DEFINITE-CLAUSE GRAMMARS FOR THE ANALYSIS OF *CIS*-REGULATORY REGIONS IN *E. COLI*

D. THIEFFRY[1], D.A. ROSENBLUETH[2], A.M. HUERTA[1], H. SALGADO[1],
and J. COLLADO-VIDES[1*]

[1]*Centro de Investigación sobre Fijación de Nitrógeno. Universidad Nacional Autónoma de México. Cuernavaca A.P. 565-A, Morelos 62100, México*

[2]*Instituto de Investigaciones en Matemáticas Aplicadas y en Sistemas. Universidad Nacional Autónoma de México, Ciudad Universitaria, México D.F., 04510, México*

* *Corresponding author*

Based on an extensive collection of σ70 associated regulatory mechanisms, a grammatical model has been constructed that define the functional positions and combinations of sites within DNA regulatory regions. The syntactic rules and the dictionary implemented in a Prolog program were coupled to consensus matrices used as "sensors" to integrate a syntactic recognizer. A systematic comparison between the syntactic recognizer and the standard weight matrix methodology is presented using 12 regulatory proteins and the whole collection of about 130 σ70 DNA regulatory regions. On the average an increased sensitivity of 5 to 10 fold is obtained with this novel approach.

1. Introduction

As DNA sequence data are accumulating at a fast pace, matching them with their functional characterization is becoming more urgent. Besides the established experimental methods, one thus looks more and more for theoretical methods able to automatically generate a preliminary, trustable, characterization of new sequences (see, for example, EBI's GeneQuiz [3] and GeneCrunch [20]). In this respect, the characterization of transcriptional regulatory sites is of particular importance. However, even though the DNA binding sites of many regulatory proteins have already been thoroughly experimentally determined, it is still difficult to accurately predict the occurrence of similar sites in new sequences.

In the seventies, the characterization of *cis*-regulatory sites involved the definition of consensus sequences. But, in most cases, these consensus sequences performed poorly in discriminating true sites from the genomic background. These consensus sequences were soon significantly improved by taking into account the frequency of each base in each site position ("consensus matrix"), sometimes including the possibility of gaps and/or position correlation [1, 10, 11, 13-15, 17, 21-23, 27, 30]. However, in spite of being mathematically well grounded, these methods are still often clumsy in discriminating true from false sites. A common drawback of all these programs results from a compromise between high specificity (i.e., recognition of only true sites) and high sensitivity (i.e., recognition of all true sites).

Seaching for improvements of these methods, several authors are developing sophisticated scoring schemes or filters. With the same goal in mind, we prefer to keep the consensus definition rather simple, but to take advantage of additional biological information that restricts the significant sites to certain positions and combinations of them. The biological principles of the σ70 *E.coli* promoter collection have been formalized in the form of a grammar that uses transformational rules and terminal symbols corresponding to the complete protein binding sites within a regulatory DNA region [4-8].

In this paper, we attempt to show how the combination of syntactic rules and consensus matrices allows a drastic selection of the sites matched by a given consensus matrix. Since the sites identified by the parser must occur at positions and/or in combinations which are known to be functional in other sequences, they are more likely to be functional than other matching sites. We summarize the results obtained for a set of 12 σ70 associated regulatory protein, chosen among those having the highest number of characterized sites and the best established consensus matrices according to their random expected frequencies.

2. Methods

DNA sequences for the regulatory domains of σ70 promoters were obtained from GenBank. The precise location of the initiation of transcription was assigned after comparison of (i) the GenBank file, with (ii) the position indicated in the literature, and with (iii) the position indicated in a review paper [16].

Scripts were written in Perl [28] to generate the following set of sequences:

1) A set of 128 extended and oriented σ70 associated promoter sequences, covering from around -200 to +40, with respect to the annotated transcription starting site (+1) of each promoter. Each sequence corresponds to a line in Figure 1 of [12] where the position and regulatory function for each site has been experimentally determined.

2) A collection of functional binding sites for 39 transcriptional regulatory proteins, mostly coming from our σ70 promoter collection. Sequences for binding sites that regulate one or more parallel promoters are included only once, whereas sequences for sites that regulate two divergent promoters are included twice, one sequence being the inverted complement of the other.

The matching of weight matrices and DNA sequences discussed below was always done with the set of 128 oriented promoter sequences. The input for generating the weight matrices of the different regulatory proteins was a collection of files, each with the set of all extended functional sites for a given protein. These extended sites were extracted from the set of functional sites mentioned above, using the central positions obtained from the literature. Their size was that reported in the literature plus six bp on each side (see Table I in Collado-Vides 1993b).

A multiple-alignment program that selects the alignment matrix optimizing information content, “wconsensus”, was used to generate a set of best matrices. Wconsensus determines ungapped multiple alignments of unknown prior width [14, 15]. The alignment matrix selected for each protein was the one with the lowest expected frequency that includes all the sites. Once the matrix and the aligned sequences for each protein were obtained, we re-computed the new central positions for each sequence in relation to the +1 of the respective promoters, and used them as reference in the following computations. Another program, developed in the group of G. Stormo, “patser”, is then used to score DNA sequences with the matrices obtained [25]. In all runs of wconsensus and patser, we used equal base frequencies and the default parameters.

The grammatical model used has already been published elsewhere [4-8]. Briefly, the principles incorporated into the grammatical model are: i) the existence of a proximal obligatory site for any promoter, ii) binding sites in a given promoter can be grouped into sets that work coordinately in a mechanism of regulation, and iii) each regulator can bind to a characteristic set of functional positions. In addition, we include a summary of the corresponding rules in appendix 1a and an example of a parse tree for a “promoter sentence” in appendix 1b. For a more detailed description of the methodology, see the references.

This model has been implemented using the programming language Prolog. As Prolog was designed as a language for programming natural-language applications [24], it is therefore specially suited for describing grammars like ours, that involve non context-free dependency relationships. A short description of our implementation is given in appendix 2.

3. Results

We developed a series of Perl scripts that perform the following tasks automatically:

1) To run “wconsensus”, and extract: (i) the best alignment (i.e., with the lower expected frequency), and (ii) the corresponding consensus matrix for each of the 39 sets of cis-regulatory binding sites.

2) To generate a table summarizing the main characteristics for each protein of the alignments obtained: (i) the number of sequences used, (ii) the alignment/matrix width, and (iii) the corresponding sample size adjusted information content [14].

3) To run patser to calculate the lowest matching scores among each set of original sites; these scores are then included in a table, later used to fix the

thresholds (i.e., the lower site scores) when searching for matching sites in putative regulatory sequences.

4) To run patser with pre-defined scoring intervals for each consensus matrix, counting the number of matching sites, and saving these figures in a table.

5) To repeat again step 4 for each matrix, but this time looking for matching sites at positions and in combinations pre-defined by the syntactic rules.

Here, we present the results obtained for 12 proteins, selected among 39 because they correspond to both a significant number of sites and a reasonably good consensus matrix. We used these 12 matrices to compare the performances of their sole use vs. their combination with syntactic rules (points 4 and 5). Note that, for each protein, only a small subset of the 128 promoter sequences were used to generate the corresponding dictionary, whereas the test sequence set always consists of the whole collection of 128 oriented promoter sequences.

The results are summarized in Table 1. It can be seen that the sets of protein sites are quite heterogeneous. Indeed, the information content corresponding to the best alignments/matrices of the 12 sets of sites covers a wide range of scores, from 5.80 to 19.59 bits. For each of the 12 selected proteins we give the number of sites found per score interval, first using the grammatical recognizer combined with patser, and then using patser only. The number of original sites falling in each scoring interval have also been included for comparison.

In some cases, including FNR, LexA and PurR, the consensus matrix alone is already a good discriminant of the corresponding regulatory sites. Not surprisingly, these three proteins are related to consensus matrices with high information content. However, the ArgR and PhoB consensus matrices are also characterized by a high information content but are much less selective. This is because these two proteins involve weak sites that lower the threshold.

In the last column of each sub-table, one can compare the selectivity of both methods through the corresponding (matching sites/true sites). In most cases, it can be observed that the combination of syntactic rules with consensus matrices is about 5 to 10 times more selective. This is already a significative amelioration which could still be improved by sharpening either the matrices and/or the syntactic rules.

Looking at the scoring classes in more detail, it can be seen that most of the time both methods give the same number of matching sites for the highest scores, usually very close to the number of functional sites. This might imply that selective pressure forbids such strong sites at wrong positions.

Thus, the syntactic rules further select the consensus matching sites, especially in the case of low score classes. In other words, the grammatical rules are particularly useful to eliminate weak presumptive sites.

4. Conclusions and perspectives

Cis-regulatory syntactic rules summarize functional information which is not included in the widely used consensus matrices. The combination of both types of information in a simple algorithm proves to be quite powerful, especially in the case of poorly conserved sites, increasing selectivity by a factor of 5 to 10. With the simple matrices and rules used here, significative numbers of false positive sites are still found in some cases, including those of ArgR, CRP, MetJ, PhoB, and TyrR. A deeper analysis of each of these cases might help us derive more selective algorithms (a preliminary analysis of ArgR, LexA and TyrR cases can be found in [19]). However, the remaining "false positive" sites are to be evaluated carefully. Certainly, these sites are more likely to be functional, as they occur in known functional positions relative to the transcription start site, as well as in known functional combinations (occurrence of a proximal obligatory site, etc.).

The main goal of implementing the syntactic recognition of regulatory regions is to apply it to reveal potential regulatory signals in new unannotated DNA sequences. We have initiated a systematic analysis of a collection of about 300 new putative promoter sequences. In these cases, the transcription start site ("+1") is already characterized and the syntactic recognizer can be directly used. In other cases, specific programs have to be used to first localize putative +1 sites. In collaboration with G. Hertz (University of Colorado, Boulder, USA), we are working on a combination of the syntactic parser together with a specific σ70 promoter recognizer. This strategy may help to strengthen the predictive power of standard recognition methods of promoters thanks to the additional biological information that is incorporated into the syntactic recognizer.

However, definitive answers could only come from experiments. In this respect, it would be interesting to compare our results with, for instance, observations on gene regulation derived from 2-D gel global protein analyses performed in *E.coli* [26] and studies on global patterns of gene expression [2, 29].

In parallel, we are working on the extension of our syntactic parser to the whole collection of σ70 associated proteins. To deal with the proteins that only have a few characterized sites, a version of the syntactic parser that can use sequences of specific sites instead of consensus matrices has to be developed. In the process, we should also be able to compare the use of consensus matrices vs. the use of original aligned sequences for the whole collection.

It is important to emphasize that the grammatical methodology is not limited to use consensus matrices to detect binding sites for individual proteins. Any other algorithm could equally be used as a "sensor" to detect signals located at particular positions and in specified combinations. See for instance the application of this methodology in gene recognition [9]. For instance, a method based on secondary structure to identify protein-binding sites could in principle be equally used in a syntactic recognition system.

The selectivity when using the syntactic recognizer raises on the average 5 to 10 times when compared with the standard methodology. Although these are preliminary results, they are encouraging and provide a motivation to work on collecting, organizing, and analyzing larger sets of regulatory collections of sequences. These could include other types of promoters in bacteria, and more generally, eukaryotic regulatory domains.

Acknowledgments

This work was supported by a grant from DGAPA-UNAM to J.C.-V. and D.A.R., and by a grant from Conacyt to J.C.-V. We acknowledge Gerald Hertz for the use of his programs, for fruitful discussions, and for sharing regulatory sequences that helped us to correct inconsistencies in the location of the initiation of transcription. Collaboration with G. Hertz was initiated at the 1995 Aspen Workshop on Patterns in Biological Sequences. We acknowledge Ernesto Leyva for writing the program to re-compute positions of sites.

Appendix 1a: Context-free skeleton of the σ70 grammar

This appendix shows the context-free skeleton of our grammar. For a justification of this skeleton see (Collado-Vides, 1992).

$$\begin{array}{lcl}
Pr''' & \rightarrow & D\text{-}Op \quad Pr'' \\
Pr'' & \rightarrow & Pr' \mid I' \quad Pr' \mid IC' \quad Pr' \mid HI' \quad Pr' \\
Pr' & \rightarrow & Pr \mid Pr \quad Op' \\
I' & \rightarrow & D\text{-}I \quad I(R) \\
D\text{-}I & \rightarrow & Is \\
Op' & \rightarrow & Op(R) \quad D\text{-}Op \\
D\text{-}Op & \rightarrow & Ops \\
IC' & \rightarrow & D\text{-}IC \quad IC(R) \\
D\text{-}IC & \rightarrow & ICs \\
HI' & \rightarrow & D\text{-}HI \quad HI(R) \\
D\text{-}HI & \rightarrow & Is \quad Ops \\
Is & \rightarrow & \varepsilon \mid I \quad Is \\
Ops & \rightarrow & \varepsilon \mid Op \quad Ops \\
Ics & \rightarrow & \varepsilon \mid IC \quad ICs
\end{array}$$

where the standard notation is used, that is, X → Y is read "rewrite X as Y"; the "|" is used in optional rules where the left symbol can be rewritten in different alternative ways; ε stands for the empty string. The start symbol is *Pr'''*. The nonterminals are: *Pr''', Pr'', Pr', I', D-I, Op', D-Op, I-C', D-IC, HI',* and *D-HI*. The terminals are: *Pr, I(R), Op(R), IC(R), HI(R), I, Op*, and *IC*. The dictionary entries are: *Pr, I(R), Op(R), IC(R),* and *HI(R)*. Except for *Pr*, all of these entries have contextual information.

Regulatory sites in a given promoter are grouped into "phrases". There are three basic types of phrases, positive phrases (I', or IC'), negative phrases Op' and heterologous phrases HI'. The simplest phrases contain one or several sites for the same regulator, whereas IC' contain sites for different activators, and HI' contain both activator and repressor sites. The fact that any σ70 promoter must have a proximal site is reflected in the condition that any phrase involves a referential "(R)" proximal site. Any phrase X contains also a category for duplicated or multiple optional sites which are generated from the D-X symbols (i.e., D-I for duplicated activator sites). Since a derivation can involve combinations of phrases and different functional positions for each protein, the grammar generates a large number of regulatable arrays.

Appendix 1b: An example of a parse tree

The parse tree corresponding to the phrase [[Op,d,i,-93],[I,d,j,-44],[I,c,j,CRP,-62.5],[Pr,lac],[Op,c,i,LacI,+9],[Op,d,i,+402]] is presented in the Figure 1.

Appendix 2: Prolog implementation of the σ 70 grammatical model

Difference lists. As a first approximation for representing the productions of the σ70 grammar, we can employ “difference lists,” as is commonly done in Prolog when using “definite-clause grammars” (DCG) [18]. The first production, for example:

$$Pr''' \rightarrow D\text{-}Op \;\; Pr''$$

is written in DCG syntax as:

```
‘Pr3’ --> ‘D-Op’, ‘Pr2’.
```

where we have replaced the original primes by digits, and we have enclosed the predicate symbols in quotes, to conform to standard Prolog syntax for predicate symbols. This DCG clause abbreviates the following Prolog clause:

```
‘Pr3’(X,Z) :- ‘D-Op’(X,Y), ‘Pr2’(Y,Z).
```

The intended meaning of each predicate p(X,Y) is that there is a string starting at X and ending at Y which belongs to the formal language p. String concatenation is achieved by using the same variable Y as the ending of the string in ‘D-Op’ and the beginning of the string in ‘Pr2’.

Contextual information. A problem with the above representation is that it lacks contextual information. (Such information is depicted in Figure 1 as arrows connecting the leaves of the parse tree.) The DCG formalism, allows us to incorporate contextual information with additional argument places. These argument places can be viewed as transferring information between nodes which are siblings in the parse/proof tree. Note, however, that we wish to transfer information between *leaves* of the parse tree. Hence, given two leaves that must communicate with each other, we added argument places to all their ancestors which are not common to both leaves.

For example, the information transfer represented with the arrow labeled A, is implemented with an argument place added to the subgoals in the first production:

```
‘Pr3’ --> ‘D-Op’(A), ‘Pr2’(A).
```

Right-to-left construction. Another elaboration results from the order in which we want to construct certain portions of the parse tree. Prolog builds parse/proof trees using a left-to-right discipline. If we followed such a regime, we would have to build some parts of the tree which depend on information contained in the dictionary *before* accessing the dictionary. For instance, the number (and contents) of the ‘Op’ leaves to the left of the ’I’ leaves depends on the information represented by the arrow labeled A. Such information is contained in the dictionary under an ‘Op(R)’ entry. Therefore, to build trees in an efficient way, we must override Prolog's default left-to-right order. It is possible to do so by making explicit the hidden arguments of the DCG shorthand, and writing the subgoals from right to left, so that by selecting the leftmost subgoal, Prolog first constructs the rightmost sibling.

```
‘Pr3’(X,Z) :- ‘Pr2’(A,Y,Z), ‘D-Op’(A,X,Y).
```

Dictionary. Consider now the representation of the dictionary entries. The tree in Figure 1 can be generated with the following dictionary:

```
‘Op(R)’([p(i,-93)],[p(i,+402)],[[i,’LacI’,+9]|X],X).
‘I(R)’([p(i,-44)],[[i,’CRP’,-62.5]|X],X).
```

where the terms with the p function symbol are used to group an *index*, say i, with a site position, say -93.

The first argument place of the ‘Op(R)’ predicate is a list of the optional sites to the *left* of the obligatory site (arrow A in Figure 1) of that dictionary entry. The second argument place of this predicate is a list with the optional sites to the right of its associated obligatory site (arrow B in appendix 1b). Finally, the third argument place of such a predicate contains the information about the obligatory site.

Similarly, the first argument place of the ‘I(R)’ predicate is a list of the optional sites, all of which are to the left of the associated obligatory site. The second argument place of this predicate represents the obligatory ‘I(R)’ site.

For example, the term p(i,-93) generates the leftmost ‘Op’ leaf, which is an optional site associated with the obligatory site ‘LacI’.

Indexes. Note that to be able to determine which obligatory site is associated with a given optional site, we must rename the indexes of the different dictionary entries. In this case, we map the i index for the ‘I(R)’ entry to a new j index, as appears in appendix 1b. Such a renaming is performed with additional argument places.

Subsets of optional sites. The last step in our implementation includes the computation of *subsets* of optional sites. Each list of optional sites in a dictionary entry represents a set of which *any* subset is taken as a valid set of optional sites. This change is achieved by incorporating a predicate subset(Sub,Xs) which is intended to hold when Sub is a subset of Xs.

References

1. O.G. Berg and P.H von Hippel, “Selection of DNA binding site by regulatory proteins. Statistical-mechanical theory and application to operators and promoters” *J. Mol. Biol*. **193**, 723 (1987)
2. S.E. Chuang, D.L. Daniels and F.R. Blattner, “Global regulation of gene expression in *Escherichia coli*” *J. Bacteriol*. **175**, 2026 (1993)
3. G. Casari, M. Andrade, P. Bork, J. Boyle, A. Daruvar, C. Ouzounis, R. Schneider, J. Tamames, A. Valencia, and C. Sander, “Challenging times for bioinformatics” *Nature* **376**, 647 (1995)
4. J. Collado-Vides, “The search of a grammatical model of gene regulation is formally justified by showing the inadequacy of context-free grammars” *CABIOS* **7**, 321 (1991)
5. J. Collado-Vides, “A grammatical model of the regulation of gene expression” *Proc. Natl. Acad. Sci.USA* **89**, 9405 (1992)
6. J. Collado-Vides, “A Linguistic Representation of the Range of Transcription Initiation of σ70 Promoters: I. An Ordered Array of Complex Symbols with Distinctive Features” *Biosystems* **29**, 87 (1993)
7. J. Collado-Vides, “A Linguistic Representation of the Range of Transcription Initiation of σ70 Promoters: II. Distinctive Features of Promoters and Their Regulatory Binding Sites” *Biosystems* **29**, 105 (1993)
8. J. Collado-Vides in *Integrative Approaches to Molecular Biology*, “Integrative representations of the regulation of gene expression” Eds. J. Collado-Vides, B. Magasanik and T.F. Smith (MIT Press, Cambridge Mass, 1996)
9. S. Dong and D.B. Searls, “Gene structure prediction by linguistic methods” *Genomics* **23**, 540 (1994)
10. K. Frech, G. Herrmann and T. Werner, “Computer-assisted prediction, classification, and delimitation of protein binding sites in nucleic acids” *Nucleic Acids Res*. **21**, 1655 (1993)
11. J.A. Goodrich, M.L. Schwartz and W.R. McClure, “Searching for and predicting the activity of sites for DNA binding proteins: compilation and analysis of the binding sites for *Escherichia coli* integration host factor (IHF)” *Nucleic Acids Res.* **18**, 4993 (1990)
12. J.D. Gralla and J. Collado-Vides in *Cellular and Molecular Biology: Escherichia coli and Salmonella*, “Organization and Function of Transcription Regulatory Elements” Eds. F.C. Neidhardt, R. Curtiss III, J. Ingraham, E.C.C. Lin, K.B. Low, B. Magasanik, W. Reznikoff, M. Schaechter, H.E Umbarger and M. Riley (2nd ed., Washington, D.C. American Society for Microbiology, 1996)
13. D.K. Hawley and W.R. McClure, “Compilation and analysis of *Escherichia coli* promoter DNA sequences” *Nucleic Acids Res.* **11**, 2237 (1983)
14. G.H. Hertz, G.W. Hartzell-III and G.D. Stormo, “Identification of consensus patterns in unaligned DNA sequences known to be functionally related” *CABIOS* **6**, 81 (1990)

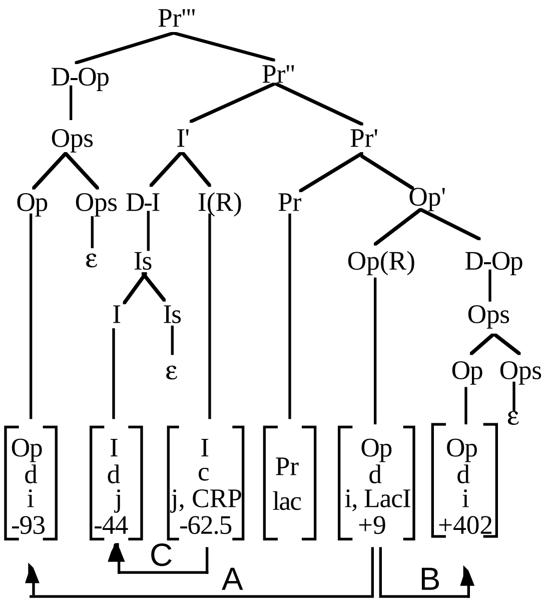
15. G.H. Hertz and G.D. Stormo in *Bioinformatics and Genome Research*, “Identification of consensus patterns in unaligned DNA and protein sequences: a large-deviation statistical byasis for penalizing gaps” Ed. H.A. Lim and C.R. Cantor (World Scientific Publ., Singapore, 1995).
16. Lisser S. and Margalit H. (1993). "Compilation of *E.coli* mRNA promoter sequences". *Nucleic Acids Res*. **21**: 1507-1516.
17. M.C. O´Neill, “Consensus methods for finding and ranking DNA binding sites. Application to *Escherichia coli* promoters” *J. Mol. Biol.* **207**, 301 (1989)
18. F.C.N. Pereira and , D.H.D. Warren, “Definite clause grammars for language analysis - A survey of the formalism and a comparison with augmented transition networks.” *Artificial Intelligence* **13**, 231 (1980)
19. D. Rosenblueth, D. Thieffry, A. M. Huerta, H. Salgado and J. Collado-Vides, “Syntactic recognition of regulatory regions in Escherichia coli” *CABIOS* **12** (in press).
20. M. Scharf, R. Schneider, G. Casari, P. Bork, A. Valencia, C. Ouzounis and C. Sander in *Intelligent Systems for Molecular Biology 1994 (ISMB94)*, “GeneQuiz: a workbench for sequence analysis” (AAAI Press, Stanford, CA, 1994).
21. T.D. Schneider, G.D. Stormo, L. Gold and A. Ehrenfeucht, “Information content of binding sites on nucleotide sequences” *J. Mol. Biol*. **188**, 415 (1986)
22. R. Staden, “Computer methods to locate signals in nucleic acid sequences” *Nucleic Acids Res.* **12**, 505 (1984)
23. R. Staden, “Methods for calculating the probabilities of finding patterns in sequences” *CABIOS* **5**, 89 (1989)
24. L. Sterling, and E. Shapiro, *The Art of Prolog* (2nd edition. MIT Press, Cambridge, MA, 1994)
25. G. Stormo, “Consensus patterns in DNA” *Meth. Enzym*. **183**, 211 (1990)
26. R.A VanBogelen, K.Z. Abshire, A. Pertsemlidis, R.L. Clark and F.C. Neidhardt in *Cellular and Molecular Biology: Escherichia coli and Salmonella*, “Gene-protein database of *Escherichia coli* K-12, Edition 6” Eds. F.C. Neidhardt, R. Curtiss III, J. Ingraham, E.C.C. Lin, K.B. Low, B. Magasanik, W. Reznikoff, M. Schaechter, H.E Umbarger and M. Riley (2nd ed., Washington, D.C. American Society for Microbiology, 1996)
27. P.H. von Hippel and O.G. Berg, “On the specificity of DNA-protein interactions” *Proc. Natl. Acad. Sci.USA* **83**, 1608 (1985)
28. L. Wall and R.L. Schwartz, *Programming Perl* (O´Reilly and Associates, Inc., Sebastopol, Ca, 1991).
29**.** M.X. Wang and G.M. Church, “A whole-genome approach to *in vivo* DNA-protein interactions in *E.coli*” *Nature* **360**, 606 (1992)
30. F. Wolfertstetter, K. Frech, G. Herrmann, and T. Werner, “Identification of functional elements in unaligned nucleic acid sequences by a novel tuple search algorithm” *CABIOS* **12**, 71 (1995).

Table 1. Comparison of weight matrices vs. grammatical recognition.

Protein (W:alignment width, IC: information content (bits), T: threshold)	**Scores intervals: Functional sites/Grammar+Patser/Patser**							
	5-10	**10-15**	**15-20**	**20-25**	**25-30**	**30-35**	**Totals**	**Ratios**
AraC (W:12, IC:8.38, T:10.55)	-	6/8/161*	3/3/4	0/0/0	0/0/0	0/0/0	9/12/165	1/1.33/18.33
ArgR (W:24, IC:15.71, T:18.82)	-	1/49/646*	0/9/116	5/6/12	6/6/6	0/0/0	12/70/780	1/5.83/65
CRP (W:24, IC:12.21, T:6.94)	4/34/209*	12/31/102	17/18/26	4/5/5	0/0/0	0/0/0	37/88/342	1/2.38/9.24
FNR (W:19, IC:13.48, T:19.59)	-	-	1/1/2*	4/5/10	3/3/3	0/0/0	8/9/15	1/1.13/1.88
GlpR (W:23, IC12.27, T:16.49)	-	-	2/4/48*	8/8/11	0/0/0	1/1/1	11/13/60	1/1.18/5.45
LexA (W:21, IC:18.27, T:17.01)	-	-	1/1/5*	1/1/2	8/8/8	1/1/1	11/11/16	1/1/1.46
MalT (W:11, IC:6.10, T:10.66)	-	8/9/43*	0/0/0	0/0/0	0/0/0	0/0/0	8/9/43	1/1.13/5.38
MetJ (W:10, IC:8.27, T:5.80)	3/49/596*	10/12/47	0/0/0	0/0/0	0/0/0	0/0/0	13/61/643	1/4.69/49.46
PhoB (W:18, IC:13.30, T:8.85)	1/17/260*	1/30/452	2/2/20	7/8/10	0/0/0	0/0/0	11/49/742	1/4.45/67.45
PuR (W:17, IC:19.09, T:19.01)	-	-	1/1/1*	3/3/6	5/5/6	0/0/0	9/9/13	1/1/1.44
PutA (W:12, IC:8.03, T:11.53)	-	6/11/86*	2/3/5	0/0/0	0/0/0	0/0/0	8/14/91	1/1.75/11.38
TyrR (W:19, IC:13.38, T:9.35)	1/9/53*	0/23/161	7/12/21	7/8/8	0/0/0	0/0/0	15/52/243	1/3.47/16.2

Three classes of sequences within the different threshold intervals are indicated: the functional sites used to construct the weight matrices, those found with the syntactic recognizer coupled to weight matrices as sensors, and those found with *Patser* alone (for information about the scoring scheme, see Stormo, 1990). For instance, for AraC in the interval of 10 to 15 bits there are 6 functional sites, 8 sites found by the grammar with patser, and 161 sites found with patser alone. *In this interval we only counted scores greater or equal to the corresponding threshold.

Figure 1. Example of a parse tree.



Pr'''
D-Op
Pr''
Ops
I'
Pr'
Op
Ops
D-I
I(R)
Pr
Op'
ε
Is
Op(R)
D-Op
I
Is
Ops
ε
Op
Ops
ε
Op d i -93
I d j -44
I c j, CRP -62.5
Pr lac
Op d i, LacI +9
Op d i +402
C
A
B