PReach: Reachability in Probabilistic Signaling Networks

Haitham Gabr, Andrei Todor, Helia Zandi, Alin Dobra, Tamer Kahveci
University of Florida, CISE Department, Gainesville, Florida 32611
{hgabr, atodor, helia, adobra, tamer}@cise.ufl.edu

ABSTRACT

Extracellular molecules trigger a response inside the cell by initiating a signal at special membrane receptors (i.e., sources) which is then transmitted to reporters (i.e., targets) through various chains of interactions among proteins. Understanding whether such a signal can reach from membrane receptors to reporters is essential in studying the cell response to extracellular events. This problem is drastically complicated due to the unreliability of the interaction data. In this paper, we develop a novel method, called *PReach* (**P**robabilistic **Reach**ability), that precisely computes the probability that a signal can reach from a given collection of receptors to a given collection of reporters when the underlying signaling network is uncertain. This is a very difficult computational problem with no known polynomial-time solution. PReach represents each uncertain interaction as a bivariate polynomial. It transforms the reachability problem to a polynomial multiplication problem. We introduce novel polynomial collapsing operators that associate polynomial terms with possible paths between sources and targets as well as the cuts that separate sources from targets. These operators significantly shrink the number of polynomial terms and thus the running time. PReach has much better time complexity than the recent solutions for this problem. Our experimental results on real datasets demonstrate that this improvement leads to orders of magnitude of reduction in the running time over the most recent methods.

Categories and Subject Descriptors

E.1 [**Data**]: DATA STRUCTURES—*Graphs and networks*; J.3 [**Computer Applications**]: LIFE AND MEDICAL SCIENCES—*Biology and genetics*

General Terms

Algorithms, Theory

BCB '13, September 22 - 25, 2013, Washington, DC, USA

http://dx.doi.org/10.1145/2506583.2506586.

Keywords

Signaling Networks, Probabilistic Networks, Reachability

1. INTRODUCTION

Major discoveries in biology have revealed the complexity of living organism at the cell level and beyond. A multitude of molecules such as proteins or genes interact to sustain the various functions served within cells. These interactions are often modeled as biological networks, where molecules are represented as nodes and interactions as edges. Among a diversity of biological networks, signaling networks describe how extracellular molecules trigger a response inside the cell. This often takes place by initiating a signal at special membrane proteins, (i.e., *receptors*). This signal travels through chains of interactions among proteins, until it reaches a transcription factor (i.e., *reporter*). The transcription factor then materializes the cell response as increase or reduction of the production of other proteins. Understanding this mechanism is of utmost importance since its malfunctioning is the cause of many disorders such as cancer, type 2 diabetes, neurodegeneration, Alzheimer, congenital malformations, obesity and osteoporosis [22, 24, 25].

One of the first steps in studying signaling networks is to establish their topologies. A common way to discover the physical interaction data is to use large scale experiments like yeast-two-hybrid and co-immunoprecipitation assays [9, 12]. These techniques produce high throughput results but they also introduce many false positives. The data produced are therefore uncertain. In other words an interaction may or may not take place with some probability.

Another known form of uncertainty in signaling networks is whether the underlying interactions hold under extreme conditions like severe trauma and burn injury. Research shows that stresses from such conditions affect more than 80% of cellular functions and pathways, resulting in a genomic storm [40]. Therefore even for curated signaling networks, an uncertain model encompassing such conditions is of utmost importance.

In the rest of the paper, we will use the term *probabilistic network* to denote a network with at least one uncertain reaction. To quantify the level of uncertainty, some recent interaction databases like MINT and STRING provide confidence values for each interaction [35, 7]. Various other methods have been proposed for rigorous assessment of the quality of the interaction data [37, 3].

In order to tell which external factors lead to various kinds of alterations in cell functions, it is very important to know whether a signal can reach from a receptor to a reporter

which is the problem considered in this paper. For example, the MAPK signaling network is responsible for oncogenesis; the tumor necrosis factor (TNF) is linked to jun proto-oncogene (c-Jun) transcription factor and the knowledge about this connection is very important in cancer and neurodegenerative diseases research [20]. In what follows we define the computational problem we address here.

Problem statement. *In this paper, we address the problem of finding whether a signal travels from a source (a given set of receptors) to the intended target (a given set of reporters) when each edge is subject to failure with a different probability, independently of the state of other edges.*

We formally define a *probabilistic signaling network* as a directed graph $G = (V, E, P)$, where V is the set of nodes (i. e. proteins), E is the set of edges (i.e. interactions) and P is a function that associates to each edge the probability of the existence of the interaction it represents. We assume that each edge exists independently of all other edges. This assumption is commonly used in the literature for similar problems[7, 35]. This problem applies to undirected networks by replacing each undirected edge with two edges in opposite directions.

The problem tackled in this paper is #P-complete [11] One way to appreciate the difficulty of the problem is to realize that a probabilistic network represents a large number of possible configurations of deterministic networks, depending on the presence or absence of each probabilistic edge. More precisely, a network with n probabilistic edges yield 2^n possible network configurations, as each one of the n edges may be present or absent. For instance, the probabilistic network in Figure 1 yields 32 alternative topologies (two of them shown at the bottom of Figure 1). An exact solution for the described problem would be to naively consider all of the 2^n possible cases. Also attesting to the difficulty of the problem is the fact that many approximate solutions have been proposed, for example based on Monte Carlo sampling. We elaborate on this in Section 2.

Contributions. In this paper, we develop a novel method, called *PReach* (**P**robabilistic **Reach**ability), that solves the problem described above exactly. PReach is inspired by the probability generating functions. In a recent study, we showed that these functions can be employed for aligning probabilistic networks [36]. We associate a polynomial function to each edge in the given network, defined by the edge probability. We call these polynomials the edge polynomials. We prove that the solution to the problem is given by a subset of the terms of the polynomial obtained as the multiplication of all the edge polynomials. We avoid excessive growth of the number of terms of the resulting polynomial by collapsing combinations of terms that we already know are solutions, and those that cannot lead to solutions, into a single term each. We also show that the order in which the edge polynomials are multiplied dictates how soon we can collapse the polynomial terms. Thus, it affects the speed at which the final polynomial grows. We develop a strategy to order the edge polynomials to maximize the number of collapsed terms (thus to minimize the polynomial size).

Our experimental results on real datasets demonstrate up to 10,000 times improvement in running time over the competing method when the competing method could run till completion. PReach scales up to larger networks easily. On such networks, it returned the results in minutes when the competing method failed to complete. We also perform biological validation experiments for our method. Our validation experiments show that the highly reachable proteins in the examined networks have important roles in biological functions like cytoneme morphogenesis and mitotic nuclear pore complex reassembly.

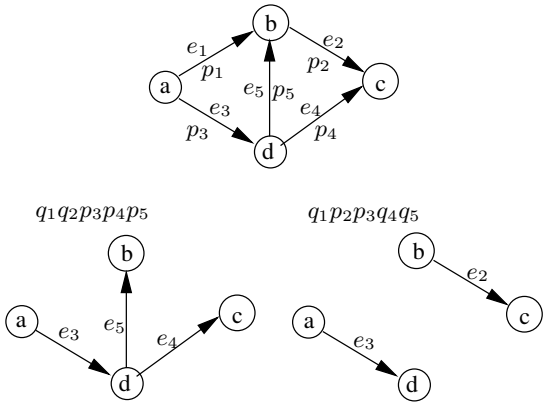


Figure 1: A probabilistic network (top) and two of the deterministic networks corresponding to it (bottom). Each of the deterministic networks is obtained from the probabilistic network with some probability determined by the probabilities of the edges that are included or not in the deterministic network. The expression above each deterministic network is the probability of observing it.

The rest of the paper is organized as follows. Section 2 discusses relevant background. Section 3 describes our method in its mathematical and algorithmic formulation. Section 4 presents the experimental evaluation of PReach and compares it with a recent existing method. Section 5 concludes the paper with a brief discussion.

2. BACKGROUND AND RELATED WORK

In the literature, the reachability problem in probabilistic networks has been studied mostly in the context of network reliability [17, 1, 39, 23]. The common drawback of all the available methods is that they do not scale well with growing network size.

A major class of existing methods is based on enumerating paths or cuts [30, 14], sometimes combined with the inclusion-exclusion method [39, 23]. One of the most recent solutions (and possibly the most relevant one from the point of this paper) that fall into this category is used by a method called SPINE [27]. This method aims to infer signaling and regulatory pathways. As a part of this goal, it computes the reachability probability between pairs of nodes. SPINE uses the inclusion-exclusion method to compute the reachability probability without error in order to reconstruct signaling pathways. SPINE assigns an activation/inhibition attribute to each protein in such a way that the network can satisfy the maximum number of observed gene knockouts. For a given knockout pair, it first calculates the probability that the target node is affected by the knockout gene using the inclusion-exclusion principle. After finding the reachability probability, it provides a linear programming formulation to assign the attributes to proteins. In this paper, we will compare the performance of PReach with the approach used in SPINE for calculating the reachability probability, since both methods compute this probability precisely. The time

complexity of this method is $O(2^N)$ with N being the number of all simple paths from the source to the target. *As we present in our experimental results (see Section 4) this method does not work for many real networks even for those with less than 40 interactions.*

Due to the computational difficulty of the problem, some methods resort to Monte Carlo approximations rather than computing precise probabilities [19, 10, 8, 21]. The disadvantage of these methods are (*i*) they do not produce an exact result and (*ii*) they need too many iterations to obtain a low error margin. Some of them focus on how to deploy a sampling policy or plan to reduce the sampling error. Zhu *et al.* [41] tried to overcome the need for a large number of samples by proposing a dynamic Monte Carlo simulation. This methods uses pruning based on some effective bounding techniques and if the pruning fails, it uses the dynamic Monte Carlo simulation.

A related, yet slightly different problem is the connectedness reliability problem. This problem seeks the probability that a network is connected (i.e., all the nodes are reachable from all the remaining nodes), under the assumption that each edge is subject to failure with the same probability p. A version of this problem is the s-t connectedness reliability, in which one seeks to find the probability that a target node t is reachable from a source node s. Both versions are known to be #P-complete [29, 6]. The time complexity of connectedness reliability was shown to be exponential in $\Omega(m/\log^2 m)$ [16].

Probabilistic networks have been analyzed in the past via polynomial manipulation. The reliability polynomial, a particularization of the Tutte polynomial, has received wide attention [28, 5]. This model, however, is limited to a single parameter, representing the probability that any edge in the network fails. Thus it is limited to deal only with global properties of the network, such as whether it is connected or not. It cannot encode more specific information, such as s-t connectivity. The multivariate Tutte polynomial was proposed as a generalization for the case in which each edge has a different failure probability, but again is limited to all-nodes reliability [34]. In the field of biological networks, some of the authors have proposed a detailed, comprehensive model based on probability generating functions for probabilistic alignment [36].

3. METHOD

In this section we describe our method. First, we build the mathematical foundation of our method (Section 3.1). We then establish the relationship between this theory and probabilistic networks (Section 3.2). We then use this relationship to model the reachability problem in probabilistic networks (Section 3.3) Finally, we present the PReach algorithm that solves the reachability problem on probabilistic networks (Section 3.4).

3.1 A special class of polynomials

In this section, we develop an expressive polynomial class, called the *xy-polynomial*, and the basic algebra that operates on it. As we will demonstrate in Sections 3.2 and 3.4, these polynomials naturally describe the topologies of all the deterministic instances of a given probabilistic network. Thus, the reachability problem in a given probabilistic network will turn into a simple evaluation of this polynomial. We start by introducing the notation that will help present our method.

Consider two sets of n variables $X = \{x_1, \ldots, x_n\}$ and $Y = \{y_1, \ldots, y_n\}$. Also consider the set of indices $U = \{1, \ldots, n\}$. Each subset S of U determines a subset of X and a subset of Y: $\{x_i | i \in S\}$ and $\{y_i | i \in S\}$, respectively. We denote the product of the variables in these two subsets with $x_S = \prod_{i \in S} x_i$ and $y_S = \prod_{i \in S} y_i$. Next, we introduce a class of polynomials, called the *xy-polynomials*, that play an important role in our method.

DEFINITION 1. *Let Θ be a subset of U. Let $S_1, \ldots, S_k$ be k different subsets of Θ, where k is a positive integer. Let x^* and y^* be two free variables. Let $a_1, \ldots, a_k$ be real numbers. Similarly, let b and c be two real numbers. An xy-polynomial over Θ is a polynomial of the form $F = \sum_{i=1}^{k} a_i x_{S_i} y_{\Theta \setminus S_i} + bx^* + cy^*$*

Notice that in the above summation, each term — except for the free variables — contains each of the indices $j \in \Theta$ either as a product term x_j or y_j. The following example clarifies this.

EXAMPLE 1. *Assume that $\Theta = \{1, 2, 3, 4\}$. This implies that $x_1x_3y_2y_4 + 2y_1y_2y_3y_4 + 3y^*$ is an xy-polynomial. However, the two polynomials x_3x_4 and $x_1x_2x_3x_4y_4 + 4x^* + y^*$ are not valid xy-polynomials. This is because the former one has a term that is missing the variables with indices 1 and 2 (i.e., x_3x_4 has indices 3 and 4 only). The latter one has a term that has more than one variable with the same index (i.e., $x_1x_2x_3x_4y_4$ contains both variables x_4 and y_4).*

Let Θ be a subset of U. Let $\mathcal{P}(\Theta)$ be the power set of Θ. Consider a set $S \in \mathcal{P}(\Theta)$. Consider a predicate R defined on $\mathcal{P}(\Theta)$. Next, we define a *set indicator function* $\chi : \mathcal{P}(\Theta) \to \{0, 1\}$, such that $\chi(S) = 1$ if S satisfies R and $\chi(S) = 0$ otherwise. We define an operator, called the *x-collapsing operator*, with respect to a set indicator function χ. This operator is applied to an xy-polynomial F. It replaces all the terms in F of the form $ax_S y_{\Theta \setminus S}$ with ax^* if $\chi(S) = 1$.

The *y-collapsing operator* is defined analogously. We will denote the set indicator function for the *y-collapsing operator* with $\omega : \mathcal{P}(\Theta) \to \{0, 1\}$. The following definition presents these concepts formally.

DEFINITION 2. *(*COLLAPSING OPERATOR*) The x-collapsing operator, applied to an xy-polynomial over Θ, $F = \sum_{i=1}^{k} a_i x_{S_i} y_{\Theta \setminus S_i} + bx^* + cy^*$, with respect to set indicator function χ is defined as*

$$\rho^x_\chi(F) = \sum_{i=1}^{k} a_i x_{S_i} y_{\Theta \setminus S_i} (1 - \chi(S_i)) + \left(b + \sum_{i=1}^{k} a_i \chi(S_i)\right) x^* + cy^*$$

We call the result of the collapsing operator the *collapsed polynomial*. Notice that the xy-polynomials are closed under the x- and y-collapsing operations.

We observe a special pair of set indicator functions (χ, ω) when $\chi(S)\omega(\Theta \setminus S) = 0$ for all $S \in \mathcal{P}(\Theta)$. The x- and y-collapsing operators using such indicators χ and ω never collapse the same term since we never have $\chi(S) = 1$ and $\omega(S) = 1$ for the same set S. We also assume that χ and ω also satisfy the property that if they evaluate to 1 for a set S, then they evaluate to 1 for any superset of S. The collapsing operators $\rho^x_\chi()$ and $\rho^y_\omega()$ satisfying this property commute with each other. In other words, $\rho^y_\omega(\rho^x_\chi(F)) = \rho^x_\chi(\rho^y_\omega(F))$. We can combine the successive application of two operators into a single one, the collapsing operator. We denote the resulting operator with the notation $\rho_{\chi,\omega}(F) = \rho^y_\omega(\rho^x_\chi(F))$.

EXAMPLE 2. *Let $\chi(S) = 1$ if $2 \in S$ and 0 otherwise. Let $\omega(S) = 1$ if $\{1, 2\} \subset S$ and 0 otherwise.*

$\rho_{\chi,\omega}(3x_1x_2x_3y_4 + 2x_1x_3y_2y_4 + 4x_3x_4y_1y_2 + 5y*) = 3x^* + 2x_1x_3y_2y_4 + 9y^*$. This is because the first term contains x_2, so it collapses to $3x^*$. Also the third term contains y_1 and y_2, so it collapses to $4y^*$.

We define the multiplication of xy-polynomials as regular polynomial multiplication with a minor difference; any variable multiplied with x^* or y^* is absorbed into x^* or y^* respectively. In other words, for any $S, T \subseteq U$, $x_S y_T x^* = x^*$ and $x_S y_T y^* = y^*$. In particular, $x^*x^* = x^*, y^*y^* = y^*$. We formally define the multiplication of two xy-polynomials below.

DEFINITION 3. *(*XY-PRODUCT*) Given $U = \{1, \ldots, n\}$, let Θ_1 and Θ_2 be two disjoint subsets of U. Let $S_1, \ldots, S_k$ be subsets of Θ_1, and $T_1, \ldots, T_r$ be subsets of Θ_2. Consider two xy-polynomials, F_1 over Θ_1 and F_2 over Θ_2, such that $F_1 = \sum_{i=1}^{k} \alpha_i x_{S_i} y_{\Theta_1 \setminus S_i} + b_1 x^* + c_1 y^*$, $F_2 = \sum_{i=1}^{r} \beta_i x_{T_i} y_{\Theta_2 \setminus T_i} + b_2 x^* + c_2 y^*$. We define the xy-product $F_1 \odot F_2$ as $F_1 \odot F_2 = \sum_{i=1}^{k} \sum_{j=1}^{r} \alpha_i \beta_j x_{S_i} x_{T_j} y_{\Theta_1 \setminus S_i} y_{\Theta_2 \setminus T_j}$*

$$+ \left(b_2 \sum_{i=1}^{k} \alpha_i + b_1 \sum_{i=1}^{r} \beta_i + b_1 b_2\right) x^*$$

$$+ \left(c_2 \sum_{i=1}^{k} \alpha_i + c_1 \sum_{i=1}^{r} \beta_i + c_1 c_2\right) y^* + (b_1 c_2 + c_1 b_2) x^* y^*$$

EXAMPLE 3. $(2x_1y_2 + 2y_1y_2 + x^*) \odot (x_4y_3 + 3x^* + y^*) = 2x_1x_4y_2y_3 + 6x^* + 2y^* + 2x_4y_1y_2y_3 + 6x^* + 2y^* + x^* + 3x^* + x * y* = 2x_1x_4y_2y_3 + 2x_4y_1y_2y_3 + 16x^* + 4y^* + x^*y^*$

The following theorem establishes a very useful result on the interplay between x-y polynomial multiplication and collapsation.

THEOREM 1. *If F_1 and F_2 are xy-polynomials and χ and ω are two set indicator functions, then $\rho_{\chi,\omega}(F_1 \odot F_2) = \rho_{\chi,\omega}(\rho_{\chi,\omega}(F_1) \odot \rho_{\chi,\omega}(F_2))$*

We see that to obtain the collapsed polynomial of an xy-product of two polynomials it is not enough to multiply the collapsed versions of the two factors; we need to apply the collapse operator again. To see this, suppose that a term x_S is collapsed in $F_1 \odot F_2$ on the right hand side of the equality in Theorem 1. Some of the indices in S come from F_1 and others from F_2. Let S_1 and S_2 denote the indices in S appearing in F_1 and F_2 respectively (i.e., $S = S_1 \cup S_2$ and $S_1 \cap S_2 = \emptyset$). We have $x_S = x_{S_1} x_{S_2}$. Suppose that $\chi(S_1) = \chi(S_2) = 0$ and $\chi(S) = 1$. Then on the right hand side neither x_{S_1} nor x_{S_2} are collapsed in F_1 and F_2, respectively, so another application of the collapsing operator is necessary on the xy-product. A rigorous proof can be found in our online technical report [11].

3.2 Probabilistic networks as xy-polynomials

So far we have defined the xy-polynomials and the key operators on them. In this section, we explain how we model probabilistic networks and the reachability problem considered in this paper using these polynomials.

Let $G = (V, E, P)$ be a probabilistic network with $E = \{e_1, \ldots, e_n\}$. For each edge $e_i \in E$, let $p_i = P(e_i)$ be the probability that e_i exists and $q_i = 1 - p_i$ be that it is absent. We start by transforming a single edge of G into a polynomial.

DEFINITION 4. *(*EDGE POLYNOMIAL*) We define the* edge polynomial *for each edge $e_i \in E$ as the first degree polynomial of two variables x_i and y_i, $F_i(x_i, y_i) = p_i x_i + q_i y_i$.*

Consider a set of edges $E' \subset E$. We define the product of the edge polynomials of the edges in E' as the *edge aggregation polynomial*. The following definition introduces this formally.

DEFINITION 5. *(*EDGE AGGREGATION*) The* edge aggregation polynomial *of the set of edges E' is the polynomial $F = \prod_{e_i \in E'} F_i(x_i, y_i)$.*

Following from Definition 5, the edge aggregation polynomial for the given set of edges E' is
$F = \sum_{\mathcal{E} \subset E'} \prod_{e_i \in \mathcal{E}} p_i x_i \prod_{e_j \in E' \setminus \mathcal{E}} q_j y_j$

EXAMPLE 4. *The edge aggregation polynomial of the set $\{e_2, e_3\}$ in Figure 1 is $F(x_2, x_3, y_2, y_3) = (p_2x_2 + q_2y_2)(p_3x_3 + q_3y_3) = p_2p_3x_2x_3 + p_2q_3x_2y_3 + p_3q_2x_3y_2 + q_2q_3y_2y_3$.*

Notice that the edge aggregation polynomial defined above is an xy-polynomial with $b = c = 0$. An edge aggregation polynomial has two important properties. First, each term contains the product of the existence probability (i.e., p_i) for a subset of edges of E' and the absence probability (i.e., q_i) for all the remaining edges in E'. Thus, each term's coefficient is the probability to observe a specific network instance where only the edges corresponding to the x_i variables exist. In other words, a term precisely models a subnetwork. For instance in Example 4, $p_2q_3x_2y_3$ models the subnetwork where e_2 is present and e_3 is absent. The coefficient p_2q_3 is the probability that e_2 is present and e_3 is absent. Second, each possible subnetwork (i.e., all subsets of edges in E') appears as exactly one term in the aggregation polynomial. For instance in Example 4, the two edges in E' yield four possible subnetworks (i.e., each edge is absent or present). The edge aggregation polynomial has four terms, each modeling one of these subnetworks.

When $E' = E$ we call the resulting polynomial the *graph polynomial*. The graph polynomial is an exhaustive representation of all the deterministic instances that could be generated by the probabilistic network model, along with their probability of realization. For instance, consider the probabilistic network in Figure 1. In its graph polynomial, the term that contains $x_3x_4x_5y_1y_2$ corresponds to the network instance that only contains edges e_3, e_4 and e_5 (bottom-left network in Figure 1). The coefficient of each term is the probability that the deterministic network corresponding to that term is realized. For example, the probability of observing the deterministic network shown on the bottom-left of Figure 1 is $p_3p_4p_5q_1q_2$. Using the possible worlds semantics [31], the probability that the network satisfies any predicate can be computed by summing the probabilities of the possible deterministic worlds in which the network satisfies the predicate. This is equal to the sum of the coefficients of the polynomial terms corresponding to the deterministic networks that satisfy the given predicate.

3.3 Modeling reachability in probabilistic networks

We are now ready to describe how we use xy-polynomials to find the probability that a given target node t can be reached from a given source node s. To achieve this, we need a predicate that can characterize the event that t can be reached from s in a deterministic network instance. We do it by considering a special deterministic instance of the network $G = (V, E, P)$ in which all edges in E exist with

certainty. We denote this network with $\overline{G} = (V, E)$. We compute two pieces of information from $\overline{G}$. The first one is the set of all possible *simple paths* from s to t. The second one is the set of all possible *minimal s-t cuts*. An s-t cut is a subset of edges of the given network such that their removal from the network disconnects s from t. We say that an s-t cut is *minimal* if none if its subsets is an s-t cut. We denote the set of paths and the set of cuts with $\Psi = \{\pi_1, \ldots, \pi_k\}$ and $\Phi = \{\kappa_1, \ldots, \kappa_l\}$ respectively. For example in Figure 1, $\{e_1, e_2\}$ is a path and $\{e_1, e_3\}$ is a minimal s-t cut. In this example, $\{e_3, e_4\}$ is not an s-t cut because t is still reachable from s after removing e_3 and e_4. Also, $\{e_1, e_3, e_4\}$ is not a *minimal* s-t cut because it has a subset (i.e., $\{e_1, e_3\}$) that is also an s-t cut.

We are now ready to present our main result, which is the mathematical foundation of the PReach algorithm. It shows that, given a probabilistic network and a start and a target node, by applying the collapsing operator, we can reduce the graph polynomial to only two terms; one of them containing only the variable x^* and the other only y^*. It uses Ψ and Φ to construct the x-collapsing and y-collapsing operators. The coefficient of x^* is the probability that the target node is reachable from the start node and the coefficient of y^* is the complement of the coefficient of x^*. We give the proof in our online technical report [11].

THEOREM 2. *Let $G = (V, E, P)$ be a probabilistic network and F be the graph polynomial for G. Let $s, t \in V$ be two nodes of G. Let Ψ be the set of all simple paths from s to t and Φ the set of all minimal s-t cuts in $\bar{G}$. For all $\pi \in \Psi$ and $\kappa \in \Phi$, let $Ind(\pi)$ and $Ind(\kappa)$ be the set of indices of the edges in π and κ respectively. Define set indicator variable $\chi(S)$=1 if $\exists \pi \in \Psi$ such that $Ind(\pi) \subseteq S$ and 0 otherwise. Similarly, define $\omega(S)$=1 if $\exists \kappa \in \Phi$ such that $Ind(\kappa) \subseteq S$ and 0 otherwise.*

If $\Psi \neq \emptyset$, then the collapsed graph polynomial with respect to χ and ω is $\rho_{\chi,\omega}(F) = \gamma x^* + (1 - \gamma) y^*$, where γ is the probability that t is reachable from s.

3.4 PReach: the algorithm

In Section 3.3, we showed that we can compute the reachability probability in a probabilistic network by collapsing its graph polynomial with an appropriate collapsing operator $\rho_{\chi,\omega}()$. The central hurdle in doing this is to compute the graph polynomial. This is because it requires multiplying all edge polynomials of the given probabilistic network. Given that each edge polynomial has two unique variables, such multiplication leads to a growth of the number of terms exponential in the number of edge polynomials. For instance, the probabilistic network in Figure 1 has five edges. Therefore, the product of the corresponding five edge polynomials yield $2^5 = 32$ terms. The complexity of computing the reachability probability is dominated by the number of terms in the graph polynomial. In this section, we describe the PReach algorithm which computes the reachability from the graph polynomial while dramatically reducing the number of its terms. PReach uses the collapsing operator carefully to reduce the size of the graph polynomial while multiplying edge polynomials. First, we present Theorem 3 which lays the foundation of the PReach algorithm. We give its proof in our online technical report [11].

THEOREM 3. *Let $G = (V, E, P)$ be a probabilistic network. Let $E_1 \subset E$ and $E_2 \subset E$ be two disjoint sets of edges of G. Let F_1 and F_2 be the edge aggregation polynomials of E_1 and E_2, respectively. Let χ and ω be two set indicator functions. The collapsed edge aggregation polynomial with respect to χ and ω corresponding to $E_1 \cup E_2$ is $\rho_{\chi,\omega}(F_1 F_2) = \rho_{\chi,\omega}(\rho_{\chi,\omega}(F_1) \odot \rho_{\chi,\omega}(F_2))$*

Following Theorem 3, our method first finds the set of all paths Ψ and all minimal $s-t$ cuts Φ between the given start and target nodes s and t. We build the indicator functions χ and ω from these paths and cuts respectively as described in Theorem 2. In other words, $\chi()$ returns 1 only if the underlying subset of edges contains at least one path from Ψ. Similarly, $\omega()$ returns 1 only if the underlying subset of edges contains at least one s-t cut from Φ. Then we compute the graph polynomial by multiplying the edge polynomials and reduce the size of the resulting polynomials using the collapsing operator one edge at a time. Notice that Theorem 3 ensures that we can avoid computing the entire uncollapsed graph polynomial by collapsing the product polynomial after each xy-multiplication as soon as a term contains a path or a cut.

EXAMPLE 5. *Let us apply the PReach algorithm to the network in Figure 1, where for simplicity we assume that all probabilities are 0.5. We will compute the probability that node c is reachable from node a. The set of all simple paths between the two nodes is $\{\{e_1, e_2\}, \{e_3, e_4\}, \{e_2, e_3, e_5\}\}$. The set of all minimal $s-t$ cuts is $\{\{e_1, e_3\}, \{e_2, e_3\}, \{e_2, e_4\}, \{e_1, e_4, e_5\}\}$. In Table 1 we present the collapsed graph polynomial after adding each edge.*

As we discussed earlier, after multiplying k edge polynomials, without collapsation the edge aggregation polynomial has 2^k terms. Notice however that the collapsed polynomial remains very small throughout this example. More specifically, we obtain the first free variable x^* in the product $F_1 F_2$ since edges e_1 and e_2 form a path from source (i.e., a) to target (i.e., c) in Figure 1. At this step both the collapsed and uncollapsed edge aggregation polynomials have four terms. However, when we multiply the result with the next edge polynomial F_3, the free variable absorbs all the terms in F_3. Furthermore, this product creates the next free variable y^* since edges e_1 and e_3 form a minimal $s-t$ cut. As a result, the collapsed edge aggregation polynomial has nearly half as many terms as the uncollapsed one.

As we multiply the next edge polynomial F_4, both x^* and y^* absorb the new terms. Also edge e_4 enables collapsing some of the existing terms in either x^* or y^* by forming new paths or cuts on top of the existing ones. As a result, the collapsed polynomial has less than a quarter terms of the uncollapsed one.

Finally, after multiplying the last edge polynomial F_5, our method collapses even more terms since including e_5 introduces new paths and cuts. We end up with only two terms while the graph polynomial has 32 terms. The coefficient of x^* (i.e., 0.46875) in this polynomial is the reachability probability we are aim to find. The coefficient of y^* is simply 1 minus the coefficient of x^*.

Further improvements in the algorithm. Collapsing the terms of the edge aggregation polynomial reduces the computational cost of the PReach algorithm greatly. Here, we make two optimizations that will cut down this cost further by preprocessing the given network quickly. The first one will shrink the size of the input network by filtering and

Table 1: Computation of reachability probability from node a to node c in the probabilistic network shown in Figure 1. Column denoted by EP shows the number of terms in the edge aggregation polynomial, and the column denoted by CP shows the number of terms in the corresponding collapsed polynomial.

Edge aggregation polynomial	Collapsed polynomial	#Terms	
		CP	EP
F_1	$0.5x_1 + 0.5y_1$	2	2
F_1F_2	$0.5^2(x_1y_2 + x_2y_1 + y_1y_2 + x^*)$	4	4
$F_1F_2F_3$	$0.5^3(x_1x_3y_2 + x_2x_3y_1 + x_3y_1y_2) + 0.25x^* + 0.375y^*$	5	8
$F_1F_2F_3F_4$	$0.5^4x_2x_3y_1y_4 + 0.4375x^* + 0.5y^*$	3	16
$F_1F_2F_3F_4F_5$	$0.46875x^* + 0.53125y^*$	2	32

merging edges whenever possible. The second one will order the edge polynomials that go into the polynomial multiplication to maintain a small number of polynomial terms.

CONTRACT THE NETWORK. PReach requires a polynomial multiplication for each edge in the network. Thus, fewer edges implies fewer polynomial multiplications. We reduce the size of the input network in the preprocessing stage in three steps, illustrated in Figure 2. The first step is the removal of nodes that are *irrelevant* to the problem, along with their edges. A node is irrelevant if (i) it is not a source node and it has no incoming edges (i.e., is unreachable from a source node), or (ii) it is not a target node and it has no outgoing edges (i.e., no target node can be reached from it). In Figure 2(a), this step removes nodes a and c and obtains the network in Figure 2(b). The second step replaces a path that consists of a sequence of nodes that do not have any other incoming or outgoing edges by a single edge. The probability of the new edge is the product of the probabilities of the edges on the path it replaces. In Figure 2(b), this step replaces the edges on the path $s \to b \to t$ with a single edge $s \to t$ and obtains the network in Figure 2(c). After the second step, it is possible to have multiple edges with the same direction between pairs of nodes. For instance, in Figure 2(c), this happens after merging the edges (a, b) and (b, c) into (a, c). The third step replaces multiple edges with the same start and end nodes by a single edge. The probability of the new edge is the probability that at least one of the initial edges is present. More specifically, if the initial edges have probabilities $p_1, \ldots, p_k$, the probability of the new edge is $1 - (1 - p_1) \ldots (1 - p_k)$. In Figure 2(c), the step merges the two edges from s to t and obtains the network in Figure 2(d). We repeat these three steps until no further reduction is possible.

ORDER THE EDGES. The collapsing operator shrinks the polynomial. Notice that for collapsing operator to work, the set indicator functions should evaluate to one. Our second improvement follows from these observations. By carefully deciding the order in which the edge polynomials get multiplied, we can force the set indicator function to evaluate to one early in the polynomial multiplication. We have the following heuristic for choosing the edge ordering. Consider each path or cut as the set of edges they contain. We order all such sets by their sizes in increasing order. We first multiply the edge polynomials in the smallest set. When we multiply an edge polynomial, we remove that edge from all the remaining sets. We then repeatedly pick the next smallest set until we multiply all edge polynomials. This greedy approach ensures that we collapse the graph polynomial as soon as possible.

Time complexity of PReach. Let us denote the number of edges in the probabilistic network before and after contracting the network with n and n' (with $n' \leq n$). A trivial upper bound for the time complexity of PReach is $O(2^{n'})$ as it requires multiplying n' edge polynomials. The actual time complexity is however significantly less than this as PReach collapses polynomials, and it depends on the network topology.

Let E_k be the set of first k edges whose polynomials are multiplied. Let Ψ_k and Φ_k be the set of paths and cuts in E_k (i.e., $\Psi_k = \{\pi_i | \pi_i \in \Psi, \pi_i \subseteq E_k\}$ and $\Phi_k = \{\kappa_i | \kappa_i \in \Psi, \kappa_i \subseteq E_k\}$). The exact time complexity of aggregating the polynomials for the edges in E_k is $2^k + \sum_{C \subseteq \Psi_k} (-1)^{|C|} 2^{k - |\cup_{\pi_i \in C} \pi_i|} + \sum_{C \subseteq \Phi_k} (-1)^{|C|} 2^{k - |\cup_{\kappa_i \in C} \kappa_i|}$. The formula follows by deriving the number of terms collapsed by each subset of paths and cuts. Let d be the maximum out-degree of a node in the contracted network. Let σ_Ψ and σ_Φ be the maximum size of a path and a cut in the contracted network respectively. The complexity of finding all paths is $O(d^{\sigma_\Psi})$. The complexity of finding all minimal cuts is $O(d^{\sigma_\Phi})$.

As we demonstrate in the next section, PReach's running time is dominated by polynomial multiplication and collapsing. We also show that PReach is many orders of magnitude faster than the existing precise methods and the gap between them increases with increasing network size.

4. RESULTS

This section evaluates the performance of PReach. We experimented with signaling networks taken from KEGG [18]. We have used all the signaling networks of *Homo sapiens* (*H. sapiens*) from this database. We report results only for networks with more than 13 edges as the problem is trivial for small networks. In order to experiment with networks of growing sizes we also combined subsets of these signaling networks. We have obtained the source and targets of each signaling network based on the hierarchical organization of the genes [13]. We set the genes at the top of the hierarchy as the source nodes and the ones at the bottom as the target. You can refer to our online technical report [11] for the details of a listing of these networks and and their sizes. We have extracted the confidence scores for each interaction from STRING [35] and used them as edge probabilities. STRING has confidence values in the [1,1000] interval. We divided this number by 1000 to obtain a number in the [0,1] interval for each interaction.

We compared PReach to the inclusion-exclusion algorithm of Ourfali *et al.* [27]. Although this method does more than solving the reachability problem, we compare against the inclusion-exclusion approach introduced in it as the most recent exact solution available for the problem. The inclusion-exclusion algorithm did not scale to networks with more than 40 paths. We therefore improved its running time dramatically by implementing our network contraction method (see Section 3.4) as a preprocessing step to it. In all our experiments, we compare against the improved implementation of Ourfali *et al.* [27]

We implemented both PReach and the inclusion-exclusion method in C++. We ran our experiments on a multiprocessor computer with 48 cores and 256 GB RAM and measured the running times of both methods.

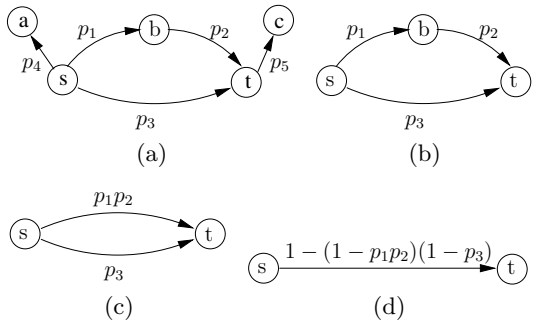


Figure 2: **Contracting the network as a preprocessing step. 2(a): initial network. 2(b): Eliminating irrelevant nodes. 2(c): replacing paths without deviations. 2(d): replacing parallel edges.**

Table 2: **Network statistics and running time comparison between our method (PReach) and the inclusion-exclusion method (IE). $|V|$, $|E|$ and $|\Psi|$ denote number of nodes, edges and paths respectively. Time is in seconds unless otherwise specified. The values marked with $\sim$ are estimates we obtained from the number of paths after running IE for over one day. In parenthesis, we show the time to compute all cuts for PReach, which is included already in the total running time of PReach. In the second row, All − Wnt indicates all networks but Wnt. In other words it is the union of the Hedgehog, Jak-STAT and mTOR networks.**

				Execution time (s)	
Network	$\|V\|$	$\|E\|$	$\|\Psi\|$	IE	PReach
Wnt	58	91	66	0.001	0.03 (0.01)
All − Wnt	64	100	136	17238.43	1.75 (0.02)
Wnt + Hedgehog	70	114	111	583.97	17.69 (1.3)
Wnt + mTOR	84	129	89	0.29	57.31 (3.54)
Wnt + TGF-beta	87	139	220	$\sim$102 yrs	1151.23 (39.13)
Wnt + Jak-STAT	77	127	81	$\sim$146 hrs	1805.45 (48.81)

4.1 Running time evaluation

Both PReach and inclusion-exclusion return the same result when they run till completion as they both compute the reachability probability precisely. We first evaluate how long it takes for them to find the result.

In this experiment, we compute the probability that at least one target is reachable from a source. To do that, we created a hypothetical source node s that has an outgoing edge to all the actual source nodes. Similarly, we created an artificial target node t with incoming edges from all actual target nodes. We set the probabilities of all artificially created edges to 1. We then computed the reachability probability from s to t. We considered networks of *H. Sapiens* obtained by combining different KEGG pathways. Table 2 presents the results. The time to preprocess and contract the network as well as the time for finding all paths was negligible (milliseconds), so we do not report them. Results demonstrate that PReach is orders of magnitude faster than the competing method. Only on the two networks with the least number of paths is the inclusion-exclusion method faster than PReach. The inclusion-exclusion method is exponential in the number of paths from source to destination. As a result, this method quickly becomes infeasible as the number of paths grows. In [27], Ourfali et al. were able to report results on the entire *S. cerevisiae* network by limiting the length of their paths to three edges, and thus limiting

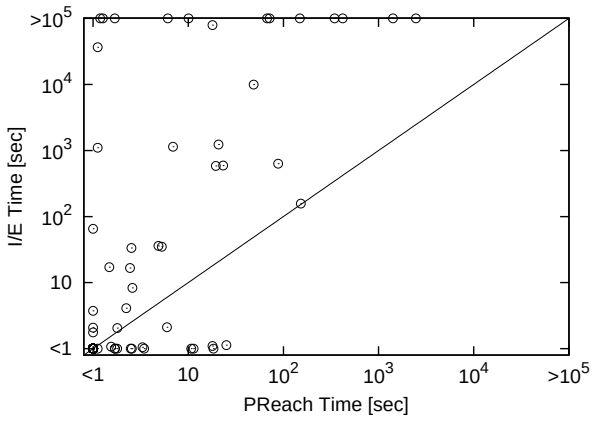
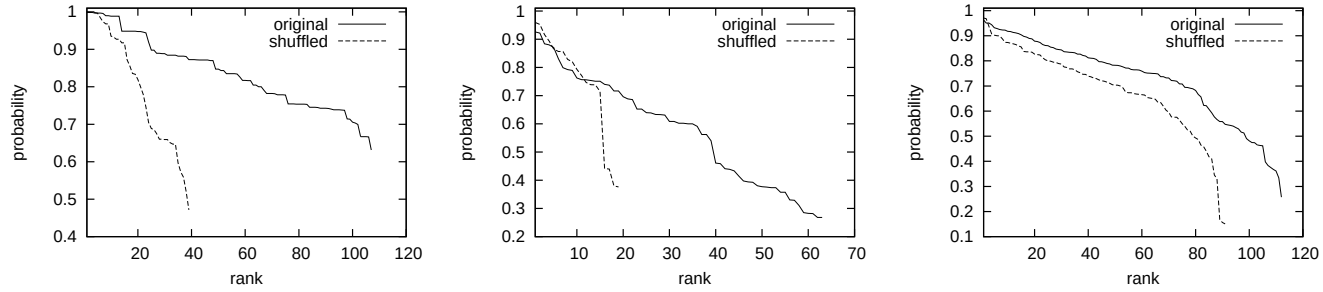


Figure 3: **Predictability of the running time for PReach and the inclusion-exclusion method. Each circle represents the running times for one network. The network sizes vary between 58 and 62. x axis: running times for PReach (s). y-axis: running time for the inclusion-exclusion method (s). Scales on both axes are logarithmic. The circles at 10^5 s running time represent all running times greater than 10^5 s.**

the number of paths. This however leads to incorrect results if there are paths longer than three edges. When we ran this method after limiting the path length to three, in five out of six experiments, it found incorrect results. The percentage of error was enormous; it varied between 6 to 100%. On the other hand, paths and cuts help PReach collapse the polynomial terms and cut down the running time dramatically. Thus, a fundamental difference between PReach and the inclusion-exclusion method is that growing number of paths rapidly hurts the performance of the latter, while it actually improves that of the former. Moreover, our method does not impose an arbitrary maximum path length, which may be difficult to determine and lead to incorrect conclusions, since the magnitude of the error cannot be properly estimated.

We observe this in our experiments; as the number of edges grows, we expect the running time of PReach to grow. However, the paths and cuts help us collapse the polynomial terms and cut down the running time dramatically.

The running times of both methods can grow rapidly as they have time complexities exponential in different terms. So their cost can grow highly when the network grows slightly. To observe how severe this problem is, we assessed running times of PReach relative to the inclusion-exclusion method by growing the network size in a controlled fashion. To do this, we slightly grew a network by adding one node at a time in a way that preserves global properties, such as density and power law distribution of node degrees. To add a new node to the network, we choose an existing node uniformly at random and assign the in-degree of the existing node to the new node. To connect the new node to the network, we randomly select nodes from the current network. The probability that a node will be selected is proportional to the out-degree of that node. We add an incoming edge from each chosen node to the new one. We add the outgoing edges from the new node similarly. This way the power law characteristic of the degree distribution in the network is preserved. Following this procedure, starting with the Wnt



(a) ErbB (b) Wnt (c) MAPK

Figure 4: Reachability probabilities in KEGG signaling networks and randomly generated networks. Left: ErbB; middle: Wnt; right: MAPK.

network of *H. sapiens*, which has 58 nodes, we generated 200 networks. 50 networks are generated by adding one node to the initial network, other 50 are generated from one of the networks obtained in the previous step by adding one more node and so on. We ran both PReach and the inclusion-exclusion method on each network. We report the running times in Figure 3. We observe that, because of its exponentiality, the running times for the inclusion-exclusion method vary wildly, spanning several orders of magnitude increase, even if the network sizes differ by four nodes at most, between 58 and 62. By contrast, PReach finishes in 100 seconds in most of the cases with much smaller variation in running time. In a small number of instances, the inclusion-exclusion method is faster than PReach. Typically, this happens when there are only a small number of paths from the source to target nodes. Nevertheless, PReach has a small running time even for those datasets.

4.2 Biological significance of the results

In order to assess the biological significance of our results, we have compared the results obtained on the KEGG signaling networks with results obtained on randomly generated networks. To generate these random networks, we start with an initial KEGG network and we shuffle the edges in a manner that preserves the node count and the degree of each node. At each step of the shuffling process, we select two edges uniformly at random and switch the endpoints between the two edges, i. e., edges (a, b) and (c, d) become (a, d) and (c, b). We repeat this for $100,000$ times. For each KEGG network, each source node and each target node, we computed the probability that the target node is reachable from the source node. We ranked all source-target pairs by their corresponding reachability probabilities and plotted the resulting graphs in Figure 4. We omitted from the plots the pairs with zero probability. The plots clearly show higher reachability probability for the KEGG networks compared to the randomly generated networks, especially for the ErbB and Wnt pathways. For example, the pair ranked 40th in the ErbB network has a reachability probability of close to 0.9, while the pair with the same rank in the corresponding randomly generated network is disconnected. The gap between the plots for the real and shuffled networks demonstrate that signaling networks are heavily biased towards enabling pathways from receptors to reporters to maximize the chances that signals can travel to reporters. Interestingly, the gap between the probability values in the MAPK network and its random counterpart is less wide. This observation complies with the fact that the average in-degree of MAPK reporters is 1.68, while it is much larger for ErbB and Wnt. This suggests that MAPK signaling network is more robust and less sensitive to perturbations as long as the node degrees are preserved.

Table 3: Top sources and targets according to PReach and the deterministic alternative in ErbB, MAPK and Wnt. For PReach: most common sources and targets having reachability probability greater than 0.9. For Deterministic: sources connected to most targets and targets connected to most sources.

		Top sources and targets	
		PReach	Deterministic
ErbB	Sources	EREG, HBEGF EGF, TGFA	HBEGF, BTC, TGFA, EGF, EREG
	Targets	STAT5, FAK, ELK1	All tied
MAPK	Sources	TNF, ILA1, GLK	TNF, IL1A
	Targets	ELK1, p53, CREB	NLK, ATF4
Wnt	Sources	TAK1, PKA, PS-1	WNT5A, DKK1
	Targets	NLK, c-Myc	ROCK1, PRKCA

Table 4: Common GO terms for targets having reachability probability greater than 0.9 in ErbB, MAPK and Wnt.

Network	GO ID	Term name
ErbB	GO:0000209	protein polyubiquitination
	GO:0007087	mitotic nuclear pore complex reassembly
MAPK	GO:0000209	protein polyubiquitination
	GO:0003399	cytoneme morphogenesis
	GO:0006268	DNA unwinding involved in replication
	GO:0031265	CD95 death-inducing signaling complex
	GO:0042598	vesicular fraction
Wnt	GO:0000209	protein polyubiquitination
	GO:0042598	vesicular fraction

We have further investigated the results of this experiment by examining the most common source and target proteins in the pairs that achieved reachability probability greater than 0.9. We also devised an intuitive deterministic alternative for measuring reachability. Using the underlying deterministic version of each network, we count the number of targets that a given source can reach to, as well as the number of sources that can reach to a given target. We also list the top sources and targets according to this metric. Table 3 lists such proteins for each examined network. Among the target proteins with more than 0.9 reachability probability in ErbB is *STAT5*. It plays an important role in the development and function of B- and T-lymphocytes, which are the major cel-

lular components of the adaptive immune response [15]. One of the highest reachable target proteins in MAPK is *CREB*, which has a well-known role in cell proliferation, differentiation and survival, as well as specific functions in immune responses [38]. In Wnt, *NLK* is found among the highest reachable target proteins, which positively regulates Wnt/β-catenin signaling by phosphorylating LEF1 in neural progenitor cells [26]. The top ErbB sources found by PReach were also found by the deterministic method. However, the deterministic method produces a tie between all ErbB targets, providing no ranking information. In cases of MAPK and Wnt, PReach produces top sources and targets that are different than the ones produced by the deterministic method. This provides new insights about proteins of potentially important roles, which cannot be found by the deterministic method.

We also extracted the GO annotations that are common to the target proteins that achieve more than 0.9 reachability probability. Table 4 lists such annotations for each examined network. We considered all possible GO terms that exist in at least one protein among all the target proteins. We then eliminated nonspecific GO terms (i,e., GO-terms that are too high in the GO-hierarchy) in order to avoid having a positive bias towards our method. Such elimination strategy is used in the literature for the same purpose (see [33, 36]).

For each GO term, say t_i in the list of terms described above we computed the probability to reach to that GO term as follows. Let X_i denote the set of target proteins that are annotated with t_i. We created a hypothetical target node h_i. We also included a hypothetical edge from each node in X_i to h_i. The probability of all those edges are set to 1.0. Thus reaching to h_i means reaching to at least one protein with annotation t_i. In short, we compute the reachability probability to each t_i without error as the reachability probability to h_i. The GO terms listed in this table are those with the highest reachability probability. One of the common annotations of MAPK highest-reachable target nodes is *cytoneme morphogenesis* (GO:0003399). This signifies a role in shaping the anatomical structure of cytonemes, which connect adjacent cells to enable transfer of surface-associated cargoes from cell to cell[32]. Among the common annotations for ErbB highest-reachable target nodes is *mitotic nuclear pore complex reassembly* (GO:0007087). In the context of mitotic cell division, this signifies a role in reforming nuclear pores which facilitate the movement of macromolecules between the nucleus and cytoplasm [4]. For the top targets found by the deterministic method, we could not extract the top GO annotations for ErbB targets because all the targets are tied in ranking. For MAPK and Wnt, PReach outputs GO terms that are different than the ones outputted by the deterministic method. This points out different areas of investigation about more important functions of each network.

5. CONCLUSION AND FUTURE WORK

We have presented a fast and exact method for computing the reachability probability in probabilistic networks, where each edge represents an actual interaction with a different probability. Our method, PReach, relies on a theory we developed around a special class of polynomials, the xy-polynomials, used as a representation for probabilistic networks. The biological significance of PReach is validated by experimentation. Other methods that have considered this problem in the context of signaling networks, based on the inclusion-exclusion principle, are far behind our method in terms of computational cost.

PReach has numerous potential biological applications on multiple levels. On the node level, a new notion of node centrality in probabilistic networks can be devised by measuring the difference in reachability probability if the node is removed. On the network level, network robustness can be assessed based on difference in reachability probability after changing interaction probabilities and/or network topology. On the level of collective functions, reachability probability of a biological function can be measured as that of the targets annotated by corresponding functional annotation. As a result, we can tell which functions are more important (i.e., more supported) than others in a given network.

Further improvements to the algorithm are possible, suitable especially for large input networks. In such cases, a divide and conquer strategy can be adopted, similar to the method used in [2]. We can partition the initial network in two loosely connected clusters and solve the problem for each cluster. The looser the components are connected between themselves, the easier it is to merge the solutions. This can yield important benefits in terms of running time. We plan to explore this further improvement in future work.

Acknowledgements

This work was supported partially by NSF under grant IIS-0845439

6. REFERENCES

[1] K. K. Aggarwal, K. B. Misra, and J. S. Gupta. Reliability evaluation A comparative study of different techniques. *Microelectronics Reliability*, 1975.

[2] F. Ay, G. Gulsoy, and T. Kahveci. Finding steady states of large scale regulatory networks through partitioning. *GENSIPS*, 2005.

[3] J. S. Bader, A. Chaudhuri, et al. Gaining confidence in high-throughput protein interaction networks. *Nat. Biotechnol.*, 2004.

[4] K. K. Bodoor et al. Function and assembly of nuclear pore complex proteins. *Biochemistry and cell biology*, 1999.

[5] D. B. Brown. A computerized algorithm for determining the reliability of redundant configurations. *IEEE Trans. on Reliability*, 1971.

[6] J. I. Brown and C. J. Colbourn. Non-stanley bounds for network reliability. *Journal of Algebraic Combinatorics*, 1996.

[7] A. Ceol, A. Chatr Aryamontri, et al. MINT, the Molecular INTeraction database: 2009 update. *Nucleic Acids Res.*, 2010.

[8] M. C. Easton and C. K. Wong. Sequential destruction method for monte carlo evaluation of system reliability. *IEEE Transactions on Reliability*, 1980.

[9] S. Fields and O. Song. A novel genetic system to detect protein-protein interactions. *Nature*, 1989.

[10] G. S. Fishman. A monte carlo sampling plan for estimating network reliability. *Operations Research*, 1986.

[11] H. Gabr, A. Todor, H. Zandi, A. Dobra, and T. Kahveci. Preach: Reachability in probabilistic signaling networks. Technical Report REP-2013-564,

CISE, University of Florida, Gainesville, Florida, April 2013.

[12] A. C. Gavin, M. Bosche, et al. Functional organization of the yeast proteome by the systematic analysis of protein complexes. *Nature*, 2002.

[13] G. Gulsoy, N. Bandhyopadhyay, and T. Kahveci. HIDEN: Hierarchical decomposition of regulatory networks. *BMC Bioinformatics*, 2012.

[14] C. Hanzhong. A new algorithm for computing the reliability of complex networks by the cut method. *Microelectronics Reliability*, 1994.

[15] L. M. Heltemes-Harris, M. J. L. Willette, et al. The role of stat5 in the development, function, and transformation of b and t lymphocytes. *Annals of the New York Academy of Sciences*, 2011.

[16] T. Husfeldt and N. Taslaman. The exponential time complexity of computing the probability that a graph is connected. *International Symposium on Parameterized and Exact Computation*, 2010.

[17] G. L. Hwang, T. A. Tillman, and M. H. Lee. System-Reliability Evaluation Techniques for Complex/Large Systems-A Review. *IEEE Trans. on Reliability*, 1981.

[18] M. Kanehisa, S. Goto, et al. The KEGG resource for deciphering the genome. *Nucleic Acids Res.*, 2004.

[19] R. M. Karp and M. G. Luby. A new monte carlo method for estimating the failure probability of an n-component system. *EECS TR UCB/CSD-83-117, University of California, Berkley*, 1983.

[20] E. Kim and E. Choi. Pathological roles of MAPK signaling pathwaysin human diseases. *Biochimica et Biophysica Acta (BBA)-Molecular Basis*, 2010.

[21] H. Kumamoto, K. Tanaka, and K. Inoue. Efficient evaluation of system reliability by Monte Carlo method. *IEEE Trans. on Reliability*, 1977.

[22] M. Laplante and D. M. Sabatini. mTOR signaling in growth control and disease. *Cell*, 2012.

[23] C. Lucet and J.-F. Manouvrier. Exact methods to compute network reliability. *Mathematical Methods in Reliability*, 1997.

[24] B. MacDonald, K. Tamai, and X. He. Wnt/β-Catenin Signaling: Components, Mechanisms, and Diseases. *Developmental Cell*, 2009.

[25] S. Mizuno, R. Iijima, et al. Alzpathway: a comprehensive map of signaling pathways of alzheimer's disease. *BMC Systems Biology*, 2012.

[26] S. Ota, S. Ishitani, et al. Nlk positively regulates wnt/β-catenin signalling by phosphorylating lef1 in neural progenitor cells. *EMBO J*, 2012.

[27] O. Ourfali, S. T., et al. SPINE: a framework for signaling-regulatory pathway inference from cause-effect experiments. *Bioinformatics*, 2007.

[28] J. Oxley and D. Welsh. Chromatic, flow, and reliability polynomials: the complexity of their coefficients. *Combinatorics, Probability and Computing*, 2012.

[29] J. S. Provan and M. O. Ball. The Complexity of Counting Cuts and of Computing the Probability that a Graph is Connected. *SIAM Journal of Computing*, 1983.

[30] J. S. Provan and M. O. Ball. Computing network reliability in time polynomial in the number of cuts. *Operations Research*, 1984.

[31] C. Ré and D. Suciu. Efficient evaluation of having queries on a probabilistic database. *Proc. Database Programming Languages*, 2007.

[32] N. M. Sherer and W. Mothes. Cytonemes and tunneling nanotubules in cellâĂŞcell communication and viral pathogenesis. *Trends in Cell Biology*, 2008.

[33] R. Singh, J. Xu, and B. Berger. Global alignment of multiple protein interaction networks with appication to functional orthology detection. *PNAS*, 2008.

[34] A. D. Sokal. The multivariate tutte polynomial (alias potts model) for graphs and matroids. *Surveys in Combinatorics*, 2010.

[35] D. Szklarczyk, A. Franceschini, et al. The STRING database in 2011: functional interaction networks of proteins, globally integrated and scored. *Nucleic Acids Res.*, 2011.

[36] A. Todor, A. Dobra, and T. Kahveci. Probabilistic biological network alignment. *TCBB*, 2013.

[37] C. von Mering, R. Krause, et al. Comparative assessment of large-scale data sets of protein-protein interactions. *Nature*, 2002.

[38] A. Y. Wen, K. M. Sakamoto, and L. S. Miller. The role of the transcription factor creb in immune function. *The Journal of Immunology*, 2010.

[39] R. Wilkow. Analysis and design of reliable computer networks. *IEEE Transactions on Communications*, 1975.

[40] W. Xiao, M. N. Mindrinos, et al. A genomic storm in critically injured humans. *The Journal of experimental medicine*, 2011.

[41] K. Zhu, W. Zhang, et al. Bmc: An efficient method to evaluate the probabilistic reach ability queries. *Database Systems for Advanced Applications*, 2011.