

18.417 Introduction to Computational Molecular Biology

Lecture 2: September 16, 2004

Lecturer: Ross Lippert

Scribe: Jerome Mettetal

Editor: Jerome Mettetal

Brute Force Algorithms: Motif Finding

Introduction

Although some problems in biological systems can be solved with very simple searching algorithms, the large search space can cause the run times to grow exponentially with system size. To combat this problem, it is usually possible to use an understanding of the constraints of the search space to cleverly design algorithms that produce reasonable run-times when compared to the size of biological systems.

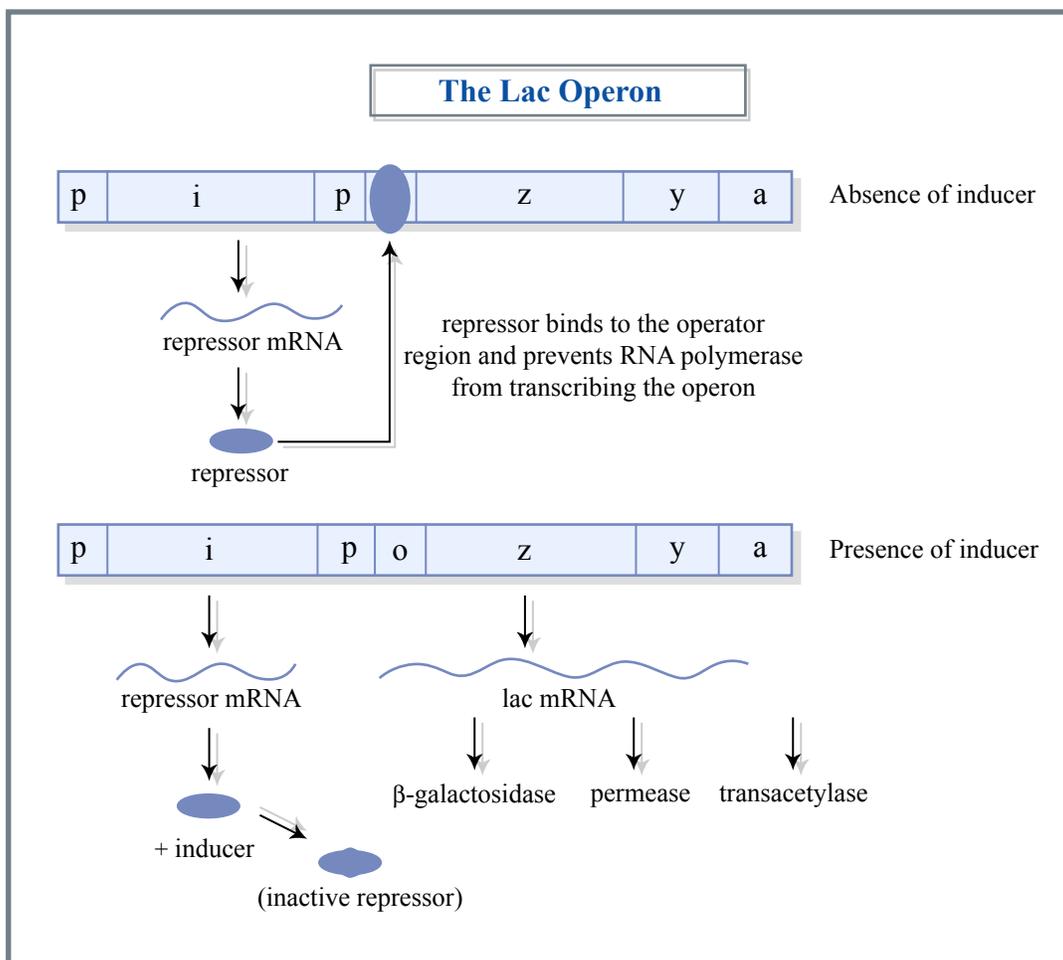
In the previous lecture we examined two algorithms for solving the partial digest problem. The Brute Force method searches through every possible set of $(n - 2)$ restriction sites for original string consisting of $\binom{n}{2}$ elements until the digest set L is produced. This is accomplished by using the place and select functions to create strings of length $(n - 2)$. The run time of this algorithm however goes as $O(W^{(n-2)})$ where W is the length of the original string.

By realizing that the largest element in L will be the length of the original string, and that the next largest elements in the set will be distances from the restriction sites to the ends of the original string, we created a new algorithm called Branch and Bound. This reduces the run time to $O(n^2)$. In the following sections we will discuss a similar approach taken to the problem of motif finding by first outlining the biological relevance of the problem, then generating a simple yet slow algorithm, and finally refining it to run on practical time scales.

Gene Regulation In Biology: Lac and Trp Operons

Now we turn our attention to the problem of motif finding in DNA sequences, but first we must first understand a few of the methods that biological systems use to control the flow of information. DNA contains the data necessary for the production of proteins, but cells need a way to control the rate at which this process happens. One

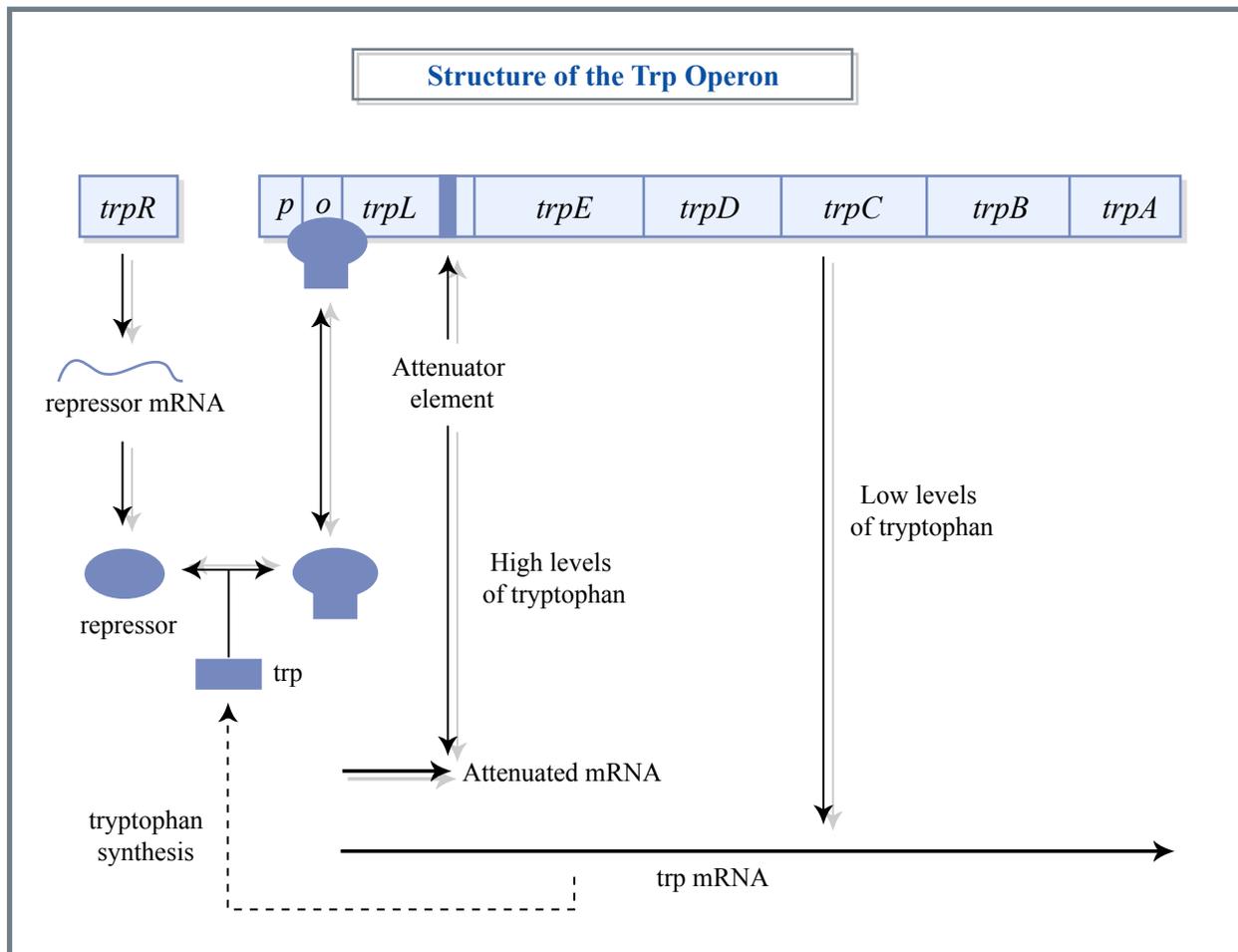
method for doing this is to produce other proteins that will bind to the DNA in the region upstream of the gene to be regulated. An example of this process occurs in the Lac Operon of *E. Coli*. The cell produces a steady supply of the upstream regulator that we call the 'repressor', which binds to a short sequence of nearby DNA labeled region 'o' (fig 2.1a). This prevents the RNA Polymerase from binding and producing downstream proteins 'z', 'y', and 'a' which are associated with the metabolism of lactose. However when the cell is put into a lactose rich environment, the lactose will bind with the 'repressor' protein and prevent it from binding to region 'o' which allows for the transcription of DNA into RNA and the subsequent production of Beta-galactosidase, permease, and transacetylase (fig 2.1b). This gives a net effect that the cell will only produce lactose-digesting enzymes when lactose is present to be utilized.



Adapted from Figure 2.1: Lac Operon In *E. Coli*

The Trp Operon of *E. Coli* presents another method through which regulation occurs. As transcription occurs, there is a pause when a ribosome can attach to the mRNA and begin translation. As transcription progresses, a region of the mRNA is reached in which tryptophan is required to progress. If the level of trp is high, then the step

proceeds quickly, if it is low, the step proceeds slowly. Depending on the speed at which transcription and translation occur simultaneously, two secondary structures of mRNA are produced: one allows for the completion of trp mRNA while the second will not allow the ribosome to progress and the production is stalled. These secondary structures require that 'complementary' sequences are proximal on the mRNA so that the polymer may fold upon itself in a stable way.



Adapted from Figure 2.2: Trp Operon In *E. Coli*

Eukaryotes are much more complex organisms than prokaryotes such as *E. coli*, and therefore have developed much more complex methods of gene regulation including modification of chromatin structure, RNA transport, RNA stability, and expression of introns and exons within genes. The most relevant method is still related to upstream binding sites acting to promote or repress the expression of an entire gene. An interesting note on this topic is that protein sequences and combinations between species are very similar and that most differences arise in response to regulation levels of each protein.

Motifs and Profile Matrices

The question to address at this point is one of computationally predicting transcriptional binding sites given several segments of DNA. To begin we must first understand what similarities the binding sites must have and then search upstream regions for common motifs. A motif is defined to be a small length of code that occurs frequently in a DNA sequence, but it is not required to be an exact copy (i.e. we allow some of the bases to differ between the occurrences). The differences between copies of a regulatory motif cause differences in regulation rates, which can lead to either beneficial or detrimental behaviors. This is in stark contrast to the restriction enzymes discussed in the digestion problem where cutting the DNA in the wrong place even on rare instances will most likely result in death.

Motif Finding Premises

- Start with a collection of upstream regions and suspect a motif is present
- Locations of the motifs are unknown
- Have an idea of the number of bases included in motif (usually 6 – 15 bases)
- Expect that the strings should look very similar

To analyze the sequences, we align the DNA sequences $\{S_1, S_2, \dots, S_k\}$ along the rows of a $k \times n$ table called an Alignment Matrix. Each string is positioned so that the first element in row j is the s^{th} element in string j . From this a $4 \times n$ Profile Matrix can be created by counting the number of times each base $\{\text{A}, \text{T}, \text{C}, \text{G}\}$ appears in each column. The Consensus Sequence is then defined by taking the base with the highest occurrence from each column with the consensus score is defined as the sum of the number of times the consensus base appears in each column. The best possible consensus score is kn while the worst score possible is $\frac{kn}{4}$. We now want to find the best profile and consensus for the set of k strings.

Alignment	Position							
	i	j	k	l	m	n	o	p
1	A	T	C	C	A	A	C	T
2	G	G	G	C	A	T	C	T
3	A	T	G	G	A	A	C	T
4	A	A	G	C	A	A	C	C
5	T	T	G	C	A	A	C	T
6	A	T	G	C	C	A	T	T
7	A	T	G	G	G	C	C	T
Profile	i	j	k	l	m	n	o	p
A	5	1	0	0	5	5	0	0
C	0	0	1	5	0	1	6	1
G	1	1	6	2	0	0	0	0
T	1	5	0	0	2	1	1	6
Consensus	A	T	G	C	A	A	C	T

Motif Finding Problem

- Idea: Find profile that matches well to each string
- Idealization: We can guess the length of the substring. We score the profile matrix based on the consensus.
- Input: Sequences with suspected common binding sites
- Output: Profile of the binding site

More accurately we will conduct a search over locations $\{s_1, s_2, \dots, s_k\}$ with $1 \leq s_i \leq L_i - n$ and find the consensus sequence that gives the best score.

Brute Force Method

The brute force approach is simply to iterate over all $(L_1 - n) \times (L_2 - n) \times \dots \times (L_k - n)$ such starting positions $\{s_1, s_2, \dots, s_k\}$ and keep the sequence whose profile matrix yields the lowest consensus score. Since we need to evaluate on the order of L^k matrices each with $n \times k$ elements, the run time of this method grows as $T = O(nkL^k)$ which is exponential in the number of DNA sequences we wish to examine. Thus this method works when only a few sequences were being compared.

Modified Approach: Guess Profile String

This method is based on guessing a sequence, then finding the substring producing the closest match in each of the k strings. The profile matrix can then be produced from these substrings and a consensus with score will arise (note that the consensus string should be the same as our initial guess).

Median String Problem:

- Idea: Find consensus string that aligns well to each full string
- Idealization: We can guess the length of the substring. We score the profile matrix based on the consensus.
- Input: Sequences with suspected common binding sites
- Output: String minimizing alignment scores

It is convenient to define the distance $d(s_1, s_2)$ between any two strings s_1 and s_2 to be the number of mismatched elements between the two strings. This leads to a relationship between the score and distance between the best matches in each of the k strings.

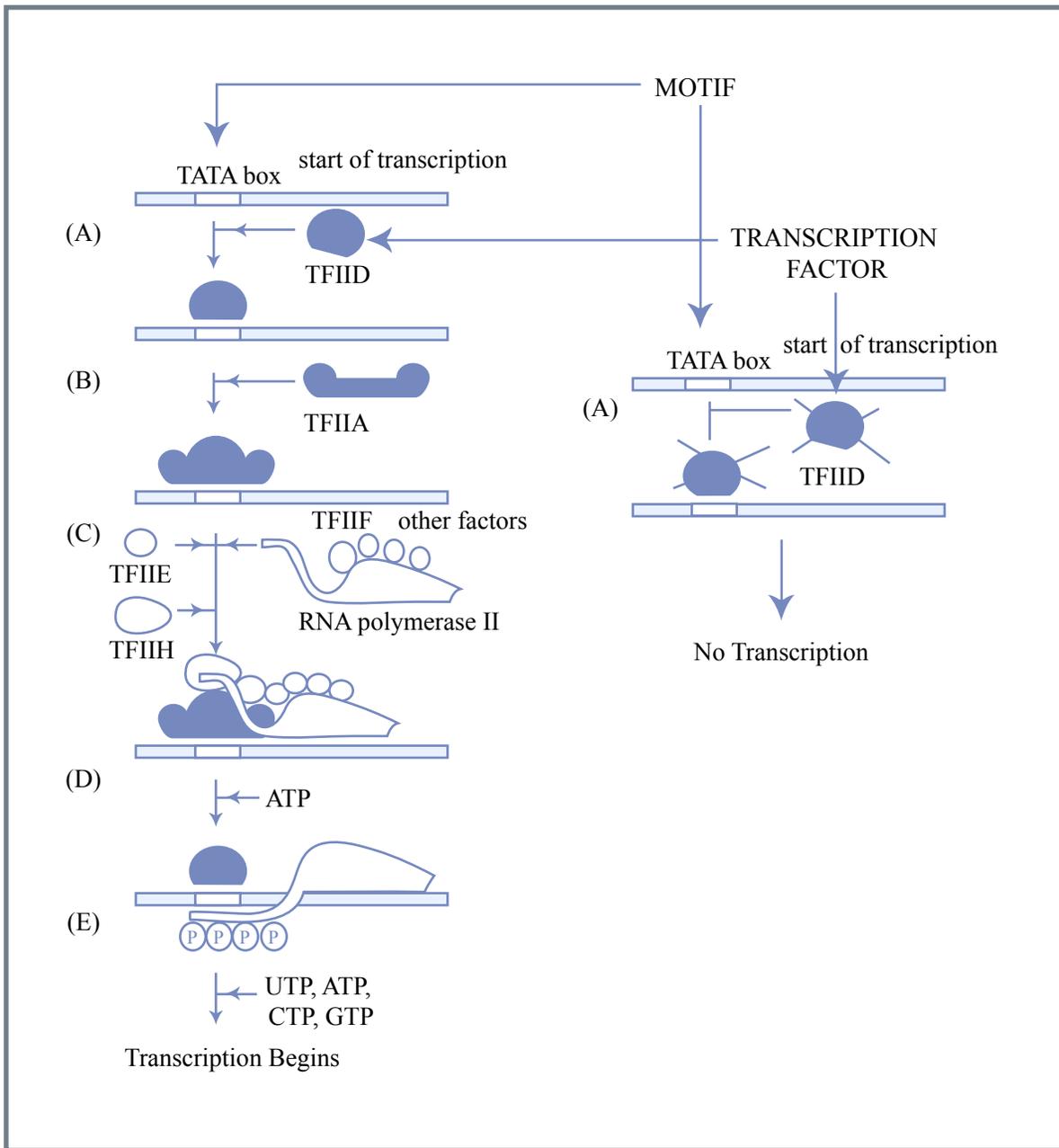
$$\sum d_i = k \times n - \text{score}$$

Now instead of searching over all starting locations $\{s_1, s_2, \dots, s_k\}$ we simply search over all possible substrings $[\text{ATCG}]^n$. This causes the run time to become $T = O(nLk4^n)$ which is no longer exponential as L^k , but grows exponentially in sub-sequence length n .

As in the digest problem we can now improve this algorithm by noting that the distance between any portion of a string and the same portion of our guess is going to be less than the distance from the full string to the full guess.

$$d(\text{AGT}, s) \geq d(\text{AG}, s)$$

This means that at any point in our search tree, if we find that a node gives an optimistic distance greater than the best distance obtained so far, we can quit searching through that node and all branches below it. Although this does not improve the worst-case speed of the algorithm, it usually works in practice to provide an increase in speed.



Adapted from Figure 2.3: Eukaryotic Motif