Towards a format for describing networks
Part 2. NetsJSON

Vladimir Batagelj
IMFM
Ljubljana, Slovenia
UP, IAM and FAMNIT
Koper, Slovenia
UL, FMF
Ljubljana, Slovenia
vladimir.batagelj@fmf.uni-lj.si

Tomaž Pisanski
UP, FAMNIT
Koper, Slovenia
IMFM
Ljubljana, Slovenia
tomaz.pisanski@upr.si

Iztok Savnik
UP, FAMNIT
Koper, Slovenia
iztok.savnik@upr.si

Ana Slavec
UP, FAMNIT
Koper, Slovenia
UP, IAM InnoRenew CoE
Koper, Slovenia
ana.slavec@famnit.upr.si

Nino Bašić
UP, FAMNIT
Koper, Slovenia
IMFM
Ljubljana, Slovenia
nino.basic@famnit.upr.si

Abstract

The key elements that a common format for describing networks should include are discussed. In this (second) part, we present components that an exchange/archive format for describing networks should contain.

Keywords

Network analysis, Network types, Format, Exchange, Archive, Data repository, Factorization, JSON, FAIR.

1 Introduction

The first part of the article provided an overview of the most important libraries and software tools for working with networks, main network description formats, network data repositories, and the various types of networks encountered in their use.

In this (second) part, we will present components that an exchange/archive format for describing networks should contain.

We have many years of experience in developing formats for describing graphs and networks [11, 10, 5]. We will present the NetsJSON format currently used to describe networks with structured data, and some ideas for improving it. This could be a starting point for the development of a common format for exchanging and archiving networks.

We will use the notation introduced in the first part of the article.

2 Description of traditional networks

2.1 Description of networks using a spreadsheet

How to describe a network $\mathcal{N} = (\mathcal{V}, \mathcal{L}, \mathcal{P}, \mathcal{W})$? In principle the answer is simple – we list its components $\mathcal{V}$, $\mathcal{L}$, $\mathcal{P}$, and $\mathcal{W}$. The simplest way is to describe a network $\mathcal{N}$ by providing $(\mathcal{V}, \mathcal{P})$ and $(\mathcal{L}, \mathcal{W})$ in a form of two tables.

As an example, let us describe a part of the network determined by the bibliographical data about the following works: Generalized blockmodeling, Clustering with relational constraint, Partitioning signed social networks, The Strength of Weak Ties.

There are nodes of different types (modes): persons, papers, books, series, journals, publishers; and different relations among them: author_of, editor_of, contained_in, cites, published_by. For some types of nodes, additional properties are known: sex, year, volume, number, first and last page, etc.

Both tables are often maintained in some spreadsheet program. They can be exported as text in CSV (Comma Separated Values) format. Tables for our example are given in Figures 1 and 2. In large networks, we split a network into some subnetworks – a collection, to avoid the empty cells.

```
name;mode;country;sex;year;vol;num;fPage;lPage;x;y
"Batagelj, Vladimir";person;SI;m;;;;;;809.1;653.7
"Doreian, Patrick";person;US;m;;;;;;358.5;679.1
"Ferligoj, Anuška";person;SI;f;;;;;;619.5;680.7
"Granovetter, Mark";person;US;m;;;;;;145.6;660.5
"Moustaki, Irini";person;UK;f;;;;;;783.0;228.0
"Mrvar, Andrej";person;SI;m;;;;;;478.0;630.1
"Clustering with relational constraint";paper;;;1982;47;;413;426;684.1;380.1
"The Strength of Weak Ties";paper;;;1973;78;6;1360;1380;111.3;329.4
"Partitioning signed social networks";paper;;;2009;31;1;1;11;408.0;337.8
"Generalized Blockmodeling";book;;;2005;24;;1;385;533.0;445.9
"Psychometrika";journal;;;;;;;;741.8;086.1
"Social Networks";journal;;;;;;;;321.4;236.5
"The American Journal of Sociology";journal;;;;;;;;111.3;168.9
"Structural Analysis in the Social Sciences";series;;;;;;;;310.4;082.8
"Cambridge University Press";publisher;UK;;;;;;;534.3;238.2
"Springer";publisher;US;;;;;;;884.6;174.0
```

Figure 1: File `bibNodes.csv` – $(\mathcal{V}, \mathcal{P})$ table for nodes

2.2 Factorization and description of large networks

To save space and improve computing efficiency, we often replace values of categorical variables with integers. In R, this encoding is called a *factorization*.

We enumerate all possible values of a given categorical variable (coding table) and afterward replace each value with the corresponding index in the coding table. Since node labels/IDs can be considered a categorical variable, factorization is also usually applied to them.

Information Society 2025, 6–10 October 2025, Ljubljana, Slovenia

https://doi.org/10.70314/is.2024.sikdd.12

```
from;relation;to
"Batagelj, Vladimir";authorOf;"Generalized Blockmodeling"
"Doreian, Patrick";authorOf;"Generalized Blockmodeling"
"Ferligoj, Anuška";authorOf;"Generalized Blockmodeling"
"Batagelj, Vladimir";authorOf;"Clustering with relational constraint"
"Ferligoj, Anuška";authorOf;"Clustering with relational constraint"
"Granovetter, Mark";authorOf;"The Strength of Weak Ties"
"Granovetter, Mark";editorOf;"Structural Analysis in the Social Sciences"
"Doreian, Patrick";authorOf;"Partitioning signed social networks"
"Mrvar, Andrej";authorOf;"Partitioning signed social networks"
"Moustaki, Irini";editorOf;"Psychometrika"
"Doreian, Patrick";editorOf;"Social Networks"
"Generalized Blockmodeling";containedIn;"Structural Analysis in the Social Sciences"
"Clustering with relational constraint";containedIn;"Psychometrika"
"The Strength of Weak Ties";containedIn;"The American Journal of Sociology"
"Partitioning signed social networks";containedIn;"Social Networks"
"Partitioning signed social networks";cites;"Generalized Blockmodeling"
"Generalized Blockmodeling";cites;"Clustering with relational constraint"
"Structural Analysis in the Social Sciences";publishedBy;"Cambridge University Press"
"Psychometrika";publishedBy;"Springer"
```

Figure 2: File `bibLinks.csv` – $(\mathcal{L}, \mathcal{W})$ table for links

```
# transforming CSV file to Pajek files, by Vladimir Batagelj, June 2016
colC <- c(rep("character",4),rep("numeric",5)); nas=c("","NA","NaN")
nodes <- read.csv2("bibNodes.csv",encoding='UTF-8',colClasses=colC,
  na.strings=nas)
n <- nrow(nodes); M <- factor(nodes$mode); S <- factor(nodes$sex)
mod <- levels(M); sx <- levels(S); S <- as.numeric(S); S[is.na(S)] <- 0
links <- read.csv2("bibLinks.csv",encoding='UTF-8',colClasses="character")
F <- factor(links$from,levels=nodes$name,ordered=TRUE)
T <- factor(links$to,levels=nodes$name,ordered=TRUE)
R <- factor(links$relation); rel <- levels(R)
net <- file("bib.net","w"); cat('*vertices ',n,'\n',file=net)
clu <- file("bibMode.clu","w"); sex <- file("bibSex.clu","w")
cat('%',file=clu); cat('%',file=sex)
for(i in 1:length(mod)) cat(' ',i,mod[i],file=clu)
cat('\n*vertices ',n,'\n',file=clu)
for(i in 1:length(sx)) cat(' ',i,sx[i],file=sex)
cat('\n*vertices ',n,'\n',file=sex)
for(v in 1:n) {
  cat(v,' "',nodes$name[v],'"\n',sep='',file=net);
  cat(M[v],'\n',file=clu); cat(S[v],'\n',file=sex)
}
for(r in 1:length(rel)) cat('*arcs :',r,' "',rel[r],'"\n',sep='',file=net)
cat('*arcs\n',file=net)
for(a in 1:nrow(links))
  cat(R[a],': ',F[a],' ',T[a],' 1 l "',rel[R[a]],'"\n',sep='',file=net)
close(net); close(clu); close(sex)
```

Figure 3: `CSV2Pajek.R` – program for converting tables into network in Pajek format

```
*vertices  16
1 "Batagelj, Vladimir"
2 "Doreian, Patrick"
3 "Ferligoj, Anuška"
4 "Granovetter, Mark"
5 "Moustaki, Irini"
6 "Mrvar, Andrej"
7 "Clustering with relational constraint"
8 "The Strength of Weak Ties"
9 "Partitioning signed social networks"
10 "Generalized Blockmodeling"
11 "Psychometrika"
12 "Social Networks"
13 "The American Journal of Sociology"
14 "Structural Analysis in the Social Sciences"
15 "Cambridge University Press"
16 "Springer"
*arcs :1 "authorOf"
*arcs :2 "cites"
*arcs :3 "containedIn"
*arcs :4 "editorOf"
*arcs :5 "publishedBy"
```

```
*arcs
1: 1 10 1 l "authorOf"
1: 2 10 1 l "authorOf"
1: 3 10 1 l "authorOf"
1: 1 7 1 l "authorOf"
1: 3 7 1 l "authorOf"
1: 4 8 1 l "authorOf"
4: 4 14 1 l "editorOf"
1: 2 9 1 l "authorOf"
1: 6 9 1 l "authorOf"
4: 5 11 1 l "editorOf"
4: 2 12 1 l "editorOf"
3: 10 14 1 l "containedIn"
3: 7 11 1 l "containedIn"
3: 8 13 1 l "containedIn"
3: 9 12 1 l "containedIn"
2: 9 10 1 l "cites"
2: 10 7 1 l "cites"
5: 14 15 1 l "publishedBy"
5: 11 16 1 l "publishedBy"
```

Figure 4: File `bib.net` – bibliography network in Pajek format (split in two columns)

This approach is used in most programs dealing with large networks. Unfortunately, the coding table is often considered as a kind of metadata and is omitted from the description.

In data analysis, indices start with 1, but real computer scientists start counting from 0. Therefore, it is desirable to include the minimal index value in the description.

In Pajek, node property can be represented in the associated file as a vector (numbers, **.vec**), a partition (nominal, **.clu**), or a permutation (order, **.per**). All network files can be combined into a single project file (**.paj**). Metadata can be added as comments written on lines starting with the **%**.

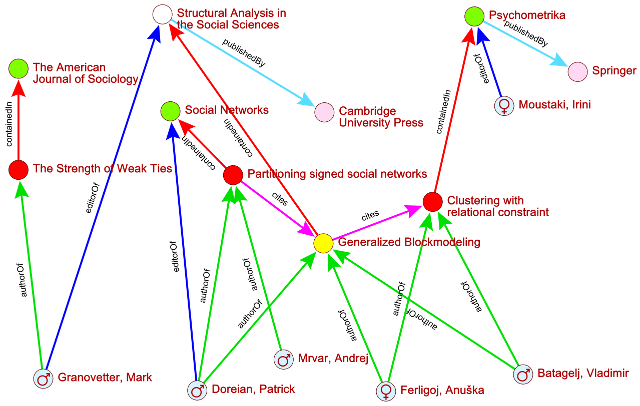

Figure 5: Bibliographic network – picture / Pajek

Using a short program in R (see Figure 3), we may transform both tables into Pajek files: a network file **`bib.net`** (see Figure 4) and partition files **`bibMode.clu`** and **`bibSex.clu`**. Finally, using Pajek, we produce a visualization of our network (see Figure 5).

All the files related to the bibliographic example are available at GitHub/Bavla [2].

3 Nets and NetsJSON

We were satisfied with the "traditional" network description, as implemented in Pajek [15], until we became interested in networks with node/link properties that are not measured in standard scales (ratio, interval, ordinal, nominal), but have structured values (text, subset, interval, distribution, time series, temporal quantity, function, etc.). In topological graph theory, an embedding is described by assigning a rotation to each node [20]. For describing temporal networks, we initially extended the Pajek format, defined and used the Ianus format [12]. We needed a format that could describe structured values. There were two obvious choices for the format base – XML and JSON. They are both widely known and suitable as structured data formats. However, JSON can represent the same data as XML in a more concise and (human) readable way. We chose JSON and in 2015 started developing and using the NetJSON format and the Nets Python package to handle networks with structured-valued properties [6, 5, 4].

On February 26, 2019, the format was renamed to NetsJSON because of the collision with http://netjson.org/rfc.html. NetsJSON has two versions: a *basic* and a *general* version. The current implementation of the Nets library supports only the basic version.

In addition to describing networks with structured values, NetsJSON is expected to offer the capabilities of (most) existing network description formats [13, 22] (archiving, conversion) and provide input data for D3.js visualizations.

3.1 Informal description of the basic NetsJSON format

A network description in NetsJSON follows the JSON (JavaScript) syntax and consists of five main fields (`netsJSON`, `info`, `nodes`, `links`, `data`).

The `netsJSON` field identifies the format, the `info` field contains metadata, the nodes field contains a table $(\mathcal{V}, \mathcal{P})$ and the `links` field contains a table $(\mathcal{L}, \mathcal{W})$.

In recent years, we also analyzed bike systems (link weight is a daily number of trips distribution), bibliographies (yearly distributions of publications or citations), and multiway networks [8, 9, 1]. It turned out that it was necessary to add another main field, data, to the basic NetsJSON format, in which we provide additional data about the properties of values (translations of labels in selected languages, algebraic structure [7]).

```
{
"netsJSON": "basic",
"info": {
   "org":1, "nNodes":n, "nArcs":mA, "nEdges":mE,
   "simple":TF, "directed":TF, "multirel":TF, "mode":m,
   "network":fName, "title":title,
   "time": { "Tmin":tm, "Tmax":tM, "Tlabs": {labs} },
   "meta": [events], ...
   },
"nodes": [
   { "id":nodeId, "lab":label, "x":x, "y":y, ... },
   ***
   ],
"links": [
   { "type":arc/edge, "n1":nodeID1, "n2":nodeID2, "rel":r, ... },
   ***
   ],
"data": {
   "data1":description1,
   ***
   }
}
```

where ... are user-defined properties and *** is a sequence of such elements.

An event description can contain the following fields:

```
{  "type": type,
   "date": date,
   "title": short description,
   "author": name,
   "desc": long description,
   "url": URL,
   "cite": reference,
   "copy": copyright
}
```

It is intended to provide information about the "life" of the dataset – collection/creation, changes, releases, uses, publications, etc.

For describing temporal networks, a node element and a link element have an additional required property tq – a temporal quantity. For example, see violenceU.json at GitHub/Bavla/Graph/JSON describing the Franzosi's violence network.

The general NetsJSON format is also expected to support the description of network collections.

4 Elements of a common network format

Our experience with network analysis to date is summarized in the following recommendations on the elements of a common format for describing networks.

For data integrity, it makes sense to combine data and metadata into a single file. To preserve the structure of data, it makes sense to base the format on JSON, which fits well with the data structures of modern programming languages. JSON also supports Unicode.

We would also encourage the provision, as metadata, of information about the context of the network, additional knowledge about it, articles or notebooks on its analysis, comments of users, etc. Kaggle is a good example. An improved ICON repository or Networkrepository (we disagree with their "citation request") could be the way to go. Existing metadata standards should be taken into account (Dublin Core, FAIR, Schema).

By FAIR principles, the format should support:

- Findability: Globally unique and persistent identifier, rich metadata.
- Accessibility: Open, free, and universally implementable standardized communication protocol.
- Interoperability: Formal, accessible, shared, and proudly applicable language for knowledge representations.
- Reusable: Metadata are richly described and associated with detailed provenance.

Data has a "life". When selecting data, its age is often important. Metadata should include at least the collection / creation date and the last modification date.

The format must support all types of networks (simple, 2-mode, linked, multi-relational, multi-level, temporal). The network can contain both arcs and edges, as well as parallel links. To describe some knowledge graphs, it would be necessary to allow other links to act as the end nodes of a link [17].

As mentioned earlier, using factorization produces a more concise description of the network. In cases where the node names are not too long and are readable, we sometimes want to avoid factorization. This can be achieved by using a switch that indicates whether factorization is used. We can also shorten the description length by introducing default values. If we also allow counting from 0, it makes sense to add information about the smallest index.

Long labels cause problems when printing/visualizing (parts of) networks and results. Therefore, it is useful to have abbreviated versions of labels available. For language-based labels, it is sometimes useful to offer additional versions in selected other languages, which increases the accessibility of the data and the understandability of the results.

Most of the network datasets produced by network science have no node labels. Node labels are not needed if you study distributions, but they are essential in the interpretation of the obtained "important substructures". We would encourage providing node labels, or at least some typological information, in cases where privacy issues arise.

The common format should support descriptions of networks specific to specialized fields of application, such as molecular graphs, genealogies (p-graphs) [26], and topological graph embeddings [20, 21], among others. The format must be extensible. In addition to the agreed-upon fields, the users can add their own, allowing for a comprehensive description of their data.

One option to make descriptions more compact is to use the default values of selected properties.

It is also interesting to ask whether and, if so, how to include descriptions of its displays in the network description. Perhaps it would be worth relying on VEGA-lite [23, 25] and D3.js [14]. Some ideas can also be taken from the section on "defining visualization parameters in the input file" in the Pajek manual / 5.3 Exporting pictures to EPS/SVG [18, p. 89].

Although we are committed to a single-file approach, there may be times when external files are needed (for example, images to display nodes). Consideration should be given to how to support this option.

Given the basic purpose of the common format, standard tools (ZIP) can be used to compress large networks.

We have not yet started working on a general format. It is supposed to enable descriptions of collections of networks. The question arises about the scope of validity of IDs - does the same ID in different networks represent the same or other units? This is important for operations such as the union or intersection of networks. Which way to go - introducing contexts or using matchings? Maybe some ideas from the Open Archives Initiative Object Reuse and Exchange (OAI-ORE) and GraphX could be used [19, 16]. An interesting option is the constructive network description – building a network from smaller components [10] or describing a network by its construction sequence [3].

Additional ideas may be found on the page "*A Python Graph API?*" [24]. For now, we would leave aside descriptions of generalizations of networks (multiway networks and hypernets), but we must not forget about them.

The agreed format must be well documented and supported by examples of the use of supported options.

5 Conclusions

The availability of the data used in the article enables the reproducibility and verifiability of the analyses performed. In addition, the obtained results can be verified or supplemented with other methods. When developing new methods, accessible and well-documented data are also very important - it is good to test a new method on several data sets and check whether it gives meaningful/expected results.

Acknowledgements

The computational work reported in this paper was performed using programs R and Pajek [15]. The code and data are available at Github/Bavla [2].

V. Batagelj is supported in part by the Slovenian Research Agency (research program P1-0294, research program CogniCom (0013103) at the University of Primorska, and research projects J5-2557, J1-2481, and J5-4596) and prepared within the framework of the COST action CA21163 (HiTEc).

T. Pisanski is supported in part by the Slovenian Research Agency (research program P1-0294 and research projects BI-HR/23-24-012, J1-4351, and J5-4596).

N. Bašić is supported in part by the Slovenian Research Agency (research program P1-0294 and research project J5-4596).

References

[1] Vladimir Batagelj. 2024. Cores in multiway networks. *Social Network Analysis and Mining*, 14, 1, 122.
[2] Vladimir Batagelj. 2016. GitHub/Bavla: bibliographic network example. accessed: 2025-03-18. (2016). https://github.com/bavla/netsJSON/tree/master/example/bib.
[3] Vladimir Batagelj. 1985. Inductive classes of graphs. In *Proceedings of the 6th Yugoslav Seminar on Graph Theory*, 43–56.
[4] Vladimir Batagelj. 2016. Nets – Python package for network analysis. accessed: 2025-03-18. (2016). https://github.com/bavla/Nets.
[5] Vladimir Batagelj. 2016. NetsJSON – a JSON format for network analysis. accessed: 2025-03-18. (2016). https://github.com/bavla/netsJSON.
[6] Vladimir Batagelj. 2016. Network visualization based on JSON and D3.js. slides. (2016). https://github.com/bavla/netsJSON/blob/master/doc/netVis.pdf.
[7] Vladimir Batagelj. 2021. Semirings in network data analysis / an overview. slides. (2021). https://github.com/bavla/semirings/blob/master/docs/semirings.pdf.
[8] Vladimir Batagelj and Anuška Ferligoj. 2016. Symbolic network analysis of bike sharing data / Citi Bike. accessed: 2025-03-18. (2016). https://github.com/bavla/Bikes/blob/master/bikes.pdf.
[9] Vladimir Batagelj and Daria Maltseva. 2020. Temporal bibliographic networks. *Journal of Informetrics*, 14, 1, 101006.
[10] Vladimir Batagelj and Andrej Mrvar. 1995. NetML. accessed: 2025-03-18. (1995). https://github.com/bavla/netsJSON/blob/master/doc/snetml.pdf.
[11] Vladimir Batagelj and Andrej Mrvar. 2018. Pajek and pajekxxl. In *Encyclopedia of Social Network Analysis and Mining*. Springer, 1–13.
[12] Vladimir Batagelj and Selena Praprotnik. 2016. An algebraic approach to temporal network analysis based on temporal quantities. *Social Network Analysis and Mining*, 6, 1–22.
[13] Jernej Bodlaj and Monika Cerinšek. 2014, 2017. Network data file formats. In *Encyclopedia of Social Network Analysis and Mining*. Reda Alhajj and Jon Rokne, editors. Springer New York, New York, NY, 1076–1091. ISBN: 978-1-4614-7163-9. DOI: 10.1007/978-1-4614-7163-9_298-1.
[14] Bostock, Mike and Davies, Jason and Heer, Jeffrey and Ogievetsky, Vadim. [n. d.] The JavaScript library for bespoke data visualization. accessed: 2025-08-29. (). https://d3js.org/.
[15] Wouter De Nooy, Andrej Mrvar, and Vladimir Batagelj. 2018. *Exploratory social network analysis with Pajek: Revised and expanded edition for updated software*. Vol. 46. Cambridge university press.
[16] Joseph E Gonzalez, Reynold S Xin, Ankur Dave, Daniel Crankshaw, Michael J Franklin, and Ion Stoica. 2014. {Graphx}: graph processing in a distributed dataflow framework. In *11th USENIX symposium on operating systems design and implementation (OSDI 14)*, 599–613.
[17] Aidan Hogan et al. 2021. Knowledge graphs. *ACM Computing Surveys (Csur)*, 54, 4, 1–37.
[18] Andrej Mrvar and Vladimir Batagelj. 2025. Pajek reference manual. *Version 6.01*. http://mrvar.fdv.uni-lj.si/pajek/pajekman.pdf.
[19] Open Archives Initiative. 2014. Object Reuse and Exchange (OAI-ORE). 2014-08-14. accessed: 2025-03-18. (2014). https://www.openarchives.org/ore/.
[20] Tomaž Pisanski. 1980. Genus of cartesian products of regular bipartite graphs. *Journal of Graph Theory*, 4, 1, 31–42.
[21] Tomaž Pisanski and Arjana Zitnik. 2004. Representations of graphs and maps. In *26th International Conference on Information Technology Interfaces, 2004*. IEEE, 19–25.
[22] Matthew Roughan and Jonathan Tuke. 2015. Unravelling graph-exchange file formats. *arXiv preprint arXiv:1503.02781*.
[23] Arvind Satyanarayan, Dominik Moritz, Kanit Wongsuphasawat, and Jeffrey Heer. 2016. Vega-lite: a grammar of interactive graphics. *IEEE transactions on visualization and computer graphics*, 23, 1, 341–350.
[24] The Python Wiki. 2011. A Python Graph API? accessed: 2025-03-18. (2011). https://wiki.python.org/moin/PythonGraphApi.
[25] University of Washington Interactive Data Lab. [n. d.] Vega-Lite – A Grammar of Interactive Graphics. accessed: 2025-08-29. (). https://vega.github.io/vega-lite/.
[26] Douglas R White, Vladimir Batagelj, and Andrej Mrvar. 1999. Anthropology: analyzing large kinship and marriage networks with pgraph and pajek. *Social Science Computer Review*, 17, 3, 245–274.