Algorithms and Analysis for Combining Sequences and Maps: Application to the Malaria Parasite *P. falciparum*[*]

Marco Antoniotti † Thomas Anantharaman ¶
Juliette Colinas †‡ Salvatore Paxia † David C. Schwartz § Bud Mishra †
† Courant Bioinformatics Group, ‡ Department of Biology,
New York University, 719 Broadway 12th Floor, New York, NY, U.S.A.
¶ Biostatistics and Medical Informatics Department,
§ Laboratory for Molecular and Computational Genomics, Departments of Genetics and Chemistry,
University of Wisconsin Madison, WI, U.S.A.
{tsa,dcschwartz}@facstaff.wisc.edu
{marcoxa,paxia,mishra}@cs.nyu.edu
+1 - 212 - 998 3488

1 Introduction

The study of genetics relies on complete nucleotide sequences of the organism together with a description of the transcription units. While this information at its finest level is not often available, or when available, suffers from various errors due to sequencing or assembly, one can garner much information from significantly coarser descriptions that are easily available in genomic maps. Such maps with high resolution and accuracy as well as partially assembled sequences at various degrees of completion exist for many of the microbial organisms, yeasts, worms, flies and now humans. In general, genetically or physically mapped collections of objects derived from the genome under study are still of immense utility, and require robust bioinformatics tools to validate their mutual consistency and integration. The integrated genomic databases derived from all available sources are likely to prove useful even at an early stage for annotation, gap detection (in sequences) and targeted gap closing, sequence contig phasing and map assisted sequence assembly. In this paper, we examine many algorithmic issues involved in such a process. For the sake of concreteness, we focus on an example involving ordered restriction map data based on optical mapping and partially assembled sequences for the malaria parasite, *Plasmodium falciparum*.

In this paper, we focus on validation and genome-wide alignment of sequence data against optical maps using efficient matching algorithm. The validation algorithm derives its efficiency from a dynamic programming formulation that explicitly models all the known error parameters involved in optical mapping. The alignment algorithm, on the other hand, involves resolution of certain inherent ambiguities and appears closely connected to the class of $\mathcal{NP}$-complete problems—particularly, the INDEPENDENT SET problem. We present a greedy algorithm, an efficient approximation algorithm and a search algorithm based on A^* to solve this problem, and study the behavior of these algorithms on the available data. Subsequently, we will expand our work to analyze this partially assembled sequence data to find reading frames, genes, regulatory sequences, etc. The general software structure for the validation and alignment processes is illustrated in Figure 1. In addition to the implementation of the algorithms described here,

[*] This research was conducted under the Department of Energy Grant DoE-25-74100-F1799 and under the National Cancer Institute Grant NCI 5 RO1 CA79063-03.

our system (dubbed Valis) also involves many innovations in hardware and software engineering: a Linux Beowulf cluster, an integrated database with XML interface, and the ConVEx (Contig Visualization and Exploration) visualization tool.

Paper Organization. The paper is organized as follows. First, we give some motivations for our work, especially with respect to the research on Malaria. Secondly, we present a probabilistic analysis of the feasibility of a given (optical) mapping experiment, be it a *validation* or *sequence alignment* (*anchoring*) one. Thirdly, we present our mathematical-statistical model for the validation algorithm (eventually implemented as a dynamic programming procedure [6]), and we pose and discuss a special *alignment problem* that takes into consideration the results of the validation procedure. Finally we report on preliminary results that we found by examining our *Plasmodium falciparum* optical maps (*cf.* [7, 8]) and the content of the *PlasmoDB* (`www.plasmodb.org`) data base.

2 *Plasmodium falciparum* Genome

There are several species of Plasmodium that infect rodents, amphibians and primates, but the ones infecting human are four: *P. falciparum*, *P. vivax*, *P. malariæ*, and *P. ovale*. *P. falciparum* has been responsible for almost all deaths, whereas *P. vivax* has been also widespread but largely responsible for non-fatal morbidity. As a result there has been considerable attention focussed on understanding the genome of *P. falciparum*. Years of research has now yielded a high-resolution genetic map which has localization within 30 Kb or less, physical maps based on YAC contig maps and high-resolution ordered restriction map using genome-wide optical mapping method and complete sequences of only two chromosomes and many unaligned sequence contigs. These efforts have been slow due to the instability of cloned DNA fragments in E. coli, the AT-richness of Plasmodium genome and other technical reasons. But, the available data, if integrated properly at this stage, promise important advances through understanding of gene function, pattern of gene expression and other biological processes.

The haploid *P. falciparum* genome contains 14 nuclear chromosomes with a DNA content of about 30 Mb. These chromosomes are linear and range in length between 800 Kb to 3.5Mb. The telomere sequences cap subtelomeric regions containing repeats and often result in large variations of the sizes of individual chromosomes. Average gene density within *P. falciparum* chromosome has been estimated to be about a gene every 3 to 5 Kb. Most genes do not contain any intron and few genes that do have large exons and extremely short (≤ 1 Kb) introns. Codon usage is biased towards triplets containing A or T in third positions and the non-coding intergenic regions have extremely rich AT content of more than 90%.

With the availability of this genomic data, and because of the difficulties that the sequencing projects must face, we believe that our bioinformatics project aimed at *P. falciparum* must deal with many unique challenges, is likely to identify important issues to be further addressed and will be of interest to the general scientific community.

3 Probabilistic Analysis and Feasibility

3.1 Sequence Validation

Among the many important applications of the high-resolution and high accuracy maps, its use in verifying existing sequences and its help in sequence assembly has the highest immediate impact.

We provide a simple statistical analysis here to show that the sequence verification strategies based on optically-mapped ordered restriction maps, is capable of detecting errors in even conservatively assembled sequences. For instance, consider an assembly scenario, where a genome of length G has been shotgun sequenced (sequence length $= l = 500$) with a coverage c and resulting in $K = Gc/le^{-c}$ islands (about 2000 for a coverage of 5×) and $K-1$ oceans. Let us assume that the sequence contigs and islands are correct with high probability, but as one closes the gaps (represented by the oceans) there is a small probability p_o that the gap is filled by an unrelated random sequence erroneously. Note that the probability that there are r incorrect gap-filling is given by:

$$\binom{K-1}{r}(1-p_o)^{K-r-1}p_o^r.$$

Next as we compare a short region around the gap with the corresponding region in an ordered restriction map (created with m enzymes, β relative accuracy and p_e the cutting probability of each enzyme), we see that given that there are r errors the probability that all the errors will go undetected is simply: $(\beta/2)^{rmp_eG/K}$. Thus the probability that the sequence verification process will admit a sequence that is incorrect is given by

$$\begin{aligned} &\sum_{r=1}^{K-1}\binom{K-1}{r}(1-p_o)^{K-r-1}p_o^r(\beta/2)^{(mp_eG/K)r} \\ =\; & \left(1-p_o\left[1-\frac{\beta}{2}^{mp_eG/K}\right]\right)^{K-1} - (1-p_o)^{K-1} \\ \approx\; & e^{-p_oK(1-(\beta/2)^{mp_eG/K})} - e^{-p_oK}. \end{aligned}$$

A simple calculation shows that if *P. falciparum* genome is assembled from a 5× coverage shotgun sequences (with a $p_o = 10\%$ or smaller)[1], and the map is made with 2 enzymes and a relative sizing accuracy of 15% then the probability that a wrong sequence is incorporated into the final data base is $< 10^{-5}$.

3.1.1 Sequence Anchoring

Once a reference map is created, one can increase its accuracy as well as resolution by optically mapping more and more genomic fragments with the same enzyme or different enzymes. At this point, one can use the high resolution reference map to anchor short sequence contigs and thus generate the complete sequence without relying on the expensive large-scale sequence assembly.

Here we evaluate aspects of the anchoring scheme, where a sequence contig of length L is to be anchored onto a genomic ordered restriction map (of length G) created with m enzymes, where each enzyme is assumed to cut with probability p. Assume that the relative accuracy of the genome-wide restriction map with respect to any enzyme is β. Consider an arbitrary random location s on the restriction map, and we wish to compute the probability that the sequence contig can be placed there. Let an ordered restriction map be created (by purely computational method) for the sequence contig, corresponding to a particular enzyme, and let this computed map be compared to the BAC map at site s.

Thus it is of considerable interest to estimate the probability of false positive as a function of the number of enzymes (m), length (L), probability that the given enzyme cuts at an arbitrary location (p) and the relative accuracy of the restriction map (β).

First consider the case when $m = 1$ and the sequence contig is being placed with a fixed orientations

[1] In practice p_o is significantly smaller.

(out of two) at site s. The false positive probability for a fixed location is then approximately

$$\begin{aligned}
& \sum_{k=1}^{\infty} \Pr[\text{The sequence contig has exactly } k \text{ cuts}] \\
& \qquad \times \Pr[\text{The } k-1 \text{ internal fragments ``match''}] \\
& \qquad \times \Pr[\text{The 2 end fragments ``match''}] \\
\leq & \sum_{k=1}^{\infty} e^{-pL} \frac{(pL)^k}{k!} (\beta/2)^{k-1} \\
= & \ e^{-pL} \sum_{k=1}^{\infty} \frac{(pL\beta/2)^{k-1}}{(k-1)!} \left(\frac{pL}{k}\right) \\
\leq & \ (pL) e^{-pL(1-\beta/2)}
\end{aligned}$$

Note that the matching rule, we have used is fairly simple: given an internal fragment of length x from the sequence contig and a corresponding fragment of length y from the BAC map, we say they match if

$$x(1-\beta) \leq y \leq x(1+\beta).$$

Using m enzymes, and both orientations for the sequence contig, we see that in general this probability is

$$r \approx 2(pL)^m e^{-mpL(1-\beta/2)}$$

Let us assume that we will consider only the cut sites in the genomic map to anchor the sequence contigs; there are on the average Gp such sites to be considered. Thus the probability that the sequence contig does not get anchored at any of the Gp possible false sites is then bounded by e^{-Gpr} and the false positive probability is roughly:

$$FP \approx 1 - e^{-Gpr}.$$

On the other hand, given a sequence contig from the genome, we will fail to place it at the appropriate location, if it has no cut with respect to any enzyme. The probability with which this event occurs is bounded by $FN = e^{-mpL}$, the false negative probability.

Thus we conclude that for a fixed G, as the number of enzymes m increases or the length of a sequence contig L increases, we will be able to place these sequence contigs in the correct location almost surely.

Thus the overall utility of high resolution restriction maps is to enormously facilitate the closure of gaps in several ways: 1) sequence contigs are confidently ordered by alignment to the scaffold maps, and 2) gap lengths are well characterized, thus enabling closure techniques based on PCR. Additionally, such maps provide a means to verify sequence alignments, especially critical when dealing with large regions of repetitive sequence.

3.1.2 Map Assisted Sequence Assembly Algorithm

With the analysis in hand, we can now propose a very efficient and highly accurate sequence assembly algorithm that works based on a Bayesian approach similar to the one proposed for genomic map assembly. The algorithm proceeds in several steps. The first step uses a *perfect hashing* scheme that organizes the set of short sequence reads in a hash table. This organization, implicitly provides a clustering scheme where small clusters of overlapping sequence reads can be assembled. Once short sequence contigs are of length above a predetermined threshold they can be anchored.

Using the anchored sequence contigs, a new hashing function can be recomputed and sequence contigs can be rehashed and reclustered. These steps are iterated until no more unanchored sequence reads

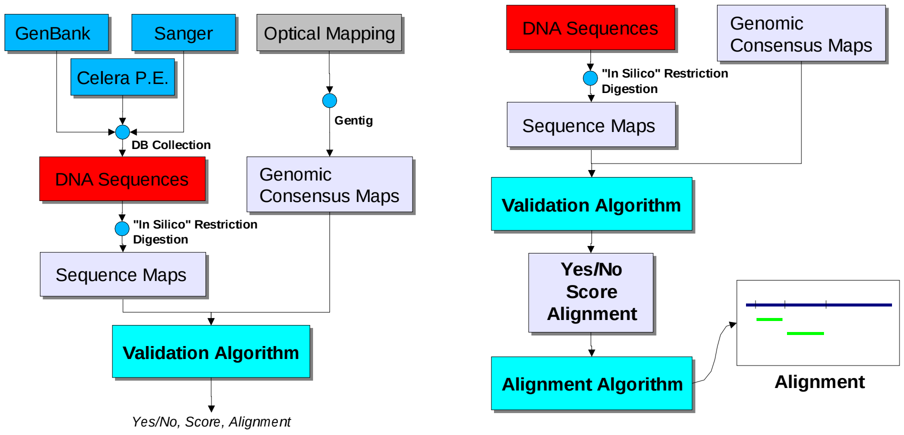


Figure 1: The validation and alignment processes overall flow.

remain. The number of iterations can be easily bounded using a martingale analysis and shown to keep the algorithmic complexity close to linear.

As indicated earlier, at the end of this process not only are sequences properly assembled, but also phased, thereby, providing much more efficient means for gap closing. We postpone a detailed description of the map assisted sequence assembly algorithm to a future publication.

4 Algorithms and Methods

In order to test our approach for validation and alignment, optical maps of the *P. falciparum* genome were "matched" to assembled and unassembled sequence fragments obtained from `www.plasmodb.org`. The *P. falciparum* sequence in `plasmodb` is based on the efforts of by three genomic centers: Sanger, TIGR and Stanford. At present, chromosomes 2 and 3 have been fully assembled and the rest are still being sequenced and assembled.

4.1 Validation and Local Alignment

The *validation* of a sequence map against a (optical) map can be set up as a simple *dynamic programming routine* (DPR) (*cf.* [5, 6]). By itself, the algorithm can be derived from the original formulation of Bellman directly and shares many concepts with a large class of sequence and map alignment algorithms. However, the DPR recurrence and algorithm described here are carefully tailored to a detailed mathematical model based on an MLE function, and handles all the possible sources of errors in the overall map making procedure, by modeling them as a combination of Gaussian, Poisson and Bernoulli processes. This approach to bioinformatics data is original and leads further to interesting algorithmic issues when the scores of many individual DPRs have to be combined as in the alignment problem. In [2] we provide a detailed analysis and derivation of the validation *Maximum Likelihood Estimator* (MLE) for maps against sequences. Here, we only summarize the key ideas.

4.1.1 Statistical Description of the Problem

Figure 2 shows a simple setup of the matching problem involving a sequence map and a consensus map. The sequence map is considered to be *correct* and it is viewed as the *hypothesis* $\mathcal{H}$ of a Bayesian problem to be analyzed, while the consensus map is considered to be a piece of *data* $\mathcal{D}$ to be *validated* against $\mathcal{H}$.

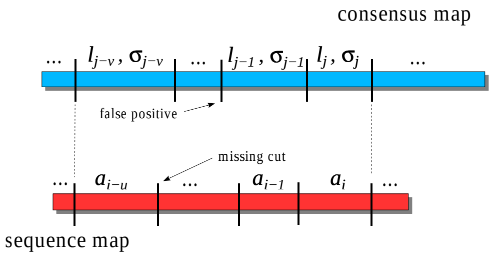

consensus map
... l_{j-v}, σ_{j-v} ... l_{j-1}, σ_{j-1} l_j, σ_j ...
false positive
missing cut
... a_{i-u} ... a_{i-1} a_i ...
sequence map

Figure 2: The most complex matching of a sequence map against a consensus map we consider.. In this case, the fragments $\langle \ldots, l_{i-v}, \ldots \rangle$ to $\langle \ldots, l_i, \ldots \rangle$ from the consensus map, and the fragments $\langle \ldots, a_{j-u}, \ldots \rangle$ to $\langle \ldots, a_j, \ldots \rangle$ from the sequence map match. The intervening cuts are classified by the validation procedure as *missing cuts* (cuts in the sequence maps that were not found in the consensus map) and *false cuts* (vice versa).

As a result, a validation problem can be formulated simply as an optimization problem maximizing the following probability density function

$$\Pr[\mathcal{D} \mid \mathcal{H}(\hat{\sigma}, p_c, p_f)],$$

where $\hat{\sigma}$ is a standard deviation (which summarizes maps wide standard deviation data, i.e. $\hat{\sigma} = \mathrm{f}(\sigma_i)$ for some function ‘f’), and the overall function depends on other parameters as well: p_c, the *cut probability*, and p_f, the *false positive cut probability*. For a detailed description of the sources of these errors and how these map parameters are estimated by the Gentig algorithm, see [1].

Ideal Case

To begin, we formulate the problem in the ideal case where we have *known orientation* of the sequence map, *no false cuts*, and *no missing cuts*. I.e. $p_c = 1$, and $p_f = 0$.

Consider a position h in the consensus map and the consensus map fragment sub-vector from h to $N - 1$. Further, consider the full sequence map fragment vector from 0 to $M - 1$. For the sake of simplicity, we drop the h term and count the consensus map fragments from 0 as well, thus allowing us to write expressions like l_i instead of l_{h+i}.

Let us now consider the “match” between the i-th fragments of the consensus map and of the sequence map. We want to evaluate how much the hypothesis consensus map deviates from the “correct” sequence map. We assume a Gaussian distribution, therefore for the i-th fragment we will need to evaluate the following expression:

$$\frac{1}{\sqrt{2\pi}\sigma_i} e^{-\frac{(l_i - a_i)^2}{2\sigma_i^2}}.$$

Given this expression, and the assumption that the sequence map is correct[2] the overall $\Pr[\mathcal{D} \mid \mathcal{H}(\hat{\sigma})]$ function can be written as

$$\Pr[\mathcal{D} \mid \mathcal{H}(\hat{\sigma})] \quad = \quad \prod_{i=0}^{n} \left(\frac{1}{\sqrt{2\pi}\sigma_i} e^{-\frac{(l_i - a_i)^2}{2\sigma_i^2}} \right).$$

[2] I.e. $\Pr[\mathcal{H}] = 1$.

Maximizing Likelihood. Now we can take the logarithm of the simplified expression and obtain:

$$\ln(\Pr[\mathcal{D} \mid \mathcal{H}(\hat{\sigma})]) \;=\; \sum_{i=0}^{n} \ln\left(\frac{1}{\sqrt{2\pi}\sigma_i}\right) - \sum_{i=0}^{n}\left(\frac{(l_i - a_i)^2}{2\sigma_i^2}\right).$$

This expression maximizes *log likelihood*, therefore it provides a Maximum Likelihood Estimate (MLE).

Minimizing "Weighted Sum-of-Squares". Since we can assume that the first term of the MLE does not vary much from location to location, we simplify the problem by minimizing a "weighted sum-of-error-square" *cost function.*

$$\mathbf{F}(\mathcal{D}) \;=\; \sum_{i=0}^{n}\left(\frac{(l_i - a_i)^2}{2\sigma_i^2}\right)$$

Minimizing $\mathbf{F}(\mathcal{D})$ yields the "best match" of the sequence map represented as $\mathcal{H}$, against the consensus map represented as $\mathcal{D}$.

General Case

Allowing for full generality, we take into account *orientation* of the molecules, *false negatives*, and *false positives* in cut identification. Intuitively, this approach implies that the sizing error is now computed by lumping together a "block" of fragments from the consensus map, and a similar "block" of fragments from the sequence maps. Missing cuts and false cuts contribute to the score function within the blocks and the contributing terms for these cases are based on the underlying Bernoulli and Poisson processes. The DPR *main recurrence* described in Section 4.1.2 shows all these terms. See [2] for a full derivation.

4.1.2 Dynamic Programming "Main" Recurrence

In what follows, the consensus map has m fragments and the sequence map, n fragments; the DPR uses an $n \times m$ matching table $\mathtt{T}$; and the entry $\mathtt{T}[i,\, j]$ contains the (partially evaluated) value of a matching function $\mathbf{F}(\dots)$[3]: i.e. $\mathbf{F}(\dots)$ is incrementally computed from "left" to "right", by considering all possible fragment by fragment matches. We use the index i (respectively, j) to indicate a fragment in the consensus (respectively, sequence) map.

The main recurrence for entry $\mathtt{T}[i, j]$ is shown below:

$$\mathtt{T}[i,j] \;:=\; \min_{\substack{0<u\le i\\ 0<v\le j}} \left(\begin{array}{l} \mathtt{T}[i-u,\, j-v] \\ +\ln\left(\dfrac{\sqrt{2\pi\left(\sigma_i^2+\sigma_{(j-1)}^2+\ldots+\sigma_{(i-v)}^2\right)}}{p_c}\right) \\ +\dfrac{\left(\left(l_j+l_{(j-1)}+\ldots+l_{(j-v)}\right)-\left(a_i+a_{(i-1)}+\ldots+a_{(i-u)}\right)\right)^2}{2\left(\sigma_j^2+\sigma_{(j-1)}^2+\ldots+\sigma_{(j-v)}^2\right)} \\ +(u-1)\,\ln\left(\frac{1}{1-p_c}\right)+(v-1)\,\ln\left(\frac{1}{p_f}\right) \end{array} \right).$$

Note that the main recurrence depends on two parameters u and v. These parameters depend on the σ_i's[4]. However, a pragmatic bound can be set around $3 \times \sigma_i$ (given the current σ_i). This will become a

[3] Because of space limitations we abuse the notation and present only an abbreviated form of the actual mathematical model.

[4] In an even more accurate model, u and v would also depend also on the *digestion rate* of the "in vitro" experiment that breaks up the DNA molecule.

parameter of the DPR. In this way, the computation for each entry $\mathtt{T}(\cdot,\cdot)$ must consider about 9 nearby entries. The overall space complexity of the algorithm is $O(mn)$ and time complexity is $O(c_\sigma \cdot mn)$ (i.e. the actual time complexity depends on a constant, which in turn depends on the overall σ of the experiment instance). The result of a validation algorithm is presented as an ordered sequence of triples, each triple identifying the sequence with the assigned orientation, the matching position and the alignment score; they are sorted in the ascending order in terms of their score values.

4.2 Global Alignment

The validation procedure matches a single sequence map against a consensus map. However, in practice, the result of the validation procedure is interpreted in terms of a global alignment of several sequences against a given consensus map. We call this procedure *alignment*[5].

The result of n "validation experiments" is n sets of *possible sequence positions* along the consensus map: denote these sets as S_i (with $0 < i \leq n$). Formally each of the k items in each S_i is a triple $\langle s_i,\ x_{(i,j)},\ v_{(i,j)}\rangle$, where s_i is a *sequence map identifier*, $x_{(i,j)}$ is the j-th *alignment* of s_i against the consensus map, and $v_{(i,j)}$ is the alignment score (with $0 < j \leq k$). Let $\boldsymbol{S} = \bigcup_i S_i$. The definitions lead to the following problem.

Definition: *Map Based Alignment Problem.* The *Map Based Alignment Problem* requires one to choose at most one triple from each S_i satisfying the following conditions:

1. The chosen s_i's do not *overlap* (although this requirement may be relaxed);
2. $\sum_i \left(I_i \times v_{(i,j)}\right)$ is minimized;
3. $n - \sum_i I_i$ is minimized;

where I_i is an *indicator* variable assuming a value 1 if a triplet from S_i is included in the chosen set, and 0 otherwise. □

It should be immediately clear that objectives (2) and (3) conflict: the minimum of objective (2) is achieved when no sequence is chosen, while (3) requires to choose as many sequences as possible,irrespective of the score values. We resolve this conflict by a weighting scheme involving a Lagrangian-like term linearly combining the two contradictory objectives.

In the following we consider several approximation algorithms to solve this problem. We start by considering the problem for the special case when $k = 1$ and devise an efficient algorithm. Next, we consider the general case when $k > 1$ and devise good approximation heuristics. We discuss three different solutions: a (yet another) Dynamic Programming solution, a Greedy solution, and a Graph Search A^* solution, and show when they are applicable.

4.2.1 Characterization of the Alignment Problem for $k = 1$

If we restrict the number of sequences present in each S_i to 1 (just the best score) then the problem yields to a feasible solution. In general, if the sequence matches uniquely to one map location, then this case applies and the general case is only of theoretical interest.

The complete alignment algorithm, constructing a solution $\boldsymbol{P}$, is sketched below.

1. Sort all the $\langle s_i, x_{(i,1)}, v_{(i,1)}\rangle$'s in ascending $x_{(i,1)}$ order and store the result into a list L; from now on, the indices i and j's range over L.

[5] The term "global alignment" is used here in a somewhat nonstandard manner.

2. Construct two the vectors $C[i]$ and $B[i]$ $(0 < i \leq n)$, where each entry in C is defined to be the *cost* of including s_i in an alignment that already contains (a subset of the) sequences up to s_j; the index j is stored in $B[i]$.
   The update rules for $C[i]$ and $B[i]$ basically searches backward in the C vector for values minimizing the cost function and setting B to "point back" to the chosen point.

$$\begin{aligned} C[i] &= \max_{0<j<i} (C[j] + W(\lambda;\, i)) \text{ such that } s_i \text{ does not overlap with } s_j, \\ B[i] &= j. \end{aligned}$$

   The form of the $W(\lambda;\, i)$ function takes into account the conflicting nature of the objectives. Since we cannot optimize both objectives simultaneously, we build a weight function (where a user may supply the parameter λ) which will account for both. Two possible W's are shown below:

$$\begin{aligned} W_1(\lambda;\, i) &= |s_i| - \lambda \cdot v_i, \\ W_2(\lambda;\, i) &= I_i - \lambda \cdot v_i. \end{aligned}$$

   W_1 takes into account the "span" covered by the chosen sequences ($|s_i|$ is the size of the sequence). W_2 takes into account the number of sequences chosen (I_i is the indicator variable previously introduced. The parameter λ is under user control.

4.2.2 Characterization of the Alignment Problem for $k > 1$

If $k > 1$ then the problem becomes much more complex. Since for each sequence s_i, we have k alignments to choose from, the complexity, involved in a straightforward generalization of the preceding algorithm, grows exponentially in n. We conjecture that this problem is $\mathcal{NP}$-complete and closely related (with respect to p-time reduction) to INDEPENDENT SET problem.

Apart from these complexity considerations, one can use the following heuristic algorithm to produce a good (though suboptimal) solution in the case $k > 1$. The idea is to simply iterate the basic dynamic programming algorithm (i.e., $k = 1$ case) on an input set that takes the best possible solutions from each S_i while ignoring the non-overlapping constraint. The solution is further improved in the subsequent iteration, by constructing a new input to the basic ($k = 1$) DPR algorithm, that consists of the preceding solution augmented with an element from each S_i excluded in the preceding solution. Clearly, since the preceding solution is also a solution (possibly non-optimal) of the new problem, the new solution is no worse than the solution so far. In each iteration, the basic optimal solution is also a general (suboptimal) solution. Since an item once removed from consideration, is never reconsidered, there can be only $O(kn)$ iterations, and each iteration involves $O(n^2)$ work. Hence a naive analysis yields an $O(kn^3)$ time algorithm.

A* Graph Search. The construction of the best solution $\boldsymbol{P}_n$ (n being – in our case – the number of sequences) to the alignment problem can also be cast as a *heuristic graph search* problem [9]. A heuristic graph search problem constructs a solution $\boldsymbol{P}_n$ by augmenting a partial one $\boldsymbol{P}_t$ for which we know a cost $g(\boldsymbol{P}_t)$. One particular case of these class of algorithms is A*. In A*, the choice of the next item $x_{(t+1)}$ to add to $\boldsymbol{P}_t$ is done by taking into account g and an *estimate* $h(\boldsymbol{P}_{(t+1)})$ of the "distance" to an optimum solution $\boldsymbol{P}_n$. The function

$$f = g + h$$

is called the *heuristic evaluation function* for a partial solution.

Since the partial solutions are generated in an incremental way, and the set of choices is maintained as a graph (and since the core of the algorithm is not much different from Dijkstra's Shortest Path algorithm), these algorithms are classified as "graph search algorithms". The difficulty in devising a good A* implementation for the problem at hand lies in finding a "good" h function which satisfied the conditions described in [9].

We have explored this venue as well and looked into several h functions; while none of which completely satisfied the desired conditions, nevertheless they appear to work well in practice and require more analysis.

5 Experimental Results

We used the NYU-BiG ApplE (Bioinformatics Group Application Environment) software to run several experiments involving *P. falciparum* genome. Our first experiments checked "in silico" maps obtained from *P. falciparum* sequence data against optical ordered restriction maps for the same organism. Our second experiment produced a putative ordering of the published contigs for chromosomes 14.

We obtained the sequences for the *P. falciparum*'s 14 chromosomes from the `www.plasmodb.org` site. Our experiment cut the sequences "in silico" using the `BamHI` restriction enzyme. The resulting maps are input to the validation program along with appropriate optical ordered restriction maps.

5.1 Validation of Chromosomes 2 and 3

Here, we report results of our extensive experiments on chromosome 2 and chromosome 3. In the full paper, we describe the complete experiment with other enzymes (e.g. `NheI`) as well.

We produce two "in silico" maps for the chromosome 2 and chromosome 3 sequences with the enzyme `BamHI`. For chromosome 2 we built maps with 30 and 23 fragments, for chromosome 3 we built maps with 36 and 28 fragments (taking into account orientation). The molecule maps thus produced are then sent to the validation checker along with the possible consensus maps. The optical ordered restriction maps we used were published in [7, 8]. Since the published maps omitted all the statistically relevant information, we also used maps generated subsequent to the publication, by an improved version of the `gentig` program.

The `gentig` program gave us an indication of the overall standard deviation to be used for each fragment of the consensus map. The parameter used was $\hat{\sigma} = 4.4754$ Kbps, and each fragment was assigned a standard deviation of $\hat{\sigma}\sqrt{\frac{l}{L}}$ Kbps, where l is the *fragment size* and L is the *average* consensus map fragment size.

Our implementation is not completely optimized. However, speed did not turn out to be an issue. The system runs the $75 \times 4 = 300$ DPR instances in about 3 minutes[6].

We produced the results reported in Tables 1, 2, and 3. Tables 1 and 3 show the match of the sequence maps for chromosomes 2 and 3 against the consensus maps generated by `gentig`. Table 2 show the match of the sequence maps against the consensus map published in [4].

The data we report here are a summary of the data we actually produced. In particular we also have the position of the matches of the sequence maps against the consensus maps.

5.2 Alignment Experiment of Contigs for Chromosome 14

For the contigs assigned to chromosomes 14 in the `www.plasmodb.org` database, we ran an alignment experiment using our validation post-processing tool.

The alignment tool proposed an alignment of all contigs of length roughly greater that 20 Kbps. We are limited to this number by the resolution of the particular optical maps used. Parts of the alignment are shown in Figures 3 and 4. Figure 4 in particular shows an interesting overlap of few sequences. In the

[6] The system may appear slow, but in reality it keeps track of all the intermediate results and makes them available for interactive inspection after the actual run. Also, the sequence, the sequence map, and the consensus maps, are always available for inspection and manipulation.

Chromosome 2 Validation Summary A

rank	matches	score	map id	# missing cuts	# false cuts
1	29	80.869	1302	0	1
2	28	105.861	1302	2	1
3	18	126.956	1326	12	4
4	22	127.488	1305	8	4
5	18	132.890	1414	12	2

Table 1: The data reported shows the best "matches" found by the validation checker in the case of *P. falciparum* chromosome 2. The "in silico" sequence map was obtained from the TIGR database sequence. The sequence map (as well as its reversed) was checked against 75 (optical) consensus maps produced by gentig. The 75 optical maps cover the entire *P. falciparum* genome. The validity checker found its best matches against the map tagged 1302.

Chromosome 2 Validation Summary B

rank	matches	score	map id	# missing cuts	# false cuts
1	29	77.308	NYU-WISC	1	0
2	22	125.088	NYU-WISC	8	2
3	22	130.866	NYU-WISC	8	4
4	24	131.475	NYU-WISC	6	1
5	24	132.838	NYU-WISC	6	4

Table 2: The data reported shows the best "matches" found by the validation checker in the case of *P. falciparum* chromosome 2. The "in silico" sequence map was obtained from the TIGR database sequence. The sequence map (as well as its reversed) was checked against the map published in [4].

Chromosome 3 Validation Summary

rank	matches	score	map id	# missing cuts	# false cuts
1	35	108.360	1365	1	0
2	32	117.571	1365	4	1
3	32	119.956	1365	4	2
4	35	121.786	1296	1	3
5	31	125.265	1365	5	1

Table 3: The data reported shows the best "matches" found by the validation checker in the case of *P. falciparum* chromosome 3. The "in silico" sequence map was obtained from the Sanger Institute database sequence. The sequence map (as well as its reversed) was checked against 75 (optical) consensus maps produced by gentig. The 75 optical maps cover the entire *P. falciparum* genome. The validation checker found its best matches against the map tagged 1365.

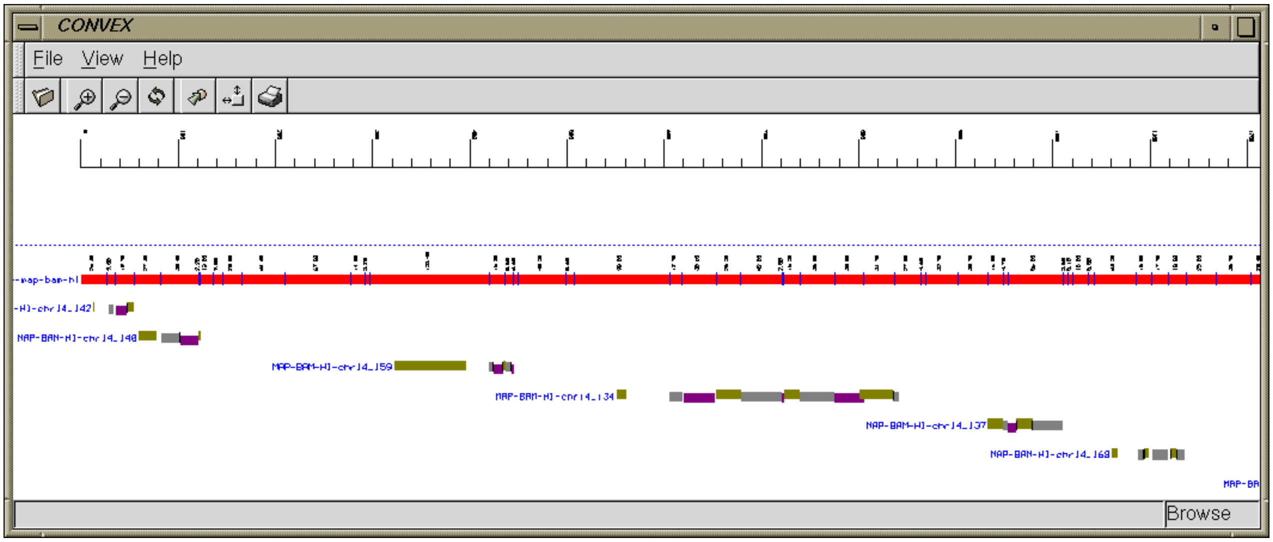


Figure 3: Alignment of Chromosome 14 maps against Chromosome 14 Optical Map [8].

full paper we will report the results of running BL2SEQ on such overlapping sequences to check whether the contigs can be extended.

6 Conclusions

We have been building a complex set of tools (the "NYU BIG ApplE") based on a number of technologies centered around the idea of single molecule physical (optical) mapping. These tools are built on a well founded statistical-mathematical analysis of the biochemical and optical processes at hand. In this paper we presented three such analysis and we described the tools we built upon them: *feasability analysis*, *validation*, and *alignment*.

We showed very preliminary results from the application of these tools to the analysis of the *P. falciparum* DNA.

Our results point out how our technology can be useful in assessing and improving the goodness of various sequence and map data currently being published in a variety of formats from a variety of sources.

References

[1] T. Anantharaman and B. Mishra. A Probabilistic Analysis of False Positives in Optical Map Alignment and Validation. submitted to ISMB 2001.

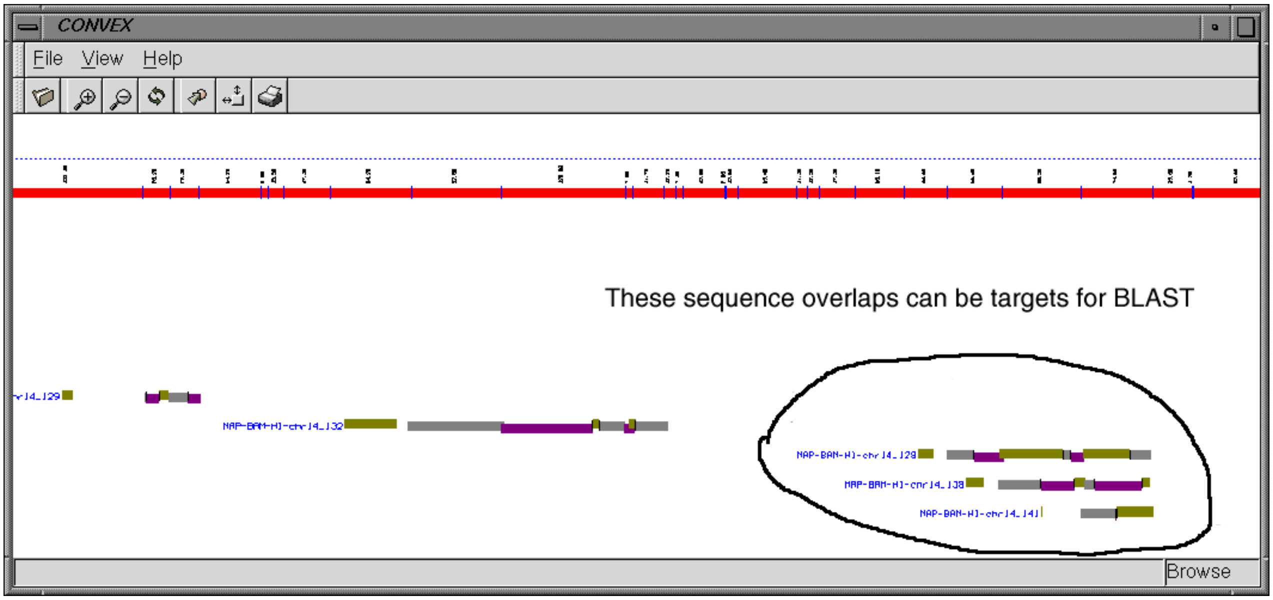
[2] M. Antoniotti, T. Anantharaman, S. Paxia, and B. Mishra. Genomics via Optical Mapping IV: Sequence Validation via Optical Map Matching. Technical Report CIMS-TR-811, NYU Courant Bioinformatics Group, 719 Broadway 12th Floor, New York, NY, 10003, U.S.A., 2001.

[3] C. Aston, B. Mishra, and D. C. Schwartz. Optical Mapping and Its Potential for Large-Scale Sequencing Projects. *Trends in Biotechnology*, 17:297–302, 1999.

[4] M. J. Gardner et al. Chromosome 2 sequence of the human malaria parasite *Plasmodium Falciparum*. *Science*, 282:1126–1132, 1998.

[5] D. Gusfield. *Algorithms on Strings, Trees, and Sequences*. Cambridge University Press, 1997.

[6] X. Huang and M. S. Waterman. Dynamic Programming Algorithms for Restriction Map Comparison. *Comp. Appl. Bio. Sci.*, 8:511–520, 1992.



These sequence overlaps can be targets for BLAST

Figure 4: Another region of the same alignment of Chromosome 14. In this case the alignment procedure decided that some maps overlap. This shows the value of our tool as a mean to direct experimental efforts.

[7] J. Jing, Z. Lai, C. Aston, J. Lin, D. J. Carucci, M. J. Gardner, B. Mishra, T. Anantharaman, H. Tettelin, L. M. Cummings, S. L. Hoffman, J. C. Venter, and D. C. Schwartz. Optical Mapping of *Plasmodium Falciparum* Chromosome 2. *Genome Research*, 9:175–181, 1999.

[8] Z. Lai, J. Jing, C. Aston, V. Clarke, J. Apodaca, E. T. Dimalanta, D. J. Carucci, M. J. Gardner, B. Mishra, T. Anantharaman, S. Paxia, S. L. Hoffman, J. C. Venter, E. Huff, and D. C. Schwartz. A shotgun optical map of the entire *Plasmodium Falciparum* genome. *Nature Genetics*, 23:309–313, 1999.

[9] J. Pearl. *Heuristics: Intelligent Search Strategies for Computer Problem Solving*. Addison-Wesley Publishing Co., 1984.

[10] X. Su, M. T. Ferdig, Y. Huang, C Q. Huynh, A. Liu, J You, J. C. Wootton, and T. E. Wellems. A Genetic Map and Recombination Parameters of the Human Malaria Parasite *Plasmodium falciparum*. *Science*, 286, 1999.

[11] X.-Z. Su and T. E. Wellems. Genome Discovery nd Malaria Research: Current Status and Promise. In I. W. Sherman, editor, *Malaria: Parasite Biology, Pathogenesis, and Protection*. ASM Press, Washington, D.C., U.S.A., 1998.