SOFTWARE TOOL ARTICLE

The PathLinker app: Connect the dots in protein interaction networks [version 1; referees: 1 approved, 2 approved with reservations]

Daniel P. Gil¹, Jeffrey N. Law², T. M. Murali¹,³

¹Department of Computer Science, Virginia Tech, Blacksburg, USA
²Genetics, Bioinformatics, and Computational Biology, Virginia Tech, Blacksburg, USA
³ICTAS Center for Systems Biology of Engineered Tissues, Virginia Tech, Blacksburg, USA

Abstract
PathLinker is a graph-theoretic algorithm for reconstructing the interactions in a signaling pathway of interest. It efficiently computes multiple short paths within a background protein interaction network from the receptors to transcription factors (TFs) in a pathway. We originally developed PathLinker to complement manual curation of signaling pathways, which is slow and painstaking. The method can be used in general to connect any set of sources to any set of targets in an interaction network. The app presented here makes the PathLinker functionality available to Cytoscape users. We present an example where we used PathLinker to compute and analyze the network of interactions connecting proteins that are perturbed by the drug lovastatin.

Keywords
signaling pathways, pathway reconstruction, protein interaction networks, PathLinker, Cytoscape, k-shortest paths

This article is included in the International Society for Computational Biology Community Journal gateway.

This article is included in the Cytoscape Apps gateway.
**Introduction**

Signaling pathways are a cornerstone of systems biology. While several databases store high-quality representations of these pathways, they require time-consuming manual curation. **PathLinker** is an algorithm that automates the reconstruction of any human signaling pathway by connecting the receptors and transcription factors (TFs) in that pathway through a physical and regulatory interaction network. In previous work, we have demonstrated that **PathLinker** achieved much higher recall (while maintaining reasonable precision) than several other methods. Furthermore, it was the only method that could control the size of the reconstruction while ensuring that receptors were connected to TFs in the result. We have also experimentally validated **PathLinker**’s novel finding that CFTR, a transmembrane protein, facilitates the signaling from receptor tyrosine kinase Ryk to the phosphoprotein Dab2, which controls signaling to β-catenin in the Wnt pathway. These encouraging results suggest that **PathLinker** may serve as a powerful approach for discovering the structure of poorly studied processes and prioritizing both proteins and interactions for experimental study.

More generally, **PathLinker** can be useful for connecting sources to targets in protein networks, a problem that has been the focus of many studies in the past. Applications have included explaining high-throughput measurements of the effects of gene knockouts, discovering genomic mutations that are responsible for changes in downstream gene expression, studying crosstalk between different cellular processes, and linking environmental stresses through receptors to transcriptional changes.

In this paper, we describe a Cytoscape app that implements the **PathLinker** algorithm. We describe in detail a use case where we employ **PathLinker** to analyze the Environmental Protection Agency’s ToxCast data. Specifically, we compute and analyze the network of interactions connecting proteins that are perturbed in this dataset by lovastatin, a drug used to lower cholesterol. We conclude by comparing **PathLinker** to other path-based Cytoscape apps.

**Methods**

**Implementation**

**PathLinker** requires three inputs (Figure 1): a (directed) network $G$, a set $S$ of sources, and a set $T$ of targets. Each element of $S$ and $T$ must be a node in $G$. Each edge in $G$ may have a real-valued weight. The primary algorithmic component of **PathLinker** is the computation of the $k$ best-scoring loopless paths in the network from any source in $S$ to any target in $T$ (Figure 1). By loopless, we mean that a path contains any node at most once. The definition of the score of a path depends on the interpretation of the edge weights, as described in “Operation.” **PathLinker** computes the $k$-highest scoring paths by integrating Yen’s algorithm with the A* heuristic, which allows very efficient computation for very large $k$ values, e.g., 20,000, on networks with hundreds of thousands of edges; see Table 2 below for statistics on the running time. **PathLinker** outputs the sub-network composed of the $k$ best paths.

One of the first steps in Yen’s algorithm is to compute the shortest path from $T$ to $S$. Initially, we implemented this step by running Dijkstra’s algorithm after reversing $G$. Reversing the network using the Cytoscape API proved to be time costly. Therefore, we modified our implementation of Dijkstra’s algorithm to traverse edges from target to source. Yen’s algorithm periodically requires the temporary removal of edges from the network. However, it transpires that using the Cytoscape API to delete and add edges is inefficient. Therefore, we maintain a set of “hidden edges,” which our implementation of Yen’s algorithm ignores. When **PathLinker** completes, the app renders the computed network using the built-in hierarchical layout, if $k \leq 200$. Since this layout renders the network upside down, i.e., with source nodes at the bottom and target nodes at the top, we reflected node coordinates around the x-axis before displaying the layout.

**Operation**

We have implemented **PathLinker** in Java 7. We have tested it with Cytoscape v3.2, 3.3, and 3.4. **PathLinker** requires a network to be...
already loaded in Cytoscape. To run PathLinker on the currently selected network, the user needs to fill in the inputs and press the “Submit” button. The input panel has three sections (Figure 2(a)):

Sources/Targets: The names of the sources and the targets, separated by spaces. If there are sources or targets that are not nodes in the network, PathLinker will warn the user, identify the errant nodes, and ask the user for permission to continue with the remaining nodes. If none of the sources or none of the targets are in the network, PathLinker will exit. There are two options here:

  Allow sources and targets in paths: Normally, PathLinker removes incoming edges to sources and outgoing edges from targets before computing paths. If the user selects this option, PathLinker will not remove these edges. Therefore, source and target nodes can appear as intermediate nodes in paths computed by PathLinker.

  Targets are identical to sources: If the user selects this option, PathLinker will copy the sources to the targets field. This option allows the user to compute a subnetwork that connects a single set of nodes. In this case, PathLinker will allow sources and targets to appear in paths, i.e., it will behave as if the previous option is also selected. Note that since PathLinker computes loopless paths, if the user inputs only a single node and selects this option, PathLinker will not compute any paths at all.

Algorithm: There are two parameters here.

  $k$: the number of paths the user seeks. The default is $k = 200$. If the user inputs an invalid value (e.g., a negative number or a non-integer), PathLinker will use the default value.

Figure 2. PathLinker screenshots. (a) The input panel for the app. (b) PathLinker lovastatin results (described in “Use Case”).
**Edge penalty:** This value is relevant only when the network has edge weights. In the case of additive edge weights, **PATHLINKER** will penalize each path by a factor equal to the product of the number of the edges in the path and the value of this parameter. In other words, each edge in the path will increase the cost of the path by the value of this parameter. When edge weights are multiplicative, **PATHLINKER** performs the same penalization but only after transforming the weights and the edge penalty to their logarithms. The default value is one for multiplicative weights and zero for the other two cases.

**Edge weights:** There are three options for the edge weights to be used in the algorithm:

- **No weights:** The score of a path is the number of edges in it. **PATHLINKER** computes the \( k \) paths of lowest score.

- **Edge weights are additive:** The score of a path is the sum of the weights of the edges in it. **PATHLINKER** computes the \( k \) paths of lowest score in this case as well.

- **Edge weights are probabilities:** This situation arises often with protein interactions networks, since such a weight indicates the experimental reliability of an edge. **PATHLINKER** treats the edge weights as multiplicative and computes the \( k \) highest cost paths, where the cost of a path is the product of the edge weights. Internally, **PATHLINKER** transforms each weight to the absolute value of its logarithm to map the problem to the additive case.

**Output:** The user can select a checkbox to generate a sub-network containing the nodes and edges in the top \( k \) paths. If \( k \leq 200 \), **PATHLINKER** will display this sub-network using the built-in hierarchical layout (Figure 3). If \( k > 200 \), **PATHLINKER** will use the default layout algorithm.

When it completes, **PATHLINKER** opens a table containing the \( k \) paths. Each line in the table displays the rank of each path, its score, and the nodes in the path itself. The user may analyze the network computed by **PATHLINKER** using other Cytoscape apps. The next section describes a use case that further elaborates on these possibilities.

**Use Case: analysis of ToxCast data for lovastatin**

The Environmental Protection Agency’s (EPA) Toxicity Forecaster (ToxCast) initiative and its extension Tox21, have screened over 9,000 chemicals (such as pesticides and pharmaceuticals) using high-throughput assays designed to test the response of many receptors, TFs, and enzymes in the presence of each chemical\(^1\),\(^2\),\(^3\). Here we show a use case on how to integrate **PATHLINKER** with the ToxCast data to examine possible signaling pathways by which the chemical lovastatin could affect a cell.

**Input datasets and pre-processing.** We downloaded the “ToxCast & Tox21 Summary Files” data from the ToxCast website\(^4\). In these data, lovastatin perturbed three receptors (EGFR, KDR and TEK) and five TFs (MTF1, NFE2L2, POU2F1, SMAD1 and SREBF1). We used these proteins as the sources and targets, respectively, for **PATHLINKER** (Figure 2(a)). Rather than use the default Cytoscape human network, we used the interactome used in the original **PATHLINKER** paper\(^1\), which contained 12,046 nodes and 152,094 edges.

**Figure 3.** A hierarchical layout of the lovastatin network computed by **PATHLINKER**. We have mapped UniProt ids to gene names in this network.
directed edges (http://bioinformatics.cs.vt.edu/~murali/supplements/2016-sys-bio-applications-pathlinker). We preferred this network as we had used a popular Bayesian approach to estimate edge weights so as to favor signaling interactions.

Running PathLINKER. We used \( k = 50 \), no edge penalty (i.e., a penalty of 1), and the option for edge weights that indicated that they are like probabilities (Figure 2(a)). The results appear in Figure 2(b) and Figure 3. Each row in Figure 2(b) describes a path: its index (from 1 to \( k = 50 \)), the score of the path, and the nodes in the path, ordered from receptor to TF. Note that the score of the path is the product of the weights of the edges in it, due to the edge weight option we selected. Since PathLINKER prefers high-scoring paths in this case, the paths appear in decreasing order of score. Figure 3 displays a hierarchical layout of the sub-network composed of the paths computed by PathLINKER.

Further analysis. We mapped the node UniProt accession number names to gene names using UniProt’s ID mapping tool (http://www.uniprot.org/uploadlists), imported the mapping results to the PathLINKER network, and then changed the node labels using the Style tab. Finally we applied a hierarchical layout to the (lovastatin) sub-network and spread apart overlapping nodes to make the paths easier to visualize (Figure 3). We noted that the target MTF1 did not appear in any of the top 50 paths.

Functional Enrichment. Since the result from PathLINKER is a network in the current session of Cytoscape, it is amenable for analysis by other Cytoscape apps. As an example, we demonstrate how we applied the ClueGo app for functional enrichment to see if the lovastatin sub-network was enriched for any Gene Ontology (GO) terms or KEGG pathways. Table 1 displays the top 15 enriched terms/pathways. Most of the paths in the PathLINKER result come from the EGFR source node, so it is not surprising the ErbB signaling pathway is highly significant. We found considerable support in the literature for this pathway and other significant GO terms/pathways. Lovastatin has been shown to inhibit epidermal growth factor (EGF) and insulin-like growth factor 1 (IGF-1)\(^2\). Moreover, the PathLINKER sub-network for lovastatin includes an interaction from EGFR to AKT1, which agrees with a study showing that lovastatin inhibits EGFR dimerization and results in the activation of AKT\(^2\). Lovastatin has also been shown to inhibit the T cell receptor–mediated phagocytosis by macrophages\(^2\). Thus, the network computed by PathLINKER for lovastatin promises to capture several possible mechanisms by which the chemical inhibits cellular pathways.

Running time. As we mentioned earlier in “Implementation,” PathLINKER is very efficient. In Table 2, we show the running time for the PathLINKER app for lovastatin and for a representative set of signaling pathways. Even for \( k = 10,000 \), the app completed in

| Table 1. The top 15 functional enrichment results from the ClueGO app for the Lovastatin network computed by PathLINKER. The column titled “# of Genes” displays the number of genes in the PathLINKER network that are annotated to that GO term/pathway. The column titled “% Associated Genes” shows the percentage of genes annotated to that term/pathway that are in the PathLINKER network. |
|-----------------------------------------------|-------------------|-----------------|-----------------|
| Ontology Source | Ontology Term | Corrected \( p \)-value | # of Genes | % Associated Genes |
| GO | cellular response to peptide hormone stimulus | 3.17 \( \times 10^{-21} \) | 22 | 4.1% |
| GO | response to insulin | 6.18 \( \times 10^{-19} \) | 20 | 4.1% |
| KEGG | ErbB signaling pathway | 6.95 \( \times 10^{-17} \) | 12 | 13.7% |
| GO | Fc receptor signaling pathway | 1.62 \( \times 10^{-15} \) | 17 | 4.0% |
| GO | insulin receptor signaling pathway | 2.62 \( \times 10^{-15} \) | 16 | 4.6% |
| KEGG | AGE-RAGE signaling pathway in diabetic complications | 3.35 \( \times 10^{-14} \) | 11 | 10.8% |
| KEGG | T cell receptor signaling pathway | 4.62 \( \times 10^{-14} \) | 11 | 10.5% |
| KEGG | Focal adhesion | 5.26 \( \times 10^{-14} \) | 13 | 6.4% |
| KEGG | Chronic myeloid leukemia | 7.28 \( \times 10^{-14} \) | 10 | 13.6% |
| KEGG | Acute myeloid leukemia | 5.73 \( \times 10^{-13} \) | 9 | 15.7% |
| GO | DNA-templated transcription, initiation | 1.55 \( \times 10^{-12} \) | 14 | 4.1% |
| KEGG | Prolactin signaling pathway | 5.14 \( \times 10^{-12} \) | 9 | 12.5% |
| GO | positive regulation of T cell activation | 8.00 \( \times 10^{-12} \) | 12 | 5.2% |
| KEGG | Chemokine signaling pathway | 2.98 \( \times 10^{-11} \) | 11 | 5.8% |
| KEGG | FoxO signaling pathway | 3.55 \( \times 10^{-11} \) | 10 | 7.4% |
Comparison to related Cytoscape apps

In this section, we compare PathLinker to other Cytoscape apps that compute paths in networks. A difficulty we faced in understanding the functionality of some of these apps was that they did not precisely define their output in the documentation. Therefore, we had to take recourse to studying the source code for some of these apps in order to understand precisely the properties of the computed paths. We focus the comparison mainly on these properties and not on other features of the apps.

PathExplorer. (http://apps.cytoscape.org/apps/pathexplorer) This app uses breadth first search (BFS) to compute the shortest path from a single node (that the user can select) to every other node in the network. The app can also compute the shortest path from every node in the network to a single node. Since the app uses BFS, the shortest path property is guaranteed only for unweighted networks. If there are multiple shortest paths to a node, it appears that the app will select one.

StrongestPath. (http://apps.cytoscape.org/apps/strongestpath) This app computes the “strongest” paths from a group of source nodes to a group of target nodes. The authors do not provide a definition of “strongest” paths. We describe our understanding of their algorithm now. Suppose the input network is $G$. Their software takes a real-valued threshold $\tau > 0$ as input; the user can manipulate a slider to select this value. The app appears to operate as follows:

1. Connect a super source $s$ to each source in $G$. Connect each target to a super target $t$ in $G$.
2. Use Dijkstra’s algorithm to compute the shortest path in $G$ from $s$ to every node in $G$.
3. Create a new network $G'$ with the same node set as $G$. For every edge $(u, v)$ in $G$, add the reverse of that edge $(v, u)$ to $G'$.
4. Use Dijkstra’s algorithm to compute the shortest path in $G'$ from $t$ to every node in $G$.
5. For every node $v$ in $G$, record $d(v)$ the sum of the length of the shortest $s-v$ path in $G$ and the length of the shortest $t-v$ in $G'$. Compute the corresponding $s-t$ path $\pi$ that goes through $v$.
6. Sort all the nodes in $G$ in increasing order of $d(v)$.
7. Let $a$ be the smallest value of $d(v)$.
8. For every node $v$ such that $d(v) \leq a + \tau$, output the path $\pi$.

In other words, for every node $v$, the app computes the shortest path that starts at some source node, goes through $v$, and ends at some target node. The number of such paths returned depends on the value of the threshold $\tau$ selected by the user. This app can operate on weighted and directed networks. We believe that the algorithm will compute the shortest path from any source to any target correctly. However, when $\tau > 0$, it is not possible to guarantee that the algorithm will compute all paths from a source to a target of length $\leq a + \tau$, since the method computes at most $n$ distinct paths, where $n$ is the number of nodes in the network.

PesCa [25]. (http://apps.cytoscape.org/apps/pesca30) For a single node, this app computes the shortest path from that node to every other node in the network. If the user selects multiple nodes, PesCa computes the shortest path(s) between each pair of selected nodes. A useful feature is that if there are multiple shortest paths between a pair of nodes, the app computes all of them. This app focuses on shortest paths.

PathLinker. Our algorithm is strikingly different in that it allows the user to compute as many $(k)$ shortest paths from sources to targets as desired. For example, if $k = 1$, PathLinker will compute the shortest path from some source to some target using Dijkstra’s algorithm on a graph with a new super source and a super target. For larger values of $k$, Yen’s algorithm (used by PathLinker) uses a dynamic program to mathematically guarantee the following property: if $\pi_{k-1}$ is the $(k-1)$st path and $\pi_k$ is the $k$th path, then there is no source-to-target path in the graph whose length is strictly between the lengths of $\pi_{k-1}$ and $\pi_k$. The other Cytoscape apps discussed here either cannot guarantee this property (e.g., StrongestPath) or do not compute less-than-optimal paths (e.g., PathExplorer and PesCa).

<table>
<thead>
<tr>
<th>Pathway</th>
<th>Lovastatin</th>
<th>TNF$\alpha$ Pathway</th>
<th>TGF$\beta$ Pathway</th>
<th>Wnt Pathway</th>
</tr>
</thead>
<tbody>
<tr>
<td># of sources</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>14</td>
</tr>
<tr>
<td># of targets</td>
<td>5</td>
<td>44</td>
<td>77</td>
<td>14</td>
</tr>
<tr>
<td>$k$</td>
<td>time (sec)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>100</td>
<td>3.6</td>
<td>3.3</td>
<td>4.7</td>
<td>5.2</td>
</tr>
<tr>
<td>1,000</td>
<td>9.8</td>
<td>7.7</td>
<td>10.5</td>
<td>13.8</td>
</tr>
<tr>
<td>10,000</td>
<td>94.3</td>
<td>86.0</td>
<td>116.8</td>
<td>144.4</td>
</tr>
</tbody>
</table>
Summary
We have described a new Cytoscape app that implements a mathematically rigorous, computationally-efficient, and experimentally-validated network connection algorithm called PathLinker. While we had originally developed PathLinker for reconstructing signaling pathways, the method is general enough to connect any set of sources to any set of targets in a weighted and directed network. As a specific example, we used PathLinker to compute the network of interactions connecting proteins perturbed by the drug lovastatin in the ToxCast dataset and showed how the literature supported PathLinker’s findings. The app may also be used to compute a sub-network connecting a single set of nodes. This app promises to be a useful addition to the suite of Cytoscape apps for analyzing networks.

Data and software availability
Software available from: http://apps.cytoscape.org/apps/pathlinker
Latest source code: https://github.com/Murali-group/PathLinker-Cytoscape
Archived source code as at time of publication: 10.5281/zenodo.1651626
License: GNU General Public License version 3

The original Python implementation is available at https://github.com/Murali-group/PathLinker for users who seek to integrate PathLinker directly into their own computational pipelines or want to apply PathLinker for large values of $k$.

Datasets: We obtained the lovastatin data from the following three files in the INVITRODB_V2_SUMMARY.zip file that we downloaded:

- hitc_Matrix_151020.csv
- Chemical_Summary_151020.csv
- Assay_Summary_151020.csv

Author contributions
DPG implemented and tested the Cytoscape app. JNL performed the lovastatin analysis. TMM proposed the development of the PathLinker app and the lovastatin analysis, and supervised DPG and JNL. All three authors wrote the paper.

Competing interests
No competing interests were disclosed.

Grant information
The National Institute of General Medical Sciences of the National Institutes of Health grant R01-GM095955 (TMM) and National Science Foundation (NSF) grant DBI-1062380 (TMM) supported this work.

The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

References

17. USEPA: ToxCast & Tox21 Summary Files from invitrodb_v2. 2015; Data released October 2015. Reference Source.


Open Peer Review

Current Referee Status: ✔ ✔ ✔

Stefan Wuchty
Department of Computer Science, Center for Computational Science, University of Miami, Coral Gables, FL, USA

The manuscript 'The PathLinker app: Connect the dots in protein interaction networks' by Gil, Law and Murali introduces a Cytoscape app that allows the user to apply their PathLinker algorithm to find potential signaling pathways from a user-defined set of sources, targets and molecular interaction data. The underlying PathLinker algorithm has been introduced in the paper in Ritz et al., NPJ Syst. Biol. Appl. 2016, 2:16002, indicating that the current manuscript is an extension in that it provides a Cytoscape application. The manuscript provides a crash-course in using the PathLinker algorithm, allowing the reader to quickly get into the game determining signalling paths based on the user's data. As it stands, it seems to be a popular one and will be used frequently.

While the manuscript gives enough information to get the user going, I would add a bit more information about the specifics of the underlying algorithm. It is based on Yen's algorithm but uses the A* algorithm instead of a shortest path algorithm. While many readers are probably familiar with the latter, the A* algorithm may need an introduction to avoid that users operate a 'black box'. In particular, the A* algorithm makes at each step an assessment of the distance to a target to find an optimal path. In this regard, it would be beneficial to add more details how this assessment works and the ways in which A* was embedded in the framework of Yen's algorithm. As for the latter, also Yen's algorithm deserves more detail as it is an algorithm that users rather rarely encounter to make the user fully aware what she is doing. In particular, such considerations are important as the authors describe in the paper different weights on interactions that may be used in different ways to assess and find optimal paths.

With that said, a bit more technical information about the 'ingredients' of the algorithms that are used to compare (with regard to the ways weighting information is used) would be helpful too. Such details would allow the reader to see where the differences to (and the advantages of) the PathLinker algorithm and apps are.

**Competing Interests:** No competing interests were disclosed.

I have read this submission. I believe that I have an appropriate level of expertise to confirm that it is of an acceptable scientific standard.

Referee Report 22 March 2017

https://doi.org/10.5256/f1000research.10680.r20285
The paper of Gil et al. describes a new Cytoscape App, Pathlinker, which is the Cytoscape implementation of the previously published approach by the Murali group with the same name. It is always useful for the community to implement network analyzing algorithms to Cytoscape.

The paper and the abstract is well written and clear. The figures were well selected.

In order to facilitate the application of the PathLinker App, it would be useful to provide more, tutorial type comments and guidelines for new users. Given the important task PathLinker is meant to solve, many users would find it useful. Currently the Methods section contains the key steps but it does not read as a protocol or suggest alternatives for troubleshooting.

The current version of the paper does not contain the limitations of PathLinker. When this App should not be used, for which datatypes it is not good, or cases when the user should pay attention to any bias or problem?

The comparison with existing Apps focuses on the differences in the algorithms. As this is an App paper, it would be useful to include a comparison of the functional differences (features) between the Apps.

If possible, maybe for a new version, it would be nice if the App allows to input the source and target node names by node selection function, instead of typing it in (or pasting it in) to the requested fields.

Finally, a small bug in the App: When the user select the checkbox to generate a sub-network as an output, it does not generate a subnetwork within Cytoscape but a new network. The problem with this that it means the attributes of the original network will be lost. This should be fixed easily.

I believe PathLinker will be a popular and often used App for the biomedical and systems biology communities. I think the next step to increase its impact is to make the application of it as clear and as didactical as possible.

**Competing Interests:** No competing interests were disclosed.

**We have read this submission. We believe that we have an appropriate level of expertise to confirm that it is of an acceptable scientific standard, however we have significant reservations, as outlined above.**

Referee Report 01 February 2017

https://doi.org/10.5256/f1000research.10680.r19605

Barry Demchak
Department of Medicine, University of California, San Diego, La Jolla, CA, USA
This paper describes the PathLinker Cytoscape app, including the mathematical algorithms and a comparison to similarly-focused Cytoscape apps. It is well written and address the important problem of deducing relationships that can advance biology.

It is very economical in its explanation of the app/algorithm, its uses and its relationship to other apps, and in several places needs more explanation. Explanations tend to weigh in favor of expert Cytoscape users, though this app would be of interest to less expert users, too, particularly those trying to relate PathLinker to biological investigation. The paper would benefit from better enabling the reader to follow a use case in Cytoscape using actual data and actual app settings.

1. In Methods | Operation, please explain how to acquire and run PathLinker.

2. In "Allow sources and targets in paths" and "Targets are identical to sources", please explain the biological implications of these settings ... it's difficult to jump from the graph implications to the biological implications.

3. In "Algorithm", why is the default chosen, and what are the biological ramifications of choosing a higher or lower $k$?

4. The output in Figure 2B seems to be a standalone window. How can the user capture the results? It's unclear how the user should be using this report in investigating relationships.

5. In "Edge penalty", please explain when a edge penalty would be used in a network and what its biological implication would be.

6. In "Input datasets and pre-processing", I attempted to download the ToxCost data and could not. The site requires a credential and does not give instructions regarding how to get the credential. Without this data, the user is hard pressed to reproduce these results and then evolve his/her own questions. The web site apparently identifies this data as freely available. Can it be included as supplementary material (as a Cytoscape session file?) to assist the user in following this paper?

7. In "Input datasets and pre-processing", I tracked down the referenced original PathLinker paper. It took a while to determine which network was being used. I downloaded it and imported it into Cytoscape. During the import, there were a number of options available, and it was unclear which options should be chosen. Can this network be included as supplementary material (as a Cytoscape session file?) to assist the user in following this paper?

8. In "Running PathLinker", can you explain the biological ramifications behind the $k=50$ and edge penalty settings?

9. In "Further Analysis", can you explain which Cytoscape tool or feature you used to spread the nodes apart? I'm thinking of the biological user that's trying to follow the paper.

10. In "Functional Enrichment", can you specify which ClueGO settings you used? This is a very valuable step, and it's hard for the user to follow without giving settings.

11. In "Running Time", how many CPU cores and how much RAM were on the test machine?

12. In the "Comparison to related Cytoscape Apps", the discussion focuses on differences in graph analysis approaches, and assumes the reader can appreciate the reasons why PathLinker gives
better results. The discussion could use a little more justification, and also some grounding in the biological consequences of these differences.

13. In the Introduction, the claim "any human signaling pathway" is overbroad. I suggest claiming "human signaling pathways".

**Competing Interests:** No competing interests were disclosed.

I have read this submission. I believe that I have an appropriate level of expertise to confirm that it is of an acceptable scientific standard, however I have significant reservations, as outlined above.