



Discussion Paper

Complexity and simplification of networks (v2)

Léon Willenborg

December 21, 2021

The paper deals with two topics: the quantification of the complexity of networks (graphs and digraphs) and the simplification of networks by identifying their most important parts (nodes and arcs / edges) and leaving out the less important parts. The first topic is a preparation for the second one. It provides measures to quantify the most important nodes and arcs in a network. Complexity for graphs is first considered, by the average degree. Then the complexity of digraphs is studied on the basis of reachability. The main goal of this paper is to simplify complex networks by focusing on their essential parts. This is in fact complexity reduction. In this way one obtains an overview by removing distracting details. Edges or arcs may need to be added in order to preserve the topology of the original network. Network reduction can be compared to (and was in fact inspired by) zooming in or out at cartographic maps: for an overview of an entire country information on hamlets and villages is not needed. Only, cities, towns and other more significant geographic features that are of interest at that level are shown. Zooming in to a small part of the country yields information on less prominent features. So there is a trade-off between scale and detail: global scale and limited detail go together as well as local scale with an abundance of detail. For networks the same kind of trade-off can be envisioned: for an overview of the entire network the hubs are important and the way they are interconnected. For a small part of the network, however, detailed information on less important nodes should also be provided. This begs the question what are in fact the important parts of a network? How do we define them? Various measures (node ranks) are discussed to quantify the relative importance of nodes. With such measure one can in turn define arc ranks, which can be used to select important arcs.

1 Introduction

Complexity, and in particular complexity reduction, in networks is what this paper¹⁾ is about. Complexity of a network is a multifaceted concept. Part of the paper explores this concept, but the other focus point is simplification of networks. This is a practical way of dealing with complexity, and in particular complexity control. This topic can actually be tackled with only an intuitive understanding of the concept of complexity, which can be very sophisticated and computationally intensive to apply. To start with, it is necessary to explore measures to characterize the complexity of networks. In fact, 'complexity of networks' is a concept that may invoke mental pictures of networks, that are highly branched and look complicated, but actually capturing this concept in more precise measures is a challenge. It may be the case that it is so complicated because there are several aspects involved. Because of this, there may not be a single measure that expresses all these properties. Or such a measure is possible if the various aspects involved are explored, and for each at least one suitable measure has been defined. In that case an overall measure of complexity can be defined by combining the various measures of aspects of complexity.

The remainder of the paper is organized as follows.

In Section 2 some motivating examples of networks are discussed. Some of the examples are represented as (undirected) graphs as they involve symmetric relationships between elements.

¹⁾ This is the second version of [15]. Major changes have been made in Section 9. The master's thesis of Simon van Wageningen ([11]) was the direct cause for this adaptation. It made me realize that the probabilistic method of adding arcs to obtain a complexity measure for digraphs was misguided and should be removed from the paper.

Others are represented by directed graphs (digraphs) as they are about asymmetric relationships between elements. The latter type of relationship includes the former, but it makes sense to distinguish graphs as a special kind of network, in general but, particularly in the present paper.

In Section 3 the complexity of networks is discussed in a general way, as an introduction to what is coming in the following chapters. As a network comprises both undirected and directed graphs (or 'digraphs'), this chapter's aim is to explain why the complexity of digraphs is different from that of graphs. In the sequel first the complexity of graphs is studied and then that of digraphs. This is a natural way to proceed as it is simpler to define the complexity of graphs than the complexity of digraphs. This is a result of the symmetry that graphs possess and digraphs (in general) lack.

In Section 4 the complexity of (undirected) graphs in terms of the average degree of its nodes is considered. This is done by looking at classes of graphs which have the same number of edges. The only freedom in such a class is to rearrange the arcs and 'glue them together' to form a graph. This 'glueing' is in fact identification of nodes. So the number of nodes in the class of graphs with a fixed number of edges varies. As a measure of complexity the average degree of the nodes in the graph is assumed. This turns out to be a good measure for how compact a graph is. More compact graphs appear to be more complex. Because we assume that we are dealing with graphs and not multigraphs, so that there can be at most one edge between any two points, the number of points in these graphs has a lower bound typically well above 1.

In Section 5 the complexity of graphs is defined in terms of the length of an optimal path. This path is of minimal length and covers all edges of the graph. The path can be viewed as the result of an optimal search procedure that searches the entire graph, traversing all of its edges at least once, jumping from a node to a node linked to it by an edge. The length of the search path depends on the choice of the start and finish point. We are interested in the shortest path of this type. To find such a path, a nontrivial optimization problem needs to be solved. The complexity measure for this search approach is the length of the shortest path covering all edges divided by the length of a tour starting and ending at the same point in the graph (which is twice the number of edges in the graph).

In Section 6 the average distance of different points in a graph is used as such a measure. The distance used is the 'natural' distance where each edge has length 1. This complexity measure is related to the Wiener index used in mathematical chemistry to quantify the branching of organic molecules; it is topological in nature, not geometric.

In Section 7, we consider the complexity of a special class of digraphs, namely routing digraphs, which are acyclic digraphs with a single source and a single sink. Such digraphs often appear in questionnaires. These digraphs and some of their properties, among them complexity, were studied by the author in his PhD thesis (of which [13] is a slightly modified version).

In Section 8 a simple complexity measure for digraphs is considered, namely one that is based on the asymmetry of the adjacency matrix of the digraph. This is the same as the number of pairs of nodes for which one arc is defined, but not its counter-arc. The idea is that the more a digraphs differs from its underlying graph, the more complex it is, in the class of digraphs with the same underlying graph.

In Section 9 the complexity of digraphs based on the concept of reachability is considered. Using this concept one looks at the nodes that can be reached from each of the nodes in the digraph.

In case of a connected graph (viewed as a digraph), one can reach every node from every other node. The more deviation is found in reachability the more complex the digraph is considered to be. Reachability is a concept that is closely linked to transitive closure of the network (and its adjacency matrix), which is explored in this section. The minimum number of arcs needed to be added to obtain a full reachability is a measure for the complexity of a digraph. The more such arcs are needed the complexer the digraph is. This is one measure of complexity for digraphs that is explored in this section. Another complexity measure for digraphs is also considered in this section. This measure is based on certain probability distributions on the reachability sets.

So far only global measures of complexity of digraphs have been considered. This means that the entire network is involved to compute the measure. However, this is not always possible or practical. It is attractive to have a local concept of complexity. It is this issue that Section 10 explores. Central to the idea of local complexity is the use of neighbourhoods of nodes of a network. Local complexity is not fully explored in the present paper; only some first steps are made into this area.

In Section 11 we switch to the second major topic of this paper namely network reduction. We start with the concept of node rank. This is an attribute that can be described as a kind of popularity measure of the nodes in a digraph, on the basis of being pointed at by arcs. The popularity of nodes that point to other nodes can be taken into account: a node is more popular if a high ranking node is pointing to it than a lower ranking one. Several possibilities of defining node ranks are discussed. Node ranks can be used to select nodes. By selecting the more important ones one can reduce the original network (which may be rather big) and concentrate on the more interesting parts, while discarding the distracting details.

Once node ranks have been defined, arc ranks can be derived. This is shown in Section 12. We use Iterative Proportional Fitting (IPF) as an algorithm to achieve this, but other, similar, algorithms could be used just as well. As marginals in this algorithm the tables with indegrees and outdegrees, respectively, are used. The arc ranks can also be used when reducing a (complex) network, as is shown in the next section.

This reduction process is considered in Section 13, which should lead to the essence of a network. This features only the important nodes of the original network and otherwise faithfully represents its topology. Arc ranks can be used to modify node ranks, if one wishes to do so. In this case nodes and arcs not represented in the reduced digraph do not contribute to the node ranks of the reduced digraph. In case one chooses not to do this but to keep the original ranks, the nodes in the reduced graph actually represent a cluster of nodes. This mimicks the idea of the renormalization group idea in statistical physics: Ising models for spin lattices, which are graphs, not merely digraphs. This section also considers the complexity of reduced digraphs.

Section 14 closes the main part of the paper with a discussion of the main results. Also some topics for future research that have been noted in the main text are collected here.

The paper is completed by a list of references and four appendices. Appendix A contains several examples illustrating the reachability concept graphically, to bolster the intuition. Appendix B presents some graphs where average distances of pairs of points in a graph are used to define complexity measures. Appendix C contains an overview of the notation used in this paper. Appendix D contains a glossary.

2 Motivating examples

2.1 Networks: Graphs and digraphs

In the examples to be presented the underlying network is a directed graph (for short: digraph), in which we have arcs instead of edges. The arcs are directed, whereas edges are not.

Alternatively we can view an edge $\{a, b\}$ as a pair of arcs (a, b) and (b, a) . In general digraphs are more complicated than graphs, because of the asymmetry of the arc distribution: an arc may not have a counter-arc (pointing in the other direction). A road network with bidirectional streets is easier to understand, because a path connecting point a to b automatically yields a path in the opposite direction, i.e. from b to a .

Sometimes it is useful to forget about directions and replace an arc (a, b) by the corresponding edge $\{a, b\}$.

2.2 Transportation networks

Transportation networks can be defined depending on the type of vehicles involved, such as cars, trains, ships, airplanes, etc. Each such network consists of links between locations. These links are roads connecting villages, cities, shops, etc., waterways connecting harbours, or airways between airports, etc.

In an overview map of a transportation network one would want to present the important connections (in terms of average traffic) and discard the unimportant ones. For a detailed local map one would focus on those links that are important for the area. This may also mean that links that are unimportant at a given level are discarded.

2.3 Routing structures in questionnaires

A questionnaire consists of a finite number of questions. For each question an answer is an element of a domain. An answer to a particular question may lead to a specific next question. The idea is that in this way one can be more efficient in the questions put to interviewers. If some question reveals that the respondent is unemployed it does not make sense to bother them with questions about their current work.

In a questionnaire one can usually group questions into themes and treat these as single nodes to describe the questionnaire at a higher level of abstraction. To understand a detailed questionnaire it is of interest to be able to start at a high level of abstraction and click at nodes to unfold them to see the next level of nodes. The process may be repeated a few times until one arrives at the most detailed level, consisting of the questions in the questionnaire.

The zooming in that has just been described uses only information describing the logical structure of the questionnaire. But suppose that the questionnaire has been used for some time. Then information about how it is actually filled in is available. This can be used to focus on the most important paths through the questionnaire, at different levels of aggregation of the questions (subjects).

2.4 LANs

A local area network (LAN) is a network of units (computers) that are interconnected by cables.²⁾ The units are represented by points in the network, and cables connecting the units are represented by edges.³⁾ They are typically connected in such a way that any two of them can communicate with each other. When the LAN is connected to the internet they can all exchange information with the internet as well. The way the units (PCs, say) are connected in a LAN is restricted. It is in general not possible to connect each pair of units with a separate cable. The number of sockets for each unit is typically limited.⁴⁾ Also the entire network configuration would change if a unit is added or removed from the LAN. This would affect the global structure of the LAN. It is desirable that such an operation is a local matter. Of course, if one would opt for a minimal network connecting all the units (a network in the form of a spanning tree) there are other problems one faces. If one link would be damaged the LAN would be disconnected. Also, there would possibly be a lot of data traffic over the network. So in practice, a structure is adopted which is a compromise between a minimal network (in the form of a tree) and a maximal network (represented by a complete graph). Several things have to be balanced in an optimal compromise for a LAN-network, with a variety of constraints to be taken into account.

2.5 Internet

At the lowest level of description the internet is a collection of (virtual) webpages that link to each other using hyperlinks. The internet is also not a static structure but changes all the time: new URLs or webpages are created, modified or deleted continuously.⁵⁾ Not only is the web dynamic, it is also huge in size. There are far too many webpages to detect them all (or even a significant portion) at a certain moment in time to get a good impression of the structure of the Web.⁶⁾ Networks such as the Internet are so huge (and volatile) that they cannot be instantly known. And if time is taken to investigate them they change. At best one can only probe them by taking a sample. The sample is then used to estimate certain properties of the network. Such networks are called random access networks (RANs). See [14] for more information on such networks.

2.6 Genealogical network

A directed network in which for each person their biological father or mother are given, assuming that they are known. This is another example of a random access network (like the Internet; see Section 2.5, and one that is incomplete, as not for all persons it is known who their (biological) parents are. The network also contains errors, as in some cases the person who is registered as a (biological) parent of a child may in fact not be so. In fact such a person might not even be aware of this.

²⁾ In practice one would use routers and hubs as well. But for simplicity we assume there is only one type of point. A more realistic picture is one of a network of hubs and routers, and each unit/computer is connected to exactly one hub or router.

³⁾ The tacit assumption is that bidirectional communication is possible.

⁴⁾ A unit/PC has one socket, and a hub or router has several sockets.

⁵⁾ Since there also exist dynamically created webpages that are produced as a result of a query, and typically do not persist for very long, the situation is even more confusing. But we consider the more stable part of the internet, consisting of webpages that exist for a longer time.

⁶⁾ There is a trade-off: observing in a very short time gives a more accurate picture of the internet at that time, but this portion is small compared to the whole thing. Or it is big, in which case it is not a crisp and sharp picture at an instant, but a blurred one taken over a sizeable time interval.

2.7 Human society

The points are persons in some society or community, which could be a country, a village, a school. The relationship studied could be that of 'being a friend of'. The corresponding network is a digraph. If A declares themselves to be a friend of B then it is not necessarily the case that B considers A to be their friend as well.⁷⁾ So the relationship 'being a friend of' is not necessarily symmetric.

2.8 Businesses

The points in the network are businesses in a country. An arc connecting one point A with another point B means that a payment of A to B has been made for a delivery or service from B for A. Such a network would bring to light how businesses interact with each other. Using this information, it is possible to find the businesses that are hubs in the sense that they are either big receivers or big spenders (or both).

2.9 Cartographic maps

We consider maps that divide an area into sub-areas. Think of a piece of land divided into different parcels, each of which is owned by some individual. We view this division of parcels of land as a partitioning of the piece of land. Intuitively some subdivisions look more complex than other. The question is how to capture this kind of complexity. We first note that this problem can be turned into a graph problem. Each parcel of land is represented by a node. Two nodes are connected by an edge if the corresponding parcels of land have a (1-dimensional) boundary in common. So if they only have a point in common they are not joined by an edge. It is clear that we are dealing with a topological property and not a geometric one: the sizes of the plots of land are immaterial. If they are blown up or shrunk, they still do (or do not) have 1-dimensional boundaries in common. Multiplication by a factor of 0 is not allowed, as this would not be a transformation that leaves the topology invariant: any configuration is transformed to a point.

3 Complexity of networks

We propose to consider the complexity of networks in two steps. First we consider graphs and then, as an extension, digraphs. A graph is an easier object than a digraph. Think of a street network consisting of two-way streets only. It is easier planning a trip from one point a to another point b in such a graph, which we assume to be connected. In this case one knows that there exists a route leading from a to b . Also if one has found a route from a to b then one automatically has a route from b to a , namely the reverse route.

⁷⁾ This seems to be related to social status. If B is ranked higher in a social group than A then B might not count A as a friend, whereas A may view B as a friend. People generally prefer to be associated with persons with at least their social status in a social group.

However in a streetplan with many one-way streets and few two-way streets, it is not certain that a path from node a to node b exists in the digraph. And also, if one has found such a path then it is not certain that a reverse path from node b to node a exists in the digraph. The reverse path is not necessarily an admissible solution. Even if a path from node b to node a exists in the network, it may be very different from the path from a to b , in the sense that it may have few nodes in common.

To make the discussion sensible, one should look at the right classes of graphs and digraphs. For graphs we look at sets with the same number of edges (and a variable number of nodes). The complexity is determined by the distribution of the edges over the network, and hence by its topology. In fact, the average degree of the nodes is taken as a measure of the complexity of a graph.

For digraphs we consider reachability as a property to determine the complexity. Which nodes can be reached from a given node? And what can be said about the reachability of an average node? This information characterizes the complexity of a digraph. For a complex digraph the reachability from a node may vary considerably. In a digraph with full reachability – in which every node can be reached from every other node – the complexity is small. Each graph has the full reachability property, which indicates that reachability is not a suitable complexity measure for graphs, as it does not discriminate among them. But reachability does for general digraphs. In fact full reachability is equivalent to the transitive closure of the adjacency matrix being equal to the ‘all 1s matrix’, that is $A^* = J$, where J is the matrix with all elements equal to 1 and of the same order as A (namely $|V|$).

For a digraph without full reachability intuitively a measure for complexity could be the proximity to a digraph G with this property, proximity being measured by the number of arcs added to G . This adding can be done deterministically in which case we want to add as few arcs as possible. The question then is which arcs to add.⁸⁾ There are other ways to use reachability to define the complexity of digraphs, several of which are discussed in Section 9. In Section 9.6 a probabilistic method is proposed to obtain a complexity measure for digraphs without full reachability.

Although these approaches are intuitively justifiable, they have the drawback that they are computationally demanding. It would be attractive to have a simpler way to compute the complexity of a digraph. One way to do this is by looking at the reachability sets for the nodes, in particular their sizes. In case $G = (V, E)$, where V is the set of nodes and E the set of edges, is fully reachable, they all have the same size namely $|V|$. So the size distribution is one with all mass (equal to 1) concentrated in one point (namely $|V|$). The entropy of this distribution is 0. For a digraph which is not fully reachable, there is a nondegenerate size distribution of the reachability sets. The entropy measures how strongly this distribution deviates from the distribution with all its mass concentrated at a single value. It is maximum for uniform distributions (that is, for various values of n).

⁸⁾ Adding arcs probabilistically was also suggested as an option in the first version of the present paper. But this does not make a lot of sense as certain arcs need to be added to obtain full reachability. Therefore this ‘option’ has been dropped in the current version of the paper.

4 Complexity of graphs using average degree

4.1 Average degree of a graph

In topology the concept of connectedness is based on the idea of separability. If a topological space is the union of two nonempty, disjoint open sets it is disconnected. In a connected topological space such a decomposition is not possible. In the network context the equivalent of a disconnected space is that of a network consisting of two or more connectivity components. The idea of a connectivity component is based on the concept of path connectedness in networks.

It is important to consider, for a moment, connectivity in graphs and digraphs. In fact the digraph case is the general one, if a graph is viewed as a digraph in which each edge is represented by two arcs of opposite orientation. A digraph is connected if for each pair a and b of its nodes, there is a path from a to b . A path is a finite sequence of arcs in the digraph, where the tail of the first arc is a and the head of the final arc coincides with b and for subsequent arcs (a, b) and (c, d) holds that $b = c$, provided that (a, b) is before (c, d) . As noted before, in a digraph there may be a path from a to b but not a path from b to a . And if there is a path from b to a it may not be the reverse of the path from a to b .

A topological space is path connected if any two of its points can be joined by a continuous path that is entirely within this topological space. Every path connected space is connected, but the converse is not necessarily true. We shall not elaborate these concepts here as they are, in their generality, not of interest to us.⁹⁾ We will concentrate on path connectivity in networks—which is simply referred to as connectivity—as it is the only concept concerning connectivity in networks that we use in the present paper.

An evident measure for the complexity of networks is related to the way nodes in a network are connected. Intuitively, the more branching exists in a network, the more complex it is. But how should this be quantified? Here we want to take a closer look at this problem.

To make things comparable we look at the class \mathcal{C}_m of graphs with the same number m of edges for $m \in \mathbb{N}$. This is a good way to compare graphs. The complexity now depends solely on how the edges are interconnected, within each class \mathcal{C}_m . Within such a class the number of points control the complexity: the fewer nodes are present, the higher the complexity of the graph, and vice versa.

As a candidate complexity measure we now look at the average degree of a graph. This is defined as the ratio of the sum of the degrees of the nodes in a graph, divided by the number of nodes in this graph. So if $G = (V, E)$ is a graph in \mathcal{C}_m with n nodes, the average degree is

$$\Delta^{av} \triangleq \frac{\Delta^{tot}}{n}, \quad (1)$$

⁹⁾ The interested reader should consult a book on general topology.

where Δ^{tot} is the sum of degrees of all the nodes in G . But this is $2m$ for all graphs in \mathcal{C}_m . This is easy if one looks at the adjacency matrix of a graph, which is a symmetric $(0,1)$ -matrix. So we find

$$\Delta^{av} = \frac{2m}{n}. \quad (2)$$

So in the class \mathcal{C}_m the number of nodes n is the only free parameter. This parameter controls not only the complexity of the graph, but also a topological property such as the number of connectivity components. We consider this aspect in the sequel, but for the moment we concentrate on average degrees.

In Figure 4.1 there are twelve examples of graphs in \mathcal{C}_{12} , ordered by nondecreasing average degree. For each node the degree is given (in blue) as well as the average degree (Δ^{av}).

We are interested in the graphs in \mathcal{C}_m with the smallest average degree and the highest average degree. This is equivalent to graphs in \mathcal{C}_m with the highest number of nodes, and the smallest number of nodes, respectively, which in turn, should correspond to graphs with the lowest complexity and the highest complexity, respectively. This is indeed what Figure 4.1 shows in case of \mathcal{C}_{12} .

We can view this as a measure of density: how many nodes are used in a graph to accommodate m edges? We can compare this with the minimum number of nodes needed to put in m edges. How to compute this characteristic as a function of m , is discussed below.

Smallest average degree is obtained when the graph with m edges is totally unconnected (consists of m unconnected edges), and so has $2m$ nodes. In case of \mathcal{C}_{12} the graph in Figure 4.1 on the top left position is an example of such a graph. The graph in \mathcal{C}_m with the largest average degree, is the most compact one, that is, with the fewest number of nodes. Since we do not allow loops or parallel edges in a graph, a nontrivial most compact graph with m edges exists.¹⁰⁾ In Figure 4.1 the most compact graph (and largest average degree) is pictured at the bottom right, in case the graph has 12 edges.

We now construct a graph in \mathcal{C}_m with a minimum number of nodes and with m edges. For a graph with m edges determine $n_m \in \mathbb{N}$ such that

$$\binom{n_m}{2} \leq m < \binom{n_m + 1}{2}. \quad (3)$$

If

$$\binom{n_m}{2} = m \quad (4)$$

¹⁰⁾ Of course, if they would be allowed, a 'most compact' graph would still exist: it would be a generalized graph with one node and m parallel loops.

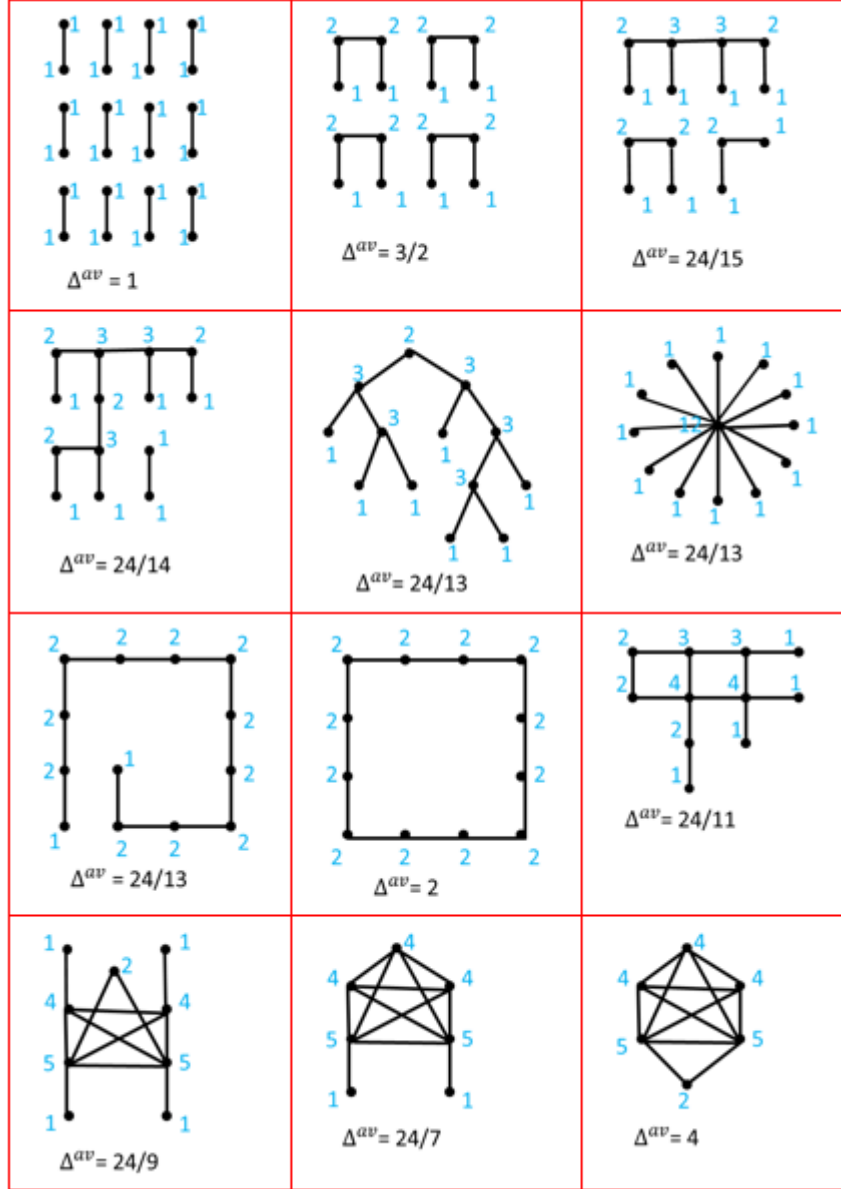


Figure 4.1 Examples of graphs in \mathcal{C}_{12} in increasing order of complexity, based on the average degree (Δ^{av}). The graphs with the lowest complexity ($\Delta^{av} = 1$) and with the highest complexity ($\Delta^{av} = 4$) in \mathcal{C}_{12} are included.

then we are done: take the full graph \mathcal{F}_{n_m} with n_m nodes. In this case m , as defined in (4), is a triangular number.

If

$$\binom{n_m}{2} < m < \binom{n_m + 1}{2}. \quad (5)$$

take the full graph \mathcal{F}_{n_m} with n_m nodes and expand it by first adding one new node v to its node set and then connect this to $m - \binom{n_m}{2}$ nodes in \mathcal{F}_{n_m} . This can usually be done in a variety of ways. In any case, the resulting graph has m edges and $n_m + 1$ nodes. This number of nodes is, obviously, the minimum number of nodes for the graphs in \mathcal{C}_m .¹¹⁾

To see how n_m depends on m assume for the moment that $m = \binom{n}{2}$ for some n . This means that the graph with minimum number of nodes is a full graph on n points. So we have the equation

$$n(n-1)/2 = m, \quad (6)$$

in which m is given and n has to be determined. The solution is

$$n_m = \frac{1 + \sqrt{1 + 8m}}{2} \quad (7)$$

assuming the expression on the right-hand side of (7) is in \mathbb{N} .

An interesting aspect is also how the connectivity of the graphs in \mathcal{C}_m changes when the number of nodes is varied. When this number is the highest possible (namely $2m$) so is the number of connectivity components (namely n). When this number is the lowest possible (namely $n_m + 1$ with n_m as in (3)) there is only one connectivity component. For values of n between those extremes it is more complicated to say what the number of components is. However to find this number the so-called graph Laplacian can be used, which is the matrix

$$\Delta \triangleq D - A, \quad (8)$$

where $D = (d_{ij})$ is the degree matrix of the graph, defined as follows

$$\begin{aligned} d_{ij} &= \text{the degree of } i, \text{ if } i = j, \\ &= 0, \text{ if } i \neq j, \end{aligned} \quad (9)$$

¹¹⁾ This is the case precisely because loops and parallel edges are forbidden in the graphs we consider.

and A is the adjacency matrix of the graph. The dimension of the kernel of Δ , symbolically $\dim(\text{Ker}(\Delta))$, equals the number of connectivity components of the graph.

4.2 A refinement

Here we want to apply the complexity concept introduced in Subsection 4 to an equivalence class of graphs, namely those which are the same modulo linear subgraphs. Put another way, given a graph $G = (V, E)$ we look for its compressed form G^c . G^c is obtained from G by replacing all linear subgraphs by a single edge. A linear subgraph $L(a, b)$, where a and b are nodes in G , is a path from a to b , say (n_1, \dots, n_k) , where $k \in \mathbb{N}$, $k \geq 3$, $n_1 = a$ and $n_k = b$, and $\Delta(n_j) = 2$ for $j = 2, \dots, k-1$, where $\Delta(v)$ denotes the degree of node $v \in V$, that is the numbers of arcs incident with v . Each G has a compressed form G^c , which is unique.

The idea now is to base the complexity $\kappa^{av,c}(G)$ of a graph G on the complexity of its compressed form G^c :

$$\kappa^{av,c}(G) \triangleq \Delta^{av}(G^c). \quad (10)$$

The idea behind this complexity measure is that linear (sub)graphs are of the same complexity as an edge, no matter which size they have. The point is that they do not branch. By replacing each of them by any (finite) linear graph leaves the complexity unchanged. So one can just as well replace each of them by an edge. This is precisely the compressed form of the original graph that one obtains. Two graphs G_1 and G_2 which have the same compressed form, i.e. for which $G_1^c = G_2^c$, also have the same $\kappa^{av,c}(G)$ -complexity, that is complexity as defined in (10). Obviously, the relation 'having the same compressed form' is an equivalence relation on the class of graphs.

$\kappa^{av,c}(G)$ -complexity as defined in (10) is more difficult to apply than Δ^{av} in (1) and it also lacks the nice property to look at 'natural' and simple class of graphs, namely those with the same number of edges. Despite these 'drawbacks' we believe that $\kappa^{av,c}(G)$ -complexity is superior to Δ^{av} -complexity. Since it is simpler to apply we concentrate on Δ^{av} -complexity in the remainder of the present paper. However, $\kappa^{av,c}(G)$ -complexity deserves further exploration.

4.3 Reducing graphs

We can reduce graphs through the removal of nodes or arcs in two ways. We have a general way (Reduction method A) and a strict way (Reduction method B) to reduce graphs. In the general reduction method the reduced graph may be disconnected whereas the original graph is connected. In the strict way this is impossible. In fact this method is aimed at preserving connectedness in the reduction process.

Let $G = (V, E)$ be a graph, with both V and E nonempty .

Reduction method A

- If v is removed from V and $v \in e$ then e is removed from E as well.

Reduction method B

- If v is removed from V and $v \in e$ then e is removed from E as well.
- If e is removed from E and $v \in e$ and $\Delta(v) = 1$ then v is removed from V as well.

So both methods agree on node elimination but differ on edge elimination. Method B has a rule concerning the removal of edges, whereas Method A does not: an edge can be removed without any consequence for its incident nodes. So in case edge $e = \{v, w\}$ is removed according to Method A and $\Delta(v) > 1$ and $\Delta(w) = 1$ then the reduced graph has node w as an isolated node, i.e. with $\Delta(w) = 0$. If the removal of e is done according to Method B, node w will also be removed.

4.4 Exploring graph complexity

In Section 4 we proposed the average degree as a measure of complexity for graphs. We provided some examples that supported the idea that it indeed quantifies an intuitive notion of graph complexity. But here we want to look more closely at this measure.

First we introduce some notation. If $G = (V, E)$ a graph then a subgraph $G_S = (V_S, E_S)$ of G is a graph with $V_S \subseteq V$ and $E_S \subseteq E$. We write $G_S \subseteq G$ in case G_S is a subgraph of G .

We consider the following two questions:

1. Does $G_S \subseteq G$ imply $\Delta^{av}(G_S) \leq \Delta^{av}(G)$?
2. For $q \in \mathbb{Q}^+$ is there a graph G with $\Delta^{av}(G) = q$?

The first question asks if a subgraph has lower complexity, using the 'average degree' as the complexity measure. As removal of arcs (directly or indirectly via removal of nodes) produces a graph in a different class of graphs (one with the same number of nodes as the reduced graph), anything can happen. We illustrate this with a few examples in Figure 4.2, where the reduction consists of removing an edge and possibly also a node on the edge.








Before reduction	After reduction
$\Delta^{av} = 1$ 	Δ^{av} not defined for empty graph
$\Delta^{av} = 2$ 	$\Delta^{av} = 2$ 
$\Delta^{av} = 2$ 	$\Delta^{av} = 1\frac{1}{2}$ $\Delta^{av} = 1\frac{1}{2}$ $\Delta^{av} = 1\frac{1}{2}$ 
$\Delta^{av} = 2\frac{2}{5}$ 	$\Delta^{av} = 2\frac{1}{2}$ 

Figure 4.2 Examples of graph reductions (following method B) and their effect on Δ^{av} .

In Figure 4.2 four types of reductions are shown, producing all the possible outcomes for the complexity of the reduced graph compared to that of the original graph. In the top row a graph is shown consisting of a single edge. If this is reduced the empty graph is produced, for which Δ^{av} is not defined. The second row shows an example of a graph with the same complexity as its reduction. In the third row examples of reductions are shown with a lower complexity than the original graph. The bottom row shows an example of a graph with a lower complexity than its reduction. In this case the reduction decreases both the number of edges as well as the number of nodes by 1.

We now turn to the second problem raised at the beginning of the present subsection: can we produce a graph with a given complexity, assuming the number specified is in the appropriate range? The answer turns out to be affirmative if q has the correct form $q = 2m/n$, for some integers $m, n > 0$. We construct two graphs, depending on q , as follows:

- $q < 1$. Start with a graph consisting of m disconnected edges, and hence, $2m$ nodes. Now create a new graph by adding $n - 2m > 0$ nodes.
- $q \geq 1$. Start with a complete graph with n nodes, and hence $\binom{n}{2}$ edges. We may assume that $m \leq \binom{n}{2} = n(n-1)/2$. Delete $n(n-1)/2 - m \geq 0$ edges from the complete graph.

In both cases we have constructed a graph with m edges and n nodes and hence with the requested complexity q .

4.5 Glueing

To illustrate the idea to build new graphs by identifying nodes from given, simple, linear graphs (edges), consider Figure 4.3. This shows a powerful mechanism to produce graphs from simpler graphs by a kind of glueing process, which is the identification of different nodes to represent the same node.

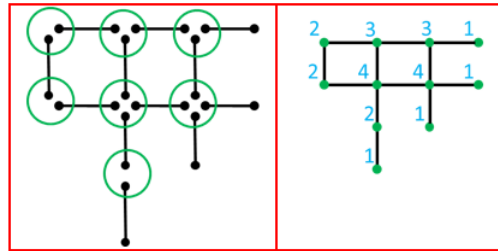


Figure 4.3 On the left-hand side is shown how to produce a graph from edges using identification of nodes. The nodes in each green circle are identified as a single node. The resulting graph is depicted on the right-hand side.

A similar glueing process can be defined for edges from different graphs, instead of nodes. In this case edges are identified. If $\alpha = \{a, b\}$ and $\beta = \{c, d\}$ are edges that are to be identified, there are two possibilities:

1. a and c are identified and b and d are identified, or
2. a and d are identified and b and c are identified.

Of course, if two graphs are glued together one needs to specify which nodes are identified or which pairs of nodes. The glueing process is represented by an operator $*$, or $*_{v_1, v_2}$ or $*_{e_1, e_2}$, where v_1, v_2 are nodes to be identified and e_1, e_2 are edges to be identified. In the latter case it should be indicated how the edges should be identified, i.e. which nodes should be identified. This glueing has the following algebraic properties:

1. $G * U = G$, where U is the graph existing of one node and no edges. (*Existence of a unit element*)
2. $G_1 * G_2 = G_2 * G_1$. (*Commutativity*)
3. $(G_1 * G_2) * G_3 = G_1 * (G_2 * G_3)$. (*Associativity*)

It is interesting to study the complexity of the graph that emerges when two graphs are glued, either by glueing nodes or edges. It is simply a matter of counting nodes and edges. Let $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ with $|V_i| = n_i$ and $|E_i| = m_i$ are graphs with disjoint node and edge sets, that are glued together by identifying a node from G_1 and one node from G_2 to obtain $G_1 *_n G_2$, then

$$\Delta^{av}(G_1 *_n G_2) = \frac{2(m_1 + m_2)}{n_1 + n_2 - 1}. \quad (11)$$

Likewise, if we glue G_1 and G_2 by identifying an edge from G_1 and an edge from G_2 to obtain $G_1 *_e G_2$ then

$$\Delta^{av}(G_1 *_e G_2) = \frac{2(m_1 + m_2 - 1)}{n_1 + n_2 - 2}. \quad (12)$$

Note that the complexity in both glueing operations $*_n$ and $*_e$ does not depend on which nodes or edges are identified.

5 Complexity of graphs based on search

Here we consider another approach to complexity. The intuition behind it is that of an efficient search through a maze or labyrinth. The begin and end points are chosen in such a way that

1. The search is continuous.
2. The search covers all edges of the graph.
3. The path associated with the search should be of minimal length.

A continuous search in a graph is one so that it follows a path in the graph. This means that the transition from a node v to the next one w is only possible if $\{v, w\}$ is an edge in the graph. Given a starting point s and a finishing point f the path followed by the (continuous) search procedure should be as short as possible, under the restriction that all edges in the graph are

visited. The restriction is important. And it should be stressed that all edges should be visited at least once, not just the nodes. Visiting all edges implies visiting all nodes, but not the other way round. In general, shortest path problems do not have this restriction of visiting all edges in the graph at least once. But for us it is essential. The shortest path under this restriction depends on the nodes s and f . The goal is to find a combination for which the length is minimal. We assume that the edges have equal length 1. We call this metric a natural metric.

We now consider some simple examples. In the graph on the left-hand side of Figure 5.1 the path $((1, 2), (2, 3), (3, 4), (4, 5), (5, 6))$ from node 1 to node 6 has minimal length (namely 5) and contains all edges (at least) once. In the graph on the righthand side of Figure 5.1 the path $((1, 2), (2, 3), (3, 4), (4, 5), (5, 6), (6, 1))$ is a path from node 1 to node 1 of minimal length (equal to 6) containing all edges of the graph. So the minimal paths we were looking for are different, because it was requested that all edges should be on the path. If it was only required that all nodes should be situated on the path, the path $((1, 2), (2, 3), (3, 4), (4, 5), (5, 6))$ would do for both graphs. Consequently both graphs in Figure 5.1 would have had the same complexity. But from a search perspective these graphs are quite different. The one on the left has two end points, whereas the one on the right has none. By requesting that the path contain all edges (at least once) we obtain this distinction. Of course, for the graph on the right-hand side, from each node one can walk around in two directions. If one node (or edge) would be blocked one can then still walk around it by going in the opposite direction. This is not possible in the graph on the left-hand side. If nodes 2, 3, 4 or 5 are blocked, it is as if the graph is cut into two separate pieces.

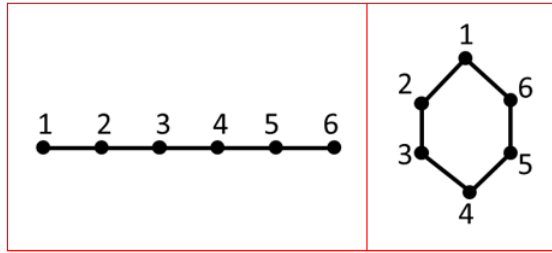


Figure 5.1 Two graphs, G_l on the left and G_r on the right, with the same number of nodes but with different topologies.

We define a formal complexity measure κ_s for a connected graph $G = (V, E)$ with $n = |V|$ and $m = |E|$ by comparing the length λ_G of an optimal path (satisfying the three requirements above) with that of a tour covering all edges starting at some node $v \in V$, which has length τ_G . So we can write

$$\kappa_s(G) \triangleq \frac{\lambda_G}{\tau_G} \quad (13)$$

For the graph G_l on the left-hand side in Figure 5.1 an optimal path covering all edges is $((1, 2), (2, 3), \dots, (5, 6))$ which has length $\lambda_{G_l} = 5$ and a tour covering all edges starting at one is $((1, 2), (2, 3), \dots, (5, 6), (6, 5), \dots, (2, 1))$ which has length $\tau_{G_l} = 10$. So $\kappa_s(G_l) = 5/10 = 0.5$. For the graph G_r on the right-hand side of Figure 5.1 an optimal path covering all edges is $((1, 2), (2, 3), \dots, (6, 1))$ which has length $\lambda_{G_r} = 6$ and an optimal tour starting at 1 covering all edges, is the same path, so that $\tau_{G_r} = 6$. Hence $\kappa_s(G_r) = 6/6 = 1$.

6 Complexity of graphs based on average distance

In Section 5 a metric on a graph was used to define a measure of complexity. In the present section we also use the natural metric to obtain a complexity measure, where each arc has length 1. For a connected graph $G = (V, E)$ with $|V| = n$ we introduce the distance matrix

$$D = (d_{ij}), \quad (14)$$

where d_{ij} is the minimum length of the path in G connecting nodes i and j . The length of a path is computed in terms of the number of edges it contains. For each of these graphs we compute the average distance for any pair of (different) nodes. The distance matrix contains the adjacency matrix as a submatrix. We obtain the adjacency matrix A from D by setting all entries $d_{ij} > 1$ to 0.

We define the average distance over all different pairs of nodes in a (connected) graph G as a complexity measure $\kappa_d(G)$:

$$\kappa_d(G) \triangleq \frac{\sum_{i < j} d_{ij}}{\binom{n}{2}}, \quad (15)$$

where the sum is over the n nodes in G .

In mathematical chemistry the Wiener¹²⁾ index (cf. [9]) is used to measure the branching of organic molecules. It is defined as the total distance of any pair of carbon atoms in such a molecule, which together form its skeleton, so to speak. So, if $\mathcal{W}(G)$ denotes the Wiener index then

$$\mathcal{W}(G) \triangleq \sum_{i < j} d_{ij} = \binom{n}{2} \kappa_d(G), \quad (16)$$

where the sum is over the nodes in G , or rather the upper triangle of its (natural) distance matrix.

We now consider a variant of the complexity measure based on the average distance of different pairs of nodes in a graph $G = (V, E)$. It is based on the distance of edges in G . To be able to do this we use the line graph $G_L = (V_L, E_L)$ derived from G . The nodes of G_L are the edges of G , that is $V_L \triangleq E$. Edges $e, f \in E$ (nodes in V_L) form an edge in G_L iff $e \cap f \neq \emptyset$. So

¹²⁾ Named after the chemist and physician Harry Wiener. He introduced this measure (which he called the Path Number) to quantify the branching of organic molecules. His goal was to relate this property at the molecular level to macroscopic physical properties of the substances consisting of these molecules. Alongside the Path Number he introduced the Polarity Number which is the number of pairs of carbon atoms separated by three carbon atoms.

$E_L \triangleq \{(e, f) \in E \mid e \neq f \text{ and } e \cap f \neq \emptyset\}$. We then apply the complexity measure (15) to G_L instead of to G , but view it as a complexity measure for G . So formally, we then have

$$\kappa_{d^\ell}(G) \triangleq \frac{\sum_{i < j} d_{ij}^\ell}{\binom{m}{2}}, \quad (17)$$

where the d_{ij}^ℓ 's are the 'natural' distances in the line graph G_L of G . The sum is over the nodes of G_L , which are the edges in G .

In Appendix B some example graphs are used to compute the quantities defined in (15), (17) and (16). Comparing the results is difficult if the number of edges in G is different. This is familiar from Section 4.

From the examples in Appendix B it is clear that the Wiener index (16) itself does not qualify as a complexity measure. It is an ingredient of the complexity measure (15). But there is no particular reason to single out this component. The complexity measure (15), which is the average distance for different nodes in the original graph, seems to agree with the intuition about complexity. However, applying this measure to the line graphs associated with graphs, the results are less convincing. It seems that the reciprocal of this measure for line graphs does a better job, at least for the examples in Table B.2. This suggests that the average of the reciprocal distances:

$$\frac{\sum_{i < j} (1/d_{ij}^\ell)}{\binom{m}{2}} \quad (18)$$

could do well too as a measure for graph complexity. Note that (18) is the reciprocal of the harmonic mean of the distances d_{ij}^ℓ .

7 Complexity of routing digraphs

In [13] (Appendix B) the total number of paths from source to sink in routing digraphs has been used to define the complexity of the routing structure of a questionnaire.¹³⁾ A routing digraph is an acyclic digraph with a single source and a single sink. In [13] such digraphs were called 'routing graphs', somewhat confusingly, as this was the term that was used in practice. We have changed this name to 'routing digraph' to stress that we are dealing with *directed* graphs, not graphs. We denote the class of such digraphs by \mathcal{Y} . An example of a simple routing graph is shown in Figure 7.1. Its source (node 1) is coloured green and its sink (node 7) is coloured brownish red.

In fact the (natural) logarithm of the number of paths from source (i.e. the first question) to sink (i.e. the final question) is defined in [13] as the complexity of the routing structure of a

¹³⁾ The routing structure defines which next question should be posed depending on the answer to the last question asked to a respondent.

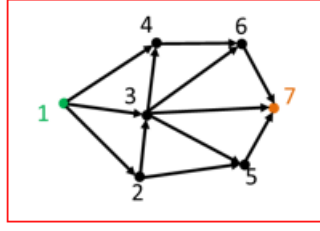


Figure 7.1 Example of a routing digraph.

questionnaire. If we denote by $\pi(G)$ the number of paths from source to sink in G we define the complexity $\kappa_Y(G)$ of G as

$$\kappa_Y(G) \triangleq \log \pi(G). \quad (19)$$

In (19) ‘log’ denotes the natural logarithm.¹⁴⁾ It turns out that this complexity measure has some nice properties (see [13], Appendix B), that we present here, after introducing some convenient notation.

Let G_1 and G_2 be two routing digraphs. Then we can form a new digraph with a single source and sink $G_1 \odot G_2$ by identifying the sink of G_1 with the source of G_2 .¹⁵⁾ To distinguish it from the glueing of routing digraphs we call the latter operation \odot -glueing. An example of \odot -glueing is presented in Figure 7.2.

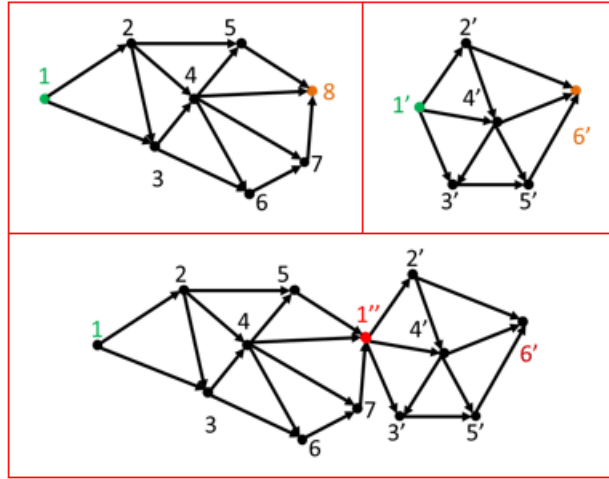


Figure 7.2 \odot -glueing two routing graphs.

In Figure 7.2 the two routing digraphs to be \odot -glued are in the top row. The result is represented in the bottom row. The \odot -glueing was done by identifying the sink of the routing digraph on the

¹⁴⁾ In [13] the logarithm at base 2 was taken. The complexity measure was denoted by ‘ ρ ’ instead of ‘ κ_Y ’. The resulting complexity measures are equivalent, in the sense that they differ by a constant.

¹⁵⁾ In [13] this operation is symbolized by ‘ $*$ ’ instead of \odot . But in the present paper ‘ $*$ ’ is already used for another, similar operation, called glueing (of graphs), defined in Section 4.5.

top left-hand side of Figure 7.2 with the source of the routing digraph on the top right-hand side of this figure.

If G is an acyclic digraph with a single source and a single sink, so does G^\leftarrow , the inverse digraph of G that we get when we reverse the direction of each arc in G . In Figure 7.3 the reverse of the routing digraph in Figure 7.1 is shown.

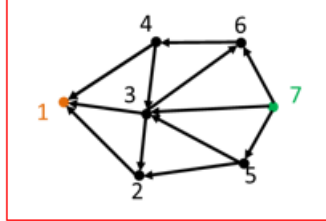


Figure 7.3 Reverse of the routing digraph in Figure 7.1.

Then we have (cf. [13], p.136):

1. $\kappa_Y(G_1 \odot G_2) = \kappa_Y(G_1) + \kappa_Y(G_2)$
2. $\kappa_Y(G_2 \odot G_1) = \kappa_Y(G_1 \odot G_2)$.
3. $\kappa_Y(G^\leftarrow) = \kappa_Y(G)$.
4. $\kappa_Y(G_1) \leq \kappa_Y(G_2)$ if G_1 is in Y and a subdigraph of G_2 .
5. $\kappa_Y(G_0) = 0$, if G_0 is a point graph, consisting of a single node.
6. $\kappa_Y(G)$ is invariant under contractions of G .

A contraction of a digraph G is a replacement of a linear subdigraph of G by a simpler linear structure, such as a node or an arc or two arcs, depending on the situation. The result of the contraction should be a digraph. This implies that it is not possible to have parallel arcs between two nodes. A maximally contracted digraph is one that cannot be contracted any more. So a digraph \bar{G} is maximally contracted if $\varpi(\bar{G}) = \bar{G}$, for any contraction ϖ of \bar{G} . In other words, \bar{G} is a fixed point for any contraction operator ϖ operating on \bar{G} . In Figures 7.4, 7.5 and 7.6 several examples are shown of contractions of digraphs.

Figure 7.4 shows three contractions, starting with the digraph at the top left-hand side. In each contraction step one arc is removed. The digraph at the bottom right-hand side is maximally contracted. Note that in this case a linear subdigraph is replaced by a single node.

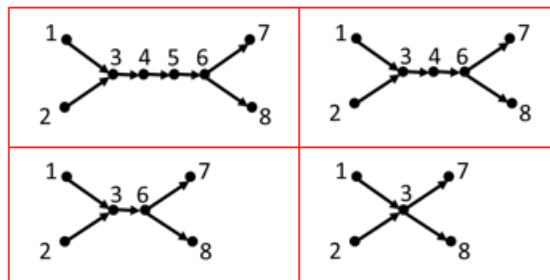


Figure 7.4 Contracting a digraph in a number of steps.

In Figure 7.5 the digraph on the left-hand side is contracted to that on the right-hand side. The blue and the yellow linear parts on the left-hand side are replaced by blue and yellow arcs shown on the right-hand side. So in this case two linear subdigraphs are replaced by two arcs.

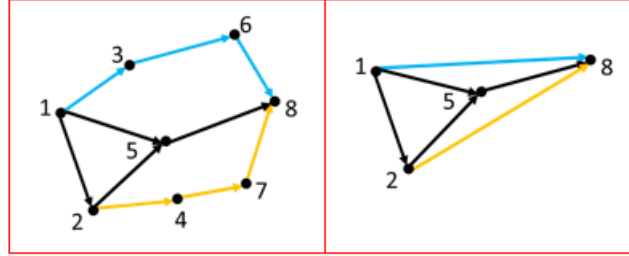


Figure 7.5 Contracting a digraph in two locations.

Figure 7.6 is like the digraph in Figure 7.5 except that the arc $(2, 5)$ is removed. In this case there are three linear subdigraphs that can be contracted. But because the result has to be a digraph, we cannot replace each of these subdigraphs by arcs, because they would be parallel. So one of them is replaced by a single arc (the red part) and each of the other two by two arcs (the blue and yellow parts). Of course, it would also have been possible to replace the blue part by a single arc and the remaining red and yellow parts by a single arc. Similarly could the yellow part be replaced by a single arc and the red and blue parts by two arcs each. This shows that maximal contraction may not lead to a unique result.

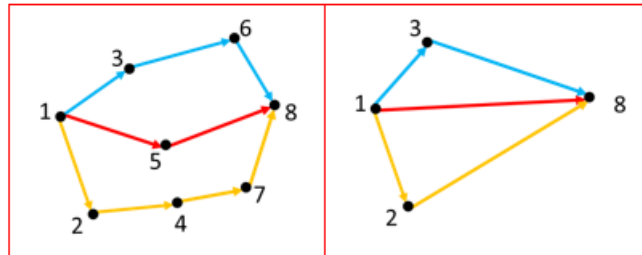


Figure 7.6 Contracting a digraph like that in Figure 7.5 but with arc $(2, 5)$ removed.

As the examples in Figures 7.5 and 7.6 show removal of a single arc can have quite an impact on the contraction result. These examples also show that a contraction of a digraph is non-unique. But the topology, i.e. the cycle structure of the underlying graph is the same. Contracting is also non-invertible in the sense that a whole class of digraphs may yield the same contraction.

We can compute $\kappa_Y(G)$ algebraically, using the adjacency matrix. Let A be the $m \times m$ adjacency matrix of an acyclic digraph. Then there is an n such that $A^n = 0$ and $A^{n-1} \neq 0$, which in fact says that G is nilpotent. Then we have that¹⁶⁾

$$\kappa_Y(G) = \log \left((I_n - A)^{-1} \right)_{1,n}. \quad (20)$$

¹⁶⁾ cf. [13], p. 135

8 Complexity of digraphs based on arc symmetry

Let $G = (V, E)$ be a digraph with an $n \times n$ adjacency matrix A . An aspect that determines the complexity of G is the degree to which it is asymmetric, in terms of arcs (a, b) with or without a counter-arc (b, a) . To quantify this arc symmetry we define:

$$\Theta(A, A') \triangleq \max\{A, A'\} - \min\{A, A'\}. \quad (21)$$

$\Theta(A, A')$ is the $n \times n$ matrix with entry (i, j) equal to $\max\{a_{i,j}, a_{j,i}\} - \min\{a_{i,j}, a_{j,i}\}$. The matrix Θ has the following properties:

- $\Theta(A, A') \geq 0$. (*Nonnegativity*)
- $\Theta(A, A') = \Theta(A', A)$. (*Symmetry*)
- $\Theta(A, A') = 0$ iff $A = A'$ iff G is a graph. (*First lower bound*)
- $\iota'_n \Theta(A, A') \iota_n = 0$ iff $A = A'$ iff G is a graph.¹⁷⁾ (*Second lower bound*)
- $\iota'_n \Theta(A, A') \iota_n \leq n(n-1)$. (*Upper bound*)

The upper bound in the last item is obtained by digraphs with n nodes, for which the underlying graph is complete, that is for each pair of vertices there is an edge, and such that for each edge of the underlying graph $\{a, b\}$ with $a \neq b$ there is exactly one arc, either (a, b) or (b, a) .

We can use Θ to define the following complexity measure for digraphs:

$$\kappa_\Theta \triangleq \frac{\iota'_n \Theta(A, A') \iota_n}{n(n-1)}. \quad (22)$$

Due to the lower bound and upper bound for $\iota'_n \Theta(A, A') \iota_n$ we have $0 \leq \kappa_\Theta \leq 1$. The higher the value of κ_Θ the more complex the digraph is supposed to be. Complexity in this case measures deviance from symmetry under arc reversion. A graph is fully symmetric under arc reversal, and therefore not complex — as a digraph, that is.

It should be noted that κ_Θ as defined in (22) is a rather crude measure of complexity. It does not consider the topology of a digraph. Digraphs with the same number of arcs without counter-arcs (and the same number of nodes) have the same complexity. Other complexity measures that we consider, differentiate among this class of digraphs by using their topology.

For each of the digraphs in Figure 8.1 we have that $\kappa_\Theta = 6/12 = 1/2$. In both cases there are four nodes and three arcs without counter-arcs. The topology of these digraphs, however, is quite different. For instance removal of point 4 in the right-hand side digraph produces a digraph consisting of three separate points. For the digraph on the left-hand side of Figure 8.1, deletion of a node results in digraphs with one or two connectivity components.

¹⁷⁾ $\iota_n = (1, \dots, 1)'$ is the all 1s (column) vector of length n .

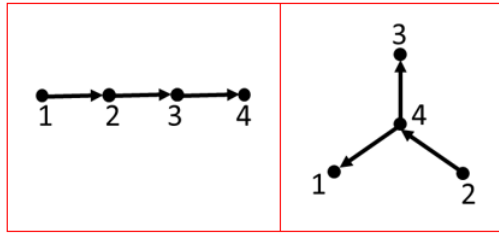


Figure 8.1 Two digraphs with the same κ_{Θ} -complexity.

9 Complexity of digraphs based on reachability

In a digraph, reachability is about the ‘problem’ which nodes can be reached when starting a walk in the digraph at a given node. In graphs, reachability is the same as connectedness. In digraphs the two concepts are different, as a result of the asymmetry in the arcs that are present: in a digraph an arc does not need to have a counter-arc. If one can travel from node a to node b in a digraph it is not necessarily the case that one can also travel from node b to node a . In a graph (viewed as a digraph) each arc has its counter-arc, so that if one can travel from node a to node b , one can then also travel from node b to node a , in fact using the same path in reverse.

It should be stressed that reachability in digraphs is the natural generalization of connectivity in graphs. The counterpart of a connected graph is a fully reachable digraph, in which there is a path $\pi_{v,w}$ connecting v and w , for every pair of nodes v and w . Even if the underlying graph G^{un} of a digraph G is connected, this does not mean that a path $\pi_{v,w}$ exists in G , for any nodes v and w in G . Also the fact that a path $\pi_{v,w}$ exists connecting v with w , does not imply that a path $\pi_{w,v}$ exists. And if it exists, it does not necessarily mean that the reverse path $\pi_{v,w}^{\leftarrow}$ is a path in G from w to v . If there is a path connecting w to v it may be quite different from $\pi_{v,w}^{\leftarrow}$.

So which digraphs are the more complex ones and which are the easier ones? To answer this question think for a moment about the digraph as a street network, consisting of one-way and two-way streets. Navigating with a car in such a network is easiest in case the network consists of only two-way streets and is connected. In such a network a driver knows that he can drive his car from any location (node) to any other location (node) in the network. Quite the opposite is the case when the network has one-way streets only. In that case it is a priori not evident that a particular location (node) can be reached from another location, by car. And if it can be reached, the question is how, by which route? And if it cannot be reached, an obvious question would be which is a nearest location that can be reached by car? The driver, if he was able to drive to the desired location (or one nearby), faces a similar challenge if he wants to drive to his starting location. In theory it is possible that such a route does not exist. But assuming it exists, reversing his original route to the desired location (or a proxy) may not be possible. So it is clear that reachability is a key notion here. And it is also clear that, generally speaking, the more on-way streets the network has, the more difficult it is to answer questions about reachability of destinations in the network.

Of course, if the driver would be able to use the adjacency matrix A of the street network to

compute its transitive closure A^* , these reachability questions would be easy to answer. If reachability is possible, then a little more work would be needed to find out the route that leads to the destination. If the destination is not reachable by car it is even more works to find a reachable node nearby. But all this only underlines the complexity of the undertaking. How easy is it compared to a connected street network with only two-way streets (for cars).¹⁸⁾

This also shows that searchability is also a basis to consider to quantify the complexity of digraphs, as it is for graphs, perhaps even more so. If we continue with the street network example, we would be dealing with this, if no information about the network would be available prior to the trip. One would simply have to proceed by trial and error. If the searching has to be continuous it is more complicated in case of digraphs than it is in case of graphs. The reason is that it is not guaranteed that one can retrace steps at any point during the search. One may even get stuck in a 'trap'.¹⁹⁾ In case the search ends in a trap one should be allowed to resume search at the start location. Without any prior of the network it will be essentially a random search in the network to find a path to the desired location.

We shall not explore searchability of digraphs any further in the present paper, as it is long enough already. But it is certainly worthwhile doing so elsewhere.

9.1 Checking full reachability

When reachability is used as a criterion for digraph complexity, it is in particular checking for full reachability that is called for. So it is important to know how to do this. A straightforward way is to compute the transitive closure A^* of the adjacency matrix A of the digraph, which we assume to be of the order n . This can be done by e.g. Warshall's algorithm for Boolean matrices (proposed in [12]). This algorithm is discussed in e.g. [5] and similar books on algorithms. Warshall's algorithm is essentially repeated matrix multiplication specialised for Boolean matrices applied to the adjacency matrix A . The matrix multiplication is as the 'usual' matrix multiplication but with addition ('+') replaced by the Boolean 'or' ('V') and with multiplication ('×') replaced by the Boolean 'and' ('^'). Full reachability holds if $A^* = J$, where J is the $n \times n$ all 1s matrix.

9.2 Adding the minimum number of counter-arcs

In the present section an approach is presented based on adding a minimum number of arcs to the given digraph in such a way that the resulting digraph is fully reachable. The minimum number of arcs added gives an idea how far the digraph is removed from a fully reachable one. As a measure of complexity of the digraph we take: the more arcs need to be added, the further removed it is from full reachability and hence the more complex it is. In fact, it is not the absolute number of arcs added, but the relative number, compared to the number of arcs of the augmented digraph.

The arcs that are added in the augmentation process are only counter-arcs of existing arcs in the digraph considered. So pairs of nodes v, w where both (v, w) and (w, v) exist are excluded, as is

¹⁸⁾ In practice many drivers nowadays benefit from a navigation system that solves these problems for them. The point is, however, that these navigation problems have to be solved by some party.

¹⁹⁾ In reality it is probably not that bad, but in theory this could be a possibility.

the case for pairs of nodes v, w for which neither (v, w) nor (w, v) exists. We call such pairs saturated. Only those pairs for which either (v, w) or (w, v) exist, but not both, can be used for the augmentation. Such pairs we call unsaturated. If G has v unsaturated pairs, $2^v - 1$ different augmentations are possible with additional counter-arcs.

Using reachability we can define digraph complexity. If the transitive closure of this digraph is not a complete digraph, we want to add arcs (also called augmenting the digraph) such that the resulting digraph is fully reachable. And we want to add as few extra arcs as possible. The more such extra arcs are needed the more complex the original digraph is.

Before we consider the problem in general, we first look at an example, namely the digraph presented in the top left in Figure 9.1. It contains arcs and edges, which can be seen as a convenient simplification for an arc and its counter-arc.

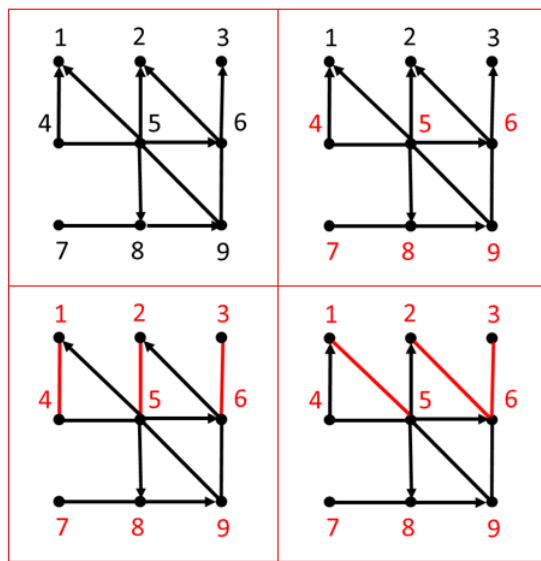


Figure 9.1 At the top left is a digraph that is not fully reachable. On the top right the nodes that can be mutually reached are labeled red and isolated nodes labeled black. The bottom row shows two augmented digraphs which are fully reachable.

The digraph on the top right of Figure 9.1, which is the same digraph as the one on the top left, has the nodes that are mutually reachable labeled red. It is possible to reach the black labeled nodes from each red one, but not vice versa. Also, the black labeled nodes are isolated, in the sense that from none of them one can reach any of the remaining ones.

A simple way to make all nodes communicate is to add counter-arcs to some of the arcs pointing to the isolated points. This is shown in the digraph on the bottom left of Figure 9.1. Another way to do this is shown on the bottom right of Figure 9.1.²⁰⁾

To look at the problem more formally, we first introduce some notation. Let $G = (V, E)$ be a digraph, with $|V| = n$ and $|E| = m$. Let A be the $n \times n$ adjacency matrix. Formally we are looking for an $n \times n$ $(0, 1)$ -matrix B such that the transitive closure of the augmented digraph has full reach. That is, B should be such that

²⁰⁾ It is easy to see that with adding only two counter-arcs full reachability cannot be obtained.

$$(A + B)^* = J, \quad (23)$$

and the number of 1s in B , i.e. the number of arcs, which can be expressed as

$$l' B l \quad (24)$$

should be minimal. J is the $n \times n$ all 1s matrix.

The digraph complexity is defined in terms of the minimum number of arcs that have to be added to G (or, in fact E) so that the augmented digraph G^{aug} has full reach. The more arcs have to be added the higher the digraph complexity of G . This number should be related to the number of arcs in G , i.e. m . If the minimum number of arcs to be added to G is denoted by v_G the digraph complexity based on adding counter-arcs is defined by

$$\kappa_G \triangleq \frac{v_G}{m}. \quad (25)$$

Although the definition of this type of complexity of a digraph is easily stated, the problem of how to actually determine the minimum number of counter-arcs to be added is not so easily solved. The computational complexity of this problem is unknown to the present author.²¹⁾

As the underlying graph of the digraph is assumed to be connected, we can infer that the minimum number of counter-arcs to be added is less than the number of arcs in the digraph without a counter-arc. It is easy to find out if fewer counter-arcs are required. An arc (i, j) may be without a counter-arc (j, i) , it may be possible that there is a path in the digraph from j to i . To find such cases it suffices to compute the transitive closure of the digraph, by computing A^* from the adjacency matrix A . The number of arcs without counter-arcs in the transitive closure of the initial digraph, yields a sharper upper-bound. But is it possible that the minimum number of counter-arcs to be added is strictly less than this?

9.3 Using the size distribution of the reachability sets

In Table 9.1 we have listed the size distributions of the reachability sets of the examples in Appendix A. There are two extreme distributions, namely the ones associated with Figure A.3 and Figure A.27. The first is a uniform distribution over all possible sizes of reachability sets in the corresponding digraph, the other one has all mass concentrated in a single point. In the latter case we are dealing with a digraph with full reachability. For the remaining cases the size distributions are between those two extremes.

We can use the entropy as a measure of the non-uniformity of the size distributions in Table 9.1. The entropies of the distributions listed in Table 9.1 are presented in Table 9.2.

²¹⁾ Perhaps the decision version of the problem stated: 'Can at most k (a specified number) counter-arcs be added to the digraph in such a way that the resulting digraph has full reachability?' is NP-complete.

Size	Fig. A.3	Fig. A.6	Fig. A.9	Fig. A.12	Fig. A.15	Fig. A.18	Fig. A.21	Fig. A.24	Fig. A.27
1	1	2	2	2	1	1	0	0	0
2	1	1	2	1	1	1	0	0	0
3	1	1	0	1	1	1	0	4	0
4	1	1	0	0	1	1	4	0	0
5	1	1	0	0	1	0	0	0	0
6	1	0	0	0	0	0	0	0	0
7	1	0	0	0	0	0	0	0	0
8	1	0	0	0	0	0	0	0	0
9	1	0	5	0	0	6	0	0	0
10	1	5	1	6	6	0	6	0	0
11	1	0	1	1	0	0	0	0	0
12	1	1	1	1	1	2	2	0	0
13	1	1	1	1	1	1	1	9	13

Table 9.1 Size distributions of the reachability sets in the examples (Figures) in Appendix A.

Figure	Entropy
Fig. A.3	1.114
Fig. A.6	0.799
Fig. A.9	0.753
Fig. A.12	0.709
Fig. A.15	0.755
Fig. A.18	0.709
Fig. A.21	0.523
Fig. A.24	0.268
Fig. A.27	0

Table 9.2 Entropies of the distributions in Figure 9.1.

Notice that the entropies of the digraphs represented in the Figures A.3, A.6, A.15, A.9, A.12, A.18, A.21, A.24 and A.27 are a descending series. The smaller the entropy is the closer the corresponding digraph is to a digraph with full reachability, and hence the less complex it is.

The size distribution of the reachability sets as a measure of complexity is not ideal, as it does not involve the interplay of the reachability sets. In theory, it would even be possible that the size distribution of the reachability sets is peaked, even though there is no full reachability. The entropy is small in that case although the complexity of the digraph would be high. But as the example in Table 9.2 shows the measure may also yield sensible results. Further investigation is needed to find out how this measure behaves in general. If it turns out to work it has the advantage that it is quite easy to apply.

9.4 Using the multiplicities of nodes in the reachability sets

In this approach we consider how often a node appears in a reachability set. This yields a distribution over the nodes that we use to derive a complexity measure. In case of full reachability this distribution is uniform. The entropy is maximal for this distribution, equal to $\ln(|V|)$. For all other distributions on the set of nodes it is smaller.²²⁾

²²⁾ In case the digraph consists of isolated points, each reachability set consists of one point only. Reachability sets corresponding to different points are disjoint. The entropy in this case also equals $\ln(|V|)$, although there is no full reachability, to put it mildly.

The distribution we are interested in can be easily computed from the transitive closure A^* of the adjacency matrix A of the digraph as follows:

$$\vartheta \triangleq \frac{\iota' A^*}{\iota' A^* \iota}, \quad (26)$$

where ι is the all 1s vector. The complexity measure proposed is the entropy of the distribution ϑ in (26), i.e.

$$\kappa_E(\vartheta) \triangleq - \sum_{v \in V} \vartheta(v) \ln \vartheta(v). \quad (27)$$

Entropy is used as a measure of uncertainty, as in case of information theory. For a uniform distribution the entropy is biggest, whereas for a peaked distribution (with all probability mass concentrated in a single point) it is minimal, that is, equal to 0.

9.5 Partial order of the reachability sets

In the examples in Appendix A ditrees are presented that show how the various reachability sets are related, or ordered, to be more specific. These ditrees also can be used to get an insight into the complexity of a digraph in terms of reachability. The examples in Appendix A indicate that the smaller the tree, the closer the digraph is to being a digraph with full reachability, which is the least complex case.

First we want to point out an interesting property about reachability sets. Let $G = (V, E)$ be a digraph and let $i, j \in V$ be nodes in G . Suppose furthermore that j can be reached from node i , in notation $i \rightsquigarrow j$. Then $\bar{j} \subseteq \bar{i}$. If also holds that $j \rightsquigarrow i$ then it follows $\bar{i} = \bar{j}$. Here, \bar{i}, \bar{j} denotes the reachability sets of nodes i and j , respectively. This explains why the ‘subset’ property of reachability sets of a digraph leads to a tree: nontrivial cycles cannot exist: they collapse to a point. This implies that the reachability sets of a digraph form a partial order, \leq , interpreted as follows: for sets a, b it holds: $a \leq b$ iff $b \subseteq a$. That the properties for partial orders hold for \leq follows from the following properties for for all $i, j, k \in V$:

1. $i \rightsquigarrow i \Rightarrow \bar{i} \subseteq \bar{i}$.
2. $i \rightsquigarrow j$ and $j \rightsquigarrow i \Rightarrow \bar{i} = \bar{j}$ (since $\bar{j} \subseteq \bar{i}$ and $\bar{i} \subseteq \bar{j}$).
3. if $i \rightsquigarrow j$ and $j \rightsquigarrow k$ then $i \rightsquigarrow k \Rightarrow \bar{k} \subseteq \bar{i}$ (since $\bar{j} \subseteq \bar{i}$ and $\bar{k} \subseteq \bar{j}$).

In terms of \leq these results can be stated as follows:

1. $\bar{i} \leq \bar{i}$ (*reflexivity*)
2. $\bar{i} \leq \bar{j}$ and $\bar{j} \leq \bar{i} \Rightarrow \bar{i} = \bar{j}$ (*antisymmetry*)
3. $\bar{i} \leq \bar{j}$ and $\bar{j} \leq \bar{k} \Rightarrow \bar{i} \leq \bar{k}$. (*transitivity*)

These properties define \leq as a partial order. The ditrees in Appendix A are graphical representations of such partial orders.

We can use the underlying trees of the ditrees representing the partial ordering \leq to define a complexity measure for ditrees. So the complexity measure to be used is given by (1). However, certain nodes (reachability sets) appear multiple times. We can use the multiplicity of each node to weigh its degree. We denote the complexity measure in this case by $\Delta^{av,mult}$. We can also choose to ignore the multiplicity, which actually means that it is taken to be equal to 1 for each node. We then get another complexity measure denoted by $\Delta^{av,1}$. If we apply these measures to (the underlying trees of) the ditrees in Appendix A we find the results in Table 9.3.

Figure	$\Delta^{av,mult}$	$\Delta^{av,1}$
Fig. A.4	$24/13 = 1.846$	$24/13 = 1.846$
Fig. A.7	$28/13 = 2.154$	$16/9 = 1.778$
Fig. A.10	$28/13 = 2.154$	$16/9 = 1.778$
Fig. A.13	$24/13 = 1.846$	$14/8 = 1.750$
Fig. A.16	$19/13 = 1.462$	$14/8 = 1.750$
Fig. A.19	$24/13 = 1.846$	$12/7 = 1.714$
Fig. A.22	$21/13 = 1.615$	$6/4 = 1.500$
Fig. A.25	1	1
Fig. A.28	0	0

Table 9.3 Complexities of the ditrees in Appendix A.

The examples in Table 9.3 show that $\Delta^{av,1}$ yields a strictly decreasing sequence, as one should expect, whereas $\Delta^{av,mult}$ has a tendency to decrease, however with exceptions. This suggests that $\Delta^{av,1}$ is superior to $\Delta^{av,mult}$ as a complexity measure for digraphs, at least in the example presented in Table 9.3.

9.6 Using random coverings with reachability sets

In this approach one uses the reachability sets associated with each node in a digraph to quantify complexity. The idea is to draw k nodes v_1, \dots, v_k randomly and consider the union of the reachability sets $\bar{v}_1, \dots, \bar{v}_k$ and compute the portion of the node set it covers. If this is often close to 1, then the value of a critical value k^c for which this is obtained is a measure of the complexity of the digraph: the higher k^c the complexer the digraph.

So the complexity measure κ_{cov} we propose is the fraction of node set covered by the reachability sets sampled, that is

$$\kappa_{cov}(k) \triangleq \frac{E(|\cup_{i=1}^k \bar{v}_i|)}{|V|}. \quad (28)$$

For a given $\epsilon > 0$ we are interested in the critical value k_c which is defined as the smallest value k such that for samples of size k we have

$$k_c \triangleq \min_{k \in \mathbb{N}} \{ \kappa_{cov}(k) \geq 1 - \epsilon \}. \quad (29)$$

We denote the critical value that (29) produces by k_{ϵ}^c . The larger this value is, the more the digraph is removed from one with full reachability and the same number of points, and hence the more complex it is.

9.7 Taking the complexity of the underlying graph into account

The complexity measures of digraphs considered above are in fact defined relative to the underlying graph of the digraphs considered. The underlying graph in turn defines a class of digraphs that is a sort of universe in which to work. The role of the underlying graph in the complexity measures is therefore somewhat implicit. But in the present section we want to make its role explicit. We then view the digraph complexity measures above as conditional complexity measures, conditional on the universe of digraphs implied by the underlying graph of the digraph in question. But, of course, the underlying graph is a graph itself which also has a complexity. We want to present complexity measures of digraphs that take its complexity also into account.

If we follow the reasoning in probability theory with conditional probabilities and unconditional probabilities, we define an (unconditional) measure of complexity of a digraph DG (κ_{DG}) as the product of the (conditional) measure of complexity of the digraph DG given its underlying graph UG ($\kappa_{DG|UG}$) times the complexity of the underlying graph UG (κ_{UG}). Symbolically we can write this as:

$$\kappa_{DG} = \kappa_{DG|UG} \times \kappa_{UG}. \quad (30)$$

To make (30) precise we can take for $\kappa_{DG|UG}$ = any of the complexity measures for digraphs that we have studied in the present section, such as the one defined in (22),²³⁾ (25), (27) or (28); as UG is a graph we can take for κ_{UG} a complexity measure for graphs such defined in (1).

In case DG is in fact a graph $\kappa_{DG|UG} = 1$ then $\kappa_{DG} = \kappa_{UG}$, where DG is in fact equal to UG .

10 Local complexity of networks

10.1 Intuitive idea

In the complexity measure for the networks so far, the (tacit) assumption was that the neighbourhood of each point in the network is (in principle) the entire network.²⁴⁾ In the present section we want to consider an extension by considering local versions of these complexity measures. The aim is not to present the details for the ‘localization’ of every complexity measure considered above, but to present some examples. From these it should be clear how one can produce local versions of other complexity measures.

²³⁾ With reservations, as the measure is a rather crude one.

²⁴⁾ Local complexity is an example of a concept that is defined locally. Such concepts abound in mathematics, in areas such as geometry, algebraic geometry, topology and algebra, to mention but a few.

But what is the advantage of working locally? The obvious situation is the one in which the network is very big, or in which it is a random access network (RAN) which is a network for which essential information is lacking that is estimated by taking some probes of the network.²⁵⁾ A RAN is like an unknown population, such as a population of fish living in a particular lake: How many fish are there at a certain point in time?²⁶⁾ How many species of fish live there at a certain point in time? These are typical questions ichthyologists or ecologists may be interested in. And, typically, such a population is dynamic: fish are born in the lake; they migrate to the lake from another lake or river; they die of disease or old age; they are eaten by other animals; they migrate to another lake or river, et cetera. These questions then need to be answered repeatedly, at different points in time, so that an idea can be formed about the dynamics of the fish population in the lake. Likewise in case of a RAN, nodes and links appear and disappear constantly. Likewise a RAN is dynamic, like the fish population in the example. Think of the internet as the archetypal example of a RAN.

Huge networks such as RANs, and especially dynamic ones, require that one works locally. They are simply too big and possibly too fleeting, to be studied globally. This forces one to look locally. In case of a fish population in a lake, if it is too big to be studied in its entirety, it should be divided into suitably defined 'sub lakes'. These should, on the one hand, be small enough so that they can be investigated, and, on the other hand, should be sufficiently isolated so that they can be considered habitats, rather than passage areas. Combined they should provide a representative sample of the entire lake.

This means that for each point p in a network N we define a neighbourhood N_p , which is a local version of N (to be made precise below). The complexity measures defined above, for both graphs and digraphs, can be localized by carrying out computations for individual points p using N_p instead of the entire network N .

Neighbourhoods for graphs and digraphs are differently defined, but are both based on metrics (in case of graphs) or quasimetrics (in case of digraphs). For both types of networks we work locally (in neighbourhoods of points) and we generate a vector with local complexity information per entry, where each entry corresponds to a node. Details of both types of local complexity measures can be found in Subsections 10.4 and 10.5. But we first we consider neighbourhoods on graphs and digraphs and how they are defined with the help of (quasi)metrics that have been specified for these structures.

10.2 Neighbourhoods in graphs

In defining neighbourhoods in graphs we look at neighbourhoods in topology, in particular in metric spaces as an example. Let (X, d) be a metric space, with X a non-empty set and d a metric defined on this space. With the metric, open balls can be defined that form a basis for a topology (X, τ) . The open balls of the type $B_p(\rho) \triangleq \{x \in X | d(p, x) < \rho\}$ for $\rho > 0$ form a basis of this topology τ . A closed ball would be of the type $\bar{B}_p(\rho) \triangleq \{x \in X | d(p, x) \leq \rho\}$.

²⁵⁾ See e.g. [14]

²⁶⁾ 'Point in time' is a figure of speech and should not be taken literally. It takes time to probe an ecosystem such as a lake. So rather than a 'point' we need an interval. But the point of time is just a reference time. It could also be the midpoint of such a time interval.

For graphs we shall also use the idea of ‘balls’ of a certain diameter around a vertex p to implement the idea of a neighbourhood of p . For this we need a distance function. For graphs lengths of paths connecting points in a graph are natural choices to derive such a function.

A neighbourhood N_p of a node p in a graph $G = (V, E)$ is defined as a set of vertices in V , that is $N_p \subseteq V$. Once a neighbourhood N_p of a node p has been identified, it can be used to define a neighbourhood subgraph $G_p = (N_p, E_p)$ of the original graph $G = (V, E)$, where $E_p \triangleq \{\{a, b\} \in E \mid a, b \in N_p\}$.²⁷⁾

In order to make this work we need to define a metric. For this we use paths in each connectivity component of the graph $G = (V, E)$ be a graph. Let v and w be two nodes in G , that is, $v, w \in V$. Suppose that v and w are in the same connectivity component, so that there is a path π in G connecting v and w , for short $v \xrightarrow{\pi} w$. Because G is a graph, the reverse path π^{\leftarrow} is a path connecting w and v in G , that is $w \xrightarrow{\pi^{\leftarrow}} v$.

We start with defining the distance between nodes in G with the natural metric. In this metric every edge has length 1. If nodes $v, w \in V$ $v \xrightarrow{\pi} w$ in G then $d(v, w)$ is defined as the length of the shortest path $\pi_{v,w}^*$ in G connecting v and w , which is the number of edges on $\pi_{v,w}^*$. In case each edge $\{a, b\}$ has a length $d_{ab} = d_{b,a} > 0$ then the length of the path is equal to the sum of the lengths of the edges of which π consists. The distance d_{vw} of v and w is the length of the shortest path $\pi_{v,w}^*$ measured as the sum of the lengths of the edges of which it consists, that is

$$d_{vw} \triangleq \ell(\pi_{v,w}^*) = \sum_{\{a,b\} \in \pi_{v,w}^*} d_{ab}. \quad (31)$$

If $D_G = (d_{vw})$, with $v, w \in V$ and in the same connectivity component then the following properties hold:

- $D_G \geq 0$,
- $D'_G = D_G$,
- $d_{vv} = 0$ for all $v \in V$.

We now consider an example. Figure 10.1 shows a graph with two neighbourhoods, one with nodes distance 1 removed from the center node 3 and the other with nodes with distance at most 2 removed from the center node 3. In both cases the natural metric is used. The edges belonging to the neighbourhoods subgraph have also been indicated. On the left-hand side the entire graph is shown, whereas on the right-hand side only the neighbourhood subgraphs are shown.

It is understandable that one wants to concentrate on the local information such as the neighbourhood graph, but this is not quite enough. The problem is that the degrees of some nodes in the neighbourhood subgraphs may be incorrect: some edges may not be taken into account, in which case the degrees of some nodes in a neighbourhood are not taken into account.

²⁷⁾ In fact this is the convenient way to define a neighbourhood subgraph. However, an edge may actually not be situated on any path from a point in the neighbourhood to its central point. We take this for granted, because otherwise the definition of a neighborhood subgraph is too complicated. And this complication does not serve a purpose.

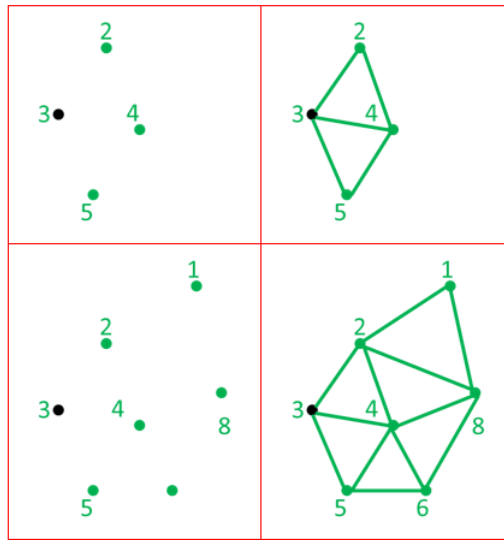


Figure 10.1 Neighbourhoods of node 3. Top left, with distance at most 1; bottom left, with distance at most 2. On the right-hand side the corresponding neighbourhood subgraphs are shown.

To avoid this problem we should consider extended neighbourhoods. This means, in case of neighbourhood N_3 , that edges should be considered in the cut set defined by N_3 and its complement $V \setminus N_3$. In Figure 10.2 we show the extended neighbourhoods displayed in Figure 10.1.

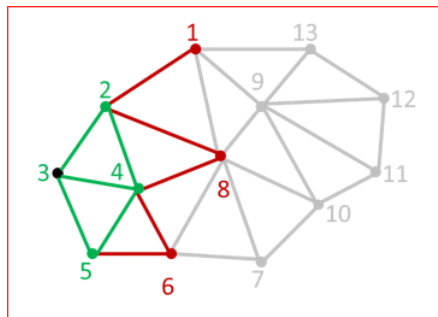


Figure 10.2 Extended neighbourhood subgraph for node 3 as part of the entire graph. It is the same graph as the one shown in Figure 10.1. The edges in red are in the extension.

10.3 Neighbourhoods for digraphs

To define neighbourhoods for digraphs we take a similar approach as in case of graphs (see Section 10.2). Because with digraphs we are now dealing with arcs (possibly without its counter-arc) instead of edges (arcs and their counter-arcs), the situation is generally quite different from that in case of graphs, due to a lack of symmetry. The distance of a node v to a node w may be different from that of node w to node v . It is actually possible that there is no path from w to v , in which case the corresponding ‘distance’ is symbolically ∞ . This means that

we (typically) are not be dealing with metrics, but with quasimetrics (see below).²⁸⁾

But we still can use the same idea to measure the distance from point v to point w , namely by considering paths π_{vw} from v to w , and to use the length of the shortest one as a measure of the distance from v to w . However, we have to measure the distance from node w to node v separately, as π_{vw}^{\leftarrow} may not be an allowable path from w to v ; it may not be a path at all.

Again, as in the case of graphs, we first define a neighbourhood as a set of nodes in the original graph. Then we add arcs to these nodes to obtain a subgraph of the original graph, which we shall call a neighbourhood subdigraph. We do this in a similar way as in case of graphs.

Let $G = (V, E)$ be a digraph, and a, b nodes in G . A path $\pi = (e_1, \dots, e_\ell)$ from a to b in G is a sequence of arcs in G , such that the end node of arc e_j is the start node of the next arc e_{j+1} for $j = 1, \dots, \ell - 1$, and the start node of e_1 is a and the end node of e_ℓ is b . The (natural) length of π is ℓ . This equals the distance of the start node of e_1 to the end node of e_ℓ . In this case each arc has length 1. If so desired one can associate a different length with an arc e_j , say d_j . The length associated with the path π would then be $\sum_j d_j$.

Let $(G, d) = (V, E, d)$ where $G = (V, E)$ be a digraph and d is a quasimetric on G . This is a function $d : E \rightarrow \mathbb{R} \setminus \mathbb{R}^-$. Also, d is not necessarily symmetric: If (a, b) is an arc in G , (b, a) need not be an arc in G , in which case $d(b, a)$ is not defined (at the moment).

Using the definition of a quasimetric on the arcs of G as a basis, we can extend this distance concept to any pair of nodes a, b in G . If $a \rightsquigarrow b$ then d_{ab} is defined as the length of the shortest path in G from a to b , where the length of a path is the sum of the d -values associated with each of the arcs on the path. If there is no path from a to b then the value of d_{ab} is set to ∞ .

If we use a matrix $D_G = (d_{ab})$, with $a, b \in V$ to represent the distances in G then it has the following properties:

- $D_G \geq 0$,
- $d_{aa} = 0$ for all $a \in V$.
- If $a \rightsquigarrow b$, then $d_{ab} = \infty$.

If we compare these properties with the corresponding ones for a graph, we see that the lack of symmetry (arcs without counter-arcs) leads to a more complicated distance concept in digraphs. This is clearly visible if we look at neighbourhoods of nodes in digraphs. Defining them requires a bit more attention than in case of graphs.

In case $G = (V, E)$ is a digraph a neighbourhood of a node p is a subset $V_{p,\delta}$ of V consisting of two parts, the incoming part $V_{p,\delta,in}$ and the outgoing part $V_{p,\delta,out}$:

$$V_{p,\delta,in} \triangleq \{v \in V \mid v \rightsquigarrow p, d(v, p) \leq \delta\}, \quad (32a)$$

$$V_{p,\delta,out} \triangleq \{v \in V \mid p \rightsquigarrow v, d(p, v) \leq \delta\}, \quad (32b)$$

$$V_{p,\delta} \triangleq V_{p,\delta,in} \cup V_{p,\delta,out} \quad (32c)$$

²⁸⁾ In case the digraph does not happen to be a graph.

To produce a neighbourhood subdigraph for a neighbourhood N_p we need to add a set of arcs. Like in case of graphs, we take the set of all arcs in E with end points in $V_{p,\delta}$ for this: $E_p \triangleq \{(a, b) \in E \mid a, b \in N_p\}$.²⁹⁾ $V_{p,\delta}$ for some choice of δ is an example of a neighbourhood N_p .

We use Figure A.2 as an example showing two neighbourhoods of a point in a digraph. In Figure 10.3 two neighbourhoods of node 4 are presented. The top row at the left-hand side depicts the neighbourhood of node 4 with nodes at distance at most equal to 1. On the top row at the right-hand side the neighbourhood subdigraph of node 4 is shown. On the bottom row at the left-hand side the neighbourhood of node 4 is shown, with the nodes at a distance of at most 2 to the center node 4. On the bottom row at the right-hand side the corresponding neighbourhood subdigraph of center node 4 is represented.

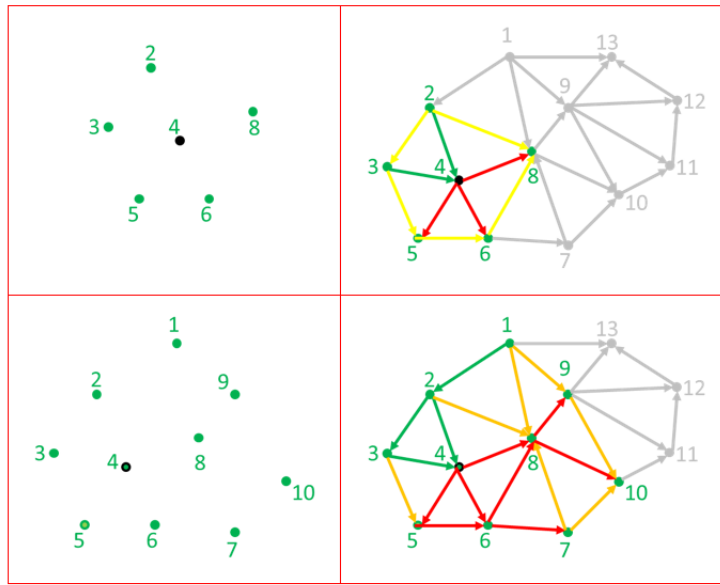


Figure 10.3 Neighbourhoods of center node 4 on the left, and the corresponding neighbourhood subdigraphs on the right. In the top row the distance to the center node 4 is at most 1, at the bottom row at most 2. Incoming arcs in green, outgoing arcs in red. Other arcs in neighbourhood subdigraphs in yellow. Remaining arcs and nodes in gray.

As in case of graphs we need to consider extended neighbourhood digraphs to obtain the correct indegrees of all the nodes of the neighbourhood. An extended neighbourhood subdigraph is a neighbourhood subdigraph with arcs added. These arcs are the ones connecting nodes in the neighbourhood subdigraph with nodes outside this neighbourhood subdigraph (or vice versa). Stated differently, the arcs are in the cut set defined by the neighbourhood and its complement in the original digraph.

By adding these arcs we make sure that every node in the neighbourhood digraph has the same indegree and outdegree as in case of the original digraph. In Figure 10.4 an extended neighbourhood digraph is shown for a neighbourhood digraph shown on the top row of Figure 10.3.

²⁹⁾ This choice may imply that some arcs in this neighbourhood are not situated on any path connecting a point in the neighbourhood to the center point. In a previous attempt to define a suitable neighbourhood subdigraph this was a requirement for each of its arcs. However, the resulting definition of a neighbourhood subdigraph turned out to be too complicated. It was therefore abandoned for the current, straightforward, and hence simpler, one.

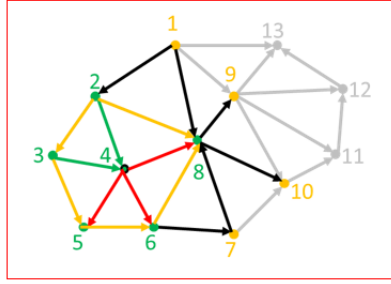


Figure 10.4 Extended neighbourhood subdigraph of center node 4. Ingoing arcs in green, outgoing arcs in red, other arcs of the neighbourhood in orange, arcs in the extension of the extended neighbourhood in black, orange nodes are nodes not in the neighbourhood but with an arc connecting them to the neighbourhood digraph. Nodes outside the neighbourhood and remaining arcs in gray.

In case one does not use extended neighbourhoods, one still is able to compute complexity measures, but for the border node of each neighbourhood, the values may not be the same as in case the original digraph would be used. If the neighbourhoods are bigger, one may expect that there are relatively fewer border points, compared to the total number of points in the interior of the neighbourhood.³⁰⁾

10.4 Local complexity for graphs

To illustrate local complexity for graphs we look at the average degree (as defined in 1). This would develop into the local variant where degrees are averaged for the nodes in the neighbourhoods of points, instead of over all nodes in the graph. We then would find:

$$\Delta_{p, N_p}^{av} = \frac{\Delta^{N_p}}{|N_p|}, \quad (33)$$

where Δ^{N_p} is the sum of the degrees of all nodes in N_p and $|N_p|$ denotes the size of N_p . In (33) we have used p as well as N_p as subscripts, because a neighbourhood for a point can usually be defined in many ways. So (33) is the average of the degrees of the points in N_p . In the original definition it was the average of the degrees of all the points in the network. For a local version we would take all the points in a neighbourhood of each point. Of course, these neighbourhoods should be defined first. This can be done in many ways. The result obviously depends on the choice of the neighbourhoods of each of the points in the network.

So instead of a single quantity to express the complexity of a network, using local complexity produces a vector of length $|N|$ in which the components are complexity measures for each node in the network. In case of the local complexity measure (33) we have

³⁰⁾ If the analogy with a ball would hold, the surface area of a ball in \mathbb{R}^3 with radius r grows as $4\pi r^2$, whereas the volume grows as $\frac{4}{3}\pi r^3$. In \mathbb{R}^n the n -volume of a ball is proportional to r^n and the $(n-1)$ -volume is proportional to r^{n-1} . If the same sort of relationship would hold for the number of border points relative to the total number of points in the neighbourhood, the influence of the values of the border points would decrease proportional to $1/r$.

$$\Delta_{loc}^{av} \triangleq (\Delta_{1,N_1}^{av}, \dots, \Delta_{|V|,N_{|V|}}^{av}), \quad (34)$$

where we have assumed that the nodes of the network (in V) have been consecutively ordered (in some way) from 1 to $|V|$. In case of a huge network one would only have the (estimates of) the values for a subset of the components.

10.5 Local complexity for digraphs

Local complexity for digraphs is also based on (extended) neighbourhoods of nodes, as in case of graphs. Only the definition of ‘neighbourhood’ for a digraph is different from that of a graph (see Subsection 10.3).

The idea is to apply reachability locally, that is, for the (extended) neighbourhoods of the nodes in G . Reachability is discussed in Section 9. In particular the complexity measure in (27) is based on the entropy of a distribution defined in (26). If reachability is defined locally, we can apply the same kind of measure as (26) locally. We shall leave the details to the interested reader.

11 Node rank

The idea behind node ranks is this: consider a criterion to define important nodes (or arcs) in a network, select them and complete the network with these nodes (or arcs) using connectivity information from the original network. For some problems it is natural to concentrate on the nodes, such as the Internet, where the nodes are (clusters of) webpages (URLs). For other problems, such as traffic problems, it is natural to consider links / arcs, which in this case would represent roads or road segments.³¹⁾

Node rank can be used as a criterion to identify the important nodes. The concept of ‘node rank’ is derived from that of ‘page rank’ defined, originally, for the WWW. Intuitively, the idea of node rank is that the node rank of a node is based on the node ranks of the nodes linking to it. A node with a higher node rank contributes more to the page rank of the node referred to. Despite this intuitive idea for node rank it can be defined in several ways. We present some examples.

³¹⁾ Technically, one can always consider top nodes. But then one should redefine the original network. One can create an arc network where the arcs are represented by dots, and two dots (a, b) and (c, d) are connected if and only if $b = d$, that is, if the head of (a, b) coincides with the tail of (c, d) . In the arc network of the original digraph the nodes are in fact the arcs of the original networks. Selecting top nodes in the arc network yields top arcs in the original network.

11.1 Node rank r_1

We present the definition of node rank from [14], which was inspired by that of page rank in [4]. It deviates slightly from the original definition. It can be fairly easily defined locally. From this definition a global one can be derived in terms of matrices. We start by looking at a node a and the $n_a > 0$ nodes pointed at from a , by arcs, b_1, \dots, b_{n_a} . Suppose that node a has rank $r_1(a)$. Then b_i gets share $r_1(a)/n_a$ of the rank $r_1(a)$ of a . To determine the rank $r_1(b)$ of b , one adds all contributions from the nodes pointing to b .

This method of determining a node rank can be linked to invariant distributions for Markov chains. See [6], pp. 392 ff. To the adjacency matrix A of a digraph a Markov matrix P can be associated with A by dividing each row i of A by the sum of the elements in this row (which is the outdegree for node i). This corresponds to the Markov chain where a jump from a node i occurs with equal probability to any of the nodes j connected to node i is, that is, such that $(i, j) \in E$.

If this matrix is denoted by P then an invariant distribution is a vector $u \geq 0$ with $u' u = 1$, such that

$$uP = u. \quad (35)$$

Under certain conditions $u > 0$ exists and is unique.³²⁾

11.2 Node rank r_2

The idea of the node rank as defined in Section 11.1 is that the rank $r_1(a)$ of a node a pointing to n nodes is equally distributed over its outgoing arcs. So if node b with $(a, b) \in E$ the 'donation' of a to the rank of b is $r_1(a)/n_a$. The implicit assumption is that a visit to a node a also implies a visit to node b . But this assumption is questionable, as it, in its ultimate consequence, would imply that users would follow very long paths of links (arcs). This is highly unlikely. Think of the Internet as an example. Typically users will only click on a limited number of links.

In a modified version of the node rank we want to distinguish between the rank contribution of a direct link to a node and rank contributions from indirect to this node. This latter contribution we want to diminish. In the context of the WWW, the modification can be motivated by the fact that only a fraction of the links pointing to a is actually used to point to b . For definiteness, we assume that only a (fixed) fraction $\sigma \in (0, 1)$ is used in this way. This is also the fraction that we use to carry over the node rank of a to a node b , with $(a, b) \in E$.

So we assume that the direct link from a to b contributes to the node rank of b in two components

1. a weight 1, because a is linked directly to b .

³²⁾ See the theorem on p. 393 in [6].

2. a weight $\sigma r_2(a)$, for the nodes indirectly linked to b via a , i.e. all the nodes c with $(c, a) \in E$. The parameter $0 < \sigma < 1$ is a control parameter, to see how the results change when this parameter is changed.

So the total contribution of a to the rank of b is: $1 + \sigma r_2(a)/n_a$. So the node rank of b is obtained by summing over all nodes a that point to b . That is

$$r_2(b) = \sum_{a \in \Xi(b)} (1 + \sigma r_2(a)) = \Delta_{in}(b) + \sigma \sum_{a \in \Xi(b)} r_2(a), \quad (36)$$

where $\Xi(b) = \{(z, b) \in E, \text{ for some } z \in V\}$ is the set of nodes with arcs pointing to node b and $\Delta_{in}(b)$ denotes the indegree of node b , i.e. $|\Xi(b)| = \Delta_{in}(b)$.

We can write (36) as a matrix equation

$$u = \Delta + \sigma A' u, \quad (37)$$

where $u = (r_2(1), \dots, r_2(n))'$, $\Delta = (\Delta_{in}(1), \dots, \Delta_{in}(n))'$. If (37) has a solution u , it can be written as:

$$u = (I - \sigma A')^{-1} \Delta. \quad (38)$$

So (38) exists if the matrix $I - \sigma A'$ is nonsingular, which is the case if σ is not an eigenvalue of A' , or equivalently, of A . If (38) has a solution, it is unique.

12 Arc rank

So far we have considered the concept of node rank of a network. In the present section we use node rank to compute arc ranks. They are used in Section 13 to reduce a (complex) network to a less complex one by removing certain nodes and arcs. That is, one method that is proposed there uses the arc ranks.

For each node v with incoming and outgoing arcs, the sum of the arc ranks of each type of arcs are the same and equal to the node rank of v . With arc ranks at our disposal we can distinguish between the importance of arcs. These arc ranks can also be used to correct node ranks in case a network is reduced to its essence. See Section 13.

The arc ranks are computed from the node ranks, by using iterative proportional fitting (IPF). In sampling applications IPF is used as a disaggregation method, to spread the values of marginal

tables with population numbers proportionally over a higher dimensional table with sampling values. The method originated in several areas, among them National Accounts.³³⁾

The node ranks act as the 1D marginals and the adjacency matrix of the digraph is the 2D table over which the node ranks are to be spread in a certain way. It should be stated that a solution does not always exist. However, in case of a 2D table with positive entries Sinkhorn's theorem guarantees the existence of a solution (see [10], pp. 75 ff.). For most digraphs this requirement is not satisfied; only for the complete digraphs does it hold. But we could approximate it by replacing each value 0 by a small value $\epsilon > 0$, thus obtaining A_ϵ instead of A . Then by taking the limit $\epsilon \downarrow 0$ we hope to find a solution to the original IPF problem with A instead of A_ϵ .³⁴⁾

If a solution exists³⁵⁾ we have that the sum of the arc ranks of the incoming arcs of a node i equals that of the sum of the ranks of the outgoing arcs of this node, which are both equal to the node rank of i .

In Figure 12.1 a general setting for the IPF problem we need to solve is presented. The marginal tables are provided by the node ranks ω_i . The adjacency matrix $A = (a_{ij})$ is used as the 2D table over which the node ranks are to be distributed.

	ω_1	...	ω_n
ω_1	a_{11}	...	a_{1n}
\vdots	\vdots	\ddots	\vdots
ω_n	a_{n1}	...	a_{nn}

Figure 12.1 IPF problem setting: node ranks ω_i and adjacency matrix a_{ij} .

In Figure 12.2 the solution — if it exists — of Figure 12.1 is presented. The weights w_{ij} are the arc weights.

	ω_1	...	ω_n
ω_1	w_{11}	...	w_{1n}
\vdots	\vdots	\ddots	\vdots
ω_n	w_{n1}	...	w_{nn}

Figure 12.2 Solution of the IPF problem in Figure 12.1: arc ranks w_{ij} .

Row-wise and column-wise the weights sum to the node ranks: $W\iota = W'\iota = \omega$, where $\omega = (\omega_1, \dots, \omega_n)'$, ι is the all 1s vector of length n and $W = (w_{ij})$, the matrix of arc ranks.

³³⁾ The name associated with applications in economics is that of R. Stone from Cambridge University, who called it the RAS method. In the sampling area W. Deming and F. Stephan are early proponents of this method. [2] was written at the department of Stone and provides an account that is motivated by economic applications. [10], Section 2.6 is another source for a discussion of this method, which has a more statistical orientation. The R package mipfp (see [3]) can be used to apply IPF (and similar methods).

³⁴⁾ Provided a direct computation does not yield a solution. However, in many cases if some of the entries are 0 a solution can be obtained by direct computation, which means iteratively.

³⁵⁾ To find necessary and sufficient conditions for IPF to converge is a complicated matter. Sufficient conditions for the 2D case are known: 1. the marginals of the values in the table (i.e. a_{i+} and a_{+j}) are strictly positive, and 2. the table is inseparable, that is, it does not permute to a block-diagonal form. But are they also necessary?

13 The essence of a network

13.1 The idea

The problem is how to present the highlights of a complicated network, from which all the distracting details have been removed. But what are the highlights, and the distracting details? The same problem is faced by a cartographer who wants to produce a map of an entire country that should give a viewer a clear picture of its main features: the bigger cities, the main waterways, the main roads, etc. And if one zooms in at a smaller part of the country more details become visible – but only for a smaller part of the country. So there is a trade-off between scale and detail.

Depending on the scale level chosen, the corresponding amount of detail provided is chosen. The reason for this is to focus on the important things at the chosen scale level. In a sense, less is more: presenting all the details available swamps the message and hides the bigger picture.

Translating this idea to networks one can employ node and arc ranks to differentiate among the nodes and the arcs in a network and choose the more important ones. One can select the nodes that have a rank value above the threshold value δ . So δ is a parameter controlling the detail to be made visible of the reduced network, and hence its complexity (the more detail the more complex). But only showing the selected nodes or arcs may result in a reduced network with a topology that deviates from that of the original network. This we want to avoid. It can be achieved by adding more arcs, carefully chosen.

Selecting the important nodes is the easy bit of the complexity reduction. More work is required to find the arcs that should be added (if any). This is done by looking at the topology, in particular the connectivity, of the original network. It is clear that this process reduces the complexity of the original network. The reduction is controlled by the parameter δ .

The first approach is focussed on node selection. The arcs follow suit, and they are mainly used to produce a reduced network with the right topology, mimicking that of the original network. In the second approach the arcs are selected first and the nodes follow suit. We consider these approaches in separate sections.

An application of computing the essence of a network is to apply it to a big (and complex) network and to draw it in a picture, on screen or on paper. As software to render networks already exists, it is only required to be able to compute the essence of networks, following the specifications of users.

13.2 Selecting nodes

A node rank defined for a network can be used to distinguish the important nodes from the less important ones: the higher the value, the higher their importance. So given a network $G = (V, E)$, a node rank function $\rho : V \rightarrow \mathbb{R} \setminus \mathbb{R}^-$ and a threshold $\delta > 0$, we can define $V_\delta = \{v \in V \mid \rho(v) \geq \delta\}$ as the subset of nodes in V with rank at least δ .

For the adjacency matrix A of G let $A|_{V_\delta}$ be the adjacency matrix obtained from A by only selecting those rows and columns that correspond to elements in V_δ . The first idea is to consider $A|_{V_\delta}$ as the adjacency matrix for the selection of nodes V_δ . But it is not the correct choice, as it only gives the arcs that are also in G . But two nodes may be connected indirectly, via nodes not in V_δ . So we should know for each $v_i, v_j \in V_\delta$ whether $v_i \leadsto v_j$ holds in G , that is, whether there is a path in G from v_i to v_j .

In order to decide this, the transitive closure A^* of A is needed. In fact, the adjacency matrix of the digraph best presenting the essence of the original digraph is $A^*|_{V_\delta}$, which denotes A^* restricted to the rows and columns corresponding to the nodes in V_δ .³⁶⁾ But both $A|_{V_\delta}$ and $A^*|_{V_\delta}$ are needed to distinguish arcs in G from paths in G that are not arcs. If $A|_{V_\delta}(v_i, v_j) = 1$ there is an arc in G from v_i to v_j , which we colour blue. In case $A^*|_{V_\delta}(v_i, v_j) = 1$ and $A|_{V_\delta}(v_i, v_j) = 0$ there is a path from v_i to v_j which is not an arc in G , which we colour red. A red arc corresponds to a path in the original digraph connecting top nodes. However, not all the nodes in such a path need to be top nodes.

13.3 Selecting arcs

In Section 13.2 nodes were selected to define an essence digraph. But one can also derive an essence digraph based on a selection of arcs.

Let the arcs in a digraph $G = (V, E)$ be ranked according to some arc rank. On the basis of this ranking and a threshold ϵ top arcs from E are selected, that is with arc rank at least ϵ . Denote the selection by $T = \{e_1, \dots, e_l\}$. Each arc is an ordered pair of vertices: $e_j = (v_{j_1}, v_{j_2})$. Let $\check{e}_j = \{v_{j_1}, v_{j_2}\}$ be the underlying edge of e_j . T implies the set $TV = \bigcup_{j=1}^l \check{e}_j = \{v_{11}, v_{12}, \dots, v_{l1}, v_{l2}\}$ of endpoints of these arcs.

As in Section 13.2 the adjacency matrix that describes the essence of G based on the node set TV and the arc set T is $A^*|_{TV}$, which denotes A^* restricted to the rows and columns corresponding to the nodes in TV . We can apply the same colouring scheme as in Section 13.2 to colour the arcs in the essence digraph with the nodes in TV .

13.4 Example

We consider an example of a small network, where the various nodes have different node ranks, indicated by colour. See Figure 13.1, in the upper left corner. The nodes are coloured to indicate different degrees of node ranks. The digraph in this example is somewhat special as it is a ditree (and hence is acyclic).

In the cell in the right-upper corner of Figure 13.1 the nodes with the lowest ranks have been removed, as well as the arcs that are incident to these nodes. This is the first reduction. In the left lower corner the nodes with the highest rank in this digraph are shown. This is the second reduction. Finally the digraph obtained by reducing the second reduction is shown in the right-hand lower corner. It is a single node, which is the hub of the original digraph.

³⁶⁾ Or perhaps the transitive reduction of this matrix, or the digraph corresponding to it, if one wants a smaller set of arcs from which others can be deduced by transitive closure.

in the reduced digraph. In this way node information from the original digraph is preserved and the arc ranks are adjusted to obtain consistency.

It should be pointed out that, tacitly, this reduction process has redefined the meaning of the remaining nodes. Some nodes in the reduced graph seem the same nodes in the original network, but this is not really the case. There is a ‘semantic shift’ in the meaning of these nodes. To understand this, consider the nodes 3, 4, 5, 6, 8, 14 in Figure 13.2. Each of them has neighbourhoods in the original digraph and in the reduced digraph that are different. So, for instance, node 14 of the reduced network still also represents nodes 15 and 16 in the original digraph, although these nodes do not appear in the reduced network. However, their influence is still present in the node and arc ranks used, as they have not been updated. To mark this difference it is perhaps preferable to indicate explicitly that these nodes are slightly different by using primes with the original labels. So we then would use, for instance, $3', 4', 5', 6', 8', 14'$ in the reduced network instead of 3, 4, 5, 6, 8, 14 in the original network. However, this convention shows the relation between corresponding nodes in the reduced and original digraph.

13.4.2 Adjust node ranks

If this option is chosen, the node ranks from the original digraph are adjusted by subtracting the ranks of the arcs removed from the node ranks of the original network. If we denote the node ranks for the reduced network by $w(j)'$ and the arc ranks of the original network by $\alpha_{i,j}$, we find from Figure 13.2 the following set of linear constraints:

$$\begin{aligned}
w(1)' &\triangleq \alpha_{2,1} + \alpha_{3,1} + \alpha_{4,1}, \\
w(2)' &\triangleq \alpha_{5,2} + \alpha_{6,2}, \\
w(3)' &\triangleq \alpha_{6,3}, \\
w(4)' &\triangleq \alpha_{8,4}, \\
w(5)' &\triangleq \alpha_{11,5}, \\
w(6)' &\triangleq \alpha_{11,6}, \\
w(8)' &\triangleq \alpha_{14,8}, \\
w(14)' &\triangleq \alpha_{17,14}, \\
\alpha_{17,14} &= \alpha_{14,8} = \alpha_{8,4} = \alpha_{4,1},
\end{aligned} \tag{39}$$

The final equality of (39) implies

$$w'(14) = w(8)' = w(4)' = \alpha_{4,1} \tag{40}$$

For the reduced network the arc nodes have to be computed using the adjacency matrix of the reduced network and the adjusted node ranks as marginals. This can be done using the IPF algorithm, as described in Section 12, but for the new situation.³⁷⁾

³⁷⁾ We assume that in this way new arc ranks can actually be found. But this depends of the reduced network that has been created, and in particular the structure of its adjacency matrix. If there are too many zero cells, awkwardly placed, a solution may not exist.

It should be stressed that in this case the meaning of the nodes has not been changed: each node in the reduced network represents the same object as in the original network. The rank information from the nodes and arcs eliminated has also been eliminated along with these objects.

13.5 Graph reduction and complexity

Obviously, the methods described in Sections 13.2 and 13.3 lead to smaller digraphs, i.e. with fewer nodes or fewer arcs. The expectation is that this usually leads to networks with smaller complexity. But this is not necessarily the case. In Figure 4.2 we provided a simple example of a graph that has a subgraph of higher complexity, if we take the average degree, Δ^{av} , as the measure of complexity.

The same phenomenon exists for digraphs: the reachability of a subdigraph can be less than that of the original digraph, and hence there can be an increase in complexity.

By removing nodes or arcs from a network with graph reductions one ends up in a class of networks different from the one to which the original network belongs, because the underlying graph is different. This makes comparison of complexities of digraphs belonging to different classes of underlying graphs not very meaningful. Within a class of digraphs with the same underlying graph direct comparison of our complexity measures for digraphs is meaningful and informative. But not between networks belonging to different classes. The problem is that the relation between number of nodes and number of arcs on the one hand and a complexity on the other is not monotonous: more nodes and / or arcs does not automatically imply a higher complexity. Important is how these elements are interconnected.

14 Discussion and conclusions

The present paper has two major objectives: to define complexity measures for networks and to find methods to simplify networks, in particular those that are complex. First complexity measures for graphs were defined, as they are simpler due to the fact that their adjacency matrix is symmetric. For graphs, the first idea to quantify the complexity of graphs is to use the degree of 'compression' of a graph as a guiding principle. A second idea was to use a special kind of full, continuous search of the graph as a basis for graph complexity. A third idea was to use natural distances between nodes in graphs or line graphs.

Then complexity measures for digraphs were defined. This is a natural order to proceed, as digraphs form a wider class of objects than graphs. New ideas have to be used to describe their complexity. We have used reachability and the number of paths (for routing digraphs). We hinted at the possibility of using search, and pointed at additional complications. Our approach lead also to complexity measures for digraphs that are defined relative to the underlying graphs. However some complexity measures for digraphs look at augmentation by arcs in order to reach full reachability, which implies a transition to different underlying graphs. Then direct comparison of the complexity of digraphs is not possible anymore. It would, however, if the complexity of the underlying graph is also taken into account. We have suggested an approach

where the digraph complexity measure is viewed as a complexity measure conditional on the underlying graph. By multiplying this by the complexity of the underlying graph the idea is that an unconditional complexity measure for digraphs is thus obtained.

Another extension that is considered in this paper is local complexity. The measures considered initially in the present paper tacitly assume that the neighbourhood of each point is the entire network. However it is possible to choose a neighbourhood of each node. Complexity information is collected locally, that is in the neighbourhood associated with each node. Local complexity measures were constructed by adapting the global measures considered before by using local information, that is the neighbourhoods associated to the respective nodes in the network.

In mathematics there are many structures that are defined locally. Manifolds, vector fields, sheaves are examples of such local structures. The challenge is to derive global results from such locally defined constructs. In case of local complexity of a network the challenge is to link it to its global complexity. This is left as future research topic. The present paper does not go into this issue.

The complexity measures introduced in this paper have not been thoroughly explored. Only a few examples have been presented for the purpose of illustration. Full exploration on real data is reserved for the future. Only then does it become clear what the value of each of the complexity measures is, what their limitations are, how they are interconnected, how they should possibly be modified to provide more attractive complexity measures, etc.

The second theme explored in the present paper is the reduction of networks. A reduction similar as in cartography where one can zoom out and get information about a larger area but with less detail, or zoom in and get detailed information about a smaller area. In a network context one should be able to indicate what one considers important nodes or arcs and a simplified network should then be generated with these elements present, and, if necessary, supplemented with additional information to produce a network that has the same topology as the original network. A reduced network embodies, so to speak, the essence of the original network, at a level chosen by a user.

An application of this reduction option would be complex networks. One often would like a view of the network that leaves out distracting details – unimportant nodes or arcs – so that one can focus on the parts that matter. In a network representing a snapshot of the Internet, the important nodes are the hubs of a certain minimum size.

One can reduce a network using only the original objects, with exactly the same meaning as in the original network, or by using modified objects, which are in fact aggregate objects. Original objects plus some neighbouring ones. So a node is then not a hub as in the original Internet, but a hub and less important nodes (representing pages) pointing to it.

To be able to value, and rank, nodes and arcs in a network we consider node ranks and arc ranks. Node ranks are weights on nodes that measure popularity, so to speak, in terms of being referenced, that is pointed, at by arcs. A node b pointing to a node a contributes its own node rank $\sigma(b)$ to the node rank $\sigma(a)$ of node a . In one type of node rank, the full node rank $\sigma(b)$ of node b is contributed to the node rank $\sigma(a)$ of the node a . This tacitly assumes that if node c points to b it also points to a , although this may only be through an indirect link. In a modified

version of this node rank, the contribution of each node pointing to another one is modified: the direct link counts for one and the indirect ones are discounted by a fixed factor λ .

Given a network with node ranks, arc ranks were computed from them. We use the IPF algorithm for this purpose, with the adjacency matrix of the network as the $2D$ table and with the node ranks as the marginal tables. As with the nodes one can also select important arcs and define a reduced network on their basis. Like in the nodes case, one has to make sure that the topology of the reduced digraph reflects that of the original digraph.

It would be nice if network reduction would imply a lower level of complexity of the reduced network than the original one. We have seen that this does not necessarily hold for the complexity measures considered, with the possible exception of an unconditional complexity measure. There was no opportunity to investigate this problem here, so it is left to future research to answer it. It would be nice to have complexity measures that have lower values for reduced networks.

In the paper several problems were mentioned that need to be solved, possibly only approximately, if one wants to have a set of routines at one's disposal for complexity computations, or at least implemented. Possibly some problems are quite difficult to solve and cannot be solved exactly, in which case approximation methods are called for.

Apart from elaborating some ideas further theoretically, what is most needed is to apply the methods proposed to real data.

References

- [1] A. Aho, M. Garey & J. Ullman (1972). The transitive reduction of a directed graph, *SIAM Journal on Computing*, 1 (2), pp. 131–137.
- [2] M. Bacharach (1970). *Biproportional Matrices & Input-Output Change*, Cambridge University Press.
- [3] J. Barthelemy, T. Suesse & M. Namazi-Rad (2018). Package 'mipfp' – multidimensional iterative proportional fitting and alternative, CRAN repository.
- [4] S. Brin & L. Page (1998). The anatomy of a large-scale hypertextual web search engine, Report, Computer Science Department, Stanford University, Stanford, Cal., USA.
- [5] T. Cormen, C. Leieron & R. Rivest (2009). *Introduction to Algorithms*, MIT Press.
- [6] W. Feller (1968). *An Introduction to Probability Theory and Its Applications*, Vol. I (3rd ed.), Wiley.
- [7] Munro, I. (1971). Efficient determination of the transitive closure of a directed graph, *Information Processing Letters*, 1, pp. 56–58.
- [8] Purdom, P., Jr. (1970), A transitive closure algorithm, *BIT*, 10, pp. 76–94.
- [9] D. Rouvray & R. King (eds.) (2002). *Topology in Chemistry*, Woodhead Publishing.
- [10] E. Seneta (1981). *Non-negative Matrices and Markov Chains*, Springer.
- [11] S. van Wageningen (2020). Quantifying network complexity: An evaluation of measures. Master's thesis, University of Leiden.
- [12] S. Warshall (1962). A theorem on Boolean matrices, *J. of the ACM*, 9, pp. 11-12.
- [13] L. Willenborg (1988). *Computational Aspects of Survey Data Processing*. CWI Tract 54, Center for Mathematics and Computer Science, Amsterdam.
- [14] L. Willenborg (2018). Sampling restricted access networks. Discussion paper, CBS, The Hague.
- [15] L. Willenborg (2019). Complexity and simplification of networks. Discussion paper, CBS, The Hague. [first version of the present paper]

Appendix

A Example digraphs and their complexities

We present here some examples of digraphs in which we look at reachability for each of the nodes. The examples we consider, all have the same underlying graph G , namely the one shown in Figure A.1. We denote this class of digraphs as $\mathcal{G}_{G,k}$, where G denotes the underlying graph and k denotes the number of arcs in the class of digraphs. If $G = (V, E)$ with $|V| = n$ and $|E| = m$, we have $m \leq k \leq 2m$. To compare the behaviour of the digraphs it is interesting to first consider the digraphs in a class $\mathcal{G}_{G,k}$, for a fixed parameter k , and then to see what happens when the parameter k is varied from m to $2m$. The digraphs considered in this way will more and more look like the graph G , and so will the reachability properties. One can also vary G in the class of graphs with n points, by varying the number of edges. As we are interested in connected graphs this number should take values between $n - 1$ (a tree) and $\binom{n}{2}$ (the full graph on n points). Below we restrict ourselves to the first step and consider, for a given graph G , a class $\mathcal{G}_{G,k}$ for $k = m$ and also for some values of k between m and $2m$. It would take too much space to also vary over the G s and to study for each choice what happens within a class $\mathcal{G}_{G,k}$ and when varying k . So we only consider one graph G below. The graph we use in all our examples is the one in Figure A.1, that we shall refer to by \mathcal{H} . This is a graph with 13 nodes and 26 edges.

We start with a digraph in which the arrows are neatly ordered. This case is a kind of a benchmark. It is an example of a digraph with the smallest complexity due to arc orientation in the class of digraphs with the same underlying graph. Then we look at a digraph in which the arcs can be considered as randomly oriented. Looking at reachability we see a more chaotic picture than in case of the very orderly first example. The third example shows a digraph that is very close to the one in the second example; they differ only in the orientation of one arc. This example is considered to indicate the consequences of a small change in orientation of the arcs in terms of reachability.

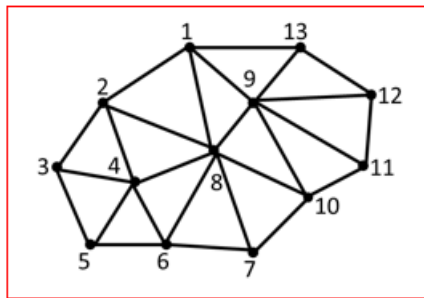


Figure A.1 The graph \mathcal{H} that is the underlying graph for the digraphs considered in the examples of the present appendix.

Example 1: An orderly digraph in $\mathcal{G}_{\mathcal{H},26}$

We consider the digraph represented in Figure A.2, which can be considered to be one of the two most orderly graphs in the class of digraphs, with the graph \mathcal{H} in Figure A.1 as the underlying

graph.³⁸⁾ Note that for all the arcs (a, b) we have that $a < b$. So an arc is always 'pointing forward', so to speak.

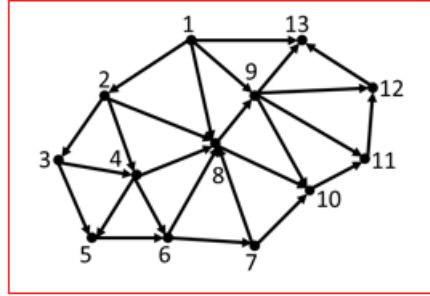


Figure A.2 Digraph with underlying graph in Figure A.1 from Example 1, where all arcs are 'pointing forward'.

Figure A.3 shows the reachability sets for each node in the digraph in Figure A.2. The reachability sets are neatly nested: the reachability set of a node contains those associated with a higher number.

We are interested in the nodes that can be reached from each of the nodes in Figure A.2. These follow from the transitivity properties of this graph. In Figure A.3 they are presented in graphical form. The entrance node is given in blue with blue and underlined label. Some nodes have the same reachability set. The reachability set corresponding to node k is presented as \bar{k} . We see that these sets differ widely. Sometimes they consist of a single point, sometimes they consist of several points, and in one case all the nodes can be reached from a single point. So the situation is far more diverse than in the graph case, where reachability and connectedness coincide.

The reachability sets can be ordered by inclusion. Figure A.4 represents this inclusion relation for the reachability sets in Figure A.3. Each arc indicates an inclusion. So reachability set $\bar{1}$ contains reachability set $\bar{2}$.³⁹⁾

Example 2: A random digraph in $\mathcal{G}_{\mathcal{H},26}$

The digraph we consider in this example is given in Figure A.5. In this case the arcs have been randomly oriented. In that sense this digraph is more chaotic, and intuitively more complex than the one in Figure A.2.

The reachability sets in Figure A.6 are partially ordered through inclusion.⁴⁰⁾ The tree showing this inclusion relation is presented in Figure A.7. We have

³⁸⁾ The other most orderly digraph in this class is the digraph with all the arcs reversed.

³⁹⁾ Or equivalently, reachability set $\bar{2}$ is included in reachability set $\bar{1}$.

⁴⁰⁾ In the sense of 'contains', not in the sense of 'is contained in'.

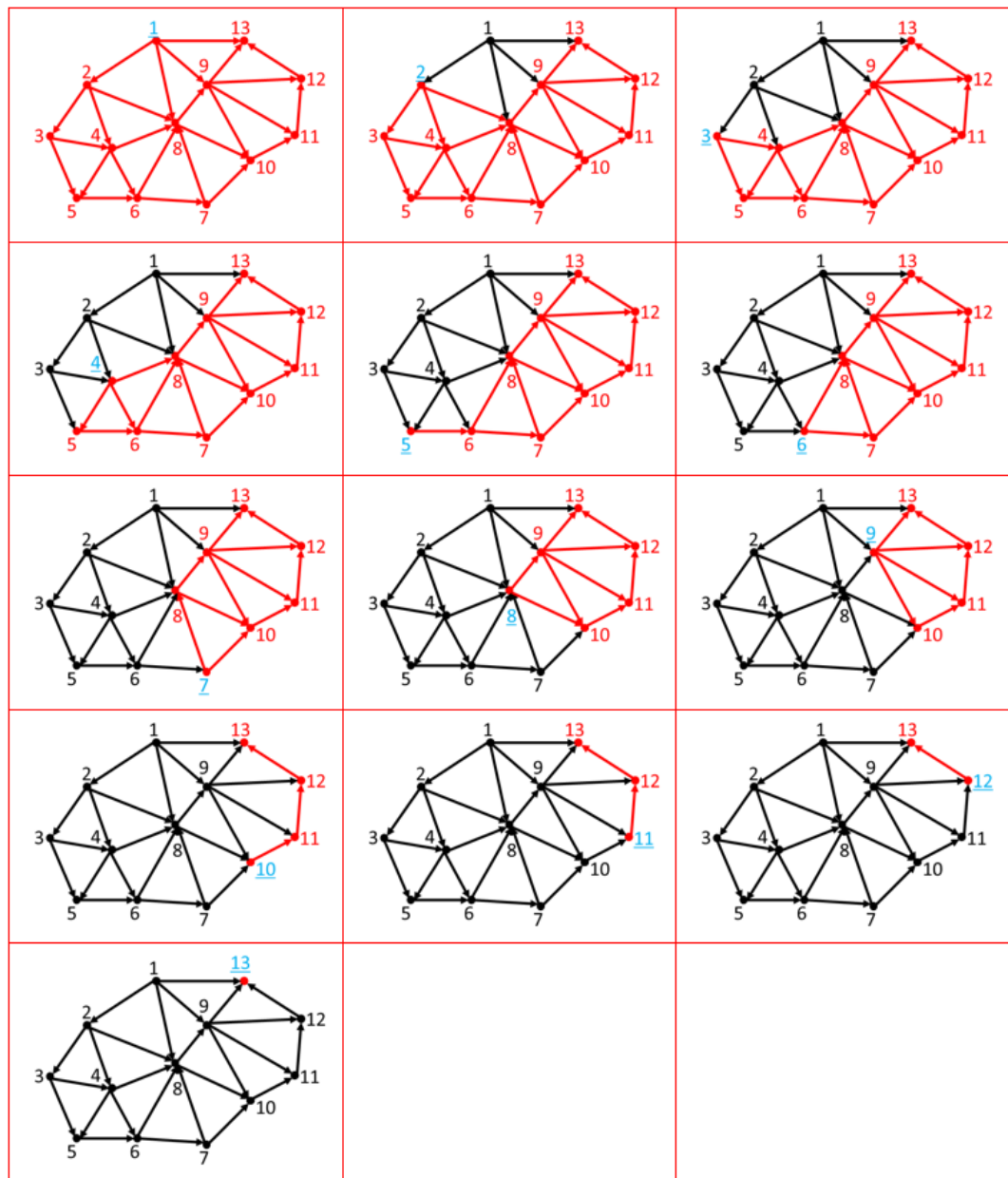


Figure A.3 The reachability sets of the nodes in digraph for Example 1 in Figure A.2.

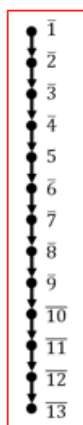


Figure A.4 The reachability sets of the nodes in digraph for Example 1 in Figure A.2.

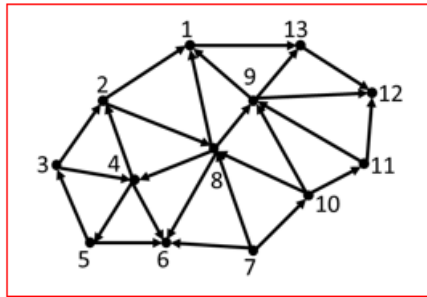


Figure A.5 A digraph for Example 2 with randomly oriented arcs.

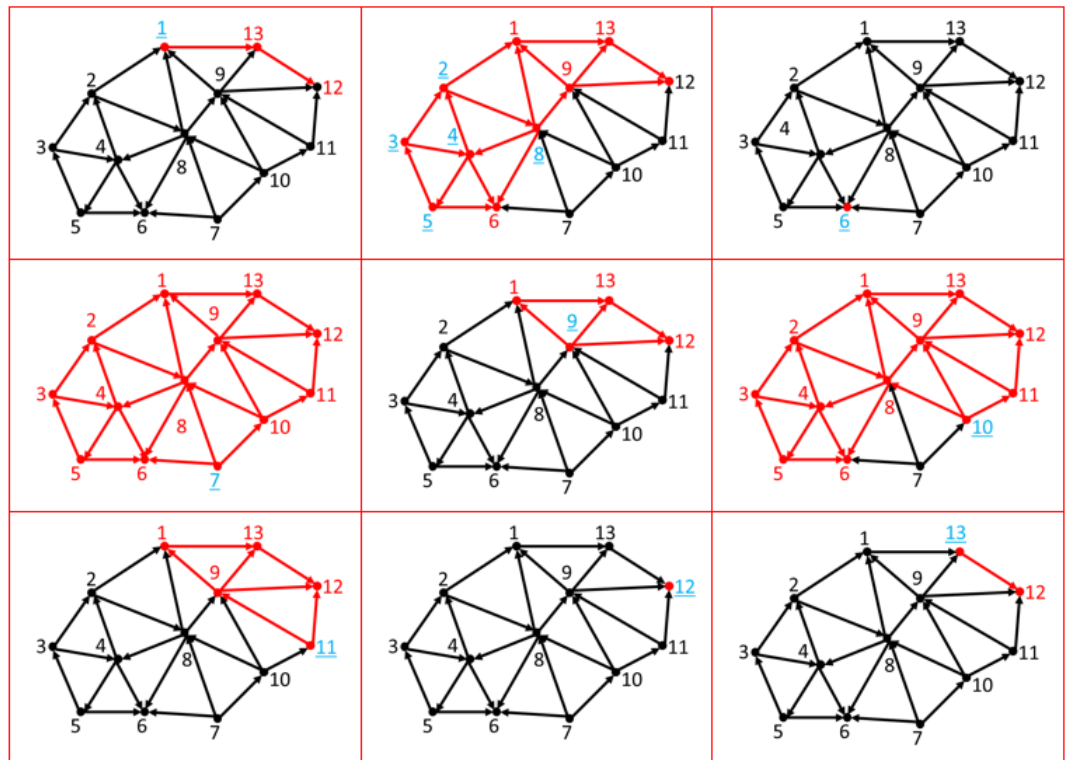


Figure A.6 Reachability sets for each of the nodes in the digraph of Example 2 in Figure A.5. The picture in the middle of the top row shows a reachability set that is the same for several entrance nodes.

$$\begin{aligned}
\bar{1} &= \{1\} \cup \bar{13} \\
\bar{2} &= \{2\} \cup \bar{1} \cup \bar{8} \\
\bar{3} &= \{3\} \cup \bar{2} \cup \bar{4} \\
\bar{4} &= \{4\} \cup \bar{2} \cup \bar{5} \cup \bar{6} \\
\bar{5} &= \{5\} \cup \bar{3} \cup \bar{6} \\
\bar{6} &= \{6\} \\
\bar{7} &= \{7\} \cup \bar{6} \cup \bar{8} \cup \bar{10} \\
\bar{8} &= \{8\} \cup \bar{1} \cup \bar{4} \cup \bar{6} \cup \bar{9} \\
\bar{9} &= \{9\} \cup \bar{1} \cup \bar{12} \cup \bar{13} \\
\bar{10} &= \{10\} \cup \bar{8} \cup \bar{9} \cup \bar{11} \\
\bar{11} &= \{11\} \cup \bar{9} \cup \bar{12} \\
\bar{12} &= \{12\} \\
\bar{13} &= \{13\} \cup \bar{12}.
\end{aligned} \tag{A.1}$$

The \bar{k} 's act as nonterminal symbols. By repeatedly substituting the expressions in (A.1) we can eliminate them and what remains are sets of nodes:

$$\begin{aligned}
\bar{1} &= \{1, 12, 13\} \\
\bar{2} = \bar{3} = \bar{4} = \bar{5} = \bar{8} &= \{1, 2, 3, 4, 5, 6, 8, 9, 12, 13\} \\
\bar{6} &= \{6\} \\
\bar{7} &= \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13\} \\
\bar{9} &= \{1, 9, 12, 13\} \\
\bar{10} &= \{1, 2, 3, 4, 5, 6, 8, 9, 10, 11, 12, 13\} \\
\bar{11} &= \{1, 9, 10, 11, 12, 13\} \\
\bar{12} &= \{12\} \\
\bar{13} &= \{12, 13\}
\end{aligned} \tag{A.2}$$

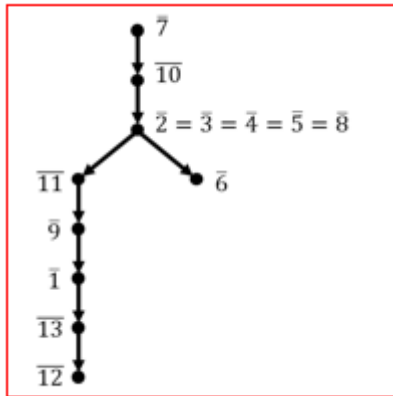


Figure A.7 Nesting of the reachability sets of Example 2 in Figure A.6.

The reachability tree in Figure A.7 can be used to produce the reachability sets in (A.2). By starting with node $\bar{12}$ (which must be the set $\{12\}$) and $\bar{6}$ (which must be the set $\{6\}$) and working 'upwards', one can retrieve the other reachability sets in (A.2).

Example 3: Another digraph in $\mathcal{G}_{\mathcal{H},26}$

The digraph we consider is given in Figure A.8. This digraph is the same as the one in Figure A.5 except for one arc that has been reversed: arc $(8, 9)$ was replaced by arc $(9, 8)$. To stress this, the arc $(9, 8)$ is represented in a different colour in Figure A.8. This example was chosen to illustrate how one small change in a digraph (reversal of an arc) can give rather different results concerning reachability.

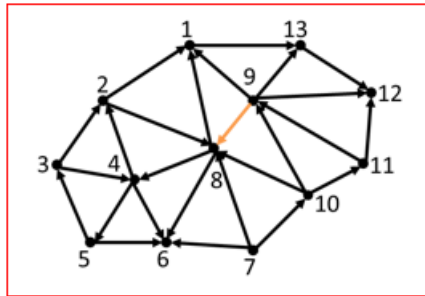


Figure A.8 A digraph similar to that in Figure A.5 used in Example 3. Only arc $(8, 9)$ has been replaced by arc $(9, 8)$.

As in Example 1 we are interested in the nodes that can be reached from each of the nodes in Figure A.8. In Figure A.9 they are presented in a graphical form. As in Figure A.5 the initial node is given in blue and is underlined. As in Example 2 some nodes have the same reachability set. But we also note that these sets may differ considerably.

The reachability sets in Figure A.9 are partially ordered through inclusion, as in Example 1.⁴¹⁾ The tree showing this inclusion relation is presented in Figure A.10. The reachability set corresponding to node k is presented as \bar{k} . We have the following sets of equations for the reachability sets indicated in Figure A.9:

⁴¹⁾ In the sense of 'contains', not in the sense of 'is contained in'.

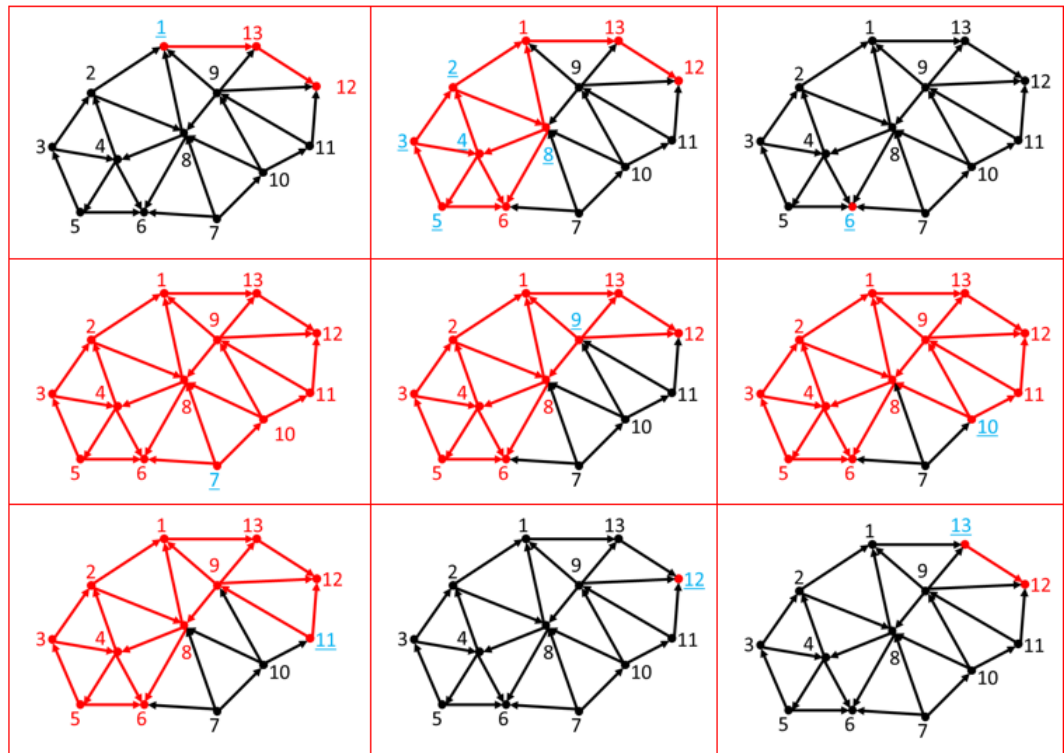


Figure A.9 Reachability sets for each of the nodes in the digraph in Figure A.8 in Example 3. The entrance nodes are coloured blue and are underlined. The picture in the middle of the top row shows a reachability set that is the same for several entrance nodes.

$$\begin{aligned}
\bar{1} &= \{1\} \cup \bar{1}\bar{3} \\
\bar{2} &= \{2\} \cup \bar{1} \cup \bar{8} \\
\bar{3} &= \{3\} \cup \bar{2} \cup \bar{4} \\
\bar{4} &= \{4\} \cup \bar{2} \cup \bar{5} \cup \bar{6} \\
\bar{5} &= \{5\} \cup \bar{3} \cup \bar{6} \\
\bar{6} &= \{6\} \\
\bar{7} &= \{7\} \cup \bar{6} \cup \bar{8} \cup \bar{1}\bar{0} \\
\bar{8} &= \{8\} \cup \bar{1} \cup \bar{4} \cup \bar{6} \\
\bar{9} &= \{9\} \cup \bar{1} \cup \bar{8} \cup \bar{1}\bar{2} \cup \bar{1}\bar{3} \\
\bar{1}\bar{0} &= \{10\} \cup \bar{8} \cup \bar{9} \cup \bar{1}\bar{1} \\
\bar{1}\bar{1} &= \{11\} \cup \bar{9} \cup \bar{1}\bar{2} \\
\bar{1}\bar{2} &= \{12\} \\
\bar{1}\bar{3} &= \{13\} \cup \bar{1}\bar{2}.
\end{aligned} \tag{A.3}$$

Elaborating the equations in (A.3) by repeated substitution until all nonterminal symbols have been eliminated, we obtain the following set of solutions:

$$\begin{aligned}
\bar{1} &= \{1, 12, 13\} \\
\bar{2} = \bar{3} = \bar{4} = \bar{5} = \bar{8} &= \{1, 2, 3, 4, 5, 6, 8, 12, 13\} \\
\bar{6} &= \{6\} \\
\bar{7} &= \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13\} \\
\bar{9} &= \{1, 2, 3, 4, 5, 6, 8, 9, 12, 13\} \\
\bar{1}\bar{0} &= \{1, 2, 3, 4, 5, 6, 8, 9, 10, 11, 12, 13\} \\
\bar{1}\bar{1} &= \{1, 2, 3, 4, 5, 6, 8, 9, 11, 12, 13\} \\
\bar{1}\bar{2} &= \{12\} \\
\bar{1}\bar{3} &= \{12, 13\}
\end{aligned} \tag{A.4}$$

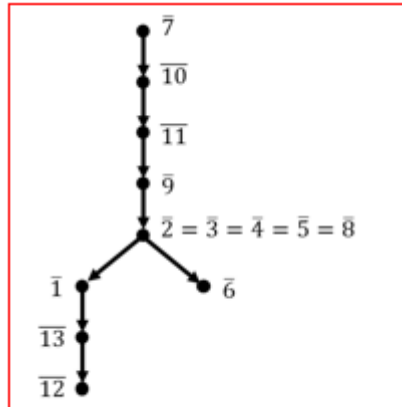


Figure A.10 Nesting of the reachability sets of Example 3 in Figure A.9.

Again, the reachability tree in Figure A.10 can be used to produce the reachability sets in (A.4). By starting with node $\overline{12}$ (which must be the single point set $\{12\}$) and $\bar{6}$ (which must be the single point set $\{6\}$) and working 'upwards', one can retrieve the other reachability sets in (A.4).

Example 4: A digraph in $\mathcal{G}_{\mathcal{H},27}$

This example considers a digraph in $\mathcal{G}_{\mathcal{H},27}$ that is close to the ones presented in Examples 2 and 3, which are both in $\mathcal{G}_{\mathcal{H},26}$, so each with one arc less. This time the edge $\{8, 9\}$ has remained from graph \mathcal{H} , which comprises of both the arcs $(9, 8)$ and $(8, 9)$. So this graph is a bit closer to the saturated digraph \mathcal{H} .⁴²⁾ Again the aim of this example is to see what the impact is of a small change of a digraph, concerning the orientation of one of the arcs.

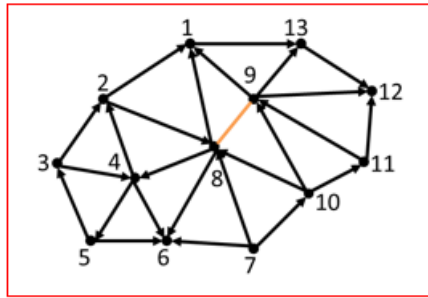


Figure A.11 A digraph in $\mathcal{G}_{\mathcal{H},27}$ used in Example 4, that is close to both the one in Figure A.5 and the one in Figure A.8.

In this case, as in case of the previous example, a small change in the digraph studied may lead to considerable changes in the reachability sets, as Figure A.12 shows. On the other hand, this example also shows that some of the reachability sets are not affected at all by this particular change of the structure of the digraph.

We can write down the equations that the reachability sets have to obey:

⁴²⁾ Saturated within the class of digraphs with \mathcal{H} as the underlying graph.

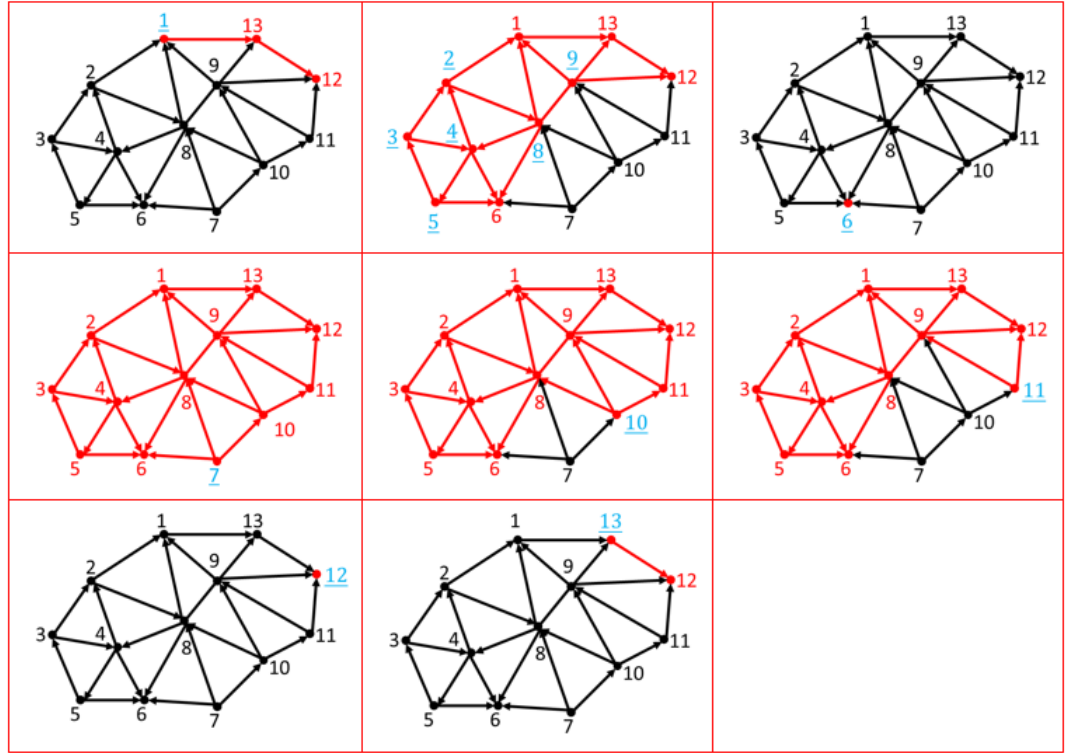


Figure A.12 Reachability sets for the nodes of the digraph in Example 4 in Figure A.11.

$$\begin{aligned}
 \bar{1} &= \{1\} \cup \bar{13} \\
 \bar{2} &= \{2\} \cup \bar{1} \cup \bar{8} \\
 \bar{3} &= \{3\} \cup \bar{2} \cup \bar{4} \\
 \bar{4} &= \{4\} \cup \bar{2} \cup \bar{5} \cup \bar{6} \\
 \bar{5} &= \{5\} \cup \bar{3} \cup \bar{6} \\
 \bar{6} &= \{6\} \\
 \bar{7} &= \{7\} \cup \bar{6} \cup \bar{8} \cup \bar{10} \\
 \bar{8} &= \{8\} \cup \bar{1} \cup \bar{4} \cup \bar{6} \cup \bar{9} \\
 \bar{9} &= \{9\} \cup \bar{1} \cup \bar{8} \cup \bar{12} \cup \bar{13} \\
 \bar{10} &= \{10\} \cup \bar{8} \cup \bar{9} \cup \bar{11} \\
 \bar{11} &= \{11\} \cup \bar{9} \cup \bar{12} \\
 \bar{12} &= \{12\} \\
 \bar{13} &= \{13\} \cup \bar{12}.
 \end{aligned} \tag{A.5}$$

As before, we can solve (A.5) by repeated substitution of the nonterminal symbols, that is, indicated as numbers with a bar on top. The result is presented in (A.6):

$$\begin{aligned}
\bar{1} &= \{1, 12, 13\} \\
\bar{2} = \bar{3} = \bar{4} = \bar{5} = \bar{8} = \bar{9} &= \{1, 2, 3, 4, 5, 6, 8, 9, 12, 13\} \\
\bar{6} &= \{6\} \\
\bar{7} &= \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13\} \\
\bar{10} &= \{1, 2, 3, 4, 5, 6, 8, 9, 10, 11, 12, 13\} \\
\bar{11} &= \{1, 2, 3, 4, 5, 6, 8, 9, 11, 12, 13\} \\
\bar{12} &= \{12\} \\
\bar{13} &= \{12, 13\}
\end{aligned} \tag{A.6}$$

We can represent the reachability sets in (A.6) in a tree, to show the nesting of these sets graphically. See Figure A.13.

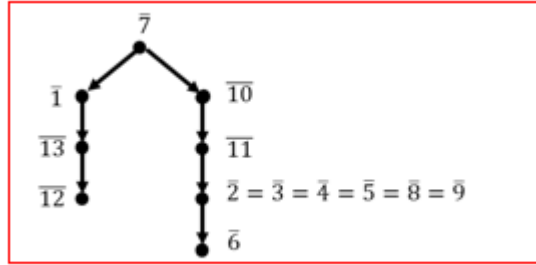


Figure A.13 Nesting of the reachability sets in Figure A.12 of Example 4.

Example 5: Another digraph in $\mathcal{G}_{\mathcal{H},27}$

In the previous two examples so far we modified a specific arc in \mathcal{H} . We now look at the effect of adding another arc, so that we obtain another digraph in $\mathcal{G}_{\mathcal{H},27}$. We then obtain the example shown in Figure A.14.

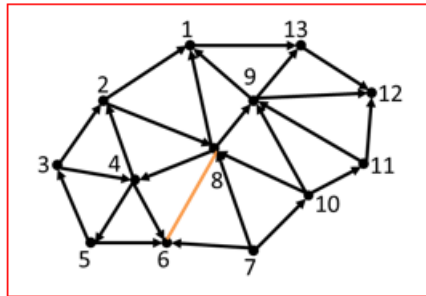


Figure A.14 Another digraph in $\mathcal{G}_{\mathcal{H},27}$ used in Example 5. Note the edge $\{6, 8\}$.

We can write down the reachability equations for each node:

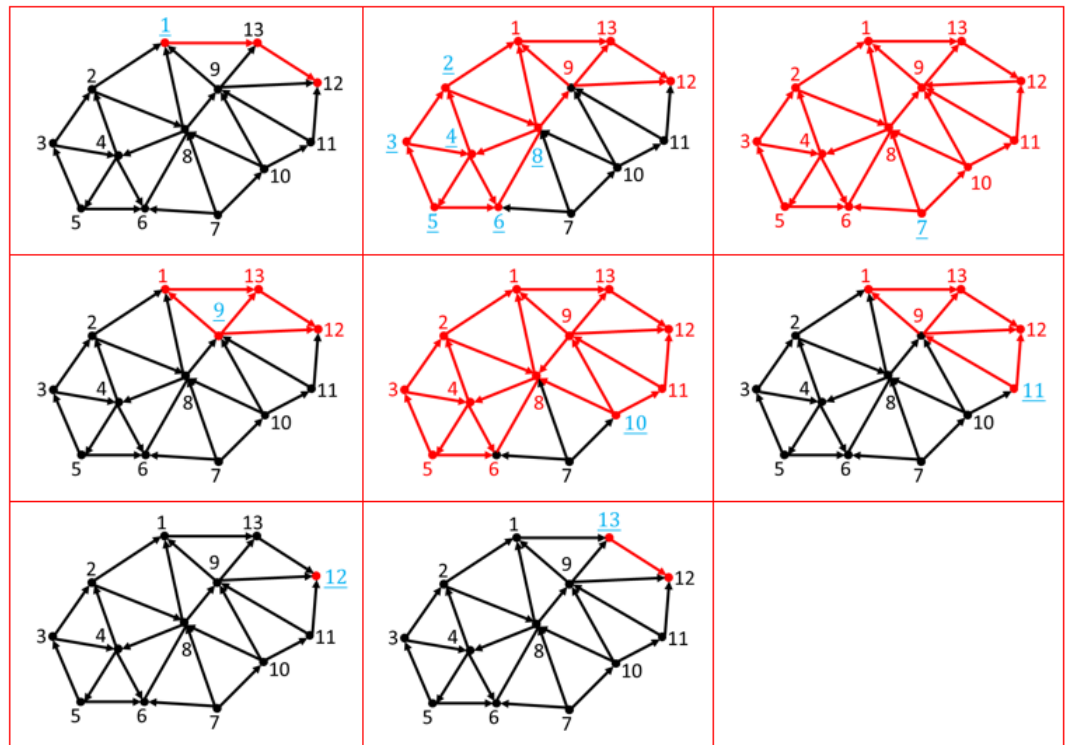


Figure A.15 Reachability sets for the nodes of the digraph in Figure A.14 in Example 5.

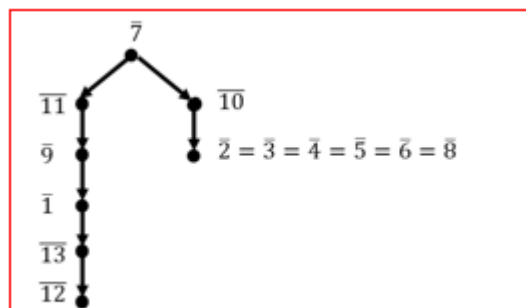


Figure A.16 Nesting of the reachability sets in Figure A.15 in Example 5.

$$\begin{aligned}
\bar{1} &= \{1\} \cup \bar{13} \\
\bar{2} &= \{2\} \cup \bar{1} \cup \bar{8} \\
\bar{3} &= \{3\} \cup \bar{2} \cup \bar{4} \\
\bar{4} &= \{4\} \cup \bar{2} \cup \bar{5} \cup \bar{6} \\
\bar{5} &= \{5\} \cup \bar{3} \cup \bar{6} \\
\bar{6} &= \{6\} \cup \bar{8} \\
\bar{7} &= \{7\} \cup \bar{6} \cup \bar{8} \cup \bar{10} \\
\bar{8} &= \{8\} \cup \bar{1} \cup \bar{4} \cup \bar{6} \cup \bar{9} \\
\bar{9} &= \{9\} \cup \bar{1} \cup \bar{8} \cup \bar{12} \cup \bar{13} \\
\bar{10} &= \{10\} \cup \bar{8} \cup \bar{9} \cup \bar{11} \\
\bar{11} &= \{11\} \cup \bar{9} \cup \bar{12} \\
\bar{12} &= \{12\} \\
\bar{13} &= \{13\} \cup \bar{12}.
\end{aligned} \tag{A.7}$$

Solving the equations in (A.7), as before, we obtain:

$$\begin{aligned}
\bar{1} &= \{1, 12, 13\} \\
\bar{2} = \bar{3} = \bar{4} = \bar{5} = \bar{6} = \bar{8} &= \{1, 2, 3, 4, 5, 6, 8, 9, 12, 13\} \\
\bar{7} &= \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13\} \\
\bar{9} &= \{1, 9, 12, 13\} \\
\bar{10} &= \{1, 2, 3, 4, 5, 6, 8, 9, 10, 11, 12, 13\} \\
\bar{11} &= \{1, 9, 11, 12, 13\} \\
\bar{12} &= \{12\} \\
\bar{13} &= \{12, 13\}
\end{aligned} \tag{A.8}$$

Example 6: A digraph in $\mathcal{G}_{\mathcal{H},28}$

To get some feeling about the effect of adding an extra arc to \mathcal{H} , we consider another example, which is shown in Figure A.17. Here two arcs are added to the initial digraph in Example 2, namely the one in Figure A.5. In the following examples we study the effect of adding more edges to the digraph. With every step the resulting digraph is one step closer to the saturated digraph, that is \mathcal{H} in Figure A.1.

The reachability sets for each node are presented in Figure A.18. The first thing that is evident is that there are less different such sets when we compare it to the examples considered so far.

By now it should be clear how the reachability sets can be computed. So we shall not present the equations that express them in terms of unknowns. We also shall not present the solution, as it can be gleaned from Figure A.18.

In Figure A.19 the tree is presented that shows how the reachability sets in Figure A.18 are ordered by inclusion.

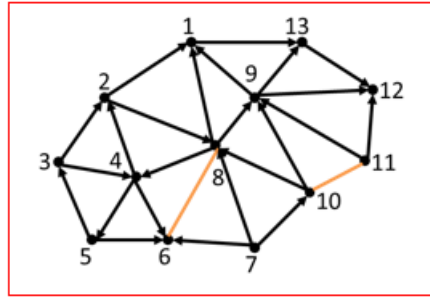


Figure A.17 A digraph in $\mathcal{G}_{\mathcal{H},28}$ in Example 6. Note the edges $\{6,8\}$ and $\{10,11\}$.

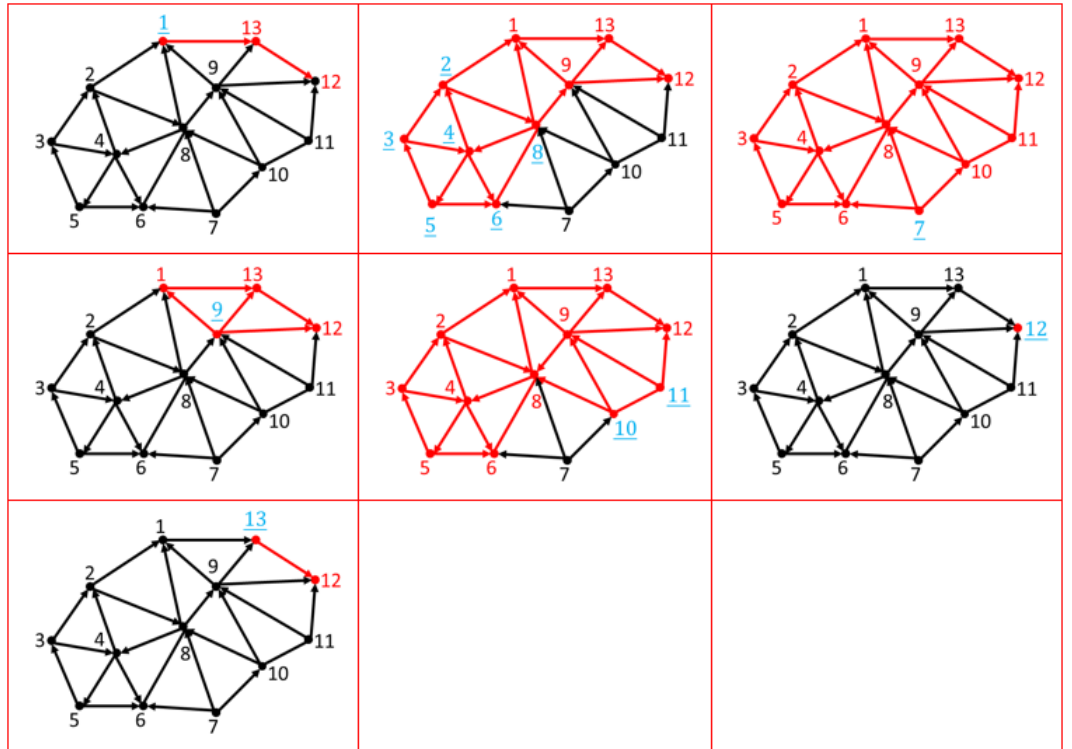


Figure A.18 Reachability sets for the nodes of the digraph from Example 6 in Figure A.17.

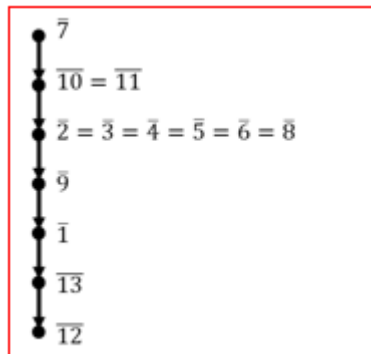


Figure A.19 Nesting of the reachability sets in Figure A.18 used in Example 6.

Example 7: A digraph in $\mathcal{G}_{\mathcal{H},29}$

Compared to the previous example the digraph in the present example has one more arc. So we get one more step closer to the saturated graph \mathcal{H} . The resulting digraph is shown in Figure A.20. Compared to the original digraph it has three more arcs.

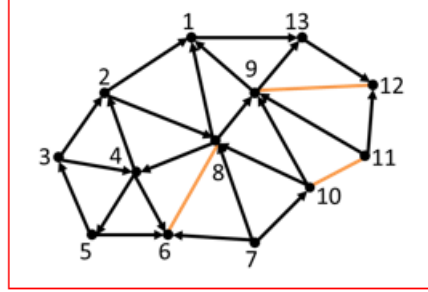


Figure A.20 A digraph in $\mathcal{G}_{\mathcal{H},29}$ used in Example 7. Note the edges $\{6,8\}$, $\{9,12\}$ and $\{10,11\}$.

The reachability sets are now shown in Figure A.21. This time there are only four different reachability sets, three less than in Example 6.

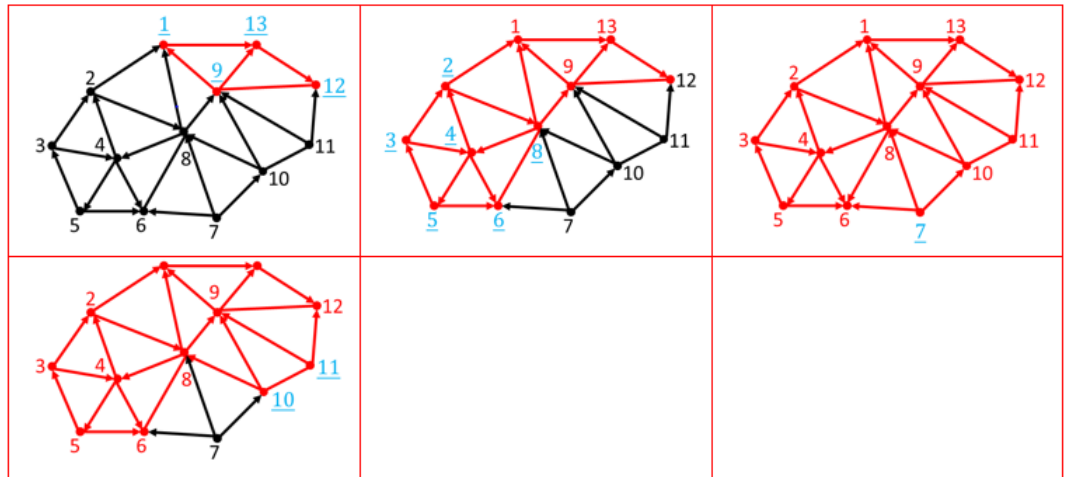


Figure A.21 Reachability sets for the nodes of the digraph in Figure A.20 used in Example 7.

Again we do not present the solution method, as it should be clear. The results are implicit in Figure A.21 and therefore will not be represented explicitly as well. However, the way they are nested is shown in Figure A.22. The tree is more 'compressed' than any of the previous trees. As the following examples show, this trend will continue.

Example 8: A digraph in $\mathcal{G}_{\mathcal{H},30}$

For this example we have added one more arc to the digraph of Example 7, that is, the one which is represented in Figure A.20. So we are now yet one more step closer to the saturated graph \mathcal{H} . The resulting digraph is shown in Figure A.23. Compared to the original digraph it has four more arcs.

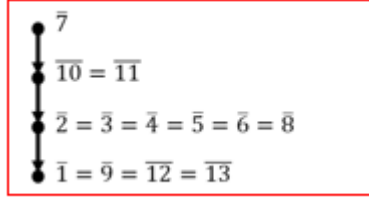


Figure A.22 Nesting of the reachability sets in Figure A.21 used in Example 7.

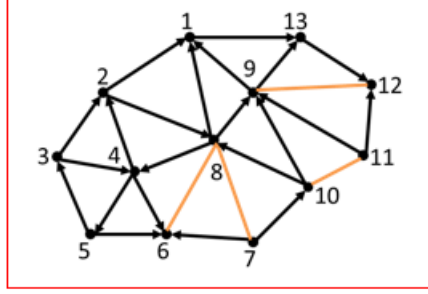


Figure A.23 A digraph in $\mathcal{G}_{\mathcal{H},30}$ used in Example 8. Note the edges $\{6, 8\}$, $\{7, 8\}$, $\{9, 12\}$ and $\{10, 11\}$.

In Figure A.24 the reachability sets of the digraph in Figure A.23 are shown. Now there are only two different such sets. This shows that the digraph in Figure A.23 is close to the saturated digraph in terms of reachability.

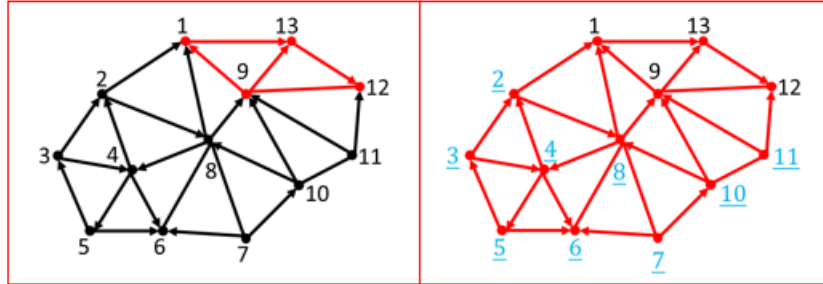


Figure A.24 Reachability sets for the nodes of the digraph in Figure A.23 used in Example 8.

The ordering of the reachability sets by inclusion is presented in Figure A.25.

Example 9: A digraph in $\mathcal{G}_{\mathcal{H},31}$

With the present example we reach the finale, so to speak, of the incremental example of digraphs with increasing numbers of arcs. We again have added an arc, so that the resulting digraph has 5 more arcs compared to the initial digraph. The resulting digraph is presented in Figure A.26.

In this case we find that all points have the same reachability set, namely the one equal to the node set of all the digraphs considered in the examples in the present appendix. See Figure A.27.

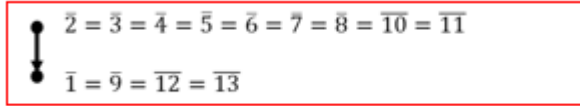


Figure A.25 Nesting of the reachability sets in Figure A.24 used in Example 8.

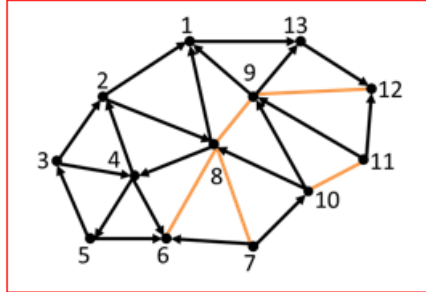


Figure A.26 A digraph in $\mathcal{G}_{\mathcal{H},31}$ used in Example 9. Note the edges $\{6, 8\}$, $\{7, 8\}$, $\{8, 9\}$, $\{9, 12\}$ and $\{10, 11\}$.

So the digraph in Figure A.26 has the saturated digraph \mathcal{H} as its transitive closure.

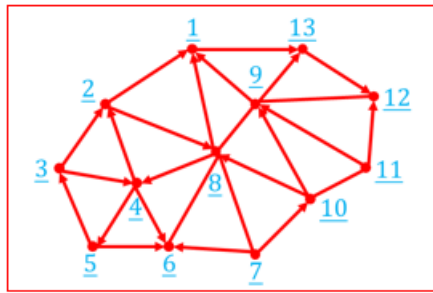


Figure A.27 Reachability sets for the nodes of the digraph in Figure A.26 used in Example 9.

The tree showing the ordering of the reachability sets has shrunk to a single point in this example. See Figure A.28.

So this example is a kind of final result: adding more arcs will not yield digraphs that show a different 'reachability behaviour'. They all have the same transitive closure, namely the entire node set. So nothing new will be shown concerning reachability.

Example 10: A digraph with reachability sets with smaller variation in size

The examples presented above have reachability sets that vary considerably in size. From one-point sets to the entire node set of the digraph. In the present example the sizes of the various reachability sets do not differ so much. The example is just to show that digraphs exist that are less diverse.

The digraph we are interested in in the present example is presented in Figure A.29.

The reaches of the nodes in digraph in Figure A.29 are presented in Figure A.30.

$$\bullet \quad \bar{1} = \bar{2} = \bar{3} = \bar{4} = \bar{5} = \bar{6} = \bar{7} = \bar{8} = \bar{9} = \bar{10} = \bar{11} = \bar{12} = \bar{13}$$

Figure A.28 Nesting of the reachability sets in Figure A.27 used in Example 9.

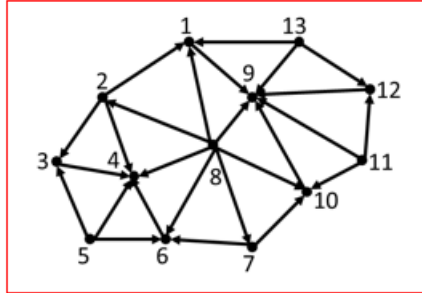


Figure A.29 A digraph with smaller reachability sets.

Example 11: A digraph with drainage areas

So far in the examples in this appendix we have considered reachability sets from nodes in digraphs, that is, sets that can be reached from a given node. It is also of interest to know the drainage set of a node v in a digraph, which consists of the nodes in the digraph from which paths lead to v .

It is a very simple to compute the drainage set of v : reverse the arcs in the original digraph and compute the reachability set of v in this dual of the original digraph. Reversing the arcs in the digraph yields a digraph with A' as its adjacency matrix, if A denotes the adjacency matrix of the original digraph.

We illustrate this using the graph in Figure A.29. Its dual is shown in Figure A.31.

The reachability sets of the digraph in Figure A.31 are shown in Figure A.32. These are the drainage sets of the original digraph in Figure A.29.

So from Figure A.32 we can infer that point 1, for instance, can be reached, in the original digraph in Figure A.29, from points 1, 2, 8, 13; and point 9 can be reached in this digraph from points 1, 2, 8, 9, 10, 11, 12, 13. Etcetera.

Reflecting on the examples above

Here we consider the examples in Appendix A and try to see what can we learn from them with respect to the complexity of the class of digraphs with the same underlying graph. First of all, we should note that the pictures shown are very direct. But the same information can be obtained more quickly by computing the transitive closure of the adjacency matrix of the digraph in question.

The digraph in the first example is regular and forms a partial order. The reachability sets are nested in a linear fashion, so to speak. In the other examples considered, the reachability set were ordered in a more complicated partial ordering. So this may perhaps be taken as a hint that

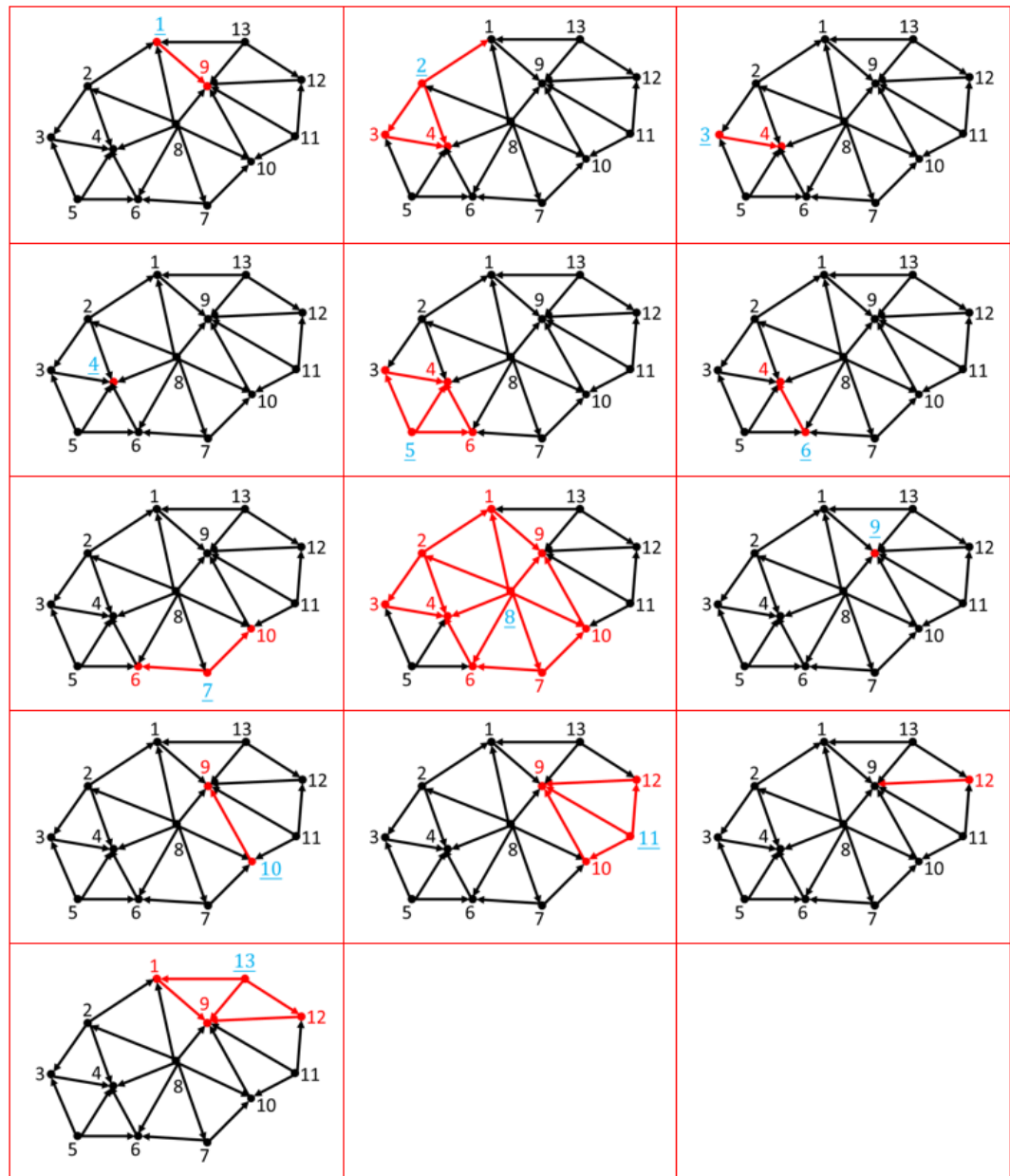


Figure A.30 The reachability sets of the nodes of the digraph in Figure A.29.

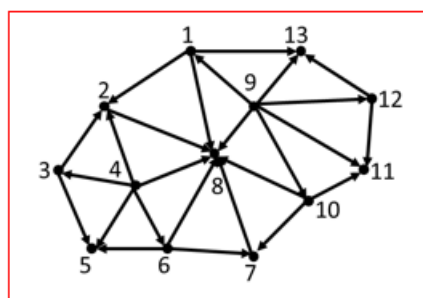


Figure A.31 The dual of the digraph in Figure A.29.

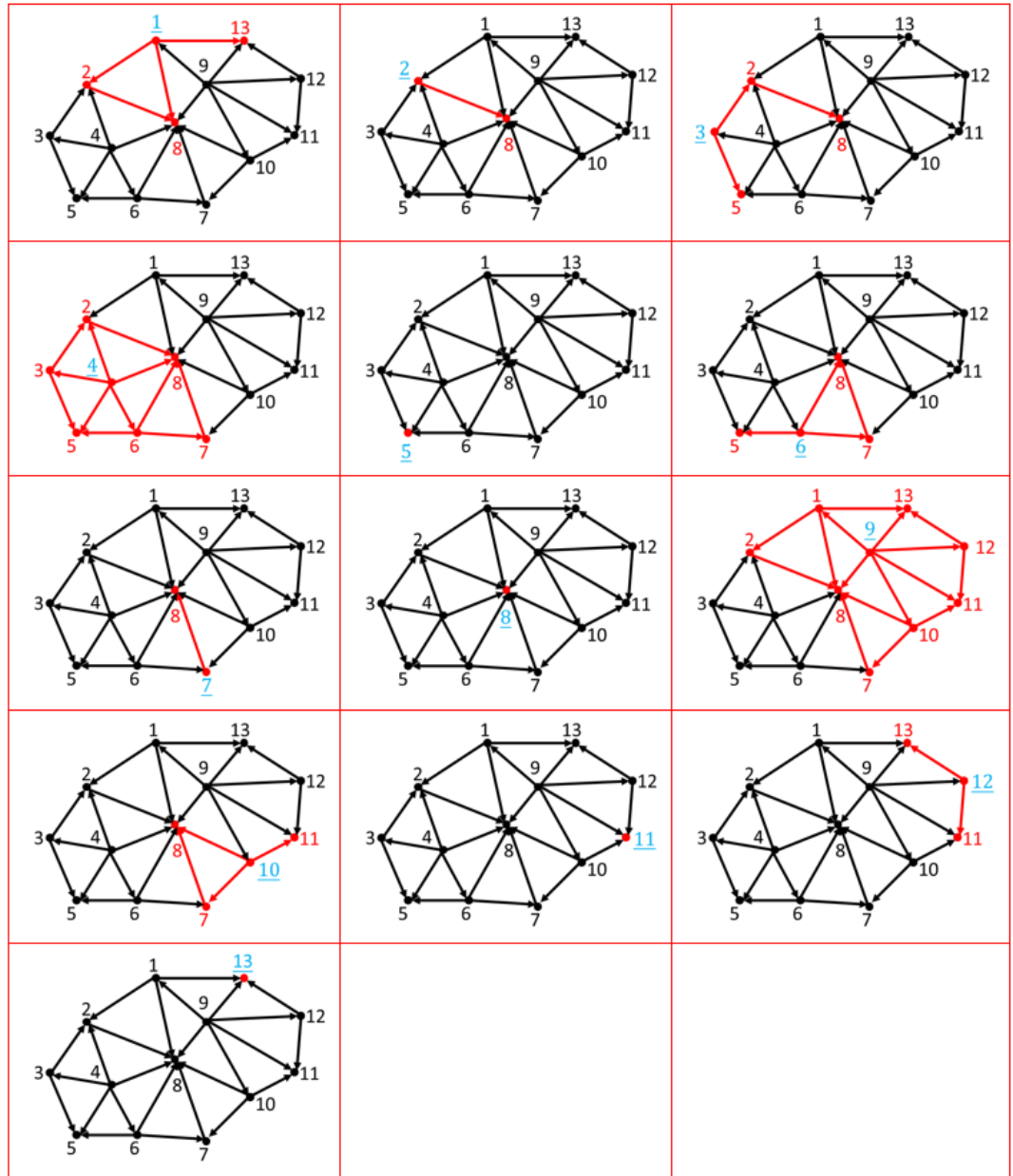


Figure A.32 Reachability sets of the digraph in Figure A.31. These are the drainage sets of its dual in Figure A.29.

the structure of the digraph corresponding to the nesting of the reachability sets, can be used to quantify the complexity of the digraphs in the class considered: the more it branches, the more complex the corresponding digraph is.

In some of the examples, some nodes had the same reachability sets, although the digraphs differ. This implies that these nodes are in a sense equivalent. One can take them as a single node. This node can then be connected to the other nodes in the digraph. It yields a new digraph with fewer nodes that represents a partial order.

Some of the examples show that small changes in the structure of a digraph may have considerable impact on some reachability sets, and none at all on other such sets. This suggests some form of locality.

However, from this one may not conclude that such changes only have local impact: these impacts may be global, and affecting many reachability sets.

The examples in Appendix A all have in common that there is at least one node for which the reachability set is the entire set of nodes. This is probably not typical for most networks encountered in practice. So the examples shown there are not intended to exhibit species of digraphs of very different complexities. Its purpose was to show how, starting with some digraph, changing the orientation of a few arcs or adding a small number of them affects the reachability of the various nodes.

B Average distances in graphs: Examples

In this appendix we present some examples to illustrate the complexity measure based on average distances of different points in graphs as treated in Section 6. Two methods are illustrated that are presented there to compute complexity measures based on average distances. The first one applies it to the graph in question directly. The second one applies it to the line graph of this graph. Each method is described in a separate section.

B.1 Using the graphs directly

In Figure B.1 several example graphs are shown that we will use to illustrate the complexity measure based on average distance of distinct nodes. The metric used is the one for which the length of each edge in the graph is 1, so that the length of a path equals the number of edges situated on the path.

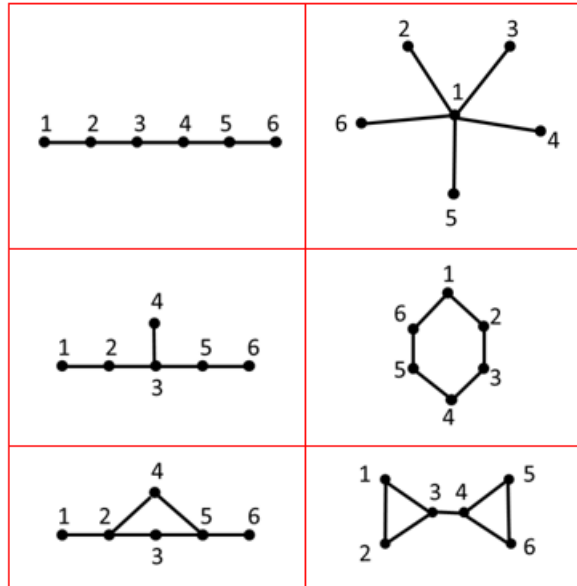


Figure B.1 Examples of some graphs with 6 nodes and varying numbers of edges.

The distance matrix D_{tl} of the graph G_{tl} on the top left cell of Figure B.1 is

$$D_{tl} = \begin{pmatrix} 0 & 1 & 2 & 3 & 4 & 5 \\ 1 & 0 & 1 & 2 & 3 & 4 \\ 2 & 1 & 0 & 1 & 2 & 3 \\ 3 & 2 & 1 & 0 & 1 & 2 \\ 4 & 3 & 2 & 1 & 0 & 1 \\ 5 & 4 & 3 & 2 & 1 & 0 \end{pmatrix} \quad (B.1)$$

The distance matrix D_{tr} of the graph G_{tr} on the top right cell of Figure B.1 is

$$D_{tr} = \begin{pmatrix} 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 2 & 2 & 2 & 2 \\ 1 & 2 & 0 & 2 & 2 & 2 \\ 1 & 2 & 2 & 0 & 2 & 2 \\ 1 & 2 & 2 & 2 & 0 & 2 \\ 1 & 2 & 2 & 2 & 2 & 0 \end{pmatrix} \quad (B.2)$$

The distance matrix D_{ml} of the graph G_{ml} on the mid left cell of Figure B.1 is

$$D_{ml} = \begin{pmatrix} 0 & 1 & 2 & 3 & 3 & 4 \\ 1 & 0 & 1 & 2 & 2 & 3 \\ 2 & 1 & 0 & 1 & 1 & 2 \\ 3 & 2 & 1 & 0 & 2 & 3 \\ 3 & 2 & 1 & 2 & 0 & 1 \\ 4 & 3 & 2 & 3 & 1 & 0 \end{pmatrix} \quad (B.3)$$

The distance matrix D_{mr} of the graph G_{mr} on the mid right cell of Figure B.1 is

$$D_{mr} = \begin{pmatrix} 0 & 1 & 2 & 3 & 2 & 1 \\ 1 & 0 & 1 & 2 & 3 & 2 \\ 2 & 1 & 0 & 1 & 2 & 3 \\ 3 & 2 & 1 & 0 & 1 & 2 \\ 2 & 3 & 2 & 1 & 0 & 1 \\ 1 & 2 & 3 & 2 & 1 & 0 \end{pmatrix} \quad (B.4)$$

The distance matrix D_{bl} of the graph G_{bl} on the bottom left cell of Figure B.1 is

$$D_{bl} = \begin{pmatrix} 0 & 1 & 2 & 2 & 3 & 4 \\ 1 & 0 & 1 & 1 & 2 & 3 \\ 2 & 1 & 0 & 2 & 1 & 2 \\ 2 & 1 & 2 & 0 & 1 & 2 \\ 3 & 2 & 1 & 1 & 0 & 1 \\ 4 & 3 & 2 & 2 & 1 & 0 \end{pmatrix} \quad (B.5)$$

The distance matrix D_{br} of the graph G_{br} on the bottom right cell of Figure B.1 is

$$D_{br} = \begin{pmatrix} 0 & 1 & 1 & 2 & 3 & 3 \\ 1 & 0 & 1 & 2 & 3 & 3 \\ 1 & 1 & 0 & 1 & 2 & 2 \\ 2 & 2 & 1 & 0 & 1 & 1 \\ 3 & 3 & 2 & 1 & 0 & 1 \\ 3 & 3 & 2 & 1 & 1 & 0 \end{pmatrix} \quad (B.6)$$

With the matrices in (B.1), (B.2), (B.3), (B.4), (B.5) and (B.6) we can compute the values for the complexity κ_d and the Wiener index \mathcal{W} . The results are collected in Table B.1.

Graph	$ V $	$ E $	κ_d	\mathcal{W}
G_{tr}	6	5	$1\frac{2}{3} = 1.67$	25
G_{ml}	6	5	$2\frac{1}{15} = 2.07$	31
G_{tl}	6	5	$2\frac{1}{3} = 2.33$	35
G_{mr}	6	6	$1\frac{4}{5} = 1.80$	27
G_{bl}	6	6	$1\frac{13}{15} = 1.87$	28
G_{br}	6	7	$1\frac{4}{5} = 1.80$	27

Table B.1 Results for the graphs in Figure B.1.

where κ_d is defined in equation (15) and $n = |V|$ denotes the number of nodes in G .

The complexity measure κ_d and the Wiener index \mathcal{W} are presented in Table B.1. The results were sorted, first on the number of edges and then on the scores of κ_d .

Within the classes of graphs with 5 or 6 edges, κ_d and \mathcal{W} seem to make sense as complexity measures if one takes lower scores to mean higher complexities, and vice versa. In the group of graphs with 6 nodes and 5 edges G_{tl} is then the simplest graph and G_{tr} the most complex one; G_{ml} has a complexity in between these two. So actually, $1/\kappa_d$ and $1/\mathcal{W}$ act more like complexity measures, in the sense that higher values indicate more complex graphs, that is, graphs with more branching.

Of course, the number of example graphs is too small to draw far reaching conclusions from the results in Table B.1.

B.2 Using line graphs

We now consider the line graphs for each of the graphs presented in Figure B.1. These are presented in Figure B.2. The graphs are on the left-hand side and the corresponding line graphs are on the right-hand side of the table. A graph and its corresponding line graph are represented in the same row.

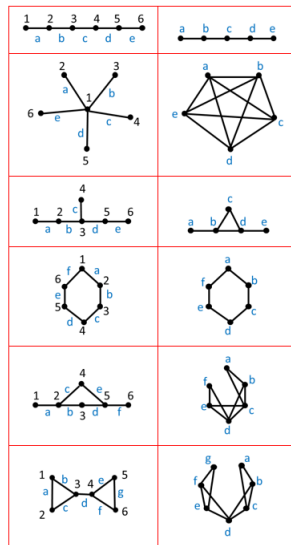


Figure B.2 Examples of graphs and their line graphs. A graph (in the left column) and its line graph (on the right column) are on the same row.

To apply (17) to the line graphs in Figure (B.2) we first determine the six distance matrices. They are as follows.

The distance matrix D_1^ℓ of the line graph $G_{L,1}$ in the first line of Figure B.2 is

$$D_1^\ell = \begin{pmatrix} 0 & 1 & 2 & 3 & 4 \\ 1 & 0 & 1 & 2 & 3 \\ 2 & 1 & 0 & 1 & 2 \\ 3 & 2 & 1 & 0 & 1 \\ 4 & 3 & 2 & 1 & 0 \end{pmatrix} \quad (\text{B.7})$$

The distance matrix D_2^ℓ of the line graph $G_{L,2}$ in the second line of Figure B.2 is the circulant matrix

$$D_2^\ell = \begin{pmatrix} 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 0 \end{pmatrix} \quad (\text{B.8})$$

The distance matrix D_3^ℓ of the line graph $G_{L,3}$ in the third line of Figure B.2 is

$$D_3^\ell = \begin{pmatrix} 0 & 1 & 2 & 2 & 3 \\ 1 & 0 & 1 & 1 & 2 \\ 2 & 1 & 0 & 1 & 2 \\ 2 & 1 & 1 & 0 & 1 \\ 3 & 2 & 2 & 1 & 0 \end{pmatrix} \quad (\text{B.9})$$

The distance matrix D_4^ℓ of the line graph $G_{L,4}$ in the fourth line of Figure B.2 is the circulant matrix

$$D_4^\ell = \begin{pmatrix} 0 & 1 & 2 & 3 & 2 & 1 \\ 1 & 0 & 1 & 2 & 3 & 2 \\ 2 & 1 & 0 & 1 & 2 & 3 \\ 3 & 2 & 1 & 0 & 1 & 2 \\ 2 & 3 & 2 & 1 & 0 & 1 \\ 1 & 2 & 3 & 2 & 1 & 0 \end{pmatrix} \quad (\text{B.10})$$

The distance matrix D_5^ℓ of the line graph $G_{L,5}$ in the fifth line of Figure B.2 is

$$D_5^\ell = \begin{pmatrix} 0 & 1 & 1 & 2 & 2 & 3 \\ 1 & 0 & 1 & 1 & 2 & 3 \\ 1 & 1 & 0 & 1 & 1 & 2 \\ 2 & 1 & 1 & 0 & 1 & 1 \\ 2 & 2 & 1 & 1 & 0 & 1 \\ 3 & 3 & 2 & 1 & 1 & 0 \end{pmatrix} \quad (\text{B.11})$$

The distance matrix D_6^ℓ of the line graph $G_{L,6}$ in the sixth line of Figure B.2 is

$$D_6^\ell = \begin{pmatrix} 0 & 1 & 1 & 2 & 3 & 3 & 4 \\ 1 & 0 & 1 & 1 & 2 & 2 & 3 \\ 1 & 1 & 0 & 1 & 2 & 2 & 3 \\ 2 & 1 & 1 & 0 & 1 & 1 & 2 \\ 3 & 2 & 2 & 1 & 0 & 1 & 1 \\ 3 & 2 & 2 & 1 & 1 & 0 & 1 \\ 4 & 3 & 3 & 2 & 1 & 1 & 0 \end{pmatrix} \quad (\text{B.12})$$

Table B.2 contains the values of $\kappa_d(G_L)$ and $\mathcal{W}(G_L)$. Their purpose is to quantify the complexities of the original graphs. $\kappa_d(G_L)$ gives higher values to less complex graphs (less branching), as we saw before. The results of Table B.1 and Table B.2 point in a similar direction.

Graph	Line graph	$ V $	$ E $	$\kappa_d(G_L)$	$\mathcal{W}(G_L)$
G_1	$G_{L,1}$	5	4	2	20
G_2	$G_{L,2}$	5	10	1	10
G_3	$G_{L,3}$	5	5	$1\frac{3}{5} = 1.60$	16
G_4	$G_{L,4}$	6	6	$1\frac{4}{5} = 1.80$	27
G_5	$G_{L,5}$	6	10	$1\frac{8}{5} = 1.53$	23
G_6	$G_{L,6}$	7	7	$1\frac{17}{21} = 1.81$	38

Table B.2 Results for the graphs in Figure B.2.

C Notation

Here we define the most important notation that is used in the present paper. The items are alphabetically ordered. In the explanation of the notation concepts are used that are explained in Appendix D.

- \Rightarrow : ‘ $a \Rightarrow b$ ’ is an abbreviation for ‘ a implies b ’ or, alternatively, ‘if a then b ’.
- $|\cdot|$: $|S|$ is the cardinality of a finite set S , that is, the number of elements of S .
- A : Adjacency matrix of a network.
- A' : Transpose of the matrix A .
- A^* : Transitive closure of adjacency matrix A .
- A^\downarrow : Transitive reduction of adjacency matrix A .
- E : Arc set or edge set. An edge v, w can be represented by two arcs: the arc (a, b) and its counter-arc (b, a) .
- $G = (V, E)$: A network with node set V and edge or arc set E .
- G^c : The compressed graph of a graph G .
- G^\leftarrow : The network that is the reverse of the digraph $G = (V, E)$. $G^\leftarrow = (V, E^\leftarrow)$ and $(b, a) \in E^\leftarrow$ if $(a, b) \in E$. The adjacency matrix of G^\leftarrow is A' , the transposed of A .
- G^* : The network that is the transitive closure of the network $G = (V, E)$. $G^* = (V, E^*)$, where E^* is the set of arcs corresponding to the adjacency matrix A^* .
- G^\downarrow : The network that is the transitive reduction of the network $G = (V, E)$. $G^\downarrow = (V, E^\downarrow)$, where E^\downarrow is the set of arcs corresponding to the adjacency matrix of A^\downarrow .
- $\mathcal{G}_{G,k}$: The set of digraphs with $G = (V, E)$ as its underlying graph and with k arcs, with $|E| \leq k \leq 2|E|$.
- J : The all 1s matrix. A square matrix with all entries equal to 1. The order of J may vary, but it should be clear in a particular context. To indicate it explicitly J_n is used if it is an $n \times n$ matrix.
- π : in a network G , π denotes a path in G .
- $\ell(\pi)$: if π is a path in a network, $\ell(\pi)$ denotes the length of path π , that is, the number of edges it contains. If an edge appears multiple times in π it is counted for the number of times it appears on π .
- $\pi_{v,w}$: in a network G , $\pi_{v,w}$ denotes a path in G from node v to node w .
- π^\leftarrow : in a network G , if π denotes a path in G , π^\leftarrow denotes the reverse path. It is formally obtained by reversing the order of the arcs in the path π and by reversing the order of the nodes in each arc. If any of the counter-arcs thus obtained is not a part of G , the reversed path is not in G .
- $\pi_{w,v}^\leftarrow$: in a network G $\pi_{w,v}^\leftarrow$ denotes the reverse path of π starting in w and ending in v . It may not be a path in G .
- Tr : The Trace operator, with $Tr(M) = \sum_{i=1}^n m_{ii}$, if $M = (m_{ij})$ an $n \times n$ matrix.
- V : Vertex set of a network $G = (V, E)$.
- \underline{v} : Entrance point of a node v in a digraph.
- \overline{v} : Reachability set of node v in a digraph.
- $v \rightsquigarrow w$: If v and w are nodes in a network G , there is a path in G from v to w .
- $v \xrightarrow{\pi} w$: If v and w are nodes in a network G , there is a path π in G from v to w .
- $v \not\rightsquigarrow w$: there is no path from node v to node w in a network G .
- Υ : the set of acyclic digraphs with a single source and sink. Such digraphs are called ‘routing graphs’ in [13] (although they are digraphs).

D Glossary

Here we describe a few key-concepts in the present paper. These concepts are alphabetically ordered. Some of the concept definitions are either taken literally from [14] or in a slightly modified form.

Adjacency matrix 0-1 matrix where element (i, j) indicated if there is an arc from i to j (if the value = 1) or not (if the value = 0). An adjacency matrix can be viewed as a representation of a (di)graph. In case of a graph it is symmetric. In case of a digraph it needs not to be symmetric.

Arc An ordered pair of nodes (a, b) . In a picture an arc (a, b) is denoted by an arrow pointing from a to b . The node a is called the tail and b is called the head of the arc (a, b) .

Arc rank Arc ranks are weights on arcs of a digraph, to indicate their importance. They are computed from node ranks in such a way that the sum of the arc ranks of the arcs into a node equal its node rank. Likewise, the sum of the arc ranks of the arcs out of a node are equal to its node rank.

Augmented digraph A digraph that is considered as a kind of benchmark, to which some arcs have been added, in the case of the present paper, to see how the complexity of this new digraph has changed with respect to that of the benchmark. If D is a digraph in $\mathcal{G}_{G,k}$, the augmentation is a process of adding counter-arcs in cases where this is possible, i.e. for those nodes a and b of D for which either (a, b) or (b, a) is an arc but not both.

Border point A point in a neighbourhood that is linked to at least one point outside of this set.

Center point The reference point of a neighbourhood in a network defined with the help of a metric or quasi-metric. It is comparable to the center of a circle or disk in geometry.

Circulant matrix A square matrix of order n of the form

$$C = \begin{pmatrix} c_1 & c_2 & \cdots & c_{n-1} & c_n \\ c_2 & c_3 & \cdots & c_n & c_1 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ c_n & c_1 & \cdots & c_{n-2} & c_{n-1} \end{pmatrix}$$

is a circulant matrix which is fully specified by the vector (c_1, \dots, c_n) , which is the first row of C .

Complexity of a digraph This measures how the digraph differs from a digraph with the same underlying graph which has full reachability, that is, which is connected.

Complexity of a graph In the present paper several such quantities are defined to express the way a graph branches.

Compressed graph A graph G^c is the compressed graph of a graph G if each linear subgraph of G is replaced by an edge in G^c .

Connected network A network for which each pair of points v, w can be connected by a path in the network, which is denoted as $v \rightsquigarrow w$. In a graph, if $v \rightsquigarrow w$ then $w \rightsquigarrow v$. In a digraph $v \rightsquigarrow w$ does not automatically imply $w \rightsquigarrow v$. And if it does, the path from w to v may not be the reversed path connecting v to w .

Continuous search A search procedure in a graph $G = (V, E)$ in which nodes are visited in such a way that consecutive points are adjacent, that is form an edge in E . In other words if $(a_1, a_2, \dots, a_\ell)$ are the nodes visited in the actual order, this is a path in G .

Contraction A process to simplify a digraph by eliminating linear sub-digraphs from it. These parts are replaced by a node, and arc or two connected arcs, depending on the situation. Contraction does not change the branching structure of a digraph.

Counter-arc If (a, b) is an arc in a digraph G then (b, a) . This may or may not be an arc in G .

Cut set Let $G = (V, E)$ be a (di)graph and $C \subseteq V$ be a non-empty set of nodes in V . The arcs/edges with one node in C and the other in $V \setminus C$ defines a cut set which is defined by the

partition $\{C, V \setminus C\}$ of V .

Cycle A path in a digraph or graph with the same start and finish.

Degree In a graph the number of edges attached to the node.

Degree matrix The degree matrix $D = (d_{ij})$ of a graph is a diagonal matrix with d_{ii} equal to the degree of node i .

Digraph See: Directed graph.

Directed graph Consists of nodes and arcs. An arc is an ordered pair of nodes. If (a, b) is an arc it means that node a is connected to node b . If (a, b) is an arc in a directed graph it is not necessarily the case that (b, a) is also an arc.

Distance See: Metric.

Drainage set The drainage set of a node v in digraph G is the reachability set of v in the reverse digraph G^{\leftarrow} . It is the set of nodes a in G that lead to v , that is, such that $a \rightsquigarrow v$.

Dual digraph The dual of a digraph $G = (V, E)$ is the digraph $G^{\leftarrow} = (V^{\leftarrow}, E^{\leftarrow})$ with $V^{\leftarrow} = V$ and $(a, b) \in E^{\leftarrow}$ iff $(b, a) \in E$. So the dual of a digraph is the same digraph but with the arcs reversed. The dual of the dual of a digraph G is G itself. If G is a graph its dual is G itself. That is, a graph is self-dual.

Edge In a graph, an edge is a set $\{a, b\}$ of two nodes a and b . In a picture an edge is often represented by a line segment or arc without arrow heads (as there is no direction). Viewed in a digraph context an edge $\{a, b\}$ is represented by the arcs (a, b) and (b, a) .

Endpoints In edge $e = \{a, b\}$ in a graph the node a and b are its endpoints. The endpoints a and b define the edge e .

Entrance point If v is a node in a digraph G . The nodes that can be reached from v form the reachability set \bar{v} . v is the entrance point of \bar{v} , which is stressed by using underlining: \underline{v} , to distinguish the entrance point from other points in the corresponding reachability set.

Essence of a network The most important part of a network, in terms of node rank or arc rank. The nodes (arcs) are selected using a threshold δ specified by a user. The network is then completed according to the original network. Selected nodes v, w are connected by arcs (v, w) if $v \rightsquigarrow w$ in the original network. Selected arcs imply selected nodes (the ones that define the arcs) and these nodes are connected if necessary as in case of selected nodes.

Extended neighbourhood subdigraph A neighbourhood subdigraph with arcs added. These arcs connect a node in the neighbourhood subdigraph with a node outside of it (or vice versa). By adding these arcs every node in the neighbourhood subdigraph has the same indegree and outdegree as in the original digraph.

Extended neighbourhood graph A neighbourhood subgraph with edges added. These edges connect a node in the neighbourhood subgraph with a node outside of it. By adding these edges every node in the neighbourhood subgraph has the same degree as in the original graph.

Finish The node where a path in a digraph or graph ends.

Full reach A digraph has full reach if from every of its nodes every other node can be reached.

This means that for every pair of nodes a and b there is a path from a to b (and reversely).

The path from b to a does not have to be the same as the reverse path from a to b .

Graph A graph is a self-dual digraph, that is, if G is a graph then $G^{\leftarrow} = G$. A graph is also a digraph with a symmetric adjacency matrix: $A' = A$.

Graph Laplacian The graph Laplacian of a graph G is the matrix $\Delta \triangleq D - A$, where Δ is the degree matrix of G and A its adjacency matrix. The graph Laplacian is the graph equivalent of the Laplace operator in calculus which is the divergence of the gradient, $\Delta \triangleq \nabla \cdot \nabla$, which in Cartesian coordinates x_i , can be written as $\Delta = \sum_{i=1}^n \frac{\partial^2}{\partial x_i^2}$. The graph Laplacian contains all the properties of its corresponding graph (as the adjacency matrix already does, which can be derived from the off-diagonal elements in the graph Laplacian). The only challenge is to derive

them from this matrix, which is not always easy.

Head The head of an arc (a, b) in a digraph is b . In a graphical representation the arc is typically depicted as an arrow or directed (curvy) line segment. The head is the dot (representing an end node) where the arc is pointing at.

Hub A node in a digraph with a (relatively) large node rank.

Iff If and only if.

Incidence For a graph $G = (V, E)$ we say that an edge $e \in E$ is incident with a node $a \in V$ if $v \in e$, which means that $e = \{v, x\}$ for some $x \in V$.

Indegree For a node $v \in V$ in a network $G = (V, E)$ the indegree is the number of arcs that end in v . That is, the number of nodes $a \in V$ such that $(a, v) \in E$. Its dual concept is outdegree.

Isolated node A node v in a network with $\Delta^{in}(v) = \Delta^{out}(v) = 0$. In case the network is a graph we have that $\Delta(v) = 0$.

IPF Iterative Proportional Fitting. "The iterative proportional fitting procedure ... is an iterative algorithm for estimating cell values of a contingency table such that the marginal totals remain fixed and the estimated table decomposes into an outer product."⁴³⁾ An outer product $\alpha \otimes \beta$ of two vectors α and β is a tensor (multilinear) product that equals $\alpha\beta'$, where α and β are presented as column vectors.

Line graph If $G = (V, E)$ is a graph, the line graph $G_L = (V_L, E_L)$ is a graph with $V_L \triangleq E$ and $E_L \triangleq \{(e, f) \in E \mid e \neq f \text{ and } e \cap f \neq \emptyset\}$. In words, a line graph represents the adjacency structure of the edges of a graph. That is, it indicates which edges have nodes in common.

Local complexity Complexity for a network defined on the basis of a neighbourhood system, that is a neighbourhood N_p for each point p in a network N . For each point p the neighbourhood N_p plays the role of N in case of a (global) complexity measure. In this way, for each point p we can compute a complexity, which depends on p as well as on the choice of its neighbourhood N_p .

Loop A loop is an arc (or edge) with the same head and tail (or endpoints). So if a is a point in a network, a loop on a would be an arc (a, a) or an edge $\{a, a\}$. In our approach we exclude graphs with loops.

Metric For a graph $G = (V, E)$ a metric is a function $d : V \times V \rightarrow \mathbb{R} \setminus \mathbb{R}^- \cup \{\infty\}$ such that $d(a, a) = 0$, $d(a, b) = d(b, a)$ and $d(a, c) \leq d(a, b) + d(b, c)$ for all nodes $a, b, c \in V$. Nodes a and b are in different connectivity components of G iff $d(a, b) = \infty$. Suppose that for each edge $\{a, b\}$ a nonnegative number d_{ab} is given, satisfying the constraints just mentioned (for each triangle $\{a, b, c\}$, where $\{a, b\}$, $\{b, c\}$ and $\{a, c\}$ are arcs we should have that the triangle inequality holds: $d_{ac} \leq d_{ab} + d_{bc}$). With this we can extend the metric to the entire graph: for a pair of nodes $a, b \in V$ in the same connectivity component of G we define Γ_{ab} is the set of paths in G connecting a and b . Then $d(a, b) \triangleq \min_{\gamma \in \Gamma_{ab}} \sum_{e=\{c,d\} \in \gamma} d_{cd}$, which the minimum length of the paths connecting a and b in terms of the d -values associated with the edges on the path. If a and b are in different connectivity components then $d(a, b) \triangleq \infty$.

Natural metric Metric defined for a graph, where each edge has length 1.

Neighbourhood of a node in a digraph A subset N of the node set V of the digraph $G = (V, E)$ of nodes close to a node $p \in N$, which is called the center point of the neighbourhood, in terms of a suitable quasimetric on G .

Neighbourhood of a node in a graph For a graph $G = (V, E)$ and a point $p \in V$, a neighbourhood N_p , intuitively, is a subset of V containing p and points close to p . Assuming a metric d on G we can define $N_p = \{q \in V \mid d(p, q) \leq \delta\}$ for some $\delta \geq 0$.

Neighbourhood subdigraph The subdigraph of a digraph $G = (V, E)$ with neighbourhood $N \subseteq V$ as a node set and with $F = \{(a, b) \in E \mid a, b \in N\} = (N \times N) \cap E$ as its set of arcs.

⁴³⁾ Definition from Wikipedia: https://en.wikipedia.org/wiki/Iterative_proportional_fitting.

Neighbourhood subgraph The subgraph of a graph $G = (V, E)$ with neighbourhood $N \subseteq V$ as a node set and with $F = \{\{a, b\} \in E | a, b \in N\} = (N \times N) \cap E$ as its set of edges.

Network A collective name for 'graph' and 'digraph'. Although, technically, a graph is a special kind of digraph, it is useful in practice to single out graphs as a special category of networks. Formally, a network is a digraph, and a graph is a special kind of digraph.

Nilpotent A square matrix M is nilpotent if there is a $p \in \mathbb{N}$ such that $M^p = 0$.

Node A node in a network is one of the ingredients that defines the network. In graphical displays a node is often depicted as a (solid or hollow) dot or circle. Nodes are used to define arcs or edges, which are the complementing objects in a network, typically represented by line segments or curves (directed or undirected) connecting the defining nodes.

Node rank A weight associated with each node in a digraph. Intuitively, it is based on the notion of 'being pointed at', directly or indirectly. The more this is the case, the higher the node rank. It is a measure of popularity, so to speak.

Outdegree For a node $v \in$ in a network $G = (V, E)$ the outdegree is the number of arcs that start in v . That is, the number of nodes $a \in V$ such that $(v, a) \in E$. Its dual concept is indegree.

Parallel arcs / edges Two arcs (or edges) are parallel if they have the same head and tail (or endpoints). In our approach we avoid graphs with parallel arcs / edges, so that two endpoints a, b define an arc (a, b) or an edge $\{a, b\}$ uniquely.

Path A path in a digraph $G = (V, E)$ from $a \in V$ to $b \in V$ is a function $\pi : \{1, \dots, k\} \rightarrow V$, with $k \geq 2$ such that $\pi(1) = a$, $\pi(k) = b$ and $(\pi(i), \pi(i+1)) \in E$ for each $i = 1, \dots, k-1$. $k-1$ is the path length. The path π connects nodes a to b . By definition, a node is also a path, of length 0. If $a = b$, the path is a cycle. a is called the start and b the finish of the path.

Point See: Node.

Quasimetric Let $d(\cdot, \cdot)$ be a quasimetric on a space X . It has the following properties:

$d(x, y) \geq 0$ (positivity), $d(x, y) = 0$ implies $x = y$ (positive definiteness),

$d(x, y) \leq d(x, z) + d(z, y)$ (triangle inequality). In contrast to a metric, it needs not to be symmetric, that is $d(x, y) = d(y, x)$ is not required to hold. ⁴⁴⁾

RAN See: Random Access Network.

Random Access Network A network that is too big to comprehend at once. It requires some kind of sampling procedure to collect relevant characteristics. For instance, one can randomly choose points in the network and explore it by following links to other nodes, using a certain strategy. This exploration has to stop at some point in time. The information gathered in this way about nodes and how they are linked is then to be used to estimate certain connectivity characteristics of the entire network.

Reachability set If v is a node in a digraph G , the set of nodes in G that can be reached from v is the reachability set. It is denoted by \bar{v} .

Reduction of a network See 'Simplification of a network'.

Reverse path Let π be a path in a graph $G = (V, E)$ from $a \in V$ to $b \in V$, defined as

$\pi : \{1, \dots, k\} \rightarrow V$, with $k \geq 2$ such that $\pi(1) = a$, $\pi(k) = b$ and $(\pi(i), \pi(i+1)) \in E$ for each $i = 1, \dots, k-1$. Then the reverse path π^{\leftarrow} is defined by $\pi^{\leftarrow}(t) = \pi(k-t)$ for $t = 1, \dots, k$.

Routing graph A routing graph⁴⁵⁾ is an acyclic digraph with one source and one sink. In [13] such structures were introduced in the context of questionnaires, in particular to describe their routing structure, which prescribes which question is to be asked next on the basis of the answers provided to the questions so far. Despite the name, they are used in the present paper in a neutral context. In the present paper this class of digraphs is denoted by Y

⁴⁴⁾ See [https://en.wikipedia.org/wiki/Metric_\(mathematics\)#Quasimetrics](https://en.wikipedia.org/wiki/Metric_(mathematics)#Quasimetrics).

⁴⁵⁾ This is strictly speaking a misnomer, but it is faithful to the naming of these structures in [13]. 'Routing digraph' would be more appropriate.

(epsilon).

Saturated pair of nodes A pair of nodes a, b in a digraph for which both (a, b) and (b, a) are arcs, or, equivalently, for which $\{a, b\}$ is an edge.

Self-dual A network G for which $G^{\leftarrow} = G$. A self-dual network is a graph.

Simplification of a network The goal of this process is to present the essence of a network. It either selects the nodes or arcs with the highest ranks (above a threshold $\delta > 0$, specified by the user) completed in such a way that the reachability properties of the original digraph are preserved.

Sink A node v in a digraph with only ingoing arcs. So for v holds: $\Delta^{in} > 0$ and $\Delta^{out} = 0$.

Source A node v in a digraph with only outgoing arcs. So for v holds: $\Delta^{in} = 0$ and $\Delta^{out} > 0$.

Start The node in a graph where a path begins.

Tail The tail of an arc (a, b) in a digraph is a . In a graphical representation the arc is typically depicted as an arrow or directed (curvy) line segment. The tail is the dot (representing an end node) where the arc starts.

Tour See: Cycle.

Transitive closure If $G = (V, E)$ is a digraph. G is transitive if for all nodes a, b, c in V it is the case that if (a, b) and (b, c) are in E then also (a, c) is in E . If G is not transitive we can make it transitive as follows: if a, b, c are nodes in V and $(a, b), (b, c) \in E$ but $(a, c) \notin E$ then add (a, c) to a new arc set $E^* \supseteq E$. In this way we create a new digraph $G^* = (V, E^*)$, of which G is a subdigraph, that is transitive. In fact, G^* is the smallest digraph of which G is a subdigraph that is transitive. G^* is called the transitive closure of G . If A is the adjacency matrix of G then A^* is used to denote the adjacency matrix of G^* . A^* can be computed from A using Warshall's algorithm (cf. [12]).⁴⁶⁾ We have: $(G^*)^* = G^*$ and likewise $(A^*)^* = A^*$. For more properties see 'transitive reduction'.

It is also possible to compute the transitive closure using transitive reduction, as is shown in [1]. In fact this paper shows that computing transitive closure or transitive reduction is of the same computational complexity.

Transitive reduction Let A be an adjacency matrix of some digraph G . The transitive reduction of G is the minimal digraph G^\downarrow with $(G^\downarrow)^* = G^*$. Likewise we define the transitive reduction of A , i.e. A^\downarrow , as the minimal adjacency matrix with the same transitive closure as A .

The following commutative properties hold for transitive closure (*) and transitive reduction ($^\downarrow$) in combination with transposition ('):

- $(A')^\downarrow = (A^\downarrow)'$,
- $(A')^* = (A^*)'$,

$$\begin{aligned} & - (A')^\downarrow = (A^\downarrow)', \\ & - (A')^* = (A^*)', \end{aligned}$$

For transitive closure (*) and transitive reduction ($^\downarrow$) combined the following properties hold:

$$\begin{aligned} & - (A^\downarrow)^* = A^*, \\ & - (A^*)^\downarrow = A^\downarrow, \end{aligned}$$

For iteration of these operators the following properties hold:

$$\begin{aligned} & - (A^*)^* = A^* \text{ (idempotency)}, \\ & - (A^\downarrow)^\downarrow = A^\downarrow \text{ (idempotency)}, \\ & - (A')' = A \text{ (involution)}, \end{aligned}$$

Note that transposition behaves differently from transitive closure and transitive reduction.

$$\begin{aligned} & - A \leq B \Rightarrow A^* \leq B^* \text{ (monotonicity)}, \\ & - A \leq B \Rightarrow A^\downarrow \leq B^\downarrow \text{ (monotonicity)}. \end{aligned}$$

A and B are adjacency matrices of the same order $n \times n$, say. The matrix inequalities used in the final two properties operate element-wise. Similar properties hold for the graphs

⁴⁶⁾ This is intuitively a nice algorithm. From a computational point of view, however, it is not so attractive. More efficient algorithms exist, such as described in [7] or [8].

corresponding to the adjacency matrices. Instead of G' we write G^{\leftarrow} .

The following condition holds (cf. [1]): $A_{ij}^{\downarrow} = 1$ iff $A_{ij} = 1$ and $(A \odot A^*)_{ij} = 0$, which can be used to compute the transitive reduction with the help of the transitive closure.

Triangular number A natural number of the form $\binom{n+1}{2} = \frac{n(n+1)}{2}$, for $n = 0, 1, 2, 3, \dots$. So 0, 1, 3, 6, 10, 15, 21, 28, 36, 45, 55, 66, 78, 91, ... is the beginning of a list of triangle numbers. It is a theorem in number theory that each natural number is the sum of three triangular numbers.

Underlying graph of a digraph If $G = (V, E)$ is a digraph then the underlying graph

$G^{un} = (V, E^{un})$ is the graph with $E^{un} = \{\{a, b\} \mid a, b \in V \text{ and } (a, b) \in E \text{ or } (b, a) \in E\}$.

Unsaturated pair of nodes A pair of nodes a, b in a digraph for which (a, b) or (b, a) is an arc but not both.

Vertex See: Node.

Colophon

Publisher

Statistics Netherlands
Henri Faasdreef 312, 2492 JP The Hague
www.cbs.nl

Prepress

Statistics Netherlands, Grafimedia

Design

Edenspiekermann

Information

Telephone +31 88 570 70 70, fax +31 70 337 59 94
Via contact form: www.cbs.nl/information

© Statistics Netherlands, The Hague/Heerlen/Bonaire 2018.
Reproduction is permitted, provided Statistics Netherlands is quoted as the source