Guessing secrets
(Extended Abstract)

Fan Chung * Ronald Graham *† Tom Leighton †‡

1 Introduction

We consider a variant of the familiar "20 questions" problem in which one tries to discover the identity of some unknown "secret" by asking binary questions (e.g., see [2]). In this variation, there is now a set of two (or more) secrets. For each questio asked, an adversary gets to choose *which* of the secrets to use in supplying the answer, which in any case must be truthful. We will describe a number of algorithms for dealing with this problem, although we are still far from a complete understanding of the situation. Problems of this type have recently arisen in connection with certain Internet traffic routing applications.

2 The basic setup

To begin with we will restrict ourselves to the case that the *adversary* **A** has just two (distinct) "secrets", say $S = \{S_1, S_2\}$ – taken from a finite set Ω. A question is then just a function $F : \Omega \to \{0, 1\}$. The questioner **Q**'s job is to select questions so as to determine as much about the secrets as effectively as possible. We first remark that **Q** can never hope to learn with certainty more than just one of **A**'s secrets, since **A** can always answer every question using the *same* $S_i \in S$. We can model the situation in terms of graphs. Let K_N denote the complete graph on the set Ω. A pair of secrets $\{S_1, S_2\}$ corresponds to an *edge* of K_N. Each question F induces a partition $\Omega = F^{-1}(0) \cup F^{-1}(1)$ of the vertex set of K_N. The answer to the question F given by **A**, e.g., $F \to 0$, implies that $S \cap F^{-1}(1) = \emptyset$. That is, **Q** can remove all the edges spanned by $F^{-1}(1) \subset \Omega$ as possible candidates for S. **Q** now chooses another question F' and repeats the process. **Q** is finished when the set of surviving edges contains *no pair* of *disjoint edges*. Thus, these edges now either form a star on some vertex S_0 (and so S_0 is one of **A**'s secrets), or a triangle on three vertices A, B, C (and S can be any two of these three).

We first note that this is the most **Q** could hope for under these constraints. or **A** always has the option for a fixed vertex S_0 of always choosing the part $F^{-1}(\delta)$ containing S_0. In this way, no edge containing S_0 will ever be removed, although **Q** might be able to assert that S_0 is at least one of **A**'s secrets. Howev , **A** can prevent even this from happening with the following strategy. Namely, **A** chooses a fixed set $S' = \{S'_1, S'_2, S'_3\}$ of three distinct secrets. For every question F, **A** selects the part $F^{-1}(\delta)$ to keep which contains at least two of the three S'_i. In this way, any of the three pairs $\{S'_1, S'_2\}, \{S'_1, S'_3\}, \{S'_2, S'_3\}$ could have been **A**'s secret pair, and so in this case **Q** cannot even claim that any particular element S_0 had to be in S. However, this is the most ambiguity t t **A** can enforce, as we will now see.

3 Adaptive algorithms

We first focus on *adaptive* algorithms, i.e., where future questions can depend on past answers.

We say that there is an *edge-separating* strategy of length t if no matter how **A** answers, **Q** can find questions which guarantee that after t steps, the set of surviving edges contains no disjoint pairs of edges. Let $f(N)$ denote the smallest value of t such that there is an edge-separating strategy of length t.

THEOREM 3.1.

$$f(N) < 4 \log_2 N + O(1)$$

Proof sketch. Proof is by induction on N. Let $K(a, b)$ denote a complete bipartite graph with all edges between two sets of a and b vertices. Also, denote by $K(a, b, c)$ the corresponding tripartite graph, and

*University of California, San Diego
†Akamai Technologies
‡MIT

let $\bar{K}(a,b)$ denote the subgraph of $K(a,b)$ formed by adding in all the edges with both endp nts in b. Given either of the configurations $\bar{K}(a,b)$ or $K(a,b,c)$, by asking suitable questions **Q** can force **A** to create similar (but smaller) configurations of these types. For example, starting with $\bar{K}(a,N-a)$, two questions can lead to either $\bar{K}(a/2,N-a/2)$ or $K(a/2,a/2,(N-a)/2)$. Hence,

$$\begin{aligned} & f(\bar{K}(a,N-a)) \\ \leq\ & 2+\max\{f(\bar{K}(\frac{a}{2},N-\frac{a}{2}),f(K(\frac{a}{2},\frac{a}{2},\frac{N-a}{2}))\end{aligned}$$

Also, starting with $K(a,a,N-2a)$, four questions can reduce the graph to $K(a/2,a/2,N/2-a)$. Therefore,

$$f(K(a,a,N-2a)) \leq 4f(K(\frac{a}{2},\frac{a}{2},\frac{N}{2}-a))$$

A careful treatment of the corresponding recurrences yields the desired upper bound of $4\log_2 N+O(1)$ (where we observe that the starting c figuration K_N can be reduced by one question to $\bar{K}(N/2,N/2)$.

THEOREM 3.2.

$$f(N) > 3\log_2 N + O(1)$$

Proof: Observe that for any graph G on N vertices, the set of triangles of G killed by removing the edges in $F^{-1}(0)$ is *disjoint* from the set of triangles killed by removing the edges in $F^{-1}(1)$. Since **Q** is not finished until at most one triangle remains, and we start with $\binom{N}{3}$ triangles in K_N, then **A** can force **Q** to ask at least $\log_2\binom{N}{3}-1$ questions until at most one triangle is left, simply by always choosing the answer $F(\delta)$ which minimizes the number of triangles destroyed. This gives the claimed lower bound.

Note that in the classical case where **A** has a single secret, $\log_2 N$ questions are required by the standard information-theoretic bounds [3]. We suspect that in fact $f(N) \sim c_0 \log_2 N$ for an appropriate c_0 (perhaps $c_0 = 4$?).

4 Oblivious algorithms

In the case of oblivious algorithms (where all questions are asked before any answers are given), let $f_0(N)$ denote the corresponding minimum number of questions needed to separate edges in K_N.

THEOREM 4.1.

$$f_0(N) \leq (\frac{4}{\log_2 8/7})\log_2 N$$

(Note: $\frac{4}{\log_2 8/7} = 20.76\ldots$*)*

Proof sketch: Label each vertex S of Ω independently with a binary t-tuple $\lambda(S) = (S(1),S(2),\ldots,S(t))$.The value of $S(i)$ will correspond to the part of the i-th partition $\Omega = F_i^{-1}(0)\cup F_i^{-1}(1)$ to which S belongs. The assignment λ separates the disjoint pairs $S=\{S_1,S_2\}$ and $T=\{T_1,T_2\}$ provided for some i, $S_1(i)=S_2(i)\neq T_1(i)=T_2(i)$. There are therefore just 14 of the 16 possible choices for the i-th *coordinate* so that this does not happen. Hence, the probability that λ does *not* separate S from T is $\leq (\frac{7}{8})^t$. Since there are just $\frac{1}{2}\binom{N}{2}\binom{N-2}{2}$ disjoint pairs in K_N, then some separating λ must exist provided

$$(\frac{7}{8})^t\frac{1}{2}\binom{N}{2}\binom{N-2}{2} < 1$$

which yields the asserted bound.

We do not have any better lower bound at present for f_0 than what Theorem 2 gives.

5 Inner product strategies

One disadvantage of the preceding approaches is that the questions needed to ach ve the $c\log N$ bounds might in fact require cN bits for their description. We would like questions for **Q** which can bc represented very compactly, e.g., using just $c\log N$ bits. One way to do this is as follows. Let us represent Ω as $GF(2)^n$, an n-dimensional vector space over $GF(2)$ (so that $N=2^n$). A question now is represented by a vector $F=(F(1),F(2),\ldots,F(n))$, $F(i)\in\{0,1\}$. The answer to the question F will be $F\cdot S_i$, the inner product (mod 2) of F with some $S_i \in S$. We will call strategies for separating edges in this setting "inner product" strategies.

THEOREM 5.1. *There is an inner product edge-separating strategy with at most* $\frac{3}{\log_2 8/7}\log_2 N$ *questions.*

Proof sketch: Choose a random set of $\frac{3}{\log_2 8/7}n$ inner product questions. A particular question F will separate the pair $S=(S_1,S_2)$ and $T=(T_1,T_2)$ provided

$F \cdot S_1 \equiv F \cdot S_2 \not\equiv F \cdot T_1 \equiv F \cdot T_2 \pmod 2$. One can show that for $S \cap T = \emptyset$, a random F separates and T with probability $\geq 1/8$. This implies the asserted bound.

Another way of generating inner product questions is to select all the consecutive blocks of length n in a longer binary sequence B of length βn. For this case, we look at the $3 \times n$ array Δ given by

$$\Delta = \begin{bmatrix} \Delta_1(1) & \Delta_1(2) & \dots & \Delta_1(n) \\ \Delta_2(1) & \Delta_2(2) & \dots & \Delta_2(n) \\ \Delta_3(1) & \Delta_3(2) & \dots & \Delta_3(n) \end{bmatrix} = \begin{bmatrix} \Delta_1 \\ \Delta_2 \\ \Delta_3 \end{bmatrix}$$

where

$$\Delta_1(i) \equiv X_1(i) - X_2(i)$$
$$\Delta_2(i) \equiv X_2(i) - Y_1(i)$$
$$\Delta_3(i) \equiv Y_1(i) - Y_2(i)$$

for $1 \leq i \leq n$. It is not hard to show that there are always at most three columns of Δ, say, in positions i, j and k, such that by choosing the values of F in these positions (namely $F(i), F(j), F(k)$) correctly, we can guarantee that $F \cdot \Delta_1 \equiv 0$, $F \cdot \Delta_1 \equiv 1$, $F \cdot \Delta_3 \equiv 0$, i.e., F separates $X = \{X_1, X_2\}$ from $Y = \{Y_1, Y_2\}$. Thus if F does *not* separate X from Y then (at least) one of the eight choices for $F(i), F(j), F(k)$ must be forbidden. The idea is now to pack as many disjoint translates of the pattern $\{i, j, k\}$ into $\{1, 2, \dots, \beta n\}$ as possible where we know that $\{i, j, k\} \subseteq \{1, 2, \dots, n\}$. Any translate of the pattern could separate X and Y for some translated question (all cyclic translates are allowed) and so must have restricted F values. It is not hard to show that if $\beta \geq 10$, for example, then at least $\frac{1}{5}\beta n$ disjoint translates of the pattern $\{i, j, k\}$ can be found in $\{1, 2, \dots, \beta n\}$. Thus, the probability that F is bad for the pair X, Y (i.e., does not separate them) is $\leq (7/8)^{\beta n/5}$. Since there are at most 2^{3n} choices for the Δ entries then our bound follows, and we have

THEOREM 5.2. *There is an inner product edge-separating strategy made up of consecutive blocks from some sequence B of length $(\frac{15}{\log_2 8/7}) \log_2 N$.*

One way to make the sequences in Theorem 5 more constructive is to select for B a sequence which has some provable random-like properties. One such sequence is the characteristic function for the quadratic residues of a large prime.

For this construction, we choose a prime $p > 144n^2$ and we form the sequence $Q = (q(1), q(2), \dots, q(p-1))$ where $q(k) = \frac{1}{2}(1 + \chi_p(k)) \in \{0, 1\}$ and χ_p is a non-trivial quadratic character modulo p, i.e.,

$$\chi_p(k) = \begin{cases} 1 & \text{if } k \text{ is a quadratic residue of } p, \\ -1 & \text{if } k \text{ is a quadratic non-residue of } p, \\ 0 & \text{if } k \equiv 0 \pmod p. \end{cases}$$

As before, for a given disjoint pair $X = \{x_1, x_2\}$ and $Y = \{y_1, y_2\}$, we form the difference array Δ:

$$\begin{array}{rcccl} \Delta_1 & = & X_1 - X_2 & = & \Delta_1(1)\Delta_1(2)\dots\Delta_1(n) \\ \Delta_2 & = & Y_1 - X_2 & = & \Delta_2(1)\Delta_2(2)\dots\Delta_2(n) \\ \Delta_3 & = & Y_2 - Y_1 & = & \Delta_3(1)\Delta_3(2)\dots\Delta_3(n) \end{array}$$

We want to show that if p is sufficiently large then there must be a block F of Q of length n such that

$$(5.1) \quad F \cdot \begin{pmatrix} \Delta_1 \\ \Delta_2 \\ \Delta_3 \end{pmatrix} \equiv \begin{pmatrix} F \cdot \Delta_1 \\ F \cdot \Delta_2 \\ F \cdot \Delta_3 \end{pmatrix} \equiv \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}$$

For this application we would like to consider 0 to be a quadratic residue of p. So define,

$$\chi_p^*(k) = \begin{cases} \chi_p(k) & \text{if } k \neq 0, \\ 1 & \text{if } k = 0 \end{cases}$$

Next, we consider the following sum:

$$(5.2) \quad W = \sum_{x=0}^{p-1} (1 - \prod_{i=1}^{n} \chi^*(x + \Delta_1(i))) (1 + \prod_{j=1}^{n} \chi^*(x + \Delta_2(j))) (1 - \prod_{k=1}^{n} \chi^*(x + \Delta_3(k)))$$

Note that the term $1 - \prod_{i=1}^{n} \chi^*(x + \Delta_1(i))$ is 0 if an *even* number of the terms $\chi^*(x + \Delta_1(i))$ are -1, and 2 otherwise. Thus,

$$(5.3) \quad \frac{1}{2}(1 - \prod_{i=1}^{n} \chi^*(x + \Delta_1(i))) = F_x \cdot \Delta_i \pmod 2$$

where F_x is the n-block of Q shifted to the left by x. Hence, $W \geq 0$ in any case, and $W > 0$ if and only if some block F of Q satisfies (5.1). To estimate W, we

first expand it into 8 terms.

(5.4)

$$W = \sum_{x=0}^{p-1} 1 - \sum_{x=0}^{p-1} \prod_{i=1}^{n} \chi_p^*(x+\Delta_1(i)) \pm \ldots$$
$$- \sum_{x=0}^{p-1} \prod_{i=1}^{n} \chi_p^*(x+\Delta_1(i)) \prod_{j=1}^{n} \chi_p^*(x+\Delta_2(j)) \pm \ldots$$
$$+ \sum_{x=0}^{p-1} \prod_{i=1}^{n} \chi_p^*(x+\Delta_1(i))$$
$$\prod_{j=1}^{n} \chi_p^*(x+\Delta_2(j)) \prod_{k=1}^{n} \chi_p^*(x+\Delta_3(k))$$

We next recall the powerful Burgess-Weil character sum bound:

Theorem (Burgess [1]):

For distinct $a_1, a_2, \ldots, a_s$ modulo p, $s \geq 1$,

$$(5.5) \qquad |\sum_{x=0}^{p-1} \prod_{k=1}^{s} \chi_p(x+a_k)| \leq (s-1)\sqrt{p}$$

A simple modification of (5.4) with χ_p^* replacing χ_p gives

$$(5.6) \qquad |\sum_{x=0}^{p-1} \prod_{k=1}^{s} \chi_p^*(x+a_k)| \leq s\sqrt{p}$$

for $p \geq s^2$.

Observe that the only way for any of the sums in (5.6) to "collapse" (except for $\sum_{x=0}^{p-1} 1 = p$) is if each term $\chi_p(x+a_i)$ occurs in the product exactly twice. However, the assumption that $X \cap Y = \emptyset$ implies that this cannot happen. Hence, applying (??) to the terms in (5.4) gives

$$(5.7) \quad W \geq p - (3n\sqrt{p} + 3 \cdot 2n\sqrt{p} + 1 \cdot 3n\sqrt{p})$$
$$= p - 12n\sqrt{p}$$

which is greater than zero for $p > 144n^2$. This proves

Theorem 6. The inner product strategy made up of the p consecutive blocks of length $\log_2 N$ of the characteristic function sequence of the quadratic re dues of a prime p (where 0 is considered to be a quadratic residue) is edge-separating, provided $p > 144(\log_2 N)^2$.

Note that as before, our sliding window is allowed to go "around the corner" , i.e., all p cyclic translates are allowed.

We believe that this construction may well be valid for much smaller values of p, e.g., $p = O((\log N)^{3/2})$ or even $O(\log N)$. To prove, however, this would require a much more careful analysis of the various terms in (5.4). We have perfor d some limited computational experiments which are consistent with this belief.

Another way for constructing even shorter questions is to use very short inner product questions. One such example is the set of vectors with at most three 1's. This is an edge-separating set of inner product questions of size roughly $\frac{1}{6}(\log_2 N)^3$. However, in contrast to the preceding sets constructed by probabilistic methods nd also the quadratic residue construction), a secret (or the secret "triangle") canbe reconstructed from the answers in a reasonably efficient way (most of the time) by a backtrack and pruning algorithm.

6 Final remarks

One can also study this problem in the cases that **A** has more than two secrets, and/or there are more than two possible answers to the questions. Naturally, more secrets make it harder for **Q** to learn anything, while more possible answers make it easier. For example, if **Q** can ask just a single question with a 2-bit answer in the inner product scenario, then **Q** can always identify some secret of **A** (i.e., **Q** can resolve the 2-out-of-3 ambiguity). On the other hand, suppose **A** has a set of $r(t-1)+1$ secrets from which to choose to answer **Q**'s question, but each question can now have one of t different answers. Then by a simple majority strategy, **A** can make sure that **Q** will never be able to claim that any particular r-element set $T \subset \Omega$ contains one of A'ssecrets. The preceding analyses can also be carried out for these cases as well, although not as much is known here. One could also look at other variants, e.g., suppose **A** is allowed to *lie* a certain number (or fraction) of times. Now what should **Q** do?

References

[1] A. Burgess, On character sums and primitive roots, *Proc. London Math. Soc.* **12** (1962), 179-192.

[2] *I've Got a Secret*, a classic '50's television gameshow, see http://www.timvp.ivegotse.html

[3] D. E. Knuth, *The Art of Computer Programming, vol. 3, Sorting and Searching*, Addison Wesley, 2nd Edition, 1998.