ProPhyC: A Probabilistic Phylogenetic Model for Refining Regulatory Networks

Xiuwei Zhang and Bernard M.E. Moret

Laboratory for Computational Biology and Bioinformatics
EPFL (Ecole Polytechnique Fédérale de Lausanne), Switzerland
and Swiss Institute of Bioinformatics
{xiuwei.zhang,bernard.moret}@epfl.ch

Abstract. The experimental determination of transcriptional regulatory networks in the laboratory remains difficult and time-consuming, while computational methods to infer these networks provide only modest accuracy. The latter can be attributed in part to the limitations of a single-organism approach. Computational biology has long used comparative and, more generally, evolutionary approaches to extend the reach and accuracy of its analyses. We therefore use an evolutionary approach to the inference of regulatory networks, which enables us to study evolutionary models for these networks as well as to improve the accuracy of inferred networks.

We describe *ProPhyC*, a probabilistic phylogenetic model and associated inference algorithms, designed to improve the inference of regulatory networks for a family of organisms by using known evolutionary relationships among these organisms. *ProPhyC* can be used with various network evolutionary models and any existing inference method. We demonstrate its applicability with two different network evolutionary models: one that considers only the gains and losses of regulatory connections during evolution, and one that also takes into account the duplications and losses of genes. Extensive experimental results on both biological and synthetic data confirm that our model (through its associated refinement algorithms) yields substantial improvement in the quality of inferred networks over all current methods.

1 Introduction

Transcriptional regulatory networks are models of the cellular regulatory system that governs transcription. Because establishing the topology of the network from bench experiments is very difficult and time-consuming, regulatory networks are commonly inferred from gene-expression data. Various computational models, such as Boolean networks [1], Bayesian networks [9], dynamic Bayesian networks (DBNs) [15], and differential equations [6], have been proposed for this purpose. Results, however, have proved mixed: the high noise level in the data, the paucity of well studied networks, and the many simplifications in the models all combine to make inference difficult, in terms of both accuracy and computation.

Bioinformatics has long used evolutionary approaches to improve the accuracy of computational analyses. Recent work on the evolution of regulatory networks has

J. Chen, J. Wang, and A. Zelikovsky (Eds.): ISBRA 2011, LNBI 6674, pp. 344–357, 2011.

demonstrated the applicability of such approaches to regulatory networks. Although regulatory networks produced from bench experiments are available for only a few model organisms, other types of data have been used to assist in the comparative study of regulatory mechanisms across organisms. For example, gene-expression data [22], sequence data such as transcription factor binding site (TFBS) [7,21], and *cis*-regulatory elements [22] have all been used in this context. Moreover, a broad range of model organisms have been studied, including bacteria [3], yeast [7,22], and fly [21]. These studies have identified a number of evolutionary events, such as adding or removing network edges, and the duplication and loss of genes [3,20,23]. Results have also appeared on the evolution of metabolic networks and protein interaction networks [4,17].

Phylogenetic relationships are well established for many groups of organisms; as the regulatory networks evolved along the same lineages, the phylogenetic relationships informed this evolution and so can be used to improve the inference of regulatory networks. Indeed, Bourque and Sankoff [5] developed an integrated algorithm to infer regulatory networks across a group of species whose phylogenetic relationships are known, under a simple parsimony criterion. In previous work [25,26], we presented *refinement* algorithms, based on phylogenetic information and using a likelihood framework, that boost the performance of any chosen network inference method, hereafter called a *base* method. These refinement algorithms, *RefineFast* and *RefineML*, are two-step iterative algorithms. The networks to be refined are placed at the corresponding leaves of the known phylogeny. In the first step, ancestral networks for the phylogeny (strings labelling internal nodes) are inferred; in the second step, these ancestral networks are used to refine the leaf networks. These two steps are then repeated as needed. On both simulated and biological data, the *receiver-operator characteristic (ROC)* curves for our algorithms consistently dominated those of the base methods used alone.

We present *ProPhyC*, a probabilistic phylogenetic model and associated algorithms, designed to refine regulatory networks for a family of organisms. *ProPhyC* can accommodate a large variety of evolutionary models of regulatory networks with only slight modifications, as we demonstrate in the results section. Given that the evolution of regulatory networks is not yet well understood and given the several different models for regulatory network evolution [7,23,5], such flexibility is highly desirable. We present algorithms and experimental results in this refinement model for two network evolutionary models: a basic model that includes only gains and losses of regulatory interactions, and an extended model that also accounts for duplications and losses of genes. We also show how to take advantage of position-specific confidence values, if any, assigned to the input networks by the base inference method. Our probabilistic phylogenetic model confirms the usefulness of phylogenetic information in obtaining better inference of regulatory networks. Extensive experiments show that *ProPhyC* model not only brings significant improvement to base network inference algorithms, but also dominates the performance of existing refinement algorithms.

2 Background

Our approach posits that the evolution of regulatory networks correlates strongly with the evolution of the respective organisms, so that independent network inference errors can be corrected by using the phylogenetic relationships between the networks.

2.1 Base Network Inference Methods

We chose dynamic Bayesian inference (*DBI*), the method devised for DBNs, as the base inference method in our experiments. When DBNs are used to model regulatory networks, an associated structure-learning algorithm is used to infer the networks from gene-expression data [15,16]; so as to avoid overly complex networks, a penalty on graph structure complexity is usually added to the ML score, thereby reducing the number of false positive edges. In [25] we used a coefficient k_p to adjust the weight of this penalty and studied different tradeoffs between sensitivity and specificity, yielding the optimization criterion $\log Pr(D|G,\hat{\Theta}_G) - k_p \#G \log N$, where D denotes the dataset used in learning, G is the (structure of the) network, $\hat{\Theta}_G$ is the ML estimate of parameters for G, $\#G$ is the number of free parameters of G, and N is the number of samples in D.

2.2 Reconciliation of Species Tree and Gene Trees

To recover the gene contents of ancestral networks under the extended model, we need a full history of gene duplications and losses. We reconstruct this history by reconciling the gene trees and the species tree, that is, by using the differences between these trees to infer past duplication and loss events. While reconciliation is a hard computational problem, algorithms have been devised for it in a Bayesian framework [2] or using a simple parsimony criterion, as in the software Notung [8].

3 Models and Methods

We begin by presenting two network evolutionary models, then describe the *ProPhyC* refinement framework, and finally give associated refinement algorithms, one for each network evolutionary model.

3.1 Network Evolutionary Models

We present a basic model and an extended model. In both models, the networks are represented by binary adjacency matrices. For the basic model, the evolutionary operations are: *edge gain*, in which an edge between two genes is generated with probability p_{01}, and *edge loss*, an existing edge is deleted with probability p_{10}. The model parameters are thus (i) the base frequencies of 0 and 1 entries in the given networks $\Pi = (\pi_0\ \pi_1)$, and (ii) the substitution matrix of 0s and 1s, $P = (p_{ij})$. The extended model has two additional evolutionary operations, *gene duplication* and *gene loss*, with corresponding additional model parameters p_d and p_l. In gene duplication, a gene is duplicated with probability p_d; after duplication, edges for the newly generated copy are assigned according to (i) *neutral initialization*, where the new copy gets connected to other genes randomly according to the proportion π_1 of edges in the background network; or (ii) *inheritance initialization*, where the new copy inherits the connections of the original, then loses or gains connections at some fixed rate, following reports of strong correlations between the connections of the new copy and those of the original copy [3,20,23]. In gene loss, a gene is deleted along with all its connections with probability p_l.

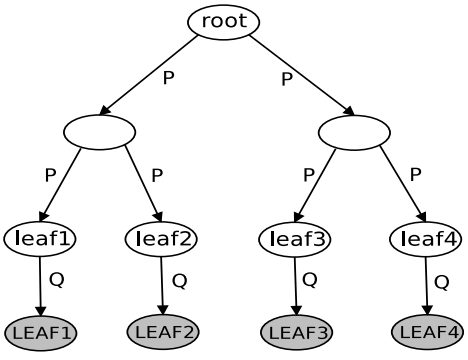


Fig. 1. *The* ProPhyC *model*

3.2 The *ProPhyC* Framework

ProPhyC is a probabilistic phylogenetic model designed to refine the inferred (and error-prone) regulatory networks for a family of organisms by making use of known phylogenetic information for the family. ProPhyC is also a graphical model: the phylogeny of this family is the main information to determine its structure as illustrated in Fig. 1. The shaded nodes labeled in upper case represent the input noisy networks, while the nodes labeled in lower case represent the correct networks for these organisms that we want to infer. In turn, the correct networks are the leaves of the rooted phylogenetic tree of these organisms, while internal nodes correspond to ancestral regulatory networks. The edges in this graph fall into two categories: (i) edges in the phylogenetic tree, representing the evolution from a parent network to a child network, and (ii) edges from correct leaf networks to noisy ones, representing the error-prone process of inferring networks from latent correct networks. The parameters for this model are thus the substitution matrices P and Q, where P represents the transition parameters from an ancestral network to its child network—subject to the network evolutionary model—and Q represents the difference between the "true" networks and the inferred (observed, from the point of view of the *ProPhyC* model) noisy networks—associated with one's confidence in the base network inference method.

The input information is thus the evolutionary model, the phylogenetic tree, and the noisy leaf networks. With a dynamic programming algorithm to maximize the likelihood of the whole graph, we can infer the ancestral networks and the "true" leaf networks. These "true" leaf networks inferred are the refined networks for these organisms and the output of the refinement algorithm. The framework can easily be generalized to fit different network evolutionary models. Some base inference methods can predict regulatory networks with different confidence on different edges or non-edges of the networks, so in this case Q can vary for different entries of different leaf networks. Our model can incorporate these position-specific confidence values to get better refinements. We name this version of the refinement algorithm *ProPhyCC*.

3.3 *ProPhyC* Under the Basic Model

Under the basic model, all networks have the same size and gene contents. Each network is represented by its binary adjacency matrix, so the character set is $S = \{0,1\}$. The parameters to calculate the likelihood are those from the evolutionary model, Π and P, and the error parameter for the base inference method, $Q = (q_{ij})$. We assume independence between the network entries, so that we can process separately each entry in the adjacency matrices. Let i, j, k denote nodes in the tree and $a,b,c \in S$ denote possible values of a character. For each character a at each node i, we maintain two variables:

- $L_i(a)$: the likelihood of the best reconstruction of the subtree with root i, given that the parent of i is assigned character a.
- $C_i(a)$: the optimal character for i, given that its parent is assigned character a.

When the phylogenetic tree is binary, our inference algorithm works as follows:

1. For each leaf node i, if its corresponding noisy network has character b, then for each $a \in S$, set $L_i(a) = \max_{c \in S} p_{ac} \cdot q_{cb}$ and $C_i(a) = \arg\max_{c \in S} p_{ac} \cdot q_{cb}$.
2. If i is an internal node and not the root, its children are j and k, and it has not yet been processed, then for each $a \in S$, set $L_i(a) = \max_{c \in S} p_{ac} \cdot L_j(c) \cdot L_k(c)$ and $C_i(a) = \arg\max_{c \in S} p_{ac} \cdot L_j(c) \cdot L_k(c)$.
3. If there remain unvisited nonroot nodes, return to Step 2.
4. If i is the root node, with children j and k, assign it the value $a \in S$ that maximizes $\pi_a \cdot L_j(a) \cdot L_k(a)$.
5. Traverse the tree from the root, assigning to each node its character by $C_i(a)$.

3.4 *ProPhyC* Under the Extended Model

The extended model includes gene duplications and losses, so that the gene content may vary across networks. While the gene content of the leaf networks is known, we need to reconstruct the gene content for ancestral networks, that is, to reconstruct the history of gene duplications and losses. This part can be solved by using an algorithm to reconcile the gene trees and species tree [2,8,19] or by the algorithms that we presented in earlier work under the *duplication-only* or *loss-only* model [27].

Under the basic model, we assumed independence among the entries of the adjacency matrices and so greatly simplified the computation. To enable us to do the same under the extended model, we embed each network into a larger one that includes every gene that appears in any network. We then represent a network with a ternary adjacency matrix, where the rows and columns of the missing genes are filled with a special character x. All networks are thus represented with adjacency matrices of the same size. Since the gene contents of ancestral networks are known thanks to reconciliation, the entries with x are already identified in their matrices; the other entries are reconstructed by the refinement algorithm using the new character set $S' = \{0,1,x\}$. The substitution matrix P' for S' can be derived from the model parameters, without introducing new parameters. Assuming that at most one gene duplication and one gene loss can happen at each evolutionary step, we have:

$$P' = \begin{pmatrix} p'_{00} & p'_{01} & p'_{0x} \\ p'_{10} & p'_{11} & p'_{1x} \\ p'_{x0} & p'_{x1} & p'_{xx} \end{pmatrix} = \begin{pmatrix} (1-p_l)\cdot p_{00} & (1-p_l)\cdot p_{01} & p_l \\ (1-p_l)\cdot p_{10} & (1-p_l)\cdot p_{11} & p_l \\ p_d \cdot \pi_0 & p_d \cdot \pi_1 & 1-p_d \end{pmatrix} .$$

We also extend the parameter Q to be Q' to fit the new character set S':

$$Q' = \begin{pmatrix} q'_{00} & q'_{01} & q'_{0x} \\ q'_{10} & q'_{11} & q'_{1x} \\ q'_{x0} & q'_{x1} & q'_{xx} \end{pmatrix} = \begin{pmatrix} q_{00} & q_{01} & 0 \\ q_{10} & q_{11} & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad .$$

The transition probabilities in Q' remain the same as in Q, since the gene contents of the "true" and corresponding noisy network are the same. For each character a at each tree node i, we calculate $L_i(a)$ and $C_i(a)$ for each site with the following procedure:

1. For each leaf node i, if its corresponding noisy network has character b, then for each $a \in S'$, set $L_i(a) = \max_{c \in S'} p'_{ac} \cdot q'_{cb}$ and $C_i(a) = \arg\max_{c \in S'} p'_{ac} \cdot q'_{cb}$.
2. If i is an internal node and not the root, its children are j and k, and it has not yet been processed, then
   - if i has character x, for each $a \in S'$, set $L_i(a) = p'_{ax} \cdot L_j(x) \cdot L_k(x)$ and $C_i(a) = x$;
   - otherwise, for each $a \in S'$, set $L_i(a) = \max_{c \in S} p'_{ac} \cdot L_j(c) \cdot L_k(c)$ and $C_i(a) = \arg\max_{c \in S} p'_{ac} \cdot L_j(c) \cdot L_k(c)$.
3. If there remain unvisited nonroot nodes, return to Step 2.
4. If i is the root node, with children j and k, assign it the value $a \in S$ that maximizes $\pi_a \cdot L_j(a) \cdot L_k(a)$, if the character of i is not already identified as x.
5. Traverse the tree from the root, assigning to each node its character by $C_i(a)$.

3.5 Refinement Algorithm *ProPhyCC* Using Confidence Values

Parameter Q (or Q') models the errors introduced in the base inference process; its values are obtained from one's confidence in that method and in the source data. The *ProPhyC* algorithm uses the same matrix for all entries in all leaf networks. When sufficient information is available to produce different confidence values for different entries in different networks, we can take advantage of the extra information through the *ProPhyCC* algorithm.

If the noisy networks are predicted from gene-expression data by DBN models, to obtain the confidence values, we first estimate the conditional probability tables (CPTs) of the *DBI* inferred networks from the gene-expression data on the inferred structure [11], and then calculate the confidence values from the CPTs. Following [16], we use binary gene-expression levels in our experiments, where 1 and 0 indicate that the gene is, respectively, *on* and *off*. For each gene g_i, if m_i nodes have arcs directed to g_i in the network, let the expression levels of these nodes be denoted by the vector $y = y_1 y_2 \cdots y_{m_i}$ and the confidence values of their arcs by the vector $c = c_1 c_2 \cdots c_{m_i}$. We use signed weights to represent the strength of these arcs, denoted by $w = w_1 w_2 \cdots w_{m_i}$. Considering that if an arc is predicted with high weight, then this arc is very likely to be true, we assign high confidence values to the arcs predicted with high absolute weight values. Let k be a coefficient value to normalize probabilities, we have $k \cdot w \cdot y = Pr(g_i \text{ is } on|y)$. Since there are 2^{m_i} configurations of y, there are 2^{m_i} such equations. The value of $Pr(g_i \text{ is } on|y)$ can be directly taken from the CPTs. So w can be obtained by solving these equations, and c derived directly from w.

4 Experimental Design

We designed a comprehensive collection of experiments to assess our model and its associated algorithms. The accuracy of the output is calculated by comparing the output with the "true" networks for the chosen family of organisms, where the "true" networks are either obtained through simulation or collected from biological datasets. We compare the accuracies of the networks produced by the base method *DBI* and of the networks after refinement, to get *absolute* assessments. We also use our previous refinement algorithms [25,26,27] to refine the same networks and compare the outcome with that of our new refinement model, to get *relative* assessments.

Since regulatory networks are usually reconstructed from gene-expression data, we follow the same path in our assessment. With networks inferred from gene-expression data as input for *ProPhyC*, *ProPhyCC*, *RefineFast* and *RefineML*, we run experiments with different combinations of networks evolutionary models and types of datasets. Under each setting, we show both absolute and relative assessments.

4.1 Biological Data Collection

Transcription factor binding site (TFBS) data is used to study regulatory networks, assuming that the regulatory interactions determined by transcription factor binding share many properties with the real interactions [7,10,21]. Given this close relationship between regulatory networks and TFBSs and given the large amount of available data on TFBSs, we chose to use TFBS data to derive regulatory networks for the organisms as their "true" networks. The TFBS data is drawn from the work of Kim *et al.* [14], where the TFBSs are annotated for the *Drosophila* family (whose phylogeny is well studied) with 12 species. They reported TFBS annotations for 7 transcription factors on 51 cis-regulatory modules (CRMs) for all 12 species. Since each CRM corresponds to a target gene, we get a regulatory network with 58 nodes for each organism as the "true" network for this organism. We add noise into these "true" networks to obtain noisy networks as input to our refinement algorithm.

4.2 Data Simulation

In simulation experiments, we generate gene-expression data from simulated leaf networks. This step helps in decoupling the generation and the reconstruction phases. The data simulation procedure consists of two main steps: (i) generate the "true" leaf networks according to the evolutionary model and (ii) generate the gene-expression data. The whole process starts from three pieces of information: the phylogenetic tree, the network at its root, and the evolutionary model. Since we need quantitative relationships in the networks in order to generate gene-expression data from each network, in the network generation process, we use adjacency matrices with signed weights.

We take specific precautions against systematic bias during data simulation and result analysis. We use a wide variety of phylogenetic trees from the literature (of modest sizes: between 20 and 60 taxa) and several choices of root networks, the latter variations on part of the yeast network from the KEGG database [13]. The root network has between 14 and 17 genes, a relatively easy case for inference algorithms and thus a more challenging case for a refinement algorithm. We explore a wide range of evolutionary rates, including rates of gene duplication and loss and rates of edge gain and loss.

Simulating networks. Denote the weighted adjacency matrix of the root network as A_p. Under the basic model, we obtain the adjacency matrix for its child A_c by mutating A_p according to the substitution matrix. By repeating this process as we traverse down the tree we obtain weighted adjacency matrices at the leaves. In other words, we evolve the weighted networks down the tree according to the model parameters, following standard practice in the study of phylogenetic reconstruction [12,18]. Under the extended model, to get the adjacency matrix for the child network of A_p, we follow two steps: evolve the gene contents and evolve the regulatory connections. First, genes are duplicated or lost by p_d and p_l. If a duplication happens, a row and column for this new copy will be added to A_p, the values initialized either according to the *neutral initialization* model or the *inheritance initialization* model. Call this intermediate adjacency matrix A'_c. Now edges in A'_c are mutated according to p_{01} and p_{10} to get A_c. Again we repeat this process as we traverse down the tree to obtain weighted adjacency matrices at the leaves.

Generating gene-expression data. From the "true" networks, we use *DBNSim* [25], based on the DBN model, to generate time-series gene-expression data. Note that, while *DBNSim* and *DBI* are both based on Bayesian networks, which might artificially improve the performance of *DBI*, this bias can only make it more difficult for *ProPhyC* to achieve significant improvements.

For all experiments on simulated gene-expression data, we run the generation process 10 times for each choice of tree structure and parameters to compute a mean and a standard deviation. Under the basic model, for each leaf network, we generate 200 time points for its gene-expression matrix. Under the extended model, we generate $13 \times n$ time points for a leaf network with n genes, since larger networks generally need more samples to gain inference accuracy comparable to smaller ones.

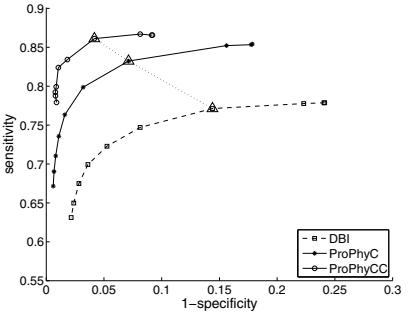
4.3 Measurements

We want to examine the predicted networks at different levels of sensitivity and specificity. With *DBI*, we use a penalty coefficient on structure complexity so as to obtain different tradeoffs between sensitivity and specificity. On each dataset, we apply different penalty coefficients to predict regulatory networks, from 0 to 0.5, with an interval of 0.05, which results in 11 discrete coefficients. For each penalty coefficient, we apply our approach (and any method chosen for comparison) on the predicted networks, measure specificity and sensitivity, and plot the values into ROC curves. (In these ROC plots, the closer the curves are to the top left corner of the coordinate space, the better the results.)

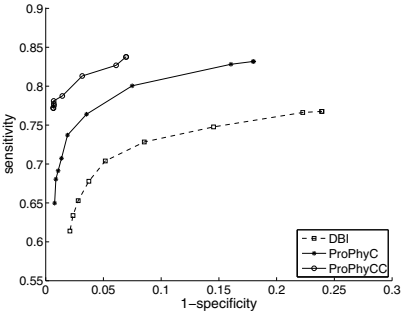
5 Results and Analysis

5.1 Performance Under the Basic Model on Simulated Data

Absolute results. We show experimental results on two representative trees: one has 37 nodes on 7 levels and the other has 41 nodes on 6 levels. We only plot part of the curves within the 11 penalty coefficients to give a more detailed view of the comparison. Fig. 2 shows the results of *ProPhyC* and *ProPhyCC* on the networks predicted by *DBI*. We can see that *ProPhyC* and *ProPhyCC* significantly improve both sensitivity and specificity over the base inference algorithm *DBI*. The improvement remains similar on different

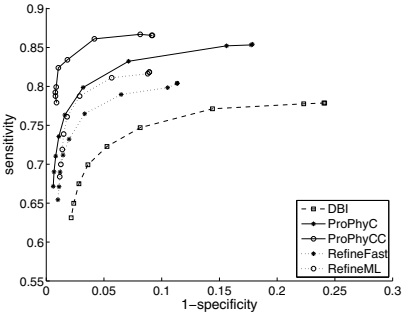


(a) On the tree with 37 nodes and 7 levels

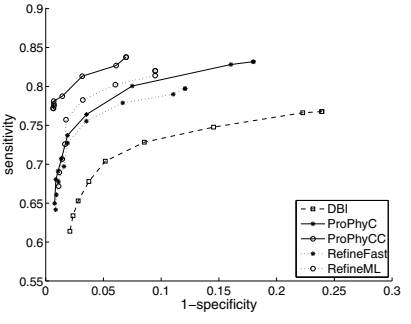


(b) On the tree with 41 nodes and 6 levels

Fig. 2. Comparison of *ProPhyC* and *ProPhyCC* with base inference algorithm *DBI* under the basic model. In part (a), the dotted lines join data points for the same model penalty coefficient.



(a) On the tree with 37 nodes and 7 levels



(b) On the tree with 41 nodes and 6 levels

Fig. 3. Comparison of *ProPhyC* and *ProPhyCC* with *RefineFast* and *RefineML* under the basic model

tree structures. *ProPhyCC* further improves *ProPhyC*, which shows the advantage of using position-specific confidence values. For example, the dots in Fig. 2(a) marked by triangles correspond to the same penalty coefficient on the three curves, showing that, in going from *DBI* to *ProPhyCC*, the sensitivity increases from 77% to 86% while the specificity increases from 86% to 96%. Similar improvements can be observed with other trees, other evolutionary rates, and other base methods.

Relative results. Fig. 3 shows the same experiments as in Fig. 2, but adds curves for *RefineFast* and *RefineML* to provide a comparison between different refinement approaches. Among the four refinement algorithms, *ProPhyCC* and *RefineML* take advantage of the position-specific confidence values, which gives them better performance than *ProPhyC* and *RefineFast*. *ProPhyCC* is obviously the best among all refinement algorithms, while *ProPhyC* outperforms *RefineFast*. From Figs. 2 and 3, we conclude that refinement algorithms under our new model outperform not only base inference algorithms, but also our previous refinement algorithms on simulated data.

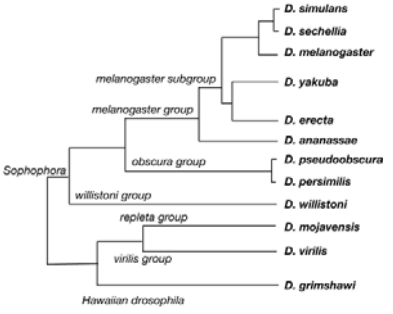


Fig. 4. The phylogeny connecting the 12 Drosophila species [24]

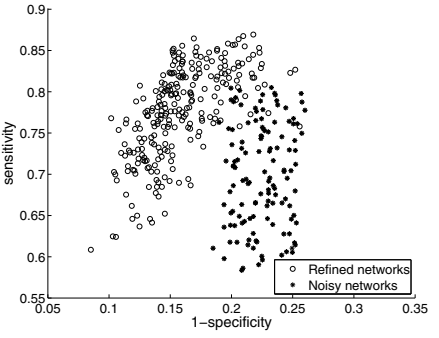


Fig. 5. Results of *ProPhyC* under the basic model on biological datasets

Table 1. Performance of ProPhyC under different tradeoffs

	sensitivity (noisy → refined)	specificity (noisy → refined)
improve both	59.9% → 66.3%	80.0% → 86.5%
focus on sensitivity	59.5% → 69.2%	69.3% → 72.7%
focus on specificity	57.7% → 58.5%	70.1% → 80.0%

5.2 Performance Under the Basic Model on Biological Data

Here we show our results on the datasets for 12 species of *Drosophila*, whose phylogenetic tree is illustrated in Fig. 4. We use different noise rates to get noisy networks with different false positives and false negatives. Then for each set of noisy networks we use *ProPhyC* to obtain refined networks with different parameter settings. Fig. 5 shows the accuracies of these networks plotted as points. The cloud of points for *ProPhyC* clearly dominates that of the noisy networks, and the two clouds are well separated; the average improvement brought by *ProPhyC* is roughly 7% in each of sensitivity and specificity.

By adjusting the penalty parameter, we can choose whether to emphasize sensitivity over specificity or the reverse, i.e., we can choose in which part of the ROC curve to operate; Table 1 gives some examples.

5.3 Performance Under the Extended Model on Simulated Data

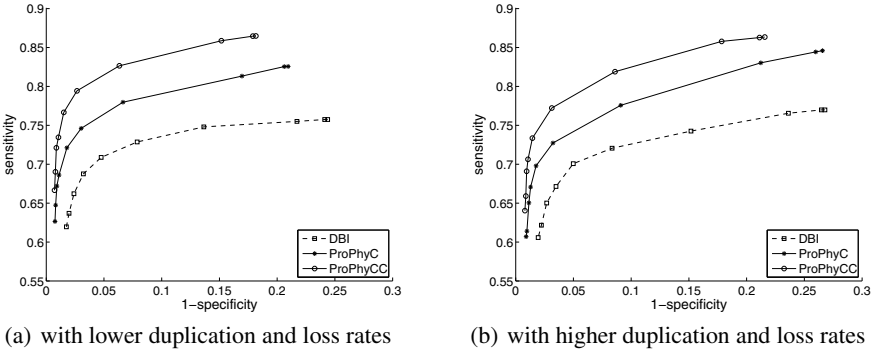
In evaluating performance under the extended model, we must first consider the effect of the first phase, in which the history of gene duplications and losses is reconstructed. In [27] we analyzed various duplication-loss history models and their effect on the performance of *RefineFast* and *RefineML*. Our experiments showed that accurate history information with reliable orthology assignments helps the refinement algorithms to get good performance. Here we test *ProPhyC* and *ProPhyCC* with two representative histories. One is the "true" history which is available in the framework of simulation experiments; with this history we can exclude the error introduced by the history inference step, and test purely the performance of the refinement algorithms. The other is the history inferred by gene tree and species tree reconciliation algorithms without any

prior information. As the rates of gene duplication and loss during evolution can also affect the performance of refinement algorithms, we conduct simulation experiments with different rates of duplication and loss.

We run refinement algorithms with the two gene duplication and loss histories: the true history and the history reconstructed by Notung [8]. In the following we show results on one representative phylogenetic tree with 35 nodes on 7 levels, and a root network of 15 genes. Since the results of using the neutral initialization model or the inheritance initialization model in data generation are very similar, we only show results with the former. For each experiment we show two plots: the left plot has relatively low rates of gene loss (resulting in 19 duplications and 15 losses along the tree on average), while the right one has significantly higher rates(with 20 duplications and 23 losses).

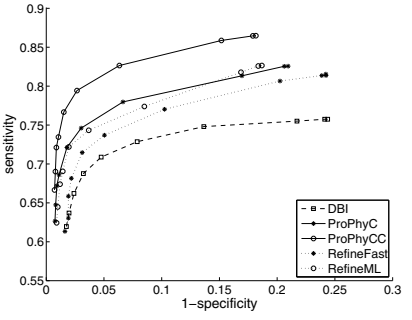
Absolute results, with true history. Fig. 6 shows the comparison of *ProPhyC*, *ProPhyCC* and the base inference algorithm *DBI*, using the true history of duplications and losses. Given the size of the tree and the root network, the rates of gene duplication and loss are quite high, yet the improvement gained by our refinement algorithms remains significant in both plots – almost as much as the improvement gained under the basic model shown in Fig. 2. *ProPhyCC* further dominates *ProPhyC* in both sensitivity and specificity, thanks to the appropriate use of the position-specific confidence values. Once again, we obtain similar improvements with other trees, other evolutionary rates, and other base methods.

Relative results, with true history. Fig. 7 shows the results of the same experiments as in Fig. 6, with *RefineFast* and *RefineML* added. Although *RefineFast* and *RefineML* still clearly improve on *DBI*, the improvement is less pronounced than with the basic model (Fig. 3). Gene duplications and losses give rise to a large overall gene population, yet many of them exist only in a few leaf networks; for these underrepresented genes, phylogenetic information is much reduced and so the refinement is less successful. *RefineFast* and *RefineML* are affected by this shortage, however, *ProPhyC* and *ProPhyCC* are more robust and easily outperform *RefineFast* and *RefineML*.

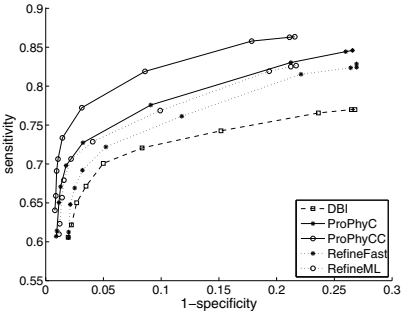


(a) with lower duplication and loss rates (b) with higher duplication and loss rates

Fig. 6. Results of refinement algorithms with extended network evolutionary model, comparison of *ProPhyC* and *ProPhyCC* with *DBI*, with true gene duplication and loss history

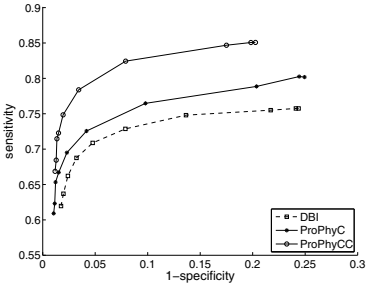


(a) with lower duplication and loss rates

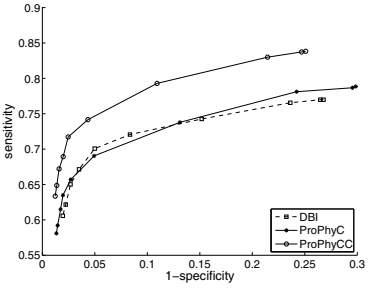


(b) with higher duplication and loss rates

Fig. 7. Results of refinement algorithms with extended model, comparison of *ProPhyC* and *ProPhyCC* with *RefineFast* and *RefineML*, with true gene duplication and loss history



(a) with lower duplication and loss rates



(b) with higher duplication and loss rates

Fig. 8. Results of refinement algorithms with extended network evolutionary model, comparison of *ProPhyC* and *ProPhyCC* with *DBI*, with inferred gene duplication and loss history by Notung

Absolute results, with inferred history. Here we use Notung to reconstruct the history of duplications and losses without any orthology input. In these experiments, with reliable gene tree input, Notung correctly predicts duplication events (modulo changes in the networks), but usually misses recent loss events (it shows those events as happening earlier on the lineages). Furthermore, Notung not only infers the gene contents for ancestral networks, but also alters the gene contents of the leaves, which causes some difficulty for the refinement procedure. Fig. 8 shows the results of *ProPhyC* and *ProPhyCC* with Notung reconstructed gene contents for the ancestral networks. We see that in Fig. 8(a), the two ends of the *ProPhyC* curve have lost a little specificity while gaining sensitivity or vice versa, a tradeoff rather than an outright gain. However, *ProPhyC* dominates *DBI* through the useful range of specificity and sensitivity. In Fig. 8(b), *ProPhyC* barely improves *DBI*, because the high rate of gene loss reduces the performance of refinement algorithms in two ways: first a high rate affects the performance of Notung (which does a poor job at inferring losses); secondly it increases the total population of genes and decreases the frequency of occurrence of an ortholog in the leaf networks, thus limiting the phylogenetic information. However, *ProPhyCC* still improves *DBI* significantly in both plots. Our probabilistic framework can incorporate the prior

information in an appropriate way, so as to gain good performance even when the phylogenetic information, including the history of gene duplication and loss, is noisy and incomplete.

6 Conclusions

We described *ProPhyC*, a probabilistic phylogenetic model designed to improve the inference of regulatory networks for a family of organisms by using the phylogenetic relationships among these organisms. This model and its associated refinement algorithms can easily be adapted to work with different network evolutionary models. We conducted experiments on both simulated and biological data to test the performance of the refinement algorithms. With both the basic and extended network evolutionary models, the corresponding versions of *ProPhyC* and *ProPhyCC* outperformed those of our previous algorithms *RefineFast* and *RefineML*, and all four refinement algorithms outperformed the base inference algorithm. The improvement of *ProPhyC* and *ProPhyCC* over *RefineFast* and *RefineML* was more significant under the extended model, where the performance of *RefineFast* and *RefineML* was affected by the decrease of the phylogenetic information for each ortholog, while *ProPhyC* and *ProPhyCC* were hardly influenced. Our probabilistic phylogenetic model is thus quite robust against changes in these network evolutionary models. Our probabilistic phylogenetic model can easily be extended into a probabilistic graphical model to incorporate the evolution of both regulatory networks and binding sites.

References

1. Akutsu, T., Miyano, S., Kuhara, S.: Identification of genetic networks from a small number of gene expression patterns under the Boolean network model. In: Proc. 4th Pacific Symp. Biocomp. (PSB 1999), pp. 17–28. World Scientific, Singapore (1999)
2. Arvestad, L., Berglund, A.-C., Lagergren, J., et al.: Gene tree reconstruction and orthology analysis based on an integrated model for duplications and sequence evolution. In: Proc. 8th Conf. Comput. Mol. Bio. (RECOMB 2004), pp. 326–335. ACM Press, New York (2004)
3. Babu, M.M., Teichmann, S.A., Aravind, L.: Evolutionary dynamics of prokaryotic transcriptional regulatory networks. J. Mol. Bio. 358(2), 614–633 (2006)
4. Berg, J., Lassig, M., Wagner, A.: Structure and evolution of protein interaction networks: a statistical model for link dynamics and gene duplications. BMC Evol. Bio. 4(1), 51 (2004)
5. Bourque, G., Sankoff, D.: Improving gene network inference by comparing expression time-series across species, developmental stages or tissues. J. Bioinform. Comput. Bio. 2(4), 765–783 (2004)
6. Chen, T., He, H.L., Church, G.M.: Modeling gene expression with differential equations. In: Proc. 4th Pacific Symp. Biocomp. (PSB 1999), pp. 29–40. World Scientific, Singapore (1999)
7. Crombach, A., Hogeweg, P.: Evolution of evolvability in gene regulatory networks. PLoS Comput. Biol. 4(7), e1000112 (2008)
8. Durand, D., Halldórsson, B.V., Vernot, B.: A hybrid micro-macroevolutionary approach to gene tree reconstruction. J. Comput. Bio. 13(2), 320–335 (2006)
9. Friedman, N., Linial, M., Nachman, I., Pe'er, D.: Using Bayesian networks to analyze expression data. J. Comput. Bio. 7(3-4), 601–620 (2000)

10. Harbison, C.T., Gordon, D.B., Lee, T.I., et al.: Transcriptional regulatory code of a eukaryotic genome. Nature 431, 99–104 (2004)
11. Heckerman, D.: Learning in graphical models. In: A Tutorial on Learning with Bayesian Networks, pp. 301–354. MIT Press, Cambridge (1999)
12. Hillis, D.M.: Approaches for assessing phylogenetic accuracy. Sys. Bio. 44, 3–16 (1995)
13. Kanehisa, M., Goto, S., Hattori, M., et al.: From genomics to chemical genomics: new developments in KEGG. Nucleic Acids Res. 34, D354–D357 (2006)
14. Kim, J., He, X., Sinha, S.: Evolution of regulatory sequences in 12 *Drosophila* species. PLoS Genet. 5(1), e1000330 (2009)
15. Kim, S.Y., Imoto, S., Miyano, S.: Inferring gene networks from time series microarray data using dynamic Bayesian networks. Briefings in Bioinf. 4(3), 228–235 (2003)
16. Liang, S., Fuhrman, S., Somogyi, R.: REVEAL, a general reverse engineering algorithm for inference of genetic network architectures. In: Proc. 3rd Pacific Symp. Biocomp. (PSB 1998), pp. 18–29. World Scientific, Singapore (1998)
17. Mithani, A., Preston, G.M., Hein, J.: A Bayesian approach to the evolution of metabolic networks on a phylogeny. PLoS Comput. Bio. 6(8), e1000868 (2010)
18. Moret, B.M.E., Warnow, T.: Reconstructing optimal phylogenetic trees: A challenge in experimental algorithmics. In: Fleischer, R., Moret, B.M.E., Schmidt, E.M. (eds.) Experimental Algorithmics. LNCS, vol. 2547, pp. 163–180. Springer, Heidelberg (2002)
19. Page, R.D.M., Charleston, M.A.: From gene to organismal phylogeny: Reconciled trees and the gene tree/species tree problem. Mol. Phyl. Evol. 7(2), 231–240 (1997)
20. Roth, C., Rastogi, S., Arvestad, L., et al.: Evolution after gene duplication: models, mechanisms, sequences, systems, and organisms. J. Exper. Zoology Part B: Mol. Devel. Evol. 308B(1), 58–73 (2007)
21. Stark, A., Kheradpour, P., Roy, S., Kellis, M.: Reliable prediction of regulator targets using 12 Drosophila genomes. Genome Res. 17, 1919–1931 (2007)
22. Tanay, A., Regev, A., Shamir, R.: Conservation and evolvability in regulatory networks: The evolution of ribosomal regulation in yeast. Proc. Nat'l Acad. Sci. 102(20), 7203–7208 (2005)
23. Teichmann, S.A., Babu, M.M.: Gene regulatory network growth by duplication. Nature Genetics 36(5), 492–496 (2004)
24. Tweedie, S., Ashburner, M., Falls, K., et al.: Flybase: enhancing Drosophila Gene Ontology annotations. Nucleic Acids Res. 37, D555–D559 (2009)
25. Zhang, X., Moret, B.M.E.: Boosting the performance of inference algorithms for transcriptional regulatory networks using a phylogenetic approach. In: Crandall, K.A., Lagergren, J. (eds.) WABI 2008. LNCS (LNBI), vol. 5251, pp. 245–258. Springer, Heidelberg (2008)
26. Zhang, X., Moret, B.M.E.: Improving inference of transcriptional regulatory networks based on network evolutionary models. In: Salzberg, S.L., Warnow, T. (eds.) WABI 2009. LNCS, vol. 5724, pp. 415–428. Springer, Heidelberg (2009)
27. Zhang, X., Moret, B.M.E.: Refining transcriptional regulatory networks using network evolutionary models and gene histories. BMC Algs. for Mol. Bio. 5(1), 1 (2010)