

# Deconvolving Sequence Variation in Mixed DNA Populations

ANDY WILDENBERG,<sup>1</sup> STEVEN SKIENA,<sup>2</sup> and PAVEL SUMAZIN<sup>3</sup>

## ABSTRACT

We present an original approach to identifying sequence variants in a mixed DNA population from sequence trace data. The heart of the method is based on parsimony: given a wildtype DNA sequence, a set of observed variations at each position collected from sequencing data, and a complete catalog of all possible mutations, determine the smallest set of mutations from the catalog that could fully explain the observed variations. The algorithmic complexity of the problem is analyzed for several classes of mutations, including block substitutions, single-range deletions, and single-range insertions. The reconstruction problem is shown to be NP-complete for single-range insertions and deletions, while for block substitutions, single character insertion, and single character deletion mutations, polynomial time algorithms are provided. Once a minimum set of mutations compatible with the observed sequence is found, the relative frequency of those mutations is recovered by solving a system of linear equations. Simulation results show the algorithm successfully deconvolving mutations in p53 known to cause cancer. An extension of the algorithm is proposed as a new method of high throughput screening for single nucleotide polymorphisms by multiplexing DNA.

**Key words:** A\* search, Sanger sequencing, single nucleotide polymorphisms, multiplexing.

## 1. INTRODUCTION

THE NEED FOR DNA SEQUENCING DID NOT END with the successful public and private projects (Lander *et al.*, 2001; Venter *et al.*, 2001) to sequence the human genome. Indeed, attention is shifting from *de novo* sequencing of new organisms to analyzing sequence variation for research and diagnostic purposes.

Contemporary electrophoresis-based sequencing machines produce curves registering the amount of each of the four nucleotide bases as a function of sequence position (see Fig. 1). For homogeneous DNA samples, the largest peaks at each position define the underlying sequence. However, more careful analysis of sequence trace data holds promise for determining the presence and frequency of mutations in inhomogeneous samples.

In this paper, we look at the problem of using sequence trace data to identify sequence variants in mixed DNA populations. Our work is motivated by a new line of capillary electrophoresis sequencing machines

---

<sup>1</sup>Genera Biosystems, 4 Research Drive, Bundoora, 3151, Victoria, Australia.

<sup>2</sup>SUNY Stony Brook, Department of Computer Science, Stony Brook, NY 11794.

<sup>3</sup>Portland State University, Department of Computer Science, P.O. Box 751, Portland, OR 97207.

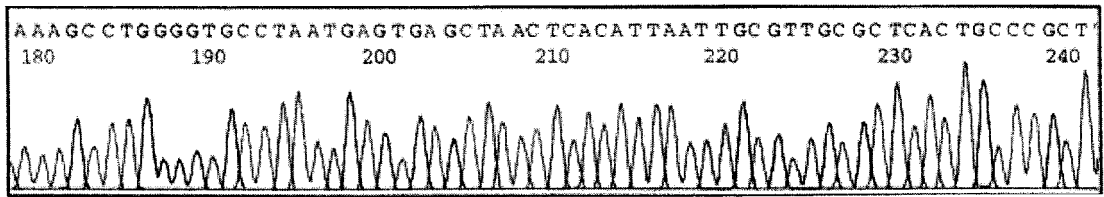


FIG. 1. Trace data from a DNA sequencing machine.

being developed by BioPhotonics Corporation (Gorfinkel and Luryi, 1998). By using advanced single-photon detectors and other technologies, BioPhotonics has the capability to not only detect but accurately determine the relative frequency of each base at each position to within 10% and expects to reduce this error rate to 1% in the near future.

This motivates a variety of questions concerning how accurately we can sequence mixed populations from a single sample using relative frequency information. Possible applications of this technology include analysis of frequency of acquired mutations, SNP generation and analysis, and viral population analysis.

**Frequency of acquired mutations:** Perhaps the greatest promise of modern genomics is that of individualized medicine, where an individual's genetic composition is determined and analyzed to determine the best course of treatment. New technologies such as microarrays (DeRisi *et al.*, 1997; Fodor *et al.*, 1991) offer promise for obtaining sequence and expression data on an individual scale. Microarray studies of leukemia and breast cancer (Ben-Dor *et al.*, 2000; Golub *et al.*, 1999) tissues have demonstrated that cancer subtypes can be accurately diagnosed on the basis of genomic data, and with them the prognosis for survival under various treatments.

Such microarray studies will continue to help develop our understanding of gene expression and disease. However, the technologies used for widespread diagnostic tests may well be different, to minimize costs and increase robustness. Indeed, a major goal of BioPhotonics efforts is developing smaller, cheaper DNA sequencing machines with the vision of placing them in doctor's offices for diagnostic applications. Particularly important for many medical applications is the need to analyze sequence from heterogeneous genomic samples. Such mixed populations naturally arise from acquired mutations, say, in cancer, where various mutations to oncogenes such as p53 can lead to dramatically different disease courses. Extensive databases of p53 mutations are being constructed, including those of Beroud and Soussi (1998), Hernandez-Boussard *et al.* (1999), and Tang *et al.* (2001).

In this paper, we provide simulation results demonstrating our ability to identify p53 mutations as a function of mutation frequency and sequencing accuracy.

**SNP generation and analysis:** Single-nucleotide polymorphisms (SNPs) represent an important part of sequence variation in humans. Cataloging SNPs is an important problem in contemporary sequence analysis (Group, 2001). Here we propose a potentially high-throughput technology to catalog SNPs. A pool of DNA from  $m$  distinct individuals is assembled, with a region of interest amplified using PCR. Sequencing the resulting product and deconvolving the results will be significantly more efficient than individual sequencing runs for large  $m$ , provided they can be accurately analyzed for large  $m$ .

In this paper, we study the potential of this approach both theoretically and through simulation. We demonstrate that, under reasonable assumptions of polymorphism rates and error probabilities, pool sizes of more than 100 people can be analyzed on a single sequencing run.

**Viral population analysis:** Viruses such as HIV evolve rapidly, and each infected patient soon comes to host a variety of different strains. Our techniques make it possible to determine the various mutations present in a sample, as well as their relative frequencies. Obtaining accurate viral population frequency data will be important to determine a patient's response to a given course of treatment and to determine which strains react best to a given therapy.

In this paper, we demonstrate that accurate determination of the relative frequencies of four distinct strains can be made, even in the face of base-frequency error rates up to 25%.

### 1.1. Problem definitions

If the population is comprised of two sequences that differ by a single base position, then identifying the single nucleotide polymorphism (SNP) is in principle easy: just look for the single base position with

two peaks. But how can we deconvolve more complicated mutation populations, with insertion and deletion operations? In this paper, we look at the algorithmic complexity of several sequence deconvolution problems—giving efficient algorithms where they exist and hardness proofs where they do not. We address three distinct problems of mutation deconvolution from sequence trace data: base calling, mutation deconvolution, and population frequency determination.

**Base calling:** The problem of calling bases from electrophoresis traces is complicated by a variety of technology-dependent factors. Programs such as Phred (Ewing *et al.*, 1998), TraceTuner (Denisov *et al.*, 2000), and LifeTrace (Walther *et al.*, 2001) successfully analyze trace data and assign each base an associated quality level or error probability.

In this paper, we assume the trace analysis is performed by an external program, which returns the percentage of each base observed at each position for subsequent analysis. We will assume that such data is subject to errors, as discussed in Section 4.

**Mutation deconvolution:** Given the set of observed bases at each position, which mutations are present in the given sample? In medical applications we can assume that the general wildtype sequence is known, as well as a catalog of commonly occurring or previously studied mutations. So the problem consists of a wildtype, a list of legal mutations on that wildtype, and a profile that lists observed nucleotides at given locations in the sequence. The goal is to find the smallest set of legal mutations that explains the observed profile. Formally, our problem is:

*Input:* A wildtype sequence  $S$  of length  $n$ , a set  $V$  of allowable variations on  $S$ , and an experimental profile, consisting of an ordered sequence of  $n$  subsets on alphabet  $\Sigma$ .

*Output:* The smallest subset  $V' \subset V$  such that the character position subsets of  $V'$  and  $S$  yield exactly the input profile.

Consider the following example, where wildtype string  $S = ACTGTTGACTCATCC$  of length  $n = 15$  with alphabet  $\Sigma = \{A, C, G, T\}$  gives rise to the following profile:

S: ACTGTTGACTCATCC  
 AGTC CTCATCG C

One solution with three mutations explaining this profile consists of a 4-base substitution starting at position 2 with the sequence AGTC, a deletion of an A at position 8, and an insertion of a G at position 14.

S: ACTGTTGACTCATCC	wildtype
a <u>AGTC</u> tgactcatcc	Sub(2,AGTC)
actgttg <u>CTCATCC</u>	Del(8,1)
actgttgactcat <u>G</u> cc	Ins(14,G)

Bases in the mutations that contribute to the profile are capitalized, and the portion of the mutation that has changed is underlined.

**Population frequency determination:** Sequence trace data provides a rough distribution of the relative frequency of each base at a given position. The medical implications of an acquired mutation rest not only on which mutations are present, but on how frequent they are in the given population. In the more general population frequency determination problem, we are given an observed fraction of each base at each position and have to reconstruct the frequency of each mutation in the population. Formally, our problem is:

*Input:* A wildtype sequence  $S$ , a set  $V$  of allowable variations on  $S$ , and an experimental profile, consisting of the observed relative frequency for each  $s \in \Sigma$  at each of  $n$  positions.

*Output:* The population frequency of each variation  $v \in V$  that best matches the observed input profile.

### 1.2. Organization

Our paper is organized as follows. In Section 2, we consider the computational complexity of several variants of the sequence deconvolution problem, establishing natural classes of mutations for which

reconstruction is NP-complete. Section 3 presents the practical algorithms which we developed for use in our implementation. This implementation was used to generate simulation results, described in Section 4. Analysis and experiments for the special case of reconstructing SNPs appears in Section 5, followed by future work in Section 6.

## 2. MUTATION DECONVOLUTION: SPECIAL CASES

Here we treat the mutation deconvolution problem from the strict standpoint of algorithmic theory. We demonstrate the boundary between classes of mutations that are algorithmically easy to reconstruct and more general classes that are hard. Further, we categorize certain classes of mutations which mask each other in trace data, thus being inherently unreconstructable.

We consider the mutation deconvolution problem for increasingly general sets of allowable variations of substitutions, deletions, and insertions.

### 2.1. Substitution sets

The simplest class of substitutions just change single bases. Given an alphabet  $\Sigma$  with  $\alpha = |\Sigma|$  letters in it, there are only  $(\alpha - 1)n$  single-base variation sequences. If only single character substitutions are allowed, there is only one possible way to create each additional profile character, and hence the algorithmic reconstruction problem is trivial.

A more interesting and general class of mutations allows larger substrings. Here, each element of the set of allowable variations substitute one substring of length  $k$  in  $S$  with a different  $k$ -substring. We assume all  $(n + 1 - k)\alpha^{2k}$  possible  $k$ -length substitutions exist in  $V$ . Again, reconstruction is trivial.

**Theorem 1.** *Mutation deconvolution for  $k$ -length substitutions can be solved in  $O(\alpha n)$  time.*

**Proof.** This can be solved optimally using a left-to-right greedy algorithm. Start with the leftmost uncovered profile character and select the string which covers them and as large a set of other uncovered profile characters as possible. Since all  $k$ -string replacements are available, no decision precludes any other covering option. ■

The incomplete substitution set problem is a restricted variant of the complete substitution set problem. Here, substitutions of  $k$ -substrings are made using only possible mutations from the set  $\bar{M}$ . Even though  $\bar{M}$  has fewer members than the previous problem, it is a much more difficult problem to solve.

**Theorem 2.** *The incomplete substitution set problem is NP-complete and is hard to approximate within a log factor.*

**Proof.** A solution to the problem can clearly be verified in polynomial time. To show hardness, we reduce the set-cover problem to the incomplete substitution set problem; this reduction maintains hardness of approximation.

The set-cover problem is defined as follows. Given an integer set  $N = \{1, \dots, n\}$  and a set  $M$  of  $m$  subsets of  $N$ , find the smallest subset  $\hat{M}$  of  $M$  such that  $\bigcup_{p \in \hat{M}} p = N$ . Given a set-cover instance  $(N, M)$ , we introduce a character-sequence representation for each of the elements of  $M$ . For all integers  $i$  from 1 to  $n$ , if  $i \in p$ , append “\*” to  $s$ , otherwise append “-” to  $s$ . Thus, for  $n = 5$ , “\*-\*-” represents the subset  $p = \{1, 3\}$ .

The wildtype  $S$  consists of  $n$  consecutive “-”s;  $a_i = \text{“-”}$  for all basepair positions of  $S$ . The profile consists of the wildtype plus a “\*” in each column. The subset  $\bar{M}$  consists of the character-sequence representations of the subsets in  $M$ . A mutation  $\hat{m}_k$  in  $\hat{M}$  is constructed using a substitution of the entire wildtype  $S$  with an  $n$ -substring in  $\bar{M}$ .

There is a clear one-to-one correspondence between the set cover instance and the encoding. Each mutation is created using a substitution of the wildtype with a substring in  $\bar{M}$ , and each substring in  $\bar{M}$  represents a subset in  $M$ . The reduction is correct and maintains hardness of approximation. ■

We note that the greedy heuristic for set cover gives an  $O(\lg |V|)$  factor approximation (Johnson, 1974) for this and indeed all variations of mutation deconvolution. The greedy heuristic identifies all elements of  $V$  consistent with the profile. While there are uncovered characters in the profile, select the survivor which covers the most remaining profile characters.

2.2. Deletion sets

In single character deletion, each possible mutation differs from wildtype  $S$  in the deletion of one character position. Thus, there are  $n$  possible variation sequences.

We claim that any realizable profile can be covered using  $S$  and at most one other sequence. Consider the profile generated by the following mutations:

S: ACTGTTGACTCATCC	wildtype
actGTTGCTCATCc	Del(4,1)
actgTtgctcatcc	Del(8,1)

Observe that Del(8,1) contributes nothing to the profile: Del(4,1) and Del(8,1) are identical to the right of the second deletion (position 8) and  $S$  is identical to Del(8,1) to the left of the deletion. A generalization of this argument yields the following.

**Lemma 1.** *Let  $V'$  be a minimum size solution for the mutation deconvolution problem. Then  $V'$  does not contain two deletion mutations of the same length  $k$  for any  $1 \leq k \leq n$ .*

**Proof.** Assume the converse, that a minimum size solution  $V'$  contains two mutations: a mutation  $m$  deleting characters from the half-open interval  $[i, i + k)$ , and a mutation  $n$  deleting characters from the half-open interval  $[j, j + k)$  (i.e., both  $m$  and  $n$  are deletions of length  $k$ ). Furthermore, assume  $i \leq j$ . Then characters  $1, \dots, j$  of  $n$  match the wildtype, and characters to the right of  $j$  match  $m$ , so the profile can be covered without  $n$ . Hence  $V'$  is not a minimum size solution. ■

Thus, to cover the rightmost uncovered character of the profile, we can select the variant which has the leftmost deletion which is consistent with the rest of the profile. Note that if this does not completely cover the profile, then the profile cannot be covered, as all other consistent profiles share a prefix with  $S$ . We obtain the following theorem.

**Theorem 3.** *The mutation deconvolution problem for  $k$ -length deletions can be solved in  $O(\alpha n)$  time.*

In the *single-range deletion* problem, each variation differs from  $S$  in the deletion of one contiguous subsequence, and all possible deletions are included. Thus, there are  $\binom{n}{2}$  possible variation sequences in  $V$ . More formally, the *single-range deletion* problem is stated as follows. Given a wildtype  $S$  and a set  $a_i$  of observed nonwildtype nucleotides for each basepair-position  $i$ , find the smallest set of mutations  $\hat{M}$  that differ from  $S$  in the deletion of a contiguous subsequence; where  $\hat{M}$  satisfies the condition  $a_i \subseteq \bigcup \hat{m}_k(i) \subseteq a_i \cup \{S(i)\}$  for each basepair-position  $i$  in  $S$ . We say that  $a_i$  is the set of alternate nucleotides at basepair-position  $i$ ,  $\hat{m}_k$  covers  $\hat{m}_k(i)$  if  $\hat{m}_k(i) \in a_i$ , and  $\hat{M}$  is a minimal set of covering mutations if it is a smallest set whose elements cover all elements of the set of altering nucleotides ( $\bigcup a_i$ ).

Next, we will show that the single-range deletion problem is NP-complete for an alphabet of size four and is hard to approximate within a log factor. The problem is clearly in NP. Hardness follows by a reduction from set cover. Given a set-cover instance  $(N, M)$ , we construct an instance of the single-range deletion problem. We begin by introducing a character sequence representation for the elements of  $M$  in a format similar to the one used in the single-range substitution reduction of the previous subsection.

Given a subset  $p \in M$ , represent  $p$  with a character sequence of length  $n + 1$ . For all integers  $i$  from 1 to  $n$ , if  $i \in p$ , append “\*” to  $s$ ; otherwise, append “-” to  $s$ . Conclude by appending “#” to  $s$ . Thus, given  $n = 5$ , the subset  $b = \{1, 3\}$  of  $N$  is represented as “\*-\*-#”; we call “-” a place holder; “\*” a filler; and “#” a terminator. We arbitrarily (say lexicographically) order the elements of  $M$  and refer to this ordering using the bijective function  $r : M \rightarrow \{1, \dots, m\}$  and its inverse  $r' : \{1, \dots, m\} \rightarrow M$ .

$$N = \{1, 2, 3, 4\} \quad M = \{\{1, 2\}, \{2, 3\}, \{3, 4\}\}$$

S:    ----#----#----#\*\*\*\*A\*\*\*\*A\*\*\*--#-\*\*-#----\*\*#  
       \*\*\*\*\*A\*\*\*\*A\*\*\*--#-\*\*-#----\*\*#

**FIG. 2.** An example for the construction of a character-alphabet single-run-deletion problem instance from a set-cover problem-instance  $(N, M)$ . A set of alternating characters  $a_i$  is composed of the characters below the  $i^{\text{th}}$  position of  $S$ , given that alternating character sets are not multi-sets and may not include  $s(i)$ .

We construct the wildtype from left to right as follows. Let  $l$  be a sequence composed of  $n$  place holders (-) terminated by the terminator (#), and  $h$  be a sequence composed of  $n$  fillers (\*) terminated by an A. The wildtype begins with  $m$  repetitions of  $l$ , followed by  $m - 1$  repetitions of  $h$ , followed by the representations of subsets  $r'(1)$  to  $r'(m)$ . The length of the wildtype is  $(3m - 1)(n + 1)$  characters.

We construct the profile from left to right as follows. Let  $t$  be a sequence composed of  $n$  fillers (\*) terminated by the terminator (#), and  $g$  be a sequence composed of  $n$  fillers (\*) terminated by an A. The profile (starting at  $a_1$ ) begins with  $t$ , followed by  $m - 1$  repetitions of  $g$ , followed by the representations of subsets  $r'(1)$  to  $r'(m)$ . An example of the complete construction is given in Fig. 2.

**Lemma 2.** *Only mutations resulting from deletions of the closed interval  $[x, y]$ , where  $x \leq n$  and  $y > (2m - 1)(n + 1)$  can be used to cover elements from alternating sets  $a_1$  to  $a_n$ .*

**Proof.** A deletion of the closed interval  $[x, y]$  where  $x > n$  will cover none of the first  $n$  elements of the profile. If  $y \leq (2m - 1)(n + 1)$ , then either  $x > n$  or  $y \leq m(n + 1)$ ; in both cases the resulting mutation will have only the place holder in positions  $[1, n]$ . ■

**Lemma 3.** *A minimal set of covering mutations includes exactly one mutation constructed by a deletion of the closed interval  $[x, y]$  where  $y \leq (2m - 1)(n + 1)$ .*

**Proof.** At least one such mutation is necessary in order to cover the “A”s in the profile. One such mutation is sufficient to cover all profile elements at positions greater than  $n$  including all “A”s. Such a mutation cannot cover alternating-set elements in index range  $[1, n]$ . Therefore, a minimal set of covering mutations will have exactly one such mutation. ■

**Theorem 4.** *The single-range deletion problem is NP-complete for an alphabet of size four and is hard to approximate within a log factor.*

**Proof.** A solution to the constructed single-range deletion instance is composed of a single deletion of the closed interval  $[x_0, y_0]$ , where  $y_0 \leq (2m - 1)(n + 1)$ ; and a set of deletions of the intervals  $[x_i, y_i]$ , where  $x_i \leq n$  and  $y_i > (2m - 1)(n + 1)$ , which correspond to a choice of a minimal set of covering mutations for the original set-cover problem. A deletion of the interval  $[x_i, y_i]$  corresponds to the subset whose representation includes the wildtype character at position  $y_i + 1$ . ■

Next, we will show that the single-range deletion problem is NP-complete for an alphabet of size three and is hard to approximate within a log factor. The problem is clearly in NP. Hardness follows by a reduction from the single-range deletion problem with alphabet of size four. Encode each character of the four-alphabet problem using a unary sequence of length five: “-” as “10002”; “\*” as “01002”; “#” as “00102”; and “A” as “00012.” A solution for the three-alphabet problem directly translates to a solution for the four-alphabet problem, as shown in the Theorem 5.

**Theorem 5.** *The single-range deletion problem is NP-complete for an alphabet of size three and is hard to approximate within a log factor.*

**Proof.** Only deletions of length  $5k$  are allowed since all encodings are of length five and terminate with the character “2.” Furthermore, all possible deletions can be mapped to deletions that start at the

beginning of a four-alphabet character representation and end at the end of a four-alphabet character representation. A deletion that starts a position  $j$  of a representation will be equivalent to a deletion that starts at the beginning of that representation, or at the beginning of the next representation because of the unary property of character representations. So, only deletions of the interval  $[5k + 1, 5l]$  need to be considered since a deletion of the interval  $[5k + j + 1, 5l + j]$ ,  $j \in \{0, \dots, 3\}$  can be read as either a deletion of the interval  $[5k + 1, 5l]$  or as a deletion of the interval  $[5(k + 1) + 1, 5(l + 1)]$ . Thus, a mutation in the minimal-mutation set is always composed of complete unary encodings. The set consisting of the encodings reversals of the mutations in the minimal-mutation set is the minimal-mutation set for the encoded instance of the single-range deletion problem with alphabet of size four. ■

2.3. Insertion sets

The case of insertion mutations is similar to that of deletion mutations, since the leftmost insertion masks the shift of all insertions further right. Thus, the single base insertion problem can be solved by including the insertion mutation defining the leftmost alternative character to the wildtype. Each remaining uncovered alternative character must be covered by a distinct insertion of the prescribed character. This yields a minimum covering if the union is consistent with the profile; otherwise, no such covering exists. An extension of this algorithm can be used to find the minimal covering of any realizable profile with insertions of exactly length  $k$  in polynomial time.

**Theorem 6.** *The mutation deconvolution problem for  $k$ -length insertions can be solved in  $O(nk)$  time.*

In the single-range insertion of maximum size  $k$  problem, each variation differs from  $S$  in the insertion of a single contiguous subsequence whose length is less than or equal to  $k$ . Thus, there are  $O(n\alpha^k)$  possible variations in  $V$ .

Next, we show that the general single-range insertion problem is NP-complete. The problem is clearly in NP. Hardness follows by a reduction from set cover. As in previous reductions, we construct an instance of the single-range insertion problem from a set-cover instance  $(N, M)$  using a character-alphabet of size four. See Fig. 3.

Use a character sequence  $s$  of length  $n + 1$  to represent each subset  $p \in M$  as follows: begin with  $s = \text{"\#"};$  then for all integers  $i$  from 1 to  $n$ , if  $i \in p$ , append  $\text{"*"};$  otherwise, append  $\text{"-"};$  to  $s$ . We refer to  $\text{"\#"}$  as an initiator,  $\text{"*"};$  as a filler, and  $\text{"-"};$  as a place holder. We arbitrarily (say, lexicographically) order the elements of  $M$  and refer to this ordering using the bijective function  $r : M \rightarrow \{1, \dots, m\}$  and its inverse  $r' : \{1, \dots, m\} \rightarrow M$ .

We construct the wildtype from left to right as follows. Let  $l$  be a sequence composed of the initiator ( $\text{"\#"}$ ) followed by  $n$  place holders ( $\text{"-"};$ ). The position occupied by these final  $n$  place holders is important to the reduction, so we refer to these positions as the “termination region.” Let  $h$  be the representations of subsets  $r'(1)$  to  $r'(m)$  followed by  $l$ . The wildtype consists of  $m$  repetitions of  $h$  and is of length  $m(m + 1)(n + 1)$ .

We construct the profile, which has characters up to position  $(m + 1)^2(n + 1)$ , as follows. First, for each character in the wildtype, insert its complement into the profile:  $\text{"\#"}$  becomes  $\text{"A"}$  (and vice versa) and  $\text{"*"};$  becomes  $\text{"-"};$  (and vice versa). Then, insert a copy of  $h$  and its complement into the profile starting at position  $m(m + 1)(n + 1)$  (i.e., off the right edge of the profile). Finally, insert  $\text{"A"}$  into the position below the termination regions in each repetition of  $l$  (i.e., characters in the half-open intervals  $(m(n + 1), (m + 1)(n + 1)], ((2m + 1)(n + 1), 2(m + 1)(n + 1)],$  and so on).

$$N = \{1, 2, 3, 4\} \quad M = \{\{1, 2\}, \{2, 3\}, \{3, 4\}\}$$

S:  $\text{\#---\#-**\#---**\#---\#---\#-**\#---**\#---\#---\#-**\#---**\#---}$   
 $\text{\#---\#-**\#---**\#---}$   
 $\text{A---A*---A*---A***A---A*---A*---A***A---A*---A*---A***A---A*---A*---A***}$   
AAAA
AAAA
AAAA
AAAA

**FIG. 3.** An example for the construction of a character-alphabet single-run-insertion problem from a set-cover problem-instance  $(N, M)$ . See text for details.

**Lemma 4.** *A minimal set of covering mutations includes exactly  $m + 1$  mutations constructed by insertions of length  $(m + 1)(n + 1)$ .*

**Proof.** An “A” at position  $i$  must be covered by an insertion of an “A” at that position. The maximum length insertion is  $(m + 1)(n + 1)$ , and the termination regions are all  $(m + 1)(n + 1)$  apart from each other; therefore,  $m + 1$  mutations are required in order to cover all “A”s in the profile. This number of legal mutations is sufficient to cover all profile elements other than the fillers (\*) or the “A”s that appear under the place holders at the end of each repetition of  $l$ . ■

**Theorem 7.** *The single-range insertion problem of maximum size  $k$  is NP-complete for  $|\Sigma| = 4$ .*

**Proof.** Let  $y_i$  designate the  $i$ th termination region, i.e., profile positions  $[i(m + 1) + m](n + 1) + 1$  to  $(i + 1)(m + 1)(n + 1)$ . Then, for  $i$  from 0 to  $m$ , the elements of  $y_i$  include both a filler (\*) and an “A.” Covering all the “A”s in the termination regions will require  $m + 1$  mutations. The remaining fillers can be covered with  $m + 1$  additional mutations, but fillers can also be covered using the wildtype representations of the set-cover subsets. A mutation that is generated by an insertion of length  $(n + 1) * (j - 1)$  starting at a position at most  $(n + 1) * (j - 1)$  corresponds to using the subset  $r'(j) \in M$  in the solution to a set-cover problem. If the set-cover instance has a solution  $\hat{M}$  of  $k$  subsets, then the corresponding mutation set will cover all fillers in the termination region. Thus, the minimal set of covering mutations will have  $m + 1$  mutations that cover all profile elements other than the fillers in the termination region and a set of  $k$  mutations corresponding to  $\hat{M}$  to cover the remaining fillers. In the case that there is no solution to the set-cover instance,  $2m + 2$  mutations are required; otherwise,  $m + k + 1$  will suffice. ■

The reduction can be rewritten using an alphabet of size three using the same encoding as described for deletion mutations.

**Theorem 8.** *The single-range insertion of maximum size  $k$  problem is NP-complete for  $|\Sigma| = 3$ .*

### 3. ALGORITHMS AND IMPLEMENTATION

In this section, we describe the algorithms we implemented for each of the three problems described in Section 1.1. Our experiments with this implementation are reported in Section 4.

#### 3.1. Base calling

The base-calling problem on actual gels is complicated by a variety of technology-dependent issues. The contribution of this paper is in higher-order determination of sequences from mixed populations, so we assume that the sequencing machine returns an observed frequency of each base at each position.

Here, we describe how we performed base calling in our simulation. Given the observed frequency  $F(i, j)$  for base  $i$  at a given position  $j$ , we classify it based on thresholds  $t_{lo}$  and  $t_{hi}$ :

$$C(i, j) = \begin{cases} \text{Present} & F(i, j) > t_{hi} \\ \text{NoCall} & t_{lo} \leq F(i, j) \leq t_{hi} \\ \text{Absent} & F(i, j) < t_{lo} \end{cases}$$

Each element  $v$  of the set of candidate mutations  $V$  can be easily tested to see whether it is consistent with the set of called bases. A mutation  $v$  is considered consistent if for all  $i$ ,  $C(i, v(i)) \neq \text{Absent}$ . In other words, a mutation is consistent if each of its basepairs correspond to a Present or NoCall entry.

#### 3.2. Mutation deconvolution

We solve the mutation deconvolution problem as a variant of the set cover problem. Given a profile  $C$ , we find a minimal set of consistent mutations such that for every  $C(i, j) = \text{Present}$ , there is some  $v \in V'$  such that  $v(i) = j$ .

Instead of using heuristics to approximately solve set cover, we explored the power of exhaustive search, specifically a DFS A\* search. As each Present value in the called-base array is accounted for, it is marked Covered. A *score* is defined that evaluates a given mutation and returns the number of Present values that it will convert to Covered if it is added to the solution set.

Mutations are added to the solution in decreasing order of their score. An aggressive pruning scheme is used for early termination. Assume the best solution so far covers the profile with  $p$  mutations and in the current solution there are already  $q$  mutations that leave  $r$  Present values to be covered. If  $score(v)$  is below  $r/(p - q)$ , then  $v$  cannot be the next largest mutation in an optimal solution, and the search tree can be pruned. Furthermore (assuming mutations are added in decreasing order), once a mutation has been rejected at level  $q$ , it can be rejected for that entire subtree.

Because of the order in which nodes are visited, the first leaf of the search tree that is visited is the greedy solution to the problem (i.e., keep choosing the node with the highest score until all nodes have zero score). The greedy algorithm is guaranteed to find a solution (but not necessarily an optimal one) if the input measurement  $C$  is consistent with the set of mutations  $M$ .

### 3.3. Population frequency determination

The third stage of the algorithm takes the solution of the set cover problem and the observed frequency matrix  $F$ . From this data, it creates a system of up to  $4|s|$  linear equations (where  $|s|$  is the length of the wildtype). The linear equation for base  $i$ , position  $j$ , is

$$\sum_{k=1}^{|M|} w_k c(i, j, k) = F(i, j) \tag{1}$$

where

$$c(i, j, k) = \begin{cases} 1 & \text{if } m_k(i) = j \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

Many of these equations will be degenerate, i.e.,  $c(i, j, k) = 0$  for all mutations. Specifically, if  $C(i, j) = \text{Absent}$ , the equation is guaranteed to be degenerate, and if  $C(i, j) = \text{NoCall}$ , the equation may or may not be degenerate. All degenerate equations are discarded from the solution.

The resulting equations will be an overconstrained system of linear equations which can be written as  $A\vec{w} = \vec{y}$  and solved by the left-multiplying the pseudo-inverse  $A^* = A'(AA')^{-1}$  on both sides of the equation. The resulting weight vector  $w$  contains the weights that minimize the squared error.

## 4. EXPERIMENTAL RESULTS

We performed a series of computational experiments to evaluate how accurately multiple mutations can be identified in a mixed population subject to experimental error and to measure the performance of our algorithms. Below, we discuss the sources of our test data and our results.

### 4.1. Test data

To accurately simulate a real-world diagnostic application, the library of mutations we used in our study was derived from a large database of p53 mutations known to cause cancers in humans. We used test data from the World Health Organization’s International Agency for Research on Cancer, Lyon, France, specifically version R5 (June 2001) of Hernandez-Boussard *et al.* (1999), which appears to be the largest p53 database available.

Table 1 summarizes the number of mutations in the database on each of the 11 exons of p53. Substitution, Insertion, and Deletion mutations are as described in section 1.1. Complex mutations are mutations not in the above categories, for example, removing three bases at a particular location and substituting two bases in their place. We limited our experiments to exon 4 mutations, since this is the largest exon with any significant number of mutations and is in principle the most challenging computationally. There

TABLE 1. A SUMMARY OF THE MUTATIONS CATALOGED IN THE P53 DATABASE<sup>a</sup>

	Exon number										
	1	2	3	4	5	6	7	8	9	10	11
<b>Distinct mutations</b>											
Substitution mutations	0	9	2	<b>167</b>	459	242	210	308	56	39	4
Insertion mutations	0	0	0	<b>22</b>	70	33	33	45	6	3	2
Deletion mutations	0	1	3	<b>76</b>	182	86	76	123	21	13	4
Complex mutations	0	0	0	<b>8</b>	19	15	11	13	1	0	0
Total mutations	0	10	5	<b>273</b>	730	376	330	489	84	55	10
<b>Reported mutations</b>											
Substitution reported	0	12	2	<b>313</b>	3896	1636	3661	3305	96	101	5
Insertion reported	0	0	0	<b>25</b>	125	54	80	64	12	3	2
Deletion reported	0	1	3	<b>109</b>	405	223	258	248	23	22	4
Complex reported	0	0	0	<b>8</b>	19	15	11	13	1	0	0
Total reported	0	13	5	<b>455</b>	4445	1928	4010	3630	132	126	11
Length	528	102	22	<b>279</b>	184	113	110	137	74	107	1288

<sup>a</sup>Hernandez-Boussard *et al.* (1999). Note that many mutations in the database are actually duplicates, so the number of distinct mutations is actually much lower than the number of entries in the database.

are 167 distinct substitutions, 22 distinct insertions, and 76 distinct deletion mutations to exon 4 in this database.

The database entries for the insert mutations contained only information about where it occurred and how long the insertion sequence was, but not the inserted sequence itself. So each insertion mutation in the database was modified by inserting a random sequence of the correct length at the correct location (i.e., each insert mutation had a sequence chosen at random, but those sequences were fixed and known before the problem was started). The vast majority of insert/delete mutations are short (length  $\leq 5$ ), although the longest reported deletion has length 278. Fig. 4 shows the length distribution of insertions and deletions in the database.

Random profiles were generated and corrupted by different amounts of noise. For each noise level considered, 50 random problems were created, each from six mutations chosen at random. From these  $m = 6$  mutation problems,  $k = \{1, \dots, 5\}$  problems were generated by selecting the first  $k$  mutations. For a noise level  $u$  and an  $m$  mutation problem, the weight of each mutation was chosen to be between  $u/2$  and  $0.6/m$ . These limits were chosen to insure that the wildtype was still a strong signal in the data, but that the individual mutations would be not be completely masked by the background noise. The base-frequency matrix  $F$  is then generated by tabulating which sequences contribute to which basepairs and adding uniform noise centered around the expected frequency.

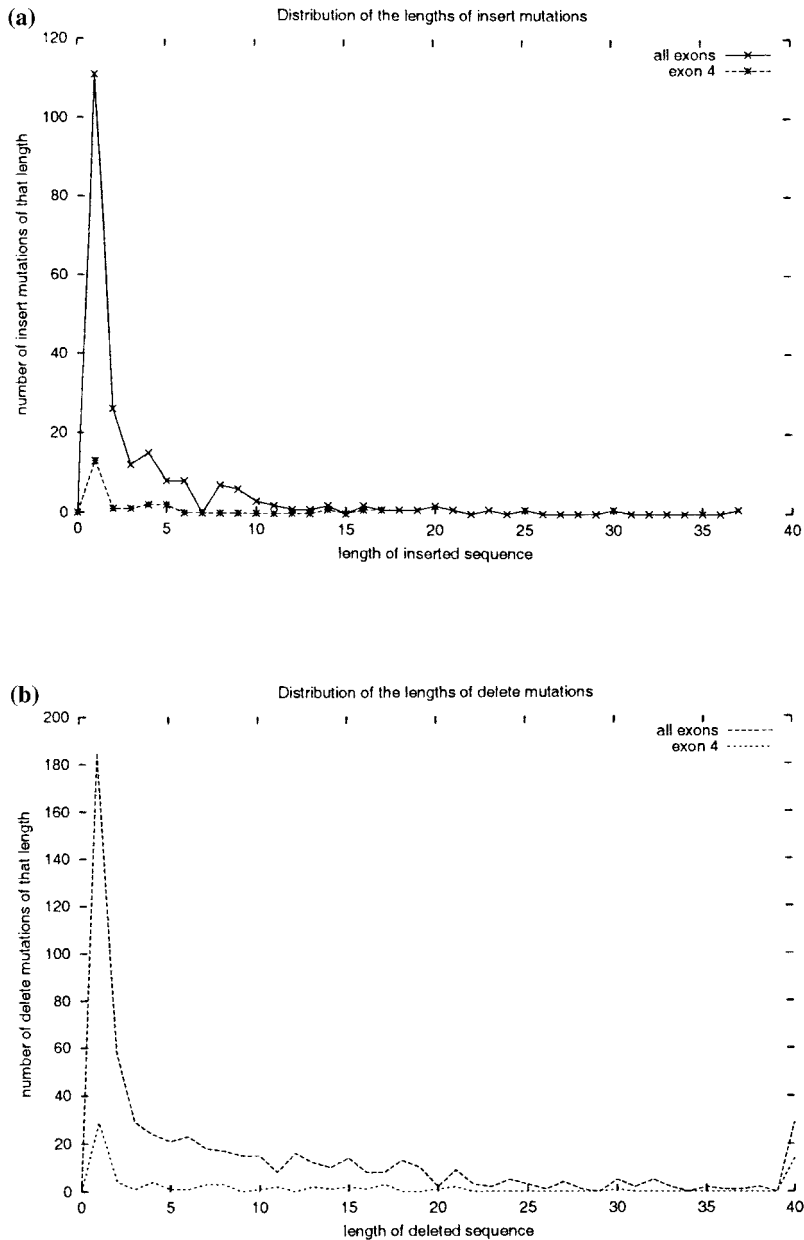
## 4.2. Results

We ran extensive simulations on reconstructing from one to six mutations, subject to noise levels from 1% to 30% relative error. Each data point is represented by 50 solved problems.

Our system seeks to explain the observed data in terms of the fewest mutations possible. There may be multiple solutions of optimal cardinality, complicating the interpretation of the quality of our results. We summarize our results in the four graphs which follow.

**Correct mutations identified:** Figure 5(a) reports the likelihood that a mutation in the correct answer was found by at least one of the “optimal” solutions found by the program. As an example, assume a profile was generated by Del(10,3), Del(15,5), and Sub(10,A) and that our algorithm found two optimal solutions: the correct one and Del(12,3), Del(15,5), Sub(11,C). In this case, the Del(15,5) is found 100% of the time, while the other two are found 50% of the time, giving a score of 66% for the solution.

These results demonstrate that we can identify at least 90% of present mutations even subject to a data error rate of 5% with up to six mutations.

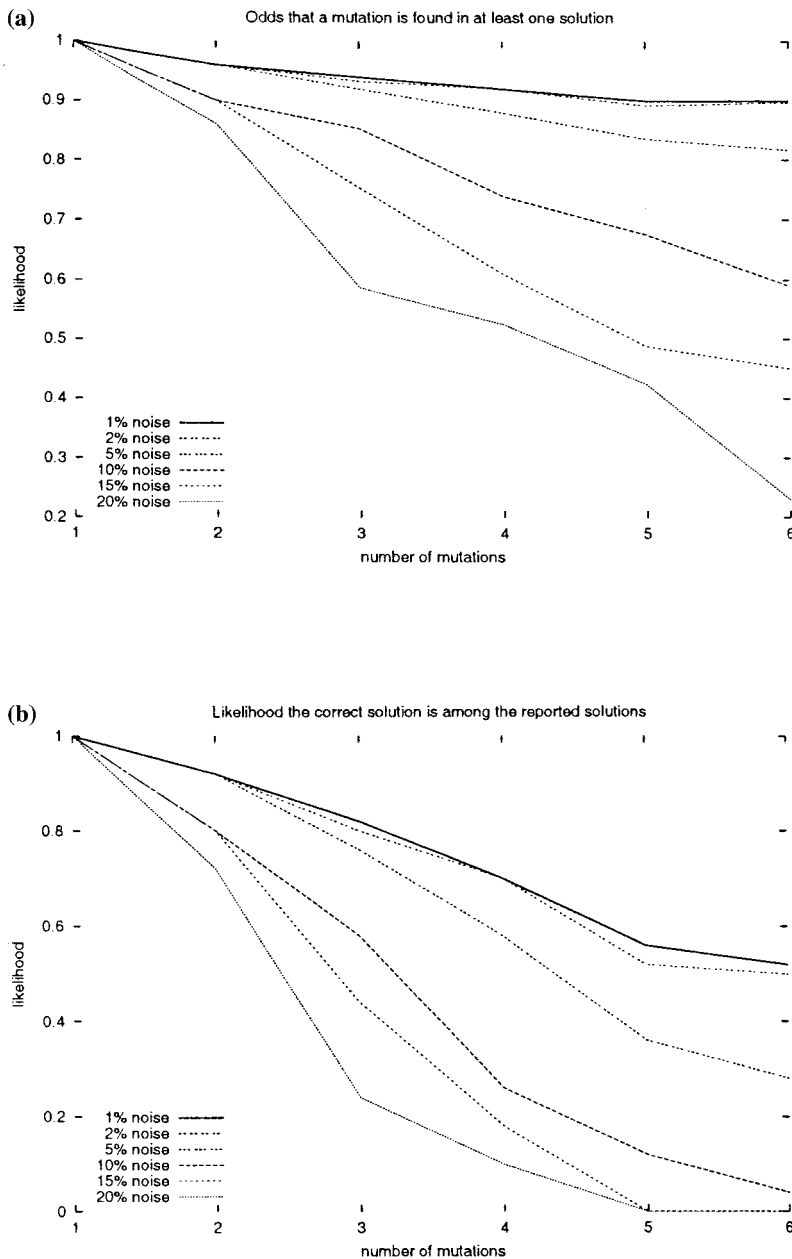


**FIG. 4.** The length distribution of p53 insertion (a) and deletion (b) mutations. The deletion graph is truncated, with the rightmost point counting all deletions of length 40 or greater. The largest reported deletion mutation is length 278.

**Correct mutation set identified:** A stronger condition requires that we identify the complete set of mutations present. Figure 5(b) returns the likelihood that an optimal solution found by the program is identical to the correct answer. In the previous example, the score would be only 50%, because the algorithm returned two solutions, but only one exactly matched the entire answer.

Our accuracy degrades with observed error and the number of mutations, but we can identify the full solution of up to four mutations roughly 60% of the time given an observed error of up to 5%.

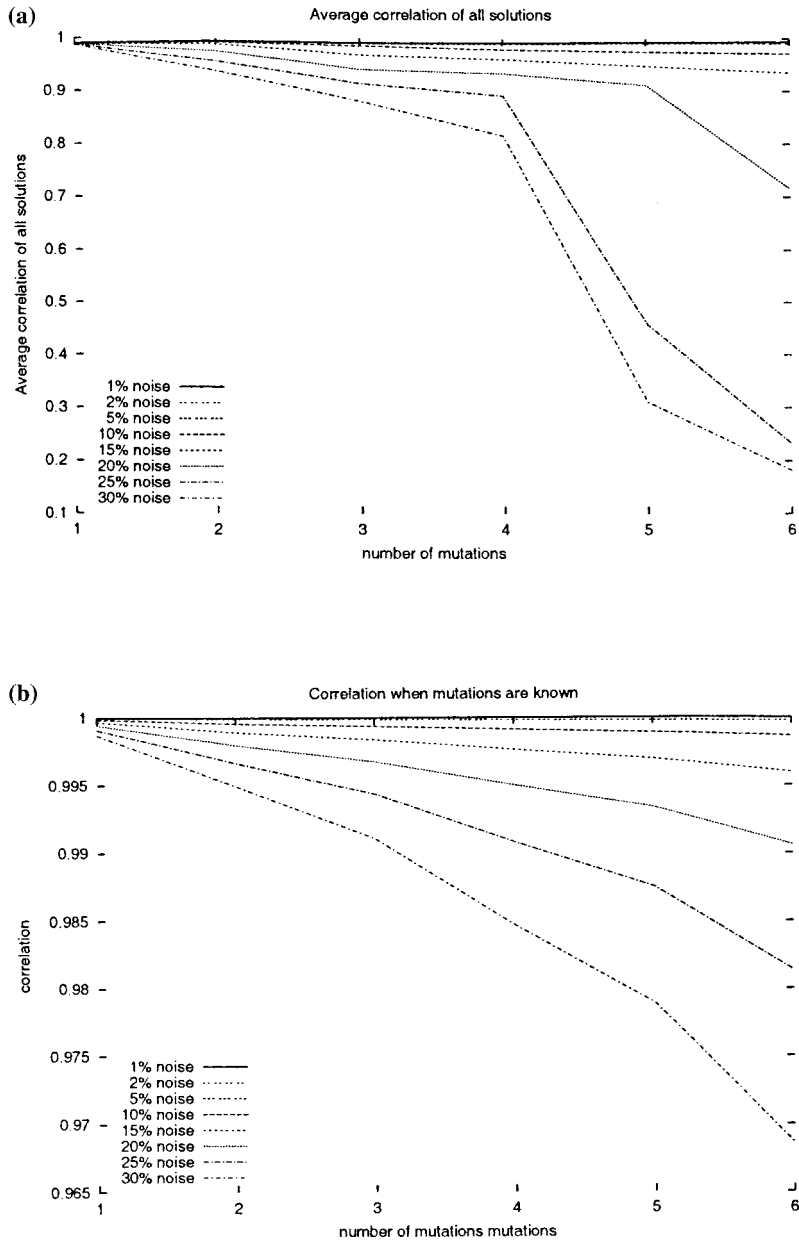
**Correlation with observed frequency:** Figure 6(a) measures the accuracy with which we reconstruct the weight of the mutations and wildtype. The weights of the correct answer and optimal solutions were converted to vectors, and the correlation of these vectors was computed as the cosine of the angle between them. When the system reported multiple optimal solutions, each was weighted equally.



**FIG. 5.** The likelihood that (a) a correct mutation and (b) all correct mutations are found by one of the reported solutions. For example, if the profile was generated by two mutations, and the algorithm returned a solution where one mutation was correct and one incorrect, graph (a) would give the solution a score of 50% since half the mutations were correct, while (b) would give it a score of 0% since the solution didn't match exactly. The noise introduced into the experiments varies between 1% and 20%.

We demonstrate a high correlation for up to four mutations, even subject to an observed error of up to 30%. For error rates of 5%, we can accurately reconstruct the distribution even with six mutations.

**Correlation with observed frequency when mutations are known:** Figure 6(b) measures the accuracy with which we reconstruct the weight of the mutations and wildtype if the set-cover problem returns the correct mutations. Correlations were computed as in the previous section. The correlation for all cases is very high, and when the noise is less than 0.1, the correlation is within 0.05% of the correct solution. Thus, we can conclude that most of the correlation error from the previous measurement is due to mistakes in identifying the mutations.



**FIG. 6.** Correlation of reconstructed and original population frequencies as mutation count and error rate, using (a) experimentally reconstructed mutations, and (b) the actual mutation set—note vertical axis scale. The introduced noise varies between 1% and 30%.

Of these four measurements, the most important is the first one, correct mutations identified. The ability to identify potentially important mutations, even with a significant number of false positives, is the most important characteristic of a diagnostic test. By comparing the “correct mutation” and “correct solution” graphs, it is possible to estimate the likelihood of the system producing a false positive. The correlation measurement is probably the least significant one in a pure diagnostic setting, although it is perhaps the best measure of success for viral or acquired mutation population studies.

4.3. Running times

Table 2 summarizes the running time of the algorithm on various different mutation counts and noise levels. For one to four mutations at all noise levels, the time to create and solve a system is under one

TABLE 2. ALGORITHM RUNNING TIMES<sup>a</sup>

Noise	4 mutations			5 mutations			6 mutations		
	Min	Ave	Max	Min	Ave	Max	Min	Ave	Max
1%	.273	.436	1.508	.271	5.590	102.563	.312	56.448	841.046
2%	.252	.429	1.638	.265	5.464	102.730	.259	53.694	821.966
5%	.265	.545	7.261	.267	4.713	103.134	.258	35.231	809.013
10%	.251	.318	.651	.256	1.401	50.040	.248	.707	8.052
15%	.246	.303	.987	.242	.356	1.253	.248	.374	1.880
20%	.239	.285	.457	.239	.325	1.191	.234	.520	8.184
25%	.250	.290	.430	.237	.677	18.667	.239	.792	16.866
30%	.244	.472	8.893	.239	.459	8.835	.240	.354	3.071

<sup>a</sup>Running times for analysis algorithms are presented as a function of error rate and mutation count. Times are in seconds on a PIII-1GHz computer running the Java JDK 1.3 (hotspot) and Linux.

second on average, and seven seconds worst case, which is essentially constant. For five and six mutations, the problem becomes much more difficult.

A surprising result is that problems with the lower noise levels seem to be more difficult to solve than those with higher noise levels. This is because there are simpler solutions for the higher noise levels; i.e., there are three-mutation solutions to high-noise six-mutation problems, so once a smaller solution is found, the number of nodes to explore in the tree decreases exponentially.

## 5. SNP GENERATION AND ANALYSIS

A single-nucleotide polymorphism (SNP) is a mutation that differs from a wildtype by a single substitution in one base position. An important current problem in genomics is cataloging all SNPs in a given population. As discussed in Section 2.1, it is easy to detect SNPs when sequencing samples sequentially. Here, we show how to employ a pooling strategy to detect SNPs in multiple individuals through a single sequencing run. This results in a significant increase in sequencing throughput when compared with sequencing each individual separately.

In our pooling strategy, we combine equal amounts of DNA from each of the  $n$  distinct individuals and amplify the region of interest using PCR. We then sequence the resulting mixture on a frequency-sensitive machine. Our analysis problem is determining the conditions under which the peak resulting from an SNP is distinguishable from the background noise of the sensor.

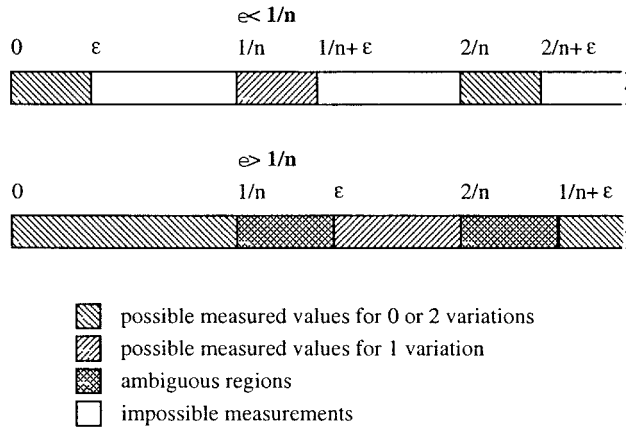
### 5.1. Analysis

If we assume that at most one SNP occurs per base location, we can look at the odds that such a mutation will be detected. We assume that the noise of our detector is uniform noise  $U(0, \epsilon)$  and the number  $m$  of individuals in the mix is known.

A SNP can be detected in two different ways: by a larger than normal measurement off the wildtype ( $v$ ) or by a smaller than normal measurement on the wildtype ( $w$ ). If  $v > \epsilon$ , then the only way that such a reading could occur is if there is a SNP. If we assume the total weight of the mixture is normalized to 1.0, a SNP in one individual will have a measurement of  $1/n$  before the sensor error is added and weight  $1/n \leq v \leq 1/n + \epsilon$  afterwards. If  $1/n > \epsilon$ , the measurements from SNPs will be measurably different from those resulting from noise, and all SNPs will be detected correctly.

In the case that  $1/n < \epsilon$ , there is a chance that a SNP will have a measurement that is less than  $\epsilon$ , just as there is a chance that the background noise will exceed  $1/n$ . The odds of such an occurrence are

$$P(v < \epsilon) = \frac{\epsilon - 1/n}{\epsilon} = 1 - \frac{1}{\epsilon n}. \quad (3)$$



**FIG. 7.** A graphical representation of the effect of noise on observed measurements in the model. While  $\epsilon < 1/n$  there is no noise to obscure SNPs. Once  $\epsilon > 1/n$ , there are some measurements that are ambiguous; i.e., it’s not possible to deconvolve them with certainty.

However, even if  $v < \epsilon$ , it still may be possible to detect the presence of a mutation because of the lower  $w$  value. Specifically,  $w = 1.0 - 1/n + U(0, \epsilon)$ , so

$$P(w > 1) = \frac{\epsilon - 1/n}{\epsilon} = 1 - \frac{1}{\epsilon n}. \tag{4}$$

This means that the odds of missing a SNP completely is

$$P(v < \epsilon) \cdot P(w > 1) = \left(1 - \frac{1}{\epsilon n}\right)^2 = 1 - \frac{2}{\epsilon n} + \frac{1}{\epsilon^2 n^2}. \tag{5}$$

However,  $w$  being too small isn’t enough information to identify which of the three off-wildtype SNPs has occurred.

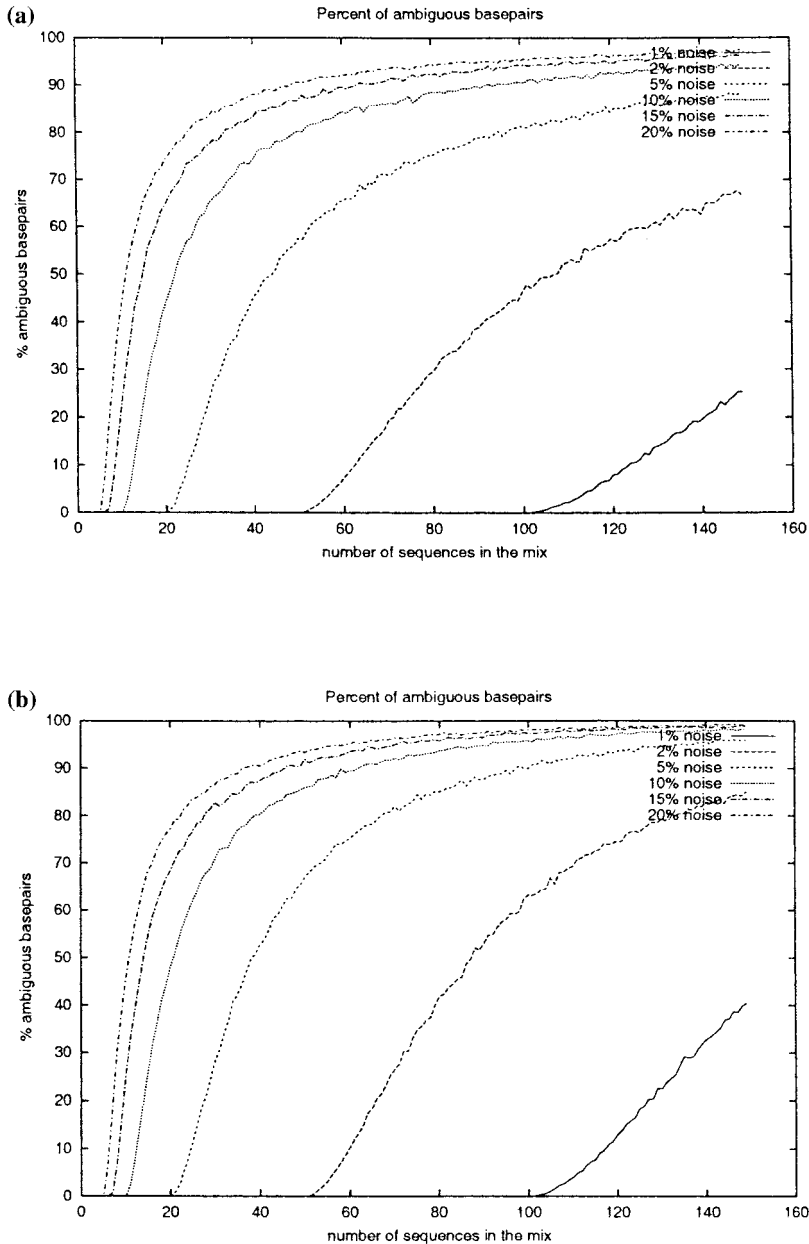
The process for detecting multiple SNPs in a single basepair is basically the same as detecting a single SNP. An elevated  $v$  value means a SNP is present, as does a depressed  $w$  value. However, a depressed  $w$  value will be less likely to disambiguate the results. As shown in Fig. 7, a system with two or more SNPs is more likely to be ambiguous because the noise is more likely to interfere than with a single SNP.

### 5.2. Simulation results

We performed a set of experiments to confirm the mathematical analysis, as well try to understand how the system behaves in unusual cases, such as multiple strains of data having the same mutation.

For the simulations below, all candidates considered were single-base mutations. A mixture of  $m$  different strains was created where strains were likely to be mutated with both probability 0.01 and 0.001. Noise drawn from  $U(0, \epsilon)$  was added to the measurements to simulate sensor error. The measurement was then checked to see if it was consistent with more than one set of mutations. In that case, the measurement was declared “ambiguous.” For example, an ambiguous measurement might be one where the original mixture contained one a–c mutation and two a–g mutations, but the measurement was also consistent with one a–c and three a–g mutations.

Figure 8 shows that even in the case of large amounts of sensor error, it is possible to reliably and simultaneously detect SNP variations in multiple sequences. When the sensor noise is reduced, it may be possible to test more than 100 sequences while detecting all SNP mutations.



**FIG. 8.** The likelihood that a mixture of  $m$  sequences under sensor error  $\epsilon$  will result in a measurement that could be ambiguously interpreted, (a) for SNP frequencies of 1/1,000 bases, and (b) for SNP frequencies of 1/100 bases. So, for example, with 1% noise and 120 samples in the mix, about 8% of the runs will come back with an ambiguous result when the underlying SNP frequency is 1/1,000 bases. But if it increases to 1/100, then 12% of the runs will be ambiguous.

## 6. FUTURE WORK

The most obvious future work is measuring how well the set cover and SNP algorithms work on real data instead of synthetic data. Probably the most difficult part of this will be working with a more realistic noise model. Uniform noise models are good for analysis, but are probably not representative of the noise in real systems. While we expect our techniques to work for other noise models, they have yet to be tested. Also, the current algorithm is intolerant of complete misclassification in the base-calling stage (i.e., “Present” instead of “Absent” or vice-versa). Extending the algorithm to handle a moderate

number of misclassifications should increase its robustness with real data. It should also be possible to adapt the algorithm to detect the likelihood that a given mutation was in a particular profile at a specific concentration, which could be useful for clinical situations.

All of the analysis so far has been on simple or simultaneous mutations. Extending this work to handle composite mutations should be fairly straightforward; a sequence containing a composite of two mutations can most times be covered by two simple mutations and vice versa. But since composite mutations tend to have a higher score than simple mutations, the current system will be highly biased towards choosing composite mutations whenever it can. The most obvious way to address this properly would be to introduce prior likelihoods for all the allowable mutations and have the search procedure maximize the likelihood instead of minimizing the number of mutations. This would have other benefits as well, assuming that prior information can be obtained.

One research area that is gaining importance is the detection of haplotypes instead of SNPs. Haplotypes are a combination of alleles of closely linked loci that are found in a single chromosome, tend to be inherited together, and in some cases can be used to determine genetic traits. Current research suggests that haplotypes may be more important than SNPs in determining genetic predisposition and suggests creating a map of all human haplotypes (Daly *et al.*, 2001).

The techniques in Section 5 do not maintain any correlation information between SNPs, so it is not possible to directly obtain haplotypes in the same way. Also, all the results are based on single runs of a sequencer. Since the loci for haplotypes is typically larger than the run length of a sequencer (10k+ basepairs versus 1k basepairs), it is often not possible to get information about a haplotype from a single run. To deal with this, the haplotyping community has developed many techniques to detect haplotypes without having to do multiple-run sequencing (Hawley and Kidd, 1995; Excoffier and Slatkin, 1995; Tishkoff *et al.*, 2000). It is not clear whether Section 5 could be used in conjunction with these techniques.

However, a modification of our method can be used to identify multiple SNPs occurring in the same individual. By combining differing amounts of DNA from each individual, it may be possible to use the frequency output of the machine to correlate SNPs. However, since the noise of this output is high, it would mean that only smaller pools can be simultaneously sequenced. A second alternative would be to create multiple, redundant pools of DNA so that each piece of DNA is sequenced more than once and correlate the output between to determine likely haplotypes. However, this would also reduce the gains in throughput and relies on the assumption that SNPs are very rare, which is not always the case.

## ACKNOWLEDGMENT

We thank Dr. Vera Gorfinkel of BioPhotonics Corp. for introducing us to this problem and for useful discussions.

## REFERENCES

- Ben-Dor, A., Bruhn, L., Friedman, N., Nachmann, I., Schummer, M., and Yakhini, Z. 2000. Tissue classification with gene expression profiles. *Proc. 4th Ann. Int. Conf. on Computational Molecular Biology (RECOMB 00)*, 54–64.
- Beroud, C., and Soussi, T. 1998. p53 gene mutation: Software and database. *Nucl. Acids Res.* 26, 200–204.
- Daly, M., Rioux, J., Schaffner, S., Hudson, T., and Lander, E. 2001. High-resolution haplotype structure in the human genome. *Nature Genet.* 29, 229–232.
- Denisov, G., Ho, D., Mettler, M., Candlin, J., and Hunkapiller, T. 2000. Tracetuner—next generation base calling. [www.paracel.com](http://www.paracel.com).
- DeRisi, J., Iyer, V., and Brown, P. 1997. Exploring the metabolic and genetic control of gene expression on a genomic scale. *Science* 278, 680–686.
- Ewing, B., Hillier, L., Wendl, M., and Green, P. 1998. Base-calling of automated sequencer traces using phred. i. accuracy assessment. *Genome Res.* 8, 175–185.
- Excoffier, L., and Slatkin, M. 1995. Maximum-likelihood estimation of molecular haplotype frequencies in a diploid population. *Mol. Biol. Evol.* 12, 921–927.
- Fodor, S., Read, J., Pirrung, M., Stryer, L., Lu, A., and Solas, D. 1991. Light-directed, spatially addressable parallel chemical synthesis. *Science* 251, 767–773.

- Golub, T.R., Slonim, D.K., Tamayo, P., Huard, C., Caasenbeek, M., Mesirov, J.P., Coller, H., Loh, M.L., Downing, J.R., Caligiuri, M.A., Bloomfield, C.D., and Lander, E.S. 1999. Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring. *Science* 286, 531–537.
- Gorfinkel, V., and Luryi, S. 1998. Method and apparatus for identifying fluorophores. US Patent 5784157.
- Group, I.S.M.W. 2001. A map of human genome sequence variation containing 1.4 million snps. *Nature* 409, 928–933.
- Hawley, M.E., and Kidd, K.K. 1995. Haplo: A program using the em algorithm to estimate the frequencies of multi-site haplotypes. *J. Hered.* 86, 409–411.
- Hernandez-Boussard, T., Rodriguez-Tome, P., Montesano, R., and Hainaut, P. 1999. Iarc p53 mutation database: A relational database to compile and analyze p53 mutations in human tumors and cell lines. *Human Mutat.* 14(1), 1–8.
- Johnson, D. 1974. Approximation algorithms for combinatorial problems. *J. Computer and System Sciences* 9, 256–278.
- Lander, E., Linton, L.M., Birren, B., Nusbaum, C., Zody, M.C., Baldwin, J., *et al.* 2001. Initial sequencing and analysis of the human genome. *Nature* 409, 860–921.
- Tang, R., Wang, P., Wang, H., Wang, J., and Hsieh, L. 2001. Mutations of p53 gene in human colorectal cancer: Distinct frameshifts among populations. *Int. J. Cancer* 91, 863–868.
- Tishkoff, S.A., Pakstis, A.J., Ruano, G., and Kidd, K.K. 2000. The accuracy of statistical methods for estimation of haplotype frequencies: An example from the cd4 locus. *Am. J. Human Genet.* 67, 518–522.
- Venter, J.C., Adams, M.D., Myers, E.W., Li, P.W., Mural, R.J., Sutton, G.G., *et al.* 2001. The sequence of the human genome. *Science* 291, 1304–1351.
- Walther, D., Bartha, G., and Morris, M. 2001. Basecalling with lifetrace. *Genome Res.* 11, 875–888.

Address correspondence to:  
Andy Wildenberg  
2/91 Alexandra Avenue  
South Yarra, 3141 Australia

E-mail: wildenberg@wehi.edu.au