PU.M.A. Vol. **16** (2005), No. 3, pp. 295–305

A variant of the Guessing Secrets game

Guy Louchard
Université Libre de Bruxelles,
Département d'Informatique, CP 212,
Boulevard du Triomphe, B-1050 Bruxelles, Belgium,
e-mail:louchard@ulb.ac.be

and
Claudio Marini
Dipartimento di Scienze Matematiche ed Informatiche,
Università degli Studi di Siena,
Pian dei Mantellini 44, 53100 Siena, Italy,
e-mail: marinic@unisi.it

and
Franco Montagna
Dipartimento di Scienze Matematiche ed Informatiche,
Università degli Studi di Siena,
Pian dei Mantellini 44, 53100 Siena, Italy,
e-mail: montagna@unisi.it

and
Giulia Simi
Dipartimento di Scienze Matematiche ed Informatiche,
Università degli Studi di Siena,
Pian dei Mantellini 44, 53100 Siena, Italy,
e-mail: simi@unisi.it

(Received: November 24, 2005)

Abstract. In [2], Chung et al. introduced the Guessing Secrets game. In [4] the authors introduce a variant of this game in which for each question the reference secret is chosen at random (with uniform distribution) by Responder. In this paper we investigate another variant in which Responder is required to answer truthfully to questions of the form *How many secrets are there in X?*, where X is a subset of the search space chosen by Questioner. We investigate the cost (number of questions needed) in the worst case, in the best case and in the average case.

Mathematics Subject Classifications (2000). 60C05, 91A05, 91A65

1 Introduction

Chung et al. [2] proposed an information-theoretic problem called *Guessing Secrets* which is a variant of the familiar "20 questions" game. The game can be formulated in the following way: a player **A** (also called *Questioner*) tries to discover the identity of a set S of $n \geq 2$ unknown *secrets* drawn by a second player **B** (also called *Responder*), from a large set Ω of N secrets. For simplicity we assume $\Omega = \{1, \ldots, N\}$, therefore secrets are numbers. Questioner **A**

tries to learn as much as possible about the elements of S by asking **B** binary (Yes/No) questions. For each question asked, **B** chooses which of the n secrets to use in supplying the answer, which in any case must be truthful. Guessing Secrets problems define a class of interesting combinatorial problems which has arisen in connection with various topics in computer science and transmission of digital data as separating systems into smaller units, diagnosing technical problems, protecting data from unauthorized reproduction, authenticating ownership claims [11].

If $N > n \geq 2$ and **B** behaves adversarily, then **A** cannot guess all of the secrets. For instance, if **B** chooses the same secret to answer every question **A** asks, **A** would never know anything about the other elements of S. Moreover, if **B** uses a simple strategy, **A** cannot even hope to determine with certainty one secret of **B**. More precisely, let $k \notin S$. Then according to the rules of the game, **B** can legally answer YES to the question *Is your number in X?* if and only if $\mid S \cap (X \cup \{k\}) \mid \geq 2$. It is easily seen that in this way **A** cannot distinguish the true secrets from k, so he cannot even guess one secret with certainty. The paper [2] contains a precise description of what **A** can learn about S: in fact the possible sets of n secrets form an intersecting hypergraph (an intersecting graph if $n = 2$). Moreover if $n = 2$, then it is shown ([2], [1], [3] and [10]) that the number of questions needed in order to obtain such an intersecting graph is $\mathcal{O}(\log_2(N))$. On the other hand, it is shown in [3] that the guessing secrets becomes intractable (NP-hard) when the number of elements of S is bigger than 2. In polynomial time only approximate solutions can be expected [1].

In [4] the authors introduce a variant of the Guessing Secrets game in which for each question the reference secret is chosen at random (with uniform distribution) by **B**. Unlike the case of classical Guessing Secrets, in this probabilistic variant **A** has a winning strategy. In fact, in [4] the authors introduce a simple probabilistic guessing algorithm based on binary search strategy that allows to guess all secrets of S with probability one. If $n \ll N$, then the number of questions needed is an asymptotically Gaussian random variable with expected value $\mathcal{O}(n^2 \log_2 N)$.

The Guessing Secrets game is also related to the (adaptive) Group Testing, cf [5]. A formulation of the problem is the following: one has N items, represented by natural numbers, n of which are defective. One may test any group of items and the test tells us if there is *at least one* defective among the group tested. The problem, given N and n, is to compute the minimum number of tests necessary to detect all defectives. The differences between classical Guessing Secrets, probabilistic Guessing Secrets and Group Testing are as follows: in the first game, to a question of the form *Is your number in X?*, **B** must answer YES if all the secrets are in X, NO if no secret is in X and can answer *arbitrarily* if some secrets are in X, but not all of them. In the probabilistic guessing secrets, **B** answers as above if X contains all the secrets or if X contains no secrets, but if X contains $0 < k < n$ secrets, then he answers YES with probability $\frac{k}{n}$. Finally, in the Group Testing, **B** answers YES if and only if X contains *at least one* secret.

By the large numbers law, in this probabilistic variant, if M is large enough and X contains k secrets, then over nM questions all of the form *Is your number in X?*, nearly $\frac{kM}{n}$ answers will be YES and the remaining ones will be NO. Thus if **A** asks *Is your number in X?* for nM times and he obtains a_Y YES answers, then with high probability the closest integer to $\frac{a_Y}{M}$ is k. In other words, with high probability with Mn questions **A** really obtains a probably correct answer to the question *How many secrets are there in X?*. This suggests the following variant:

VARIANT 1. A search space $\Omega = \{1, ..., N\}$ and a subset S of Ω of cardinality $n > 0$ are given (for simplicity we assume that N is a power of 2). Questioner can ask questions of the form *How many secrets are there in X?*, where $X \subseteq \Omega$. **B** has to answer truthfully.

In this paper we investigate a strategy for the Variant 1 shown above. This strategy is as follows:

STRATEGY 1. At each stage, **A** finds a collection $X_1, ..., X_k$ of mutually disjoint subsets of Ω ($k \leq n$), called *reference sets*, such that each X_i contains at least one secret, and each secret belongs to exactly one X_i. At the beginning, the only reference set is Ω. Suppose that at some stage r, the reference sets are $X_1, ..., X_k$. Then for each X_i of cardinality greater than 1, **A** divides X_i in two parts of the same cardinality, call them X_iL and X_iR, and asks: *How many secrets are there in X_iL?* (one question for each i). Then **A** updates the reference sets: let n_i be Responder's answer to the question: *How many secrets are there in X_iL?* If $n_i = 0$, then X_i is replaced by X_iR, if $n_i = n$, then X_i is replaced by X_iL, and if $0 < n_i < n$ then X_i is replaced by both X_iL and X_iR. As soon as all the X_i have cardinality 1, all the secrets have been guessed.

In this paper we compute the cost of our algorithm in the worst case, in the best case and in the average case. More precisely if the search space Ω has cardinality $N = 2^k$ and the number of secrets is n, in the worst case the cost is $nk - n\lceil \log_2 n \rceil + 2^{\lceil \log_2 n \rceil} - 1$, in the best case it is $k - z_r(n)$ (where $z_r(n)$ is the number of consecutive zeros at the right of the binary representation of n) and in the average case it is an asymptotically Gaussian random variable with mean $n \log_2 N - n \log_2 n + \alpha_1 n$ and variance $\alpha_2 n \log_2 n + \alpha_3 n$ (where α_i are constants).

2 Worst case analysis

When **A** plays Strategy 1, the resulting game G only depends on the choice of the secrets. We associate to any game G a tree T_G called *the game tree*. We start from an informal description. Each node of T_G represents a local state of knowledge. Nodes are labelled by a triple, whose elements respectively represent one of the reference sets, its cardinality and the number of secrets contained in it. Since **A** always divides the set in two parts of the same cardinality and N is a power of 2, the second element of the label is always a power of 2. If a node w has a label of the form $(X, h, 0)$, then we do not need the information

contained in it, as X does not contain any secret. So w has no sons (it is a leaf). If w is labelled by $(X, h, 1)$, then we use the information contained in it, but we are already able to compute the number of questions needed to guess the only secret contained in it (namely, $\log_2(h)$). Thus once again w has no sons.

DEFINITION 2.1 The *game tree* T_G corresponding to a game G in which **A** plays Strategy 1 is inductively defined as follows:

> T has a root r which is labelled by the triple (r_1, r_2, r_3) where $r_1 = \Omega$, $r_2 = N$ and $r_3 = n$. Moreover, we define $\mathrm{lev}\,(r) = 0$.
>
> Let w be a node of T, and let (w_1, w_2, w_3) be its label, where w_1 is a subset X of Ω, w_2 is its cardinality (a power of 2) and w_3 is the number of secrets contained in it. Then:
>
> > If either $w_3 = 0$ or $w_3 = 1$, then w is a leaf.
> >
> > Otherwise, **A** divides X in two parts of the same cardinality, XL and XR and asks: *How many secrets are there in XL?* Suppose that Responder's answer is j. Then w has two sons, w^l, labelled with $(XL, \frac{w_2}{2}, j)$, and w^r, labelled with $(XR, \frac{w_2}{2}, w_3 - j)$. Moreover, we put $\mathrm{lev}\,(w^l) = \mathrm{lev}\,(w^r) = \mathrm{lev}\,(w) + 1$.

The game tree T_G allows us to compute the number of questions needed to Questioner in order to win the game. Indeed, at each node w which is not a leaf, **A** asks exactly one question (more precisely, if the label of w is (X, r, s) with $s > 1$, then **A** asks *How many secrets are in XL?*). If a node w with label (X, r, s) is a leaf, then if $s = 0$, **A** does not ask any question about X, and if $s = 1$, then he asks $\log_2(r)$ questions in order to discover the only secret contained in X. We are going to prove that the worst case occurs when for every node w labelled with (X, r, s), Responder's answer to the question *How many secrets are there in XL?* is $\frac{s}{2}$ if s is even and either $\frac{s-1}{2}$ or $\frac{s+1}{2}$ if s is odd. If this case always occurs, we say that **B** *divides the secrets in two halves.*

LEMMA 2.2 *Suppose that* **A** *applies Strategy* 1 *and Responder always divides the secrets in two halves. Suppose that the cardinality of* Ω *is* $N = 2^k$*, and that* n *is the number of the secrets. Let* $H(N, n) = nk - n\lceil \log_2 n \rceil + 2^{\lceil \log_2 n \rceil} - 1$. *Then the number of questions needed to* **A** *in order to win the game is exactly* $H(N, n)$.

Proof. Note that under the assumptions of the present lemma, every leaf has a label of the form $(X, j, 1)$ (because if a node has such a label, then it is a leaf and if a node has a label of the form (X, j, i) with $i > 1$, then his sons cannot have labels of the form $(X, j, 0)$ because **B** divides the secrets in two halves. It follows that the number of leaves is equal to the number n of secrets. Let $a = \lceil \log_2(n) \rceil$, i.e., assume $2^{a-1} < n \leq 2^a$. Then, using again the fact that **B** divides the secrets in two halves, we see that the second component j of the label of a leaf is either $\frac{N}{2^a}$ or $\frac{N}{2^{a-1}}$. Thus we have some number h of leaves w with level $\mathrm{lev}\,(w) = a - 1$

and some number of leaves v of level $\mathrm{lev}(v) = a$. Note that there are 2^{a-1} nodes of level $a-1$. If h of them are leaves, then $2^{a-1}-h$ have label of the form $(X, j, 2)$ (that is X contains two secrets), therefore they split into two leaves. Since n is the total number of leaves, we have $n = h + 2(2^{a-1}-h) = 2^a - h$.

Now we compute the total number of the nodes which are not leaves: there are $2^a - 1$ nodes of level $\leq a-1$, and h of them are leaves. Since all nodes of level a are leaves, this number is $2^a - h - 1$.

Finally the total number of questions is given by the sum of the number $2^a - h - 1$ which are not leaves plus the number h of leaves of level $a-1$ multiplied by $k-a+1$, the number of questions needed to guess the only secret in the first component of the label of such leaves, plus the number $2(2^{a-1}-h)$ of leaves of level a multiplied by $k-a$, the number of questions needed to guess the only secret in the first component of the label of such leaves. In total, recalling that $n = 2^a - h$, we have

$$\begin{aligned} &2^a - h - 1 + h(k-a+1) + (2^a - 2h)(k-a) \\ &\quad = 2^a - 1 + h(k-a) + (2^a - 2h)(k-a) \\ &\quad = 2^a - 1 + (k-a)n, \end{aligned}$$

that is the claim. □

COROLLARY 2.3 *If $N = 2^k$ and $n = 2^h$ then $H(2^k, 2^h) = 2^h(k-h+1) - 1$.* □

THEOREM 2.4 *The worst case occurs if* **B** *always divides the secrets in two halves. Therefore, if Ω has cardinality $N = 2^k$ and the number of secrets is n, then in the worst case Questioner, when applying Strategy* 1*, needs $H(N, n)$ questions.*

Proof. We proceed by induction on the number of secrets n, and for fixed n, we induct on N. If there is exactly one secret, the claim is trivial. Thus suppose that the claim holds for all $r < n$ and, in the case of n secrets, for all search spaces of cardinality $2^{\overline{k}}$ with $\overline{k} < k$, and let us prove it in the case where the search space Ω has $N = 2^k$ elements and the number of secrets is n. Remind that **A** divides Ω in two parts ΩL and ΩR of cardinality 2^{k-1} and asks *How many secrets are there in ΩL?*. We assume that Responder's answer is $\lceil \frac{n}{2} \rceil + c$, where $\lceil \frac{n}{2} \rceil + c$ is an integer and $0 \leq \lceil \frac{n}{2} \rceil + c \leq n$. We may assume without loss of generality that $c \geq 0$. If $\lceil \frac{n}{2} \rceil + c = n$, then by the induction hypothesis in the worst case **A** needs $1 + H(2^{k-1}, n)$ questions to win the game, and an easy computation shows that $1 + H(2^{k-1}, n) \leq H(2^k, n)$. Thus suppose that $\lceil \frac{n}{2} \rceil \leq \lceil \frac{n}{2} \rceil + c < n$. To simplify notation, let $m = \lceil \frac{n}{2} \rceil$. Then by the induction hypothesis, in the worst case **A** wins the game using $1 + H(2^{k-1}, m+c) + H(2^{k-1}, m-c)$ questions, and we need to show that $1 + H(2^{k-1}, m-c) + H(2^{k-1}, m+c) \leq H(2^k, 2m)$. Let $\lceil \log_2(2m) \rceil = a$. Since $2^{a-2} < m + c \leq 2^a$ and $m - c < m + c$ we have to consider only the following two cases:

a) $\lceil \log_2(m+c) \rceil = a-1$ and $\lceil \log_2(m-c) \rceil = a-t$ with $t \in \mathbf{N}$ and $1 \leq t < a$

b) $\lceil \log_2 (m+c) \rceil = a$ and $\lceil \log_2 (m-c) \rceil = a - t$ with $t \in \mathbf{N}$ and $1 \leq t < a$.

In case a) we have to verify that

$$2m(k-1) - (m-c)(a-t) + 2^{a-t} - (m+c)(a-1) + 2^{a-1} \leq 2mk - 2ma + 2^a,$$

that is, that

$$-t(m-c) + (m-c) - 2^{a-t} + 2^{a-1} \geq 0. \tag{2.4.1}$$

Since $m - c \leq 2^{a-t}$, we have $-t(m-c) + (m-c) - 2^{a-t} + 2^{a-1} \geq -t2^{a-t} + 2^{a-1}$. Now let $f(t) = -t2^{a-t} + 2^{a-1}$. We have $f(1) = f(2) = 0$ and $f'(t) \geq 0$ for $t \geq \frac{1}{\ln 2}$. Thus for $1 \leq t < a$, (2.4.1) is satisfied.

Now assume case b). We have to verify that $2m(k-1) - (m-c)(a-t) + 2^{a-t} - (m+c)a + 2^a \leq 2mk - 2ma + 2^a$, that is, that

$$2m - t(m-c) - 2^{a-t} \geq 0. \tag{2.4.2}$$

Once again, $m - c \leq 2^{a-t}$, therefore we obtain $2m - t(m-c) - 2^{a-t} \geq 2m - 2^{a-t}(t+1)$. Now consider the function $f(t) = 2m - 2^{a-t}(t+1)$. We have $f'(t) \geq 0$ for $t \geq \frac{1-\ln 2}{\ln 2}$. Since $f(3) = 2m - 2^{a-2} \geq 0$, for $t \geq 3$ the formula (2.4.2) is verified. It remains to consider the cases $t = 1$ and $t = 2$.

For $t = 1$, (2.4.2) is verified because $m + c > 2^{a-1}$. For $t = 2$, (2.4.2) is verified if and only if $2c - 2^{a-2} \geq 0$. But $m + c > 2^{a-1}$ and $c - m \geq -2^{a-2}$, that is $2c \geq 2^{a-1} - 2^{a-2} = 2^{a-2}$.

The proof is now complete. □

3 Best case analysis

In this section we analyze the case in which **A** plays the **Strategy 1** and the best case occurs. Let us say that Case $(*)$ occurs if, when **A** plays Strategy 1, then to any question of the form: *How many secrets are there in X?* with X of cardinality h, **B** answers: *h secrets* if $h \leq n$, and: *n secrets* otherwise.

We first compute the number of questions needed to find all secrets if Case $(*)$ occurs, and then we prove that Case $(*)$ corresponds to the best case.

LEMMA 3.1 *Suppose that Ω has cardinality $N = 2k$ and that the number of the secrets is n. Moreover, suppose that* **A** *plays Strategy* 1 *and that Case* $(*)$ *occurs. Then the number of questions $Q^*(k, z_r(n))$ to find all the secrets is*

$$Q^*(k, z_r(n)) = k - z_r(n),$$

where $z_r(n)$ is the number of consecutive zeros at the right of the binary representation of n.

Proof. We argue by induction on the length l of the binary representation of n.
Base: for $l = 1$ the claim trivially holds.
Induction Step: suppose the claim holds if $l < h$, and let us prove it if $l = h$.

Let $1a_{h-2}...a_1a_0$ be the binary representation of n. Clearly **A** needs $k-(h-1)$ questions to find the first 2^{h-1} secrets. When he finds them, the updated reference set has cardinality 2^{h-1} and it contains $n-2^{h-1}$ secrets. Clearly the binary representation of $n-2^{h-1}$ is $a_{h-2}...a_1a_0$. By the induction hypothesis **A** needs $h-1-z_r(n-2^{h-1})$ questions in order to find the remaining secrets. Hence in total he needs $h-1-z_r(n-2^{h-1})+k-(h-1)=k-z_r(n)$ questions. □

Now we prove that Case $(*)$ is the best case.

THEOREM 3.2 *With the same notation as in Lemma* 3.1*, when playing Strategy* 1*,* **A** *needs at least* $k-z_r(n)$ *questions.*

Proof. We prove the claim by induction on n, and, with reference to the same n, by induction on k. For $n=1$, the claim is trivial. For the induction step, we distinguish the following cases:

Case a:) $n \leq 2^{k-1}$. Suppose that Responder's answer to the first question is: n_1 *secrets*. Let $n_2=n-n_1$. If $n_1=0$, then $n_2=n$, and by the induction hypothesis **A** needs at least

$$1+Q^*(2^{k-1},n_2)=1+(k-1)-z_r(n)=Q^*(2^k,n)$$

to guess all secrets. We obtain the same result if $n_1=n$. So suppose $0<n_1<n$. Then by the induction hypothesis the number of questions needed to guess all the secrets is:

$$1+k-1-z_r(n_1)+k-1-z_r(n_2)=2k-1-z_r(n_1)-z_r(n_2).$$

Clearly either $z_r(n_1)\leq z_r(n)$ or $z_r(n_2)\leq z_r(n)$. Suppose without loss of generality that $z_r(n_1)\leq z_r(n)$. Then

$$2k-1-z_r(n_1)-z_r(n_2)\geq 2k-1-z_r(n)-z_r(n_2)\geq 2k-1-z_r(n)-k+2,$$

because $z_r(n_2)<k-1$, therefore

$$2k-1-z_r(n)-k+2=k+1-z_r(n).$$

This completes the proof of case a).

Case b): $2^{k-1}<n<2^k$. We have to prove that

$$2k-1-z_r(n_1)-z_r(n_2)\geq k-z_r(n),$$

where n_1 and n_2 are the same of case a). Suppose without loss of generality that $z_r(n_1)\leq z_r(n)$. We have

$$2k-1-z_r(n_1)-z_r(n_2)\geq 2k-1-z_r(n)-z_r(n_2).$$

Since $z_r(n_2)<k$,

$$2k-1-z_r(n)-z_r(n_2)\geq 2k-1-z_r(n)-(k-1)=k-z_r(n).$$

This completes the proof of the case b), and the proof of the whole Theorem. □

4 Average case analysis

In this section we study the average case of the game. In particular we deal with the asymptotic mean and variance and asymptotic cost distribution.

4.1 First approach to the mean

Denote by $M(N,n)$ the mean number of questions necessary to find all n secrets. (of course one question is enough for each subdivision). Assume, for simplicity, that $N=2^u$ for some integer u. We have

$$\begin{aligned} M(N,n) &= 1+2\sum_0^n \binom{n}{j}\frac{1}{2^n}M(N/2,j),\\ M(N,0) &= 0,\\ M(N,1) &= \log_2 N,\\ M(1,i) &= 0, \forall i. \end{aligned}$$

This kind of recurrence has been fully analyzed in the literature: see, for instance Szpankowski [12], Sec.7.6 for an excellent treatment. Taking the exponential generating function

$$\Phi(N,z) := \sum_1^\infty \frac{z^n}{n!}M(N,n) = z\log_2 N + \ldots,$$

we derive

$$\begin{aligned} \Phi(N,z) - z\log_2 N = e^z - 1 - z + 2[\Phi(N/2,z/2) - \log_2(N/2)z/2]e^{z/2}\\ + 2(e^{z/2}-1)\log_2(N/2)z/2, \end{aligned}$$

i.e.

$$\Phi(N,z) = e^z - 1 + 2\Phi(N/2,z/2)e^{z/2}.$$

Set (this is the classical Poissonization procedure)

$$\Psi(N,z) = \Phi(N,z)e^{-z}, \text{ with } \Psi(N,z) = \sum_0^\infty \frac{z^k}{k!}b_k,$$

which leads to

$$\begin{aligned} M(N,n) &= \sum_0^n \binom{n}{k} b_k,\\ b_0 &= 0,\\ b_1 &= \log_2 N. \end{aligned}$$

We obtain

$$\Psi(N,z) = 1 - e^{-z} + 2\Psi(N/2,z/2),$$

and expanding,

$$\Psi(N,z) = 1 + 2 + 4 + \ldots + N/2 + N\Psi(1, z/N) - e^{-z} - 2e^{-z/2} - 4e^{-z/4} - \ldots - N/2e^{-z/(N/2)}.$$

After some algebra, we obtain

$$b_k = (-1)^{k-1}\frac{1/N^{k-1} - 1}{1/2^{k-1} - 1}, \qquad k \geq 2,$$

and

$$M(N,n) = \sum_2^n \binom{n}{k}(-1)^{k-1}\frac{1/N^{k-1} - 1}{1/2^{k-1} - 1} + \binom{n}{1}\log_2 N.$$

But now we can apply Rice's formula. According to Szpankowski [12], Sec. 8, we know that

$$\sum_{k=2}^{n+r} \frac{\binom{n+r}{k}(-1)^k\binom{k}{r}}{1 - 1/2^{k-1}} \sim \frac{1}{L}n[\ln n + \gamma - \delta_{r,0} + L^2/2] + nP_r(n) + e(n)$$

$r = 0, 1, n \to \infty$, where

$L :=$ $\ln 2$,

$P_r(n) :=$ periodic function of $\log_2(n)$, with period 1, mean 0 and small amplitude,

$e(n) :=$ $\mathcal{O}(1)$.

Asymptotically, using $r = 0$, this leads to

$$M(N,n) \sim \left[-n\log_2 n - \frac{n}{L}\left[\gamma - 1 + L^2/2\right] - nP_0(n) - e(n)\right] \cdot (1 + \mathcal{O}(1/N)) + n\log_2 N \quad \text{for } n \to \infty, N \to \infty, n = o(N). \tag{1}$$

4.2 Asymptotic distribution of the costs

Let us now turn to the costs.

Let c_1 be the cost to lead to subsets with one secret. We see that this is asymptotically equivalent to the internal nodes size $T(n)$ of a Trie with n keys. This is asymptotically Gaussian (see Jacquet and Régnier [7], Louchard [9]):

$$T(n) \sim \mathcal{N}(\alpha_1 n, \alpha_2 n),$$

with

$$\alpha_1 = 1/L, \alpha_2 = 1/(2L) - 1/L^2 + \alpha_3, \alpha_3 = 2\sum_1^\infty \frac{2^{-l}}{1 + 2^{-l}}/L.$$

Let c_2 be the cost starting from subsets with one secret to the secret itself. Each secret l is contained in a set (where it is the only secret) of size $N_l :=$

$N/2^{j(l)}$, where $j(l)$ is the length of the path from the root to secret l in the Trie. But this leads, for the subsequent number of subdivisions Λ of this set, to

$$\Lambda = \log_2 N_l = \log_2 N - j(l),$$

and a total cost

$$c_2 = n \log_2 N - EPL(n), \tag{2}$$

where $EPL(n)$ is the External path length of the Trie. But it is well known (see Jacquet and Régnier [8]), that $EPL(n)$ is asymptotically Gaussian

$$EPL(n) \sim \mathcal{N}(D_1, D_2),$$

with ($\alpha_.$ are constants we do not detail here)

$$D_1 = n \log_2 n + \alpha_4 n,$$
$$D_2 = \alpha_5 n \log_2 n + \alpha_6 n.$$

So finally, the total cost $c_1 + c_2$ is Gaussian with asymptotic mean

$$n \log_2 N - n \log_2 n - \alpha_4 n + \frac{1}{L} n = n \log_2 N - n \log_2 n + (\alpha_4 + \frac{1}{L}) n,$$

namely

$$n \log_2 N - n \log_2 n + \alpha_7 n,$$

(the dominant terms are the same as in (1)) and asymptotic variance bounded by

$$\left(\frac{1}{2L} - \frac{1}{L^2} + 2 \sum_{i=1}^{\infty} \frac{2^{-l}}{1 + 2^{-l}} / L \right) n + \alpha_5 n \log_2 n + \alpha_6 n,$$

namely

$$\alpha_5 n \log_2 n + \alpha_8 n.$$

Indeed, note that the size $T(n)$ and the external path length $EPL(n)$ are positively correlated. But the minus sign in (2) shows that we can safely add the variances to get an upper bound for the dispersion.

5 Conclusion and future work

In this paper we have investigated a variant of the Guessing Secrets game, in which Responder has to answer truthfully questions of the form: *How many secrets are there in X ?*. For this variant we have proposed a learning algorithm, and we have computed the cost in the worst case and in the best case. We have also investigated the average case, and we have computed the asymptotic mean, the asymptotic variance and asymptotic cost. Some problems remain open. In particular, we don't know if Strategy 1 proposed in the present paper is the best one in the worst case and in the average case.

References

[1] N. Alon, V. Guruswam, T. Kaufman and M. Sudan, *Guessing secrets efficiently via list decoding*, Proceeding of the 13th Annual ACM-SIAM Symposium On Discrete Mathematics (SODA-02), pages 254–262, 2002.

[2] F. Chung, R. Graham and T. Leighton, *Guessing Secrets*, The Electronic Journal of Combinatorics **8**, ♯R13, 2001.

[3] F. Chung, R. Graham and L. Lu, *Guessing secrets with inner product questions*, Proceeding of the 13th Annual ACM-SIAM Symposium On Discrete Mathematics (SODA-02), pages 247–253, 2002.

[4] A. Del Lungo, G. Louchard, C. Marini and F. Montagna, *The Guessing Secrets problem: a probabilistic approach*, Technical Report 2003, to appear in Journal of Algorithms.

[5] D.Z. Du and F.K. Hwang, *Combinatorial Group Testing and Its Applications*, World Scientific Pub Co, 2000.

[6] R. Durrett, *Probability: Theory and Examples*, Belmont, CA: Wadsworth, 1991.

[7] P. Jaquet and M. Régnier, *Trie partitioning process: limiting distributions*, Vol. 214 of Lecture Notes in Computer Science, pages 196–210, 1986.

[8] P. Jaquet and M. Régnier, *Normal limiting distribution for the size and external path length of tries*, Technical Report 827, INRIA, 1988.

[9] G. Louchard, *Trie size in a dynamic list structure*, Random Structures and Algorithms, **5** (5) (1994), 665–702.

[10] D. Micciancio and N. Segerlind, *Guessing two secrets with small queries*, Technical report, 2002.

[11] I. Peterson, *Guessing secrets*, Science News, **161** (14) (2002), 214–216.

[12] W. Szpankowski, *Average Case Analysis of Algorithms on Sequences*, Wiley, 2001.