$

In particular this means that the entropy of some  $C_i$  is at least  $n \frac{k-3\epsilon}{2}$ , and if the entropies of all  $C_i$ 's are equal then

$$H(C_i) \geq n \left( \frac{k - 3\epsilon}{2} \right).$$

This shows that the entropy of the coins grows linearly in the number of transfers, and furthermore, that the coins grow by approximately  $\frac{k}{2}$  bits when transferred (since  $k \gg 3\epsilon$ ).

This is less than for unconditionally secure money. The difference is due to the fact that a coin,  $c_i$ , in principle may contain all information needed to identify a payer as long as the bank cannot compute this identity from the coin. Hence, the uncertainty about another coin spent by the same person can be very small. Furthermore, Theorem 5.1 is tight in a sense to be discussed in Section 6.

The theorem is proven by combining two lemmas. The first lemma gives a lower bound on  $H(C_i)$ , and the second gives an upper bound on  $H(P_i^j)$ . The proofs use the same four rules as in the previous section. For convenience these are repeated here:

$$H(U) \geq 0 \quad (1)$$

$$H(U, V) = H(U | V) + H(V) \quad (2)$$

$$H(U, V | W) = H(U | V, W) + H(V | W) \quad (3)$$

$$H(U | V, W) \leq H(U | V) \quad (4)$$

### Lemma 5.2

For every  $i$  ( $0 \leq i < 2^n$ )

$$H(C_i) \geq \sum_{j=1}^n H(P_{i,j} | C_{f(i,1)}, \dots, C_{f(i,j)}) - 2n\epsilon$$

### Proof

The proof is very similar to that of Theorem 4.2.

Claim: For  $1 \leq j < n$ :

$$H(C_i | C_{f(i,1)}, \dots, C_{f(i,j)}) \geq H(C_i | C_{f(i,1)}, \dots, C_{f(i,j+1)}) + H(P_{i,j} | C_{f(i,1)}, \dots, C_{f(i,j)}) - 2\epsilon$$

From this claim it follows by simple induction that

$$\begin{aligned} H(C_i) &\geq H(C_i | C_{f(i,n)}, C_{f(i,n-1)}, \dots, C_{f(i,1)}) + \\ &\quad \sum_{j=1}^{n-1} H(P_{i,j} | C_{f(i,j)}, \dots, C_{f(i,1)}) - 2(n-1)\epsilon \\ &\geq H(P_{i,n} | C_{f(i,n)}, C_{f(i,n-1)}, \dots, C_{f(i,1)}) + \\ &\quad \sum_{j=1}^{n-1} H(P_{i,j} | C_{f(i,j)}, \dots, C_{f(i,1)}) - (2n-1)\epsilon \\ &\geq H(P_{i,n} | C_{f(i,n)}, C_{f(i,n-1)}, \dots, C_{f(i,1)}) + \\ &\quad \sum_{j=1}^{n-1} H(P_{i,j} | C_{f(i,j)}, \dots, C_{f(i,1)}) - 2n\epsilon \end{aligned}$$

because

$$H(P_{i,n} | C_i, C_{f(i,n)}) \leq \epsilon$$

implies that

$$H(C_i | C_{f(i,n)}, \dots, C_{f(i,1)}) \geq H(P_{i,n} | C_{f(i,n)}, \dots, C_{f(i,1)}) - \epsilon.$$

The proof of this is very similar to that of Lemma 4.1. Now we just have to prove the claim. Let  $i$  and  $j$  be given, and let

$$A_j := (C_{f(i,1)}, \dots, C_{f(i,j)}).$$

Then

$$\begin{aligned} H(C_i | C_{f(i,1)}, \dots, C_{f(i,j)}) &= H(C_i | A_j) \\ &= H(P_{i,j}, C_i | A_j) - H(P_{i,1} | C_i, A_j) \quad \text{by (3)} \\ &\geq H(P_{i,j}, C_i | A_j) - \epsilon \\ &= H(C_i | P_{i,j}, A_j) + H(P_{i,j} | A_j) - \epsilon \quad \text{by (4)} \end{aligned}$$

By Lemma 4.1

$$H(P_{i,j} | C_{f(i,j+1)}, A_j) \leq \epsilon$$

implies that

$$H(C_i | P_{i,j}, A_j) \geq H(C_i | C_{f(i,j+1)}, A_j) - \epsilon.$$

Thus

$$\begin{aligned} H(C_i) &\geq H(C_i | C_{f(i,j+1)}, A_j) + H(P_{i,j} | A_j) - 2\epsilon \\ &= H(C_i | C_{f(i,1)}, \dots, C_{f(i,j+1)}) + H(P_{i,j} | C_{f(i,1)}, \dots, C_{f(i,j)}) - 2\epsilon \end{aligned}$$

This completes the proof. ■

In order to give an upper bound on  $H(P_i^j)$  it is necessary to introduce some more notation. Consider the sub-tree  $T_{k,j}$  of height  $n - j + 1$ . This tree has two sub-trees  $T_{k,j+1}$  and  $T_{f(k,j),j+1}$  of height  $n - j$ . We define  $B_{k,j}$  to be the set of leaves in the subtree, which *does not* contain  $C_k$ . Hence for  $0 \leq k < 2^n$  and  $1 \leq j < n$  is  $B_{k,j}$  defined as the set of leaves in the subtree  $T_{f(k,j),j+1}$ . For example, for  $n = 3$ :

$$\begin{aligned} B_{2,1} &= \{C_4, C_5, C_6, C_7\} \\ B_{3,2} &= \{C_0, C_1\} \\ B_{6,2} &= \{C_4, C_5\} \\ B_{4,3} &= \{C_5\} \end{aligned}$$

$B_{k,j}$  has the property that each element in  $B_{k,j}$  is a leaf in  $\bar{T}_{k,j+1}$ , and the set of all leaves in  $\bar{T}_{k,j+1}$  for  $1 \leq j \leq n - 1$  is

$$A_{k,j} := B_{k,1} \cup B_{k,2} \cup \dots \cup B_{k,j}.$$

Furthermore,  $T_{k,j}$  is the smallest subtree containing  $C_k$  and  $C$  for every element  $C \in B_{k,j}$ . Thus

$$H(P_{k,j} | C, C_k) \leq \epsilon.$$

### Lemma 5.3

For  $0 \leq k < 2^n$ :

$$\sum_{j=2}^n H(P_{k,j}) \leq H(C_k | P_0^1) + \sum_{j=2}^n H(P_{k,j} | C_k, B_{k,1}, B_{k,2}, \dots, B_{k,j-1}) + n\epsilon$$

**Proof**

Let  $k$  be given. Since  $A_{k,j-1}$  is the set of leaves in  $\overline{T}_{k,j}$  for  $2 \leq j \leq n$ , the independence property implies that

$$H(P_{k,j}) = H(P_{k,j} | P_{k,j-1}, A_{k,j-1})$$

(this is the only time the independence property is used). Now

$$\begin{aligned} \sum_{j=2}^n H(P_{k,j}) &= \sum_{j=2}^n H(P_{k,j} | P_{k,j-1}, A_{k,j-1}) \\ &= \sum_{j=2}^n [H(C_k, P_{k,j} | P_{k,j-1}, A_{k,j-1}) - H(C_k | P_{k,j}, P_{k,j-1}, A_{k,j-1})] \quad \text{by (3)} \\ &= \sum_{j=2}^n [H(P_{k,j} | C_k, P_{k,j-1}, A_{k,j-1}) + H(C_k | P_{k,j-1}, A_{k,j-1})] - \\ &\quad \sum_{j=2}^n H(C_k | P_{k,j}, P_{k,j-1}, A_{k,j-1}) \quad \text{by (3)} \\ &= \sum_{j=2}^n H(P_{k,j} | C_k, P_{k,j-1}, A_{k,j-1}) + H(C_k | P_{k,1}, A_{k,1}) + \\ &\quad \sum_{j=2}^{n-1} [H(C_k | P_{k,j}, A_{k,j}) - H(C_k | P_{k,j}, P_{k,j-1}, A_{k,j-1})] - \\ &\quad H(C_k | P_{k,n}, P_{k,n-1}, A_{k,n-1}) \\ &\leq \sum_{j=2}^n H(P_{k,j} | C_k, A_{k,j-1}) + H(C_k | P_{k,1}, A_{k,1}) + \\ &\quad \sum_{j=2}^{n-1} [H(C_k | P_{k,j}, A_{k,j}) - H(C_k | P_{k,j}, P_{k,j-1}, A_{k,j-1})] - \\ &\quad H(C_k | P_{k,n}, P_{k,n-1}, A_{k,n-1}) \quad \text{by (4)} \\ &\leq \sum_{j=2}^n H(P_{k,j} | C_k, A_{k,j-1}) + H(C_k | P_{k,1}, B_{k,1}) + \\ &\quad \sum_{j=2}^{n-1} [H(C_k | P_{k,j}, A_{k,j}) - H(C_k | P_{k,j}, P_{k,j-1}, A_{k,j-1})] \quad \text{by (1)} \\ &\leq \sum_{j=2}^n H(P_{k,j} | C_k, A_{k,j-1}) + H(C_k | P_0^1) + \\ &\quad \sum_{j=2}^{n-1} [H(C_k | P_{k,j}, A_{k,j}) - H(C_k | P_{k,j}, P_{k,j-1}, A_{k,j-1})] \quad \text{by (4)} \end{aligned}$$

Due to the facts that the elements in  $B_{k,j}$  are leaves in  $T_{k,j}$ , and  $B_{k,j-1}$  is the set of leaves in  $T_{f(k,j-1),j}$ , and  $P_{k,j-1} = P_{f(k,j-1),j-1}$ , (\*) implies that for every  $C \in B_{k,j}$  and  $D \in B_{k,j-1}$

$$H(P_{k,j-1} | C, D) \leq \epsilon.$$

Hence,

$$H(P_{k,j-1} | B_{k,j}, B_{k,j-1}) \leq \epsilon$$

for  $2 \leq j \leq n-1$ . Using (4), we get

$$H(P_{k,j-1} | B_{k,j}, A_{k,j-1}, P_{k,j-1}) \leq \epsilon,$$

and Lemma 4.1 implies

$$\begin{aligned} H(C_k | P_{k,j}, A_{k,j}) &= H(C_k | B_{k,j}, A_{k,j-1}, P_{k,j}) \\ &\leq H(C_k | P_{k,j}, P_{k,j-1}, A_{k,j-1}) + \epsilon. \end{aligned}$$

Hence

$$\sum_{j=2}^n H(P_{k,j}) \leq H(C_k | P_0^1) + \sum_{j=2}^n H(P_{k,j} | C_k, A_{k,j-1}) + n\epsilon.$$

We are now ready to present a proof of Theorem 5.1: ■

### Proof

By using Lemma 5.2 for all even indices we get

$$\begin{aligned} \sum_{i=0}^{2^n-1} H(C_i) &= \sum_{i=0}^{2^{n-1}-1} H(C_{2i}) + \sum_{i=0}^{2^{n-1}-1} H(C_{2i+1}) \\ &\geq \sum_{i=0}^{2^{n-1}-1} \sum_{j=1}^n H(P_{2i,j} | C_{f(2i,1)}, \dots, C_{f(2i,j)}) - 2^{n-1}2n\epsilon + \\ &\quad \sum_{i=0}^{2^{n-1}-1} H(C_{2i+1}) \\ &= \sum_{i=0}^{2^{n-1}-1} \sum_{j=2}^n H(P_{2i,j} | C_{f(2i,1)}, \dots, C_{f(2i,j)}) - 2^{n-1}2n\epsilon + \\ &\quad \sum_{i=0}^{2^{n-1}-1} H(P_{2i,1} | C_{f(2i,1)}) + \sum_{i=0}^{2^{n-1}-1} H(C_{2i+1}) \end{aligned}$$

Since  $f(2i, 1)$  is always odd, and since  $f(\cdot, 1)$  is a permutation

$$\sum_{i=0}^{2^{n-1}-1} H(C_{2i+1}) = \sum_{i=0}^{2^{n-1}-1} H(C_{f(2i,1)})$$

Using (2) twice this implies

$$\begin{aligned} \sum_{i=0}^{2^n-1} H(C_i) &\geq \sum_{i=0}^{2^{n-1}-1} \sum_{j=2}^n H(P_{2i,j} | C_{f(2i,1)}, \dots, C_{f(2i,j)}) - 2^{n-1}2n\epsilon + \\ &\quad \sum_{i=0}^{2^{n-1}-1} H(P_{2i,1}, C_{f(2i,1)}) \\ &= \sum_{i=0}^{2^{n-1}-1} \sum_{j=2}^n H(P_{2i,j} | C_{f(2i,1)}, \dots, C_{f(2i,j)}) - 2^{n-1}2n\epsilon + \\ &\quad \sum_{i=0}^{2^{n-1}-1} H(C_{f(2i,1)} | P_{2i,1}) + \sum_{i=0}^{2^{n-1}-1} H(P_{2i,1}) \quad (**) \end{aligned}$$

Now consider the sum

$$\sum_{i=0}^{2^{n-1}-1} H(P_{2i,j} | C_{f(2i,1)}, \dots, C_{f(2i,j)})$$

for a fixed  $j$  ( $2 \leq j \leq n$ ). Let  $k = f(2i, j)$ . Then

$$P_{2i,j} = P_{k,j}$$

and hence

$$C_{f(2i,s)} \in B_{k,s},$$

for  $1 \leq s \leq j-1$ , but  $C_k$  is not in  $B_{k,j}$ . By (4) this implies

$$\begin{aligned} H(P_{2i,j} | C_{f(2i,1)}, \dots, C_{f(2i,j)}) &= H(P_{k,j} | C_{f(2i,1)}, \dots, C_{f(2i,j)}) \\ &\geq H(P_{k,j} | B_{k,1}, \dots, B_{k,j-1}, C_{f(2i,j)}) \\ &= H(P_{k,j} | B_{k,1}, \dots, B_{k,j-1}, C_k) \end{aligned}$$

Since  $f(2i, j)$  is odd and  $f(\cdot, j)$  is a permutation we obtain that (writing  $k = f(2i, j)$  as  $2l+1$ )

$$\sum_{i=0}^{2^{n-1}-1} \sum_{j=2}^n H(P_{2i,j} | C_{f(2i,1)}, \dots, C_{f(2i,j)}) \geq \sum_{l=0}^{2^{n-1}-1} \sum_{j=2}^n H(P_{2l+1,j} | B_{2l+1,1}, \dots, B_{2l+1,j-1}, C_{2l+1})$$

and by Lemma 5.3

$$\begin{aligned} &\sum_{i=0}^{2^{n-1}-1} \sum_{j=2}^n H(P_{2i,j} | C_{f(2i,1)}, \dots, C_{f(2i,j)}) \\ &\geq \sum_{l=0}^{2^{n-1}-1} [\sum_{j=2}^n H(P_{2l+1,j}) - H(C_{2l+1} | P_0^1) - n\epsilon] \\ &= \sum_{l=0}^{2^{n-1}-1} [\sum_{j=2}^n H(P_{2l+1,j}) - H(C_{2l+1} | P_0^1)] - 2^{n-1}n\epsilon \quad (***) \end{aligned}$$

Combining (\*\*) and (\*\*\*) results in

$$\begin{aligned} \sum_{i=0}^{2^n-1} H(C_i) &\geq \sum_{i=0}^{2^{n-1}-1} \sum_{j=2}^n H(P_{2i,j} | C_{f(2i,1)}, \dots, C_{f(2i,j)}) - 2^{n-1}2n\epsilon + \\ &\quad \sum_{i=0}^{2^{n-1}-1} H(C_{f(2i,1)} | P_{2i,1}) + \sum_{i=0}^{2^{n-1}-1} H(P_{2i,1}) \quad \text{by (**)} \\ &\geq \sum_{l=0}^{2^{n-1}-1} [\sum_{j=2}^n H(P_{2l+1,j}) - H(C_{2l+1} | P_0^1)] - 2^{n-1}n\epsilon - 2^{n-1}2n\epsilon + \\ &\quad \sum_{i=0}^{2^{n-1}-1} H(C_{f(2i,1)} | P_{2i,1}) + \sum_{i=0}^{2^{n-1}-1} H(P_{2i,1}) \quad \text{by (***)} \\ &= \sum_{l=0}^{2^{n-1}-1} [\sum_{j=2}^n H(P_{2l+1,j}) - H(C_{2l+1} | P_0^1)] - 2^{n-1}3n\epsilon + \\ &\quad \sum_{l=0}^{2^{n-1}-1} H(C_{2l+1} | P_0^1) + \sum_{l=0}^{2^{n-1}-1} H(P_0^1) \\ &= \sum_{l=0}^{2^{n-1}-1} \sum_{j=2}^n H(P_{2l+1,j}) + \sum_{l=0}^{2^{n-1}-1} H(P_0^1) - 2^{n-1}3n\epsilon \end{aligned}$$

$$\begin{aligned}
&=^*) \sum_{j=2}^n \sum_{i=0}^{2^{j-1}-1} 2^{n-j} H(P_i^j) + 2^{n-1} H(P_0^1) - 2^{n-1} 3n\epsilon \\
&= \sum_{j=1}^n \sum_{i=0}^{2^{j-1}-1} 2^{n-j} H(P_i^j) - 2^{n-1} 3n\epsilon
\end{aligned}$$

At \*) we use the fact that for every  $j = 1, 2, \dots, n$

$$\sum_{l=0}^{2^{n-1}-1} H(P_{2l+1,j}) = \sum_{i=0}^{2^{j-1}-1} 2^{n-j} H(P_i^j).$$

This completes the proof. ■

## 6 Applications to Secret Sharing

Theorem 4.2 and 5.1 give lower bounds on the size of transferred electronic money. In this section these theorems will be discussed from a different point of view.

Consider the tree in figure 2. If  $p_1, p_2, \dots, p_n$  are considered as  $k$ -bits secrets and  $c_1, c_2, \dots, c_{n+1}$  as shares of these secrets, then this tree depicts a situation, where a person (dealer) has  $n$  secrets and wants to distribute them among  $n + 1$  persons in an information theoretic secure way, such that for every  $i = 1, 2, \dots, n$ , it is possible to find  $p_i$  from  $c_i$  and  $c_j$ , where  $i < j \leq n + 1$ . Theorem 4.2 says that the share  $c_{n+1}$  must be at least  $nk$  bits. It is not hard to generalize the theorem to show that each  $c_i$  must be at least  $ik$  bits long for  $1 \leq i \leq n$ . These lower bounds on the sizes of shares are also optimal as they can be achieved by choosing  $r_1, \dots, r_n \in \{0, 1\}^k$  at random and letting

$$\begin{aligned}
c_1 &= r_1 \oplus p_1 \\
c_i &= (r_1, \dots, r_{i-1}, r_i \oplus p_i) \quad \text{for } i = 2, \dots, n \\
c_{n+1} &= (r_1, r_2, \dots, r_n)
\end{aligned}$$

The result in Section 5 can also be described in terms of secret sharing schemes — although this time the schemes are somewhat unusual. Again we consider the identities of the payers to be secrets and the coins to be shares of the secrets. Hence, there are  $2^n - 1$  secrets and  $2^n$  persons, who get shares  $c_0, \dots, c_{2^n-1}$ . The access structure is defined by the tree described in Section 5.

It is required that the share,  $c_i$  may only contain Shannon information about the secrets on the path from the root to itself. Hence,  $c_2$  may for instance contain all Shannon information about  $p_0^1, p_0^2$  and  $p_1^3$ , but it may not contain information about other secrets. If each  $p_i^j$  is a uniformly chosen  $k$ -bits secret Theorem 5.1 says that

$$\sum_{i=0}^{2^n-1} H(C_i) \geq n2^{n-1}k - 2^{n-1}n3\epsilon.$$

Again, this result is optimal, as it is possible to give values to each  $c_i$  such that  $\epsilon = 0$  and

$$\sum_{i=0}^{2^n-1} H(C_i) = n2^{n-1}k.$$



Assume that  $k$  is even. Let  $\lfloor x \rfloor$  denote the first  $\frac{k}{2}$  bits of  $x \in \{0,1\}^k$ , and let  $\lceil x \rceil$  denote the last  $\frac{k}{2}$  bits of  $x$ . For the case  $n = 8$ , the lower bound can now be achieved as follows:

$$\begin{aligned} c_0 &:= (\lfloor p_0^1 \rfloor, \lfloor p_0^2 \rfloor, \lfloor p_0^3 \rfloor) \\ c_1 &:= (\lfloor p_0^1 \rfloor, \lfloor p_0^2 \rfloor, \lceil p_0^3 \rceil) \\ c_2 &:= (\lfloor p_0^1 \rfloor, \lceil p_0^2 \rceil, \lfloor p_1^3 \rfloor) \\ &\dots \\ c_7 &:= (\lceil p_0^1 \rceil, \lceil p_1^2 \rceil, \lceil p_3^3 \rceil) \end{aligned}$$

Here each share consists of  $\frac{3k}{2}$  bits as required.

The secrets  $p_i^j$  can also be shared in an information theoretic secure way. This can be done by letting each share  $c_i$  consist of  $3k$  bits. This shows that the lower bound from Section 4 can not be improved using the tree from Section 5.

## 7 Conclusion

This paper has demonstrated that it is not possible to construct off-line electronic payment systems without allowing extra bits for transferred money. On one hand this limits the practical use of electronic money, and on the other it shows that the general method described in Section 3 is close to optimal, because the transferred money in this scheme only increases by the number of bits needed to identify double-spenders with high probability. However, the known payment systems require more bits for this purpose than the actual number of bits needed to uniquely describe a payer. It would therefore be interesting to construct an off-line payment for which fewer bits are needed to compute the identity of double-spenders.

It was mentioned that the method suggested in [vA90] has the problem of forward traceability, and it was further argued in Section 4 that a person with unlimited computing power can probably always trace his money forwards. This leaves open the following problems:

- Prove that an unlimited powerful payer can always trace his transferred money;
- Construct a payment system in which forward traceability, although possible, is not feasible (under some assumption).

## Acknowledgements

We wish to thank Eugène van Heyst for many useful comments during the construction of the lower bounds. We also wish to thank David B. Wilson for some comments on an earlier version of this paper.

## References

- [CFN90] D. Chaum, A. Fiat, and M. Naor. Untraceable electronic cash. In *Advances in Cryptology - proceedings of CRYPTO 88*, Lecture Notes in Computer Science, pages 319 – 327. Springer-Verlag, 1990.

- [Cha83] D. Chaum. Blind signatures for untraceable payments. In *Advances in Cryptology - proceedings of CRYPTO 82*, pages 199 – 203, 1983.
- [Cha84] D. Chaum. Blind signature systems. In *Advances in Cryptology - proceedings of CRYPTO 83*, 1984.
- [OO90] T. Okamoto and K. Ohta. Disposable zero-knowledge authentication and their applications to untraceable electronic cash. In *Advances in Cryptology - proceedings of CRYPTO 89*, pages 481 – 496, 1990.
- [OO92] T. Okamoto and K. Ohta. Universal electronic cash. In J. Feigenbaum, editor, *Advances in Cryptology - proceedings of CRYPTO 91*, Lecture Notes in Computer Science, pages 324 – 337. Springer-Verlag, 1992.
- [vA90] Hans van Antwerpen. Electronic cash. Master's thesis, CWI, 1990.
- [Wel88] Dominic Welsh. *Codes and Cryptography*. Oxford University Press, 1988.