Decycling Information Networks

Jean-François Baffier[1], Benjamin Renoust[2]

[1] RIKEN Center for Advanced Intelligence Project, Tokyo, Japan
jf@baffier.fr

[2] Osaka University Institute for Datability Science, Osaka, Japan
renoust@ids.osaka-u.ac.jp

Mots-clés : *information-networks, decycling, multilayer, flow*

Extended abstract

Graphs are simple objects, a set of vertices, and a set of links between those, that represent a wide variety of problems. Among the different categories of graphs, some allow a faster computation of many algorithms. A *directed acyclic graph* (DAG) is a graph where the links are directed (called arcs) and where no directed cycle exists. This structure often allows a faster computation, or even the existence, of various graphs algorithms. A *strongly connected component* of a directed graph is a subgraph where a directed path exists between any ordered pair of vertices (not necessarily distinct) in the component. Any directed graph can be decomposed in linear time into a DAG of its strongly connected components (of maximum possible size).

In our previous work on the flow of knowledge in information networks [1, 2], we designed an ascending flow framework to evaluate the influence of the different nodes in the transmission of information. As each node produces and most often receives some information, that flow framework was naturally designed for DAGs, to avoid the measure of information produced to cycle back to the node that produced it. In both that previous work and this one, we used the case of an academic publication network to evaluate our framework. In particular, we used a subgraph of the arXiv preprint network in the High-Energy Theoretical Physic. Preprint networks have the remarkable property that each article can be updated over time. We call *versioning* this process that allows, among others, the production of new knowledge and the update of the references. Hence cycles may appear over time. Of course, cycles exist in other networks of information such as news networks, or social networks.

This work provides different methods to *decycle* a graph while conserving the flow of information in the network (a graph with nodes that generate and transmit information). To decycle such networks, we apply our methods sequentially to each strongly connected component. We introduce the fundamental concepts of this work with classical graphs, but it can be easily extended to most of the more complex networks such as multiplex networks or temporal networks. We also address the fairness issue coming from versioning. That is how to evaluate the amount of information brought by each new version.

A simple and natural method, which we call here *deletion-decycling*, to avoid cycles is to delete some of the links in each connected component till all cycle ceases to exist. One might choose the links randomly or delete them according to some order (time, for instance). This operation reduced the size of the network. However, the question of fairness raises, as it is challenging to evaluate its impact on the transmission of information. Some previously connected nodes might even be disconnected in the process.

More complex methods require to add some elements to the networks. In the following methods, we differentiate each vertex v into an *in-vertex* v^{in} and an *out-vertex* v^{out}. The information brought to v from outside the strongly connected component is transmitted to v^{in}. The information sent by v outside the strongly connected component is now sent by v^{out}. A set of auxiliary vertices is present in between all in-vertices and out-vertices of the component.

This set serves to the decycling process. Note that all the vertices and links replacing the component form a DAG. Finally, we need now to set up the production of knowledge to be only on the in-vertices of each component. The measure of each vertex influence is the value transmitted by the corresponding out-vertex.

Another natural method is to consider that nodes in the same strongly connected component cannot be differentiated in their production of information. We call *contraction-decycling* this method. The decycling operation is as simple as having a central auxiliary vertex where each in-vertex is pointing at and that points to each out-vertex. This approach is not expensive to compute and conserve some fairness for small and dense strongly connected components.

The most significant contribution of this work is the *path-decycling* method. It consists of expanding the component into a DAG reproducing all possible non-cycling paths existing in the component. In our academic network, suppose we have a very simple case of a component made of two articles, a_1 and a_2, citing each other. We would expand the component into the in and out-vertices, a_1^{in}, a_2^{in}, a_1^{out}, a_2^{out}, and the auxiliary vertices $a_{\{1\}}, a_{\{1,2\}}, a_{\{2\}}, a_{\{2,1\}}$. Where the vertex $a_{\{1,2\}}$ incarnates being on the any path starting by the path $\{1, 2\}$. At each step on that path p, if the vertex v has a link going out of the component, then $v_{\{p\}}$ has an out-going arc connected to v_{out}.

This latter method is accurate but suffers an exponential growth in the size of the strongly connected component. Experiments on a complete strongly connected component show a size limit of about 10.

To complete the theoretical and practical analysis of our decycling methods, we introduce the notion of *pseudo-DAG* (PDAG). A k-PDAG is a graph with strongly connected components of at most k elements. Intuitively, for a given reasonably small k, a k-PDAG allows the following improvements on the different decycling methods. Both *deletion-decycling* and *contraction-decycling* should show an improvement in accuracy and be more fair. The exact but exponential (in time and space) *path-decycling* method has a theoretical guarantee to run in a reasonable time – in terms of complexity, this method is FPT.

We will confirm those results by practical experiments on various networks, including the arXiv subgraph mentioned above, that possesses a strongly connected component of about 7000 articles. Note that several measures defined in [2] are multiplex, which allows our experiments to be on simple and more complex networks.

Finally, based on the analysis in [2], we propose a bounded version to the *path-decycling* method that we call *partial path-decycling* (PPD). The main idea behind this approach is that the amount of information, or knowledge, decreases at each step. The contribution to the flow of knowledge of a given node after a few steps should be neglectable. Thus it can be bounded with few changes to the expected result. We refer to a PPD bounded by s steps as s-PPD.

We evaluate the accuracy and speed improvements of s-PPD on k-PDAG for $k \in \{3, 10\}$ and $s \in \{1, k\}$. We also evaluate the computation time on graphs with long cycles and showcase some use-cases on academic networks.

We provided a method to decycle graphs, efficiently for PDAG, and that balance accuracy and efficiency for more massive graphs. We expect PPD (and its bounded versions) to be of theoretical and practical interest to the community to measure various flows of knowledge and influences in different information networks. All the data and algorithms will be available as an open-source library online.

Références

[1] Benjamin Renoust, Vivek Claver, and Jean-François Baffier. Flows of knowledge in citation networks. In Hocine Cherifi, Sabrina Gaito, Walter Quattrociocchi, and Alessandra Sala, editors, *Complex Networks & Their Applications V*, pages 159–170, Cham, 2017. Springer International Publishing.

[2] Benjamin Renoust, Vivek Claver, and Jean-François Baffier. Multiplex flows in citation networks. *Applied Network Science*, 2(1) :23, Jul 2017.