2012 IEEE 26th International Parallel and Distributed Processing Symposium Workshops & PhD Forum

Parallel Pair-HMM SNP Detection

Nathan L Clement*, Brent A Shepherd‡, Paul Bodily†, Sukhbat Tumur-Ochir†, Younghoon Gim†, Quinn Snell†, Mark J Clement†, and W Evan Johnson§
*Department of Computer Science
University of Texas at Austin, Austin, TX 78701
Email: nclement@cs.utexas.edu
†Department of Computer Science
Brigham Young University, Provo, UT 84602.
‡Department of Statistics
Brigham Young University, Provo, UT 84602.
§Department of Medicine
Boston University, Boston, MA 02118

***Abstract*—**

I. Motivation:

Due to the massive amounts of data generated from each instrument run, next generation sequencing technologies have presented researchers with unique analytical challenges which require innovative, computationally efficient statistical solutions. Here we present a parallel implementation of a probabilistic Pair-Hidden Markov Model for base calling and SNP detection in next generation sequencing data. Our approach incorporates multiple sources of error into the base calling procedure which leads to more accurate results. In addition, our approach applies a likelihood ratio test that provides researchers with straightforward SNP calling cutoffs based on a p-value cutoff or a false discovery control.

II. Results:

We have developed GNUMAP-SNP, which is a highly accurate approach for the identification of SNPs in next generation sequencing data. By utilizing a novel probabilistic Pair-Hidden Markov Model, GNUMAP-SNP effectively accounts for uncertainty in the read calls as well as read mapping in an unbiased fashion. Our results show that GNUMAP-SNP has both high sensitivity and high specificity throughout the genome, which is especially true in repeat regions or in areas with low read coverage. In addition, we propose a statistical framework that accounts for the background noise using straightforward statistical cutoffs which filters out false-positive results. The parallel implementation of SNP calling achieves near linear speedup on distributed memory or shared memory platforms.

III. Availability:

GNUMAP-SNP is available as a module in the GNUMAP probabilistic read mapping software. GNUMAP is freely available for download at: http://dna.cs.byu.edu/gnumap/

***Index Terms*—next-generation sequencing; short-read mapping; sequence mappers; parallel computing; biology computing**

IV. Introduction

Over the past 30 years, Sanger-type sequencing [1] has been the standard technique for DNA sequencing. This approach has enabled, among other things, the sequencing of the first complete human genome sequence [2]. However, recent developments in sequencing technologies have led to a second generation of sequencing approaches, from Illumina/Solexa, ABI/SOLiD, 454/Roche, Pacific Biosciences, which currently produce gigabases sequence information during each instrument run. These technologies are being applied in diverse ways to address genome-wide questions and generate massive datasets under various experimental conditions. Next-generation sequencing has accelerated the usefulness and accessibility of DNA sequencing data leading to higher volume, lower resolution data at a much lower cost, which has already resulted in novel biological insights and discovery.

Next generation sequencing technologies have been promoted as a means to revolutionize the field of biomedical research by overcoming many longstanding limitations of previous genomic approaches such as Sanger sequencing and microarrays. However, the size of the datasets generated are much larger and the diverse nature of the dataset has produced novel statistical and computational challenges that must be overcome. One very important problem facing researchers today is the identification and characterization of single nucleotide polymorphisms (SNPs). Researchers are often interested in identifying SNPs that vary between one individual and a reference genome, or in comparing the sequence composition of two individuals, strains or species at homologous or orthologous regions. Researchers are most often interested in associating these SNPs with disease or important phenotypes of interest (for example, see [3]), so the accurate identification of these SNPs is extremely important.

Compared to next generation technologies, Sanger sequencing is characterized by lower throughput but higher quality base calls. Next generation approaches leverage a substantially higher volume of data at the cost of lower quality reads and base calls. Due to the size and complex nature of many eukaryotic genomes, the increased sequencing volume is often used to identify features in a larger number of genomic regions, and therefore SNPs must often be called from as few as 5-20 overlapping reads. However if the quality of the reads is low or if the region contains repetitive elements, the accurate identification of SNPs can be a difficult task and requires sophisticated and powerful computational approaches.

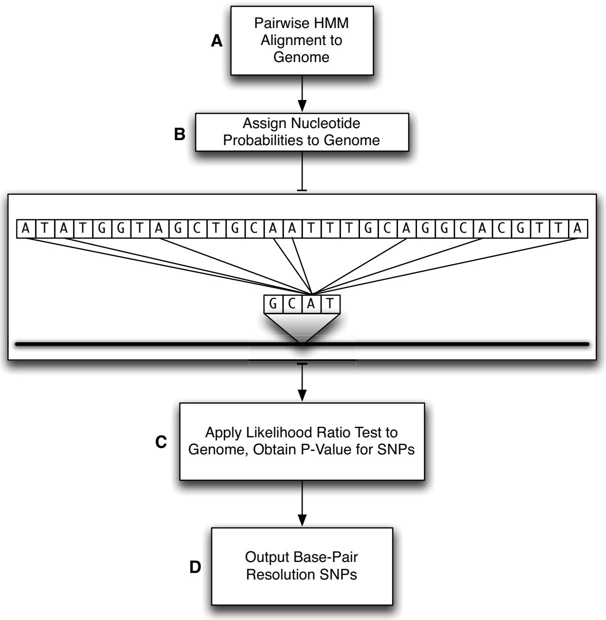

A
Pairwise HMM Alignment to Genome
B
Assign Nucleotide Probabilities to Genome
A T A T G G T A G C T G C A A T T T G C A G G C A C G T T A
G C A T
C
Apply Likelihood Ratio Test to Genome, Obtain P-Value for SNPs
D
Output Base-Pair Resolution SNPs

Fig. 1. **General GNUMAP approach.** In step (A) above, positive matching locations are identified using a PHMM. In step (B), the results of the PHMM are used to assign nucleotide probabilities to each location in the matching genomic sequence. After all of the sequences have been aligned, step (C) will use a likelihood ratio test to determine the probability that a given nucleotide is different in the reference genome. If the p-value passes a specified cutoff, step (D) will print this location to a file.

There are several existing approaches for calling bases and identifying SNPs in next generation sequencing data. We will only show comparisons with MAQ [4], because other methods have similar results (such as [5] and [6]). Although these methods are based on sound statistical approaches, their algorithmic implementations rely on *ad hoc* cutoffs and lack comparisons with background noise, leading to many false-positive results. In addition, these methods remove reads with low-quality bases and either remove or randomly assign reads that map to multiple locations which may lead to biased results. Finally, these methods rely on the single, although most plausible, mapping of the read to the genome and therefore do not use all the available information in the data expressed by other highly plausible local alignments which provide greater information and uncertainty in SNP calls.

In this research, we present GNUMAP-SNP, which utilizes a probabilistic Pair-Hidden Markov Model (PHMM) for base calling and SNP detection in next generation sequencing data. This novel methodology incorporates base uncertainty based on the quality scores from the sequencing run, as well as mapping uncertainty from multiple optimal and sub-optimal alignments of the read to a given location to the genome. This leads to more unbiased and more accurate SNP calls, as well as more accurate measures of the variance of each feature. In addition, our approach applies a likelihood ratio test that provides researchers with straight-forward SNP calling cutoffs based on a p-value cutoffs or a false discovery controls. This feature is not included in other existing SNP calling approaches. This framework makes GNUMAP-SNP highly sensitive and specific and more appropriate for the noisy data generated by current next generation sequencing platforms. Parallel performance results indicate that GNUMAP-SNP achieves nearly linear scalability on distributed memory and/or shared memory platforms. The GNUMAP-SNP software is freely available for download at: http://dna.cs.byu.edu/gnumap/.

V. Approach

GNUMAP-SNP applies a three-step approach for identifying SNPs in next-generation sequencing data. The first step is to create a genomic hash table of k-mers (default k=10), and then reference k-mers in the reads into this hash for efficient identification of putative mapping regions. Next, GNUMAP-SNP utilizes a probabilistic PHMM for a marginal alignment which allows for effective and accurate base calling and SNP detection. The third step is to apply a likelihood ratio test for SNP calling based on a p-value cutoffs or a false discovery controls. The first step is thoroughly discussed by [7], and a detailed justification of steps two and three is given below. Figure 1 contains a work-flow cartoon for the GNUMAP-SNP algorithm.

A. PHMMs for sequence alignment

PHMMs are a common alternative for sequence alignment to the standard Needleman-Wunsch Algorithm [8]. They have been applied in many different scenarios in biomedical research such as gene finding [9], and sequence alignment [10].

In order to apply a PHMM to align two sequences, researchers assume a data generating process by which two DNA sequences, $\mathbf{x}$ and $\mathbf{y}$, are generated simultaneously as an aligned pair. This process assumes a sequence of latent or *hidden* state variables, consisting of match and gap states, which have a Markov Chain correlation structure and emit the aligned elements of $\mathbf{x}$ and $\mathbf{y}$. However, only the sequences are observed whereas the alignment (i.e. the hidden state sequence) must be inferred in the analysis. Figure 2 provides a visual representation of this process in relation to a pair-wise sequence alignment.

Formally, we can assume a Hidden Markov process with three states, denoted M, G_X, and G_Y, where M is the *match state* which generates nucleotides that are paired in the alignment, and G_X and G_Y represent *gap states* where either an $\mathbf{x}$ and $\mathbf{y}$ nucleotide is generated and aligned with a gap. Letting x_i and y_j be the ith and jth elements of $\mathbf{x}$ and $\mathbf{y}$, respectively, we assume that in a match state, the probability of generating an $x_i y_j$ nucleotide pair is given by $p_{x_i y_j}$, where higher probability is associated with pairs where $x_i = y_j$. In the gap states, nucleotides are generated and matched with the gaps with probability q_{x_i} and q_{y_i} (usually $q_{x_i} = q_{y_i} = q$). Finally we assume that T_{ab} represents the Markov Chain transition probability of moving from state a to state b for $a, b \in \{M, G_X, G_Y\}$. Using this PHMM approach, the expected state sequence can be estimated using a forward-backward dynamic programming algorithm.

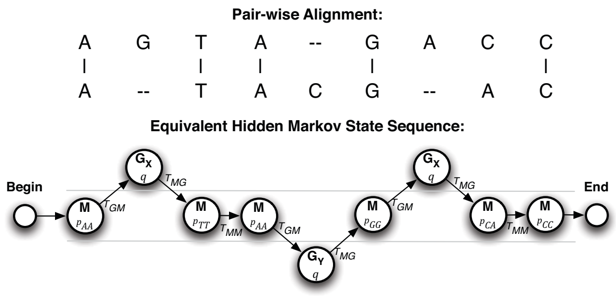


Fig. 2. Illustration of using a PHMM for aligning two sequences. We assume a PHMM with match state M, gap states G_X and G_Y, emission probabilities $p_{x_i y_j}$ and q, and transition probabilities T_{MM}, T_{MG}, T_{GM}, and T_{GG}.

B. Probabilistic extension of PHMMs for next generation sequencing

PHMMs are very advantageous for SNP detection in next generation sequencing. In addition to producing an optimal, or most plausible alignment given the data, they can also yield a marginal alignment across all possible alignments. In the case of next-generation sequencing applications, this implies that we can obtain the marginal probability that a specific base in the genome is aligned to a A,C,G, or T nucleotide in the read across *all high scoring alignments*. This is particularly useful in cases where there are multiple optimal or sub-optimal alignments for a read to the genome. Existing approaches use only a single possible alignment based on a fixed decision rule, whereas a PHMM can marginalize across all alignments and thus is not forced to choose between high-scoring alignments, potentially leading to less biased results than methods based on a single alignment.

In addition, GNUMAP-SNP employs a novel extension of PHMMs for sequencing data with quality scores. To account for uncertainty in the reads due to sequencing and mapping errors, we adapt the PHMM process described above to emit a vector of probabilities or quality scores associated with each base. Equivalently, we can assume that the emissions for each read base follow a continuous negative multinomial distribution. With this extension of the PHMM, we are able to effectively integrate the base quality scores into the PHMM marginal alignment, thus relying on higher quality bases when aligning the read to the genome and leading to more accurate alignments.

C. Likelihood Ratio Test for significance

GNUMAP-SNP applies a likelihood ratio test (LRT) to call the bases from the mapped reads and to identify SNPs compared to the reference genome. Specifically, for each base location in the genome, we obtain the marginal base contributions for all the reads that align to the location of interest. Denote these as $\mathbf{z} = (z_A, z_C, z_G, z_T, z_{gap})$. For example, suppose that there are 20 reads that map to the genomic base of interest. Now suppose that 14, 1, 3, and 2 of the reads align an A, C, G, and T to the base, respectively, and that no reads align a gap to the base. Then in this case $\mathbf{z} = (14, 1, 3, 2, 0)$. We assume that $\mathbf{z}$ follows a continuous negative multinomial distribution with read base proportions p_A, p_C, p_G, p_T, and p_{gap}.

LRT for monoploid genomes: Using the assumptions GNUMAP-SNP applies a LRT to identify bases above background in the reads. Namely, let $p_{(5)}, p_{(4)}, p_{(3)}, p_{(2)}$, and $p_{(1)}$ be the ordered values of p_A, p_C, p_G, p_T, and p_{gap}. In monoploid genomes we wish to test the null and alternative hypotheses:

$$\begin{array}{ll} H_0: & p_{(5)} = p_{(4)} = p_{(3)} = p_{(2)} = p_{(1)} \\ H_1: & p_{(5)} > p_{(4)} = p_{(3)} = p_{(2)} = p_{(1)}, \end{array} \qquad (1)$$

or in other words that the highest read base proportion is statistically larger than the others (background) and therefore there is enough information to call the base at this location. The base from the read can then be compared with the genome at this location and it can be determined if the base is a SNP. This comparison with the background provides the user with a statistical comparison with the background and allows users to control for false positive SNPs using a stringent p-value cutoff or an FDR control method. Such a comparison is a novel contribution of GNUMAP-SNP, as a comparison with the background is not included in other SNP calling approaches (e.g. MAQ, SOAP-SNP).

LRT for diploid genomes: In the diploid genomes, we test the hypotheses consist of the following H_0 and H_1:

$$\begin{array}{ll} H_0: & p_{(5)} = p_{(4)} = p_{(3)} = p_{(2)} = p_{(1)} \\ H_1: & p_{(5)} > p_{(4)} = p_{(3)} = p_{(2)} = p_{(1)} \text{ or} \\ & p_{(5)} = p_{(4)} > p_{(3)} = p_{(2)} = p_{(1)}, \end{array} \quad (2)$$

where the first alternative hypothesis implies a single significant base is present, suggesting that both alleles in the diploid genome are the same base (homozygous). The second alternative hypothesis suggests that the allele is heterozygous at that particular location.

VI. Methods

A. General Implementation

We have added the SNP calling with a PHMM functionality into GNUMAP, a previously-developed program used to probabilistically align next-generation sequencing reads to a reference genome [7]. In our approach, an array of floats representing the entire genomic sequence is stored in the program's memory, with space allocated for each nucleotide. A read is aligned to the genome with the probabilistic PHMM. The resulting alignment is a set of probabilities identifying how likely it is for each base in the read to align to a genomic position. As each read is aligned to the genome, probabilities are summed to obtain a complete alignment. When the program is finished, the genome is post-processed, and a LRT is applied to determine the significance at each base. If the p-value obtained from the LRT is significant and differs from a comparison genome, the base is identified as a potential SNP. A more complete description of the method is given below.

Step 1: Genome Storage and MPI Implementation

At each location in the genome, five floating-point values are stored, representing the likelihood of each character (including *n*'s) from a given read at that base. Since this requires nearly five times the memory for a single run, we have updated the program to use MPI, an interface that allows GNUMAP to be run simultaneously on several different machines, thus splitting the memory requirements (see [11] for a more complete description of the parallel improvements applied to GNUMAP).

MPI (Message Passing Interface) is a method for communicating between several different machines. On a single computer, all the communication is done by sharing memory locations; when more than one machine is used (either to obtain more processors or because insufficient memory is available), messages must be sent to synchronize the program running on each machine. GNUMAP utilizes both the expanded memory and the multiple processors available with many machines to dramatically decrease the running time.

If the genome is small enough to fit on a single computer, each machine will process the entire genome, then map a different portion of the reads.[1] At the end of the run, each of the machines will communicate the state of their genome and SNPs will be called accordingly. In this way, using four machines will decrease the runtime by approximately one fourth.

The second use of MPI takes advantage of the increased memory available across several machines, but is slightly more difficult. First, the genome is split into equal segments and distributed across the participating machines so no one machine performs more work than any other. In order to find the normalized posterior probability score for each read at a given location, GNUMAP must find all locations throughout the entire genome to which a given read aligns (see [7] for a complete description of the posterior probability scoring method). Communication between machines via message passing determines this additional locations and calculates the final score. In this manner, each read is scored accurately and with significantly reduced memory requirements.

Further methods to decrease memory requirements are discussed in Section VI-B.

Step 2: Align Reads to the Genome

In order to obtain the greatest accuracy on the output, the probability from each nucleotide obtained from base quality scores is used to create a position-weight matrix (PWM) for each read. Using this PWM in conjunction with the PHMM, we are able to determine the *probability* that any nucleotide in the read matches to a specific location in the genome. In other words, in the resulting alignment matrix, position i, j gives the probability that the character in the read at position i matches the genome at position j. Summing the rows of the matrix gives a total probability for each character at each genomic location. As more reads are matched to the genome, these probabilities are summed, giving a complete alignment (see Figure 3).

We use an approach and notation similar to that used by [12]. Let $\mathbf{x}$ denote a read to be mapped to the genome and $\mathbf{y}$ denote a portion of the genome that is a candidate region for mapping $\mathbf{x}$. Let x_i be the ith element of $\mathbf{x}$ and let $\mathbf{x}_{[a:b]}$ be the subsequence of $\mathbf{x}$ that starts at nucleotide a and ends on nucleotide b. Similarly define y_j and $\mathbf{y}_{[c:d]}$. The marginal probability that x_i and y_j are matched in the alignment, denoted $x_i \diamond y_j$, is given by

$$P(x_i \diamond y_j | \mathbf{x}, \mathbf{y}) \propto f_M(i,j) b_M(i,j), \quad (3)$$

where $f_M(i,j)$ is the *forward probability*, which is the conditional probability that $x_i \diamond y_j$ given all possible alignments of the subsequences $\mathbf{x}_{[1:i]}$ and $\mathbf{y}_{[1:j]}$, and where $b_M(i,j)$ is the *backward probability*, which is the conditional likelihood of observing the subsequences $\mathbf{x}_{[i+1:N]}$ and $\mathbf{y}_{[j+1:M]}$ given that $x_i \diamond y_j$.

[1]With a mer-size of 10 (default), the 150Mb human X chromosome requires approximately 12GB of RAM. The entire human genome requires nearly 90GB RAM on a single machine; split among 30 machines (a realistic number in many cluster-computing environments), each machine uses just over 4GB RAM.

In addition, the marginal probabilities that x_i or y_j are aligned to a gap, denoted $x_i \diamond G_j$ (x_i aligned to a gap between y_{j-1} and y_j) and $y_j \diamond G_i$ respectively, are given by

$$\begin{array}{l} P(x_i \diamond G_j|\mathbf{x},\mathbf{y}) \propto f_{G_X}(i,j)b_{G_X}(i,j), \quad \text{and} \\ P(y_j \diamond G_i|\mathbf{x},\mathbf{y}) \propto f_{G_Y}(i,j)b_{G_Y}(i,j), \end{array} \qquad (4)$$

where $f_{G_X}(i,j)$, $f_{G_Y}(i,j)$ are forward gap probabilities and $b_{G_X}(i,j)$, $b_{G_Y}(i,j)$ are backward probabilities. In order to estimate the probabilities above, we use a forward-backward dynamic programming algorithm as given below.

Forward Algorithm: To estimate the forward probabilities, we first let $p_{x_iy_j}$, q and T_{ab} be as defined previously. Also, for a given read, let r_{ik} be the quality score for base $k \in \{A,C,G,T\}$ at position i, and assuming that $\mathbf{x}$ is the read and $\mathbf{y}$ is the genome, let

$$p^*(i,j) = r_{iA}p_{Ay_j} + r_{iC}p_{Cy_j} + r_{iG}p_{Gy_j} + r_{iT}p_{Ty_j}.$$

Define three $(N+1)\times(M+1)$ matrices, namely $\{f_M(i,j)\}$, $\{f_{G_X}(i,j)\}$, and $\{f_{G_Y}(i,j)\}$ for $i = 0\ldots,N$ and $j = 0\ldots,M$. Now apply the following algorithm:

1) Initialization: Set $f_M(0,0) = 1$, $f_{G_X}(0,0) = f_{G_Y}(0,0) = 0$. Also for $k \in \{M,G_X,G_Y\}$ set $f_k(i,0) = 0$ for $i = 1,\ldots,N$ and $f_k(0,j) = 0$ for $j = 1,\ldots,M$.
2) Recursion: For $i = 1,\ldots,N$ and $j = 1,\ldots,M$

$$\begin{aligned} f_M(i,j) &= p^*(i,j)[T_{MM}f_M(i-1,j-1) \\ &\quad +T_{MG}f_{G_X}(i-1,j) + T_{MG}f_{G_Y}(i,j-1)] \\ f_{G_X}(i,j) &= q[T_{MG}f_M(i-1,j) + T_{GG}f_{G_X}(i-1,j)] \\ f_{G_Y}(i,j) &= q[T_{MG}f_M(i,j-1) + T_{GG}f_{G_Y}(i,j-1)] \end{aligned}$$

Backward Algorithm: For the backward probabilities, we define three $(N+1)\times(M+1)$ matrices, namely $\{b_M(i,j)\}$, $\{b_{G_X}(i,j)\}$, and $\{b_{G_Y}(i,j)\}$ for $i = 1\ldots,N+1$ and $j = 1\ldots,M+1$. Set $p^*(i,M+1) = 0$ for $i = 1,\ldots,N$ and $p^*(N+1,j) = 0$ for $j = 1,\ldots,M$. We then apply the following algorithm:

1) Initialization: For $k \in \{M,G_X,G_Y\}$ set $f_k(i,M+1) = 0$ for $i = 1,\ldots,N+1$ and $f_k(N+1,j) = 0$ for $j = 1,\ldots,M+1$. Also set $b_M(N,M) = b_{G_X}(N,M) = b_{G_Y}(N,M) = 1$.
2) Recursion: For $i = N,\ldots,1$ and $j = M,\ldots,1$ except (N,M), let

$$\begin{aligned} b_M(i,j) &= p^*(i+1,j+1)T_{MM}b_M(i+1,j+1) \\ &\quad +qT_{MG}[b_X(i+1,j) + b_Y(i,j+1)] \\ b_{G_X}(i,j) &= p^*(i+1,j+1)T_{GM}b_M(i+1,j+1) \\ &\quad +qT_{GG}b_{G_X}(i+1,j) \\ b_{G_Y}(i,j) &= p^*(i+1,j+1)T_{GM}b_M(i+1,j+1) \\ &\quad +qT_{GG}b_{G_Y}(i,j+1) \end{aligned}$$

Calculating Marginal Alignment Probabilities: Once the forward and backward probabilities are obtained, alignment and gap probabilities can be calculated based on Equations 3-4. However, in the SNP and genome base calling, we are not particularly interested in the alignment. For each base position in the genome sequence, we want to calculate the probability that an A, C, G, or T, in the read is aligned to the genome position. Assuming that $\mathbf{x}$ is the read and $\mathbf{y}$ is the genome, for a given genomic position and for a read k of length N, we let $\mathbf{z}_k = (z_{kA}, z_{kC}, z_{kG}, z_{kT}, z_{k,gap})$ be the probabilities that read k aligns a given base to the genomic location. These marginal probabilities can be calculated for the fixed genomic position j as

$$z_{kA} = \frac{\sum_{\{i:x_i=A\}} P(x_i \diamond y_j|\mathbf{x},\mathbf{y})}{\sum_{i=1}^{N} P(x_i \diamond y_j|\mathbf{x},\mathbf{y}) + \sum_{i=1}^{N} P(x_i \diamond G_j|\mathbf{x},\mathbf{y})},$$

and similarly for z_{kC}, z_{kG}, z_{kT}, and $z_{k,gap}$. These probabilities are then summed across the reads to obtain $\mathbf{z} = (z_A, z_C, z_G, z_T, z_{gap})$ (where $z_A = \sum_k z_{kA}$) and are then used for base and SNP calling in the following step.

Step 3: Apply the LRT to Identify SNPs

We apply a likelihood ratio test (LRT) to each genomic location separately. We assume that $\mathbf{z}$ follows a continuous Negative Multinomial distribution with parameters p_A, p_C, p_G, p_T, and p_{gap} as previously described. In monoploid genomes we wish to test the null and alternative hypotheses given in Equation 1. Under H_0 the probabilities of each base should be equal to each other and therefore the maximum likelihood estimators of the ps are $\hat{p}_k = 0.2$ for all $k \in \{A,C,G,T,gap\}$. Under H_1, the maximum likelihood estimators are $\hat{p}_{(5)} = z_{(5)}/n$, where $n = z_A + z_C + z_G + z_T + z_{gap}$ and $\hat{p}_{(4)} = \hat{p}_{(3)} = \hat{p}_{(2)} = \hat{p}_{(1)} = (n - z_{(5)})/4n$. Now we obtain the the LRT statistic, $\lambda(\mathbf{z})$, given by

$$\lambda(\mathbf{z}) = \frac{0.2^n}{\hat{p}_{(5)}^{z_{(5)}}\hat{p}_{(4)}^{n-z_{(5)}}}.$$

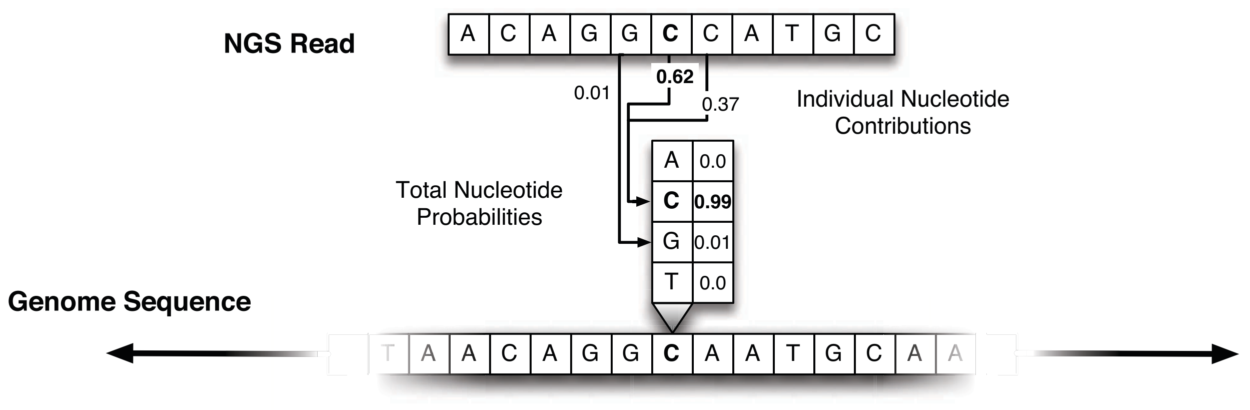
To obtain a p-value for this test, we appeal to the commonly used an asymptotic properties of LRTs, namely that

$$-2\log(\lambda(\mathbf{z})) \to \chi_1^2.$$

For our application, to obtain a test that preserves a SNP-wise false-positive rate of α, we compare $-2log(\lambda(\mathbf{z}))$ with the $(1-\alpha/5)th$ quantile of the χ_1^2 distribution, as test above actually violates the identifiability condition for the convergence of the LRT. However, this can be avoided by merely testing each base (A, C, G, T, gap) vs. background (5 tests) and adjusting the α for multiple testing.

For diploid genomes, we test the hypotheses given in Equation 2. Thus the maximum likelihood estimators of the ps are $\hat{p}_k = 0.2$ for all $k \in \{A,C,G,T,gap\}$ as in the previous case. Under the alternative, we observe that the maximum likelihood estimators are $\hat{p}_{(5)} = z_{(5)}/n$, and $\hat{p}_{(4)} = \hat{p}_{(3)} = \hat{p}_{(2)} = \hat{p}_{(1)} = (n - z_{(5)})/4n$, or $\tilde{p}_{(5)} = z_{(5)}/n$, $\tilde{p}_{(4)} = z_{(4)}/n$ and $\tilde{p}_{(3)} = \tilde{p}_{(2)} = \tilde{p}_{(1)} = (n - z_{(5)} - z_{(4)})/3n$. Therefore the LRT statistic is given by

$$\lambda(\mathbf{z}) = \frac{0.2^n}{\max\left(\hat{p}_{(5)}^{z_{(5)}}\hat{p}_{(4)}^{n-z_{(5)}}, \tilde{p}_{(5)}^{z_{(5)}}\tilde{p}_{(4)}^{z_{(4)}}\tilde{p}_{(3)}^{n-z_{(5)}-z_{(4)}}\right)}.$$



NGS Read
A C A G G C C A T G C
0.62
0.01
0.37
Individual Nucleotide Contributions
Total Nucleotide Probabilities
A 0.0
C 0.99
G 0.01
T 0.0
Genome Sequence
T A A C A G G C A A T G C A A

Fig. 3. **Nucleotide additions from Pair-HMM.** At each position in the genome, the probability of aligning all locations in the read with any of the four nucleotides is calculated ("Individual Nucleotide Contributions"), and the corresponding probabilities are recorded and summed ("Total Nucleotide Probabilities"). In this specific example, all the nucleotides in the next-generation sequencing read contribute a certain (if not insubstantial) probability to the conclusion that the genomic location contains a C, but only the closest nucleotides contribute a significant amount. Also, since the second C in the read could align well to the specified position, the probability contribution is significantly higher than the G to the left.

This test statsitic is then compared with a with the $(1-\alpha/5)th$ quantile of the χ_1^2 distribution as above.

B. Memory Distribution

A large portion of this work was dedicated to decreasing the memory requirements on distributed memory machines. Most important, this work sought to develop a version of GNUMAP with significantly decreased memory requirements and minimal loss to time and accuracy.

One of the unique aspects of GNUMAP is the ability to call SNPs *online*, instead of requiring several post-processing events. Online SNP calling in GNUMAP requires not only the basic objects (genome and hash table) to reside in main memory, but also floating point values for each base in the genome. This isn't an issue for a small genome, but for the entire human genome or larger polyploid organisms, this quickly becomes intractable. (When run on the 155Mbp human X-chromosome, GNUMAP only uses 5GB of RAM; when run on the entire 3Gbp human genome, over 100GB RAM is required.) As described above, there is a GNUMAP implementation that spreads the memory across several nodes, allowing genomes of larger sizes to be analyzed. However, as is shown in Figure 4, spreading the memory across multiple nodes is not as efficient as spreading the reads.

For this reason, we developed two additional methods to reduce the memory requirements. First, we designed an algorithm that only requires a single byte per nucleotide (instead of 4 bytes for floating-point precision). Second, we applied more extreme discretization based on practical nucleotide distributions (as described in [13]). Here we will describe the two methods.

1) Nucleotide-Byte Discretization: The memory required for the nucleotide-byte discretization is a single floating point value and single byte values for each nucleotide. The floating point value contains the total number of (possibly partial)

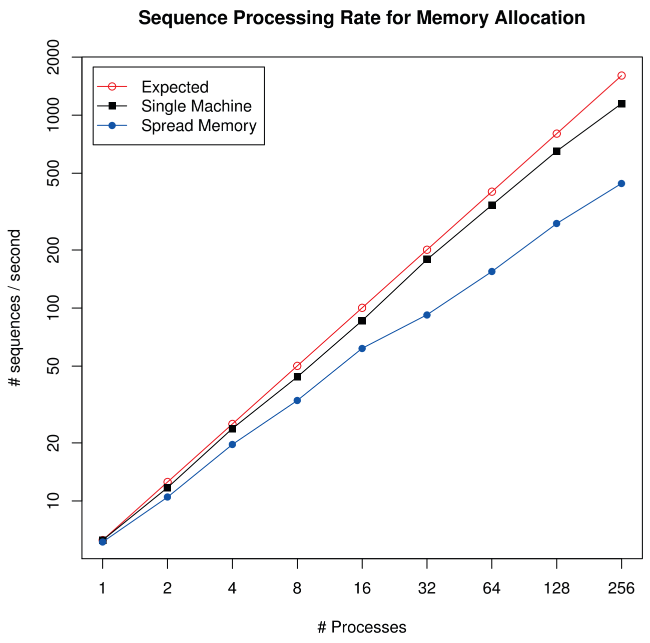


Fig. 4. Sequence processing times for two different MPI methods without any optimizations (higher is better). The solid red lines represents perfect linear performance, and the solid black line shows the performance of GNUMAP with all the genome in shared memory for every process. The solid blue line shows performance when only the memory is spread across nodes. Note that the spread memory mode does not process as many sequences, so the shared memory mode should be used when possible.

sequences that match to the given genomic position, and the single byte values contain the percentage present for each nucleotide. When updating a genomic position, G_i, with the probabilistic distribution from each nucleotide, $\delta = [0.9, 0.1, 0, 0, 0]$, each nucleotide-byte value is converted to real space (dividing by 128 and multiplying by the total score), and updated by the corresponding δ values. Each value is then converted back to nucleotide-byte space with the updated total score ($\sum_i \delta_i$).

One possible limitation to this method is speed: The normal method only requires a single addition when scoring a read at a given location; the nucleotide-byte method requires two additional multiplications and divisions. However, because the memory footprint is smaller, we can take advantage of locality caching to negate most of this loss.

The second limitation is what happens to the discretization in an online algorithm. As the total number of sequences assigned to a particular location increases beyond 255 (the maximum number of values in a byte), the amount changed at a single character becomes zero. For example, let T represent the total amount, and $\phi = [a, c, g, t, n]$ represent the single-byte values at each nucleotide. If T were 1 and there were only a single a, then the assigned discretization would be $\phi = [255, 0, 0, 0, 0]$. If there were one a and one t, then the discretization would be $\phi = [128, 0, 0, 127, 0]$. If there were 254 a's and a single t, $\phi = [254, 0, 0, 1, 0]$, adding an additional a would require either rounding up to $\phi = [255, 0, 0, 0, 0]$ or $\phi = [254, 0, 0, 1, 0]$. The former would remove any signal from t, and the latter would imply that any future addition of a would also have no effect.

While online discretization provides a serious problem when the total value is high, this is acceptable in most cases. The optimal sequencing depth (or coverage) for resequencing is anywhere from 10-40x [14], [15]. Since this is far below the maximum values for a single byte, the average genome position can expect to have values lower than this. Further, if a given location does exceed this threshold, based on the random placement of sequences we can assume that the first 255 occurrences are an accurate approximation of what will later occur. We will show in the results section that this is a valid assumption.

2) Centroid Discretization: Centroid discretization follows from the research of [13]. When using nucleotide-byte discretization, there are many character states that are not used (such as $\phi = [0, 0, 0, 0, 0]$, $\phi = [128, 128, 128, 128, 128]$, etc). In addition, there are many more states that are not biologically relevant. For example, SNP transitions (from a purine to a purine or pyrimidine to another pyrimidine) are far more likely than transversions (from a purine to a pyrimidine, etc). Creating a discrete sampling from the continuous possibilities takes both the practicality and biological relevance into account, sampling biologically-relevant states at a higher rate than those which are not as likely. For example, the single-byte discretization, γ, for a single a nucleotide would be represented as $\gamma = [0.84, 0.04, 0.04, 0.04, 0.04]$. A purine SNP from an a to a g might be represented as such: $\gamma = [0.28, 0.08, 0.48, 0.08, 0.08]$.

Because of the irregular sampling described above, converting from continuous values to the discretized γ either requires approximation or a somewhat exhaustive search to find the closest discrete neighbor. This leads to some of the same problems in the previous section, especially during the MPI reduction phase, where the discretized values from different processors are added together. This sum is not just an addition, but must be converted from and into the single-byte γ space before and after the actual sum is performed. Since there are only 256 discrete possibilities for γ, the sum can be a pre-computed table lookup, reducing the number of steps significantly.

VII. Performance Evaluation

A. Simulation Study for Accuracy

In order to validate GNUMAP's SNP caller, we created a data set with specific SNPs locations that we could verify. To make this more biologically relevant, these SNPs occur at locations that had been discovered in previous biological studies. The dbSNP file for build 37 of the human genome[2] contains roughly 800k SNPs, including insertions, deletions, and microsatellite repeats. From this file, we randomly selected 14,501 evenly-spaced SNPs from the X chromosome to occur in our simulated individual.

To create an individual with these mutations, we again used build 37 of the human X-chromosome (available from UCSC at http://hgdownload.cse.ucsc.edu, then go to the directory goldenPath/hg19/chromosomes), changing it at each location in the previously-created SNP file. We then used Metasim [16] to simulate 31M 62-bp reads with an error profile similar to that seen by the Solexa/Illumina platform. This produced 9.1Gb of sequencing data, which is almost a 12x coverage of the 153Mb X-chromosome.

We used this simulated data set to compare the accuracy of GNUMAP with MAQ [4], one of the leading mapping/SNP-calling programs. (We also made an attempt to use SOAPsnp [5], but were unable to produce any SNPs under several model conditions so we will not include this in our results) . As can be seen in Table I, GNUMAP's SNP caller is highly accurate on the experimental data, calling 75% of the total SNPs with only 6% false positives. These results are very similar to those achieved by MAQ.

TABLE I
Experimental results for simulated data

Program	Time (m)*	TP	FP	FN	Precision
MAQ	990.1	11322	830	3179	93.2%
GNUMAP-SNP	218.6	11070	676	3431	94.2%

Experimental results for simulated X-chromosome data in terms of true and false positives (TP and FP respectively) false negatives (FN), and precision (TP/(FP+TP)). There were a total of 14,501 SNPs inserted into the simulated genome. GNUMAP and MAQ performed similarly in terms of accuracy, identifying nearly 80% of the total SNPs present in the dataset. Note: The running times have not been normalized by the number of processors (to show absolute capabilities), and MAQ ran on 1 processor while GNUMAP utilized a cluster of 30 machines.

B. Performance Evaluation of Memory Optimizations

There are three criteria to test when evaluating the performance of the two memory reductions: speed, memory consumption, and accuracy.

[2]SNP file available from UCSC at http://hgdownload.cse.ucsc.edu/goldenPath/hg19/database/snp131.txt.gz

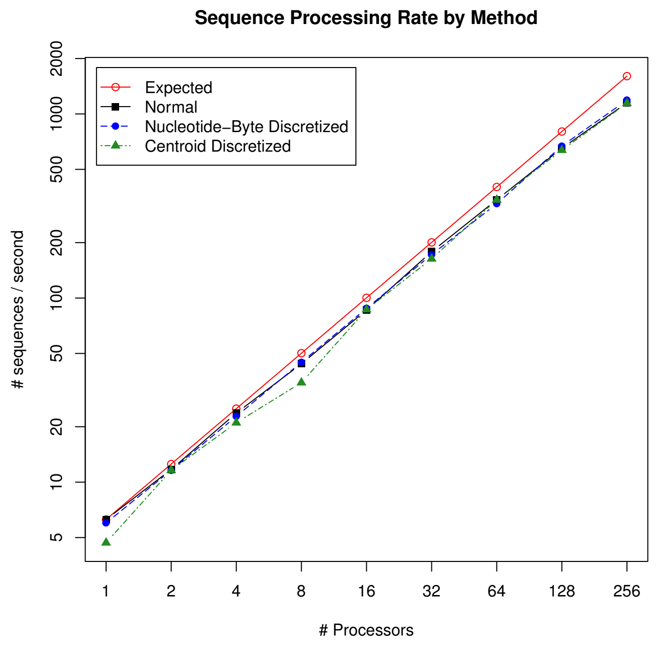


Fig. 5. Number of sequences processed/second (calculated by total runtime) for different number of processors, more is better. The solid red line represents perfect linear speedup, and the solid black line shows the running time for the normal MPI run without discretization. Speeds are nearly the same across all optimizations, with centroid discretization performing slightly worse.

TABLE II
MEMORY USAGE FOR OPTIMIZATIONS

optimization	chrX	human
NORM	4.76g	100g
CHARDISC	2.58g	58g
CENTDISC	2.91g	40g

Virtual memory used for normal (NORM), nucleotide-byte (CHARDISC) and centroid discretization (CENTDISC) for the human X chromosome (155Mbp) and entire human genome (3.1Gbp). As was expected, the most significant reduction is seen in the centroid discretization method.

1) Speed: As can be seen in Figure 5, all three methods perform at about the same rate, with the centroid discretization method performing slightly worse on a few instances.

2) Memory Footprint: Table II shows the virtual memory required. As was expected, the centroid discretization method performed the best, exhibiting the smallest memory footprint out of each of the methods. While 40GB is still too large for a desktop machine, this type of memory requirement is not uncommon for a typical cluster (for example, there are Amazon cloud instances with up to 68GB of RAM[3]). In addition, similar memory reductions can be see when the genome is spread across multiple nodes instead of entirely contained on a single node.

3) Accuracy: The biggest difference between these methods is the accuracy. Table III shows an overview of the speed and memory consumption in addition to accuracy for a single SNP calling run. There are two things to note. First, there was not a significant loss in accuracy when using the byte-per-nucleotide discretization method. In fact, in terms of precision, this method out-performed the original method. It only found about half as many true positives, but when resources are limited, this method seems to perform well.

[3]http://aws.amazon.com/ec2/instance-types/

TABLE III
MEMORY USAGE, WALL CLOCK, AND ACCURACY FOR GNUMAP WITH AND WITHOUT OPTIMIZATIONS

Optimization	MEM	WT	TP	FP	Precision
NORM	4.76GB	04:25:55	1309	127	91%
CHARDISC	2.58GB	04:36:58	677	0	100%
CENTDISC	2.01GB	04:27:29	166	9058	0.08%

Memory usage (MEM), wallclock times (WT), and accuracy (true and false positives (TP and FP, respectively) and precision: TP/(TP+FP)) for a single run of GNUMAP, without memory optimizations (NORM), discretization to a single byte per nucleotide (CHARDISC) and discretization with centroids to a single byte (CENTDISC). The genome used was human X chromosome, 155Mbp in length, and the reads were a subset of those used in the simulated accuracy analysis in Section VII-A. While all methods take about the same time to finish, the accuracy of the centroid discretized method is unacceptable.

The second point to note is that, while the centroid discretization method was able to keep up with the other methods in speed and used noticeably less memory, the accuracy was horrible. One reason for this significant drop in accuracy is that the discretization method employed is only meant to be used a few times. While the nucleotide-byte method was robust for coverage up to 255x without degrading performance significantly, the centroid method performs significant rounding approximations each time a new sequence is added to the genome. For this reason, while the memory footprint is significantly lower, this method is not be recommended for practical use.

VIII. CONCLUSIONS

In this research, we present GNUMAP-SNP, which utilizes a probabilistic Pair-Hidden Markov Model (PHMM) for base calling and SNP detection in next generation sequencing data. This novel methodology incorporates base uncertainty from the quality scores from the sequencing run, as well as mapping uncertainty from multiple optimal and sub-optimal alignments of the read to a given location to the genome. This leads to more unbiased and more accurate SNP calls, as well as more accurate measures of the variance of each feature. In addition, our approach applies a likelihood ratio test that provides researchers with straight-forward SNP calling cutoffs based on a p-value to provide false discovery controls. This feature is not included in other existing SNP calling approaches. This framework makes GNUMAP-SNP highly sensitive and specific and more appropriate for the noisy data generated by current next generation sequencing platforms.

The parallel implementation of SNP calling achieves near linear speedup. Although several methods for discretizing the probability values significantly reduce the memory footprint, these methods do not increase performance. In addition, the accuracy is not adversely affected by a mild discretization approach, so these methods could be used in environments where memory is limited and performance is not as critical.

Acknowledgement

We thank David Nix and Joshua Udall for helpful discussions regarding the content of this manuscript.

This work was supported by funds from NIH (R01 HG005692).

References

[1] F. Sanger, S. Nicklen, and A. R. Coulson, "Dna sequencing with chain-terminating inhibitors," *PNAS*, vol. 74, pp. 5463–5467, 1977.

[2] International Human Genome Sequencing Consortium, "Finishing the euchromatic sequence of the human genome." *Nature*, vol. 431, no. 7011, pp. 931–945, Oct. 2004. [Online]. Available: http://dx.doi.org/10.1038/nature03001

[3] A. F. Rope, K. Wang, R. Evjenth, K. Xing, J. J. Johnston, J. J. Swenson, W. Johnson, B. Moore, C. D. Huff, L. M. Bird, J. C. Carey, J. M. Opitz, C. A. Stevens, T. Jiang, C. Schank, H. D. Fain, R. Robison, B. Dalley, S. Chin, T. S. South, T. J. Pysher, L. B. Jorde, H. Hakonarson, J. R. Lillehaug, L. G. Biesecker, M. Yandell, T. Arnesen, and G. J. Lyon, "Using VAAST to Identify an X-Linked Disorder Resulting in Lethality in Male Infants Due to N-Terminal Acetyltransferase Deficiency," *Am J Hum Genet.*, vol. 89, pp. 28–43, 2011.

[4] H. Li, J. Ruan, and R. Durbin, "Mapping short DNA sequencing reads and calling variants using mapping quality scores," *Genome Res*, vol. 18, pp. 1851–1858, 2008.

[5] R. Li, Y. Li, X. Fang, H. Yang, J. Wang, K. Kristiansen, and J. Wang, "SNP detection for massively parallel whole-genome resequencing," *Genome Res*, vol. 19, pp. 1124–1132, 2009.

[6] B. Langmead, M. C. Schatz, J. Lin, M. Pop, and S. L. Salzberg, "Searching for SNPs with cloud computing," *Genome Biology*, vol. 10, p. R134, 2009.

[7] N. L. Clement, Q. Snell, M. J. Clement, P. C. Hollenhorst, J. Purwar, B. J. Graves, B. R. Cairns, and W. E. Johnson, "The GNUMAP algorithm: unbiased probabilistic mapping of oligonucleotides from next-generation sequencing," *Bioinformatics*, vol. 1, pp. 38–45, 2010.

[8] S. B. Needleman and C. D. Wunsch, "A general method applicable to the search for similarities in the amino acid sequence of two proteins," *J Mol Biol.*, vol. 48, pp. 443–454, 1970.

[9] L. Pachter, M. Alexandersson, and S. Cawley, "Applications of generalized pair hidden Markov models to alignment and gene finding problems." *J Comput Biol.*, vol. 9, pp. 389–399, 2002.

[10] B. D. Chuong, M. S. Mahabhashyam, M. Brudno, and S. Batzoglou, "ProbCons: Probabilistic consistency-based multiple sequence alignment," *Genome Res.*, vol. 15, pp. 330–340, 2005.

[11] N. L. Clement, M. J. Clement, Q. Snell, and W. E. Johnson, "Parallel mapping approaches for GNUMAP," *Parallel and Distributed Processing Workshops and PhD Forum, 2011 IEEE International Symposium on*, vol. 0, pp. 435–443, 2011.

[12] R. Durbin, S. R. Eddy, A. Krogh, and G. Mitchison, *Biological sequence analysis: Probabilistic models of proteins and nucleic acids*, 7th ed. Cambridge University Press, 1998, pp. 81–95.

[13] S. Lloyd and Q. O. Snell, "Accelerated large–scale multiple sequence alignment," *BMC Bioinformatics*, vol. 12, 2011.

[14] J. Du, R. D. Bjornson, Z. D. Zhang, Y. Kong, M. Snyder, and M. B. Gerstein, "Integrating sequencing technologies in personal genomics: Optimal low cost reconstruction of structural variants," *PLoS Comput Biol*, vol. 5, no. 7, p. e1000432, 07 2009. [Online]. Available: http://dx.doi.org/10.1371%2Fjournal.pcbi.1000432

[15] J. Wang, W. Wang, R. Li, Y. Li, G. Tian, L. Goodman, W. Fan, J. Zhang, J. Li, J. Zhang, Y. Guo, B. Feng, H. Li, Y. Lu, X. Fang, H. Liang, Z. Du, D. Li, Y. Zhao, Y. Hu, Z. Yang, H. Zheng, I. Hellmann, M. Inouye, J. Pool, X. Yi, J. Zhao, J. Duan, Y. Zhou, J. Qin, L. Ma, G. Li, Z. Yang, G. Zhang, B. Yang, C. Yu, F. Liang, W. Li, S. Li, D. Li, P. Ni, J. Ruan, Q. Li, H. Zhu, D. Liu, Z. Lu, N. Li, G. Guo, J. Zhang, J. Ye, L. Fang, Q. Hao, Q. Chen, Y. Liang, Y. Su, A. san, C. Ping, S. Yang, F. Chen, L. Li, K. Zhou, H. Zheng, Y. Ren, L. Yang, Y. Gao, G. Yang, Z. Li, X. Feng, K. Kristiansen, G. K.-S. Wong, R. Nielsen, R. Durbin, L. Bolund, X. Zhang, S. Li, H. Yang, and J. Wang, "The diploid genome sequence of an asian individual," *Nature*, vol. 456, no. 7218, pp. 60–65, 11 2008. [Online]. Available: http://dx.doi.org/10.1038/nature07484

[16] D. C. Richter, F. Ott, A. F. Auch, R. Schmid, and D. H. Huson, "Metasim A Sequencing Simulator for Genomics and Metagenomics," *PLoS ONE*, vol. 3, no. 10, p. e3373, 10 2008.