Abstract—Using micro arrays, researchers can measure expression levels of thousands of genes efficiently. This generates a huge amount of gene expression data to be analyzed. The analysis of this gene expression data provides important information that can be used to predict gene regulatory networks (GRN). However, since the number of genes is large and the number of regulators for each gene is varying, predicting a GRN is an NP-HARD problem. In addition the analyses of a single set of expression data can only provide potential candidates for regulation that need to be tested by some other means. We describe an elimination method that can be used to vastly reduce the candidates for GRN to manageable level on which more expensive tests can be performed. The method evaluates potential gene regulation relations by applying bit wise operation to classify changes of genes expression levels into acceptable or unacceptable patterns based on the expected behavior of such networks.

Index Terms— Gene expression, time series, data mining, gene regulatory networks.

I. INTRODUCTION

The recent influx of vast amounts of time series data from micro arrays has made it imperative to develop fast methods for finding regulation relations in microarray Open Reading Frames, ORFs, and mapping the overall gene regulatory network. These expression time series have biological and instrumentation noise that must be dealt with. This has led to the development of data mining approaches designed to work in the presence of noise, some of these are [1][2][3][4][5][6][7][8][9], but none of these claim to reach near optimal results. A regulation relation is relationship between genes that is indicated by a change of expression levels in their ORFs. Genes are called activators if they activate the production of proteins by other genes, called targets. Genes are repressors if they suppress the production of target genes. In time-series, the target gene expression level increases in response to the increase of the expression level for an activator, and decreases in response to the decreases of repressors expression level. Regulation relations can be defined for a set of genes \( G \), such that for any target gene \( t \in G \), a subset of genes called activators \( A \subset G \) affects the production of its expression level positively, and another subset of genes called repressors \( R \subset G \) affect the production of its expression level negatively.

In this paper, we introduced a new approach for predicting regulation relations between genes ORFs. This is done by eliminating all candidate regulation relations in which their gene ORFs values changes in a way that contradicts the following behavior:

A. ART (activator-repressor-target) Pattern

- If the activator expression level increases and the repressor expression level decreases or remains unchanged, the target expression level must increase.
- If the activator expression level decreases or remains unchanged and the repressor expression level increases; the target expression level must decrease.
- If the activator expression level decreases and the repressor expression level increase or remains unchanged, the target expression level must decrease.
- If the activator expression level increase or remains unchanged and the repressor expression level decreases, the target expression level must increase.
- If both the activator and the repressor expression levels did not change, the target expression level to remains unchanged.

When the activator-repressor-target expression levels changes, for a given triple \((\text{ORF}(1), \text{ORF}(2), \text{ORF}(3))\) of ORF’s, one can test for an activator-repressor-target pattern contradiction by generating from each ORF(i), i=1,2,3, a sequence, \( \text{delta}(i) \), of 0’s, +’s, and -’s as follows:

If ORF(i) increases between two sample points \( \text{ORF}(i,j) \) and \( \text{ORF}(i,j+1) \), set \( \text{delta}(i,j)=+ \), if it decreases set \( \text{delta}(i,j)=- \), and if remains unchanged set \( \text{delta}(i,j)=0 \).

Then, for instance, if \( \text{delta}(1,j)=- \), \( \text{delta}(2,j)=+ \) and \( \text{delta}(3,j)=+ \) we would have that \( \text{ORF}(1), \text{ORF}(2), \text{ORF}(3) \) cannot be an activator-repressor-target triple (but \( \text{ORF}(3), \text{ORF}(1), \text{ORF}(2) \)) might be an ART triple) as its behavior contradicts 2 above. If any of the six arrangements of \( \text{ORF}(1), \text{ORF}(2), \text{ORF}(3) \) contradicts the ART pattern at a sample point \( j \), it is eliminate from consideration. Any arrangement that escapes elimination therefore does not contradict the ART pattern and

Nawar Malhis is with the Department of computer science at Kent State university, Kent, OH 44240 USA (Phone: 001 (905) 906-0963, Email: nmalhis@cs.kent.edu).

Arden Ruttan is with the Department of computer science at Kent State university, Kent, OH 44240 USA (Phone: 001 (330) 672-9066, Email: ruttan@cs.kent.edu).
remains a viable candidate for as an actual regulation relation. At the end of the computation any remaining candidates indicate potential regulation relations and can be tested in the lab or with more computationally expensive software. In a situation where there are n ORF’s and m time steps in each ORF, this algorithm would need to be applied to n(n-1)(n-2)/6 triples and so would require O(m * n^3) operations to test all triples.

As described above the ART pattern does not consider either noise (biological or measurement) nor does allow for time delays in target expression changes following an expression level change in an activator or a repressor. Without some mechanism to account for noise, the introduction, for instance, of a random increase of in the expression level of a target activator would erroneously eliminate a candidate triple unless there was a accompanying random increase in the target. Similarly if the expression level sampling time scale is short enough, the effects on the target of expression level changes in an activator or in a repressor may not be seen in the same time series sample. In the section 4, we introduce an algorithm that extends the ART pattern elimination method to accommodate both noise and time delays.

Although we have discussed a case where a regulation relations of size three, this concept can be applied to regulation relations of any size by using some function which characterizes the activators/repressors pattern for the given number of assumed regulators. Many challenges arise in the problem of predicting large (i.e. greater than three regulators) gene regulatory networks. These include but are not limited to the following:

- One might not know the exact number of genes participating in the relation. If a regulator ORF is omitted from the regulation relation, the remaining regulators would not show the expected regulation pattern. Searching for regulation relations of all sizes is an NP-Hard problem [9].
- The cost of searching for k gene regulation relation in a set of n genes is in most cases in the order of O(m * n^k) [1][7][8], where m is the size of the time series (the number of sample points) , assuming that k is constant and small compared to n.

II. BACKGROUND

There are many different previous approaches for extracting gene regulatory network from time series of gene expressions; we discuss here some that are related to our work.

Several approaches extract potential regulation relations by analyzing pairs of genes expression series, using:
- Correlations method [5]: This tests for global similarity between pairs of gene expression series.
- Edge detection method [2]: This detects major change in expression levels “an edge” and compares edges in two gene ORFs that exist within a reasonable distance from each other.
- Event method [3]: This tracks directional changes in expression levels “an event” between two genes expression series, with a certain amount of time delay, by calculating the slope of the expression profile at each time interval.

However, when these three methods are applied on the same Cho-Spellman [10] input data sets, their true-positive output was small and there was almost no overlap among ART triples produced by the methods [3]. Many other attempts have been made to extract regulation relations by analyzing three or more gene expression series at the same time, this might give big advantage over pairs of ORFs , but on the other hand, it requires much higher computational power since the number of possible candidates for regulation is O(nk) where k is the number of genes in each potential regulation relation. Some of these approaches are:

- Fuzzy logic approach [8]: this provides a way to transform precise numbers into qualitative descriptions. So, the expression levels of an activator and a repressor are transformed from crisp values to fuzzy values in a process called “fuzzification”. Then a set of heuristic rules can be used to give scores that will generate the predicted fuzzified value of the target, this fuzzified value is transformed back into a precise number [8]. The calculated target values are then compared with each possible target gene; those that match are then considered as potential regulation relations.
- Smooth Response Surface (SRS) [1]: Also looks for triplets of genes activator-repressor-target (ART) [1]. In this model, one uses a three-dimensional response surface as a function of a pair of genes, ĉ = S(A, B) [1]. Here S(A, B)is the biological model, that is, a mathematical formula that maps the biological transcription input values into predicted target transcription level ć. This calculated target values ĉ is then compared with each possible target gene, those that match are then considered as potential regulation relations.
- A third class, Fuzzy Pruning Method [7]: In this approach, one stores the changes in expression level between each two adjacent time points using some fuzzy mapping into several bit vectors. By using bit wise operations on these vectors, one is able to prune out about 70% of the computations needed in SRS [1] approach on the Cho-Spellman datasets. This pruning method can be applied to most GRN prediction approaches.

In this work, we will extend the Fuzzy Pruning method by considering all possible neighbors for each point. This will increases the number of checkpoints (a checkpoint is a change in expression level between two points) from (n-1) to (n * (n-1) / 2). Increasing the number of check points enabled us to achieve higher pruning rate; when it was used for mining regulation relations of size 3, on the Cho-Spellman datasets, about 99% of all possible triplets were eliminated, and the remaining 1% did contain about 95% of the output from applying the SRS [1] approach on the same datasets.

III. THE ELIMINATION METHOD

The correlation, edge detection, and the event methods each search for regulation relations in pairs of expression profiles (i.e. the number of ORFs in each regulation relation k is 2).
Each detected some regulators when it was applied on the Cho-Spellman [10] yeast CDC28 and Alpha datasets [3]. However, their true positive results were about 20% of the total regulators [3]. That might be caused by the effect of other genes participating in the regulation relation. On the other hand, both the fuzzy logic [8] and the smooth response surface [1] do handle regulation relations with more than two genes (k \(>2\)). Their algorithms predict the target expression level based on the biological model in [9] and the heuristic rules in [8], then search all other genes expression profile for a match within some allowed threshold error. If a gene expression passes this threshold, then a score is generated and the relation is considered a regulation candidate. However, in both cases the target gene expression should fit the predicted profile, while in real biological environment; that profile scenario might not match wide range of regulation relations. In the Elimination Method, EM, we address this problem by avoiding the limitation of having a predicted target profile.

### A. Elimination Method Algorithm

Before starting the mining process, we take the following steps:

1. **First filtering:**
   - First, we filter out those ORFs that have unacceptable amount of missing values. Following procedures of other researcher [1][7][8], all ORFs that have two or more adjacent data points missing are filtered out. If just one point is missing between two other existing points that point was assigned the average of value before and after it.
   - **Normalization:**
     - Each ORF in the filtered expression data set is been normalized to the range [0...1].
     - **Generating checkpoints:**
       - For any ORF(a) in the normalized data, the change of transcription levels between any two time points, delta(a, j, s) is calculated, where j is the starting time point and s is the step from it to the other point:
       
       \[
       \text{delta}(a, j, s) = \text{ORF}(a, j + s) - \text{ORF}(a, j),
       \]

       \(1 <= o <= n, \ 0 <= s <= m - 1, \ 1 <= j <= m - s.\)
       
       Then, a heuristic fuzzy method is applied to overcome noise level that exists in the input data and encode these changes into 8 bits groups for each ORF, a, as defined in (1) below, which generates 8 bit-vectors for each ORF. The size of each vector is equal to the number of checkpoints. The time complexity for this coding process is O(m^2 n), since we need O(m^2) to consider all possible changes as s varies and O(n) to code for each ORF.

       In order to analyze the ART pattern of the data in the presence of noise, we add tuning factors \(\alpha > \beta\). Where \(\alpha\) and \(\beta\) are used to allow small changes in expression level that may be due to noise. The values assigned to \(\alpha\) and \(\beta\) will depend on the amount of noise present in the data. In particular, \(\alpha\) is used to distinguish significant change from marginal change and \(\beta\) to distinguish no change from marginal change.

       For each \(\text{delta}(a, j, s)\) we store the following bit values each in its corresponding vector:

       \[
       \begin{align*}
       \text{up0}(o, j, s) &= \text{true if delta}(a, j, s) \geq \alpha; \text{false otherwise.} \\
       \text{up1}(o, j, s) &= \text{true if delta}(a, j, s) \geq \beta; \text{false otherwise.} \\
       \text{up2}(o, j, s) &= \text{true if delta}(a, j, s) > -\alpha; \text{false otherwise.} \\
       \text{flat0}(o, j, s) &= \text{true if abs(delta}(a, j, s)) < \beta; \text{false otherwise.} \\
       \text{flat1}(o, j, s) &= \text{true if abs(delta}(a, j, s)) < \alpha; \text{false otherwise.} \\
       \text{down0}(o, j, s) &= \text{true if delta}(a, j, s) \leq -\alpha; \text{false otherwise.} \\
       \text{down1}(o, j, s) &= \text{true if delta}(a, j, s) \leq -\beta; \text{false otherwise.} \\
       \text{down2}(o, j, s) &= \text{true if delta}(a, j, s) < \alpha; \text{false otherwise.}
       \end{align*}
       \]

       These bit values are stored in unsigned integer variables, So, for example, we can store 32 \(\text{up0}(o, j, s)\) values in each unsigned integer variable of the \(\text{UP0}(o)\) vector.

2. **Second filtering:**
   - In addition to first filtering, those ORFs that do not show active participation in the regulation process are filtered out. That is done by looping over each ORF, and for the maximum value of \(\alpha\), the number of bits that is set to one in \((\text{UP0} \cup \text{DOWN0})\) is counted and assigned to ab(o). An average \(\text{avg}\) and variant \(\text{va}\) for all ORFs are calculated:

   \[
   \begin{align*}
   \text{UP0} &= \{1 \mid \text{up0}(o, j, s) = \text{true}\} \\
   \text{DOWN0} &= \{1 \mid \text{down0}(o, j, s) = \text{true}\} \\
   \text{UP1} &= \{1 \mid \text{up1}(o, j, s) = \text{true}\} \\
   \text{DOWN1} &= \{1 \mid \text{down1}(o, j, s) = \text{true}\} \\
   \text{UP2} &= \{1 \mid \text{up2}(o, j, s) = \text{true}\} \\
   \text{DOWN2} &= \{1 \mid \text{down2}(o, j, s) = \text{true}\}
   \end{align*}
   \]

   All ORFs where \(\text{ab}(o)\) is less than the \((\text{avg} – \text{va} / 1.5)\) are then filtered out. This will filter out all ORFs that have only a small number of significant expression level changes.

   - **Mining for regulation relations:**
     - Our basic assumption that did not consider noise in the input data, so, we will extend it in the following way:

     1. If the activator is in up0, and the repressor is in down0, the target must be at best in up1. Here, this relaxation, represented in \(\alpha\), the number of bits that is set to one in \((\text{UP0} \cup \text{DOWN0})\) is counted and assigned to ab(o). An average \(\text{avg}\) and variant \(\text{va}\) for all ORFs are calculated:

     2. If the activator and the repressor are both in flat0, the target must be in a slightly relaxed case, flat1.

     3. If the activator is in up0, and the repressor is in flat0, the target must be in case, up2.

     4. If the activator is in up1, and the repressor is in down1, the target must be in case, up2.

   Now, we loop over all ORFs, considering each ORF once as an activator ORF(a) and once as a repressor ORF(r). Target mask vectors are generated using our basic assumption (mU1, mU2, mF1, mD1, mD2) by applying bit operations on ORF(a) and ORF(r):

   \[
   \begin{align*}
   \text{mU1} &= \text{UP0}(a) \cup \text{DOWN0}(r); \\
   \text{mU2} &= (\text{FLAT0}(r) \cup \text{UP0}(a)) \cup (\text{FLAT0}(a) \cup \text{DOWN0}(r)) \cup (\text{UP1}(a) \cup \text{DOWN1}(r)); \\
   \text{mF1} &= \text{FLAT0}(a) \cup \text{FLAT0}(r); \\
   \text{mD1} &= \text{DOWN0}(a) \cup \text{UP0}(r); \\
   \text{mD2} &= (\text{FLAT0}(r) \cup \text{DOWN0}(a)) \cup (\text{FLAT0}(a) \cup \text{UP0}(r)) \cup (\text{UP1}(r) \cup \text{DOWN1}(a));
   \end{align*}
   \]
These operations need to be done for each checkpoint as an activator and as a repressor. That is not an expensive computation, since they are bit operations (in most machines, each bit operation takes only one clock cycle), plus, each 32 values can be done at one time. In the case of yeast cdc15 dataset, it has 24-time points, so:

The number of checkpoints d is:

\[ d = 24 \times 23 / 2 \]

\[ d = 276 \]

Each vector size is equal to \( 276 / 32 \) = 9 unsigned integers, so the total mask size is:

\[ 9 \times 5 = 45 \text{ unsigned integers} \]

Next, we loop over all ORFs, and test each of them for a possible target that match the mask. In this test each mask vectors must be contained fully in the target ORF(t) vectors, that is, for every bit set to 1 in the mask vectors, the corresponding bit in the target ORF(t) must also be set to 1.

That can be tested by applying the following operations on each “unsigned int” of the mask and the target bit vectors:

\[
(3) \quad (mU1 \text{ and (not UP1(t))}) \text{ or (mU2 and (not UP2(t))}) \text{ or (mF1 and (not FLAT1(t))) or (mD1 and (not DOWN1(t))}) \text{ or (mD2 and (not DOWN2(t))}) \equiv 0
\]

If an ORF(t) passes this test, we consider this ORF(t) together with ORF(a) & ORF(r) a candidate regulation relation. See figure 2.

From (1), for two different values of \( \alpha \), such that \( \alpha_1 > \alpha_2 \), with the same value of \( \beta \), it is clear that:

a) If a bit was set to one in UP0 or DOWN0 for \( \alpha_1 \), it will be set to one for \( \alpha_2 \).

b) If a bit was set to one in FLAT1, UP2, or DOWN2 for \( \alpha_2 \), it will be set to one for \( \alpha_1 \).

c) While UP1 and DOWN1 will not be effected.

```
for (int a = 0; a < numOFORFs; a++) {
    for (int r = 0; r < numOFORFs; r++) {
        mask = setMask(a, r)
        for (int t = 0; t < numOFORFs; t++) {
            testTarget(Mask, t)
        }
    }
}
```

From (2), we can see that if any bit in the mask2 for \( \alpha_2 \) was set to one, it will be set to one for mask1 \( \alpha_1 \).

From (3), we can see that if a target ORF passes mask2, it will pass mask1. So, the test can be done on many values of \( \alpha \) in sequence, that is, the target ORF(t) will be tested on a smaller \( \alpha \) value only if it passes the test on a larger \( \alpha \) value. This will speed to a large degree the mining process, since only a very small percentage will pass the test on the largest \( \alpha \) value.

IV. RESULTS

We ran this algorithm on the Plasmodium dataset [12] and on the Cho-Spellman yeast datasets CDC15, CDC28 and Alpha, since all these datasets are widely studied and used. The Plasmodium dataset features a single dataset for the malaria transcriptome numbering 7090 ORFs produced by the DeRisi lab [12]. The Cho-Spellman datasets contain expression profile for all of the yeast genome, numbering 6178 ORFs. After first filtering, normalizing, generating check points, and second filtering, we ended up with the following data sets (Table I):

Then, each of the four data sets are passed thru the EM algorithm (Figure 2) for thirty four different values of the tuning factor \( \alpha \), from \( \alpha = 0.02 \) to \( \alpha = 0.35 \), while \( \beta \) was found is best always to be set to 0.01 for these datasets (see discussion in section 4.1).

Those triplets that were not eliminated for some \( \alpha \) and \( \beta \) values, and passed as an output, did not violate the basic definition of regulation at any of the check points.

```
<table>
<thead>
<tr>
<th>Data Set</th>
<th># of ORFs after filtering</th>
<th># of time points</th>
<th># of Checkpoints</th>
</tr>
</thead>
<tbody>
<tr>
<td>CDC15</td>
<td>3885</td>
<td>24</td>
<td>276</td>
</tr>
<tr>
<td>CDC28</td>
<td>4000</td>
<td>17</td>
<td>136</td>
</tr>
<tr>
<td>Alpha</td>
<td>4203</td>
<td>18</td>
<td>153</td>
</tr>
<tr>
<td>Plasmo</td>
<td>3811</td>
<td>46</td>
<td>1035</td>
</tr>
</tbody>
</table>
```

The number of ORFs, number of time points and checkpoints for each of the Cho-Spellman’s and Plasmodium datasets.

The output size of the EM for various values of \( \alpha \) is presented in Table II and in Figure 3 as a percentage of all possible triplets \( O(n^3) \) for each input dataset.
Since SRS [1] is based on biological model we expect the EM will register a relationship whenever SRS does. So the same datasets were processed by the SRS approach and then both outputs of the SRS and EM are compared. The percentage of the SRS output size that is contained in the EM output (we call it ES) is then used as a measuring tool for evaluating the EM output. If ES was near 100% we consider that the number of false negative to be small. However, the output size of EM is much larger than the output size of the SRS. That is reasonable, since in the EM the output is not limited to those regulation relations that fit a predefined biological model, but it includes every candidate regulation relation that comply with the basic definition of regulation relations. The results show that for the three Cho-Spellman datasets and at some starting value $\alpha$ onward, almost all the SRS [1] output is contained in the EM output (Figure 4).

### Table II

<table>
<thead>
<tr>
<th>$\alpha$</th>
<th>CDC-28</th>
<th>CDC-15</th>
<th>Alpha</th>
<th>Plasmo</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.02</td>
<td>0.0006%</td>
<td>0.0000%</td>
<td>0.002%</td>
<td>0.0000%</td>
</tr>
<tr>
<td>0.04</td>
<td>0.0014%</td>
<td>0.0000%</td>
<td>0.003%</td>
<td>0.0006%</td>
</tr>
<tr>
<td>0.06</td>
<td>0.0027%</td>
<td>0.0001%</td>
<td>0.006%</td>
<td>0.0044%</td>
</tr>
<tr>
<td>0.08</td>
<td>0.0050%</td>
<td>0.0002%</td>
<td>0.013%</td>
<td>0.0177%</td>
</tr>
<tr>
<td>0.10</td>
<td>0.0093%</td>
<td>0.0007%</td>
<td>0.026%</td>
<td>0.0517%</td>
</tr>
<tr>
<td>0.12</td>
<td>0.0165%</td>
<td>0.0016%</td>
<td>0.051%</td>
<td>0.1213%</td>
</tr>
<tr>
<td>0.14</td>
<td>0.0281%</td>
<td>0.0036%</td>
<td>0.093%</td>
<td>0.2414%</td>
</tr>
<tr>
<td>0.16</td>
<td>0.0467%</td>
<td>0.0074%</td>
<td>0.162%</td>
<td>0.4220%</td>
</tr>
<tr>
<td>0.18</td>
<td>0.0746%</td>
<td>0.0143%</td>
<td>0.221%</td>
<td>0.6756%</td>
</tr>
<tr>
<td>0.20</td>
<td>0.1152%</td>
<td>0.0262%</td>
<td>0.439%</td>
<td>1.0066%</td>
</tr>
<tr>
<td>0.22</td>
<td>0.1710%</td>
<td>0.0452%</td>
<td>0.688%</td>
<td>1.4327%</td>
</tr>
<tr>
<td>0.24</td>
<td>0.2481%</td>
<td>0.0745%</td>
<td>1.056%</td>
<td>1.9558%</td>
</tr>
<tr>
<td>0.26</td>
<td>0.3500%</td>
<td>0.1183%</td>
<td>1.570%</td>
<td>2.5787%</td>
</tr>
<tr>
<td>0.28</td>
<td>0.4833%</td>
<td>0.1813%</td>
<td>2.27%</td>
<td>3.2863%</td>
</tr>
<tr>
<td>0.30</td>
<td>0.5573%</td>
<td>0.2692%</td>
<td>0.323%</td>
<td>4.0914%</td>
</tr>
<tr>
<td>0.32</td>
<td>0.6795%</td>
<td>0.3862%</td>
<td>0.459%</td>
<td>4.9780%</td>
</tr>
<tr>
<td>0.34</td>
<td>1.1477%</td>
<td>0.6496%</td>
<td>0.512%</td>
<td>5.9688%</td>
</tr>
<tr>
<td>0.35</td>
<td>1.3034%</td>
<td>0.6443%</td>
<td>0.708%</td>
<td>6.4976%</td>
</tr>
</tbody>
</table>

The Elimination method output size as percentages of total number of triples $n^3$, for the Plasmodium and the three Cho-Spellman datasets.

Fig. 3. A graph represents the output size as a percentage of total number of triples $n^3$, (from Table 1) for all three Cho-Spellman datasets as a function of $\alpha$ tuning factor.

Fig. 4. The percentage of SRS output that is contained in the Elimination method EM as a function of the $\alpha$ tuning factor.

**A. Serial Dependency**

It is clear that the results of the Plasmo dataset is larger in size than the Cho-Spellman three datasets, we attribute that to the nature of these datasets. The EM uses the alternation between increasing and decreasing expression to characterize the ART effect. If a dataset such as the Plasmo has fewer alternation points (i.e. it has infrequent changes in the direction of slope throughout the time series), the percentage of candidates regulation relations eliminated will be correspondingly small. In Figure 5, we show four sample ORFs, one from the Plasmo dataset, and three from the yeast datasets.

Fig 5. Slope Values are smaller in the Plasmo ORF (A) than in the Cho-Spellman yeast datasets ORFs (B).

If the number of the sampling time points is high compared to the number of alternations in the biological expression level, the data has a high level of serial dependency; this will result a higher output size. Even though the Plasmo dataset has much longer time series than the yeast datasets, it has...
much fewer alternation points, which result a higher output size for the Plasmo dataset than CDC-15, CDC-28 and Alpha datasets as shown in Figure 3.

Plus, if the data is suitable, one could handle time delay in the ART model by shifting the particular bits in the bit-vectors of the target ORF by an integer q, such that for a time series with a sampling period T, \((q * T)\) should account for the time-delay.

One other factor plays a role in the output size. This factor is the ORF change distribution of each dataset (Figure 6). If a large percentage of slopes are distributed in the range between \(-\alpha\) and \(\alpha\), this will increase the output size. Since the percentage of slopes in each of the 8 vectors in (1) and (2) will be affected.

The tuning factor \(\beta\) must have a small value, especially if the percentage of slopes that are around zero is high. In all of the four datasets that have been used, this value is high, so \(\beta\) was chosen to be just 0.01.

So the output size and ES varied dramatically from one dataset to another, this is due to:
- The longer the time series is, the more checkpoints there are, so more triplets will be eliminated at the testTarget().
- The more percentage of slopes is outside the range \([-\alpha, \alpha]\), the higher is the number of checkpoints that are active as shown in Figure 6.

![Slope Distribution % for the Plasmo Dataset](image)

Fig. 6. When using only the first neighbor, slope values are more concentrated in the center range than the case of all neighbors.

The more serial dependency in the slope values, the more triplets pass to the output.

V. CONCLUSION

We presented a new pre-processing approach for reducing candidate regulation relation that is needed in the process of predicting gene regulatory network from time series expression microarray data. This algorithm was tested on four datasets and it was shown that its mining efficiency could vary for each dataset based on three different factors, time series size, slope distribution, and serial dependency in the time series.