Bioinformatics, 35, 2019, i624–i633
doi: 10.1093/bioinformatics/btz360
ISMB/ECCB 2019

Reconstructing signaling pathways using regular language constrained paths

Mitchell J. Wagner, Aditya Pratapa and T. M. Murali*

Department of Computer Science, Virginia Tech, Blacksburg, VA 24061, USA

*To whom correspondence should be addressed.

Abstract

Motivation: High-quality curation of the proteins and interactions in signaling pathways is slow and painstaking. As a result, many experimentally detected interactions are not annotated to any pathways. A natural question that arises is whether or not it is possible to automatically leverage existing pathway annotations to identify new interactions for inclusion in a given pathway.

Results: We present RegLinker, an algorithm that achieves this purpose by computing multiple short paths from pathway receptors to transcription factors within a background interaction network. The key idea underlying RegLinker is the use of regular language constraints to control the number of non-pathway interactions that are present in the computed paths. We systematically evaluate RegLinker and five alternative approaches against a comprehensive set of 15 signaling pathways and demonstrate that RegLinker recovers withheld pathway proteins and interactions with the best precision and recall. We used RegLinker to propose new extensions to the pathways. We discuss the literature that supports the inclusion of these proteins in the pathways. These results show the broad potential of automated analysis to attenuate difficulties of traditional manual inquiry.

Availability and implementation: https://github.com/Murali-group/RegLinker.

Contact: murali@cs.vt.edu

Supplementary information: Supplementary data are available at *Bioinformatics* online.

1 Introduction

Signaling pathways are widely studied in systems biology. While several databases record the proteins and interactions within a diverse set of signaling pathways (Fabregat *et al.*, 2016; Kanehisa *et al.*, 2016), high-quality curation is slow and painstaking. Moreover, the vast majority of interactions in large-scale physical and signaling interactomes are not annotated with information on the signaling pathways to which they belong. This dichotomy raises the following interesting question: given the proteins and interactions in a specific curated signaling pathway, can we automatically determine which interactions in the interactome are likely to be members of that pathway?

The typical formulation of pathway reconstruction in the literature is as follows (Gitter *et al.*, 2011; Mohammadi *et al.*, 2013; Navlakha *et al.*, 2012; Ritz *et al.*, 2016; Steffen *et al.*, 2002; Supper *et al.*, 2009; Tuncbag *et al.*, 2013; Yeger-Lotem *et al.*, 2009; Yosef *et al.*, 2011): given the receptors (or sources) and transcription factors (TFs or targets) in a specific signaling pathway P and a directed, weighted interaction network G of signaling and physical interactions among proteins, compute a subnetwork of G that connects the sources to the targets and has high overlap with P. In this model, the interactions in the pathway P are also present in G (i.e. P is a subnetwork of G) but these interactions are not labeled as being members of P. Clearly, this formulation does not leverage the curated information on interactions that is present in pathway databases.

In this work, we seek to address what we call the *curation-aware* signaling pathway reconstruction problem. Here, we take as input the network G, the set S of sources, the set T of targets, and, in addition, the signaling pathway P also represented as a directed graph (Fig. 1a). This graph represents the current state of knowledge of the proteins and interactions in the signaling pathway. Informally, our goal is to compute a subgraph P' of the network G such that $P \subseteq P'$, with the proteins and interactions in $P' - P$ serving as candidates for inclusion in the pathway. We would like to rank the elements of $P' - P$ so that we may quantitatively evaluate the accuracy of P' using approaches akin to cross-validation.

We use the following intuition to guide our algorithm development. First, we model the process of signal transduction as a path in

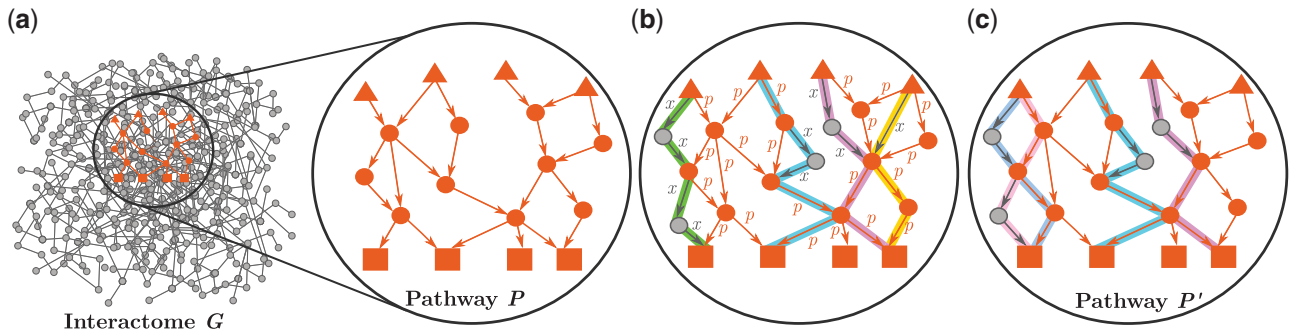

Fig. 1. Overview of RegLinker. (**a**) The curated pathway P is a subgraph of the interactome G. (**b**) Each edge in P has the label p and each edge in G–P has the label x. The figure displays P and a subset of edges in G–P. Paths start at receptors (triangles) and end at TFs (rectangles). See the text for a discussion of the four colored paths. (**c**) The output pathway P' is the union of P and a subset of edges from G–P. Each such edge belongs to a receptor–TF path that satisfies the regular expression p^*xxp^*. There are four such paths

G that starts at a receptor and ends at a TF, as is common in the literature (Gitter *et al.*, 2011; Navlakha *et al.*, 2012; Ritz *et al.*, 2016; Silverbush and Sharan, 2014; Steffen *et al.*, 2002; Yosef *et al.*, 2011) (Fig. 1). Second, we prefer high-scoring paths (when edge weights mean confidence or reliability) in analogy with cellular mechanisms to transmit signals efficiently. Finally, and most importantly, when P is partially known, we can partition a receptor-to-TF path in G into subpaths with the following structure: each subpath contains only interactions in P or only interactions not in P and these two types alternate. The challenge is that we do not know *a priori* how many edges each of these subpaths contains, especially when we also want to compute high-scoring paths.

In this article, we present the RegLinker algorithm for curation-aware reconstruction of signaling pathways (Sections 3.1–3.3). It labels every edge in G as 'positive' if it is in P or as 'unknown' otherwise (labels p and x in Fig. 1b). It then computes, for every edge in G, the shortest path through that edge that satisfies a provided regular expression, i.e. the edge labels along the path form a string in the corresponding regular language. The output is the union of these paths as P' (Fig. 1c). Our primary technical contribution is an algorithm to compute all these paths (i.e. one through each edge and constrained by the regular expression) in the same asymptotic time as taken to compute one such path.

Our key insight is that a regular language acts as a constraint that allows us to control the number of and structure by which new edges and nodes are considered for addition to P. For example, the blue path in Figure 1b satisfies the regular expression p^*xxp^*, i.e. the path starts with zero or more edges with label p, followed by two consecutive edges with label x, and ends in zero or more edges with label p. The pink path also satisfies this regular expression, since the labels along this path form the string $xxpp$. To see that this regular expression constrains the allowed paths, note that the green path on the left of Figure 1b does not satisfy p^*xxp^* since it contains four edges with label x. The yellow path on the right also violates this expression since its edge labels form the string xpp. In the output network P' (Fig. 1c), there are two additional paths (light blue and light pink) that satisfy p^*xxp^*.

Informally, a regular language is a set of strings with the property that an algorithm that uses a fixed amount of memory can examine the letters in the string sequentially to determine if it is a member of the language. For example, for the alphabet $\{p, x\}$, regular languages are expressive enough to represent strings with an even number of occurrences of p or strings where p and x strictly alternate an arbitrary number of times.

To further facilitate pathway reconstruction, we present a novel technique for weighting the interactome, which we call random walk with edge restarts (RWER, Section 3.4). Weighting edges according to our confidence in the curation or experimental method (Ritz *et al.*, 2016; Yeger-Lotem *et al.*, 2009; Yosef *et al.*, 2011) is not specific to any given pathway. RWER addresses this issue with a random walk whose transitions favor the interactions in a specific curated pathway that we want to reconstruct.

We evaluate RegLinker and five other algorithms (Section 3.4) in their ability to reconstruct signaling pathways in the NetPath database. We develop three different models of withholding information (Section 4.2), inspired by the process by which knowledge about a pathway accumulates in the literature. In our first model, we derive a subnetwork of a curated pathway by withholding a random subset of pathway interactions. This approach tests the ability of algorithms to discover the interactions in a pathway where all or most of its proteins are known. Our second model makes the more realistic assumption that we do not know all the proteins in a pathway. Therefore, we derive a subnetwork of the pathway that does not contain a random selection of proteins or any of the pathway interactions involving them. The third model combines the first two.

In Section 4, we demonstrate that RegLinker, in combination with RWER, offers a flexible and superior approach for pathway reconstruction. In particular, this approach achieves the highest or close to highest AUPRC values in all models, ranging from 0.69 for the interaction withholding model and 0.35 for protein recovery and 0.36 for interaction recovery in the combined model. We used RegLinker to propose new extensions to the pathways. Following enrichment analysis, we discuss the literature that supports the inclusion of these suggested proteins in the brain-derived neurotrophic factor (BDNF), Wnt and TNFα pathways.

2 Related research

Several algorithms exist to compute a compact subnetwork that connects a set of sources with a set of targets in an interaction network (Gitter *et al.*, 2011; Mohammadi *et al.*, 2013; Ritz *et al.*, 2016; Silverbush and Sharan, 2014; Steffen *et al.*, 2002; Supper *et al.*, 2009; Tuncbag *et al.*, 2013; Yeger-Lotem *et al.*, 2009; Yosef *et al.*, 2011). An example is the PathLinker algorithm (Ritz *et al.*, 2016) from which RegLinker takes inspiration. Given a user-defined parameter k, PathLinker computes the k shortest paths in the interactome that connect any receptor in P to any TF in P. PathLinker

achieves efficiency by integrating Yen's algorithm (Yen, 1971) with the A* heuristic. Several of these algorithms were not originally developed to explicitly solve the problem of pathway reconstruction. They share a common characteristic: like PathLinker they do not require or exploit any knowledge of the intermediate proteins or interactions in P. Hence, they cannot be directly applied to curation-aware pathway reconstruction.

Navlakha *et al.* (2012) developed a method to solve a problem similar to ours. They computed the shortest paths between all pairs of receptors and TFs in P. They searched for disconnected pairs of nodes in P such that connecting the nodes by an edge yielded a shorter path between at least one receptor and one TF. They added to P the edge that yielded the greatest decrease in the sum of shortest paths costs between all pairs of receptors and TFs, and repeated the process until no such pair of nodes could be found. Their algorithm did not add new proteins to P.

Note that other papers have considered related problems, e.g. determining the orientations of interactions (Gitter *et al.*, 2011; Silverbush and Sharan, 2014) or determining which proteins should be annotated to a specific pathway or a biological process (Jiang *et al.*, 2017). These algorithms cannot be directly applied to the problem we consider.

3 Algorithms

We first introduce the notion of paths in a directed graph that satisfy a regular language and describe an existing algorithm that computes shortest paths under this constraint (Section 3.1). Next, we present an approach that adds another criterion: compute the shortest regular language constrained path through every edge in the graph. We then describe the RegLinker algorithm (Section 3.3) as well as other algorithms that we compare to RegLinker (Section 3.4).

3.1 Regular language constrained paths

Suppose we are given a directed graph $G = (V, E, c, l)$, where V is the set of nodes, E is the set of directed edges, $c : E \to \mathbb{R}^+$ is a function that maps every edge in E to a positive, real-valued weight and $l : E \to \Sigma$ is a function that maps every edge to a label in an alphabet Σ. Given a regular language $L \subseteq \Sigma^*$, we say that a path π in G *satisfies* L if the concatenation of the labels of the edges in π forms a string that is a member of L.

Given a source node $s \in V$ and a target node $t \in V$, Barrett *et al.* (2000) provide an algorithm to compute a shortest s–t path in G that satisfies L. Briefly, they (a) construct a new graph $H = G \times M$ that is an appropriately defined cross-product of G and the deterministic finite automaton (DFA) M that recognizes L and (b) compute a shortest path in H using Dijkstra's algorithm (Fig. 2). Intuitively, as we traverse an s–t path π in G, we can use the labels of the edges in π to traverse a path π_M in M. If π_M begins at the start state and terminates at a final state in M, then π satisfies L. The cross-product graph H permits us to record both traversals simultaneously. In Figure 2c, the dark edges form two source to target paths in H, each corresponding to one of the s–t paths in Figure 2b that has a label sequence recognized by the DFA in Figure 2a. The running time of their algorithm is $O(|H| \log |H|)$.

3.2 Regular language constrained shortest paths through each edge

There are several ways in which we could use the approach of Barrett *et al.* to solve the curation-aware pathway reconstruction problem. We start with an observation made in earlier work on

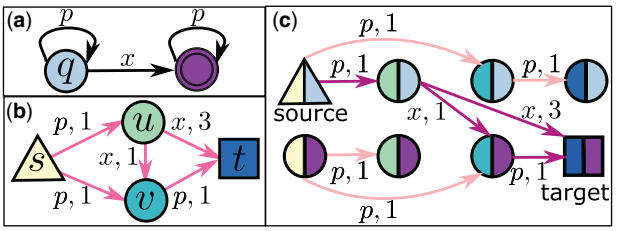

Fig. 2. An illustration of paths that satisfy a regular language. (**a**) The DFA M for the regular expression p^*xp^*. The start state is q. The final state has a double border. (**b**) A graph G with edge labels from the alphabet $\{p, x\}$. (**c**) The product graph $H = G \times M$. Node colors and shapes indicate how we pair nodes in G with states in M, while the color of an edge indicates whether it is possible (dark pink) or not possible (light pink) to find a source–target path through that edge. A path of light pink edges in H corresponds to a path in G whose labels form a string that the DFA will not recognize

curation-*unaware* pathway reconstruction (Ritz *et al.*, 2016): methods that computed compact source-to-target subnetworks, e.g. based on shortest paths between every source-target pair, network flow (Yeger-Lotem *et al.*, 2009), Steiner trees (Yosef *et al.*, 2011) and prize-collecting Steiner forests (Tuncbag *et al.*, 2013), achieved high precision but only low recall (10% or less). The PathLinker algorithm proposed by Ritz *et al.* computed the k shortest paths from any source to any target. Increasing k permitted the smooth expansion of the computed network. However, PathLinker had diminishing returns: several paths computed for larger values of k were composed entirely of edges in earlier paths.

Based on these considerations, we adopt the following approach in this article. As in Section 3.1, we are given a directed, weighted, edge-labeled graph G, a source node s, a target node t and a regular language L. For each edge in G, we compute a shortest s–t path that passes through that edge and satisfies L. We output the union of these at most $|E|$ paths.

Consider solving this problem for a single edge (u, v). The problem is NP-complete if we require the path to be simple (Barrett *et al.*, 2000). Therefore, we compute a *non-simple path* through (u, v) that satisfies L by exploiting two observations. First, if we assign a new set of labels to the edges in G, say the label λ to (u, v) and μ to every other edge, then the label sequence of every path that passes through (u, v) will satisfy the regular expression $\mu^*\lambda(\lambda|\mu)^*$. Second, the intersection of two regular expressions is also regular. Therefore, to compute the desired path, we can utilize the algorithm of Barrett *et al.* by computing an appropriate shortest path in the cross-product graph $G \times M \times M'$, where M is the DFA that recognizes L and M' is the DFA for $\mu^*\lambda(\lambda|\mu)^*$. Since M' has only two nodes, the running time of this algorithm is $O(|H| \log |H|)$, where $H = G \times M$, as in Section 3.1.

As we iterate over the edges of G, the DFA M' remains the same but the product graph $G \times M \times M'$ changes since the edge with the label λ also changes. Thus, a naive application of the above approach to each edge in G will yield a running time of $O(|G||H| \log |H|)$, which is quadratic in the size of G. This running time is undesirable in practice.

We have developed an algorithm that can compute all the desired shortest paths using only the cross-product H in $O(|H| \log |H|)$ time, thus saving a factor of $O(|G|)$. We present only the intuition here. If we did not have to satisfy the constraint imposed by L, then a shortest non-simple path through (u, v) in G is the concatenation of a shortest s–u path, the edge (u, v), and a shortest v–t path. We can compute the shortest paths from s to every node in G and from every node in G to t using two invocations of

Dijkstra's algorithm. Thus, we can compute all the shortest s–t paths through each edge of G in $O(|G|\log|G|)$ time. When we also have to satisfy the constraint imposed by the regular language L, we show that we can generalize this idea to H. We describe this algorithm in detail, prove its correctness, and analyze its running time in Supplementary Section S1. In addition, we illustrate the steps of RegLinker on a toy example in Supplementary Section S2, Figure S5.

3.3 RegLinker

We are now ready to present the RegLinker algorithm for reconstructing signaling pathways, which will use the algorithm of Section 3.2 as a subroutine. Recall that the inputs to this problem are a weighted, directed graph G representing the interactome, a directed graph P representing a curated pathway, a set S of sources and a set T of targets. We assume that P is a subgraph of G.

3.3.1 Multiple sources and targets

The algorithm of Section 3.2 permits only a single source and a single target in the input, whereas RegLinker allows multiple sources and targets. Therefore, in order to use the algorithm of Section 3.2 in RegLinker, we add a supersource s to G and an edge from s to every node in the set S of sources. Analogously, we add a supertarget t to G and an edge from every node in the set T of targets to t. All these edges have unit weight. For each edge in $G - \{s, t\}$, we use the algorithm in Section 3.2 to compute s–t paths in G that satisfy the input regular language. We delete s and t from these paths.

3.3.2 Regular-language constraints

We use an alphabet $\Sigma = \{p, n, x\}$ that contains three labels, standing for 'positive', 'negative' and 'unknown', respectively. Our choice of regular expressions depends on how we apply RegLinker. (i) When we evaluate the accuracy of RegLinker in reconstructing a specific curated pathway, we take P to be a subnetwork of this pathway. We create this subnetwork through a random sampling process. Since it is necessary to explain this process before we can motivate the different types of regular languages we use during evaluation, we present them in later sections (Sections 4.2 and 4.3). (ii) In contrast, when we use RegLinker to suggest new interactions for inclusion in a specific signaling pathway P (Section 4.7), we assign the label p to every edge in P and x to every edge in G–P; we do not use the label n in this analysis. We use the regular expression p^*xxp^*, i.e. we compute paths that contain zero or more edges in P, followed by two consecutive edges not in the pathway, and ending in zero or more edges in P.

3.3.3 Multiple regular languages

It may be desirable to specify many regular languages and to prioritize them. Our implementation can take an ordered list of regular languages in the input. We run RegLinker for each language in this order and concatenate the resulting ranked lists of nodes and edges. If an edge appears in multiple lists, we retain it only in the earliest one.

3.3.4 Ranking edges and nodes in the output

We order the edges in G in increasing order of weight. Then, for each edge e with a rank r (edges may be tied in weight), we assign a rank of r to every edge in the shortest s–t path via e, provided the edge has not already received a rank. For each r, the desired reconstruction P' is the union of P and all edges of rank r or better. We assign each node the smallest rank of any edge incident upon it.

3.4 Other algorithms

Here, we sketch five algorithms that we compare to RegLinker. For each method, we describe how we compute P' and rank its edges for the purpose of evaluation; we rank nodes just as we do for RegLinker.

3.4.1 PathLinker and EdgeLinker

PathLinker (Ritz *et al.*, 2016), which does not accept P as an input, computes the k shortest S–T paths in G in $O(k|V|(|E| + |V|\log|V|))$ time. We used $k = 10\,000$ for our results.

Inspired by RegLinker, we implemented a method we call EdgeLinker that finds the shortest (non-simple) S–T path through each edge in G. It computes these paths in $O(|G|\log|G|)$ time in the same way as RegLinker, i.e. for each edge (u, v), it concatenates the shortest S–u path, the edge, and the shortest v–T path. We define P' to be the union of these paths. To allow EdgeLinker to take advantage of curated interactions, we set the cost of each edge in P to zero, allowing paths to use these edges for free. For both methods, we sort the paths in increasing order of length and assign to each edge the index of the first path in which it appears.

3.4.2 Extended subgraph

This method extends the idea of an induced subgraph. We iteratively compute the r-neighborhood $H(r)$ of P in G, as follows: we set $H(0) = P$. For every $i > 0$, we set $H(i)$ to be the union of all paths of length i in $G - P$, where every edge has label x and each path connects two nodes in P. To compute $H(i)$, we add to G a source node s, a target node t, and for every node u in P, the edges (s, u) and (u, t). We then compute all simple paths between s and t consisting of $i + 2$ edges (adding two to account for the edges incident on s and t) using a modified depth-first search (Sedgewick, 2001). We define P' to be the union of all computed r-neighbourhoods. We rank the edges in P' in decreasing order of weight. We use $r = 3$ in our evaluations, i.e. $P' = H(0) \cup H(1) \cup H(2) \cup H(3)$.

3.4.3 Shortcuts

This method considers only pairs of proteins in P that are not connected by an edge in P (Navlakha *et al.*, 2012). For each such pair, it computes to what extent an edge connecting them can be used as a 'shortcut' to decrease the cost of the sum of shortest s–t paths in P, for each (s, t) pair in $S \times T$. The algorithm adds to P the edge that decreases this sum the most, and iterates until no further edge additions will lead to a decrease in the sum. We modified this algorithm to consider only pairs of nodes in P connected by an edge in $G - P$.

3.4.4 Random walk with edge restarts

Here, we describe a modified random walk with restarts that we use to compute a new set of edge weights for G that take into account the nodes and edges in the pathway P (Fig. 3). For each edge $e = (u, v)$ in P, we add an artificial node w_e and the directed edge (w_e, v). Given a probability $0 \leq q < 1$, at any node u, the walker has two choices: move to an outgoing neighbor v of u with a probability of $(1 - q)c(u, v)$ or move to one of the artificial nodes created above with a probability of $q/|E_P|$, where E_P denotes the set of edges in P. This approach forces a random walker that has transitioned to an artificial node (through a 'restart' step) to take its next step along an edge in P. To ensure that the transition matrix of this random walk is irreducible in practice, we add an edge from each node in G to every other node with a teleportation probability of $|V|/10^6$, as was done by Ritz *et al.* (2016); here, V is the set of nodes in G. In practice, G contains at least one cycle of length two and one cycle of

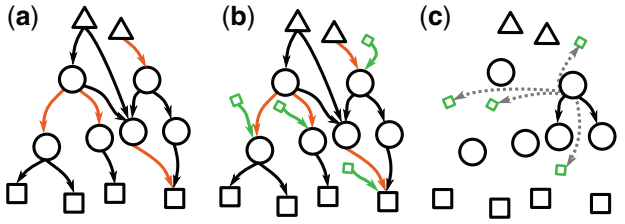

Fig. 3. An illustration of RWER. (**a**) Example interactome. Orange edges are in P and black edges in $G - P$. (**b**) Addition of one artificial node and edge for each edge in P. (**c**) At each step, the random walker can either transition along an edge in G (solid) that is incident on the current node, or follow a 'restart' edge (dashed) to an artificial node

length three. Therefore, this random walk has a stationary distribution, which we compute using power iteration. Finally, we compute a new weight $c(e)$ for each edge $e = (u, v)$ in G as follows: (i) we set $c(e)$ to be the probability of traversing e during RWER, and (ii) we increment $c(e)$ by $c(w_e, v)$, the probability of traversing the corresponding artificial edge.

We use RWER in two ways: (a) to solve the curation-aware pathway reconstruction problem by ranking the edges in $G - P$ in decreasing order of the edge weights computed by this method and (b) as a preprocessing step to reweight G before applying the other algorithms (including RegLinker).

We adapt the approach of Mohammadi *et al.* (2013) to select q such that $(1 - q)/q$, the expected number of consecutive transitions along edges in G (before a restart), is the median distance from a node in P to a target in P. Since we measured this distance to be five in the NetPath pathways that we sought to reconstruct, we set $q = 0.1667$ when we performed RWER.

4 Results

4.1 Datasets

To construct a directed human protein interactome, we used the PSICQUIC server for protein-protein interactions (Aranda *et al.*, 2011) in combination with the PhosphoSitePlus (Hornbeck *et al.*, 2012) and NetPath databases (Kandasamy *et al.*, 2010). The resulting network contains 16 636 nodes and 286 561 interactions. We assigned interaction direction based on evidence of a directed enzymatic reaction (e.g., phosphorylation and dephosphorylation) from any of the source databases. We had sufficient evidence to assign a direction for 7922 of the interactions. We considered the remaining 286 561 interactions as undirected and represent each of these as two directed edges in the interactome. Thus, our interactome G has total of 565 200 edges, many of which are supported by multiple types of evidence. We weighted each edge in the network using a Bayesian approach that computes interaction probabilities based on the sources of evidence (Yeger-Lotem *et al.*, 2009). We provide more details on how we constructed the interactome and computed weights for its edges in Supplementary Section S3.

We identified the signaling receptors and TFs from previously published lists of human receptors (Almen *et al.*, 2009) and TFs (Ravasi *et al.*, 2010; Vaquerizas *et al.*, 2009). We selected 15 NetPath pathways (Kandasamy *et al.*, 2010) that each contained at least one receptor, at least one TF, and had at least three S–T paths. We deleted every protein and interaction from the pathway that did not lie on a receptor-to-TF path in the directed graph representing the pathway. Table 1 displays the number of proteins and interactions in each pathway.

4.2 Evaluation

To evaluate RegLinker and the other algorithms on the problem of reconstructing signaling pathways, we sought to develop a method akin to cross-validation. We decided against using k-fold cross-validation since partitioning the nodes and edges into multiple groups did not appear a realistic way to mimic the state of knowledge about a signaling pathway.

Therefore, we consider three more natural models based on different assumptions about how the knowledge on a given signaling pathway accumulates in the literature. The *interaction withholding* model represents a scenario where we are confident that all the proteins in a pathway have been identified, but not all the interactions among them. Accordingly, we select a fraction α (uniformly at random) of the interactions in the pathway and give each selected interaction a label of x in G. We then assign the remaining interactions in the pathway the label p and present this subgraph as P to each algorithm. Similarly, we select a fraction α of edges from $G - P$ to derive a set of negatives, i.e. we give each sampled interaction the label x and every other interaction in G the label n. Thus, every interaction in G has the label p, x or n.

We also evaluate a second and more realistic *protein withholding* model, where we assume that both proteins and their incident interactions in the pathway are not completely known. Accordingly, we select a fraction β (uniformly at random) of proteins from the curated pathway. In G, we assign the label of x to every edge in the pathway that is incident on at least one of the selected nodes. As before, we assign the label of p to every other edge in the pathway. To derive negative edges, we select a fraction β of nodes in G that are not in the curated pathway and assign a label of x to every edge in G that is incident on at least one selected node. We evaluate each algorithm on its ability to recover withheld interactions as well as withheld proteins.

Finally, we consider a *combined* model that withholds edges based on α and nodes (and their incident edges) through β. In this work, we present the results for the three models using $\alpha = 0.1$ and for $\beta = 0.1$. We repeat the sampling process ten times for each pathway under each model. For each algorithm, we compute its precision-recall curve, aggregated over each random sample per pathway and then aggregated over each pathway.

4.3 Selecting pre-processing strategy and regular languages

We used the following intuition to guide our selection of regular language constraints. In the interaction withholding model, the more the number of x-labeled edges in a path, the less likely they are to be members of the pathway. Therefore, we allowed a small number (≤ 3) of x-labeled edges in a path, namely, p^*xp^*, $p^*xp^*xp^*$, and $p^*xp^*xp^*xp^*$. We use 'flex-a' to denote expressions of this form, where 'a' refers to the number of x-labeled interactions permitted by the expression. When we combine many such expressions, we refer to them as 'flex-a-b-c', i.e. we prefer paths with a x-labeled interactions, before paths with b such interactions and so on. Note that we are effectively using regular expressions to count the number of x-labeled interactions in a path.

Similarly, under the protein withholding model, every path must pass through at least two x-labeled interactions consecutively in order to recover a withheld pathway protein. Therefore, we used regular expressions of the form p^*xxp^* and p^*xxxp^*, which allow only paths with two or three x-labeled interactions appearing in succession. We use 'consec-a' to denote such expressions. These regular

Table 1. Sizes of 15 NetPath pathways and of the number of new proteins and interactions in their proposed extensions

	Original		Proposed extension		
Pathway	P	I	P	I	Proteins annotated to a significant GO term (%)
BDNF	35	81	24	78	21 (87.5)
EGFR1	184	1327	134	466	112 (83.58)
IL-1	35	154	29	75	26 (89.66)
IL-2	44	168	32	97	27 (84.38)
IL-3	40	126	25	91	20 (80.00)
IL-6	45	144	28	81	27 (96.43)
IL-7	13	42	11	31	9 (81.82)
KitReceptor	52	159	46	124	38 (82.61)
Leptin	35	105	23	61	12 (52.17)
Prolactin	52	160	44	141	39 (88.64)
RANKL	30	101	21	67	18 (85.71)
TCR	106	403	63	251	55 (87.30)
TGFβ	170	769	131	387	99 (75.57)
TNFα	195	836	161	512	130 (80.75)
Wnt	90	397	75	194	57 (76.00)

Note: In the final column, we consider only GO terms that are significant at the $P < 0.001$ level.
P, proteins; I, interactions.

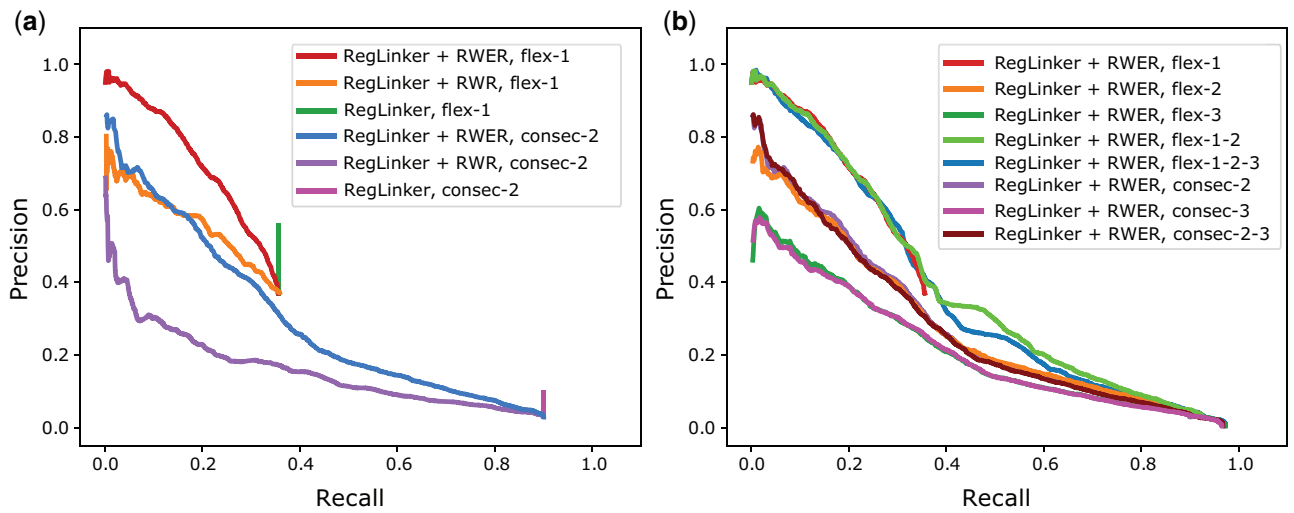

Fig. 4. Precision–recall curves of RegLinker for edge recovery in the combined withholding model for different (**a**) interactome weighting methods and (**b**) regular language constraints

expressions restrict the number of proteins in the path that are not members of the pathway to be at most one and two, respectively.

The preprocessing strategy used for RegLinker also bears consideration. Figure 4a displays the results for edge recovery with the flex-1 and consec-2 regular expressions under the combined withholding model ($\alpha = 0.1$, $\beta = 0.1$) for RegLinker using different interactome preprocessing strategies (RWER, RWR and the evidence-based weights). For RWR, we restart to nodes that are tails of edges in P. We see that RWER offers a significant advantage over the competing preprocessing methods. The results also emphasize the non-specificity of evidence-based weighting: a large number of interactions (both within and outside curated pathways) have a high weight. Consequently, the corresponding precision-recall curves begin at a high recall and moderate precision. In light of this result, we henceforth apply RegLinker on the RWER-weighted interactome, referring to this approach as 'RegLinker+ RWER'. We do the same for Extended Subgraph. We do not apply the preprocessing for EdgeLinker, as the reweighting conflicts how we input P to it.

Figure 4b demonstrates the importance of selecting the appropriate regular languages to provide as constraints for RegLinker + RWER. The constraint flex-1 (red curve) offers the best precision but much lower recall than other curves. In contrast, flex-2 (orange) and flex-3 (green) offer successively higher recall; however, their precision is lower. Prioritizing these constraints (flex-1-2, light green and flex-1-2-3, blue) combines the higher precision of flex-1 with the higher recall of flex-2 and flex-3. These results indicate that we do not incur a precision penalty for including expressions successively in this manner. The consec-a constraints, whether individual or combined (purple, pink and brown) have lower precision than flex-1-2-3. Therefore, we subsequently elect to show the results obtained when providing RegLinker + RWER the constraints 'flex-1-2-3' (in the case of edge recovery) and the constraints 'consec-2-3' (in the case of node recovery). In Supplementary Section S4.2, we present DFA sizes (Supplementary Table S1) and the product graph sizes and running times (Supplementary Table S2).

4.4 Interaction withholding model

This model offers a simple test of the reconstruction capabilities of an algorithm. In practice, most proteins in the pathway will still be connected to at least one of the interactions labeled p. Algorithms like RegLinker can potentially use these labels to greatly prune the search space. RegLinker + RWER with the flex-1-2-3 constraint (brown curve in Fig. 5) and Extended Subgraph + RWER (green) achieves the highest AUPRC values (0.69), clearly dominating the others. PathLinker's performance (red) achieves an AUPRC of only 0.18. On the other hand, EdgeLinker (blue) has a precision-recall curve that starts with recall almost at one, an artifact of the Bayesian weighting as previously discussed in Section 4.3. With an AUPRC of 0.26, RWER (orange) is superior to PathLinker but considerably worse than RegLinker + RWER and Extended Subgraph + RWER. The Shortcuts algorithm (purple) achieves moderate to low precision but very low recall since it adds only a few edges in the induced subgraph of P. We do not consider it further. We provide statistics on the running time of all methods for this model in Supplementary Section S4.2 (Table S3).

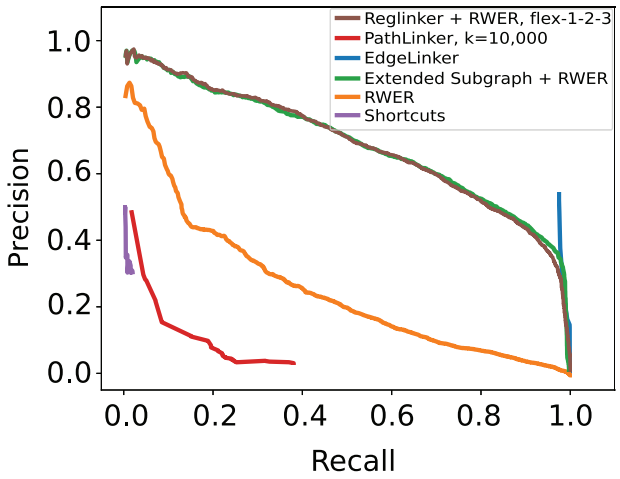

Fig. 5. Precision-recall curves under the interaction withholding model ($\alpha = 0.1$)

4.5 Protein withholding model

The protein withholding model presents a more realistic challenge by removing any knowledge of the withheld proteins from the pathway P presented to the reconstruction algorithms. Thus, to achieve high recall of withheld proteins and the interactions involving them, algorithms such as Extended Subgraph and RegLinker must recover consecutive pairs of interactions, e.g. (u, v) and (v, w), where v is the protein withheld.

4.5.1 Protein recovery

RWER (orange curve, AUPRC = 0.37, Fig. 6a) compares favorably to the other algorithms, including RegLinker + RWER (consec-2-3) (brown curve, AUPRC = 0.34). Note that RWER is not guaranteed to produce pathway-like subnetworks since it does not explicitly compute source-to-target paths. The difficulty presented by this model is also readily apparent. Notably, Extended Subgraph (green) performs much more poorly in this evaluation, achieving an AUPRC of just 0.25. Indeed, at points, the precision of PathLinker (red, AUPRC = 0.18), which serves as a control, surpasses that of Extended Subgraph. EdgeLinker (blue) also does very poorly.

4.5.2 Interaction recovery

The increased challenge of this model is also reflected in Figure 6b, which shows that all algorithms except PathLinker suffer a significant performance decrease in interaction recovery compared to the interaction withholding model (Fig. 5). Once again, RegLinker + RWER (consec-2-3) (brown, AUPRC = 0.22) and Extended Subgraph (green, AUPRC = 0.19) offer the best performance. However, RWER (orange, AUPRC = 0.16) displays comparable performance to both methods as recall increases beyond 0.4.

4.6 Combined withholding model

Figure 7 shows the performance attained by algorithms under the combined withholding model. When recovering withheld proteins, RegLinker + RWER (flex-1-2-3, purple and consec-2-3, brown) obtained a similar AUPRC (0.35); RWER (orange curve) and Extended Subgraph + RWER (green curve) achieved AUPRCs of 0.37 and 0.29, respectively (Fig. 7a). For edge recovery, RegLinker + RWER (flex-1-2-3) had an AUPRC of 0.36 and

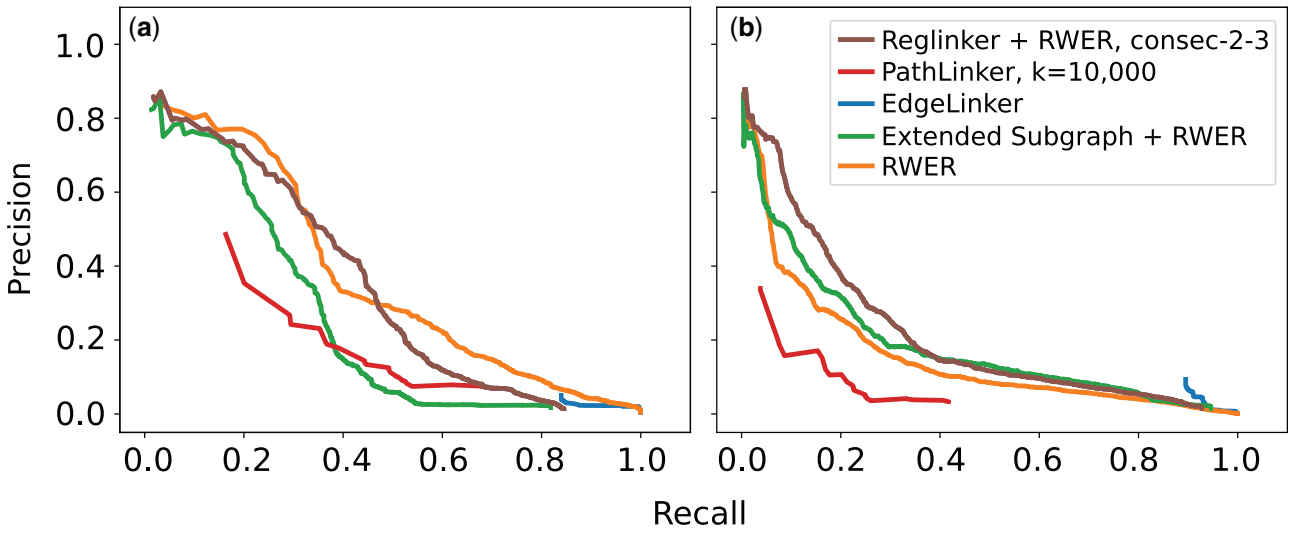

Fig. 6. Precision-recall curves under the protein withholding model ($\beta = 0.1$) for (**a**) protein recovery and (**b**) interaction recovery

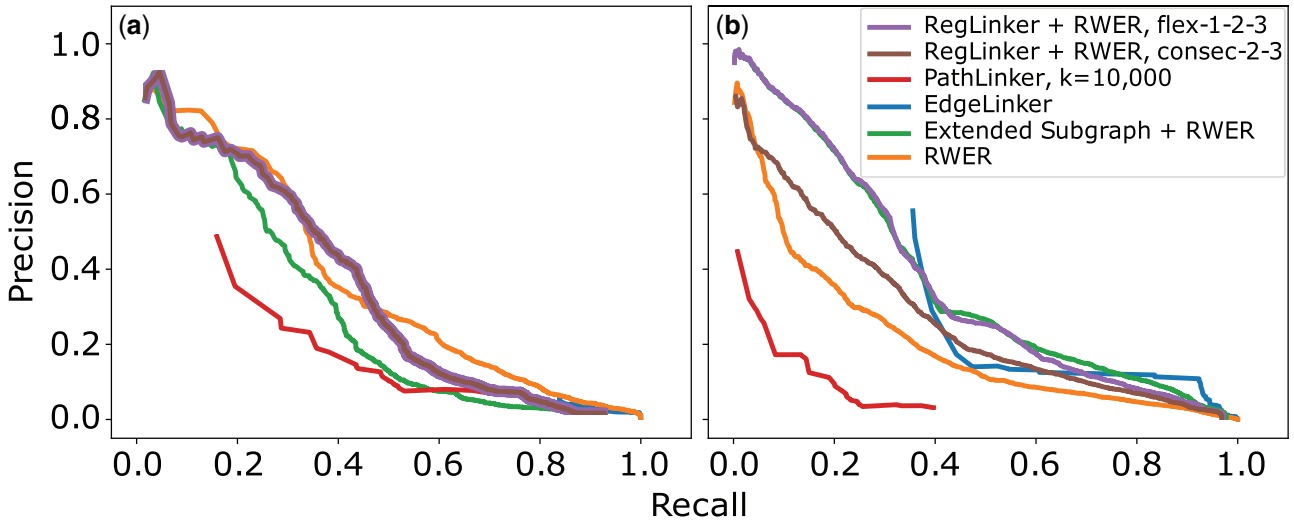

Fig. 7. Precision-recall curves under the combined withholding model ($\alpha = 0.1$, $\beta = 0.1$) for (**a**) protein recovery and (**b**) interaction recovery. In (a), we show the plot for flex-1-2-3 (purple) as a thick curve to indicate that it is almost identical to the consec-2-3 curve (brown)

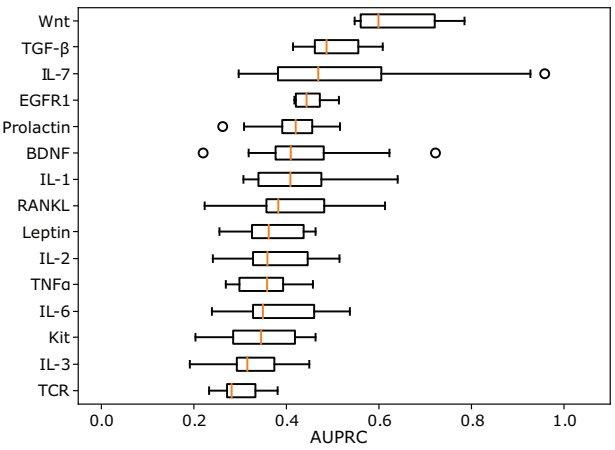

Fig. 8. Distribution of AUPRC values for each pathway for RegLinker + RWER (flex-1-2-3) for interaction recovery in the combined withholding model ($\alpha = \beta = 0.1$)

RegLinker + RWER (consec-2-3) had an AUPRC of 0.27, while RWER had an AUPRC of 0.21 and Extended Subgraph + RWER had an AUPRC of 0.37 (Fig. 7b). Considering both sets of plots together, RegLinker + RWER flex-1-2-3 offers the most consistent performance.

The curves for protein recovery (Fig. 7a) closely mirror those seen in Figure 6a, while the trends for interaction recovery (Fig. 7b) for each algorithm generally fall between edge recovery under the node and edge withholding models (Figs 5 and 6b, respectively). With $\alpha = 0.1$ and $\beta = 0.1$, we withhold nearly 20% of interactions in an average pathway. Withheld interactions are nevertheless easier for the algorithm to identify in the combined model than the protein withholding model.

4.7 RegLinker + RWER interaction recovery across individual pathways

So far, we have been studying precision-recall curves aggregated over all pathways. However, some pathways may be more amenable to reconstruction than others. To investigate this possibility, for each pathway, we examined the distribution of AUPRC scores (over the 10 runs for each pathway) of RegLinker + RWER using the flex-1-2-3 expressions. The median AUPRC for interaction recovery in the combined withholding model varied from 0.25 to 0.6 (Fig. 8).

Many of the smaller pathways (e.g. BDNF, IL-7 and RANKL) had a large variation in the AUPRC values. Smaller pathways may be more sensitive to randomized withholding in our models. For example, the IL-7 pathway has 13 proteins and 42 interactions with Janus Kinase (JAK) connected to nine proteins while the tyrosine-protein kinase Fyn (FYN) is connected to only two; hence, removal of even a single protein in this pathway can have widely varying results, depending on the protein removed. In contrast, larger pathways, such as EGFR, TGF-β and Wnt, display less variance. These results reflect the challenge in creating reconstruction methods that work robustly across a range of pathways.

4.8 Proposing pathway extensions using RegLinker

We applied RegLinker to propose extensions to the 15 NetPath pathways under consideration. To this end, for each pathway P, we assigned the label p to every interaction in P and the label x to every other interaction in the interactome G, i.e. to every edge in $G - P$. We noted that paths constrained by the 'flex–a' type of regular expression do not always suggest any proteins for inclusion in the pathway. Therefore, we focused our attention on 'consec-a' type of regular expressions that we used in the protein withholding model. Specifically, we used the 'consec-2' regular expression i.e. p^*xxp^*. We removed every x-labeled interaction in G that connected two proteins in P; this step ensured that the node incident on both x-labeled edges in every path that satisfied p^*xxp^* was not already be a member of P. We then computed the shortest path through every edge labeled p that satisfied p^*xxp^*. We proposed the union of these paths as the extension of P. For each pathway, Table 1 summarizes its size and that of the proposed extension. We also considered computing these paths through every edge in G. We did not analyze these results since the resulting extensions contained thousands of nodes and edges.

In order to investigate the cellular functions of the proposed proteins and the relevance to their respective pathways, we used Enrichr (Kuleshov *et al.*, 2016) to compute their enrichment in gene ontology (GO) biological process (BP) terms and in other biological pathway databases. For as many as 14 of the 15 pathways, we observed that over 75% of the proposed proteins were annotated to at least one significant GO BP term ($P < 0.001$) (final column of Table 1). Due to the hierarchical structure of the GO, several enriched terms were closely related to each other. Therefore, we used REVIGO (Supek *et al.*, 2011) in order to summarize the significant GO terms.

Here, we focus our attention on three pathways of different sizes: BDNF [35 proteins, 81 interactions, Wnt (90 and 395) and TNFα (195 and 386)]. We observed that a large fraction of the new proteins proposed by RegLinker had functions corresponding to the biological role of the ligand involved in the respective pathway. Supplementary Table S4 summarizes the GO terms and pathways we discuss below.

4.8.1 BDNF pathway

BDNF is a protein that promotes the survival of neurons. Its neuroprotective and anti-apoptotic effects are well known (Chen *et al.*, 2017). We observed that 11 out of 24 proteins proposed by RegLinker for this pathway are annotated to the term 'negative regulation of apoptotic process' (GO: 0043066, $P < 10^{-8}$). Furthermore, recent work has implicated BDNF in promoting angiogenesis in endothelial cells (ECs) (Kermani and Hempstead, 2007). However, the mechanism by which BDNF maintains the angiogenetic potential of ECs is still unknown (Wang *et al.*, 2019). We observed that four out of 24 proposed proteins for BDNF are annotated to 'regulation of angiogenesis' (GI: 0045765, $P < 10^{-3}$); in contrast, only one protein (NTF4) in the original BDNF pathway is annotated to this term. Therefore, our proposed extension to the current BDNF pathway may help to shed light on the role of BDNF in angiogenesis.

4.8.2 Wnt pathway

RegLinker identified 75 new proteins to add to the Wnt signaling pathway. Of these, 12 proteins are annotated to 'negative regulation of Wnt signaling' (GO: 0030178, $P < 10^{-8}$). Thus, all these proteins are ideal candidates for inclusion in the pathway. Moreover, the interactions that connect them to proteins that are already in the NetPath-curated Wnt pathway may suggest mechanisms of negative regulation. A second relevant GO term was 'macrophage differentiation' (GO: 0030225). Three of the proposed proteins are annotated to this term ($P < 10^{-3}$). A recent study reported the activation of Wnt signaling during monocyte-to-macrophage differentiation (Yang *et al.*, 2018). Enrichr also reported that seven proteins were annotated to Adherens junction pathway in the KEGG database (hsa04520, $P < 10^{-7}$), which is a well-known non-canonical Wnt pathway (Amin and Vincan, 2012). Interestingly, six of the proteins suggested by RegLinker are annotated to the Wnt pathway in the KEGG database (hsa04310, $P < 10^{-3}$).

4.8.3 TNFα pathway

TNFα is implicated in apoptosis (Rath and Aggarwal, 1999), and 39 out of 161 proposed proteins for TNFα pathway are annotated to the term 'regulation of apoptotic process' (GO: 0042981, $P < 10^{-15}$). Enrichr found that 16 proposed proteins are members of the KEGG Influenza A pathway (hsa05164, $P < 10^{-9}$) and 11 additional proteins are in the KEGG NF-κB signaling pathway (hsa04064, $P < 10^{-7}$). Studies have shown that TNFα has an anti-influenza virus activity (Seo and Webster, 2002) and that it has anti-rotavirus activity via the activation of NF-κB signaling pathway (Hakim *et al.*, 2018).

5 Discussion and conclusions

In this article, we have formulated the curation-aware pathway reconstruction problem and developed RegLinker as a solution. The key idea at the heart of RegLinker is using regular expressions to control the relative proportion of novel interactions in the reconstructed pathway. Regular expressions offer considerable flexibility. For instance, if we are given information on whether a (directed) edge indicates activation or inhibition, we can use regular expressions to compute paths where the total effect of all interactions in the path is activating or inhibitory (record whether the number of inhibitory edges is even or odd). We can also integrate other types of data, e.g. consider only paths where a certain number of genes is differentially expressed in an RNA-seq dataset. Another possibility is to encode cellular location of proteins, e.g. to compute paths whose proteins are located in a specific sequence of organelles.

We evaluated RegLinker and five competing algorithms on three models for withholding information on a curated pathway. PathLinker and EdgeLinker took no or imperfect advantage of the pathway, leading them to recover pathway proteins and interactions poorly. The different facets of Extended Subgraph and RWER revealed their different strengths. Neighborhood constraints enabled Extended Subgraph to significantly reduce the edges it considers when recovering interactions; as a result, it performs better than RWER at this task. However, RWER displays better protein recovery performance, indicating that that the constraints that aid Extended Subgraph in filtering interactions force it to make mistakes when ranking proteins. By using receptor–to–TF shortest paths that are constrained by regular languages, RegLinker + RWER presents a compromise between these approaches, proving capable in each evaluation.

These results also underscore the greatly increased difficulty of pathway reconstruction under the protein and combined withholding models. Our results reveal that evidence-based weighting procedures and reconstruction approaches rooted in graph topology are alone insufficient to reliably distinguish withheld pathway interactions. In contrast, we have demonstrated that RegLinker + RWER produces competitive reconstructions under various hypothetical models of pathway knowledge, guided by the flexibility afforded by regular languages. Our results offer insights into the promise of these automated techniques for pathway reconstruction, and the obstacles they must overcome if they are to generalize well across multiple signaling pathways.

Methods such as RegLinker are appropriate for signaling pathways and other biological processes where sources and targets can be readily identified. The vast majority of interactions are unlikely to be related to signaling. Developing approaches to annotate interactions to non-signaling related processes is an important open problem.

Acknowledgements

We thank Jeff Law for providing the interaction network.

Funding

This work was supported by grants from the National Science Foundation [CCF-1617678, DBI-1759858] and the National Institute of General Medical

Sciences [R01-GM095955] to T. M. M. We also acknowledge support from the Computational Tissue Engineering interdisciplinary graduate education program at Virginia Tech.

Conflict of Interest: none declared.

References

Almen,M. *et al.* (2009) Mapping the human membrane proteome: a majority of the human membrane proteins can be classified according to function and evolutionary origin. *BMC Biol.*, **7**, 50.

Amin,N. and Vincan,E. (2012) The Wnt signaling pathways and cell adhesion. *Front. Biosci.*, **17**, 784–804.

Aranda,B. *et al.* (2011) PSICQUIC and PSISCORE: accessing and scoring molecular interactions. *Nat. Methods*, **8**, 528–529.

Barrett,C. *et al.* (2000) Formal-language-constrained path problems. *SIAM J. Comput.*, **30**, 809–837.

Chen,S.-D. *et al.* (2017) More insight into BDNF against neurodegeneration: anti-apoptosis, anti-oxidation, and suppression of autophagy. *Int. J. Mol. Sci.*, **18**, 545.

Fabregat,A. *et al.* (2016) The Reactome pathway Knowledgebase. *Nucleic Acids Res.*, **44**, D481–D487.

Gitter,A. *et al.* (2011) Discovering pathways by orienting edges in protein interaction networks. *Nucleic Acids Res.*, **39**, e22.

Hakim,M.S. *et al.* (2018) TNF-α exerts potent anti-rotavirus effects via the activation of classical NF-κB pathway. *Virus Res.*, **253**, 28–37.

Hornbeck,P.V. *et al.* (2012) PhosphoSitePlus: a comprehensive resource for investigating the structure and function of experimentally determined post-translational modifications in man and mouse. *Nucleic Acids Res.*, **40**, D261–D270.

Jiang,B. *et al.* (2017) AptRank: an adaptive PageRank model for protein function prediction on bi-relational graphs. *Bioinformatics*, **33**, 1829–1836.

Kandasamy,K. *et al.* (2010) NetPath: a public resource of curated signal transduction pathways. *Genome Biol.*, **11**, R3.

Kanehisa,M. *et al.* (2016) KEGG as a reference resource for gene and protein annotation. *Nucleic Acids Res.*, **44**, D457–D462.

Kermani,P. and Hempstead,B. (2007) Brain-derived neurotrophic factor: a newly described mediator of angiogenesis. *Trends Cardiovasc. Med.*, **17**, 140–143.

Kuleshov,M.V. *et al.* (2016) Enrichr: a comprehensive gene set enrichment analysis web server 2016 update. *Nucleic Acids Res.*, **44**, W90–W97.

Mohammadi,S. *et al.* (2013) Inferring the effective TOR-dependent network: a computational study in yeast. *BMC Syst. Biol.*, **7**, 84.

Navlakha,S. *et al.* (2012) A network-based approach for predicting missing pathway interactions. *PLoS Comput. Biol.*, **8**, e1002640.

Rath,P.C. and Aggarwal,B.B. (1999) TNF-induced signaling in apoptosis. *J. Clin. Immunol.*, **19**, 350–364.

Ritz,A. *et al.* (2016) Pathways on demand: automated reconstruction of human signaling networks. *NPJ Syst. Biol. Appl.*, **2**, 16002.

Sedgewick,R. (2001) *Algorithms in C, Part 5: Graph Algorithms*, 3rd edn. Addison-Wesley, Boston, MA.

Seo,S.H. and Webster,R.G. (2002) Tumor necrosis factor alpha exerts powerful anti-influenza virus effects in lung epithelial cells. *J. Virol.*, **70**, 7388–7397.

Silverbush,D. and Sharan,R. (2014) Network orientation via shortest paths. *Bioinformatics*, **30**, 1449–1455.

Steffen,M. *et al.* (2002) Automated modelling of signal transduction networks. *BMC Bioinformatics*, **3**, 34.

Supek,F. *et al.* (2011) REVIGO summarizes and visualizes long lists of gene ontology terms. *PLoS One*, **6**, e21800.

Supper,J. *et al.* (2009) BowTieBuilder: modeling signal transduction pathways. *BMC Syst. Biol.*, **3**, 67.

Ravasi,T. *et al.* (2010) An atlas of combinatorial transcriptional regulation in mouse and man. *Cell*, **140**, 744–752.

Tuncbag,N. *et al.* (2013) Simultaneous reconstruction of multiple signaling pathways via the prize-collecting Steiner forest problem. *J. Comput. Biol.*, **20**, 124–136.

Vaquerizas,J.M. *et al.* (2009) A census of human transcription factors: function, expression and evolution. *Nat. Rev. Genet.*, **10**, 252–263.

Wang,Z. *et al.* (2019) The TrkB-T1 receptor mediates BDNF-induced migration of aged cardiac microvascular endothelial cells by recruiting Willin. *Aging Cell*, **18**, e12881.

Yang,Y. *et al.* (2018) Crosstalk between hepatic tumor cells and macrophages via Wnt/β-catenin signaling promotes M2-like macrophage polarization and reinforces tumor malignant behaviors. *Cell Death Dis.*, **9**, 793.

Yeger-Lotem,E. *et al.* (2009) Bridging high-throughput genetic and transcriptional data reveals cellular responses to alpha-synuclein toxicity. *Nat. Genet.*, **41**, 316–323.

Yen,J.Y. (1971) Finding the k shortest loopless paths in a network. *Manage. Sci.*, **17**, 712–716.

Yosef,N. *et al.* (2011) ANAT: a tool for constructing and analyzing functional protein networks. *Sci. Signal.*, **4**, pl1.