SOFTWARE TOOL ARTICLE

dot-app: a Graphviz-Cytoscape conversion plug-in [version 1; peer review: 2 approved, 2 approved with reservations]

Braxton Fitts¹, Ziran Zhang¹, Massoud Maher¹, Barry Demchak¹ ²

¹Department of Computer Science and Engineering, UC San Diego, San Diego, USA

²Department of Health Sciences, UC San Diego, San Diego, USA

V1 First published: 20 Oct 2016, 5:2543
<https://doi.org/10.12688/f1000research.9751.1>
 Latest published: 10 Jul 2017, 5:2543
<https://doi.org/10.12688/f1000research.9751.2>

Abstract

dot-app is a Cytoscape 3 app that allows Cytoscape to import and export Graphviz (*.dot, *.gv) files, also known as DOT files due to the *.dot extension and their conformance to the DOT language syntax. The DOT format was originally created in the early 2000s to represent graph topologies, layouts and formatting. DOT-encoded files are produced and consumed by a number of open-source graph applications, including GraphViz, Gephi, neato, smyrna, and others. While DOT-based graph applications are popular, they emphasize general graph layout and styling over the topological and semantic analysis functions available in domain-focused applications such as Cytoscape. While domain-focused applications have easy access to large networks (10,000 to 100,000 nodes) and advanced analysis and formatting, they do not offer all of the styling options that DOT-based applications (particularly GraphViz) do. dot-app enables the interchange of networks between Cytoscape and DOT-based applications so that users can benefit from the features of both. dot-app was first deployed to the Cytoscape App Store in August 2015, has since registered more than 1,200 downloads, and has been highly rated by more than 20 users.

Keywords

Network , import , export , format conversion , attribute conversion , data visualization , Cytoscape , GraphViz , DOT

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

Open Peer Review

Approval Status

	1	2	3	4
version 2				
(revision)
10 Jul 2017		view	view

version 1
20 Oct 2016	view	view	view	view

1. **Eric Bonnet** , CEA (French Alternative Energies and Atomic Energy Commission), Evry, France
2. **Matthias König** , Humboldt University of Berlin, Berlin, Germany
3. **David J. Lynn**, South Australian Health and Medical Research Institute, Adelaide, Australia
John Salamon, South Australian Health and Medical Research Institute, Adelaide, Australia
4. **Giovanni Scardoni**, University of Verona, Verona, Italy

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

This article is included in the **Bioinformatics** gateway.

Corresponding authors: Braxton Fitts (bdtfitts@outlook.com), Ziran Zhang (ziz025@ucsd.edu), Massoud Maher (mahermassoud@gmail.com), Barry Demchak (idekerlab.bdemchak@gmail.com)

Competing interests: No competing interests were disclosed.

Grant information: This material is based upon work supported by the National Institutes of Health under Grant P41 GM103504. The grant is assigned to Dr. Trey Ideker at the University of California, San Diego (UCSD). All work was performed at UCSD. *The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.*

Copyright: © 2016 Fitts B *et al.* This is an open access article distributed under the terms of the [Creative Commons Attribution License](#), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

How to cite this article: Fitts B, Zhang Z, Maher M and Demchak B. **dot-app: a Graphviz-Cytoscape conversion plug-in [version 1; peer review: 2 approved, 2 approved with reservations]** F1000Research 2016, 5:2543 <https://doi.org/10.12688/f1000research.9751.1>

First published: 20 Oct 2016, 5:2543 <https://doi.org/10.12688/f1000research.9751.1>

Introduction

Cytoscape¹ is a popular tool for visualizing and analyzing networks used in scientific and commercial analysis, most commonly in bioinformatics. It enables users to discover and load curated and uncurated networks representing molecular and genomic interactions, load ad-hoc or custom networks, and share networks that others have created. Once networks are loaded, users can manually annotate a network or automatically integrate annotations using a number of algorithms and databases. Users can perform a number of graph-oriented and semantic-aware analyses ranging from graph statistics to motif and cluster discovery to upstream and downstream structural and functional inferences. Users can also perform a number of complex graph filtering and layout operations to drive and focus the semantic understanding of network interactions and structure.

Even beyond analysis and layout, users commonly derive and demonstrate network meaning by using visual cues to distinguish relationships and attributes. For this, Cytoscape provides a visual style system that enables users to paint nodes and edges using color, border thickness, size, fonts, arrows, and other devices.

Much of the power and functionality of Cytoscape is delivered as apps available in the Cytoscape App Store (<http://apps.cytoscape.org>). The store contains nearly 300 apps that provide a range of functionality from file import/export to analysis to visualization and publishing. Based on the success of the combination of Cytoscape core and downloadable apps, Cytoscape is downloaded approximately 14,000 times per month worldwide and is started approximately 3,000 times each working day. As of 2015, Cytoscape has been cited in 700 academic peer-reviewed papers per year.

While Cytoscape is the dominant network analysis and visualization platform in bioinformatics, it is not the only platform. To support interoperability with a number of network-oriented workflows and applications, Cytoscape offers a number of natively supported file import/export modules, and leverages a number of them that are available as apps in the App Store. Some of these file formats² do not include visual information, such as SIF (.sif) and NNF (.nnf), whereas others do (e.g., GraphML's .graphml and XGMML's .xgmml formats).

Graphviz is a popular, well-established graph visualization application that produces a DOT (.dot, .gv) file containing graph structure, layout, and styling information, but for which there is no input/export module for Cytoscape. DOT files adhere to the DOT language syntax (<http://www.graphviz.org/doc/info/lang.html>). In comparison to XGMML and GraphML, Graphviz and its DOT language syntax define more visual attributes and support more visual features, such as edges composed of colored segments and single edges that are represented as multiple edges of different

colors. Some of the DOT attributes can be used only by the Graphviz software because they specify parameters for the layout algorithms that the software uses.

DOT files contain a number of visual attributes that map well to Cytoscape visualization functionality, and vice versa. However, incompatibilities do exist where some Cytoscape features cannot be represented in DOT, or where DOT represents some features that cannot be realized in Cytoscape. These incompatibilities are described in “Conversion details” section.

Note that GraphViz is one of several applications that produces or consumes DOT files, but it is by far the most commonly used. In this paper, we use “DOT file/network” and “GraphViz file/network” interchangeably.

We present dot-app as a Cytoscape app that implements both the import and export of graphs encoded in DOT files. We describe the operation of dot-app; how dot-app maps Cytoscape networks to DOT networks and vice versa; issues that arise because of incompatibilities between the Cytoscape and DOT network models; representative use cases; and prospects for future work.

Operation

Minimum system requirement

Dot-app requires Java 7 or above and Cytoscape v3.2 or above.

Import

A Graphviz network can be imported in two ways: from the welcome screen (via the From Network File... button), from the menu (“File->Import->Network->File...”), or from the toolbar (by clicking the “Import Network from File” button).

Users are presented with a file browser dialog titled “Network file to load” (as in [Figure 1](#)). The user is able to filter the dialog to display only Graphviz files by selecting Graphviz files (*.gv, *.dot) from the drop-down menu for “Files of Type.” Note that no difference exists between a Graphviz file with an extension of .dot and a Graphviz file with an extension of .gv. However, the .gv extension is preferred because versions of Microsoft Word also use the .dot extension (<https://marc.info/?l=graphviz-devel&m=129418103126092>). From this point, importing a Graphviz network is the same as importing a network from any of Cytoscape's accepted file formats. Those steps are detailed in the Cytoscape User Manual (<http://manual.cytoscape.org/en/stable/>).

Export

To export a Cytoscape network as a GraphViz network, use the “Export -> Network and View” menu. (Using “Export -> Network” is also possible, but this will result in a Graphviz file that contains no visual information and a notification to use “Export -> Network and View” instead.)

Selecting “GraphViz files (*.dot,*.gv)” in the Export dialog launches dot-app and prompts the user to choose from three options, as shown in [Figure 2](#) below. The purposes of these options are explained in the following section.

Set Parameters prompt

Pick edge style. Cytoscape provides edge-routing capabilities that cannot be conserved during the export process, so dot-app provides

three edge routing options: “Straight segments,” “Curved segments” and “Curved segments routed around nodes.” These options change the value of the “splines” attribute that appears in the exported Graphviz file. The Graphviz file for a network exported from Cytoscape is shown below, and the attribute modified by the “Pick Edge Style” option is underlined and in bold. [Figure 3](#), [Figure 4](#), and [Figure 5](#) depict pictures of the network with each edge style chosen.

Figure 1. Network file to load dialog with Graphviz files selected.

Figure 2. Set Parameter pop-up prompts users with three options.

Figure 3. The network exported with the “Straight segments” option, as “splines = ‘false’” in the output Graphviz file.

Figure 4. The network exported with the “Curved segments” option, as “splines = ‘curved’” in the output Graphviz file.

Figure 5. The network exported with the “Curved segments routed around nodes” option, as “splines = ‘true’” in the output Graphviz file.

```
graph example {
  bgcolor = "#FFFFFFF"
  splines = "false"
  outputorder = "edgesfirst"
  esep = "0"
  pad = "2"
  node [label = "",penwidth = "0.000000",
    height = "0.486111",width = "1.041667",
    tooltip = "",color = "#CCCCCFF",fillcolor =
    "#89D0F5FF",shape = "rectangle",style =
    "solid,rounded,filled",fontname = "SansSerif.
    plain",fontsize = "12",fontcolor =
    "#000000FF",fixedsize = "true",labelloc = "c"]
  edge [label = "",penwidth = "2.000000",
    tooltip = "",arrowhead = "none",arrowtail =
    "none",color = "#848484FF",fontname =
    "Dialog.plain",fontsize = "10",fontcolor =
    "#000000FF",style = "solid",dir = "both"]
  "Node 1$64" [label = "Node 1",pos =
    "-243.000000,42.000000"]
  "Node 2$66" [label = "Node 2",pos =
    "-70.975037,42.014648"]
  "Node 3$68" [label = "Node 3",pos =
    "-159.020569,-41.005199"]
  "Node 4$70" [label = "Node 4",pos =
    "-243.001038,-123.001381"]
  "Node 1$64" -- "Node 2$66"
  "Node 2$66" -- "Node 4$70"
  "Node 1$64" -- "Node 3$68"
}
```

Pick node label location. Graphviz does not offer the flexible label placement that Cytoscape offers. As such, dot-app gives the options of “Center,” “Top,” “Bottom,” and “External” to allow the user to specify the label location applied to every node. In the output Graphviz file, the “Center,” “Top,” and “Bottom” options change the value of the “labelloc” that appears in the node default attribute list. The options respectively change the value to “c,” “t,” and “b”. In contrast, the “External” option causes the node labels to set the “xlabel” attribute instead of the “label” attribute in the output Graphviz file. The “xlabel” attribute causes the label to be placed in a location near its node that does not cause it to overlap with any other nodes or labels. [Figure 6](#) shows a network exported with the “External” option.

Pick network label location. dot-app provides the options “No network label,” “Top,” and “Bottom” to allow the user to specify whether the network itself should be labeled and, if so, where the label is placed. The options “Top” and “Bottom” cause the “labelloc” attribute and “label” attribute for the graph to be written to the output Graphviz file. Furthermore, the “label” attribute will be set to the network’s name in Cytoscape. In contrast, the “No network label” option omits both the “labelloc” attribute and the “label” attribute.

Figure 6. A network exported with the “External” option for node label location.

Implementation

Import

For the import function of dot-app, we used Java-based Parser for Graphviz Documents (JPGD), a Graphviz document parser made by Alexander Merz(5). A Graphviz file typically contains information about the nodes, edges, and subgraphs (including annotations) for the layout. JPGD is a parser that transforms such a description

into a data structure. We used it to create Java objects modeling the graph, its nodes, and its edges. Each model object contains related DOT attributes represented as key-value pairs. **Figure 7** provides a high-level picture of the conversion of a DOT node declaration to the Node object that JPGD created. Detailed information about the JPGD objects can be found on JPGD’s website (<http://www.alexander-merz.com/graphviz/doc.html>).

After JPGD creates the model objects, dot-app creates a corresponding Cytoscape model, including a CyNetwork object, CyNode objects, and CyEdge objects. These associations are stored in maps: one map for each type of graph component. When the network view is being built in Cytoscape, our Reader objects use these associations to create the Cytoscape View objects. Three Reader classes exist: NetworkReader, NodeReader, and EdgeReader. At the start of the network view creation, a VisualStyle object is created for the network. Each Reader object uses the VisualStyle to set the default attributes for its class of graph components. In addition, Reader iterates through the corresponding association map to create the View objects for the graph components and to set their VisualProperties. **Figure 8** shows the high-level relationships among the JPGD Graph objects, the Cytoscape Graph objects, and the Cytoscape View objects.

Figure 7. DOT Node Declaration to JPGD Node Object.

Figure 8. Relationships between JPGD objects, Cytoscape objects and Cytoscape view objects.

After the View objects are created, the DOT attributes and their assigned values are converted into their Cytoscape equivalents, and the resulting VisualProperty and VisualPropertyValue are assigned to the View. If the DOT attribute's assigned value does not have an equivalent Cytoscape VisualPropertyValue, the VisualProperty is set to a default VisualPropertyValue.

Export

We created three classes—NodePropertyMapper class, EdgePropertyMapper class and NetworkPropertyMapper class—to accomplish the export function of dot-app. Each Mapper class contains an ArrayList into which the Mapper classes insert the DOT attribute strings for easily convertible Cytoscape VisualProperties. In addition, each Mapper class has unique helper methods that create the DOT attribute strings for the DOT attributes that have values determined by multiple Cytoscape VisualProperties. One such attribute is the “style” DOT attribute, the value of which is determined by NODE_SHAPE, NODE_BORDER_LINE_TYPE, and NODE_VISIBLE. The NodePropertyMapper class handles the conversion of the CyNodes and their VisualProperties into their DOT string equivalents. The EdgePropertyMapper class handles the conversion of the CyEdges and their VisualProperties into their DOT string equivalents. Finally, the NetworkPropertyMapper class handles the conversion of the CyNodes' and CyEdges' default VisualProperties and the CyNetwork's VisualProperties into their DOT string equivalents.

Conversion details

Import

Supported DOT attributes. The following DOT attributes contribute to the Cytoscape network during the import process. Most of the DOT attributes listed below correspond to a single Cytoscape visual property, but a few affect multiple visual properties (e.g., the “style” DOT attribute, as described below) at once due to the fact that the information is stored differently between the DOT model and the Cytoscape model. The “weight” DOT attribute is imported as an Edge table attribute (i.e., data) because no corresponding Cytoscape visual property exists. All other DOT attributes are ignored during the import process and have no effect on the visualization in Cytoscape.

Node DOT attributes. Table 1 lists the DOT attributes that can apply to nodes and the specific Cytoscape visual properties to which they map. The “pos” attribute maps to both NODE_X_POSITION and NODE_Y_POSITION because the value of the “pos” attribute is a coordinate pair of the form “x, y”.

Edge DOT attributes. Table 2 lists the DOT attributes that apply to edges and the specific Cytoscape visual properties to which they map.

Table 1. DOT node attributes and their corresponding Cytoscape visual properties.

DOT attribute	Cytoscape visual property
label	NODE_LABEL
xlabel	NODE_LABEL
color	NODE_BORDER_PAINT
fillcolor	NODE_FILL_COLOR
penwidth	NODE_BORDER_WIDTH
width	NODE_WIDTH
height	NODE_HEIGHT
shape	NODE_SHAPE
fontname	NODE_LABEL_FONT_FACE
fontsize	NODE_LABEL_FONT_SIZE
fontcolor	NODE_LABEL_FONT_COLOR
style	See section “The ‘style’ DOT Attribute”
pos	NODE_X_POSITION, NODE_Y_POSITION
tooltip	NODE_TOOLTIP

Table 2. DOT edge attributes and their corresponding Cytoscape visual properties.

DOT attribute	Cytoscape visual property
label	EDGE_LABEL
xlabel	EDGE_LABEL
weight	No Visual Property
color	EDGE_UNSELECTED_PAINT
penwidth	EDGE_WIDTH
fontname	EDGE_LABEL_FONT_FACE
fontsize	EDGE_LABEL_FONT_SIZE
fontcolor	EDGE_LABEL_FONT_COLOR
style	See section “The ‘style’ DOT Attribute”
arrowhead	EDGE_TARGET_ARROW_SHAPE
arrowtail	EDGE_SOURCE_ARROW_SHAPE
tooltip	EDGE_TOOLTIP

The “style” DOT attribute. The “style” DOT attribute applies to both nodes and edges. The attribute takes a comma-separated list of keywords as its value. These keywords directly affect which Cytoscape visual properties are modified. [Table 3](#) lists the keywords that dot-app supports, the graph components they affect and the Cytoscape visual properties to which the keywords map.

The “weight” DOT attribute. During the importing of a network using dot-app, a weight column is added to the Cytoscape network’s edge table. If the “weight” attribute is supplied for an edge, its value is assigned to the weight column entry for the edge.

Unsupported DOT features. The following features of Graphviz are not supported in the import:

1. Any HTML
2. Subgraphs
3. Clusters
4. Edges that are rendered as colored parallel lines: These are made by assigning a color list without weights to the “color” attribute. [Figure 9](#) depicts an example edge rendered in this manner.
5. Edges that are rendered as colored segments in series. These are made by assigning a color list with weights to the “color” attribute. [Figure 10](#) depicts an example edge rendered in this manner.
6. Gradients applied to the network background

Export

Unsupported Cytoscape features. When exporting the network as a GraphViz file, some Cytoscape information is lost because it cannot be represented in DOT format. dot-app does not keep a log

Table 3. The keywords for the DOT style attribute, the graph element that it can affect, and the Cytoscape visual properties that are modified as a result (the “rounded” keyword only affects NODE_SHAPE if the shape=”rectangle”).

“Style” attribute keyword	Affects nodes or edges	Cytoscape visual property
solid	Nodes/Edges	NODE_BORDER_LINE_TYPE EDGE_LINE_TYPE
dashed	Nodes/Edges	NODE_BORDER_LINE_TYPE EDGE_LINE_TYPE
dotted	Nodes/Edges	NODE_BORDER_LINE_TYPE EDGE_LINE_TYPE
invis	Nodes/Edges	NODE_VISIBLE EDGE_VISIBLE
rounded	Nodes	NODE_SHAPE*
filled	Nodes	NODE_TRANSPARENCY

Figure 9. Example of an edge rendered as colored parallel lines.

Figure 10. Example of an edge rendered as colored segments in series.

of the information that is not transferred to the Graphviz file. The following information is lost:

Visual information

1. Custom graphics
 - a. Images on nodes
 - b. Charts on nodes
2. Edge bends
3. Nested network images contained in nodes
4. Arrowhead colors (they will appear the same color as the edge itself)
5. Certain line types
 - a. Dash dot
 - b. Contiguous arrow

- c. Backward slash
- d. Separate arrow
- e. Sinewave
- f. Vertical slash
- g. Zigzag
- h. Forward slash
- i. Parallel lines
- 6. Label positioning
 - a. Edge labels only go on the midpoint of the edges
 - b. Node label positions are selected at export
- 7. The V node shape
- 8. Target arrow shape
 - a. Target arrow shape does not appear if set as a default; it only appears if set as a bypass
- 9. All annotations

Non-visual information

- 1. Node group information (groups are treated as a single node with no additional data)
- 2. All table data

Use cases

Detailed below are two cases for dot-app. The first use case describes how a DOT file can be imported into Cytoscape. The second use case describes how a Cytoscape network can be exported as a DOT file.

Import

Our first use case details how we would use dot-app to view a Graphviz-created network in Cytoscape. We used Graphviz's neato utility to create a DOT file with layout information and a PNG of the resulting network. The DOT file is shown below, and Figure 11 is the created PNG.

Figure 11. The PNG generated using Graphviz's neato utility.

```
graph toy_example {
  graph [bb="-85.648,-58.068,63.891,73.497",
  outputorder=edgesfirst, overlap=false];
  node [fillcolor="#888888",label="\N",style=filled];
  1      [height=0.5,pos="-58.648,-8.4777",width=0.75];
  2      [height=0.5,pos="36.891,3.383",width=0.75];
  2 -- 1      [pos="10.278,0.079128 -2.8626,
-1.5522 -18.68,-3.5159 -31.846,-5.1504"];
  3      [height=0.5,pos="12.665,-40.068",width=0.75];
  3 -- 1      [pos="-9.8989,-30.072 -18.223,-26.385
-27.653,-22.208 -35.986,-18.516"];
  3 -- 2      [pos="22.24,-22.895 23.933,-19.858
25.695,-16.698 27.386,-13.665"];
  4      [height=0.5,pos="8.8474,55.497",width=0.75];
  4 -- 2      [pos="18.03,38.433 21.097,32.734
24.516,26.38 27.592,20.664"];
  4 -- 3      [pos="9.5835,37.071 10.264,20.041
11.269,-5.1139 11.944,-22.022"];
}
```

Figure 12 shows the result of the import into Cytoscape version 3.4. The differences that arise between the Graphviz network and the Cytoscape network stem from how the two programs handle implicit default values. If a DOT attribute is omitted from the DOT file when using a Graphviz utility, an implicit default value for that attribute is used. The list of DOT attributes and their default values can be found on the Graphviz website (<http://www.graphviz.org/content/attrs.html>). Moreover, if a Cytoscape-compatible DOT attribute is not specified in the DOT file during import, Cytoscape supplies default values for the Cytoscape VisualProperties to which the missing DOT attribute maps. These default values come from Cytoscape's default visual style. This is why the nodes appear as ellipses in the PNG created using neato and as rounded rectangles in Cytoscape. These implicit defaults cause three apparent differences between the two results. The first difference is the node shapes, the second difference is the border colors for the nodes, and the third difference is the font used for the labels.

Figure 12. The DOT network as it looks in Cytoscape.

Export

In this second use case, we export the network in [Figure 12](#) from Cytoscape. The output file is shown below. [Figure 13](#) shows the PNG created by using Graphviz's neato utility on the output file. With this DOT file, we are able to use Graphviz and other programs that accept DOT files, such as NetworkX and PyGraphviz.

```
graph toy_example {
  bgcolor = "#FFFFFF"
  splines = "false"
  outputorder = "edgesfirst"
  esep = "0"
  pad = "2"
  node [label = "\N",penwidth = "0.000000",
  height = "0.486111",width = "1.041667",
  tooltip = "",color = "#CCCCFF",fillcolor =
  "#888888FF",shape = "rectangle",style =
  "solid,rounded,filled",fontname =
  "SansSerif.plain",fontsize = "12",fontcolor =
  "#000000FF",fixedsize = "true",labelloc = "c"]
  edge [label = "",penwidth = "2.000000",
  tooltip = "",arrowhead = "none",
  arrowtail = "none",color =
  "#848484FF",fontname = "Dialog.plain",
  fontsize = "10",fontcolor = "#000000FF",
  style = "solid",dir = "both"]
  "4$62" [label = "4",height = "0.500000",
  width = "0.750000",pos = "8.847400,
  55.497000"]
  "3$63" [label = "3",height = "0.500000",
  width = "0.750000",pos = "12.665000,
  -40.068000"]
  "2$64" [label = "2",height = "0.500000",
  width = "0.750000",pos = "36.891000,
  3.383000"]
  "1$65" [label = "1",height = "0.500000",
  width = "0.750000",pos = "-58.648000,
  -8.477700"]
  "4$62" -- "3$63" [color = "#404040FF"]
  "4$62" -- "2$64" [color = "#404040FF"]
  "3$63" -- "2$64" [color = "#404040FF"]
  "3$63" -- "1$65" [color = "#404040FF"]
  "2$64" -- "1$65" [color = "#404040FF"]
}
```

Again, slight differences exist in the network's appearance: the node shape and the labels' font. The variation in the node shape is due to how Graphviz and Cytoscape render rounded rectangles different. The difference in the labels' font is due to the font chosen in Cytoscape. In the output file shown above, we can see that the fontname attribute for the node default list is set to "SansSerif.plain". This font is not an actual font family; rather, it is one of Java's logical fonts (<https://docs.oracle.com/javase/tutorial/2d/text/fonts.html#logical-fonts>). It is a font name that the Java Runtime Environment used and that maps to a physical font. When neato encounters the font name, it attempts to find an

Figure 13. The PNG of the exported DOT file created with Graphviz's neato utility.

actual font named "SansSerif.plain"; if it cannot find one, it uses a default font.

Testing

We verified the dot-app import and export functions separately.

For import, we downloaded DOT files from Graphviz's gallery page (<http://graphviz.org/Gallery.php>) and wrote our own DOT files. We then ran Graphviz's neato utility on these files to generate DOT files that contained layout information and PNG files to use as references. We then imported the DOT files to Cytoscape and visually compared the Graphviz-created PNG files to the Cytoscape display to validate the import process of dot-app.

For export, we loaded each of the Cytoscape test session files (<https://github.com/cytoscape/cytoscape-tests/blob/master/docs/Session-Files/Session%20Files.md>) into Cytoscape and exported them to DOT files, which were then loaded into GraphViz. We visually compared the GraphViz display to the Cytoscape display to determine the correctness and completeness of the Cytoscape-to-DOT translation.

Conclusion

This article describes the dot-app Cytoscape app, which enables a user to import a DOT-formatted into Cytoscape and to export a Cytoscape network to a DOT-formatted file. We demonstrated the operation of dot-app and explained its implementation and the limitations of DOT-to-Cytoscape and Cytoscape-to-DOT translation. Finally, we explained typical use cases and how dot-app delivers value in each situation.

Whereas the dot-app conversions are diligent and true to Cytoscape 3.3 (with limitations), we recognize that future versions of Cytoscape may introduce new visual effects (e.g., new arrow heads) that present opportunities for rendering DOT files more truly or for the loss of formatting information if the DOT format cannot represent them. These issues will be dealt with in future versions of dot-app.

Software availability

Software available from:

<http://apps.cytoscape.org/apps/dotapp>

Latest source code:

<https://github.com/idekerlab/dot-app>

Archived source code as at the time of publication:

<http://doi.org/10.5281/zenodo.1596373>³

License:

GNU General Public License

and implementing dot-app. BD is the principal investigator for this dot-app project.

Competing interests

No competing interests were disclosed.

Grant information

This material is based upon work supported by the National Institutes of Health under Grant P41 GM103504. The grant is assigned to Dr. Trey Ideker at the University of California, San Diego (UCSD). All work was performed at UCSD.

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

Author contributions

BF, ZZ, and MM were involved in writing the article. BF, ZZ, and BD were involved in revising the draft manuscript and have agreed on the final content. BF, ZZ, and MM were involved in designing

Acknowledgements

Keiichiro Ono and Christian Zmasek helped with identifying the Cytoscape VisualProperties during the dot-app development.

References

1. Shannon P, Markiel A, Ozier O, *et al.*: **Cytoscape: a software environment for integrated models of biomolecular interaction networks**. *Genome Res.* 2003; **13**(11): 2498–504.
[PubMed Abstract](#) | [Publisher Full Text](#) | [Free Full Text](#)
2. Roughan M, Tuke J: **Unravelling graph-exchange file formats**. arXiv preprint arXiv:1503.02781. 2015.
[Reference Source](#)
3. Fitts B, Zhang Z, Maher M: **dot-app**. *ZENODO*. 2016.
[Data Source](#)

Open Peer Review

Current Peer Review Status:

Version 1

Reviewer Report 20 December 2016

<https://doi.org/10.5256/f1000research.10513.r17222>

© 2016 Scardoni G. This is an open access peer review report distributed under the terms of the [Creative Commons Attribution License](#), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Giovanni Scardoni

Center for Biomedical Computing (CBMC), University of Verona, Verona, Italy

The article introduces an app to import/export Graphviz (.gv-.dot) files to/from Cytoscape. Since the Graphviz format is of wide usage, the app is very useful to the community.

The article is well-written and the user-guide is complete and easy to read.

As minor comments I ask more details about which characteristics can be lost passing from a format to another. I tried some network choosing them randomly from internet, and sometimes I lost features that I didn't expect to lose.

A table explaining which features are preserved and which not will be very useful in the paper. If there are problem of space maybe the implementation details can be described in supplementary materials and not in the main paper.

The table should contains also short comments about the missing features that can be added in the future, and the problem that can be encountered implementing those characteristics that are not of immediate conversion.

I think that with these updates the paper is suitable for indexing.

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 22 Dec 2016

Ziran Zhang

Thank you Dr. Scardoni for your comment. We will take your valuable suggestions into

consideration while we are refining the article. We appreciate your effort in examining the dot-app and the article for us in detail.

Competing Interests: No competing interests were disclosed.

Reviewer Report 08 November 2016

<https://doi.org/10.5256/f1000research.10513.r17125>

© 2016 Lynn D et al. This is an open access peer review report distributed under the terms of the [Creative Commons Attribution License](#), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

David J. Lynn

European Molecular Biology Laboratory (EMBL) Australia Group, South Australian Health and Medical Research Institute, Adelaide, SA, Australia

John Salamon

European Molecular Biology Laboratory (EMBL) Australia Group, South Australian Health and Medical Research Institute, Adelaide, SA, Australia

dot-app is a Cytoscape App that enables the inter-conversion of Graphviz and Cytoscape formatted files. As far as we are aware no similar application exists and this is therefore a potentially useful development.

We found no problems with the usage of the app; it integrates well into Cytoscape, and can convert back and forth between formats without issue aside from the loss of information which is documented in the article. The code is readable and very well commented.

We did have an issue when compiling from source, however. The DOT parsing jar from Alex Merz isn't being included properly when Maven builds the project.

The major concern we had was that the concept for the application is really very straight-forward and following on from this, its functionality is therefore pretty limited. We wondered initially whether there was really a need for such an application, however, dot-app has been downloaded >1200 times so there is obviously a market for the tool. A remaining question, however, is whether the app justifies a 10-page paper? The paper itself reads a bit more like documentation and we would have preferred to have seen a more concise paper introducing the concepts together with more detailed documentation.

We think that the paper could also benefit from a better discussion in the introduction of the motivation behind the development of dot-app. Can you give examples where one would want to create a network in one format and then convert it to another? How hard would it be to manually re-create the visualisation in Graphviz/Cytoscape?

The application also appears very reliant on the parser developed by Alex Merz (this is not

referenced properly). It would be good to clarify how much of the functionality is provided by this parser and how much is extended by dot-app.

A key limitation of the app is that many of features of either format are lost when converting between them. This is because there isn't an equivalent feature in the other format. It is good that the authors are clear about which ones cannot be converted but this problem still means that manual intervention would likely be needed to customise the desired style.

One thing that could be improved is how the implicit default values are handled by dot-app. Couldn't you set reasonable values for these if they are not specified in a particular visualisation to enable more faithful conversion of the visualisation style?

One final concern is that it is very likely that the visual styles in both Graphviz and Cytoscape will continue to evolve over time. How will dot-app keep pace with these changes? Have the authors considered trying to get the Graphviz and Cytoscape communities to adopt at least the equivalent visual style features in both applications? Or at least to indicate in the Cytoscape and Graphviz documentation what is the equivalent feature (when there is one).

Minor comment:

Under Operation it would be good to start by explaining how to install the App (for novice Cytoscape users).

Competing Interests: No competing interests were disclosed.

We confirm that we have read this submission and 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.

Author Response 08 Nov 2016

Ziran Zhang

Thank you Dr. David Lynn and Dr. John Salamon for your comments. We completely agree with your suggestions, and we'll add those (e.g. network conversion examples, dot-app's future adaptation, detailed installation instructions, etc.) to our article. Please expect to see the changes you recommended in the second version of our article.

Competing Interests: No competing interests were disclosed.

Reviewer Report 31 October 2016

<https://doi.org/10.5256/f1000research.10513.r17148>

© 2016 König M. This is an open access peer review report distributed under the terms of the [Creative Commons Attribution License](#), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Matthias König

Institute for Theoretical Biology, Humboldt University of Berlin, Berlin, Germany

The article "dot-app: a Graphviz-Cytoscape conversion plug-in" is well written and describes background, implementation and use of the application in a comprehensible way.

major comments

"The differences that arise between the Graphviz network and the Cytoscape network stem from how the two programs handle implicit default values. If a DOT attribute is omitted from the DOT file when using a Graphviz utility, an implicit default value for that attribute is used. The list of DOT attributes and their default values can be found on the Graphviz website (<http://www.graphviz.org/content/attrs.html>)."

-> The implicit dot default values should be used as default values in dot-app for the Cytoscape styles. Users of the dot language rely on the default values and expect rendering tools to use them.

-> Especially, in combination with dot-app writing the Cytoscape default values in the exported dot this results in unnecessary changes of the graph rendering in the round trip (dot -> cytoscape -> dot). This round trip should introduce minimal changes to the rendering.

Dot default values should be used and the respective text passage and figure be updated.

minor comments

>>>

In the implementation all visual styles are hard coded in the visual mapping via bipasses. It would be better to create node attributes for the dot attributes and subsequently use them in visual styles.

In this way users have access to the dot attributes, can use them in other visual mappings, create derived node attributes from them, or use them in analyses.

>>> It is unclear how dot height and width are transformed to Cytoscape node height and width. There is some scaling factor, but how this is actually handled should be described in the manuscript. What information is lost in the roundtrip due to scaling.

For instance

height = "0.486111", width = "1.041667"

result in height > 40 in Cytoscape.

>>> The dot node font color is not rendered correctly.

See: <https://github.com/idekerlab/dot-app/issues/7>

>>> Saving as *.gz always adds the .dot extension so that files are called *.gz.dot.

If the user selects "name.gz" the file should be saved as 'name.gz', not as 'name.gz.dot'.

>>>

p5 "For the import function of dot-app, we used Java-based Parser for Graphviz Documents (JPGD), a Graphviz document parser made by Alexander Merz(5)"

-> reference missing/wrong. There should be a reference for (5).

>>>

p5 "After JPGD creates the model objects, dot-app creates a corresponding "
-> something wrong with first half-sentence, probably better: "After JPGD has created the model objects, ...

>>>

p6 "All other DOT attributes are ignored during the import process and have no effect on the visualization in Cytoscape."

If the ignored DOT attributes can be listed, state them in the manuscript as list. This is very helpful to see what information should not be used in dot files if one wants full Cytoscape support of the file. If the unsupported attributes are only the features listed in "Unsupported DOT features" than this should be clearly stated.

>>>

p2 "The store contains nearly 300 apps that ..."

At the state of review this is already >= 300. Add a date to the statement:

-> The store contains nearly 300 apps (October 2016) that ...

>>>

Add reference for graphviz software/url

see for instance <http://www.graphviz.org/content/citing-graphviz-paper>

The preferred citation is

@ARTICLE{Gansner00anopen,

author = {Emden R. Gansner and Stephen C. North},

title = {An open graph visualization system and its applications to software engineering},

journal = {SOFTWARE - PRACTICE AND EXPERIENCE},

year = {2000},

volume = {30},

number = {11},

pages = {1203--1233}

}

and the URL is www.graphviz.org

>>>

p8 use cases

Provide the *.gv files and Cytoscape files of the use cases in the supplement.

This will provide the necessary materials to follow the provided examples. I tried to save the files from the pdf which did not work (due to line breaks in pdf), copying the examples from the HTML version of the manuscript worked. Adding the two example files to the supplementary information will improve this.

>>>

p9 testing

Provide the url/repository for the test files and test results.

There should be a page which shows all the comparison images with the respective dot files. In the article it is mentioned that this exists, but no resource is given for the test files and test results.

>>>

p9 conclusion

"This article describes the dot-app Cytoscape app, which enables a user to import a DOT-formatted **here** into Cytoscape"

-> word missing

>>>

p9 conclusion

"we recognize that future versions of Cytoscape may introduce new visual effects"

-> formulation. Better: 'visual shapes and styles', visual effects are something else.

>>>

p9 conclusion

"we recognize that future versions of Cytoscape may introduce new visual effects (e.g., new arrow heads) that present opportunities for rendering DOT files more truly or for the loss of formatting information if the DOT format cannot represent them."

-> second half of sentence does not make sense (... or for the loss of formatting). Please reformulate to clarify what is meant.

>>>

The github release corresponding to the zenodo code is missing. The latest github release is 0.9.1 on 11.September, the zenodo was created in February. This makes it difficult to try to build the app from the mentioned source code, by checking out a corresponding tag.

Which version is packed in zenodo? 0.9.2 ?

Also the current app version in the app store is 0.9.3, but there is no 0.9.3 release on github.

Please create a github release which is corresponding to the zenodo release and the mentioned code state of the publication.

>>>

I was unable to build the app from latest source code. The app is working via the app store, but in addition it should be possible to build the latest source code.

<https://github.com/idekerlab/dot-app/issues/8>

git clone <https://github.com/idekerlab/dot-app.git>

mvn clean install

results in

[INFO] Compiling 16 source files to /home/mkoenig/git/dot-app/target/classes

[INFO] -----

[INFO] BUILD FAILURE

[INFO] -----

[INFO] Total time: 7.250 s

[INFO] Finished at: 2016-10-31T12:48:57+01:00

[INFO] Final Memory: 21M/238M

[INFO] -----

[ERROR] Failed to execute goal org.apache.maven.plugins:maven-compiler-plugin:2.0.2:compile (default-compile) on project dot-app: Compilation failure: Compilation failure:

[ERROR] /home/mkoenig/git/dot-

app/src/main/java/org/cytoscape/intern/read/reader/NodeReader.java:[46,36] error: package

com.alexmerz.graphviz.objects does not exist
[ERROR]

...

This should be fixed, I would recommend to setup some Continuous Integration with travis to make sure the latest version always builds.
see <https://github.com/idekerlab/dot-app/issues/8>

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, however I have significant reservations, as outlined above.

Author Response 31 Oct 2016

Ziran Zhang

Thank you so much Dr. Matthias König for your insightful comments. We really appreciate your efforts of carefully testing the dot-app features and thoroughly examining our article. We thank you for bringing the problems you found to our attention. While waiting for another referee's upcoming peer review report, we will take a close look into the issues and address them for our next version.

Competing Interests: No competing interests were disclosed.

Reviewer Report 27 October 2016

<https://doi.org/10.5256/f1000research.10513.r17124>

© 2016 Bonnet E. This is an open access peer review report distributed under the terms of the [Creative Commons Attribution License](#), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Eric Bonnet

Institute of Genomics, CEA (French Alternative Energies and Atomic Energy Commission), Evry, France

This article describes a new Cytoscape plugin called “dot-app” designed for the import and export of Graphviz files conforming to the DOT format for representing graph topologies, layouts and formatting.

Graphviz is a popular set of open source software tools initiated at the AT&T Labs Research that can generate or process DOT files. Many other software tools are using Graphviz to represent all kinds of graphs.

The DOT language support visual features and attributes that are not currently included by Cytoscape network formats. There is currently no Cytoscape plugin for the import and export of networks to the DOT language. Therefore the dot-app plugin may be useful to convert and import biological networks, and might be very interesting for the large community of Cytoscape users.

The article is globally clear, well-organized and logically structured. I particularly appreciated the efforts of the authors to explain the details of the conversion process and exactly what kind of visual properties are imported and exported, including the properties that are not supported.

I would suggest minor improvements to the article that could make it more precise and valuable for the readers.

1. Add a table for at least some of the main visual features and attributes supported by the DOT format and indicate which ones are supported by XGMML/GraphML. The sentence mentioning the differences in the introduction section is too vague.
2. Compile also in a small table some examples of applications (related to biology or not) that produces or consumes DOT files. Here also the paragraph discussing this in the introduction section is quite vague.

Minor comments:

- The reference to the JPGD parser do not appear in the bibliography.
- A few sentences on how to install the app in Cytoscape might be helpful.

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 30 Oct 2016

Ziran Zhang

Thank you Dr. Bonnet for your valuable comments. We will look into the possible improvements and minor issues you brought up, and fix them in our next version. Again, we really appreciate your detailed observations!

Competing Interests: No competing interests were disclosed.

The benefits of publishing with F1000Research:

- Your article is published within days, with no editorial bias
- You can publish traditional articles, null/negative results, case reports, data notes and more
- The peer review process is transparent and collaborative
- Your article is indexed in PubMed after passing peer review
- Dedicated customer support at every stage

For pre-submission enquiries, contact research@f1000.com

F1000Research