Electronic Colloquium on Computational Complexity, Revision 1 of Report No. 9 (2003)

Private Computation — k-connected versus 1-connected Networks*

Markus Bläser Andreas Jakoby
Maciej Liśkiewicz† Bodo Manthey‡

Institut für Theoretische Informatik
Universität zu Lübeck
Wallstraße 40, 23560 Lübeck, Germany
blaeser/jakoby/liskiewi/manthey@tcs.uni-luebeck.de

Abstract

We study the role of connectivity of communication networks in private computations under information theoretic settings. We show that some functions can be computed by private protocols even if the underlying network is 1-connected but not 2-connected. Then we give a complete characterisation of non-degenerate functions that can be computed on non-2-connected networks.

Furthermore, we present a general technique for simulating private protocols on arbitrary networks. Using this technique every private protocol can be simulated on arbitrary k-connected networks using only a small number of additional random bits.

Finally, we give matching lower and upper bounds for the number of random bits needed to compute the parity function on k-connected networks.

Keywords: private computation, connectivity, parity.

1 Introduction

Consider a set of players, each knowing an individual secret. The goal is to compute a function depending on these secrets such that after the computation none of the players knows anything about the secrets of the other players that cannot be derived from function value and its own secret. An example for such a computation is the "secret ballot problem": The members of a committee wish to decide whether the majority

*A preliminary version appeared in *Proc. of the 22nd Ann. Int. Cryptology Conf. (CRYPTO 2002)*, volume 2442 of *Lecture Notes in Comput. Sci.*, pages 194–209, Springer, 2002.

†On leave from Instytut Informatyki, Uniwersytet Wrocławski, Poland.

‡Birth name: Bodo Siebert. Supported by DFG research grant Re 672/3.

ISSN 1433-8092

votes for yes or no. But after the vote nobody should know anything about the opinions of the other committee members, not even about the exact number of yes and no votes, except for whether the majority has voted for yes or no. To come to a decision, any two members can talk to each other in private. If however the members are distributed in a network, then only those members can talk to each other that are connected by a link. In this work, we investigate the influence of the underlying network on the ability to perform private computations.

Depending on the computational power of the players we distinguish between cryptographically secure privacy and privacy in information theoretic sense. In the first case we assume that no player is able to recompute any information about the input within polynomial time (see e.g. [5, 16, 22, 23]). In the second case we do not restrict the computational power of the players (see e.g. [3, 6]). Hence, the latter notion of privacy is much stronger than the former. In this paper, we use the information theoretic approach.

1.1 Previous Results

Private computation has been the subject of a considerable amount of research. Traditionally, one investigates the number of rounds and random bits as complexity measures for private protocols. Chor and Kushilevitz [10] have studied the number of rounds necessary to compute the sum modulo an integer. This function has also been investigated by Blundo et al. [4] and Chor et al. [8]. The number of random bits needed to compute the parity function, i.e. the sum modulo 2, has been examined by Kushilevitz and Mansour [18] and Kushilevitz and Rosén [20]. Gál and Rosén [14] have shown that the parity function cannot be computed by any private protocol in $o(\log n/\log d)$ rounds using d random bits. They have also given an almost tight randomness-round tradeoff for private computations of arbitrary Boolean functions depending on their sensitivity. Bounds on the maximum number of rounds needed in the worst-case to compute a function by a private protocol have been given by Bar-Ilan and Beaver [2] and by Kushilevitz [17].

The number of random bits necessary to compute a Boolean function by a private protocol is closely related to its circuit size. Kushilevitz et al. [19] have shown that every function that can be computed with linear circuit size can also be computed by a private protocol with only a constant number of random bits. Using this result one can show that the majority function can be computed by a private protocol using a constant number of random bits and simultaneously a linear number of bits exchanged between players (for the circuit complexity of majority see e.g. [21]).

So far we have assumed that players do not attempt to cheat. Depending on the way players attempt to acquire information about the input of the other players we distinguish between dishonest players and players who can work in teams (e.g. [3, 5, 6, 12]). The goal in this approach is to investigate the number of dishonest players or players in a team that are necessary to learn anything about the input of the remaining players. Chor and Kushilevitz [9] have shown that Boolean functions with one bit

output can be computed with teams either of size at most $\lfloor \frac{n-1}{2} \rfloor$ or of any size up to n. For extensions, see Chor, Geréb-Graus, and Kushilevitz [7, 8].

All papers mentioned above do not restrict the communication capabilities of the players. In other words, they use complete graphs as underlying communication networks. However, most realistic parallel architectures have a restricted connectivity and nodes of bounded degree. Franklin and Yung [13] were the first who studied the role of connectivity in private computations. They have presented a protocol for k-connected bus networks, which simulates communication steps of a private protocol that was originally written for a complete graph. To simulate a single communication step, their protocol uses $O(n)$ additional random bits.

1.2 Our Results

In this paper we investigate the number of random bits needed to compute functions by private protocols on k-connected networks. We present a new simulation technique that allows us to reduce the number of additional random bits by taking the connectivity of the network into account (Section 3). Our technique simulates any oblivious private protocol on any given network on an arbitrary k-connected network ($k \geq 2$) with only a small number of additional random bits.

In Section 4, we study the parity function to a greater extent. For every k-connected graph with $k \geq 2$, we design a private protocol for computing the parity function that uses only $\lceil \frac{n-2}{k-1} \rceil - 1$ random bits. This considerably reduces the number of random bits compared to the general simulation technique of Section 3 for the specific case of the parity function. This result is tight: There are k-connected graphs on which every private protocol needs that many random bits to compute the parity function.

All of the above results hold for $k \geq 2$. In Section 5, we investigate graphs that are not 2-connected. Our first insight is the following: The parity function over $n > 2$ bits cannot be computed by a private protocol on any network that is not 2-connected. This can be generalised to a large class of non-degenerate functions. We call a function $f : \{0,1\}^n \to \{0,1\}$ *non-degenerate* if for every $1 \leq i \leq n$, there are bit strings x and y of length n that differ only in the ith bit such that $f(x) \neq f(y)$. In other words, a non-degenerate function depends on all of its input bits. It turns out that there are functions that can be computed by private protocols, even if the underlying network is not 2-connected. An example is the following non-degenerate function $f : \{0,1\}^{2n+1} \to \{0,1\}$ (for $n \geq 2$):

$$f(z,x,y) := (z \wedge \textstyle\bigwedge_{i=1}^{n} x[i]) \vee (\overline{z} \wedge \bigwedge_{i=1}^{n} y[i]) \ .$$

Here, z is a single bit and both x and y are bit strings of length n; $x[i]$ and $y[i]$ denote the ith bit of x and y, respectively. We construct a communication network G for f as follows: Let G_x and G_y be complete networks with n players each. Then connect another player P_z with all players in both G_x and G_y. The network obtained is not 2-connected. Using a slight modification of the protocol presented by Kushilevitz et

al. [19], one can compute the subfunctions

$$\begin{aligned} f_x(z,x) &:= z \wedge \textstyle\bigwedge_{i=1}^{n} x[i] \text{ and} \\ f_y(z,y) &:= \overline{z} \wedge \textstyle\bigwedge_{i=1}^{n} y[i] \end{aligned}$$

by a private protocol on the networks G_x with P_z and G_y with P_z, respectively. After the computation has been completed, P_z is the only player that knows the results of both subfunctions. Due to symmetry we only consider the case $z = 1$. Then $f_y(z, y) = 0$ and therefore, since f_y has been computed by a private protocol, P_z does not learn anything about y. Furthermore, P_z does not learn anything about x except what he is able to deduce from $f_x(z, x)$.

We fully characterise the class of non-degenerate functions that can be computed on non-2-connected networks. It turns out that the above example is fairly representative: Each such function has this `if-then-else` structure. The corresponding non-2-connected network consists of two 2-connected components of appropriate sizes. This characterisation can be generalised to the case where the players can work in teams.

2 Preliminaries

2.1 Notations

For $i, j \in \mathbb{N}$ with $i \leq j$ define $[i] := \{1, \ldots, i\}$ and $[i..j] := \{i, i+1, \ldots, j\}$. Throughout this paper, we will often use the following string operations. Let $x = x[1]x[2]\ldots x[n] \in \{0,1\}^n$ be a string of length n. Then for $I \subseteq [n]$ and $\alpha \in \{0,1\}^{|I|}$, $x\lceil_{I \leftarrow \alpha}$ is defined by

$$z = x\lceil_{I\leftarrow\alpha} :\Longleftrightarrow \forall i \in [n] : z[i] = \begin{cases} x[i] & \text{if } i \notin I \\ \alpha[j] & \text{if } i \in I \text{ and } i \text{ is the} \\ & j\text{th smallest element in } I\,. \end{cases}$$

In other words, we substitute the bits of x by the bits of α at the positions specified by I. For sets $I_1, I_2, \ldots, I_k \subseteq [n]$ and strings $\alpha_1, \alpha_2, \ldots, \alpha_k \in \{0,1\}^*$ with $|\alpha_i| = |I_i|$ we define

$$x\lceil_{I_1,I_2,\ldots,I_k\leftarrow\alpha_1,\alpha_2,\ldots,\alpha_k} := (x\lceil_{I_1\leftarrow\alpha_1})\lceil_{I_2,\ldots,I_k\leftarrow\alpha_2,\ldots,\alpha_k}\,.$$

Let $\overline{x}$ denote the bitwise negation of x, i.e. $\forall i \in [n] : \overline{x}[i] = \overline{x[i]}$. For a function $f : \{0,1\}^n \to \{0,1\}$, a set of indices $I \subseteq [n]$, and a string $\alpha \in \{0,1\}^{|I|}$ define the partially restricted function $f\lceil_{I\leftarrow\alpha} : \{0,1\}^{n-|I|} \to \{0,1\}$ by assigning the values given by α to the positions in I, i.e.

$$\forall x \in \{0,1\}^{n-|I|} : f\lceil_{I\leftarrow\alpha}(x) := f(0^n\lceil_{I,J\leftarrow\alpha,x})\,,$$

where $J = [n] \setminus I$. Finally, for a string $x \in \{0,1\}^n$ and a set $I \subseteq [n]$ define $x[I] \in \{0,1\}^{|I|}$ by

$$\forall j \leq |I| \ : \ (x[I])[j] = x[i] \ :\Longleftrightarrow \ i \text{ is the } j\text{th smallest element in } I \ .$$

$x[I]$ is the substring obtained by deleting from x all bits at positions not in I.

A graph G is called k-connected if, after deleting an arbitrary subset of at most $k-1$ nodes, the resulting node-induced graph remains connected. Equivalently, for any two nodes u and v of G, there are at least k pairwise internally node-disjoint paths between u and v. In particular, if G is k-connected, then each node has degree at least k.

2.2 Private Computation

We consider the computation of Boolean functions $f : \{0,1\}^n \to \{0,1\}$ on a network of n players. In the beginning each player knows a single bit of the input x. The players can send messages to other players via point-to-point communication using secure links where the link topology is given by an undirected graph $G = (V, E)$. When the computation stops, all players know the value $f(x)$. The goal is to compute $f(x)$ such that no player learns anything about the other input bits in an information theoretic sense except for the information he can deduce from his own bit and the result. Such a protocol is called private.

Definition 1 *Let C_i be a random variable of the communication string seen by player P_i, and let c_i be a particular string seen by P_i. A protocol $\mathcal{A}$ for computing a function f is* private with respect to player P_i *if for every pair of input vectors x and y with $f(x) = f(y)$ and $x[i] = y[i]$, for every c_i, and for every random string R_i provided to P_i,*

$$\Pr[C_i = c_i \mid R_i, x] \ = \ \Pr[C_i = c_i \mid R_i, y] \ ,$$

where the probability is taken over the random strings of all other players. A protocol $\mathcal{A}$ is private *if it is private with respect to every player.*

We call a protocol *synchronous* if the communication takes place in rounds and each message consists of a single bit. We call a synchronous protocol *oblivious* if the number of bits (which is either zero or one) that player P_i sends to P_j in round t depends only on i, j, and t but not on the input and the random strings. Furthermore, we do not bound the computational resources of the players. We assume that all of them are honest, i.e. the computation and the interactions between players are determined only by the protocol.

For an oblivious protocol $\mathcal{A}$ let $L(P_i, P_j, \mathcal{A})$ be the total number of bits sent from P_i to P_j in $\mathcal{A}$ and

$$L(\mathcal{A}) \ := \sum_{i \in [n]} \ \sum_{j \in [n] \setminus \{i\}} L(P_i, P_j, \mathcal{A}) \ .$$

We distribute the input bits among the nodes of the graph. For convenience, we call the node that gets bit $x[i]$ player P_i. The players P_i and P_j can communicate directly with each other if and only if they are connected by an edge in the graph.

3 Private Computation on k-connected Networks

Most known private protocols are written for specific networks. A simulation of such a private protocol on a different network can be done in such a way that each player of the new network simulates a player of the original network step-by-step. Hence, we have to find a way to realize the communication steps between all players that are not directly connected. Franklin and Yung [13] have presented a strategy to simulate a transmission of one single bit on a hypergraph by using $O(n)$ additional random bits. Thus, the whole simulation presented by them requires $O(m + nL(\mathcal{A}))$ random bits where m is the number of random bits used by the original protocol. If we consider 2-connected graphs we can simulate each communication step between two players P_i and P_j by one additional random bit r as follows: Assume P_i has to send bit b to P_j. Then P_i chooses two disjoint paths to P_j and sends r to P_j along the first path and $r \oplus b$, the parity of r and b, along the second path. In this way, $O(m + L(\mathcal{A}))$ random bits are sufficient.

To reduce the number of random bits even further, we consider the following optimisation problem:

Definition 2 (Max-Neighbour-Embedding) *Let $G = (V, E)$ be a graph with edge weights $\sigma : E \to \mathbb{N}$ and $G' = (V', E')$ a graph with $|V| = |V'|$. Let $\pi : V \to V'$ be a bijective mapping. Then the performance of π is defined as*

$$\rho(\pi) \ := \sum_{\substack{\{u,v\} \in E \text{ and} \\ \{\pi(u),\pi(v)\} \in E'}} \sigma(\{u,v\}) \, .$$

The aim is to find a bijection $\pi : V \to V'$ that maximises $\rho(\pi)$ over all bijections.

By reduction from 3-Dimensional-Matching [15, SP1], it can be shown that the decision problem corresponding to Max-Neighbour-Embedding is $\mathcal{NP}$-hard, even if both graphs have maximum degree 4.

In the following lemma we estimate the performance for the case that G' is k-connected.

Lemma 1 *Let $G = (V, E)$ be a graph with n nodes and edge weights σ. Let $G' = (V', E')$ be a k-connected graph with n nodes. Then we have*

$$\max_{\substack{\pi : V \to V' \\ \pi \text{ is bijective}}} \rho(\pi) \ \geq \ \frac{k}{n-1} \sum_{e \in E} \sigma(e) \, .$$

Proof: By the definition above, there is no difference between edges with weight 0 and nonexistent edges. Therefore, we treat nonexistent edges like edges with weight 0 and restrict ourselves to the case that G is a complete graph. The graph G' is k-connected. Thus, every node in V' has degree at least k.

Let Π be a random bijection from V to V'. Since every node in V' has degree at least k, the probability that two arbitrary nodes u and v are neighbours under Π, i.e. $\{\Pi(u), \Pi(v)\} \in E'$, is at least $\frac{k}{n-1}$. Thus, the edge $e = \{u, v\} \in E$ yields weight $\sigma(e)$ with probability at least $\frac{k}{n-1}$ and its expected weight is at least $\frac{k}{n-1} \cdot \sigma(e)$. Hence, the expected performance $\rho(\Pi)$ fulfils

$$\mathbb{E}(\rho(\Pi)) \geq \sum_{e \in E} \frac{k}{n-1} \cdot \sigma(e) = \frac{k}{n-1} \cdot \sum_{e \in E} \sigma(e) \, .$$

Thus, there exists a bijection with performance at least $\frac{k}{n-1} \cdot \sum_{e \in E} \sigma(e)$. □

A bijection that fulfils the requirements of the above lemma can be computed in polynomial time using the method of conditional expectation (see e.g. Alon et al. [1]).

Theorem 1 *Every oblivious private protocol $\mathcal{A}$ using m random bits can be simulated with $m + (1 - \frac{k}{n-1}) \cdot \min\{L(\mathcal{A}), \frac{k-2}{k-1} \cdot (n^2 - n) + \frac{L(\mathcal{A})}{k-1}\}$ random bits on every k-connected graph.*

Proof: Let $G = (V, E)$ be the network used in protocol $\mathcal{A}$ and $G' = (V', E')$ be the k-connected network for protocol $\mathcal{A}'$. To simulate $\mathcal{A}$ we first choose a bijection between the players in G and the players in G'. For every edge $\{P_i, P_j\} \in E$ define $\sigma(\{P_i, P_j\}) := L(P_i, P_j, \mathcal{A}) + L(P_j, P_i, \mathcal{A})$. In Lemma 1 we have seen that there exists a bijection $\pi : V \to V'$ with performance $\rho(\pi) \geq \frac{k}{n-1} L(\mathcal{A})$. Using this bijection, at least $\frac{k}{n-1} \cdot L(\mathcal{A})$ bits of the total communication in $\mathcal{A}$ are sent between players that are also neighbours in G'. Thus, this part of the communication can be simulated directly without additional random bits.

For the remaining $(1 - \frac{k}{n-1}) \cdot L(\mathcal{A})$ bits we proceed as follows: Let P_i and P_j be two players that are not directly connected in G'. Then P_i partitions the bits he will send to P_j into blocks $B_1, \ldots, B_{\lceil L(P_i, P_j, \mathcal{A})/(k-1) \rceil}$ of size at most $k - 1$. Furthermore, P_i chooses k node-disjoint paths from P_i to P_j. P_i uses a separate random bit $r[\ell]$ for each block B_ℓ. He sends $r[\ell]$ along the first path and $b \oplus r[\ell]$ for each $b \in B_\ell$ along the remaining paths, each bit on a separate path.

$\sum_{i \in [n], j \in [n] \setminus \{i\}} \lceil L(P_i, P_j, \mathcal{A})/(k-1) \rceil \leq \frac{k-2}{k-1} \cdot (n^2 - n) + \frac{L(\mathcal{A})}{k-1}$ holds, because we round at most $n^2 - n$ fractions with denominator $k - 1$. (This is a worst-case estimate. Given a concrete protocol, additional knowledge about the distribution of the bits on the links may be used to get a better bound.) But we surely never need more than $(1 - \frac{k}{n-1}) \cdot L(\mathcal{A})$ bits altogether. Both observations together imply the bound proposed. □

4 Computing Parity on k-connected Networks

It is well known that the parity function of n bits can be computed on a cycle by using only one random bit. On the other hand, using our simulation discussed in Section 3 we get an upper bound of $n-1$ random bits for arbitrary 2-connected networks. The aim of this section is to close this gap. We present a private protocol for parity that uses $\lceil \frac{n-2}{k-1} \rceil - 1$ random bits and show that there are k-connected networks on which parity cannot be computed with less than $\lceil \frac{n-2}{k-1} \rceil - 1$ random bits.

Lemma 2 *There exist k-connected networks with $n \geq 2k$ players on which the parity function cannot be computed by a private protocol with less than $\lceil \frac{n-2}{k-1} \rceil - 1$ random bits.*

Proof: We consider the bipartite graph $K_{k,n-k}$ (which is k-connected) and show that every private protocol that computes the parity function on this network needs at least $\lceil \frac{n-2}{k-1} \rceil - 1$ random bits. Let $\{P_1, P_2, \ldots, P_k\}$ and $\{P_{k+1}, P_{k+2}, \ldots, P_n\}$ be the two sets of nodes of $K_{k,n-k}$. For every $i = 1, \ldots, k$ and $j = k+1, \ldots, n$ we have an edge $\{P_i, P_j\}$ in $K_{k,n-k}$. Now assume to the contrary that there exists a private protocol $\mathcal{A}$ on $K_{k,n-k}$ using less than $\lceil \frac{n-2}{k-1} \rceil - 1$ random bits.

Let $R = \langle R_1, \ldots, R_n \rangle$ be the contents $R_1, \ldots, R_n$ of all random tapes. For a string $x \in \{0,1\}^n$ and $i \in [n]$, let $\mathcal{C}_i(x, R)$ be a full description of the communication received by P_i during the computation of $\mathcal{A}$ with random bits R on input x. Moreover, let

$$\mathcal{C}(x) = \{\langle c_1, c_2, \ldots, c_k \rangle \mid \exists R \, \forall i \in [k] \;\; c_i = \mathcal{C}_i(x, R)\} \, .$$

We consider computations of $\mathcal{A}$ on inputs

$$X = \{x \mid x[1] = x[2] = \ldots = x[k] = 0 \;\text{ and }\; \textstyle\bigoplus_{i=1}^{n} x[i] = 0\} \, .$$

For every $x \in X$ and every communication c_1 we define

$$\mathcal{C}(c_1, x) = \{\langle c_2, \ldots, c_k \rangle \mid \langle c_1, c_2, \ldots, c_k \rangle \in \mathcal{C}(x)\} \, .$$

From the fact that $\mathcal{A}$ is private it follows:

Claim: $\exists c_1 \, \forall x \in X \quad \mathcal{C}(c_1, x) \neq \emptyset$.

Let $x \in X$. Because x is a valid input for the protocol $\mathcal{A}$, there exists at least one tuple $\langle c_1, \ldots, c_k \rangle$ in $\mathcal{C}(x)$. Hence, there exists at least one c_1 with $\mathcal{C}(c_1, x) \neq \emptyset$. If for some $y \in X$ the set $\mathcal{C}(c_1, y)$ is empty, then this contradicts the definition of private protocols — the claim is proved.

Note that $|X| = 2^{n-k-1}$ and that we have $\bigcup_R \mathcal{C}_i(x, R) = \bigcup_R \mathcal{C}_i(y, R)$ for all $x, y \in X$ and $i \in [k]$. Furthermore, from a bound on the number of different communication strings from Kushilevitz and Rosén [20] it follows that $|\bigcup_R \mathcal{C}_i(x, R)| < 2^{\frac{n-k-1}{k-1}}$ because $\mathcal{A}$ uses less than $\frac{n-k-1}{k-1}$ random bits. Hence, we have $|\bigcup_{x \in X} \mathcal{C}(c_1, x)| < 2^{n-k-1}$. Therefore, by the pigeon hole principle and the previous claim we obtain

$$\exists c_1, c_2, \ldots, c_k \, \exists x, y \in X : \quad x \neq y \;\text{ and }\; \langle c_2, \ldots, c_k \rangle \in \mathcal{C}(c_1, x) \cap \mathcal{C}(c_1, y) \, .$$

This means that there are two different input string $x, y \in X$ such that on either string the players $P_1, \ldots, P_k$ receive $c_1, \ldots, c_k$, respectively. Let i be a position where x and y differ, i.e. $x[i] \neq y[i]$. Note that $k+1 \leq i \leq n$. Let $R = \langle R_1, \ldots, R_n \rangle$ and $R' = \langle R'_1, \ldots, R'_n \rangle$ be the contents of the random tapes such that $c_i = \mathcal{C}_i(x, R) = \mathcal{C}_i(y, R')$ for all $1 \leq i \leq k$.

During a computation of protocol $\mathcal{A}$ on input $x^{\lceil_{\{i\} \leftarrow y[i]}}$ with random strings $R'' = \langle R_1, \ldots, R_{i-1}, R'_i, R_{i+1}, \ldots R_n \rangle$ the players $P_1, P_2, \ldots, P_k$ again receive the communication strings $c_1, c_2, \ldots, c_k$. This is because the graph is bipartite and P_i can only communicate with $P_1, \ldots, P_k$. Hence, for this input they compute the same result as for x — a contradiction. □

Now we show that this bound is best possible. To obtain a private protocol that computes the parity function with $\lceil \frac{n-2}{k-1} \rceil - 1$ random bits, we use the result from Egawa, Glas, and Locke [11] that every k-connected graph G with minimum degree at least d and with at least $2d$ nodes has a cycle of length at least $2d$ through any set of k nodes. From this result we get the following lemmas.

Lemma 3 *Let $G = (V, E)$ be a k-connected graph with $|V| \geq 2k$. Then for every subset $V' \subseteq V$ with $|V'| = k$, there exists a simple cycle of length at least $2k$ containing all nodes in V'.*

Proof: Since G is k-connected, every node has degree at least k. Thus, G contains a simple cycle of length $2k$ running through all nodes in V' by [11, Thm. 3]. □

Lemma 4 *Let $G = (V, E)$ be a k-connected graph with $|V| \geq 2k$. Then for every subset $V' \subseteq V$ with $|V'| = k+1$, there exists a simple path containing all nodes in V'.*

Proof: By Lemma 3, G contains a cycle C running through k of the nodes in V'. If the last node v of V' is also on C, we simply delete one edge of C and are done. Otherwise, since G is connected there is path from v to a node u of C, such that each internal node of this path is not in C. By deleting one edge of C incident with u, we obtain the desired path. □

Lemma 5 *Let $G = (V, E)$ be a k-connected graph with $|V| \geq 2k+1$. Then G has a simple path with at least $2k+1$ nodes.*

Proof: By Lemma 3, G has a cycle of length at least $2k$. If this length is strictly greater than $2k$, we delete one of its edges and are done. Otherwise, there is a node v not in the cycle. We now construct a path as in the proof of Lemma 4. □

To compute the parity function by a private protocol on an arbitrary k-connected network G, we proceed as follows. We first assume that G has at least $2k+1$ nodes.

1. Mark all nodes in G red. Set $z[i] := x[i]$ for each player P_i.

2. Choose a path in G of length $2k+1$. According to Lemma 5 such a path always exists. The first player P_i in the path generates a random bit r. Then P_i computes $r \oplus z[i]$, sends the result to the next player in the path, and sets $z[i] := r$.

   Each internal player P_j on the path receives a bit b from its predecessor in the path, computes $b \oplus z[j]$, sends this bit to its successor, and changes its colour to black.

   The last player P_ℓ on the path receives a bit b from its predecessor and computes $z[\ell] := z[\ell] \oplus b$.

   After this step, $2k-1$ players have changed their colour.

3. We repeat the following step $\lceil \frac{n-3k+1}{k-1} \rceil$ times.

   Choose $k+1$ red nodes and a path in G containing all these nodes. According to Lemma 4 such a path always exists. We can assume that the start and the end node of the path are among the $k+1$ given players, hence both are red. Then the first player P_i on this path generates a random bit r, computes $r \oplus z[i]$, sends the result to the next player in the path, and sets $z[i] := r$.

   Each internal player of the path P_j receives a bit b from its predecessor in the path. If P_j is a black player, it sends b to its successor. If P_j is a red player, it computes $b \oplus z[j]$, sends this bit to its successor, and changes its colour to black.

   The last player P_ℓ on the path receives a bit b from its predecessor and computes $z[\ell] := z[\ell] \oplus b$.

   After this step, $k-1$ players have changed their colour. Hence, after $\lceil \frac{n-3k+1}{k-1} \rceil$ iterations of this step we have at least

   $$\lceil \tfrac{n-3k+1}{k-1} \rceil \cdot (k-1) + 2k - 1 \;\geq\; n - k$$

   black players. Thus, at most k are red.

4. Choose a cycle in G containing all red nodes. According to Lemma 3 such a cycle always exists. Let P_{i_0} be a red player. Then P_{i_0} generates a random bit r, computes $r \oplus z[i_0]$, and sends the result to the next player in the cycle.

   Each other player P_j on the cycle receives a bit b from its predecessor. If P_j is a black player, it sends b to its successor. If P_j is a red player, it computes $b \oplus z[j]$, sends this bit to its successor, and changes its colour to black.

   If P_{i_0} receives a bit b, he computes $b \oplus r$. The result of this step is the result of the parity function.

Let us count the number of random bits used in the protocol above. In the second and in the last step we use one random bit. In the third step we need $\lceil \frac{n-3k+1}{k-1} \rceil$ random bits. Hence, the total number of random bits is

$$\lceil \tfrac{n-3k+1}{k-1} \rceil + 2 \;=\; \lceil \tfrac{n-2}{k-1} \rceil - 1\,.$$

It remains to show that the protocol is private and computes the parity function. The correctness follows from the fact that each input bit $x[i]$ is stored by exactly one red player and each random bit is stored by either none or two players that are red after each step. By storing a bit b we mean that a player P_i knows a value $z[i]$ that depends on b. Since P_{i_0} is the last red player, he knows the result of the parity function.

Every bit received by some player in the second and third step is masked by a separate random bit. Hence, none of these players can learn anything from these bits. The same holds for all players except for player P_{i_0} in the last step. So we have to analyse the bits sent and received by P_{i_0} more carefully. In the last step $z[i_0]$ is either $x[i_0]$, a random bit, or the parity of a subset of input bits masked by a random bit. In neither case P_{i_0} can learn anything about the other input bits from the bit he receives and the value of $z[i_0]$ except for what can be derived from the result of the function and $x[i_0]$.

Theorem 2 *Let G be an arbitrary k-connected network with n nodes such that $n \geq 2k$. Then the parity function of n bits can be computed by a private protocol on G using at most $\lceil \frac{n-2}{k-1} \rceil - 1$ random bits. There exist k-connected networks for which this bound is best possible.*

Proof: The case $n \geq 2k+1$ has already been demonstrated. If $n = 2k$, then G has a cycle containing all nodes of G by Lemma 3. On a cycle, we can compute parity with only one random bit.

The lower bound follows from Lemma 2. □

For 2-connected networks G with $n \geq 4$ nodes, the previous theorem implies that the parity function can be computed using at most $n-3$ random bits. There are such networks for which this is sharp.

5 Private Computation on Non-2-connected Networks

Throughout this section, $f : \{0,1\}^n \to \{0,1\}$ denotes the function we want to compute, where $n \geq 3$. We assume that f is non-degenerate (as defined in Section 1.2). Furthermore, I_1, I_2, J_1, and J_2 denote both subsets of input positions and sets of indices of players.

We say that a pair (J_1, J_2) of two disjoint subsets $J_1, J_2 \subseteq [n]$ has the *flip-property* if there exist an input $x \in \{0,1\}^n$ and two strings $\alpha \in \{0,1\}^{|J_1|}$ and $\beta \in \{0,1\}^{|J_2|}$ with

$$f(x\lceil_{J_1,J_2 \leftarrow \alpha,\beta}) \neq f(x\lceil_{J_1,J_2 \leftarrow \overline{\alpha},\beta}) = f(x\lceil_{J_1,J_2 \leftarrow \alpha,\overline{\beta}}) .$$

We call the strings α and β *flip-witnesses* for (J_1, J_2).

Lemma 6 *If a function $f : \{0,1\}^n \to \{0,1\}$ is non-degenerate, then for every partition I_1, I_2 of $[n]$ and every $i \in I_1$ and $j \in I_2$ we have: There exist subsets $J_1 \subseteq I_1$ and $J_2 \subseteq I_2$ with $i \in J_1, j \in J_2$ such that (J_1, J_2) has the flip-property.*

Loosely speaking, this lemma says that each non-degenerate function behaves on subsets of input positions in some sense like the parity function.

Proof: From the definition of non-degenerate it follows that for every $i \in I_1$ and $j \in I_2$ there exist input strings $y, z \in \{0,1\}^n$ with

$$f(y\lceil_{\{i\}\leftarrow 0}) \neq f(y\lceil_{\{i\}\leftarrow 1}) \quad \text{and} \quad f(z\lceil_{\{j\}\leftarrow 0}) \neq f(z\lceil_{\{j\}\leftarrow 1}) .$$

$f(y\lceil_{\{i\}\leftarrow 0}) \neq f(y\lceil_{\{i\}\leftarrow 1})$ can be rewritten as $f(y\lceil_{\{i\},\{j\}\leftarrow 0,y[j]}) \neq f(y\lceil_{\{i\},\{j\}\leftarrow 1,y[j]})$. If $f(y\lceil_{\{i\},\{j\}\leftarrow 0,y[j]}) = f(y\lceil_{\{i\},\{j\}\leftarrow 1,\overline{y[j]}})$, then $J_1 = \{i\}$, $J_2 = \{j\}$ fulfil the flip-property with witnesses $(1, y[j])$ and we are done. If $f(y\lceil_{\{i\},\{j\}\leftarrow 1,y[j]}) = f(y\lceil_{\{i\},\{j\}\leftarrow 0,\overline{y[j]}})$, then $J_1 = \{i\}$, $J_2 = \{j\}$ fulfil the flip-property with witnesses $(0, y[j])$. If neither case holds, we have

$$f(y\lceil_{\{i\},\{j\}\leftarrow 0,0}) = f(y\lceil_{\{i\},\{j\}\leftarrow 0,1}) \neq f(y\lceil_{\{i\},\{j\}\leftarrow 1,0}) = f(y\lceil_{\{i\},\{j\}\leftarrow 1,1}) . \quad (1)$$

Analogously, $J_1 = \{j\}$, $J_2 = \{i\}$ fulfil the flip-property, if $f(z\lceil_{\{i\},\{j\}\leftarrow z[i],0}) = f(z\lceil_{\{i\},\{j\}\leftarrow \overline{z[i]},1})$ or $f(z\lceil_{\{i\},\{j\}\leftarrow z[i],1}) = f(z\lceil_{\{i\},\{j\}\leftarrow \overline{z[i]},0})$. If neither case holds, we also have

$$f(z\lceil_{\{i\},\{j\}\leftarrow 0,0}) = f(z\lceil_{\{i\},\{j\}\leftarrow 1,0}) \neq f(z\lceil_{\{i\},\{j\}\leftarrow 0,1}) = f(z\lceil_{\{i\},\{j\}\leftarrow 1,1}) . \quad (2)$$

Next we construct two sets with the flip-property from y and z. By (1) and (2), we may assume that $y[i] \neq z[i]$ and $y[j] \neq z[j]$. Otherwise, we can flip $y[j]$ or $z[i]$ since $f(y)$ and $f(z)$ do not depend on $y[j]$ and $z[i]$, respectively. Analogously, we can assume that $f(y) \neq f(z)$. Otherwise, we can flip the bits $y[j]$ and $z[j]$ simultaneously. Since $f(y)$ does not depend on $y[j]$, we have $f(y) \neq f(z)$ afterwards. Define

$$Y_1 := \{\, k \in I_1 \mid y[k] \neq z[k] \,\} \quad \text{and} \quad Y_2 := \{\, k \in I_2 \mid y[k] \neq z[k] \,\} .$$

Let $Y_1 = \{i_1, \ldots, i_{|Y_1|}\}$ with $i_1 < i_2 < \ldots < i_{|Y_1|}$ and $Y_2 = \{j_1, \ldots, j_{|Y_2|}\}$ with $j_1 < j_2 < \ldots < j_{|Y_2|}$. By construction, $i \in Y_1$ and $j \in Y_2$. Define $\rho \in \{0,1\}^{|Y_1|}$ and $\sigma \in \{0,1\}^{|Y_2|}$ such that

$$\forall \ell \in [1, |Y_1|] \;:\; \rho[\ell] := y[i_\ell] \qquad \text{and} \qquad \forall \ell \in [1, |Y_2|] \;:\; \sigma[\ell] := y[i_\ell] .$$

Note that

$$y\lceil_{Y_2\leftarrow \overline{\sigma}} = z\lceil_{Y_1\leftarrow \rho} . \quad (3)$$

To prove the lemma, we distinguish the cases $f(y) \neq f(y\lceil_{Y_2\leftarrow\overline{\sigma}}) = f(z)$ and $f(y) = f(y\lceil_{Y_2\leftarrow\overline{\sigma}}) \neq f(z)$.

1. If $f(y) \neq f(y\lceil_{Y_2\leftarrow\overline{\sigma}}) = f(z)$, we choose

   $$\alpha := y[i], \quad \beta := \sigma, \quad J_1 := \{i\}, \quad J_2 := Y_2, \quad x := y .$$

   We have

   $$f(x\lceil_{J_1,J_2\leftarrow\alpha,\beta}) = f(y) \neq f(y\lceil_{\{i\}\leftarrow\overline{y[i]}}) = f(x\lceil_{J_1,J_2\leftarrow\overline{\alpha},\beta})$$

by the choice of y and i. Furthermore,

$$f(x\lceil_{J_1,J_2\leftarrow\alpha,\beta}) = f(y) \neq f(y\lceil_{Y_2\leftarrow\overline{\sigma}}) = f(x\lceil_{J_1,J_2\leftarrow\alpha,\overline{\beta}})$$

by assumption. Thus, (J_1, J_2) fulfils the flip-property.

2. If $f(y) = f(y\lceil_{Y_2\leftarrow\overline{\sigma}}) \neq f(z)$, then $f(y) = f(z\lceil_{Y_1\leftarrow\rho}) \neq f(z)$ by (3). We set

$$\alpha := \overline{\rho}, \quad \beta := z[j], \quad J_1 := Y_1, \quad J_2 := \{j\}, \quad x := z\,.$$

We have

$$f(x\lceil_{J_1,J_2\leftarrow\alpha,\beta}) = f(z) \neq f(z\lceil_{\{j\}\leftarrow\overline{z[j]}}) = f(x\lceil_{J_1,J_2\leftarrow\alpha,\overline{\beta}})$$

by the choice of z and j. Furthermore,

$$f(x\lceil_{J_1,J_2\leftarrow\alpha,\beta}) = f(z) \neq f(z\lceil_{Y_1\leftarrow\rho}) = f(x\lceil_{J_1,J_2\leftarrow\overline{\alpha},\beta})$$

by assumption. Thus, (J_1, J_2) fulfils the flip-property.

Since this case distinction is exhaustive, we can always find subsets $J_1 \subseteq I_1$ and $J_2 \subseteq I_2$ fulfilling the claim of the lemma. □

A set I_1 with $\emptyset \neq I_1 \neq [n]$ is *dominated* by an input position $k \in I_1$ if the following holds: For each pair of subsets $J_1 \subseteq I_1$ and $J_2 \subseteq [n] \setminus I_1$ that fulfils the flip-property, we have $k \in J_1$. A function is *ℓ-dominated* if there exists a set $I_1 \subseteq [n]$ of size ℓ that is dominated by some $k \in I_1$. A function f is called *dominated* if there exists an $\ell > 1$ such that f is ℓ-dominated. Otherwise, f is called *non-dominated*.

Theorem 3 *Let f be a non-degenerate function and G be a network that can be separated into two networks G_1 and G_2 of size n_1 and n_2, respectively, by removing one bridge node from G. If f can be computed by a private protocol on G, then f is $(n_1 + 1)$- and $(n_2 + 1)$-dominated.*

Theorem 3 follows from Lemma 6 and the next Lemma 7 as follows: Assume on the contrary that f is not $(n_1 + 1)$-dominated. The other case is treated symmetrically. Let I_1 and I_2 be the sets of indices corresponding to players in G_1 and G_2, respectively, and let P_k be the bridge player connecting G_1 and G_2. By Lemma 6, there are $J_1 \subseteq I_1 \cup \{k\}$ and $J_2 \subseteq I_2$ such that (J_1, J_2) has the flip-property, because f is non-degenerate. Since f is not $(n_1 + 1)$-dominated, there are also subsets $J_1' \subseteq I_1 \cup \{k\}$ and $J_2' \subseteq I_2$ such that (J_1', J_2') has the flip-property and $k \notin J_1'$. Lemma 7 implies (setting $I_1 = J_1'$ and $I_2 = J_2'$ in the claim of the lemma) that f cannot be computed by a private protocol — a contradiction. This proves Theorem 3.

Lemma 7 *Let G be a network with n nodes. Assume that there exist $I_1, I_2 \subseteq [n]$ and $k \in [n]$, such that the following conditions hold:*

1. $I_1, I_2 \neq \emptyset$ *and* $k \notin I_1 \cup I_2$, $I_1 \cap I_2 = \emptyset$,

2. *for every path* $W_{i,j}$ *from* P_i *to* P_j *with* $i \in I_1$ *and* $j \in I_2$, *we have* $P_k \in W_{i,j}$, *and*

3. (I_1, I_2) *has the flip-property.*

Then f *cannot be computed on* G *by a private protocol.*

Proof: Assume that there exists such a protocol. Let M_i^t be a message sent by player P_i in round t and $T(\mathcal{A})$ be the maximum number of rounds of $\mathcal{A}$ for all inputs of length n and all random tapes. M_i^t is a function of the input string z and the random tapes R. Player P_i receives in round $t \leq T(\mathcal{A})$ the messages

$$C_i^t(z, R) := M_{i_1}^t(z, R), \ldots, M_{i_s}^t(z, R) \,,$$

where $P_{i_1}, \ldots, P_{i_s}$ are all the players incident to player P_i. We denote the sequence $C_i^1(z, R), C_i^2(z, R), \ldots, C_i^{T(\mathcal{A})}(z, R)$ by $C_i(z, R)$.

Now let k, I_1, I_2 fulfil the three conditions of the lemma and choose x, α, and β such that

$$f(x\lceil_{I_1,I_2 \leftarrow \alpha,\beta}) \neq f(x\lceil_{I_1,I_2 \leftarrow \alpha,\overline{\beta}}) = f(x\lceil_{I_1,I_2 \leftarrow \overline{\alpha},\beta}) \,.$$

Keep R fixed. Then consider $C_k(x\lceil_{I_1,I_2 \leftarrow \alpha,\overline{\beta}}, R)$, which is the sequence of messages received by P_k during the computation on $x\lceil_{I_1,I_2 \leftarrow \alpha,\overline{\beta}}$ with random bits R. Since the protocol is private and $k \notin I_1 \cup I_2$, there exists $R' = \langle R'_1, \ldots, R'_n \rangle$, with $R_k = R'_k$, such that

$$C_k(x\lceil_{I_1,I_2 \leftarrow \overline{\alpha},\beta}, R') = C_k(x\lceil_{I_1,I_2 \leftarrow \alpha,\overline{\beta}}, R) \,. \tag{4}$$

Let

$$Y := \{ \ell \mid \text{there is a path } W_{\ell,i} \text{ from } \ell \text{ to a node } i \in I_1 \text{ such that } k \notin W_{\ell,i} \} \,.$$

We have $I_1 \subseteq Y$ and $I_2 \cap Y = \emptyset$. Now let $R'' = \langle R''_1, \ldots, R''_n \rangle$ be a content of the random tapes defined as follows: for every $\ell \in Y$ let $R''_\ell := R_\ell$ and for every $j \in [n] \setminus Y$ let $R''_j := R'_j$. Note that $R''_k = R'_k = R_k$. From (4), it follows that on input $x\lceil_{I_1,I_2 \leftarrow \alpha,\beta}$ and with random tapes R'' the protocol generates the following messages for every player $i \in [n]$ and every $t \geq 1$:

$$M_i^t(x\lceil_{I_1,I_2 \leftarrow \alpha,\beta}, R'') = \begin{cases} M_i^t(x\lceil_{I_1,I_2 \leftarrow \alpha,\overline{\beta}}, R) & \text{if } i \in Y \,, \\ M_i^t(x\lceil_{I_1,I_2 \leftarrow \overline{\alpha},\beta}, R') & \text{if } i \in [n] \setminus Y \,. \end{cases}$$

Hence, given the input string $x\lceil_{I_1,I_2 \leftarrow \alpha,\beta}$ the protocol computes the same value as on the input string $x\lceil_{I_1,I_2 \leftarrow \overline{\alpha},\beta}$ and $x\lceil_{I_1,I_2 \leftarrow \alpha,\overline{\beta}}$ — a contradiction. □

Corollary 1 *A non-dominated non-degenerate function cannot be computed by a private protocol on a network that is not 2-connected.*

Examples of non-dominated non-degenerate functions are parity, majority, conjunction, and disjunction. Hence, these functions cannot be computed by private protocols on networks that are not 2-connected.

In the remainder of this section, we show that for every ℓ-dominated function f with $n-1 > \ell > 2$, there is a non-2-connected network on which f can be computed by a private protocol. For a subset I_1 of input positions define the *flip-witness-set* for I_1 by

$$\begin{aligned}\text{f-set}(I_1) \;:=\; \{(\alpha, J_1) \mid\;& J_1 \subseteq I_1, \alpha \in \{0,1\}^{|J_1|} \\ & \text{and there exist } J_2 \subseteq [n] \setminus I_1, \beta \in \{0,1\}^{|J_2|} \\ & \text{such that } \alpha, \beta \text{ are flip-witnesses for } J_1, J_2\}\,.\end{aligned}$$

We start with some technical lemmas.

Lemma 8 *Let $I_{1,1}, I_{2,1} \subseteq [n]$ be nonempty and disjoint and let $k \notin I_{1,1} \cup I_{2,1}$. If there are strings $w_1, w_2 \in \{0,1\}^n$ with $w_1[k] = w_2[k]$, $\alpha_1 \in \{0,1\}^{|I_{1,1}|}$, and $\beta_1 \in \{0,1\}^{|I_{2,1}|}$ such that*

$$f(w_1\lceil_{I_{1,1}\leftarrow\alpha_1}) \;=\; f(w_2\lceil_{I_{2,1}\leftarrow\beta_1}) \;\neq\; f(w_1\lceil_{I_{1,1}\leftarrow\overline{\alpha}_1}) \;=\; f(w_2\lceil_{I_{2,1}\leftarrow\overline{\beta}_1})\,,$$

then every set $I_1 \supseteq I_{1,1}$ with $I_1 \cap I_{2,1} = \emptyset$ is not dominated by k.

Proof: Let $I_1 \supseteq I_{1,1}$ with $I_1 \cap I_{2,1} = \emptyset$ be given. If $k \notin I_1$, the claim is trivial. Therefore, assume $k \in I_1$. Define $\gamma, \delta \in \{0,1\}^*$ by

$$\begin{aligned} I_{1,2} &:= \{i \in I_1 \mid w_1[i] \neq w_2[i]\}\,, \\ \gamma &:= w_2[I_{1,2}]\,, \\ I_{2,2} &:= \{i \in [n] \setminus I_1 \mid w_1[i] \neq w_2[i]\}\,, \\ \delta &:= w_2[I_{2,2}]\,. \end{aligned}$$

Note that $w_1\lceil_{I_{1,2},I_{2,2}\leftarrow\gamma,\delta} = w_2$.

Consider $w_1\lceil_{I_{1,1},I_{2,2},I_{2,1}\leftarrow\alpha_1,\delta,\overline{\beta}_1}$. The latter substitutions only change values of $w_1\lceil_{I_{1,1}\leftarrow\alpha_1}$ at positions in $[n] \setminus I_1$. We distinguish the following two cases:

1. If $f(w_1\lceil_{I_{1,1},I_{2,2},I_{2,1}\leftarrow\alpha_1,\delta,\overline{\beta}_1}) \;=\; f(w_1\lceil_{I_{1,1}\leftarrow\alpha_1})$, then by the assumption of the lemma, we have

$$f(w_1\lceil_{I_{1,1},I_{2,2},I_{2,1}\leftarrow\alpha_1,\delta,\overline{\beta}_1}) \;=\; f(w_2\lceil_{I_{2,1}\leftarrow\beta_1}) \;\neq\; f(w_2\lceil_{I_{2,1}\leftarrow\overline{\beta}_1})\,. \qquad (5)$$

Define $J_1 := \{i \in I_1 \mid w_1\lceil_{I_{1,1},I_{2,2},I_{2,1}\leftarrow\alpha_1,\delta,\overline{\beta}_1}[i] \neq w_2\lceil_{I_{2,1}\leftarrow\overline{\beta}_1}[i]\ \}$ and $\alpha_2 := w_2\lceil_{I_{2,1}\leftarrow\overline{\beta}_1}[J_1]$. Note that $k \notin J_1$, since $w_1[k] = w_2[k]$ and $k \notin I_{1,1} \cup I_{2,1} \cup I_{2,2}$. Since $I_{1,1} \subseteq I_1$ and $I_{2,1} \cap I_1 = \emptyset$,

$$\begin{aligned} w_2\lceil_{I_{2,1}\leftarrow\overline{\beta}_1} &= w_2\lceil_{J_1,I_{2,1}\leftarrow\alpha_2,\overline{\beta}_1}\,, \\ w_1\lceil_{I_{1,1},I_{2,2},I_{2,1}\leftarrow\alpha_1,\delta,\overline{\beta}_1} &= w_2\lceil_{J_1,I_{2,1}\leftarrow\overline{\alpha}_2,\overline{\beta}_1}\,, \\ w_2\lceil_{I_{2,1}\leftarrow\beta_1} &= w_2\lceil_{J_1,I_{2,1}\leftarrow\alpha_2,\beta_1}\,. \end{aligned}$$

Thus, (5) can be rewritten as

$$f(w_2\lceil_{J_1,I_{2,1}\leftarrow\alpha_2,\beta_1}) = f(w_2\lceil_{J_1,I_{2,1}\leftarrow\overline{\alpha}_2,\overline{\beta}_1}) \neq f(w_2\lceil_{J_1,I_{2,1}\leftarrow\alpha_2,\overline{\beta}_1}) .$$

Therefore, $(\alpha_2, J_1) \in$ f-set(I_1). But $k \notin J_1$. Thus, I_1 is not dominated by k.

2. If $f(w_1\lceil_{I_{1,1},I_{2,2},I_{2,1}\leftarrow\alpha_1,\delta,\overline{\beta}_1}) \neq f(w_1\lceil_{I_{1,1}\leftarrow\alpha_1})$, then

$$f(w_1\lceil_{I_{1,1},I_{2,2},I_{2,1}\leftarrow\alpha_1,\delta,\overline{\beta}_1}) = f(w_1\lceil_{I_{1,1}\leftarrow\overline{\alpha}_1}) \neq f(w_1\lceil_{I_{1,1}\leftarrow\alpha_1}) . \quad (6)$$

Define $J_2 := \{i \in [n] \setminus I_1 \mid w_1\lceil_{I_{1,1},I_{2,2},I_{2,1}\leftarrow\alpha_1,\delta,\overline{\beta}_1}[i] \neq w_1\lceil_{I_{1,1}\leftarrow\alpha_1}[i]\}$ and $\beta_2 := w_1\lceil_{I_{1,1}\leftarrow\alpha_1}[J_2]$. Thus,

$$\begin{aligned} w_1\lceil_{I_{1,1}\leftarrow\alpha_1} &= w_1\lceil_{I_{1,1},J_2\leftarrow\alpha_1,\beta_2} , \\ w_1\lceil_{I_{1,1},I_{2,2},I_{2,1}\leftarrow\alpha_1,\delta,\overline{\beta}_1} &= w_1\lceil_{I_{1,1},J_2\leftarrow\alpha_1,\overline{\beta}_2} , \\ w_1\lceil_{I_{1,1}\leftarrow\overline{\alpha}_1} &= w_1\lceil_{I_{1,1},J_2\leftarrow\overline{\alpha}_1,\beta_2} . \end{aligned}$$

We rewrite (6) as follows:

$$f(w_1\lceil_{I_{1,1},J_2\leftarrow\alpha_1,\overline{\beta}_2}) = f(w_1\lceil_{I_{1,1},J_2\leftarrow\overline{\alpha}_1,\beta_2}) \neq f(w_1\lceil_{I_{1,1},J_2\leftarrow\alpha_1,\beta_2}) .$$

Therefore $(\alpha_1, I_{1,1}) \in$ f-set(I_1). Since $k \notin I_{1,1}$, I_1 is not dominated by k.

□

Lemma 9 *Let $I_1 \subseteq [n]$ with $n > |I_1| \geq 2$, $I_{1,1}, I_{1,2} \subseteq I_1$, and $I_{2,1}, I_{2,2} \subseteq [n] \setminus I_1$, all non-empty. Let $k \in I_1$. If there are $\alpha_1 \in \{0,1\}^{|I_{1,1}|}$, $\beta_1 \in \{0,1\}^{|I_{2,1}|}$, $\alpha_2 \in \{0,1\}^{|I_{1,2}|}$, $\beta_2 \in \{0,1\}^{|I_{2,2}|}$, and u_1, u_2 with $u_1\lceil_{I_{1,1},I_{2,1}\leftarrow\alpha_1,\beta_1}[k] \neq u_2\lceil_{I_{1,2},I_{2,2}\leftarrow\alpha_2,\beta_2}[k]$ fulfilling*

$$f(u_1\lceil_{I_{1,1},I_{2,1}\leftarrow\alpha_1,\beta_1}) \neq f(u_1\lceil_{I_{1,1},I_{2,1}\leftarrow\alpha_1,\overline{\beta_1}})$$

and

$$f(u_2\lceil_{I_{1,2},I_{2,2}\leftarrow\alpha_2,\beta_2}) \neq f(u_2\lceil_{I_{1,2},I_{2,2}\leftarrow\alpha_2,\overline{\beta_2}}) ,$$

then I_1 is not dominated by k.

Proof: Choose an input position $\ell \in I_1 \setminus \{k\}$ and $u_3 \in \{0,1\}^n$ with

$$f(u_3\lceil_{\{\ell\}\leftarrow 0}) \neq f(u_3\lceil_{\{\ell\}\leftarrow 1}) . \quad (7)$$

Such an ℓ exists, since $|I_1| \geq 2$. By assumption, either $u_1\lceil_{I_{1,1},I_{2,1}\leftarrow\alpha_1,\beta_1}[k] = u_3[k]$ or $u_2\lceil_{I_{1,2},I_{2,2}\leftarrow\alpha_2,\beta_2}[k] = u_3[k]$. W.l.o.g. assume that the first equation is true, i.e., $u_1\lceil_{I_{1,1},I_{2,1}\leftarrow\alpha_1,\beta_1}[k] = u_3[k]$. Let $x = u_1\lceil_{I_{1,1}\leftarrow\alpha_1}$. Then by assumption

$$f(x\lceil_{I_{2,1}\leftarrow\beta_1}) \neq f(x\lceil_{I_{2,1}\leftarrow\overline{\beta}_1}) .$$

Let $\gamma \in \{0,1\}$ such that $f(u_3\lceil_{\{\ell\}\leftarrow\gamma}) = f(u_1\lceil_{I_{1,1},I_{2,1}\leftarrow\alpha_1,\beta_1})$. Such a γ exists by (7). Now

$$f(x\lceil_{I_{2,1}\leftarrow\beta_1}) = f(u_3\lceil_{\{\ell\}\leftarrow\gamma}) \neq f(x\lceil_{I_{2,1}\leftarrow\overline{\beta}_1}) = f(u_3\lceil_{\{\ell\}\leftarrow\overline{\gamma}}).$$

By construction, $\ell \notin I_{2,1}$. Therefore, we can apply Lemma 8. Thus, I_1 is not dominated by k. □

Lemma 10 *Assume that a set I_1 with $n > |I_1| \geq 2$ is dominated by an input position $k \in I_1$. Then every pair $(\alpha, J_1) \in$ f-set(I_1) assigns the same value to input position k.*

Proof: The proof is by contradiction. Assume that there exist two pairs $(\alpha_1, I_{1,1})$, $(\alpha_2, I_{1,2}) \in$ f-set(I_1) such that α_1 and α_2 assign different values to input position $k \in I_1$, i.e. $x\lceil_{I_{1,1} \leftarrow \alpha_1}[k] \neq x\lceil_{I_{1,2} \leftarrow \alpha_2}[k]$ for all $x \in \{0,1\}^n$. Furthermore, let $(\beta_1, I_{2,1})$ and $(\beta_2, I_{2,2})$ be counterparts of $(\alpha_1, I_{1,1})$ and $(\alpha_2, I_{1,2})$, respectively, i.e. there are $w_1, w_2 \in \{0,1\}^n$ with

$$f(w_1\lceil_{I_{1,1}, I_{2,1} \leftarrow \alpha_1, \beta_1}) \neq f(w_1\lceil_{I_{1,1}, I_{2,1} \leftarrow \overline{\alpha_1}, \beta_1}) = f(w_1\lceil_{I_{1,1}, I_{2,1} \leftarrow \alpha_1, \overline{\beta_1}})$$

and

$$f(w_2\lceil_{I_{1,2}, I_{2,2} \leftarrow \alpha_2, \beta_2}) \neq f(w_2\lceil_{I_{1,2}, I_{2,2} \leftarrow \overline{\alpha_2}, \beta_2}) = f(w_2\lceil_{I_{1,2}, I_{2,2} \leftarrow \alpha_2, \overline{\beta_2}}) .$$

Lemma 9 implies that I_1 is not dominated by k — a contradiction. □

For $c \in \{0,1\}$, we call a set I_1 (k,c)-*dominated* by input position k, if I_1 is dominated by k and for each pair $(\alpha, J_1) \in$ f-set(I_1), α assigns c to input position k.

Lemma 11 *Assume that a set I_1 with $n > |I_1| \geq 2$ is (k,c)-dominated with $k \in I_1$ for some $c \in \{0,1\}$. Then for every $\alpha \in \{0,1\}^{|I_1|}$ such that α assigns $\bar{c}$ to the kth position, for every $w \in \{0,1\}^n$, $J_2 \subseteq [n] \setminus I_1$, and $\beta \in \{0,1\}^{|J_2|}$ we have*

$$f(w\lceil_{I_1, J_2 \leftarrow \alpha, \beta}) = f(w\lceil_{I_1, J_2 \leftarrow \alpha, \overline{\beta}}) .$$

Proof: Proof by contradiction. Assume that there are a string $\alpha_1 \in \{0,1\}^{|I_1|}$ that does not assign c to input position k as well as $w_1 \in \{0,1\}^n$, $I_{2,1} \subseteq [n] \setminus I_1$, and $\beta_1 \in \{0,1\}^{|I_{2,1}|}$ such that

$$f(w_1\lceil_{I_1, I_{2,1} \leftarrow \alpha_1, \beta_1}) \neq f(w_1\lceil_{I_1, I_{2,1} \leftarrow \alpha_1, \overline{\beta}_1}) .$$

Choose $(\alpha_2, I_{1,2}) \in$ f-set(I_1) and $(\beta_2, I_{2,2})$ as a counterpart of $(\alpha_2, I_{1,2})$, i.e. there exists w_2 with

$$f(w_2\lceil_{I_{1,2}, I_{2,2} \leftarrow \alpha_2, \beta_2}) \neq f(w_2\lceil_{I_{1,2}, I_{2,2} \leftarrow \overline{\alpha}_2, \beta_2}) = f(w_2\lceil_{I_{1,2}, I_{2,2} \leftarrow \alpha_2, \overline{\beta}_2}) .$$

Since I_1 is (k,c)-dominated, we have $w_1\lceil_{I_1, I_{2,1} \leftarrow \alpha_1, \beta_1}[k] \neq c = w_2\lceil_{I_{1,2}, I_{2,2} \leftarrow \alpha_2, \beta_2}[k]$. Lemma 9 implies that I_1 is not dominated by k — a contradiction. □

By the previous lemma, we can conclude that for each set I_1 with $n > |I_1| \geq 2$ that is (k,c)-dominated by an input position $k \in I_1$, there exists a function $f_1 : \{0,1\}^{|I_1|-1} \rightarrow \{0,1\}$ with

$$f(x) = ((x[k] = c) \wedge f(x)) \vee ((x[k] \neq c) \wedge f_1(x[I_1 \setminus \{k\}])) .$$

This reduces the set of significant variables to $I_1 \setminus \{k\}$ if $x[k] \neq c$. We next study what happens on input strings with $x[k] = c$.

Lemma 12 *Assume that a set I_1 with $n > |I_1| \geq 2$ is (k,c)-dominated with $k \in I_1$ for some $c \in \{0,1\}$. Then for every pair $w_1, w_2 \in \{0,1\}^n$ with $w_1[k] = w_2[k] = c$ and $w_1[i] = w_2[i]$ for all $i \in [n] \setminus I_1$ we have*

$$f(w_1) = f(w_2) \, .$$

Proof: Proof by contradiction. Assume that there exists a pair of strings $w_1, w_2 \in \{0,1\}^n$ with $w_1[k] = w_2[k] = c$ and $w_1[i] = w_2[i]$ for all $i \in [n] \setminus I_1$ such that

$$f(w_1) \ \neq \ f(w_2) \, .$$

Choose $(\alpha, J_1) \in$ f-set(I_1) and (β, J_2) as a counterpart of (α, J_1), i.e. there exists a w_3 with

$$f(w_3\lceil_{J_1,J_2 \leftarrow \alpha,\beta}) \ \neq \ f(w_3\lceil_{J_1,J_2 \leftarrow \overline{\alpha},\beta}) \ = \ f(w_3\lceil_{J_1,J_2 \leftarrow \alpha,\overline{\beta}}) \, .$$

W.l.o.g. we may assume that $f(w_3\lceil_{J_1,J_2 \leftarrow \alpha,\beta}) = f(w_1)$. Define $\beta_1 = \beta$, $I_{2,1} = J_2$, $I_{1,1} := \{i \in [n] \mid w_1[i] \neq w_2[i]\}$, and $\alpha_1 := w_1[I_{1,1}]$. Then $w_1 = w_1\lceil_{I_{1,1} \leftarrow \alpha_1}$ and $w_2 = w_1\lceil_{I_{1,1} \leftarrow \overline{\alpha}_1}$. Hence,

$$f(w_1\lceil_{I_{1,1} \leftarrow \alpha_1}) \ = \ f(w_3'\lceil_{I_{2,1} \leftarrow \beta_1}) \ \neq \ f(w_1\lceil_{I_{1,1} \leftarrow \overline{\alpha}_1}) \ = \ f(w_3'\lceil_{I_{2,1} \leftarrow \overline{\beta}_1}) \, ,$$

where $w_3' = w_3\lceil_{J_1 \leftarrow \alpha}$. By construction, $k \notin I_{1,1}$ and $k \notin I_{2,1}$. Moreover, $I_{1,1} \subseteq I_1$. Lemma 8 yields that I_1 is not dominated by k — a constradiction. □

Thus, we can conclude that for each set I_1 with $n > |I_1| \geq 2$ that is (k,c)-dominated with $k \in I_1$, there exists a function $f_2 : \{0,1\}^{|I_2|} \to \{0,1\}$ with

$$f(x) \ = \ ((x[k] = c) \wedge f_2(x[I_2])) \vee ((x[k] \neq c) \wedge f_1(x[I_1 \setminus \{k\}])) \, .$$

Summarising the previous three lemmas we get the following result.

Theorem 4 *Assume that a set I_1 with $n > |I_1| \geq 2$ is (k,c)-dominated with $k \in I_1$ for some $c \in \{0,1\}$. Let $I_2 = [n] \setminus I_1$. Then there are two functions $f_1 : \{0,1\}^{|I_1|-1} \to \{0,1\}$ and $f_2 : \{0,1\}^{|I_2|} \to \{0,1\}$ such that*

$$f(x) \ = \ ((x[k] \neq c) \wedge f_1(x[I_1 \setminus \{k\}])) \ \vee \ ((x[k] = c) \wedge f_2(x[I_2])) \, .$$

Note, that k, I_1, and I_2 are uniquely determined by f in the following sense.

Lemma 13 *Let f be dominated function and let I_1 with $n > |I_1| \geq 2$ be (k,c)-dominated. Then $\ell = k$ and $J_1 = I_1$ holds for every set J_1 with $n > |J_1| \geq 2$ that is (ℓ,c)-dominated.*

Proof: By Theorem 4, f can be written as

$$f(x) \ = \ ((x[k] \neq c) \wedge f_1(x[I_1 \setminus \{k\}])) \ \vee \ ((x[k] = c) \wedge f_2(x[I_2])) \qquad (8)$$

and

$$f(x) \;=\; ((x[\ell] \neq c) \wedge f_1(x[J_1 \setminus \{\ell\}])) \;\vee\; ((x[\ell] = c) \wedge f_2(x[J_2]))\,, \qquad (9)$$

where $I_2 = [n] \setminus I_1$ and $J_2 = [n] \setminus J_1$. Assume on the contrary that $k \neq \ell$. W.l.o.g., $\ell \in I_1 \setminus \{k\}$. Choose strings $a \in \{0,1\}^{|J_1|-1}$ and $b \in \{0,1\}^{|J_2|}$ such that $f_1(a) \neq f_2(b)$ and such that the value c is assigned to $x[k]$. (If $k \in J_1 \setminus \{\ell\}$, choose a such that $x[k]$ gets the value c. Then choose b such that $f_1(a) \neq f_2(b)$. This is possible, since f is non-degenerate. The case $k \in J_2$ is treated in the same manner.) Now (8) says that the restriction of $f\lceil_{J_1,J_2 \leftarrow a,b}$ is a constant function whereas (9) tells us that it is not — a contradiction. Thus, $\ell = k$.

It follows that also $J_1 = I_1$ must hold. If there is, say, an $m \in I_1 \setminus J_1$, then from (8) it follows that $f\lceil_{\{k\} \leftarrow c}$ depends on $x[m]$ whereas (9) yields that is does not — a contradiction. □

The only degree of freedom in Theorem 4 is to replace I_1 by $I_1 \setminus \{k\}$ and I_2 by $I_2 \cup \{k\}$ and flip the value c. That does not change the expression for f. Hence, every dominated function can be described by an `if-then-else` construction, i.e. it is of the form `if` $x[k] = c$ `then` $f_1(x[I_1 \setminus \{k\}])$ `else` $f_2(x[I_2])$.

Theorem 4 immediately implies that dominated functions can be computed on networks that are not 2-connected.

Theorem 5 *If f is ℓ-dominated with $n - 1 > \ell > 2$, then f can be computed by a private protocol on a network that consists of two 2-connected components with one node in common. One of the components has size ℓ and the other one size $n - \ell + 1$. If f is 2-dominated, then f cannot be computed by a private protocol.*

Proof: The corresponding protocol works as the one presented in Section 1.2. Any Boolean function can privately be evaluated, if at least three players are involved.

It remains to show that if f is 2-dominated, then f cannot be computed by a private protocol. We show that if this would be the case, then two players can privately compute $x \wedge y$. This is a contradiction.

Assume that f can be computed by a private protocol on a non-2-connected network G. By Theorem 3 and Lemma 13, G can be decomposed into two networks G_1 and G_2 such that G_1 and G_2 have sizes 2 and $n-1$, respectively, and share exactly one player. From the proof of Theorem 3, it also follows that f is dominated by the set of indices corresponding to the two players in G_1. Let $\{\ell, k\}$ be the indices of these two players and let P_k be the bridge node.

By Theorem 4, f is of the form $(x \wedge y) \vee (\overline{x} \wedge f_2(z))$. Here x is the input bit of P_k or its negation, depending on the value of c in Theorem 4. Furthermore y is the input of P_ℓ or its negation, depending on f_1. (There are only two non-degenerate boolean functions of one variable.) z is a vector of size $n-2$. From a private protocol for f, we construct a private protocol for $x \wedge y$ that can be executed by only two players. Player P_k chooses a vector ζ such that $f(\zeta) = 0$. Now P_ℓ and P_k simulate the protocol for f. P_k simulates all other players in G_2 presuming that their input is ζ. This protocol

is still private, since P_ℓ can only communicate with P_k in G. This contradicts the fact that there is no private protocol for $x \wedge y$. □

Theorem 5 can be generalised to the case where we allow teams of players to work together. Assume that all members of a team belong to the component that computes, say, f_1. Then f is t-private if f_1 is t-private. If the members are distributed among both components, then this virtually decreases the team sizes for both components. Thus, f is t-private if both f_1 and f_2 are t-private.

6 Conclusions and Open Problems

We have investigated the relation between the connectivity of networks and the possibility of computing functions by private protocols on these networks. Special emphasis has been put on the amount of randomness needed.

We have presented a general simulation technique that allows us to transfer every oblivious private protocol on an arbitrary network into an oblivious private protocol on a given k-connected network of the same size, where $k \geq 2$. The new protocol needs $(1 - \frac{k}{n-1}) \cdot \min\{L, \frac{k-2}{k-1} \cdot (n^2 - n) + \frac{L}{k-1}\}$ additional random bits, where L is the total number of bits sent in the original protocol. The obvious open question here is either to further reduce the number of additional random bits or to prove general lower bounds.

The parity function can be computed on a cycle using only one random bit and only one message per link. (Strictly speaking, an additional message per link is necessary to broadcast the result in the end. But we do not need to use any random bits to encode this broadcast, hence we can assume that n bits are sent altogether.) Thus, $1 + n - \frac{kn}{n-1} \leq n - k + 1$ random bits are sufficient to compute the parity function on an arbitrary k-connected graph by a private protocol using our simulation. We have strengthened this bound by showing that on every k-connected graph, parity can be computed by an oblivious private protocol using at most $\lceil \frac{n-2}{k-1} \rceil - 1$ random bits. Furthermore, there exist k-connected networks for which this bound is tight. The latter bound even holds for non-oblivious protocols.

While every Boolean function can be computed on a 2-connected network by a private protocol, this is no longer true for 1-connected networks. Starting from this observation, we have completely characterised the functions that can be computed by a private protocol on non-2-connected networks.

Our simulation results focus on the extra amount of randomness needed. It would also be interesting to bound the number of rounds of the simulation in terms of the number of rounds of the original protocol and, say, the diameter of the new network.

Acknowledgement

We thank Adi Rosén for valuable discussions and hints to literature.

References

[1] Noga Alon, Joel H. Spencer, and Paul Erd¨os. *The Probabilistic Method*. John Wiley and Sons, 1992.

[2] Judit Bar-Ilan and Donald Beaver. Non-cryptographic fault-tolerant computing in a constant number of rounds of interaction. In *Proc. of the 8th Ann. Symp. on Principles of Distributed Comput. (PODC)*, pages 201–209. ACM, 1989.

[3] Michael Ben-Or, Shafi Goldwasser, and Avi Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation. In *Proc. of the 20th Ann. Symp. on Theory of Comput. (STOC)*, pages 1–10. ACM, 1988.

[4] Carlo Blundo, Alfredo de Santis, Giuseppe Persiano, and Ugo Vaccaro. Randomness complexity of private computation. *Comput. Complexity*, 8(2):145–168, 1999.

[5] Ran Canetti and Rafail Ostrovsky. Secure computation with honest-looking parties: What if nobody is truly honest? In *Proc. of the 31st Ann. Symp. on Theory of Comput. (STOC)*, pages 255–264. ACM, 1999.

[6] David Chaum, Claude Crépeau, and Ivan Damgård. Multiparty unconditionally secure protocols. In *Proc. of the 20th Ann. Symp. on Theory of Comput. (STOC)*, pages 11–19. ACM, 1988.

[7] Benny Chor, Mihály Geréb-Graus, and Eyal Kushilevitz. On the structure of the privacy hierarchy. *J. Cryptology*, 7(1):53–60, 1994.

[8] Benny Chor, Mihály Geréb-Graus, and Eyal Kushilevitz. Private computations over the integers. *SIAM J. Comput.*, 24(2):376–386, 1995.

[9] Benny Chor and Eyal Kushilevitz. A zero-one law for boolean privacy. *SIAM J. Discrete Math.*, 4(1):36–47, 1991.

[10] Benny Chor and Eyal Kushilevitz. A communication-privacy tradeoff for modular addition. *Inform. Process. Lett.*, 45(4):205–210, 1993.

[11] Yoshimi Egawa, Rainer Glas, and Stephen C. Locke. Cycles and paths through specified vertices in k-connected graphs. *J. Combin. Theory Ser. B*, 52:20–29, 1991.

[12] Matthew Franklin and Rebecca N. Wright. Secure communication in minimal connectivity models. *J. Cryptology*, 13(1):9–30, 2000.

[13] Matthew Franklin and Moti Yung. Secure hypergraphs: Privacy from partial broadcast. In *Proc. of the 27th Ann. Symp. on Theory of Comput. (STOC)*, pages 36–44. ACM, 1995.

[14] Anna Gál and Adi Rosén. A theorem on sensitivity and applications in private computation. *SIAM J. Comput.*, 31(5):1424–1437, 2002.

[15] Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman and Company, 1979.

[16] Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play any mental game or a completeness theorem for protocols with honest majority. In *Proc. of the 19th Ann. Symp. on Theory of Comput. (STOC)*, pages 218–229. ACM, 1987.

[17] Eyal Kushilevitz. Privacy and communication complexity. *SIAM J. Discrete Math.*, 5(2):273–284, 1992.

[18] Eyal Kushilevitz and Yishay Mansour. Randomness in private computations. *SIAM J. Discrete Math.*, 10(4):647–661, 1997.

[19] Eyal Kushilevitz, Rafail Ostrovsky, and Adi Rosén. Characterizing linear size circuits in terms of privacy. *J. Comput. System Sci.*, 58(1):129–136, 1999.

[20] Eyal Kushilevitz and Adi Rosén. A randomness-rounds tradeoff in private computation. *SIAM J. Discrete Math.*, 11(1):61–80, 1998.

[21] Ingo Wegener. *The Complexity of Boolean Functions*. Wiley-Teubner, 1987.

[22] Andrew Chi-Chih Yao. Protocols for secure computations. In *Proc. of the 23rd Ann. Symp. on Foundations of Comput. Sci. (FOCS)*, pages 160–164. IEEE, 1982.

[23] Andrew Chi-Chih Yao. How to generate and exchange secrets. In *Proc. of the 27th Ann. Symp. on Foundations of Comput. Sci. (FOCS)*, pages 162–167. IEEE, 1986.

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'