Algorithms for Molecular Biology **Fall Semester, 2001**

Lecture 9: January 3, 2002

Lecturer: Ron Shamir *Scribe: Amir Pelleg and Kobi Lindzen*[1]

9.1 Motivation

Maps are used in order to assist in a search of a specific item between many. When searching the DNA for a specific gene or chromosome DNA maps in different levels of details are required. Searching for a specific gene in a human DNA is analogous to searching for a specific human on earth. In order to find such gene, one would need maps of the genome in order to perform an efficient search. Figure 9.1 presents the analogy as presented to the U.S congress in the 1980's. DNA maps let us organize the unknown, analogous to an unexplored continent, into manageable units so that we can find our way around and locate more features of interest.

Physical mapping is the process of determining the relative position of landmarks along a genome segment. The resulting maps are used as a basis for DNA sequencing, and for the isolation and characterization of individual genes or other DNA regions of interest (e.g., transcribed regions or regulatory elements). The construction of high resolution sequence-ready physical maps for human and other organisms continues to be one of the top priority tasks of the Human Genome Project.

Scalable maps are produced using the "divide and conquer" method. This method can be achieved by taking each chromosome of the DNA and breaking it into smaller parts (0.5 - 2 Mega base pairs). each such part is put into a Yeast Artificial Chromosome (YAC), which can reproduce the desired sequence. That sequence is then broken into smaller sequences whose length is 40,000 bp in average, and put into Cosmid (which can be reproduced). Each such sequence is broken into, yet smaller sequences that are put in Plasmid whose average length is about 4,000 bp. Those plasmid are then used in different physical mapping procedures in order to map the lowest level of the DNA. Figure 9.2 shows how mapping is done using "divide and conquer".

Given a long DNA segment it is relatively easy to produce a large group of DNA fragments known as clones. The process of creating the clones consists of breaking several copies of the original DNA sequence at many locations, and then cloning each of the fragments. One of the problems with the cloning process is that the resulting fragments are obtained "out of order". This means that it is difficult to re-assemble the fragments in order to get

[1] Based on a scribe by Guy Erez and Ofer Rahat, December 25, 2000.

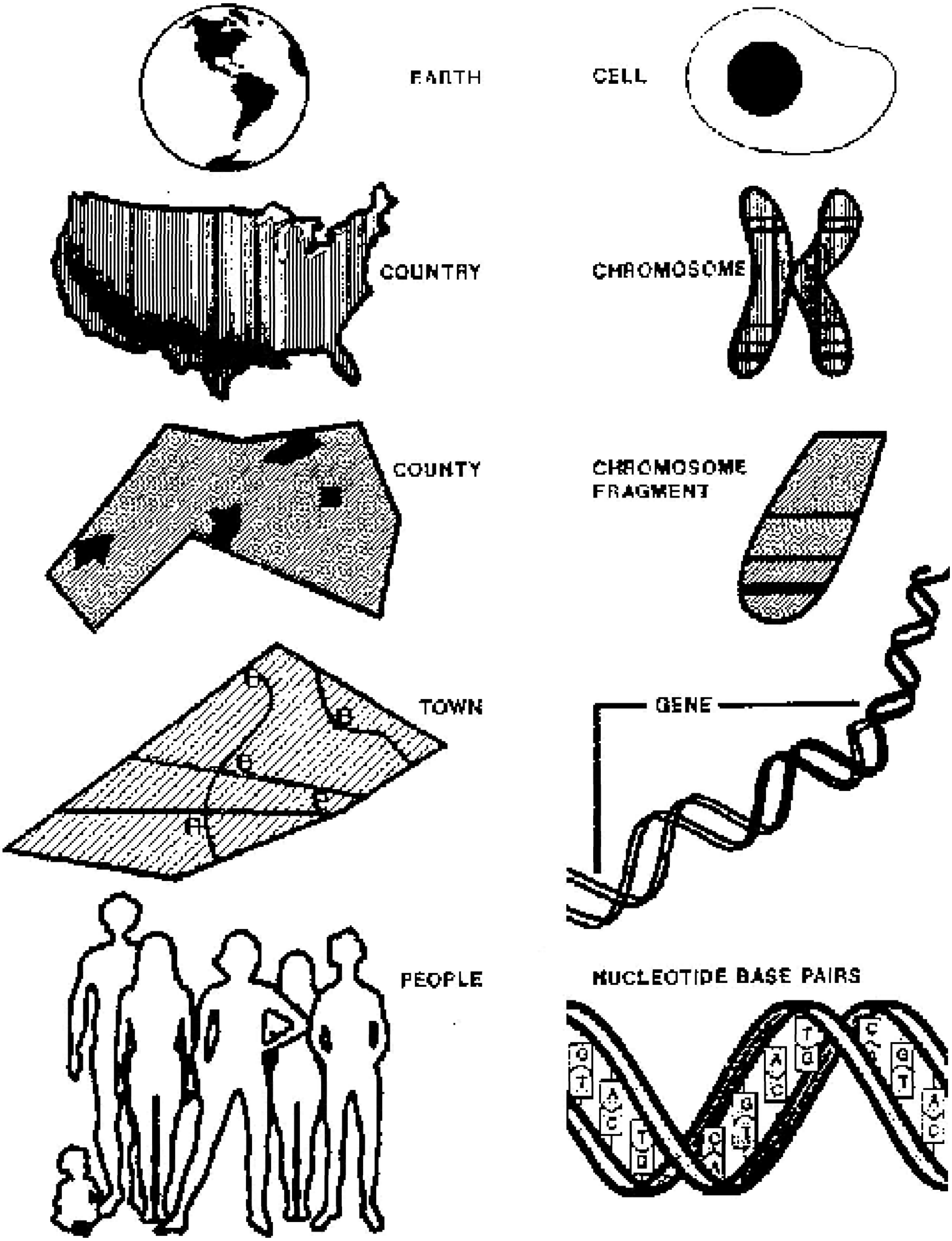


Figure 9.1: Source:[Document presented to the U.S. Congress]. Analogy between base pairs search to finding a person.

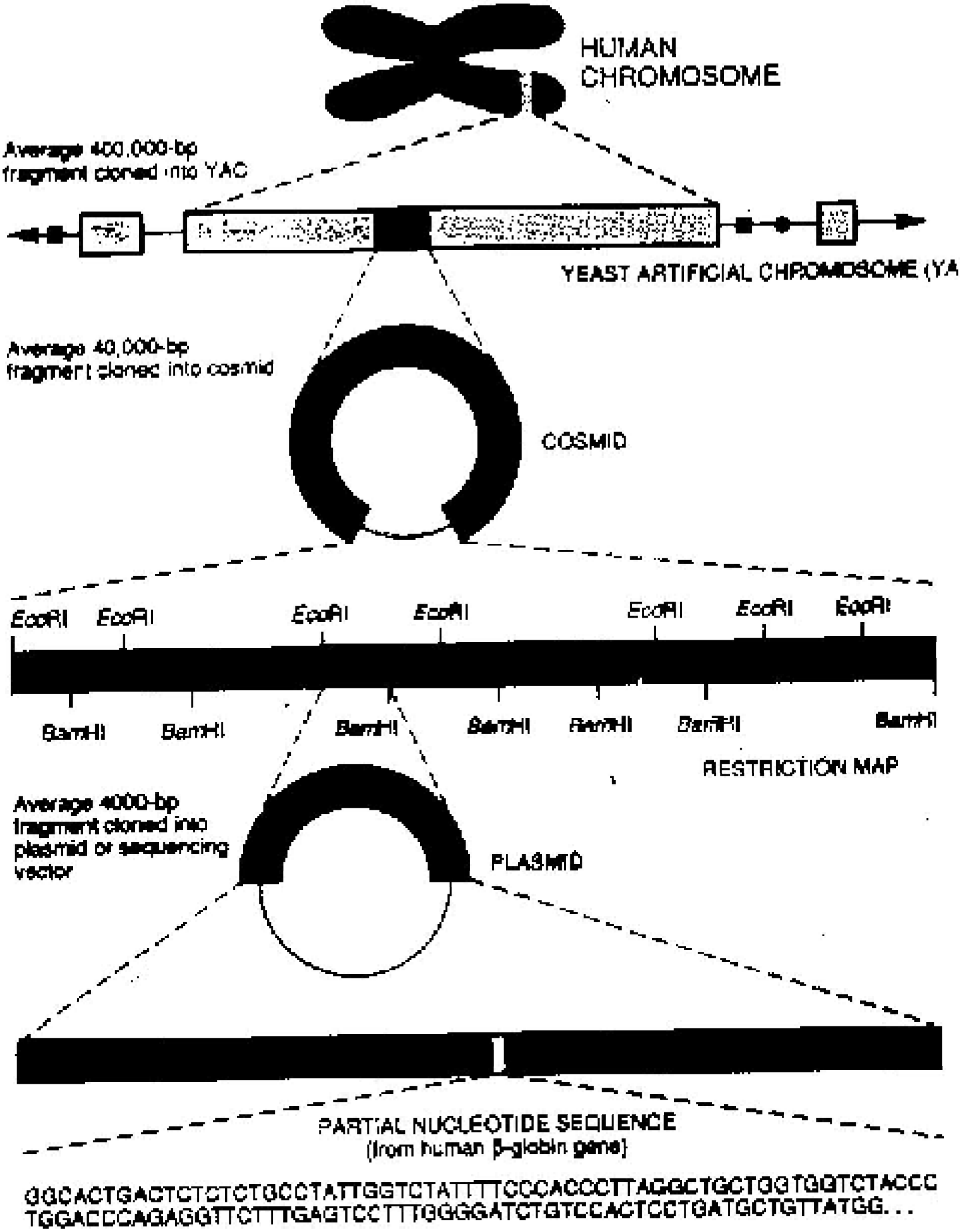

HUMAN CHROMOSOME
Average 400,000-bp fragment cloned into YAC
YEAST ARTIFICIAL CHROMOSOME (YAC)
Average 40,000-bp fragment cloned into cosmid
COSMID
EcoRI
EcoRI
EcoRI
EcoRI
EcoRI
EcoRI
EcoRI
BamHI
BamHI
BamHI
BamHI
BamHI
BamHI
BamHI
RESTRICTION MAP
Average 4000-bp fragment cloned into plasmid or sequencing vector
PLASMID
PARTIAL NUCLEOTIDE SEQUENCE (from human β-globin gene)
GGCACTGACTCTCTCTGCCTATTGGTCTATTTTCCCACCCTTAGGCTGCTGGTGGTCTACCC
TGGACCCAGAGGTTCTTTGAGTCCTTTGGGGATCTGTCCACTCCTGATGCTGTTATGG...

Figure 9.2: Source:[Document presented to the U.S. Congress]. Scalable mapping of DNA - Steps of accessing the DNA sequenced for mapping. Using YAC, Cosmid and Plasmid in order to reproduce the partitions desired.

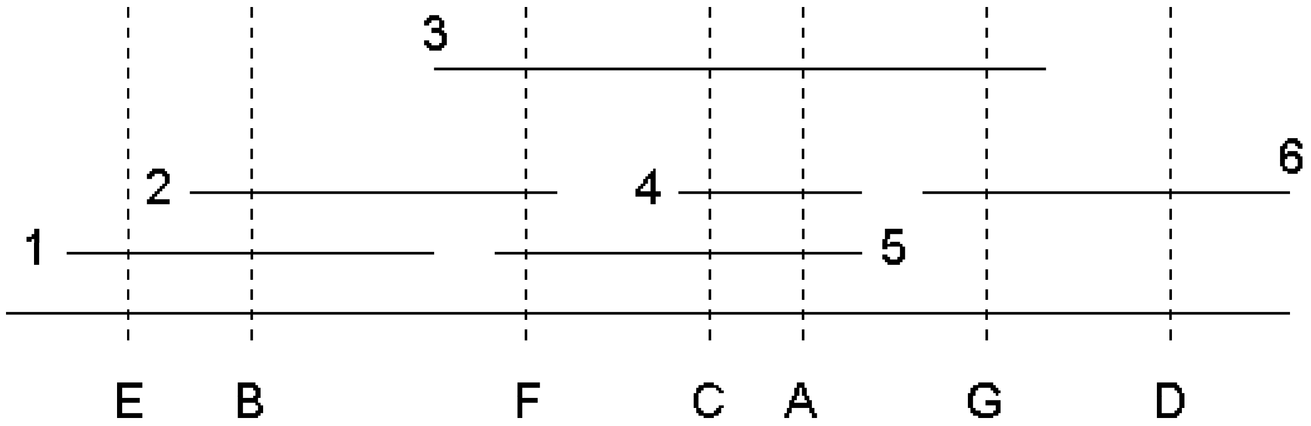


Figure 9.3: Ordered clones and several STS probes.

a map of the original sequence. Moreover, the cloning process does not ensure that a continuous sequence of DNA can be reconstructed from the fragments. The reassembly of the clones presents several combinatorial problems, which are the focus of this chapter.

9.1.1 Unique Probe Mapping

An *STS (Sequence Tagged Site) probe* or *STS discriminator* is a filter that can uniquely determine whether or not a specific short sequence of DNA appears along any given (longer) sequence. The filter can identify the existence of the short sequence but provides no information as to its location. Using a short sequence of 200 - 300 bases ensures that the probability of an error in recognition is relatively low. Running a number of STS probes against numerous clones results in a matrix cell $M_{i,j}$ with the entry 1 (0) representing a positive (negative) result of probe j against clone i. Figure 9.3 gives an example of ordered clones and corresponding STS probes. Table 9.1 presents the resulting STS matrix.

The problem faced is to find all such orders of clones covering in the correct order the desired sequence. The problem is solved by finding the order of the probes in sequence, therefore imposing the clones' order.

Problem 9.1 *The unique probes mapping problem.*
INPUT: A set of elements U (probes) and a collection of subsets $\varphi = \{S_1, S_2, \ldots, S_n\}, \forall\imath :$ $S_\imath \subseteq U$.
QUESTION: Find the set $\Pi(\varphi)$ of all permutations over U along which every $S_\imath$ is contiguous.

Clone / Probe	A	B	C	D	E	F	G
1	0	1	0	0	1	0	0
2	0	1	0	0	0	1	0
3	1	0	1	0	0	1	1
4	1	0	1	0	0	0	0
5	1	0	1	0	0	1	0
6	0	0	0	1	0	0	1

Table 9.1: Resulting STS matrix. 1 in line i column j denotes that clone i contains probe j.

Clone / Probe	E	B	F	C	A	G	D
1	1	1	0	0	0	0	0
2	0	1	1	0	0	0	0
3	0	0	1	1	1	1	0
4	0	0	0	1	1	0	0
5	0	0	1	1	1	0	0
6	0	0	0	0	0	1	1

Table 9.2: Solved STS matrix. The set of solved matrixes is equivalent to the PQ-Tree Algorithm solution.

Problem 9.1 is equivalent to the problem of rearranging the columns of the STS result matrix so that all 1's in the rows of the matrix are consecutive. Table 9.2 shows a solution to the problem presented in Figure 9.3 and represented in Table 9.1 This attribute is also known as the *consecutive ones property.*

The problem of finding the set of permutations $\Pi(\varphi)$ is a well known problem in computer science. A linear time algorithm for solving this problem was presented in 1976 by Booth and Lueker [1]. Clearly, an explicit representation of the collection of all the resulting permutations may require much more than linear space. Therefore, a linear time algorithm requires a linear space representation of this collection. This linear representation is achieved by *PQ-trees* which are described in the following section.

9.1.2 The PQ-Tree Algorithm

A PQ-Tree is a rooted, ordered tree. We will use a PQ-tree with the elements of U as leaves, and internal nodes of two types: *P-nodes* and *Q-nodes.*

A P-node whose sub-nodes are $T_1, \ldots, T_k$ for $k \geq 2$, represents k subsets of U (the leaf sets of $T_1, \ldots, T_k$), each of which is known to be a consecutive block of elements, but with the order of the blocks unknown. A Q-node whose sub-nodes are $T_1, \ldots, T_k$ for $k \geq 3$ represents that the k blocks corresponding to the leaf set of $T_1, \ldots, T_k$ are known to appear in this

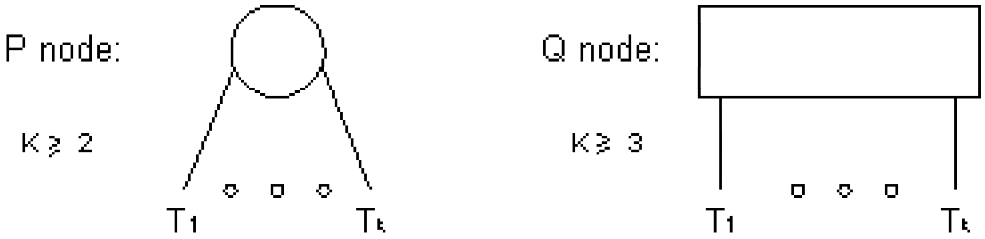


Figure 9.4: PQ-tree node types: we use circles and bars to denote P-nodes and Q-nodes, respectively.

order, up to a complete reversal (see figure 9.4).

Since the nodes (P and Q) represent legal order of their leafs, some moves on the tree are allowed. Figure 9.6 1 - 2 demonstrates the *legal transformation* allowed on a junction in a PQ Tree for each such node.

1. Reordering the sub-nodes of some P-node arbitrarily

2. Reversing the order of the sub-nodes of some Q-node

From these two ordering actions allowed on the node of the tree, a set of rules relevant for the legal transformation on combination of nodes can be concluded. Such set of rules must be based on the two basic legal transformation, and would be used in order to rearrange the PQ-Tree when new constraints on the leafs order is applied. The set of rules can be seen in Figure 9.7. Not all possible combinations are seen in the figure, since some combinations will not maintain the legality that is imposed on the leafs order by the nodes' types. In the figure, the coloring remarks the minimal subtree that contains the leafs on which the new constraint is applied upon. When rearranging the PQ-Tree, transformation actions should only be applied to the colored subtree in order to make the tree legal.

Definition The *frontier* of a PQ-tree is the set of all leaves, read in a left-to-right order (see Figure 9.5)

Two PQ-trees T and T' are said to be *equivalent* if there exists a set of legal transformations leading from one tree to the other. In such a case, we write $T \equiv T'$.

Definition We denote the set of all frontiers of trees equivalent to T as *Consistent(T)*. Formally: $Consistent(T) = \{Frontier(T')|T' \equiv T\}$

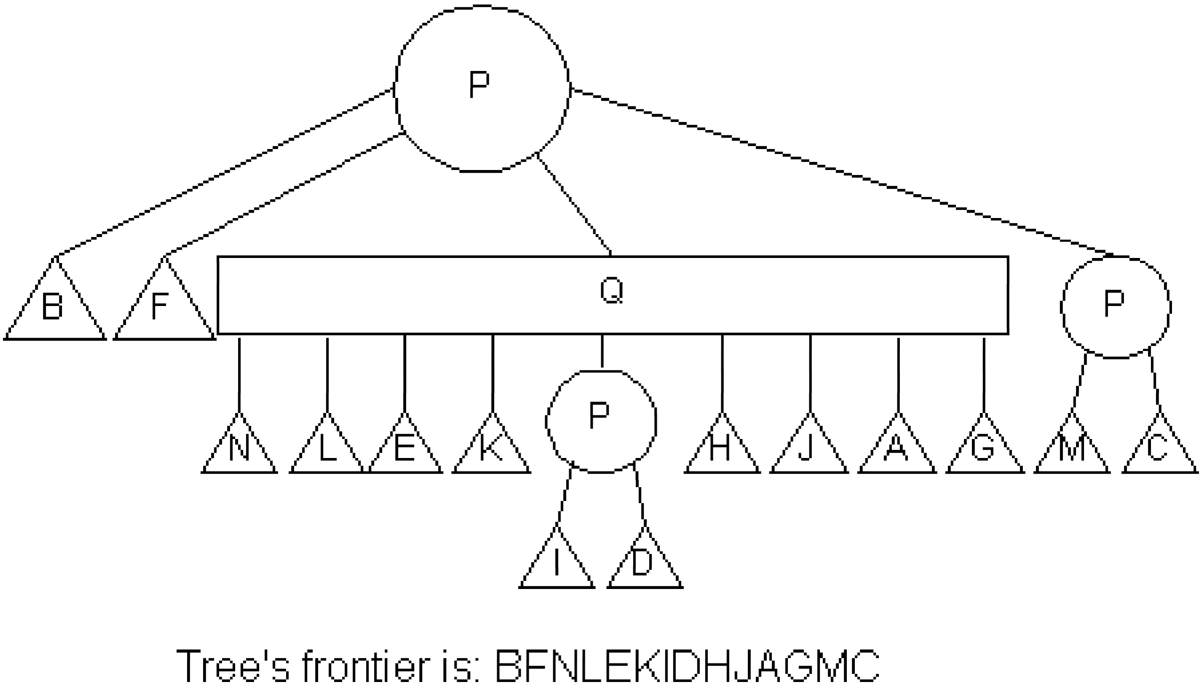


Figure 9.5: Frontier of a PQ-tree.

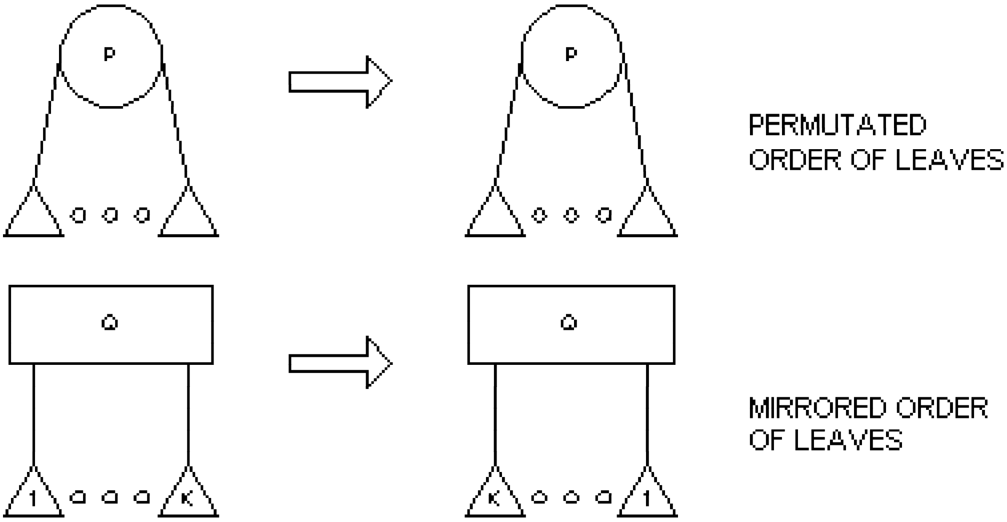
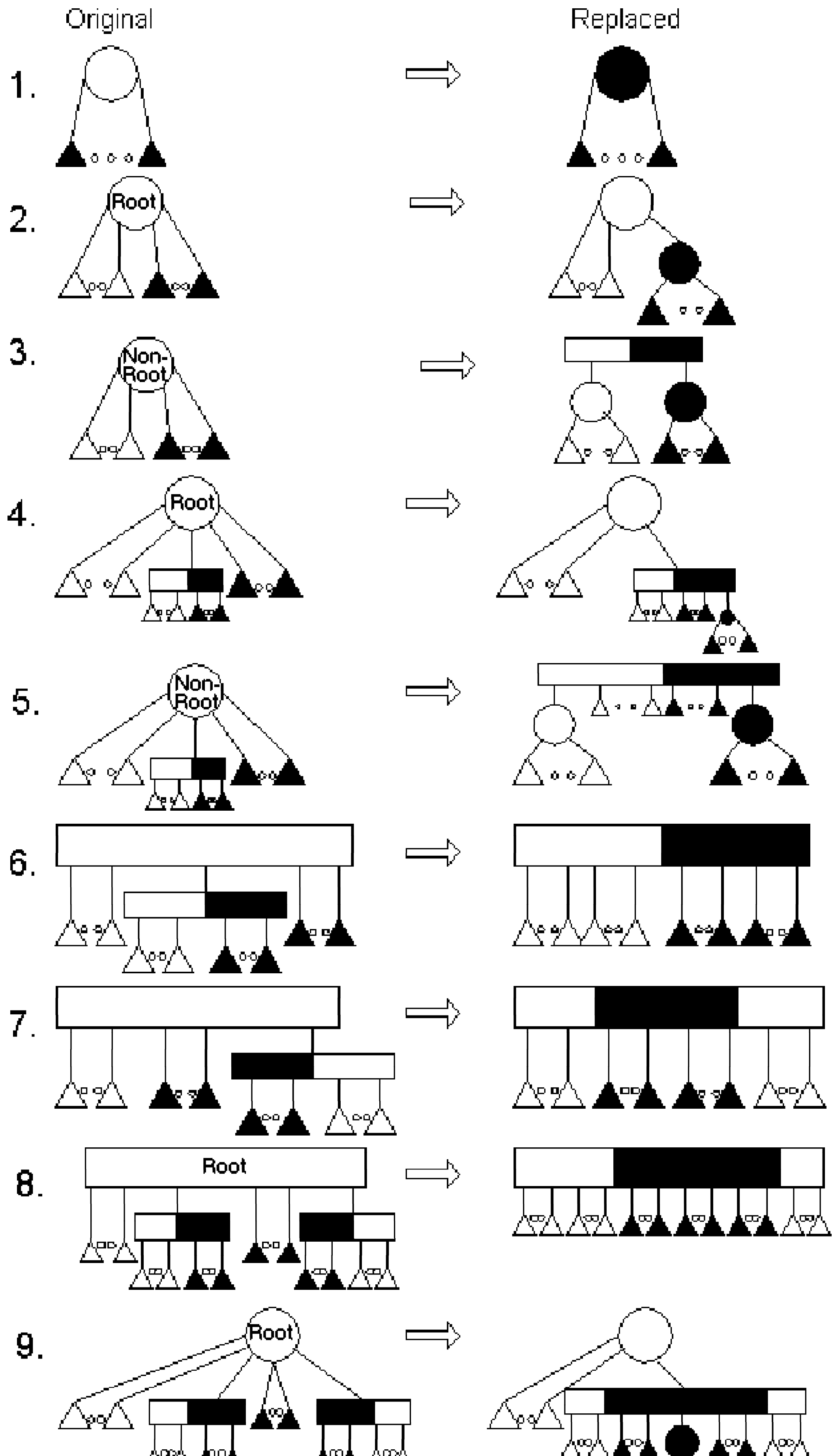


Figure 9.6: Legal transformations of a PQ-tree.


Original
Replaced
1.
2.
Root
3.
Non-Root
4.
Root
5.
Non-Root
6.
7.
8.
Root
9.
Root

Theorem 9.1 (Booth-Lueker 1976 [1])

1. *For every U, φ there exists a PQ-tree T s.t. Consistent*$(T) = \Pi(\varphi)$.
2. *For every given PQ-tree T, there exists U, φ s.t. Consistent*$(T) = \Pi(\varphi)$.

The theorem implies that the problem of permuting the probes in order to achieve the consecutive ones property of the STS matrix is equivalent to finding a PQ-tree representing $\Pi(\varphi)$.

PQ-Tree Algorithm for Unique Probe DNA Mapping:

Intuition: An empty tree is initialized. For every sub group, the leafs holding the relevant member of the group will be found and the minimal subtree containing them is found. Legal transformation moves are applied (to the subtree) until all members of the group appear consecutively in the tree's frontier. The subtree is unmarked, and a new group is added. If no legal move can be applied to a tree in order to arrange the leafs to appear consecutively, the problem cannot be solved.

Formally:

1. Initialize the tree as a root P-node with all elements of U as sub-nodes (leaves).
2. For $i = 1, \ldots, n$: *Reduce* (T, S_i)

After Stage i, T is induced, i.e. modified, so such that *consistent*(T) contains only permutations in which S_i is continuous.

Reduce(T, S_i)

1. Color all S_i leaves.
2. Apply transformations to replace T with an equivalent PQ-tree such that all of the colored leaves are consecutive along its frontier.
3. Identify the deepest node *Root*(T, S_i) whose subtree spans all colored leaves.
4. Apply *replacement rules* presented in Figure 9.7: Traverse the nodes of this subtree, working bottom-up till reaching *Root*(T, S_i), when all colored leafs are adjacent. Advance on the subtree in a bottom-up manner, rearranging the leafs of every node visited according to the legal moves. For an empty P node, and for full or empty Q node, no change is needed. If there is no matching pattern, return the null tree (meaning there is no correct arrangement).

5. Remove the coloring (all the information in S_i is now included in the structure of the tree).

Figure 9.8 shows an example of applying the PQ-Tree algorithm for unique probe DNA mapping.

The problem with using PQ-trees for solving the unique mapping problem is that the algorithm does not support noise: Unfortunately, due to measurement errors the input matrix usually has either extra or missing 1 entries. In such a case, the resulting PQ-tree[2] will not produce the best (minimum error) solution available, but rather an arbitrary solution depending on the clone order chosen. Since errors are not uncommon, this deficiency deters one from using the algorithm.

9.1.3 Interval Graph Formulation for the Unique Mapping Problem

Form a graph $G(V, E)$ whose vertices are the clones, and its edges are $E = \{(v_i, v_j)|$ there exists a probe p, $p(v_i) = p(v_j)\}$.

A graph $G(V, E)$ is said to be an *interval graph* if every node can be represented by an interval and an edge exists between two vertices if and only if the intervals corresponding to them overlap. The set of such intervals is then called a *realization* of G.

Problem 9.2 *The interval graph recognition problem.*
INPUT: A graph $G(V, E)$.
QUESTION: Determine whether G is an interval graph.

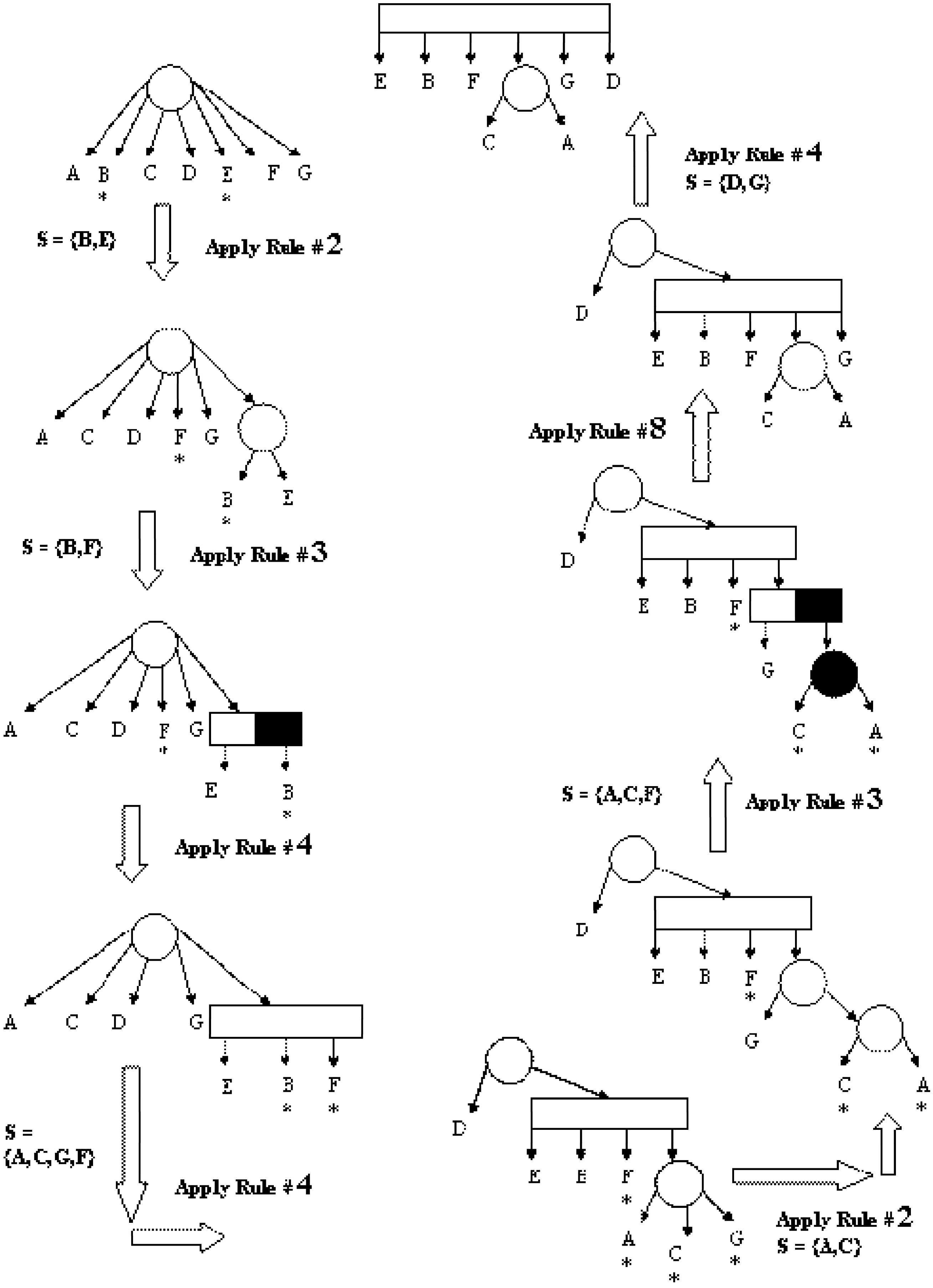
Intuitively, it is clear that the problem of checking if a graph is an interval graph is closely related to Theorem 9.1 (finding $\Pi(\varphi)$). The problem of recognizing interval graphs can be solved in polynomial time [1]. The algorithm is based on the following theorem [3]:

Theorem 9.2 *(Fulkerson - Gross 1965)*

A graph $G(V, E)$ is an interval graph iff all of the maximal cliques in the graph can be arranged in linear order so that for every vertex the set of all the cliques containing it is continuous.

To solve the consecutive in this ordering interval recognition problem a matrix is formed displaying the connection between the maximal cliques and the vertices:

[2] Of course, one has to modify the algorithm of [1] to detect upon reducing T with S_i, whether it is possible that there are no errors in S_i nor in the sets already in T.


A B C D E F G
* *
S = {B,E}
Apply Rule #2
A C D F G
*
B E
*
S = {B,F}
Apply Rule #3
A C D F G
*
E B
*
Apply Rule #4
A C D G
E B F
* *
S =
{A,C,G,F}
Apply Rule #4
D
E E F
*
A C G
* * *
Apply Rule #2
S = {A,C}
D
E B F
*
G
C A
* *
S = {A,C,F}
Apply Rule #3
D
E B F
*
G
C A
* *
Apply Rule #8
D
E B F G
C A
Apply Rule #4
S = {D,G}
E B F G D
C A

$$M_{i,j} = \begin{array}{cc} 1 & \text{if vertex } i \text{ is in clique } j, \\ 0 & \text{otherwise.} \end{array} \tag{9.1}$$

By using the algorithm of [1] given in section 9.1.2, we try to find a permutation on the cliques order satisfying the requirements of theorem 9.2. Furthermore, such a permutation allows easy computation of a set of intervals corresponding to the nodes.

Note that the construction of the matrix M above uses the following property: An interval graph has $O(n)$ maximal cliques and these cliques can be found in $O(n)$ time.

As mentioned above, solving the unique mapping problem can be done quite easily using PQ-trees in the absence of noise. In the case of either missing edges (probe not identified) or extra edges (probe identified where it should not have been) the resulting graph might not be an interval graph. The problem of creating an interval graph from the existing graph is known as the *interval graph editing problem*. Slight modifications introduce other variants like the *interval graph sandwich problem*.

9.2 Probabilistic Models for Mapping

Recall the following definition:

A *Poisson process* of rate λ is described by:

- A non decreasing function $N : R_0^+ \rightarrow N$ where $N(t)$ = number of events until time t.
- N(0) = 0.
- The number of events in disjoint intervals are independent:

$$P(N(t+s) - N(s) = n) = e^{-\lambda t}\frac{(\lambda t)^n}{n!}, \text{ for } n = 0, 1, \ldots \text{ and } s \geq 0 \tag{9.2}$$

  As a consequence the distribution of the number of events in an interval is stationary, i.e, depends only on the length of the interval. The expected number of events in an interval of length t is given by $E(N(t)) = \lambda t$.

Denote:

- T_n = time between event $n-1$ and event n.
- $S_0 = 0$.
- $S_\imath = \sum_{i=1}^{i} T_\imath$.

Recall that inter-event times in a Poisson process are i.i.d. random variables, exponentially distributed with parameter λ, i.e.,

$$Pr(T_i > t) = e^{-\lambda t} \tag{9.3}$$

If $n \geq 1$ events occurr in a Poisson process until time t, then the inter-arrival times $\{S_1, S_2, ..., S_n\}$ are distributed uniformly and independently in $[0, t]$.

Assume clone length L, genome length G, and choose N clones at random. What is the expected fraction of the genome covered by clones?

For a random point b, and an arbitrary clone C the probability of the point b being included in the clone c is given by:

$$Pr(b \in c) = \frac{L}{G} \tag{9.4}$$

and therefore, the probability of b being out of all the clones is given by:

$$Pr(\forall c : b \notin c) = (1 - \frac{L}{G})^N = (1 - \frac{L}{G})^{G\frac{N}{G}} \sim e^{-\frac{NL}{G}} \tag{9.5}$$

with the last approximation being valid when $L \ll G$ and $N \ll G$.

The fraction

$$R = \frac{NL}{G}$$

is said to be the *redundancy* of the clone set. Since NL is the total length of the clone, the *redundancy* factor is the factor of the genome length to the clone's total length. The redundancy factor is therefore also known as the *Genome Equivalent*.

The expected fraction of non-covered genome is given by

$$E(\text{region not covered}) = e^{-R} \tag{9.6}$$

where R is the redundancy of the clone set. This Formula is also known as the Clark-Carbon formula. Table 9.2 shows that using redundancy factor higher than 2 gives a good coverage of the genome segment considered.

Assume that the clone's length is one, denote N to be the number of clones, and R to be the redundancy factor, also assume that the clone starting positions follow a Poisson process with rate λ. We define a minimal overlap factor θ between clones to identify overlap, that is, two clones are said to be overlapping if and only if they share at least a θ-length section.

A set of clones covering a continuous segment of the genome, together with their physical distances is called a *contig*. Contigs are sometimes referred to as *islands*.

Theorem 9.3 (Lander-Waterman 1988 [4])

R	Coverage
1	0.63
2	0.865
3	0.95
4	0.98
5	0.993

Table 9.3: Coverage of genome segment depending on redundancy factor (R).

1. *The expected number of islands is given by*

$$Ne^{-R(1-\theta)} \tag{9.7}$$

2. *The expected number of apparent islands with exactly $j \geq 1$ clones is given by*

$$Ne^{-2R(1-\theta)}(1-e^{-R(1-\theta)})^{(j-1)} \tag{9.8}$$

3. *The expected number of clones in an apparent island is given by*

$$e^{R(1-\theta)} \tag{9.9}$$

4. *The expected length of an apparent island is*

$$\frac{e^{R(1-\theta)}-1}{R}+\theta \tag{9.10}$$

The connection between the number of apparent islands to the redundancy factor and θ can be seen if Figure 9.9. The connection between the length of apparent islands to the redundancy factor and θ can be seen in Figure 9.10.

Proof: We will prove the first item of the theorem. In order to prove the formula for expected number of apparent islands, we define $J(x)$ as follows:

$$\begin{aligned} J(x) &= Pr(\text{two points } a, b=a+x \text{ are not covered by a common clone}) \\ &= Pr(\text{there are no left-end points in the interval } [b-1,a]) \end{aligned}$$

Since $[b-1,a]$ is of length $1-x$, $J(x)$ can be computed from the redundancy factor R as follows:

$$J(x) = \begin{array}{cl} e^{-R(1-x)} & 0 \leq x \leq 1, \\ 1 & \text{otherwise.} \end{array} \tag{9.11}$$

Scanning from right to left, the number of islands is the number of times leaving a clone without detecting an overlap. Let E_c denote the event of a clone c being the right-hand clone of an island. If the right-hand side of the island is at a point t, we require that t and $t - \theta$ are not covered by a common clone (other than c) .

The probability of such an event E_c is given by:

$$P(E) = J(\theta) \tag{9.12}$$

and the expected number of apparent islands is therefore given by:

$$Exp(\text{number of apparent islands}) = N \cdot J(\theta) = Ne^{-R(1-\theta)} \tag{9.13}$$

■

9.3 Constructing Physical Maps from Noisy Non-Unique Probes Fingerprints

9.3.1 Introduction

Physical mapping using hybridization fingerprints of short oligonucleotides was first suggested by Poustka et al. in 1986 [7]. In this technique short labeled DNA sequences, or probes, attach, or hybridize, to positions along the target DNA matching their own DNA sequence. The probes are nonunique, i.e., they occur at many points along the genome, and typically hybridize with $10\% - 50\%$ of the clones. Overlapping clones can be identified by their similar fingerprints. See Figure 9.12 for an illustration of this hybridization scenario. In Figure 9.13 we can see a typical map that was produced with noisy probes data. [7] suggested this method in order to eliminate the need to process individual clones in the restriction digestion technique. They reported preliminary computer simulations demonstrating feasibility, and suggested the use of Bayesian inference in data analysis. More detailed strategies were offered by Michiels et al. [6]. A likelihood ratio based on a detailed statistical model was used to make overlap decisions, and a discussion of experimental errors was also included. Craig et al. [2] used short oligonucleotides in the ordering of cosmid clones covering the Herpes Simplex Virus (HSVI) genome. The clones were ordered manually. As each probe occurred only once or twice along the short ($\sim$ 140KB) genome, this experiment does not represent the general problem.

The location of the clones along the target genome is not directly known to the experimenters. Mapping data (such as hybridization data) produced by the experiment is used to reconstruct the map. A list assigning every clone its estimated position along the genome is a solution to the mapping problem. According to Theorem 9.3, with sufficient coverage

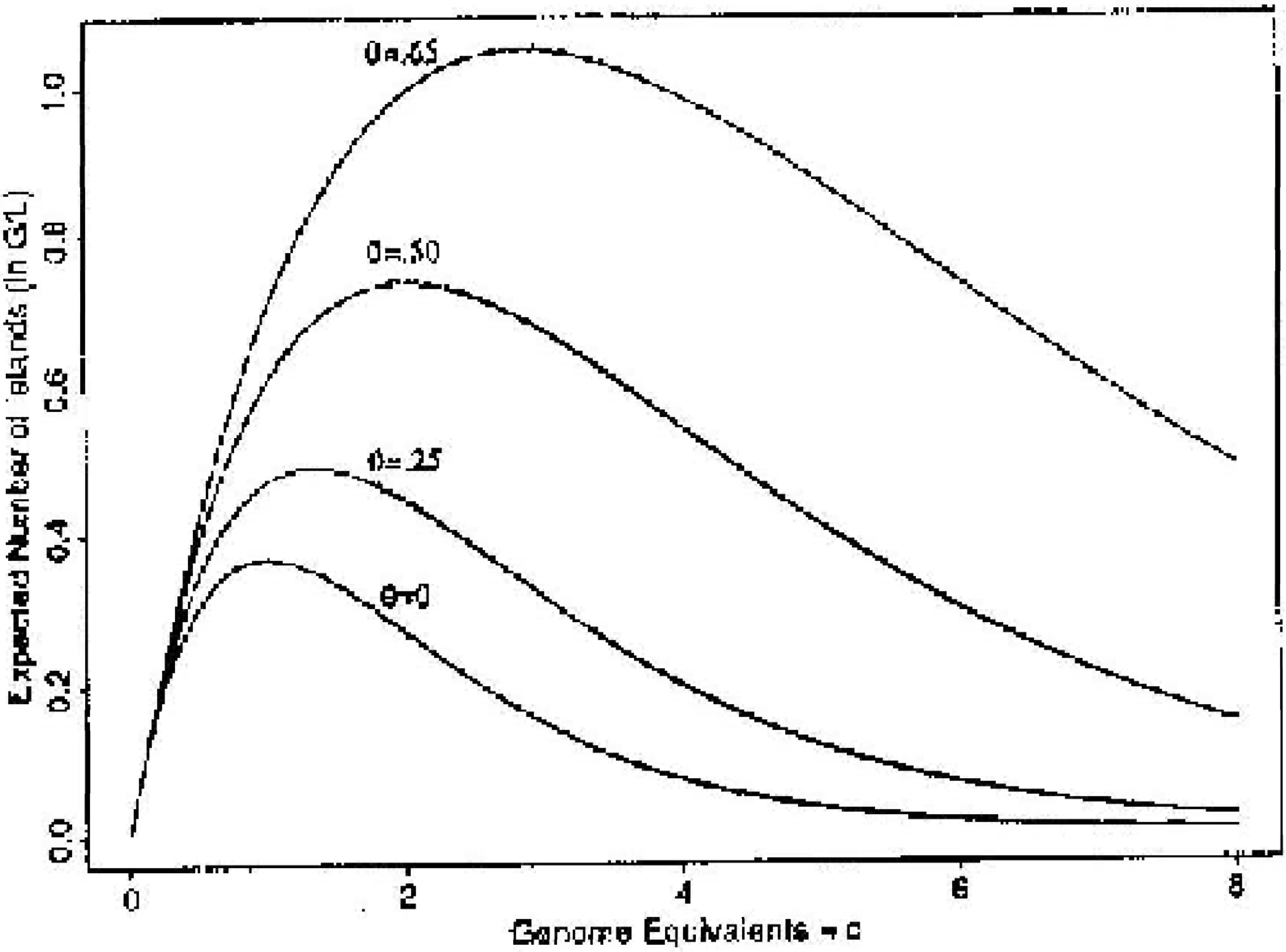


Figure 9.9: Number of islands as a function of genome equivalence and θ.

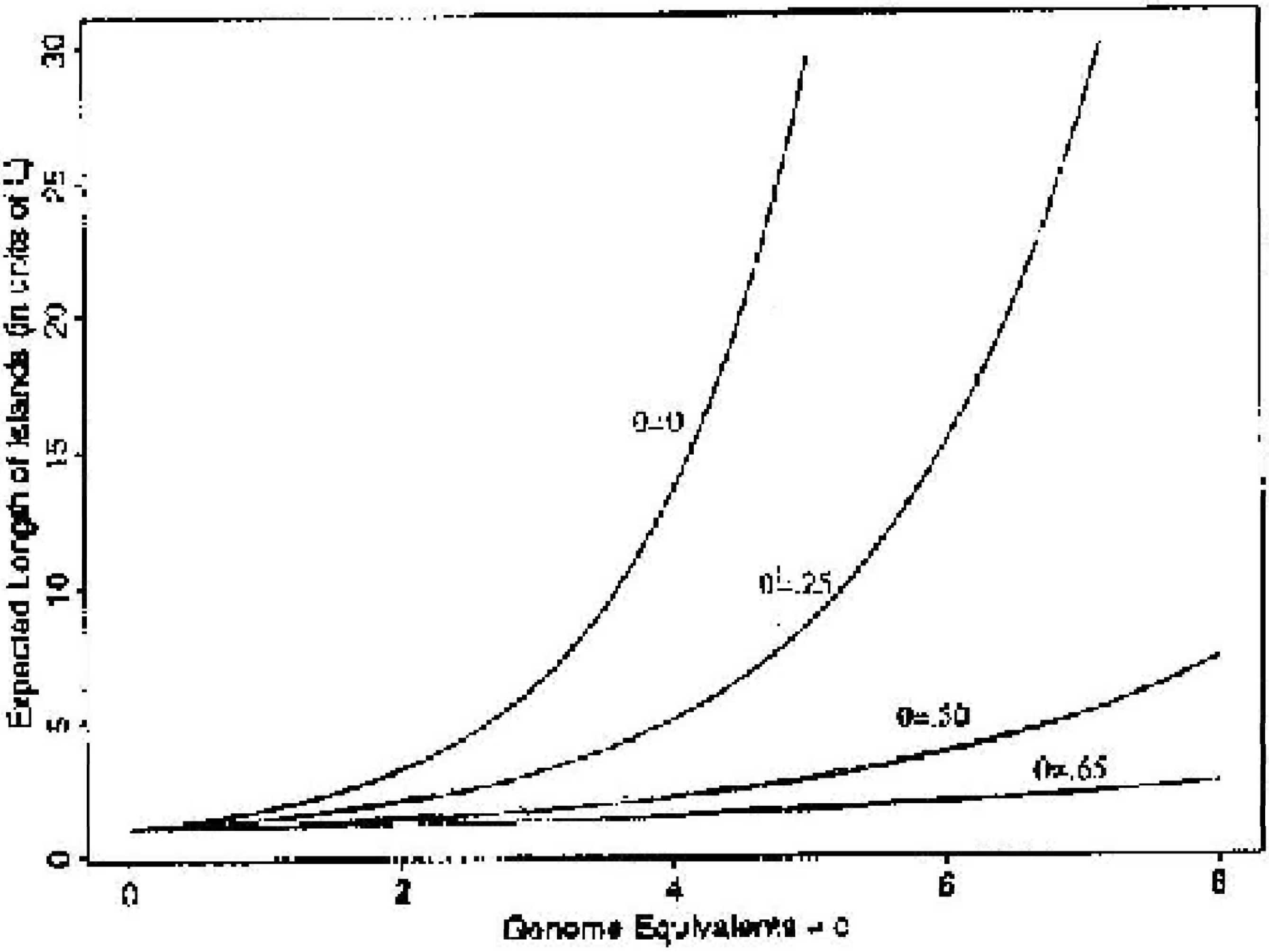


Figure 9.10: Islands' length as a function of genome equivalence and θ.

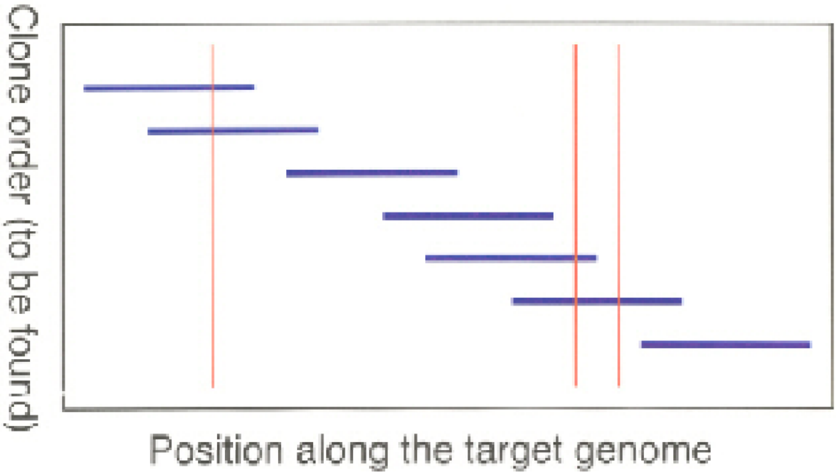


Figure 9.11: Source: [5].Clones and non - unique probes. The clones are the horizontal lines. The random occurrences of a single nonunique probe are marked by the dotted vertical lines.

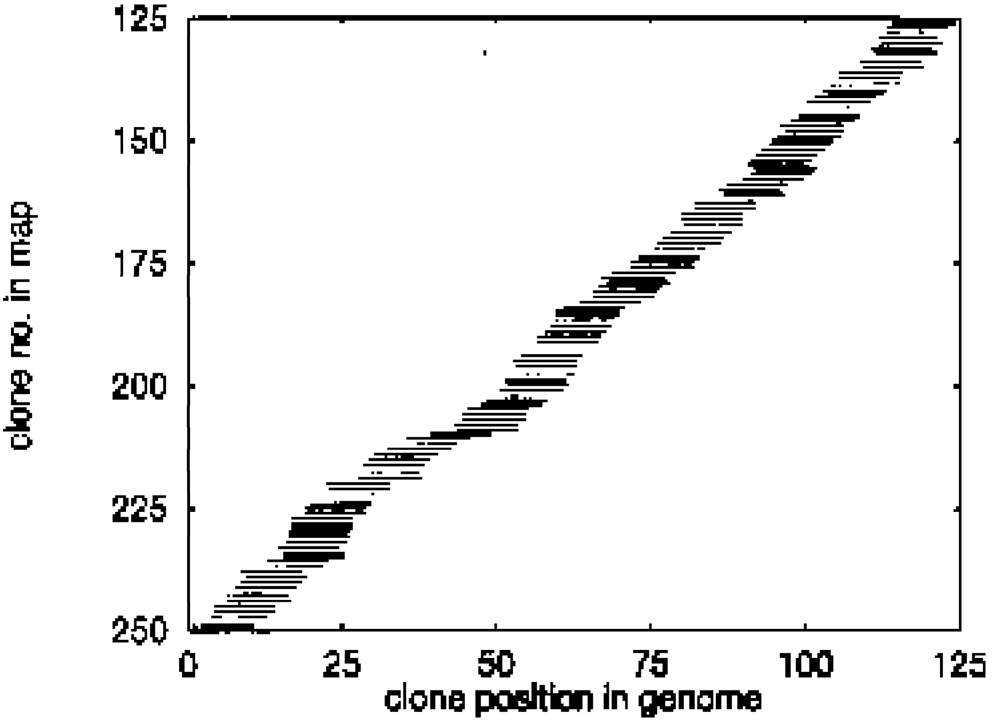


Figure 9.12: Source: [5].Physical (good) map example. The short horizontal lines are the clones with their x coordinates corresponding to the position on the target genome. The y coordinates correspond to the clone order in the constructed map. Note that each point on the target genome is covered by many clones.

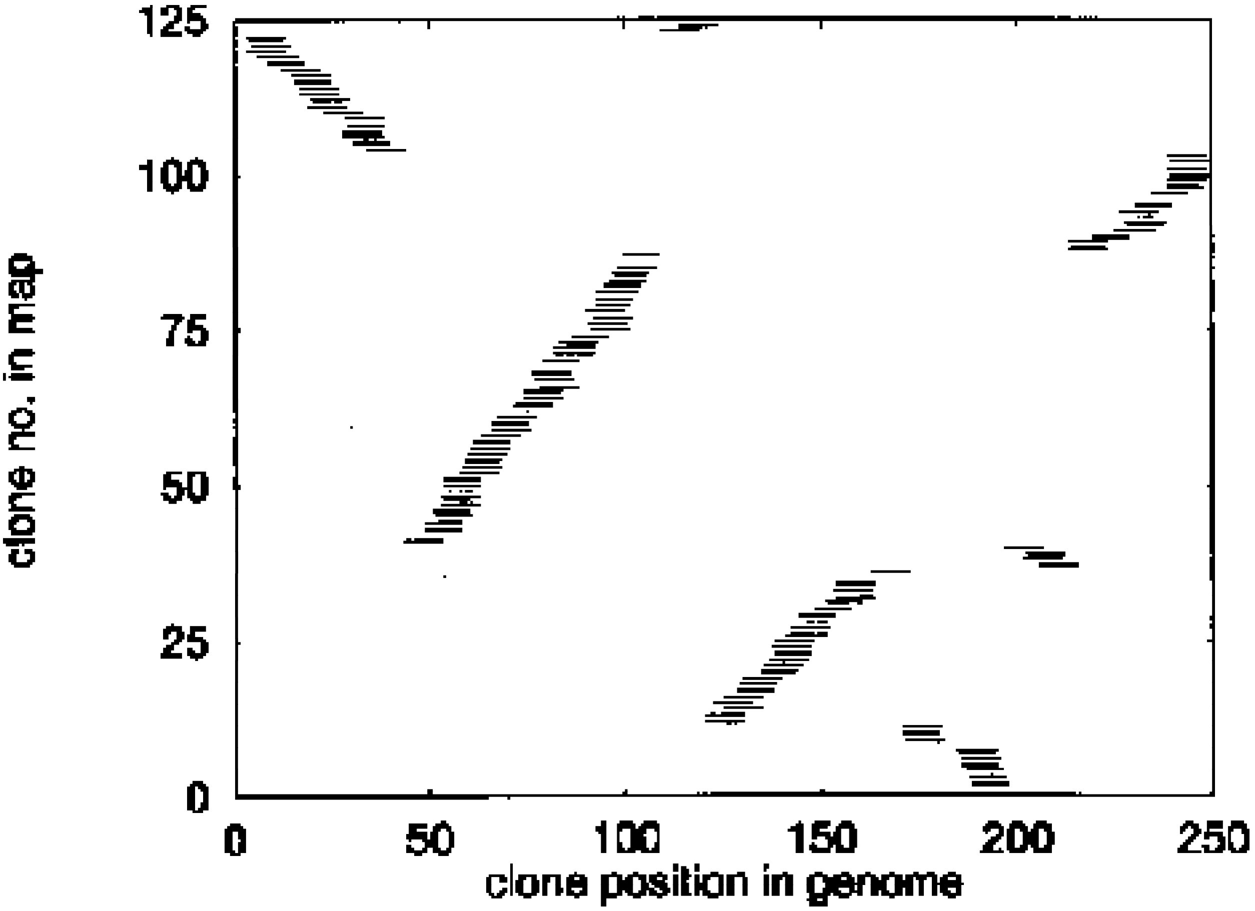


Figure 9.13: Source: [5].Physical (not so good) map example. Due to low redundancy factor (R) or noise with the probes data, we might produce this kind of maps. Most of the data within the eight contigs is correct, but still there are some contigs inversions.

the whole map is usually one contig. A plot of clone order in the constructed map vs. real clone position (see Figure 9.12) provides a visual measure for map quality. If the order of the clones in the constructed map is completely correct then the computed left endpoints of clones increase as their true value increase (or decrease, if the orders in the true and constructed maps happen to be reversed, as in the examples of Figures 9.11,9.12). Minor ordering errors are seen as small deviations from the monotonicity, as in figure 9.11, show the construction is still essentially correct. Very small errors, which do not change the clone order, cannot be observed from the plot. A completely random solution will correspond to randomly placed clones, whereas a non random solution containing several errors(noise) will translate into several randomly placed broken contigs with an approximately correct intra – contig order as in Figure 9.13.

9.3.2 The Statistical Model

We now present the work by Mayraz and Shamir [5] on the physical mapping which uses the same model as in Theorem 9.3 but also copes with noisy data. The statistical model used for the above mapping method assumes the following:

1. Clones are uniformly and independently distributed along the target genome.

2. Clones are of equal length.

3. Probe occurrences along the genome are modeled by a Poisson process.

4. The Poisson rate is identical (with parameter λ) and independent for all probes.

5. The noise statistically behaves as follows:

   - False positive errors - a Poisson process with parameter β.
   - False negative errors independently occur with probability α for each hybridization

The hybridization scenario is shown by Figure 9.11. The clones are the horizontal lines. The random occurrences of a single nonunique probe are marked by the dotted vertical lines. We denote by A the probe - clone occurrence matrix: $A_{i,j} = k$ if probe j occurs k times in clone i. The probe in this example occurs 3 times along the 7 clones in this genomic region, so its column in the occurrence matrix would be $(1, 1, 0, 0, 1, 2, 0)$. The probability of j occurring k times in i is given by:

$$Pr(A_{i,j} = k) = \frac{(\lambda l)^k e^{-\lambda l}}{k!} \tag{9.14}$$

We denote by B the probeclone hybridization matrix: $B_{i,j} = 1$ or $B_{i,j} = 0$ depending on whether probe j hybridized with clone i or not. The vector $\overrightarrow{B_i}$ of the hybridizations of clone i with all the probes is also called its *hybridization fingerprint*. In case no noise is present hybridization occurs iff there is at least one occurrence of the probe. In this case the appropriate column of B would be $(1,1,0,0,1,1,0)$. Experimental noise can result in both false positive hybridizations ($B_{i,j} = 1$ when $A_{i,j} = 0$), and false negative hybridizations ($B_{i,j} = 0$ when $A_{i,j} > 0$).

Hybridization fingerprints of intersecting clones are correlated. This fact is used in order to estimate the clone pairs overlap. Although noise reduces the correlation between fingerprints of overlapping clones, Bayesian inference can still be used to identify overlap, provided a sufficient number of probes is used. It may also be the case that "soft decision" hybridization signals are available. Such signals provide more information on probe occurrences than binary signals do. This continuous signal value does not directly correspond to the hybridization probability, and we have chosen to assume a threshold is used to transform the hybridization signal into a binary one. We therefore define the hybridization matrix B to be a binary matrix, such that $B_{i,j} = 1$ if probe j has produced a positive hybridization signal with clone i. The matrix B is the actual experimental data, which is the input for the construction algorithm. The matrix contains noise and no information on multiplicities. Using the statistical model we can write the following equation:

$$\begin{aligned} Pr(B_{i,j} = 1 | A_{i,j}) &= Pr(\text{false positive}) + \\ &\quad (1 - Pr(\text{false positive}))(1 - Pr(\text{false negative}|A_{i,j})) \\ &= (1 - e^{-\beta l}) + (1 - (1 - e^{-\beta l}))(1 - \alpha^{A_{i,j}}) \\ &= 1 - e^{-\beta l \alpha}\alpha^{A_{i,j}} \end{aligned}$$

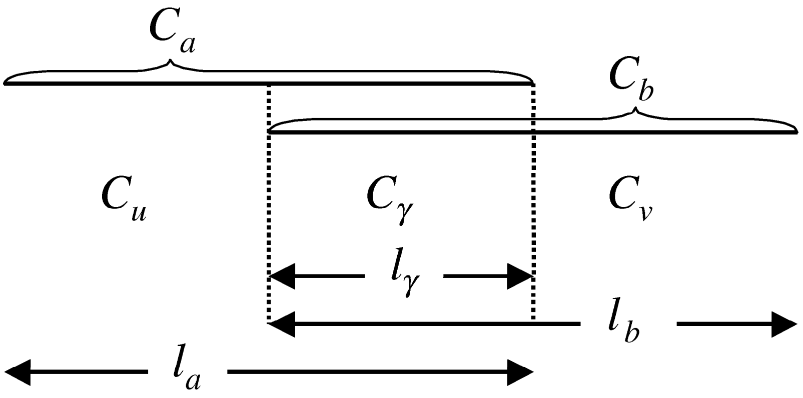


Figure 9.14: Source: [5].Clone pair overlap score.

9.3.3 Problem Statement

Problem 9.3 *The Noisy Non-unique Probes Mapping Problem*
INPUT:

- The probe hybridization matrix B ($B_{i,j} = 1$ iff probe j has hybridized with clone i). The hybridization matrix contains both false positives ($B_{i,j} = 1$ when $A_{i,j} = 0$) and false negatives ($B_{i,j} = 0$ when $A_{i,j} > 0$).
- The genome size.
- The length of the probes (a group of 200-500 oligonucleotides with a typical size of seven bases).
- The length of the clones.
- Estimates of the α, β which represent experimental noise.

GOAL: Find the relative position of the clones.

9.3.4 Clone Pair Overlap Score

Let C_a and C_b be two clones viewed as intervals of the same length l. Define $C_\gamma = C_a \bigcap C_b$ and $l_\gamma = |C_\gamma|$. The relative position of C_a,C_b and C_γ is shown in Figure 9.14. The overlap score uses the hybridization vectors $\overrightarrow{B_a}, \overrightarrow{B_b}$ to produce a vector of probabilities for each length l_γ of the overlap.

We first calculate the probability $Pr(\overrightarrow{B_a}, \overrightarrow{B_b}|l_\gamma = t)$. Let $C_u = C_a \backslash C_b$, $C_v = C_b \backslash C_a$, and recall that $A_{i,j}$ is the number of occurrences of probe j in C_i. We can thus write the following equation:

$$\begin{aligned} Pr(B_{a,j}, B_{b,j} | l_\gamma = t) &= \sum_{K_u} \sum_{K_v} \sum_{K_\gamma} Pr(B_{a,j} | A_{a,j} = K_u + K_\gamma) \\ &\cdot Pr(B_{b,j} | A_{b,j} = K_v + K_\gamma) \\ &\cdot Pr(A_{u,j} = K_u | l_\gamma = t) Pr(A_{v,j} = K_v | l_\gamma = t) \\ &\cdot Pr(A_{\gamma,j} = K_\gamma | l_\gamma = t) \end{aligned}$$

The calculation of the probabilities inside the summation is straightforward using the statistical model. Although $0 \leq K_i \leq \infty$ (with poisson distribution) we can limit the summation to small values of K_i (say $0 \leq K_i \leq 4$) due to the fact that the probability of larger hits by probes is insignificant, thereby making the score computation feasible while introducing only a negligible error. Considering each probe as an independent source of information, the conditional probability of the vector pair $(\overrightarrow{B_a}, \overrightarrow{B_b})$ is:

$$Pr(\overrightarrow{B_a}, \overrightarrow{B_b} | l_\gamma = t) = \prod_{j=1}^{n} Pr(B_{aj}, B_{bj} | l_\gamma = t) \tag{9.15}$$

Assuming uniform parameters for the probes, the expression $Pr(B_{a,j}, B_{b,j} | l_\gamma = t)$ inside the product is independent of j. Therefore, we can define $P_{x,y}[t] = Pr(B_{a,j} = x, B_{b,j} = y | l_\gamma = t)$ $(x, y \in \{0, 1\})$. In practice, instead of computing $P_{x,y}[t]$ for each t in the interval $[0, l]$, we use score quantization (d) of this interval, and perform the computation only for representative values of t $(l_\gamma \in \{0, \frac{l}{d}, \frac{2l}{d}, \frac{3l}{d}, ..., l\}$).

Denoting by $S_{x,y}(a, b)$ the set of probes $1 \leq j \leq n$, such that $B_{a,j} = x$ and $B_{b,j} = y$, we can write:

$$Pr(\overrightarrow{B_a}, \overrightarrow{B_b} | t) = \prod_{x=0}^{1} \prod_{y=0}^{1} P_{x,y}[t]^{|S_{x,y}(a,b)|} \tag{9.16}$$

In practice we use logarithm of the probability:

$$\log Pr(\overrightarrow{B_a}, \overrightarrow{B_b} | t) = \sum_{x=0}^{1} \sum_{y=0}^{1} S_{x,y}(a, b) \cdot \log P_{x,y}[t] \tag{9.17}$$

Having computed $Pr(\overrightarrow{B_a}, \overrightarrow{B_b} | t)$ we can use Bayes Theorem:

$$Pr(l_\gamma = t_0 | \overrightarrow{B_a}, \overrightarrow{B_b}) = \frac{Pr(\overrightarrow{B_a}, \overrightarrow{B_b} | l_\gamma = t_0) Pr(l_\gamma = t_0)}{\sum_{t=0}^{l_a} Pr(\overrightarrow{B_a}, \overrightarrow{B_b} | l_\gamma = t) Pr(l_\gamma = t)} \tag{9.18}$$

$(Pr(l_\gamma = t)$ can be computed by the overlap-target length ratio).

Now, we can give the proper score for every overlap probability.

9.3.5 The Construction Algorithm

1. Initialize a collection of contigs, each of which is a set of clones. Initially, the collection includes for each clone, a contig consisting of that clone.

2. Calculate $P_{x,y}[t]$ for each $x, y \in \{0, 1\}$ and for each representative t in $[0, l]$.

3. Calculate $|S_{x,y}(a, b)|$ for each pair a, b of contigs and for each $x, y \in \{0, 1\}$.

4. Calculate for all the initial contig pairs and for the two possible relative orientations their relative placement probabilities vector. The results are stored in a table.

5. Find for all contig pairs and for their two possible relative orientations their best relative placement, and its probability.

6. While more than one contig remains:
   (a) Find two contigs a, b which have relative orientation and placement that attain the highest probability.
   (b) Merge b into a.
   (c) Change the table calculated in step 4 to reflect the last merge. Only the table entries for all contig pairs (a, c) need to be changed. The required change is a simple combination of the previous entries for (a, c) and for (b, c).
   (d) Find, for contig pairs affected by the merge, their new best relative placement.

Assume that there are n probes and m clones of length d, and the genome has length L. The bottleneck steps in the computation are:

- Steps 2-3 that takes $O(m^2 n)$ time.

- Steps 4-5 that take $O(m^2 d)$ time

- Steps 6(c) and 6(d) that take $O(mL)$ time.

Step 6 is repeated $m - 1$ times thus the total complexity of the algorithm is $O(m^2(L + d + n))$.

Since we can have a large number of clones(m), m^2 still means high time complexity. The authors improved this by initially partition the clones into groups , meaning, identifying sets of clones with high probabilities that each clone in the set will overlaps the other members of the set, and then applying the algorithm first to each group, thus minimizing the amount of contigs (in the initial state of the algorithm, each clone is a contig).

9.3.6 Map Quality

We have used several criteria to quantify the quality of the constructed map, including the presence of large errors in the constructed map, the average size of errors between consecutive clones, and the number of breakpoints in the clone-order permutation. We detail how to evaluate the map quality. We measure the distance between clones a, b as the number of left clone endpoints between the left endpoints of a and b.

Using the order of the left endpoints of clones in the constructed map, we compute the distance d_1 between consecutive clones and compare it with the distance d_2 between the same two clones in the correct map (note that the two clones need not necessarily be consecutive or even close in the correct map). If the difference $|d_1 - d_2|$ is more than a clone's length, we call it a *big error* (see Figure 9.15). Otherwise, it is called a *small error*.

The average of the small errors is also calculated and is called the *average error*. As it is possible that the constructed map is oriented in the reverse direction with respect to the correct one, we repeat the calculation with reversed orientations. The correct orientation is assumed to have a longer clone sequence with no big errors. In case this is not a sufficient criterion (there are no big errors in both orientations, or the first error in both orientations occurs after the same number of clones) we choose the orientation minimizing average error. We then advance to the location of the first big error and select the orientation for the rest of the mapping using the same considerations we used for the first one. This procedure is repeated till the whole map was traversed.

A 0 or 1 variable *anybig* is defined to be a 1 if the constructed map contains at least one big error. Its average over a number of simulations is used to estimate the probability of a constructed map to contain big errors.

In order to avoid wrong mergers of non-overlapping contig and other particularly large errors the authors used the strategic of giving priority to a low estimated probability of no overlap. Instead of sorting contig pairs according to the probability of their most likely overlap, they sorted contig pairs first according to the probability of no overlap. The choice among those contig pairs minimizing the no overlap probability is made according to the probability of their most likely overlap.

Another idea to improve the algorithm was to use 'Simulated Annealing', meaning (in this context), to move (+invert) contigs along the map trying to achieve better scores. But it yielded no improvement.

9.3.7 Main Results

Results presented in this section are for the base scenario, which has the following parameters:

- Length of clones: $l_c = 40960$ base pairs.

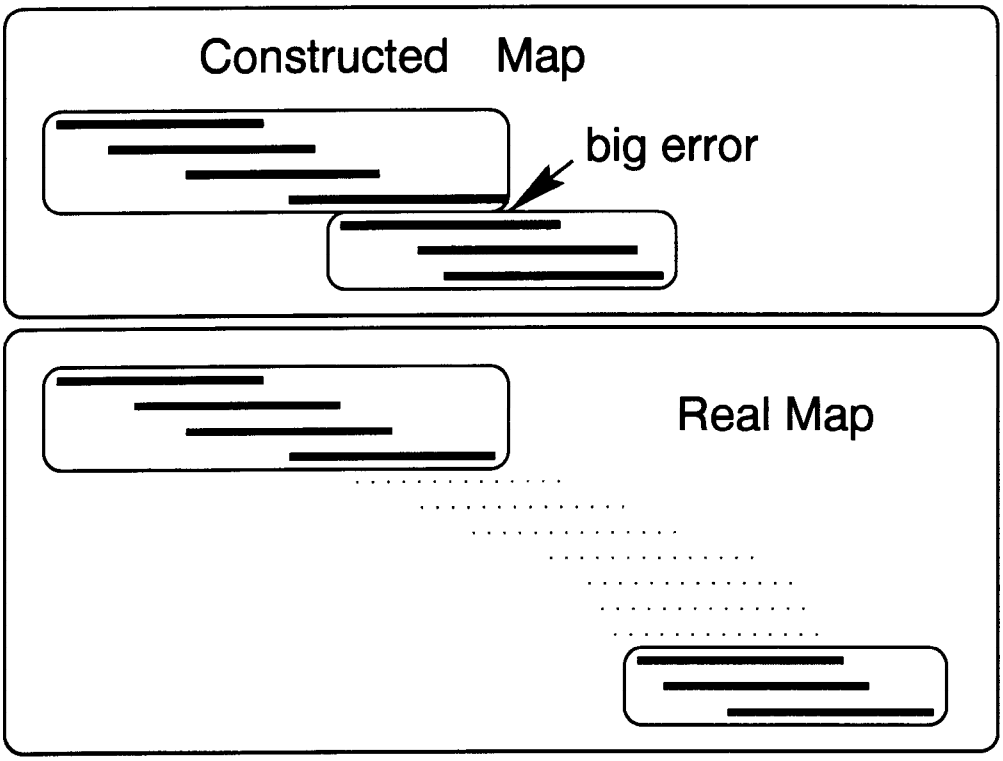


Figure 9.15: Source: [5].A big error in a constructed map. The difference between the actual distance between the two clones and the distance on the map is bigger than the average length of a clone.

- Target genome length: $L = 25l_c$ ($L = 25 \times 40960$ which is approximately 1M base pairs).
- Clone coverage: 10 (The total length of the clones divided by the length of the genome).
- False negatives probability: $\alpha = 0.2$.
- False positives probability: $\beta = 0.05$.
- Number of probes: $n = 500$.
- Length of probes: $plen = 8$.

Based on the results of 1000 simulations, the algorithm has a probability of 0.039±0.006 of making any big errors in the base scenario. The average error in the constructed map is also quite small: $1740 \pm 3bp$ (less than 5% of a clone's length). The average error can be reduced if a finer quantization unit is used (at a linear cost to memory and CPU consumption). A further experiment indicated a probability of about 0.075 of making any mistake that exceeds a clone's length in the estimation of the relative distance between any two (not necessarily adjacent) clones.

Definition The *clone pair energy* of a clone pair (C_i, C_j), with an overlap $o_{i,j} = t$ and hybridization fingerprints $(\overrightarrow{B_i}, \overrightarrow{B_j})$ is given by $E_{i,j} = -\lg Pr(\overrightarrow{B_i}, \overrightarrow{B_j} | o_{i,j} = t)$.

Definition The *contig energy* E is given by summing over all clones in the contig ,i.e : $E = \sum_{i<j} E_{i,j}$.

In analogy to the breaking up of a chemical molecule, the separation of a contig into two nonoverlapping parts should increase the energy substantially. However, if the two parts do not overlap in the real map, the separation energy should be quite small or even negative.

Definition A *weak point* is a point along the contig having separation energy below a threshold (determined experimentally).

Such information can be used by the laboratory in order to pinpoint areas where additional hybridizations should be performed. We also make use of the weak points in our algorithm in order to break up a contig and reassemble it. In case an error was made at an early stage, this process enables the algorithm (though being greedy) to correct its previous error with the benefit of the additional information from other clones added at a later stage. An example of weakpoint detection is described in Figure 9.16.

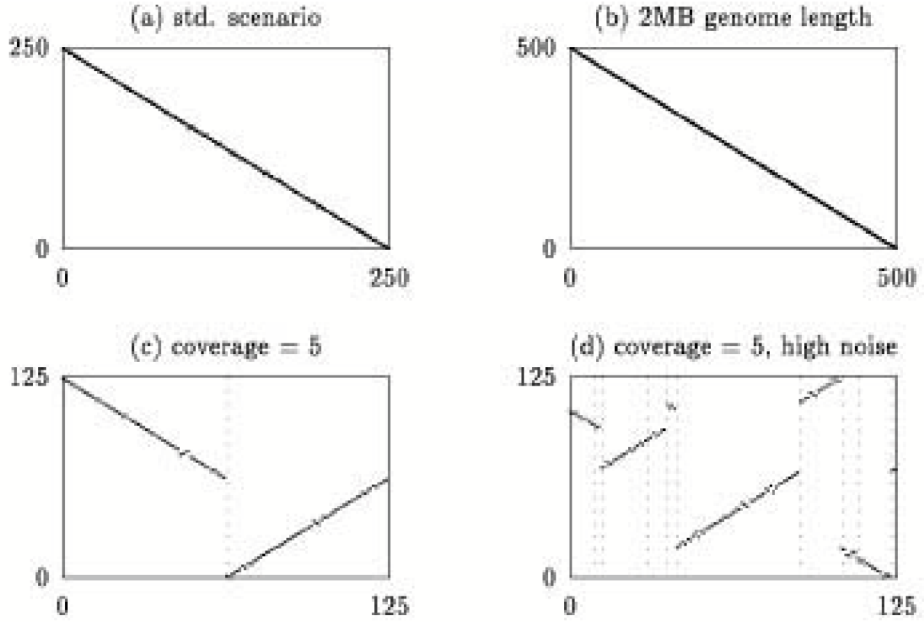


Figure 9.16: Source: [5].True clone order (y axis) vs. constructed order (x axis) in four scenarios. When applicable, weak points shown as vertical dotted lines. The results are taken from the following scenarios: (a) base scenario, (b) a long 2MB genome, (c) a simulation with very low coverage (5), (d) a simulation with very low coverage (5) and very high noise ($\alpha = 0.3$ and $\beta = 0.25$). Note that all big errors were detected as weak points, though some weak points incorrectly suggested additional big errors.

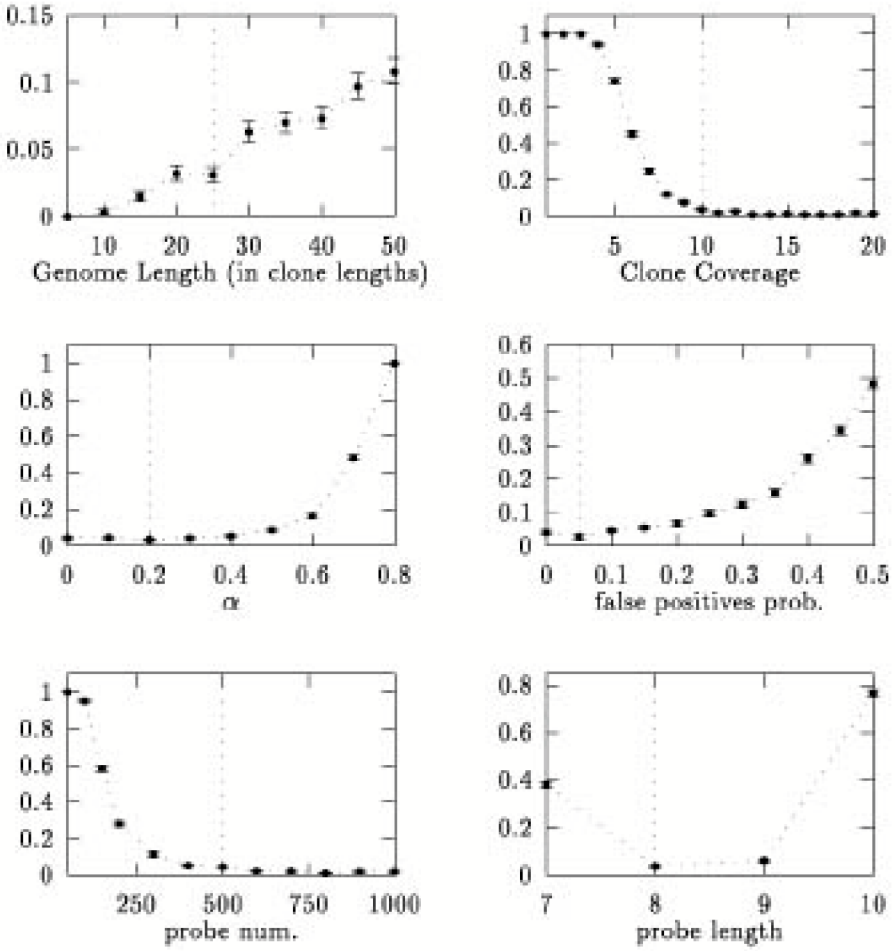


Figure 9.17: Source: [5].Influence of various simulation parameters on the probability of having big errors. The vertical dotted line indicates the value of the parameter in the base scenario. Note that the effect of a decrease in the number of probes is very similar to that of an increase in the experimental noise. This is because noise decreases the informational content of each probe, an effect that can be countered by an increase in the number of probes. It is also notable that the probe size has a very significant effect, resulting from its direct influence on the frequency of probe occurrences, and therefore on the informational content of the experiment. In contrast, the genome size only moderately effects performance.

9.3.8 Results on Real DNA

The next step of the authors was to test the algorithm in simulations involving real DNA sequences. These sequences were taken from a variety of representative organisms: the nematode *C.Elegans*, bacterium *E.Coli*, yeast, and human. The lengths of the sequences ranged from about 1.7MB to over 4MB. Non overlapping sections of length 1MB from each genome were used for the tests (1MB and 730MB for Homo Sapiens). An additional random DNA sequence was used for comparison.

With the exception of probes fitting repetitive sequences, the occurrences of most probes along target genome appear to be uniformly distributed (assumption 3 in Section 9.3.2), thus supporting the Poisson model. However, the same rate assumption also predicts a Poisson distribution of the number of probe occurrences. As Figure 9.19 demonstrates, this stronger assumption cannot be sustained. The obvious solution of fitting a separate Poisson model for each probe will not do, because most probes occur very infrequently, and will therefore provide very little information. However, Figure 9.19 leads us to an encouraging observation: In all the distributions, a significant fraction falls under the graph of the random DNA, thereby demonstrating that there are fairly many ”good” probes. Because probe hybridizations are effort and cost intensive it is impractical to make a large number of experiments and then choose a small subset of ”good” probes.

However, if probes can be chosen before the actual experiment, based on prior knowledge of the organism's typical sequences (e.g., from other sequenced parts of its genome), better results can be achieved.

This process is called *probe pre-selection.* Figure 9.20 demonstrates that a fairly good fit with the same rate model can be achieved. One important problem that cannot be seen clearly in Figure 9.20 is that the resulting distributions still have very long tails. These tails represent probes that either occur very rarely or very frequently, and thus hinder the performance of the algorithm.

To overcome this problem a procedure of *post-screening* is used. It uses the hybridization data to screen out certain probes that deviate significantly from the same-rate model. Only those probes which well fit the same-rate model are subsequently used by the mapping algorithm. As hybridization data is obtained after probe pre–selection, most probes already conform with the model, and so there is little loss of usable hybridization information. The post-screening process uses the number of actual probe occurrences in the hybridization data and the noise parameters. This technique works well when the noise estimate is good. If no good noise estimates are available, one would probably do better by computing a histogram of the number of hybridizations, and keeping only the probes in the central part of the histogram.

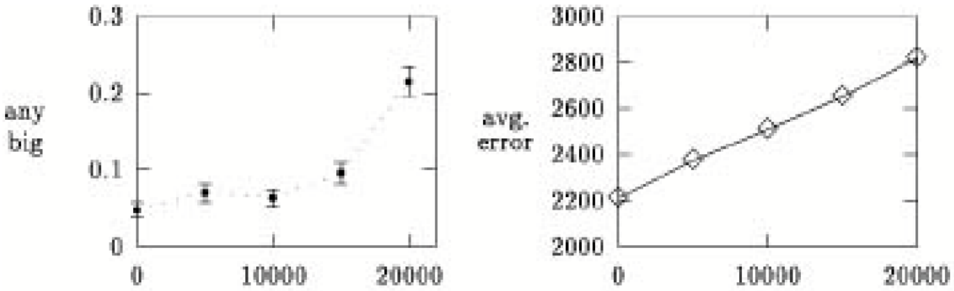


Figure 9.18: Source: [5].Influence of clone length variability. The x axis represents the maximal variability from the average clone sized length. Clone lengths were uniformly distributed in this range. The graphs show good performance with a variability of up to about 15000bp, or 37.5%.

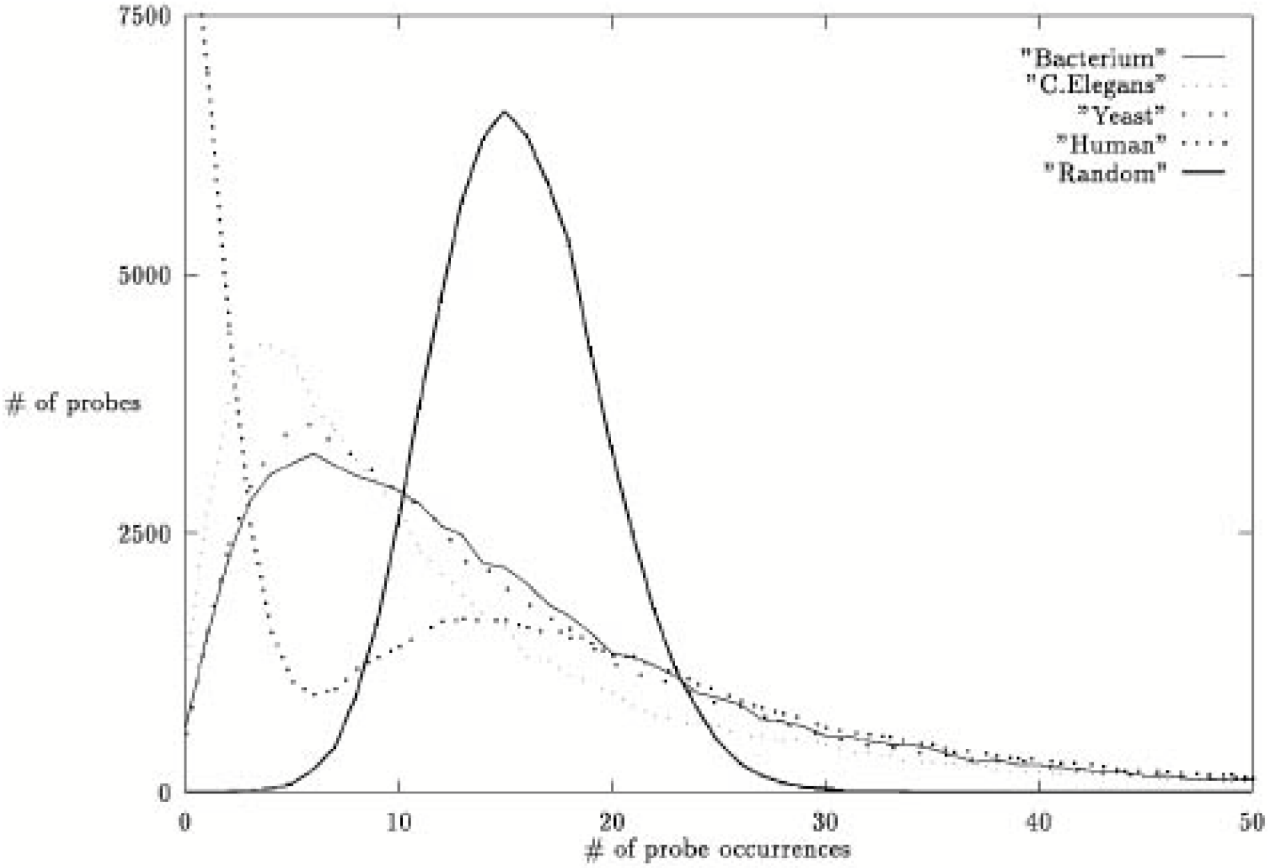


Figure 9.19: Source: [5].Histogram of the number of 8-mer probe occurrences along a genome section of length 1MB.

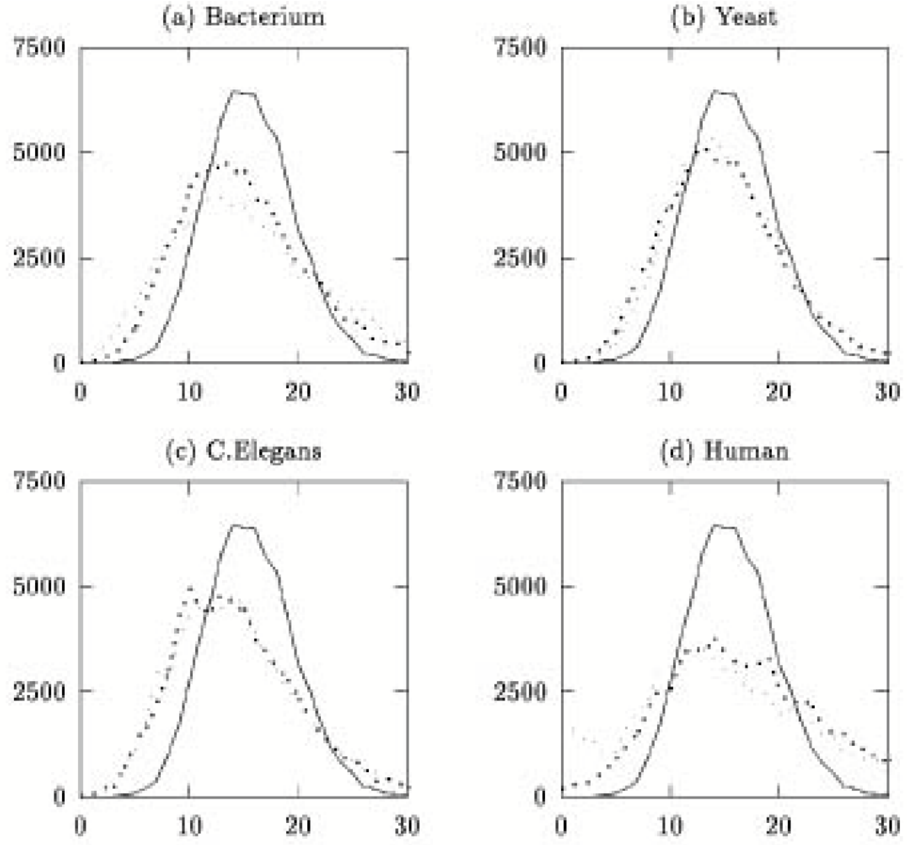


Figure 9.20: Source: [5].Histograms of the number of probe occurrences along a genome section of length 1MB, when using only probes with an average number of occurrences (between 0.9 and 1.1 of average) as estimated from a different genome section of length 1MB (730KB for human) of the same organism.

organism	test type	% maps with errors
Random	no postscreening	2.8 ± 0.7
Random	PS 500 probes before	2.2 ± 0.7
Random	PS 500 probes after	0.0 ± 0.0
Bacterium	no postscreening	60.6 ± 2.2
Bacterium	PS 500 probes before	20.6 ± 1.8
Bacterium	PS 500 probes after	10.8 ± 1.4
Yeast	no postscreening	20.4 ± 1.8
Yeast	PS 500 probes before	4.0 ± 0.9
Yeast	PS 500 probes after	1.2 ± 0.5
C. Elegans	no postscreening	34.6 ± 2.1
C. Elegans	PS 500 probes before	7.4 ± 1.2
C. Elegans	PS 500 probes after	0.0 ± 0.0
Human	no postscreening	88.2 ± 1.4
Human	PS 500 probes before	12.6 ± 1.5
Human	PS 500 probes after	0.0 ± 0.0

Table 9.4: Results of the mapping algorithm with probe pre-selection on real sequence data. The three lines for each organism correspond to: (1) No post–screening, (2) Post-screening on the 500 original probes (leaving about 300 screened probes), and (3) 500 post-screened probes (demands 500 probes, after screening - requiring more to begin with). The screened probes were chosen out of a pre-selected sample of probes occurring within 10% of the mean on a different genome section of the same organism. The post-screened probes are estimated to occur within 10% of the mean frequency on the target genome section too. Averages and standard deviations are based on 1000 simulations in each scenario. PS: postscreening.

Bibliography

[1] K. S. Booth and G. S. Lueker. Testing for the consecutive ones property, interval graphs, and planarity using PQ-tree algorithms. *J. Comput. Sys. Sci.*, 13:335–379, 1976.

[2] A. G. Craig, D. Nizetic, D. Hoheisel, G. Zehetner, and H. Lehrach. Ordering of cosmid clones covering the herpes simplex virus type I (HSV-I) genome: A test case for fingerprinting by hybridization. *Nucleic Acids Research*, 18:2653–2660, 1990.

[3] D. R. Fulkerson and O. A. Gross. Incidence matrices and interval graphs. *Pacific J. Math.*, 15:835–855, 1965.

[4] E. S. Lander and M. S. Waterman. Genomic mapping by fingerprinting random clones: A mathematical analysis. *Genomics*, 2:231–239, 1988.

[5] G. Mayraz and R. Shamir. Construction of physical maps from oligonucleotide fingerprints data. *Journal of Computational Biology*, 6(2):237–252.

[6] F. Michiels, A. G. Craig, G. Zehetner, G. P. Smith, and H. Lehrach. Molecular approaches to genome analysis: a strategy for the construction of ordered overlapping clone libraries. *CABIOS*, 3(3):203–210, 1987.

[7] A. Poustka, T. Pohl, D.P. Barlow, G. Zehetner, A. Craig, F. Michiels, E. Ehrich, A.M. Frischauf, and H. Lehrach. Molecular approaches to mammalian genetics. *Cold Spring Harb. Symp. Quant. Biol.*, 51:131–139, 1986.