SOFTWARE TOOL ARTICLE

REVISED Copycat Layout: Network layout alignment via Cytoscape Automation [version 2; peer review: 2 approved]

Brett Settle[1], David Otasek[1], John H Morris[2], Barry Demchak[1]

[1]Department of Medicine, University of California, San Diego, California, 92093-0688, USA
[2]University of California San Francisco, San Francisco, California, 94143, USA

v2 **First published:** 21 Jun 2018, **7**:822
https://doi.org/10.12688/f1000research.15144.1

Latest published: 10 Aug 2018, **7**:822
https://doi.org/10.12688/f1000research.15144.2

Abstract

The copycatLayout app is a network-based visual differential analysis tool that improves upon the existing layoutSaver app and is delivered pre-installed with Cytoscape, beginning with v3.6.0. LayoutSaver cloned a network layout by mapping node locations from one network to another based on node attribute values, but failed to clone view scale and location, and provided no means of identifying which nodes were successfully mapped between networks. Copycat addresses these issues and provides additional layout options.
With the advent of Cytoscape Automation (packaged in Cytoscape v3.6.0), researchers can utilize the Copycat layout and its output in workflows written in their language of choice by using only a few simple REST calls. Copycat enables researchers to visually compare groups of homologous genes, generate network comparison images for publications, and quickly identify differences between similar networks at a glance without leaving their script. With a few extra REST calls, scripts can discover nodes present in one network but not in the other, which can feed into more complex analyses (e.g., modifying mismatched nodes based on new data, then re-running the layout to highlight additional network changes).

Keywords

Workflow, Reproducibility, Cytoscape, Interoperability, REST, Microservice, Layout, Alignment

This article is included in the Cytoscape gateway.

Open Peer Review

Reviewer Status ✓ ✓

	Invited Reviewers	
	1	**2**
version 2 (revision) 10 Aug 2018	✓ report	
version 1 21 Jun 2018	✓ report	✓ report

1. **Bernhard Mlecnik**, Cordelier Research Center, Paris, France
Inovarion, Paris, France

2. **Katja Luck**, Dana–Farber Cancer Institute, Boston, USA
Dana–Farber Cancer Institute, Boston, USA
Harvard Medical School, Boston, USA

Any reports and responses or comments on the article can be found at the end of the article.

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

Corresponding author: Barry Demchak (bdemchak@ucsd.edu)

Author roles: Settle B: Investigation, Software, Validation, Writing – Original Draft Preparation, Writing – Review & Editing; **Otasek D**: Methodology, Software, Validation, Writing – Review & Editing; **Morris JH**: Investigation, Methodology, Software; **Demchak B**: Conceptualization, Investigation, Methodology, Project Administration, Supervision, Writing – Review & Editing

Competing interests: No competing interests were disclosed.

Grant information: This work was supported with funding from the National Resource for Network Biology (NRNB) under award number P41 GM103504 and the National Institute of General Medical Sciences (NIGMS) under award number R01 GM070743, both assigned to Trey Ideker.
The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

How to cite this article: Settle B, Otasek D, Morris JH and Demchak B. **Copycat Layout: Network layout alignment via Cytoscape Automation [version 2; peer review: 2 approved]** F1000Research 2018, **7**:822 https://doi.org/10.12688/f1000research.15144.2

First published: 21 Jun 2018, **7**:822 https://doi.org/10.12688/f1000research.15144.1

REVISED **Amendments from Version 1**

This revision aims to address the comments of the reviewers. We added details to the Operation section on applying Copycat Layout from the Cytoscape Desktop interface, and added more future plans for handling mappings that are not one-to-one. The double quotes in the JSON code have also been changed to be consistent with the rest of the code sections.

See referee reports

Introduction

The copycatLayout app[1] (hereafter "Copycat") is an evolution of the existing layoutSaver app[2], a visual network aligner that maps node locations from one network view (called the source) to another (called the target). Copycat aims to enable Cytoscape users to compare and contrast networks by highlighting and arranging nodes not common to both networks, and by showing both networks using the same layout, scale and placement.

We upgraded Copycat to enable Cytoscape Automation[3] by exposing a new REST endpoint[4,5], thereby bringing the power of network-based differential analysis to automation scripts. Cytoscape Automation enables biologists to incorporate Cytoscape network visualization and analysis functionality (via REST calls) into workflows written in a language in which they are already productive (e.g., Python and R).

For example, as Cytoscape is called to transform a network (the *target*), it's often important to be able to show a correspondence to an original network (the *source*). The Copycat endpoint does this by invoking the Copycat layout to make common and disjoint nodes obvious and easily available to a workflow. In a workflow that performs multiple network transformations, Copycat can be used to pinpoint network differences after each transformation, thereby boosting confidence in the transformation and illuminating its real effect. Figure 1 illustrates a workflow that might call Copycat repeatedly to create a frame-by-frame movie of network manipulation.

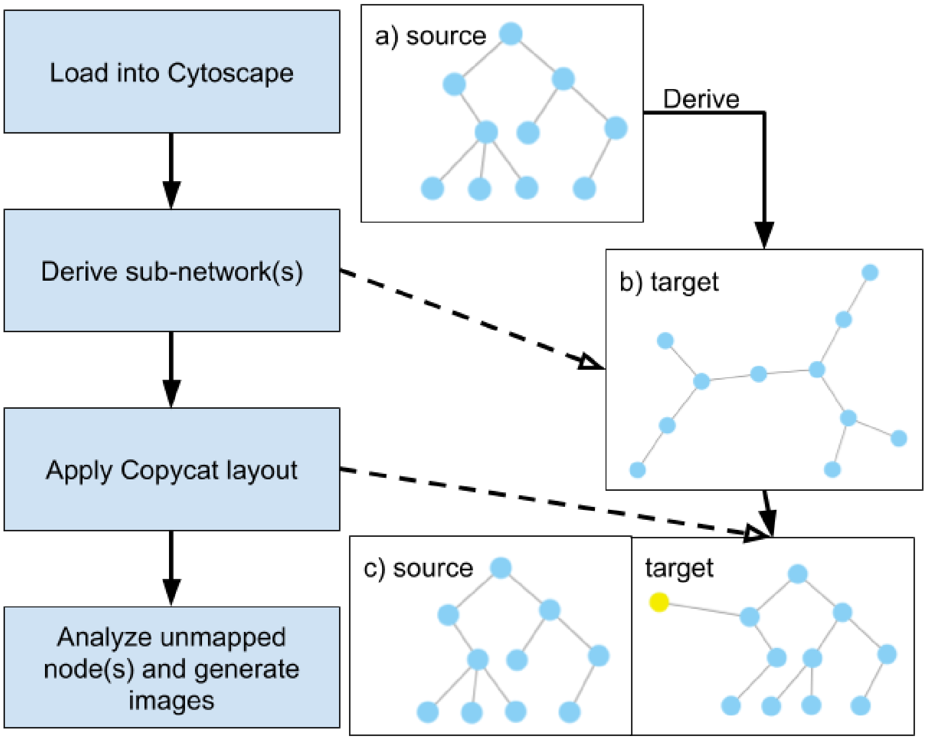

Figure 1. A workflow in which transformations are documented by applying Copycat. (a) The starting network is loaded into Cytoscape and a simple layout is applied. (b) Data-derived subnetwork. (c) Copycat layout is applied, mapping the starting network layout onto the subnetwork and selecting the unmapped node that was not found in the source.

As another example, applying Copycat layout to similar networks (i.e., rat, mouse or human) and mapping nodes by homology assignments can help verify and discover new homologous genes and gene pairs. Unmapped nodes might be useful in predicting gene positions based on successfully mapped neighbors.

In this paper, the Implementation section describes the general approach of the Copycat layout and its REST endpoint. The Operation section describes how to call the endpoint as either a Cytoscape Automation Function or Command. The Use Case section demonstrates the Copycat endpoint use in a real workflow, and the Discussion section describes the layout performance.

Methods

Implementation

The Copycat endpoint operates on two Cytoscape network views, where a source view contains a source network, and a target view contains a target network. The source view acts as a reference for laying out nodes in the target view.

Copycat matches nodes in a source view to nodes in a target view based on a common node attribute value. For each match, it sets the target node's X Location, Y Location and Z Location visual attributes to those of the source node. For each source node, the attribute value is found in a column of the source network table, and likewise for each target node. (While the columns themselves may be different, they are assumed to have the same type and meaning.)

Copycat implements a multi-pass algorithm where the first pass creates a map of source node attributes to source view coordinates. The second pass searches for target node attribute values in the source node attribute map and sets the target node's view coordinates when an attribute match is found.

A node attribute value in the source column is considered to match a value in the target column if they are lexicographically equal. Note that a poor choice of mapping attributes can lead to confusing layouts. For example, when multiple target nodes match a single source node, all of the matching target nodes will be stacked on top of each other at the coordinates of the source node. Also, if a target node matches multiple source nodes, only the last encountered source mapping is used. To avoid visual confusion, avoid having multiple nodes associated with the same attribute value. (A good choice for a mapping attribute is the node's name if each node has a different name. A bad choice could be a node's in-degree if multiple nodes have the same in-degree.)

Optionally, Copycat sets the *selected* attribute for source nodes that aren't mapped and for target nodes whose attributes aren't found in the source node map, as shown in Figure 2. (The caller can subsequently query and act upon selected source and target nodes using other CyREST endpoints). Also optionally, unmapped target nodes are laid out in a grid in the top right portion of the target view.

Operation

The Copycat Layout is accessible in the Cytoscape Desktop interface via the Layouts>Copycat Layout menu action, and is enabled if two or more networks exist in the network panel. For example, if the user has spent considerable time optimizing the layout of a network only to find that the data has changed, they can reload the data in the same Cytoscape session and use Copycat to copy the layout to the new network. To apply the layout, choose the source and target network views and their respective mapping columns (as defined in the *Implementation* section) and check boxes for selecting and gridding unmapped nodes if desired. The rest of this paper is intended for Cytoscape workflow programmers hoping to automate network view alignment via the Copycat layout algorithm.

Copycat exposes a single endpoint with both a Commands and Functions[6] variant, each with similar parameters. While both variants perform the same layout, Commands are most conveniently used in conjunction with the Cytoscape scripting facilities, and Functions are most convenient for calls from scripting languages such as Python and R.

Generally, the caller specifies the Cytoscape views containing the source and target networks (called **sourceViewSUID** and **targetViewSUID**) and provides the name of the mapping column in the source network (called **sourceColumn**) and target network (called **targetColumn**). It can control whether unmapped nodes are selected (called **selectUnmapped**) or the unmapped nodes are laid out in a grid (called **gridUnmapped**).

Note that the caller can determine columns available for a network via a CyREST endpoint, such as **/v1/networks/{networkId}/tables/defaultnode/columns**.

Note that the source and target columns must both be either string types or integer types.

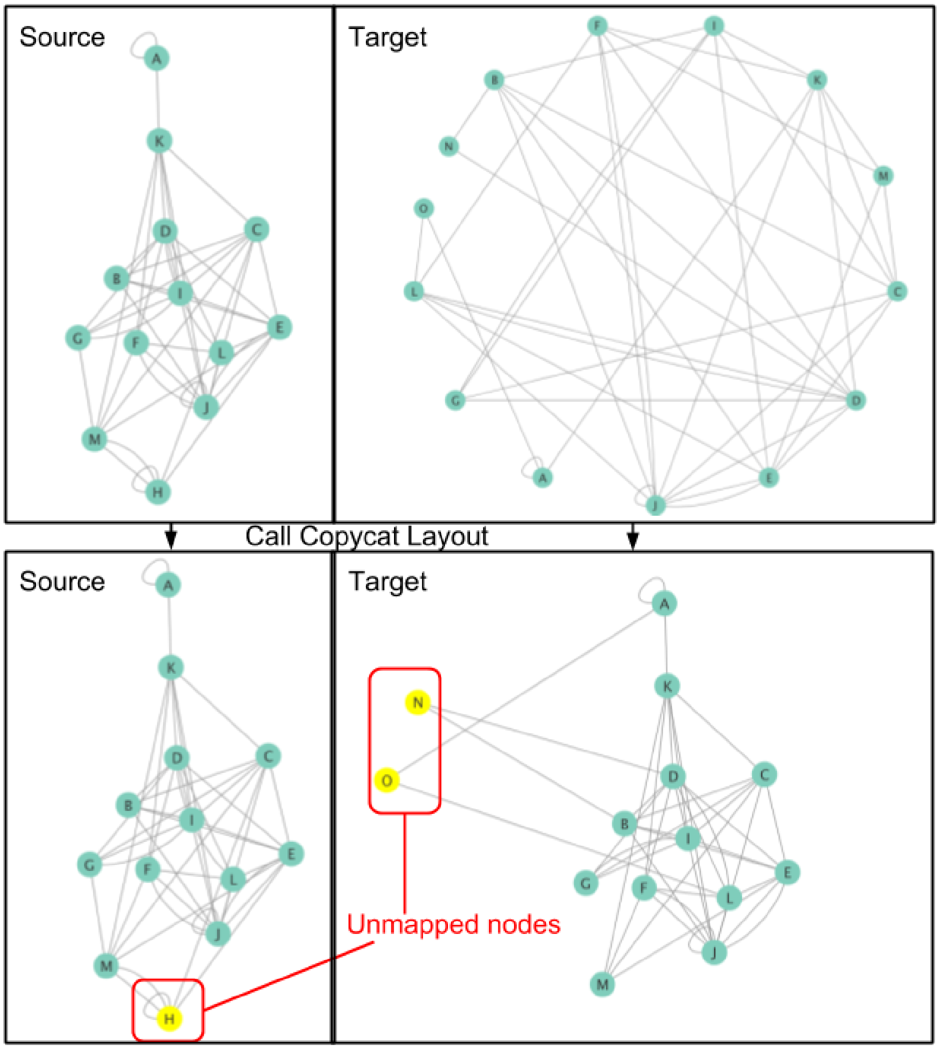

Figure 2. Calling Copycat layout on a derived subnetwork with the *selectUnmapped* parameter set to false. After the Copycat layout, the target network nodes have the same coordinates as the corresponding source network nodes. Because the node H in the source and nodes N and O in the target are unmatched, they are selected.

The Copycat endpoint returns a CIResponse[7] according to Cytoscape Automation best practices. If the call succeeds, the CIResponse contains a layout result object (as the **data** element); otherwise, it contains an explanation of the error (as the **errors** element):

```
{
  "data": {"mappedNodeCount": integer,
           "unmappedNodeCount": integer
  },
  "errors": []
}
```

The **mappedNodeCount** value contains the number of target nodes successfully mapped, and the **unmappedNodeCount** contains the number of target nodes that did not correspond to a source node. To calculate the number of unmapped source nodes, the user can execute a GET request at **/networks/{networkSUID}/nodes/selected** and get the length of the list returned.

If either of the network views cannot be resolved by the given SUIDs or the mapping columns cannot be found in the node table of the source and target network, the **errors[0].status** element returns 404, and the remainder of the **errors[0]** element contains additional information.

To apply Copycat Layout to a network, you must be running Cytoscape version 3.6.0 or later with at least 512MB of free memory.

Copycat as a Function. The Functions endpoint is documented in a Swagger page in the **Layouts** section available via **Help → Automation → CyREST API**.

The caller must pass the **SourceViewSUID** and **targetViewSUID** parameters as part of the URL (**/v1/apply/layouts/copycat/{sourceViewSUID}/{targetViewSUID}**), and the remaining parameters in a JSON payload object:

```
{
  "sourceColumn": string, // defaults to "name"
  "targetColumn": string, // defaults to "name"
  "selectUnmapped" : boolean,
  "gridUnmapped" : boolean
}
```

Note that unlike other layout REST endpoints, the Copycat Function requires an HTTP PUT request because it consumes a JSON payload with extra parameters -- the other CyREST layout endpoints require GET requests with optional parameters directly in the URL.

Note that the **sourceViewSUID** and **targetViewSUID** must be either integers or the word "current" (meaning the view currently selected in Cytoscape). The caller can determine a network view's SUID via a number of CyREST endpoints, including **/v1/networks/views/currentNetworkView**. The **sourceViewSUID** and **targetViewSUID** must reference different views.

Example code is provided in R, Python and as a Bash curl, but can easily be adapted into any language that supports REST calls. (Note that you must determine the values of sourceViewSUID and targetViewSUID independently ahead of these calls.)

R

```
# Basic settings for cyREST
port.number = 1234
base.url = paste("http://localhost:", toString(port.number), "/v1", sep="")

# Send it to Cytoscape!
copycat.url = paste(base.url, "apply", "layouts", "copycat",
sourceViewSUID, targetViewSUID, sep="/")
copycat.args = list(sourceColumn="name", targetColumn="name",
selectUnmapped=TRUE)
copycat.JSON = toJSON(copycat.args)
res <- PUT(url=copycat.url, body=copycat.JSON, encode="json")
```

Python

```
import requests, json
resp = requests.put("localhost:1234/v1/apply/layouts/copycat/{}/{}".
format(sourceViewSUID, targetViewSUID), data=json.dumps(data))
resp = resp.json()
```

Bash

```
curl -X PUT --header 'Content-Type: application/json' --header 'Accept:
application/json' -d '{ \
   "sourceColumn": "name", \
   "targetColumn": "name", \
   "selectUnmapped": true, \
   "gridUnmapped": false \
 }'
'http://localhost:1234/v1/apply/layouts/copycat/${sourceViewSUID}/
${targetViewSUID}'
```

Copycat as a Command. The Commands version is documented in the Swagger page reachable from Cytoscape's **Help → Automation → CyREST Command API** and can be found in the **layouts** section. Commands are executable via the Cytoscape Command Tool[8] as well as the CyREST Commands API.

The caller must pass the **sourceViewSUID** and **targetViewSUID** as part of the payload object instead of the URL. They must be the name of a network (e.g., "galFiltered.sif") instead of an SUID integer; Copycat will operate on the primary view for the network.

Note that like other layout Command REST endpoints, the Copycat Function requires an HTTP POST request.

Note that with the current CyREST Commands, it is difficult to query Cytoscape to find the name of a network. The caller must know the name in advance, likely as a result of creating or naming it by using a different Command prior to the Copycat call (e.g., **/v1/commands/network/create empty**).

Example code is provided in R, Python and as a Bash curl, but can easily be adapted into any language that supports REST calls. (Note that you must determine the values of **sourceNetworkName** and **targetNetworkName** independently ahead of these calls.)

R

```
# Basic settings for cyREST
port.number = 1234
base.url = paste("http://localhost:", toString(port.number), "/v1", sep="")

# Send it to Cytoscape!
copycat.url = paste(base.url, "commands", "layout", "copycat", sep="/")
copycat.args = list(sourceNetwork="sourceNetworkName", sourceColumn="name",
targetNetwork="targetNetworkName", targetColumn="name", selectUnmapped=TRUE)
copycat.JSON = toJSON(copycat.args)
res <- PUT(url=copycat.url, body=copycat.JSON, encode="json")
```

Python

```
import requests, json
resp = requests.put("localhost:1234/v1/commands/layout/copycat", data = json.
dumps(data))
resp = resp.json()
```

Bash

```
curl -X POST --header 'Content-Type: application/json' --header 'Accept:application/
json' -d '{\
   "gridUnmapped": "true", \
   "selectUnmapped": "true", \
   "sourceColumn": "name", \
   "sourceNetwork": "sourceNetworkName", \
   "targetColumn": "name", \
   "targetNetwork": "targetNetworkName" \
 }' 'http://localhost:1234/v1/commands/layout/copycat'
```

Additionally, the Copycat Command can be executed directly via the Cytoscape Command Tool or as part of a Cytoscape Command script as shown below, where each parameter is specified using a key-value pair. Optional parameters can be specified by appending additional key-value pairs.

layout copycat sourceNetwork={sourceNetworkName} targetNetwork={targetNetworkName}

Use case

Many biological analyses rely on performing differential analysis on derived subnetworks. In the case of inferring gene regulatory networks[9], it is beneficial to show mutations at each time-step without touching the unchanged parts of the network. Identifying and visualizing the changes over time is much easier with Cytoscape Automation. After loading networks into Cytoscape, the CyREST API can be used to fetch the SUID of networks and views, and the names of columns in the node table. What follows are some of the essential steps for this kind of analysis.

```
# Retrieve network and view SUID from Cytoscape in Python
REST_PORT='1234' # Set to whatever rest.port is in Cytoscape preferences
REST_ENDPOINT = "http://localhost:{}/v1".format(REST_PORT)

resp = requests.get(REST_ENDPOINT + "/networks")
network_suids = resp.json()
suid_map = {'source': {'suid': network_suids[0]}, 'target': {'suid':network_
suids[1]}}

resp = requests.get("{}/networks/{}/views".format(REST_ENDPOINT,
suid_map['source']['suid']))
suid_map['source']['view_suid'] = resp.json()[0] # only one view exists
```

Insert the network view SUIDs into the sample scripts listed above to apply the Copycat layout, and the CyREST API will return the number of mapped and unmapped nodes in the target layout. Once the layout has completed, add a GET request to **/networks/{SUID}/views/{viewSUID}.png** to generate images of the aligned networks. Adding the **selectUnmapped** parameter and a few calls to the core Cytoscape network API suite allows callers to take full advantage of the layout results by getting unmapped node information.

```
resp = requests.get("{}/networks/{}/nodes/selected".format(REST_ENDPOINT
networkId))
nodes = []
for nodeId in resp.json():
    resp = requests.get("{}/networks/{}/nodes/{}".format(REST_ENDPOINT,networkId,
nodeId))
    nodes.append(resp.json())
```

The complete working example of this workflow is available in a Jupyter notebook found in the copycat-layout Github repository at https://github.com/cytoscape/copycat-layout/blob/master/notebooks/Copycat%20Automation%20Example.ipynb.

Discussion

Using a hashmap to store node attributes and locations within solely the source network allows Copycat layout to run in O(N+M) time while only requiring O(N) memory, where N is the number of nodes in the source network and M is the number of nodes in the target network. Note that the layout performance is independent of the number of edges in either network, as edges are not accounted for in this algorithm.

It is also important to note that the Commands endpoint is automatically generated by Cytoscape Tunables and is meant to be used via Cytoscape Command Tools, whereas the CyREST Functions are crafted by the developer specifically for use in Python- and R-based automation scripts. For this reason, the Functions endpoint will be better documented, provide better errors, and accept more script-friendly parameters, such as SUIDs.

Future plans

The Copycat API follows the same Semantic Versioning[10] best practices established by Cytoscape and CyREST, and future modifications will respect the same parameters, functionality, and expected outputs specified in this paper. CopycatLayout aims to be an integral tool for layout alignment in Cytoscape, and intends to accommodate future expectations of alignment analysis without overcomplicating the interface. By integrating repetitive use cases into the API (e.g., returning the list of unmapped node SUIDs as well as their count), we can provide better tools to researchers and greatly reduce the complexity of automation scripts.

We plan to implement better handling of a non-unique mapping attribute column to provide the user with more control over node pairings that are not one-to-one. For example, we could provide a dialog for viewing and managing complex mappings. Another point of interest in visual alignments is identifying overlapping edges between networks. We hope to address both of these concerns in future versions of Copycat layout.

Summary

In this paper, we presented the Copycat layout, a network differential analysis app for Cytoscape 3. Copycat provides basic visual alignment functionality that will open the door to more data-centric alignment algorithms.

Copycat layout is also exposed via the Cytoscape CyREST API for automation users and scripted analyses. Copycat can be combined with other layouts and CyREST calls to better understand network transformations, and generate clean network difference figures.

Data availability

All data underlying the results are available as part of the article and no additional source data are required.

Software availability

The copycatLayout app is delivered as a component of Cytoscape starting with v3.6.0, and is also available on the Cytoscape App Store: http://apps.cytoscape.org/apps/copycatLayout.

Source code available from: https://github.com/cytoscape/copycat-layout.

Archived source code at time of publication: https://doi.org/10.5281/zenodo.1287307[11].

License: GNU Lesser General Public License v2.1.

Competing interests

No competing interests were disclosed.

Grant information

This work was supported with funding from the National Resource for Network Biology (NRNB) under award number P41 GM103504 and the National Institute of General Medical Sciences (NIGMS) under award number R01 GM070743, both assigned to Trey Ideker.

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

Acknowledgments

We would like to thank Justin Huang and Dan Carlin for being the first users of Copycat and for providing valuable feedback during the development process.

References

1. **Navigation and Layout — Cytoscape User Manual 3.6.0 documentation**. Reference Source
2. **Cytoscape App Store - layoutSaver**. [cited 17 Apr 2018]. Reference Source
3. **cytoscape-automation**. Github. Reference Source
4. Fielding RT, Taylor RN: **Principled Design of the Modern Web Architecture.** *ACM Trans Internet Technol.* New York, NY, USA: ACM; 2002; **2**(2): 115–150. Publisher Full Text
5. Ono K, Muetze T, Kolishovski G, *et al.*: **CyREST: Turbocharging Cytoscape Access for External Tools via a RESTful API [version 1; referees: 2 approved].** *F1000Res.* 2015; **4**: 478. PubMed Abstract | Publisher Full Text | Free Full Text
6. Cytoscape Automation FAQ: **What is the difference between Commands and Functions?** In: Google Docs. [cited 18 Apr 2018]. Reference Source
7. **App Developers: Cytoscape Function Best Practices - CIResponse.** [cited 18 Apr 2018]. Reference Source
8. **Command Tool — Cytoscape User Manual 3.6.0 documentation**. Reference Source
9. Carlin DE, Paull EO, Graim K, *et al.*: **Prophetic Granger Causality to infer gene regulatory networks.** *PLoS One.* 2017; **12**(12): e0170340. PubMed Abstract | Publisher Full Text | Free Full Text
10. Preston-Werner T: **Semantic Versioning 2.0.0**. In: *Semantic Versioning.* [cited 18 Apr 2018]. Reference Source
11. Settle B, dotasek, Morris S, *et al.*: **cytoscape/copycat-layout: Copycat Layout (Version v1.2.3).** *Zenodo.* 2018. http://www.doi.org/10.5281/zenodo.1287307

Open Peer Review

Current Peer Review Status: ✓ ✓

Version 2

Reviewer Report 10 August 2018

https://doi.org/10.5256/f1000research.17329.r36982

✓ **Bernhard Mlecnik**

[1] Laboratory of Integrative Cancer Immunology, Cordelier Research Center, Paris, France
[2] Inovarion, Paris, France

All OK for me.

Competing Interests: No competing interests were disclosed.

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

Version 1

Reviewer Report 09 July 2018

https://doi.org/10.5256/f1000research.16498.r35315

✓ **Katja Luck**

[1] Center for Cancer Systems Biology (CCSB), Dana–Farber Cancer Institute, Boston, MA, USA
[2] Department of Cancer Biology, Dana–Farber Cancer Institute, Boston, MA, USA
[3] Department of Genetics, Harvard Medical School, Boston, MA, USA

The article Copycat Layout: Network layout alignment via Cytoscape Automation describes an evolution of an existing Cytoscape App that allows the mapping of the layout of one network onto another network by matching the nodes in both networks using a common node attribute. As a frequent Cytoscape user, I have very much awaited for such a Cytoscape functionality to be developed as it clearly fills a gap.

For programmers and frequent users of the Cytoscape automation functions, the article is very understandable and provides all necessary information for judging whether it suits the user's needs and how to apply it. However, for non-computational users of Cytoscape, some more information could be provided helping them to know whether it solves their needs and how to use it. I might have missed it but I cannot locate information within the article or from a reference or link where to find information how and if the App can be used within Cytoscape without using the REST API or the command line tools of Cytoscape. If it is missing, it might add value to the article to provide such an example. For example, a common user scenario that I have already encountered many times in collaborations with biologists is that we had manually optimized the layout of a network (a very time-consuming procedure) to then later realize that the network data has changed again. This meant to reload the new network and start from scratch the manual layout optimization. I understand that with the Copycat App, the layout from the previous network (in this article corresponding to the source network) can be transferred to the new network (target network) solving the aforementioned problem. It might be helpful to many readers to quickly see from within the article or elsewhere in the documentation or tutorial how to solve this problem with this new App without using scripts.

The whole article is written in very technical terms. Even though for most of them references are provided, the article and software might be more accessible and therefore more used, especially by non-programmers, if efforts were taken to reduce some of the technicalities in the main text.

A suggestion for future improvements of the App is the possibility to highlight differences in links between the source and target network. A nice example for such a use case is shown in Figure 1 of the paper. The networks differ by one node as highlighted by the select tool, however, the networks also differ by some links connecting nodes that are present in both networks. These link differences can be easily missed by visual inspection, especially if the networks contains more nodes than in this example, but might be extremely valuable to spot or identify by the user. I think highlighting unique links to both networks would be a great plus.

Is the rationale for developing the new software tool clearly explained?

Yes

Is the description of the software tool technically sound?

Yes

Are sufficient details of the code, methods and analysis (if applicable) provided to allow replication of the software development and its use by others?

Partly

Is sufficient information provided to allow interpretation of the expected output datasets and any results generated using the tool?

Yes

Are the conclusions about the tool and its performance adequately supported by the

findings presented in the article?

Yes

Competing Interests: No competing interests were disclosed.

Reviewer Expertise: I am a computational biologist who has specialized in systems biology, network biology, and data integration. I have worked many times with Cytoscape using its interface as well as its programmatic access points.

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

Author Response 02 Aug 2018

Brett Settle, University of California, San Diego, USA

We would like to thank the referee for positive feedback and helpful comments. Below we acknowledge the notes mentioned in the review:

- This paper was published as part of a collection focused on Cytoscape Automation via CyREST, and so we did not mention applying the copycat layout via Cytoscape Desktop. A short section will be added at the beginning for non-REST users, and it will present the very use case you suggest.
- This is a very helpful note for future papers on Automation and we will take into account the technical terms that are used.
- We very much anticipate an interest in network comparison highlighting of non-matching edges as well as nodes. This will definitely be a feature in a future version of the app. Thank you for the very helpful use case.

Competing Interests: No competing interests were disclosed.

Reviewer Report 04 July 2018

https://doi.org/10.5256/f1000research.16498.r35312

✓ **Bernhard Mlecnik**

[1] Laboratory of Integrative Cancer Immunology, Cordelier Research Center, Paris, France

[2] Inovarion, Paris, France

Short summary

The Copycat Cytoscape layout plugin presented here maps the node locations of a source network to the node locations of a target network and handles missing nodes. Important is that the source node ids uniquely match the target node ids to avoid network inconsistencies after applying the layout.

The Copycat layout is a straightforward to use plugin. The application is simple but powerful and Cytoscape needs this kind of layout algorithm.

It is also important the the developers offer a REST enabled service to this plugin which is thoroughly described for different programming languages in the article.

The only point the authors could address is a more sophisticated way of handling node inconsistencies. E.g. give a name/id list of nodes that are not unique in the target network to inform the user about this problem instead of just overlaying these nodes on the target network.

Will the authors of this plugin consider edge information in the future? E.g. to give an option to map only nodes that are linked in the source and target network.

Is the rationale for developing the new software tool clearly explained?

Yes

Is the description of the software tool technically sound?

Yes

Are sufficient details of the code, methods and analysis (if applicable) provided to allow replication of the software development and its use by others?

Yes

Is sufficient information provided to allow interpretation of the expected output datasets and any results generated using the tool?

Yes

Are the conclusions about the tool and its performance adequately supported by the findings presented in the article?

Yes

Competing Interests: No competing interests were disclosed.

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

Author Response 02 Aug 2018

Brett Settle, University of California, San Diego, USA

We would like to thank the referee for the positive feedback and respond to the referee’s comments:

- In the current version, a non-unique attribute column used as the mapping variable should be handled better in future versions of the app. Currently we are deciding how best to deal with these mappings.

- We would be very interested in adding functionality to restrict layout mapping to edges instead of nodes, given use cases and/or expected results. These more difficult alignments will be considered for future releases.

Competing Interests: No competing interests were disclosed.