How to Understand Connections Based on Big Data: From Crisp Granules (e.g., Cliques) to Flexible Granules

Ali Jalal-Kamali, M. Shahriar Hossain,
and Vladik Kreinovich
Department of Computer Science
University of Texas at El Paso
El Paso, TX 79968, USA
ajalalkamali@miners.utep.edu, mhossain@utep.edu,
vladik@utep.edu

Abstract

One of the main objectives of science and engineering is to predict the future state of the world – and to come up with actions which will lead to the most favorable outcome. To be able to do that, we need to have a quantitative model describing how the values of the desired quantities change – and for that, we need to know which factors influence this change. Usually, these factors are selected by using traditional statistical techniques, but with the current drastic increase in the amount of available data – known as the advent of *big data* – the traditional techniques are no longer feasible. A successful semi-heuristic method has been proposed to detect true connections in the presence of big data. However, this method has its limitations. The first limitation is that this method is heuristic – its main justifications are common sense and the fact that in several practical problems, this method was reasonably successful. The second limitation is that this heuristic method is based on using "crisp" granules (clusters), while in reality, the corresponding granules are flexible ("fuzzy"). In this paper, we explain how the known semi-heuristic method can be justified in statistical terms, and we also show how the ideas behind this justification enable us to improve the known method by taking granule flexibility into account.

1 Understanding Connections Based on Big Data – An Important Practical Problem

What are our main objectives? The role of science and engineering. We have preferences: we want tasty food, we want a comfortable environment,

we want to stay healthy, etc. In general, we have many objectives. We are making individual and collective decisions so as to satisfy these objectives; to be more precise, we select actions which maximize our degree of satisfaction in these objectives.

To be able to select appropriate actions, we need to be able to predict the consequence of different actions. Crudely speaking, this is what we usually understand by *science*: we know the current state of the world, we describe what actions we plan to perform, and we want to predict the future state of the world.

Once we can do that, we need to select a sequence of actions which will be the most beneficial; crudely speaking, this is what we usually understand by *engineering*. For example:

- Science predicts what happens to a rocket if we launch it in a certain direction.
- Based on these predictions, we can solve an engineering problem – find in what direction we must launch a rocket so that it will, e.g., reach the Moon.

While praising successes of science and engineering, we need to remember that these successes are based on understanding connections. In the last several centuries, science and engineering achieved many things – we have successfully overcome many diseases, we drastically increases the life expectancy, we reached the Moon. These successes are based on complex quantitative methods of modern science and engineering.

In spite of all these successes, in some areas – such as economics – we still do not have good predictive models. The reason is simple. In general, there are many factors which could potentially affect the desired values. In many physics problems, we have succeeded in pinpointing a few relevant factors – and showing that all other factors can be safely ignored. For example, the acceleration of a rocket is determined by the forces acting on this rocket – gravity and aerodynamic resistance. Once we know that the desired value depends on the few parameters, we can use experiments to find the exact quantitative form of this dependence.

In contrast, in economics, we cannot dismiss any of the factors. As a result, potentially, we have a function of very many variables. To describe such functions, we need a very large number of parameters – much more parameters than the number of data points.

In other words, to be able to build a successful quantitative model, we first need to understand with which quantities the desired quantity is connected – and with which it is not. In other words, understanding connections is an important pre-requisite for successes of science and engineering.

This importance can be also illustrated on examples from medicine. For some diseases – like cholera or malaria – originally many factors were considered: e.g., that malaria is caused by swampy air, etc. (not to count such weird hypothesis

as witchcraft and divine punishment for sins). When many possible factors were considered, no easy model of these illnesses existed, and no good cure was known. Once the scientists succeeded in determining the unique factor determining each of these diseases – the corresponding bacteria – this opened the possibility for developing successful medicine.

In contrast, for many types of cancer, we still have too many possible factors – viruses, pollution, stress, genetic mutations, etc. As a result, for these cancers, we do not have a good cure.

How connections are determined now. Traditionally, connections are determined by statistical methods; see, e.g., [17]. We observe some relation between the two processes: e.g., we observe that patients getting a certain medicine tend to recover faster, that the two DNA samples match, etc. This may be a random coincidence. So, in order to check whether the observed relation is statistically significant, we compute the probability p that this observed relation can happen for two unrelated processes. If this probability is smaller than a certain threshold p_0 (called a *p-value*), we conclude that there is a statistically significant connection; if the probability p is larger than p_0, then we cannot make this conclusion. Usually, practitioners take $p_0 = 0.05$ or, sometimes, $p_0 = 0.01$.

The connecting building task has been used in a variety of contexts: entity networks [5, 8], image collections [6], cellular networks [2, 7], social networks [4], and document collections [8, 9, 11]. All these research efforts focus on finding connections between objects that are apparently disjoint. A solution to the connection building task generally depends on the commonality between some intermediaries to reach the target object. Swanson refers to the notion of neighboring commonality as complementary but disjoint (CBD) structures [19], whereby two arguments may exist separately that when considered together lead to new insights, but the objects exhibiting these two arguments are unaware of each other. The proposed solution to connection building in this chapter leverages a similar principle.

Enter big data. Modern technology has led to a drastic increase in the amount of possible observations – and in the number of parameters related to each observation that we can measure and record. In principle, with devices like Google Glass, we can record everything that we see – and more generally, everything that is happening in the world. The resulting amount of data is so huge that not only a single researcher cannot review all this data – even the existing computer algorithms cannot process all this data. This phenomenon is known as *big data*; see, e.g., [3, 14, 18].

Traditional methods do not work well for big data: formulation of the problem. In the traditional statistical approach, we made few observations, so observed connections were relatively rare. In the big data, we record so many parameters that everything appears connected.

For example, traditionally, when we had to rely on human witnesses, the fact that the victim and the suspect were seen together (or could be indirectly connected by a convincing chain of such seen-together events) was a strong argument for the suspect's guilt.

Nowadays, with numerous security cameras recording many moments of our lives – from walking the streets to attending football games on a stadium – there are so many pairs of people who happen to be together at the same time in the same place simply by accident, that it is extremely difficult to separate such random encounters from true connections.

So, for big data, we need new methods to find out which joint appearances correspond to true connections and which do not.

What we do in this paper. Our main objective is to study how to detect true connections based on the big data.

- We start with describing the semi-heuristic methods which have been proposed for solving this problem, as described, e.g., in [8, 9].
- Then, we describe the limitations of the existing methods. Some of these limitations are related to the fact that the existing methods are based on using *crisp* granules (clusters), while real-life clusters are flexible ("fuzzy").
- Finally, we describe how these limitations can be overcome – in particular, how we can use flexible granules (clusters) to understand true connections based on the big data.

Two case studies. The existing method has been tested on two big-data situations.

First case study: intelligent analysis. The paper [8] deals with *intelligence analysis*. Specifically, we have a huge database of documents. Based on these documents, we need to detect possible true connections between adversaries. The existing documents provide only possible relation – e.g., if two names appear in the same document, this may be an indication that the two prsons are connected. The document may combine the name of the person with the name of the hotel where this person stayed at a certain night – and if another document shows another person staying at the same hotel, this may be an indication of a true connection between them.

The mere fact that the two names appeared in the same document does not necessarily mean that these names are actually connected – for example, one of the authors (V.K.) graduated from the same St. Petersburg University in the same year as the Russian President Vladimir Putin – but he never met Putin in person, so there is clearly no true direct connection. However, if there are many such connecting documents, it increases the probability that the two names are actually connected – and at some point, we should be able to conclude, with a reasonable confidence, that there is a true connection.

Second case study: biomedical publications. The paper [9] deals with biomedical publications. The field of biomedical research has become so specialized that is no longer easy for a human specialist to trace all relevant papers – or even to find all relevant papers. Finding such relevant papers is extremely important because in many cases, by combining the ideas presented in related papers, we can come up with a synergistic effect of an even better cure. Here also, we have a huge database of documents – this time, of papers. Based on these documents, we want to find true connections between the papers.

Similar to the intelligence analysis case, we can come up with criteria of when two papers may be connected: e.g., if they share keywords or share references, etc. Based on this information, it is necessary to decide when the two papers are actually connected and when the seeming connection is accidental.

2 General Case: How to Describe Available Information

General situation. In general:

- We have a large set of *entities*: persons, locations, organizations, dates, etc. for the intelligence database, biomedical articles, etc.
- We also have a huge database of *features*: documents for the intelligence database, biomedical terms for the publications database, etc. – which enable us to relate some entities.

Based on this information, we have to decide which entities are actually connected and which are not.

Description of the available information. In general:

- we have entities e,
- we have features f, and we have *associations* between entities e and features f: e.g.,
  - a name e is mentioned in the document f,
  - a term f appears in a paper e, etc.

For some e and f, we may have several associations – e.g., the name e is mentioned several times in the document f, or the term f appears several times in the paper e.

Some other notations are as follows:

- we will denote the set of all entities by $\mathcal{E}$;
- we will denote the set of all features by $\mathcal{F}$; and

- for each e and f, we will denote the to number of associations between e and f by $n_{e,f}$.

The total number of entities is equal to $|\mathcal{E}|$ and the total number of features is equal to $|\mathcal{F}|$. It is also useful to describe:

- for each feature f, the set $e(f) \stackrel{\text{def}}{=} \{e \in \mathcal{E} : n_{e,f} > 0\}$ of all entities associated with the feature f, and
- for each entity e, the set $f(e) \stackrel{\text{def}}{=} \{f \in \mathcal{F} : n_{e,f} > 0\}$ of all features associated with the entity e.

First step of the usual document analysis: describing the weight $V(e, f)$ of a feature f for the entity e. Based on information about associations between entities and features, we can decide which features are more important for a given entity and which are less important.

Intuitively, the larger the number of associations between the entity and the feature, the more confident we are that this association is meaningful – for example, one mention of a name in a document may be accidental, but if the same name appears several times, we become confident that this is a connection between the name and the document.

Similarly, the fewer entities are associated with the feature, the more confident we are that this association is meaningful. For example, when a phone book has a listing for a person, this does not mean anything – since the phone book, by definition, has listings for all the persons. In contrast, a hotel bill which lists only one person is an indication of a strong connection.

Let us describe this qualitative idea in numerical terms. In situations like this, when we have several entities associated with a feature, a reasonable idea is to use the amount of information, i.e., the number of binary ("yes"-"no") questions (bits) which are needed to find the desired entity.

In general, if we know that an unknown object belongs to the set of N elements, then we can divide this set into two halves and, by asking a binary question, find out which half the desired object belongs to. In other words, each binary question divides the amount in half. Thus, q binary questions divide the original number of possible alternatives by a factor of 2^q, to $N \cdot 2^{-q}$. When we reach $N \cdot 2^{-q} = 1$, we have pinpointed the desired alternative. Thus, for the case of N alternatives, the corresponding information (number of binary questions) can be determined from the equation $N \cdot 2^{-q} = 1$, and is, thus, equal to $q = \log_2(N)$.

Originally, we have $|\mathcal{E}|$ entities; the corresponding amount of information is equal to $\log_2(|\mathcal{E}|)$ bits. Once we know that an entity is associated with the feature f, we thus limit ourselves to $|e(f)|$ entities; in this case, the corresponding amount of information is equal to $\log_2(|e(f)|)$ bits. Thus, the very fact that the entity is associated with the feature f enables us to reduce the number of questions by the value

$$\log_2(|\mathcal{E}|) - \log_2(|e(f)|) = \log_2\left(\frac{|\mathcal{E}|}{|e(f)|}\right). \tag{1}$$

Similarly, the effect of multiple associations can be describe by counting how many additional binary questions we can afford and still keep an association with the desired entity. We start with $n_{e,f}$ mentions. Each binary question decreases this number by half; q questions decrease this amount to $n_{e,f} \cdot 2^{-q}$. As long as this remaining number is ≥ 1, we still have some association. The largest number q for which we can still get as association can thus be determined from the condition that $n_{e,f} \cdot 2^{-q} = 1$, and is, thus, equal to $q = \log_2(n_{e,f})$. To take into account the fact that we deal with *additional* questions, we usually add 1, ending up with $1 + \log_2(n_{e,f})$.

The overall importance of the feature f in entity e can be obtained if we multiply $\log_2\left(\frac{|\mathcal{E}|}{|e(f)|}\right)$ by the importance factor $1 + \log_2(n_{e,f})$, resulting in the product

$$I(e,f) \stackrel{\text{def}}{=} (1 + \log_2(n_{e,f})) \cdot \log_2\left(\frac{|\mathcal{E}|}{|e(f)|}\right). \tag{2}$$

This formula is one of the versions of *term frequency – inverse document frequency (tf-idf)* modeling; see, e.g., [12, 15].

For each entity e, we thus get the importance $I(e,f)$ of different features f. These values of importance are usually normalized, i.e., multiplied by a constant so that the mean square importance is equal to 1 (this is known as *cosine normalization*). As a result, we get the formula

$$V(e,f) = \frac{(1 + \log_2(n_{e,f})) \cdot \log_2\left(\frac{|\mathcal{E}|}{|e(f)|}\right)}{\sqrt{\sum\limits_{j \in f(e)} \left((1 + \log_2(n_{e,j})) \cdot \log_2\left(\frac{|\mathcal{E}|}{|e(j)|}\right)\right)^2}}. \tag{3}$$

From weights to distance between entities. For each entity e, we have the weights $V(e,f)$ corresponding to different features f. Thus, as a measure of closeness between two entities e_1 and e_2, we can take the distance between the corresponding vectors $(V(e,f_1), V(e,f_2), \ldots)$.

In the usual Euclidean distance $d(a,b) = \sqrt{(a_1 - b_1)^2 + \ldots}$, we add the squares of the differences. Since each value $V(e,f)$ represents the number of bits, it makes more sense to take the actual differences – since each difference reflects the number of additional questions. Thus, we take

$$d(e_1,e_2) \stackrel{\text{def}}{=} \sum_{f \in \mathcal{F}} |V(e_1,f) - V(e_2,f)|. \tag{4}$$

This distance depends on the number of features: e.g., if, in addition to the documents, we store their copies, the distance increases by a factor of two. To avoid this dependence, the distance $d(e_1,e_2)$ is usually normalized to the interval $[0,1]$ – by dividing by the largest possible value of this distance. When we only know the upper bound $\overline{a}$ and $\overline{b}$ on the two non-negative numbers a and b, then the largest possible value of the difference is equal to $\max(\overline{a},\overline{b})$. Indeed:

- if $\overline{a} \le \overline{b}$, then $|\overline{a} - \overline{b}| = \overline{b} - \overline{a} \le \overline{b}$ and thus, $|\overline{a} - \overline{b}| \le \max(\overline{a}, \overline{b})$;
- similarly, if $\overline{b} \le \overline{a}$, then $|\overline{a} - \overline{b}| = \overline{a} - \overline{b} \le \overline{a}$ and thus, $|\overline{a} - \overline{b}| \le \max(\overline{a}, \overline{b})$.

Thus, in both cases, we have $|\overline{a} - \overline{b}| \le \max(\overline{a}, \overline{b})$.

The bound $\max(\overline{a}, \overline{b})$ can be attained:

- if $\overline{a} \le \overline{b}$, then it is attained for $a = 0$ and $b = \overline{b}$;
- if $\overline{b} \le \overline{a}$, then it is attained for $a = \overline{a}$ and $b = 0$.

So, for each f, the maximum possible value of the difference

$$|V(e_1, f) - V(e_2, f)| \tag{5}$$

can be estimated as $\max(V(e_1, f), V(e_2, f))$. Therefore, the largest possible value of the sum $\sum\limits_{f \in \mathcal{F}} |V(e_1, f) - V(e_2, f)|$ can be estimated as

$$\sum_{f \in \mathcal{F}} \max(V(e_1, f), V(e_2, f)). \tag{6}$$

By dividing $d(e_1, e_2)$ by this bound, we get the formula

$$D(e_1, e_2) \stackrel{\text{def}}{=} \frac{\sum\limits_{f \in \mathcal{F}} |V(e_1, f) - V(e_2, f)|}{\sum\limits_{f \in \mathcal{F}} \max(V(e_1, f), V(e_2, f))}. \tag{7}$$

This formula is known as the *Soergel distance*.

Comment. It is worth mentioning that the Soergel distance is a *metric*, in the sense that it is symmetric $D(e_1, e_2) = D(e_2, e_1)$ and satisfies the triangle inequality $D(e_1, e_3) \le D(e_1, e_2) + D(e_2, e_3)$.

Resulting description. As a result of the above preliminary analysis, we represent the given information as a *weighted graph*:

- in this graph, nodes (vertices) represent entities, i.e., the set of all the nodes is the set of all the entities $\mathcal{E}$;
- for each two entities (nodes) e_1 and e_2, we know the distance $D(e_1, e_2)$; in graph terms, this distance can be represented as the weight of the edge between e_1 and e_2.

3 A Known Semi-Heuristic Method for Detecting True Connections Based on Big Data: A Brief Description

Direct and indirect connections. In some cases, we have a *direct* connection between the two objects – e.g., when two (or more) terrorist suspects meet together to plot future attacks.

Sometimes, the two suspects never (or rarely) meet in person, but they are plotting together via intermediaries – in this case, we have an *indirect* connection. In this case, we have a direct connection between the first suspect and the intermediary, and we have a direct connection between the intermediary and the second suspect – and we can use these two direct connections to make a conclusion that the two suspects are indirectly connected.

Detecting indirect connections is based on detecting direct ones. Because of this:

- we will first describe how direct connections are detected, and then
- we will describe how detected direct connections are combined to detect indirect connections.

From the original weighted graph to a simpler (non-weighted) one. In general, for every two nodes e_1 and e_2, we know the distance $D(e_1, e_2)$. The larger the distance, the less probable it is that the corresponding entities are actually connected.

- When the distance is very small, there is a high probability that the entities are connected. So, it is possible to conclude that the entities are connected if we need to make a definite decision about the connectivity.
- When the distance is close to 1, this probability becomes very small. So, we can conclude that the entities are *not* connected when a boolean decision about the connectivity is essential.

As we increase the distance from 0 to 1, there should be a point θ at which our decision changes from "connected" to "not connected". Once this threshold value θ is determined, we can then simplify the original weighted graph into a simplified non-weighted graph $\mathcal{G}$. In this simplified graph, the nodes (entities) e_1 and e_2 are connected by an edge if and only if $D(e_1, e_2) \le \theta$.

Detecting direct connections: idea. As we have mentioned, if we have an edge between two entities e_1 and e_2, it is probable that there is an actual connection, but we cannot conclude this with confidence – since the edge may be caused by coincidence. If we also have a third entity e_3, and every two of the three entities e_1, e_2, and e_3 have an edge, then the probability that all the three edges are accidental is much smaller. As a result, our confidence that e_1 and e_2

are connected increases. Similarly, if there is a fourth entity e_4 and every two out of four entities have an edge, the probability increases.

In general, we may have ℓ entities e_1, e_2, ..., e_ℓ for which every two entities have an edge. Such a set of nodes is known as an *ℓ-clique*. The larger ℓ, the higher our degree of confidence that e_1 and e_2 are actually connected. Thus, there is a threshold value k starting from which this confidence becomes so large that we can confidently conclude that e_1 and e_2 are actually connected.

This idea leads to the following algorithm for detecting direct connections.

Detecting direct connections: resulting method. We select a distance threshold $\theta \in (0, 1)$ and an integer k. We claim that two nodes e_1 and e_2 are actually directly connected in the graph $\mathcal{G}$ is there is a k-clique containing both e_1 and e_2.

In other words, we claim that the entities e_1 and e_2 are directly connected if there exist edges $e_3, \ldots, e_k$ such that $D(e_i, e_j) \le \theta$ for all $i, j \in \{1, 2, \ldots, \ell\}$.

Detecting a general connection: resulting method. A natural idea is to claim that the nodes e_1 and e_2 are actually connected if there is a chain of nodes $c_1 = e_1$, c_2, ..., c_t, $c_{t+1} = e_2$ such that for every i, the nodes c_i and c_{i+1} are actually directly connected. This is equivalent to saying that in the graph $\mathcal{G}$, there is a chain of k-cliques G_1, G_2, ..., G_t which connect e_1 and e_2 in the sense that:

- the first clique G_1 contains the node e_1,
- every two neighboring cliques have at least one common node, that is, $G_i \cap G_{i+1} \neq \emptyset$, and
- the last clique G_t contains the node e_2.

How to select parameters of the method. The method described above used two parameters: θ and k. The values θ and k need to be determined empirically – e.g., by using examples where true connections are known and finding the values θ and k for which this method reproduces these known true connections as accurately as possible.

For example, for intelligence analysis [8], the values $\theta = 0.93$ and $k = 6$ lead to a good outcome; see Fig. 1.

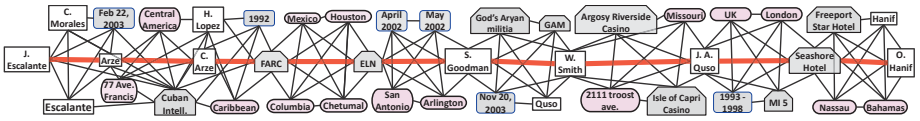

Figure 1: Connection between the two suspects (reproduced from [8])

How to implement the above method: need for approximate techniques. At first glance, the above methods can be directly translated into algorithms.

To find out whether two nodes e_1 and e_2 are part of a k-clique, i.e., whether there are $k-2$ nodes e_3, ..., e_k which form a clique, we can try all possible combinations of $k-2$ nodes. If we denote, by N, the total number of nodes in the graph G, i.e., the total number of entities, then this would require $\binom{N}{k-2} \approx N^{k-2}$ steps.

The problem with this idea is that we are dealing with big data, where the number N of entities is already huge – for example, the US no-fly list containing possible suspects has about a million people in it. For the value $k=6$ corresponding to intelligent analysis, we will need N^4 computation steps. For $N \approx 10^6$, this leads to $N^4 \approx 10^{24}$ computation steps – way beyond the capabilities of modern computers.

The situation is even worse in the general case, when we look for possible indirect connections. In this case, to check whether the given nodes e_1 and e_2 are connected, a natural idea is to try all possible k-cliques containing e_1, i.e., for all possible tuples of $k-1$ nodes $e_2, \ldots, e_k$ which, together with the given node e_1, form a k-clique. We need $\binom{N}{k-1} \approx N^{k-1}$ steps, which, for $k=6$ and $N \approx 10^6$, requires 10^{30} computational steps.

How the above method is algorithmically implemented: idea. First, the papers [8, 9] use the concept lattice algorithms to come up, for each entity e, with a list of the closest ones. Then, for each node e and for each m, we can find a *m-neighborhood* of e – i.e., the set consisting of m closest nodes.

Suppose now that we need to check whether the two nodes e_1 and e_2 are connected by a chain of k-cliques. According to the above method, we need to first find a k-clique containing the node e_1. Since, as we have mentioned, there are too many possible sets of $k-1$ nodes, instead of looking for all possible nodes, we only look for k-cliques among the m nearest nodes; thus, the value m must be selected in such a way that the resulting amount of possible combinations $\binom{m}{k-1}$ does not exceed the computational ability of the available computer.

In this manner, we find one or more k-cliques containing the node e_1. According to the method, all the nodes in all these k-cliques are thus assumed to be actually directly connected to e_1. One of these nodes should start the next k-clique. How can we select, out of these nodes, the node c_2 which is the most promising to start the new k-clique?

In order to select this node c_2, let us recall that when for some k, we claim that the existence of a k-clique confirms the existence of a true connection, in reality, there is still a probability that the observed "connection" was accidental – this probability is very small but still positive. We then conclude that two nodes related by a chain of k-cliques are actually connected. For this conclusion

to be true, *all* the k-cliques must be actually connected. If only one the k-cliques is accidental – the whole conclusion fails. Here, the probability that the conclusion is false is equal to the probability that either the first k-clique is accidental, or that the second k-clique is accidental, etc. The longer the chain, the higher this probability. Thus, it is desirable to construct chains of k-cliques which are as short as possible.

Intuitively, the larger the distance between the two nodes, the longer the chains which connect them. To be more precise, we need to take into account that different links correspond to different distance. What we thus really want to minimize is the overall distance, not just the overall number of steps. If we select a node e' as the nest step c_2, then the overall chain-following distance between e_1 and e_2 can be estimated as the sum of the distance from e to e' and from e' to e_2, i.e., as $D(e_1, e') + D(e', e_2)$. We therefore select a node for which this sum is the smallest possible.

A similar greedy-algorithm idea can be used on the next step, etc. As a result, we arrive at the following algorithm.

How the above method is algorithmically implemented: details. We want to check whether the given nodes e_1 and e_2 are actually connected – and if so, we want to design a chain of events $c_1 = e_1$, c_2, ..., c_t, and $c_{t+1} = e_2$ in which each c_i id directly connected to c_{i+1}.

In the algorithm, we start with $c_1 = e_1$, and we select the nodes c_2, c_3, ..., c_t one by one. For every i, once the node c_i is selected, we find m nodes which are the closest to c_i. Out of these m nodes, we test all possible subsets of $k-1$ nodes, and for each subset, we check whether this subset, together with c_i, forms a k-clique. (To be more precise, all m elements have an edge with c_i – otherwise why consider them; thus, it is sufficient to check that the selected $k-1$ nodes form a $(k-1)$-clique.) For each subset which leads to a k-clique, we record all its nodes.

- If one of the recorded nodes is e_2, we are done – we have found a chain of k-cliques between e_1 and e_2.
- If none of the recorded nodes coincides with e_2, then out of all recorded nodes e, we select, as the next node c_{i+1} in the chain, the recorded node for which the sum $D(c_i, e) + D(e, e_2)$ is the smallest possible.

If, after a certain number T of steps, we do not teach e_2, we conclude that e_1 and e_2 are not actually connected. (This maximum number of steps T needs to be determined empirically.)

Empirical success. In both applications – to the intelligence analysis and to the biomedical publications – the above method has led to good results, i.e., to the concluded connections for which the high percentage were confirmed by experts as meaningful.

An auxiliary comment: how to gauge our confidence in the results of the method. In general, as we have mentioned, the larger the clique size, the larger our confidence that the nodes are actually connected.

Thus, once we have found that the given nodes e_1 and e_2 are connected by a chain of k-cliques – and thus, we have concluded that e_1 and e_2 are actually connected – we can gauge our degree of confidence in this conclusion by checking whether e_1 and e_2 can be connected by a chain of $(k+1)$-cliques, $(k+2)$-cliques, etc. In this manner, we find the largest click size ℓ for which e_1 and e_2 are connected by a chain of ℓ-cliques. The larger this size ℓ, the more confident we are that e_1 and e_2 are actually connected.

4 Limitations of the Semi-Heuristic Approach

First limitation: this method is semi-heuristic. The first limitation is that this method is semi-heuristic: its main justifications are common sense and the fact that in several practical problems, this method was reasonably successful. It is desirable to provide a more formal justification for this method – ideally, a justification which would allow us not only to make conclusions, but also to provide a reasonable estimate of our degree of certainty in this conclusion.

Second limitation: need for flexible granules. The second limitation is that the above semi-heuristic method depends on "crisp" granules (clusters) – namely, k-cliques. As a result:

- If, for some nodes e_1 and e_2, there is a k-cliques which contains both e_1 and e_2, then we conclude that e_1 and e_2 are actually directly connected.
- If no such k-clique exists, then we conclude that e_1 and e_2 are not actually directly connected.

From the intuitive viewpoint, this conclusion is too crisp. Intuitively, if we have a subgraphs G which is "almost" a k-clique – i.e., a k-clique with one (or even two) edges missing, it may not affect the conclusion. For example, for $k = 6$, being a k-clique means that we have $\dfrac{k \cdot k - 1}{2} = \dfrac{6 \cdot 5}{2} = 15$ edges between $k = 6$ nodes; what if we have only 14? There should be a threshold, but this threshold does not necessary mean the threshold between a full k-clique and a graph in which one edge is missing – maybe it is OK if two or more edges are missing?

Right now, the corresponding numerical characteristic – the size k of the largest k-clique connecting two nodes – is too crisp:

- This characteristic decreases rapidly (to $k - 1$) when we delete a single edge from the k-clique.
- And then, when we delete one more edge between some other nodes, this characteristic does not change at all.

It is desirable to generalize a crisp notion of an integer clique size k into a more flexible notion of the fractional-valued "degree" of clique-ness (i.e., the degree of being a granule); see, e.g., [10, 13, 20].

Similarly, for a general connectedness:

- If, for some nodes e_1 and e_2, there is a relating chain of k-cliques, then we conclude that e_1 and e_2 are actually connected.
- If no such chain exists, then we conclude that e_1 and e_2 are not actually connected.

Intuitively, if we have a sequence of subgraphs G_1, G_2, ..., in which one of the graphs is "almost" a k-clique, it may not affect the conclusion.

The above degree of certainty – the size k of the cliques – is also too crisp:

- If e_1 and e_2 can be related by a chain of k-cliques but cannot be related by a chain of $(k+1)$-cliques, then our degree of confidence corresponds to k.
- If e_1 and e_2 can be related by a chain of $(k+1)$-cliques, then our degree of confidence corresponds to the level $k+1$ (or higher).

What about the situation when we have a chain of graphs G_1, G_2, ..., G_t in which all graphs except one are $(k+1)$-cliques but the remaining one is still a k-clique? According to the above method, we assign, to this case, the degree of certainty k – the same as if all the graphs are k-cliques. However, intuitively, we are almost in the case of $(k+1)$-cliques, so to this "almost $k+1$" case, we should be able to assign the degree of confidence which is closer to $k+1$.

We should also assign different degree of certainty depending on how long is the chain of k-cliques. As we have mentioned, the longer the chain, the less confident we are that this chain implies the actual connection. We used this intuitive idea in designing the algorithm, but this idea is not reflected in how we estimate our degree of confidence – whether we have a chain of length 1 or a chain of the maximally allowed length T, we assign the same degree of confidence k to the conclusion that the corresponding nodes e_1 and e_2 are actually connected. It is desirable to assign the degree of confidence in such a way that longer chains would indeed lead to a smaller degree of confidence.

What we plan to do. We provide an uncertainty-based theoretical statistical framework which enables us, first, to justify the empirical clique approach and, second, to come up with formulas describing to what degree a given subgraph is a granule.

5 Analysis of the Problem and the Resulting Ideas and Formulas

Detecting direct connections based on a graph: analysis of the problem. Let us start with the first part of the problem – detecting direct connec-

tions. We will first analyze it in its simplified form – when we ignore the actual distances between the nodes and we only take into account whether the corresponding distance is below the threshold θ or not. In other words, we would like to detect direct connectedness based on a graph G.

As we have mentioned, the fact that there is an edge does not necessarily mean that entities are actually connected; there is a probability r that the edge is accidental. This probability r can be obtained, e.g., by analyzing the part of the graph for which we already know which entities are actually connected and which are not. If in this part of the graph, out of E edges, E_a of them correspond to actual connections, then we can estimate r as the ratio $\dfrac{E_a}{E}$.

We would like to estimate the probability that the given graph G – in which some entities are linked by an edge and some are not – describes actually connected entities. Let us pick any entity e in this graph. If we already know that all the other entities from G (i.e., the set $G - \{e\}$) are actually connected, then:

- for e to be actually connected to *all* these entities $e' \in G - \{e\}$,
- it is sufficient to show that e is directly connected to *one* of the entities $e' \in G - \{e\}$.

Indeed, if e is actually connected to some $e' \in G - \{e\}$, then, since e' is connected to every other entity from $G - \{e\}$, this would imply that e is actually connected with all the entities from $G - \{e\}$ (and thus, that all the entities from G are indeed connected to each other).

Since at least one actual connection from e to $G - \{e\}$ makes e connected to all other entities from $G - \{e\}$, the only possibility for e to be *not* actually connected to $G - \{e\}$ is when *all* edges between e and elements of $G - \{e\}$ are accidental. In graph theory, the number of edges between a node e and all other nodes is known as the *degree* of a node – and it is denoted by $\deg(e)$. In these terms, e is not connected if all $\deg(e)$ edges are accidental.

The probability that each edge is accidental is equal to r. Since we have no reason to make any conclusion about the dependence between different edges, we will assume that different edges correspond to independent events. If we have two independent or more events, then the probability of them happening together is equal to the product of the corresponding probabilities: e.g., the probability that the coin falls heads three times in a row is the product of the three probabilities corresponding to the three coin tosses, i.e., to $\dfrac{1}{2} \cdot \dfrac{1}{2} \cdot \dfrac{1}{2} = \dfrac{1}{8}$. Thus, under the independence assumption, the probability that all $\deg(e)$ edges are accidental is equal to the product of $\deg(e)$ probabilities each of which is equal to r – i.e., to $r^{\deg(e)}$. As a result, the probability that e *is* actually connected to $G - \{e\}$ is equal to $1 - r^{\deg(e)}$.

All the entities from a graph $G = \{e, e', e'', \ldots\}$ are actually connected if each of these entities is connected to all others, i.e., if the entity e is connected to all the other entities, and the entity e' is connected to all the other entities, and the entity e'' is connected to all the other entities, etc.

- We already know the probability that the entity e is actually connected to all other entities from the graph G: this probability is equal to

$$1 - r^{\deg(e)}; \tag{8}$$

- similarly, we know the probability that the entity e' is actually connected to all other entities from the graph G: this probability is equal to

$$1 - r^{\deg(e')}; \tag{9}$$

- we know the probability that the entity e'' is actually connected to all other entities from the graph G: this probability is equal to

$$1 - r^{\deg(e'')}; \tag{10}$$

- and so forth.

It is also reasonable to assume that the corresponding events are independent. Thus, we arrive at the following conclusion.

Detecting direct connections based on a graph: the resulting formula. For each graph G, the probability $P(G)$ that all entities from the graph are actually connected is equal to the product

$$P(G) = \prod_{e \in G} \left(1 - r^{\deg(e)}\right). \tag{11}$$

Alternatively, we can describe the probability $R(G) = 1 - P(G)$ that at least some of the entities from G are *not* connected. This probability is equal to

$$R(G) = 1 - \prod_{e \in G} \left(1 - r^{\deg(e)}\right). \tag{12}$$

As usual in statistical methods, we conclude that all the entities from the graph G are actually connected if this product is greater than or equal to a certain threshold P_0:

$$P(G) = \prod_{e \in G} \left(1 - r^{\deg(e)}\right) \geq P_0. \tag{13}$$

Alternatively, this condition can be described as $R(G) \leq p_0$, where $p_0 \stackrel{\text{def}}{=} 1 - P_0$.

Towards a simplified approximate versions of the formula (13)**.** Usually, the probability r is reasonably small, and for each node e, the number of edges $\deg(e)$ is reasonably large; thus, the probability $r^{\deg(e)}$ is small. In this case, we can expand the expression $\prod\limits_{e \in G} \left(1 - r^{\deg(e)}\right)$ in Taylor series in terms of these small quantities $r^{\deg(e)}$, and keep only linear terms in this expansion.

For two variables, we have

$$(1-a)\cdot(1-b) = 1-a-b+a\cdot b \approx 1-(a+b). \tag{14}$$

For three or more variables, we similarly have

$$(1-a)\cdot\ldots\cdot(1-b) \approx 1-(a+\ldots+b). \tag{15}$$

Thus, we arrive at the following approximate formula.

The resulting simplified approximate versions of the formula (13)**.** For every graph G, the probability $R(G)$ is approximately equal to

$$R(G) \approx \sum_{e\in G} r^{\deg(e)}. \tag{16}$$

Correspondingly, for $P(G) = 1 - R(G)$, we have

$$P(G) \approx 1 - \sum_{e\in G} r^{\deg(e)}. \tag{17}$$

Particular case of a k-clique. In the particular case when the graph G is a k-clique, this graph has k nodes for each of which $\deg(e) = k-1$. In this case, the formulas (13) and (14) takes the form

$$P(G) = \left(1-r^{k-1}\right)^k; \quad R(G) = 1-\left(1-r^{k-1}\right)^k. \tag{18}$$

The simplified approximate formulas (16) and (17) take the form

$$P(G) \approx k\cdot r^{k-1}; \quad R(G) \approx 1-k\cdot r^{k-1}. \tag{19}$$

Resulting natural definition of a degree of clique-ness. Based on the above formulas (13) and (18), we can define, for each graph, its "degree of clique-ness" as a real number k for which

$$P(G) \stackrel{\text{def}}{=} \prod_{e\in G}\left(1-r^{\deg(e)}\right) = \left(1-r^{k-1}\right)^k. \tag{20}$$

Comment. If we use the simplified approximate expressions for $P(G)$, the above equation for the degree of clique-ness k gets a simplified form:

$$\sum_{e\in G} r^{\deg(e)} = k\cdot r^{k-1}. \tag{21}$$

Example. For $p = 0.1$, for a 6-clique C_6, with $k = 6$, we have $R(C_6) = 6 \cdot 10^{-5} = 0.00006$. For a 5-clique C_5, we have $R(C_5) = 5 \cdot 10^{-4} = 0.0004$.

If we delete an edge that links two nodes of the 6-clique, then in the resulting graph G, we have two nodes e with $\deg(e) = 4$ and four remaining nodes with $\deg(e) = 5$. Thus, for this graph G, we have $R(G) = 2 \cdot 10^{-4} + 4 \cdot 10^{-5} = 0.00024$.

While this value is larger than the value $R(C_6)$ corresponding to a 6-clique, it is smaller than the value $R(C_5)$ corresponding to a 5-clique: $R(C_6) < R(G) < R(C_5)$. Thus, for the graph G, the above-defined degree of clique-ness is in between 5 and 6 – exactly as wanted it to be.

We thus get a flexible degree of confidence. In contrast to the traditional case, where our degree of confidence was described by a not-very-flexible integer k, now we are allowing non-integer values as well.

- Thus, e.g., if we delete one edge in a large clique, this leads to a minor change in $P(G)$ and thus, to a minor change in k. In contrast, for integers, this was a significant decrease from k to $k - 1$.
- Similarly, if we delete the second edge, we get a new small decreases. In contrast, for integers, we had no change.

If we use the simplified approximate formula, we get an explicit formula for the degree of clique-ness. The above equation for the degree of clique-ness k is similar to the equation that describes Lambert's W-function $W(z)$ (see, e.g., [16]): namely, $W(z)$ is defined as a value w for which $z = w \cdot e^w$.

This formula is similar to the formula that defines k, but it has two differences:

- first, in the formula that defines the W-function, we raise to the power w, while here, we raise r to the power $k - 1$;
- second, in the formula that defines the W-function, we raise e to some power, while here we raise p to some power.

To reduce the above equation to this form, let us transform our formula so as to eliminate these two differences.

First, let us reduce raising to the power $k - 1$ to raising to the power k. For that, we can use the known relation $r^{k-1} = \dfrac{r^k}{r}$. Substituting this expression into the equation that defines k, we get $R(G) = k \cdot \dfrac{r^k}{r}$, or, equivalently, $k \cdot r^k = r \cdot R(G)$.

To reduce raising r to some power to raising e to some point, we take into account that, by definition of the natural logarithm, the value r can be described as $e^{\ln(r)}$. Thus, $r^k = \left(e^{\ln(r)}\right)^k = e^{k \cdot \ln(r)}$. Hence, our equation takes the form $k \cdot e^{k \cdot \ln(r)} = R(G) \cdot r$. Here, e is raised to the power $w \stackrel{\text{def}}{=} k \cdot \ln(r)$, i.e., we have $r^k = e^w$. We can explicitly describe k in terms of w, as $k = \dfrac{w}{\ln(r)}$. Substituting

the above expressions for r^k and k in terms of w into the equation $k \cdot r^k = r \cdot R(G)$, we conclude that $\dfrac{w}{\ln(r)} \cdot e^w = R(G) \cdot r$, i.e., that $w \cdot e^w = R(G) \cdot r \cdot \ln(r)$. Thus, by definition of the W-function, we have $w = W(R(G) \cdot r \cdot \ln(r))$, and hence, for the desired degree of clique-ness $k = \dfrac{w}{\ln(r)}$, we get an explicit formula

$$k = \frac{1}{\ln(r)} \cdot W(R(G) \cdot r \cdot \ln(r)). \tag{22}$$

What if we have a chain of subgraphs? In general, we have a *chain* of graphs $G_1, \ldots, G_t$ linking two entities e_1 and e_2. To be able to conclude that e_1 and e_2 are actually connected, we need to be able to conclude:

- that the first graph G_1 corresponds to the actual connection,
- that the second graph G_2 corresponds to the actual connection,
- etc.

For each graph G_i, we have already estimated the probability $P(G_i)$ that this graph corresponds to actual connections. Similarly to the above situations, it is reasonable to assume that the corresponding events are independent. Thus, the probability C that e_1 and e_2 are actually connected – i.e., the probability that all the graphs in the chain correspond to actual connections – can be estimated as the product of the corresponding probabilities:

$$C = \prod_{i=1}^{t} P(G_i). \tag{23}$$

Comment. In particular, if we take into account that $P(G_i) = 1 - R(G_i)$ and that the values $R(G_i)$ are small, we can use a similar approximation as above and get an approximate formula

$$C \approx 1 - \sum_{i=1}^{t} R(G_i). \tag{24}$$

This enables us to gauge how our confidence that e_1 and e_2 are connected decreases when the chain gets longer. In the formula (23), our degree of confidence that e_1 and e_2 are connected is equal to the product of the probabilities $P(G_i)$ corresponding to all the graphs G_i in the chain relating e_1 and e_2. Each multiplication by the number $P(G_i) < 1$ decreases the product. The longer the chain, the smaller the product and thus, the smaller our degree of confidence that e_1 and e_2 are actually connected.

This solves one of the problems that we mentioned – that, contrary to intuition, in the semi-heuristic approach, the degree of confidence (as described by the clique size) does not decrease when the length of the chain increases.

6 Towards an Algorithm

How to take distance into account when estimating the probability: idea. As we have described earlier, the existing algorithm for checking when the two nodes are actually connected uses the distances, not just the graph. We therefore need to extend the above probabilistic analysis so that it takes into account the actual distances, not just whether there is an edge or not.

In the graph version, we assumed that there is a probability r that the edge between the nodes is accidental – and does not reflect the true connection between the nodes. Since an edge is placed when the distance is $\le \theta$, we thus assign the probability r to all distances $D \le \theta$ – and this value immediately jumps to 1 when the distance exceeds θ and therefore, there is no edge. The true probability should not change that abruptly, especially since the value θ has to be empirically determined – and may thus change from situation to situation.

In other words, instead of a *single* probability value r, we should come up with the value $r(D)$ *depending on the distance* – and make sure that this dependence on D is continuous, with no abrupt jumps. This function should be non-decreasing:

- when the distance increases,
- the probability that the entities are not actually connected should also increase (or at least not decrease),

i.e., $D \le D'$ should imply $r(D) \le r(D')$.

To find such a function, let us consider the situation in which a node e' is in between nodes e and e'', in the sense that $D(e, e'') = D(e, e') + D(e', e'')$, i.e., the distance $D(e, e'')$ is equal to the sum $D + D'$, where we denoted $D \stackrel{\text{def}}{=} D(e, e')$ and $D' \stackrel{\text{def}}{=} D(e', e'')$. By definition of the function $r(D)$:

- the probability that the entities e and e' are actually connected is equal to $1 - r(D)$;
- the probability that the entities e' and e'' are actually connected is equal to $1 - r(D')$; and
- the probability that the entities e and e'' are actually connected is equal to $1 - r(D + D')$.

The nodes e and e'' are actually connected if both e is connected to e' and e' is connected to e''. Similar to the previous parts of this chapter, it is reasonable to assume that the corresponding events are independent. Thus, we get

$$1 - r(D + D') = (1 - r(D)) \cdot (1 - r(D')). \tag{25}$$

Thus, a non-increasing function $p(D) \stackrel{\text{def}}{=} 1 - r(D)$ satisfies the functional equation $p(D + D') = p(D) \cdot p(D')$.

It is known (see, e.g., [1]) that all the solutions of such an equation have the form $p(D) = \exp(-a \cdot D)$ for some constant $a > 0$. Thus, we arrive at the following conclusion.

How probability depends on the distance. The probability $p(D)$ that two nodes are actually connected is equal to $p(D) = \exp(-a \cdot D)$ for some constant $a > 0$.

The parameter a needs to be determined empirically, based on the part of our data for which we already know which entities are actually connected and which are not.

The probability $r(D) = 1 - p(D)$ that there is no connection between the two nodes is therefore equal to $r(D) = 1 - \exp(-a \cdot D)$.

Detecting direct connections: case when we take distances into account. Similar to the graph case, we first compute, for each node e, the probability that all connections from e to nodes from $G - \{e\}$ are accidental. Just like in the graph case, this probability is equal to the product of the probabilities $\exp(-a \cdot D(e, e'))$ that the distance between e and e' does not imply an actual connection. This product is equal to $\prod_{e' \neq e} \exp(a \cdot D(e, e'))$.

This formula can be simplified.

- First, we can easily add $e' = e$ to the product, since for $e' = e$, we have $D(e, e') = 0$ and thus, the factor $\exp(-a \cdot D(e, e)) = 1$ does not change the overall product.
- Second, we can use the fact that the product of the exponents is equal to the exponent of the sum. As a result, we get a simplified formula

$$\exp\left(-a \cdot \sum_{e' \in G} D(e, e')\right). \tag{26}$$

Thus, the probability that e *is* actually connected to $G - \{e\}$ is equal to

$$1 - \exp\left(-a \cdot \sum_{e' \in G} D(e, e')\right). \tag{27}$$

The probability $P(G)$ that all nodes from G are actually connected can be now estimated as the product of the probabilities corresponding to different nodes $e \in G$:

$$P(G) = \prod_{e \in G} \left(1 - \exp\left(-a \cdot \sum_{e' \in G} D(e, e')\right)\right). \tag{28}$$

Comment. In the first approximation, we get a simplified formula

$$P(G) \approx 1 - \sum_{e \in G} \exp\left(-a \cdot \sum_{e' \in G} D(e, e')\right). \tag{29}$$

Towards an algorithm. We start building a chain with $c_1 = e_1$. In the original method, we only considered k-cliques; now, we are allowing graphs which are "almost" cliques.

For each such graph, we can use the formula (29) to estimate the probability $P(G)$ that this nodes from this graph are actually connected. For each node e' from this graph, we probability that it is actually connected to c_1 is equal to $p(G)$ and the probability that it is actually connected to e_2 is equal to

$$\exp(-a \cdot D(e', e_2)). \tag{30}$$

Thus, the probability that e_1 and e_2 are connected via e' is equal to the product of these two probabilities, i.e., to $P(G) \cdot (1 - \exp(-a \cdot D(e', e_2)))$. As the next node in the connecting chain, we then select the most probable connecting node e', i.e., the node for which this product is the largest possible.

Then, we repeat the same procedure starting with c_2, etc., until we reach e_2. As a result, we arrive at the following algorithm.

7 Resulting Algorithm

Formulation of the problem: reminder. We want to check whether the given nodes e_1 and e_2 are actually connected – and if yes, we want to design a chain of events $c_1 = e_1, c_2, \ldots, c_t$, and $c_{t+1} = e_2$ in which each c_i is directly connected to c_{i+1} (and the corresponding chain of connecting graphs $G_1, \ldots, G_t$).

We also want to compute the probability P that the corresponding chain reflects the actual connection.

First preliminary step: finding the parameter $a > 0$. Based on the part of the data for which we already know which entities are actually connected and which are not, we estimate the parameter $a > 0$ for which the probability $p(D)$ that nodes at distance D are actually connected decreases as $\exp(-a \cdot D)$.

This value can be estimated, e.g., if for different values d, we estimate, among all pairs nodes of distance approximately D, the proportion $\widetilde{p}(D)$ of pairs were actually connected. Then, we try to find a for which, for all these values D, we have $\widetilde{p}(D) \approx \exp(-a \cdot D)$. To estimate a, we can, e.g., take negative logarithm of both sides, and use the Least Squares Method (see, e.g., [17]) to solve the resulting system of approximate linear equations $a \cdot D \cdot \approx -\ln(\widetilde{p}(D))$.

Second preliminary step: finding neighborhoods. Similar to [8, 9], use the concept lattice algorithms to come up, for each entity e, with a list of the closest ones. Then, for each node e and for each m, we can find a *m-neighborhood* of e – i.e., the set consisting of m closest nodes. For this, we can use, e.g., an algorithm for computing the concept lattice (as in [8, 9]).

The corresponding value m and the value k (which is used in the main part of the algorithm) are chosen in such a way that it is computationally feasible to try all possible subsets of $\le k - 1$ elements out of m.

Main part of the algorithm. We start with $c_1 = e_1$. Then, we select the nodes c_2, c_3, ..., c_t one by one.

When we reach the node c_i, we estimate the probability P_i that c_1 and c_i are actually connected. We start with the probability $P_1 = 1$ (reflecting the fact that the node e_1 is clearly connected to itself).

For every i, once the node c_i has been selected and the value P_i has been computed, we find m nodes which are the closest to c_i. Out of these m nodes, we test all possible subsets of $\le k-1$ nodes. To each of these subsets, we add the node c_i and consider the corresponding graph G. For this graph G, we compute the probability

$$P(G) = \prod_{e \in G} \left(1 - \exp\left(-a \cdot \sum_{e' \in G} D(e, e')\right)\right). \tag{31}$$

Then, for each point $e' \in G - \{c_i\}$, we compute the product

$$P(G) \cdot (1 - \exp(-a \cdot D(e', e_2))). \tag{32}$$

Once we have tested all such subsets G and computed the product for all their elements $e' \in G$, we select, as the next node c_{i+1} in the chain, the node e' for which the product corresponding to this node is the largest possible. The corresponding graph G is selected as the connecting graph G_i. We then compute $P_{i+1} = P_i \cdot P(G_i)$.

- If the probability P_{i+1} goes below a certain threshold P_0, we conclude that e_1 and e_2 are not actually connected (or, to be more precise, that, based on the available information, we cannot make such a conclusion).
- If $c_{i+1} = e_2$ and $P_{i+1} \ge P_0$, then we conclude that the given nodes e_1 and e_2 are actually connected, with degree of confidence $P = P_{i+1}$.
- If $c_{i+1} \ne e_2$ and $P_{i+1} \ge P_0$, we continue iterations.

8 Conclusions

In many practical situations, it is important to check which entities are actually connected and which are not. Usually, this checking is performed by using the traditional statistical methods – but these methods cannot be applied when we have a large amount of data points ("big data"). A semi-heuristic method was proposed to detect actual connections in the case of big data; however this method has limitations: first, it is justified by experimental results and requires theoretical justification, and second, the method depends on "crisp" granules (cliques) to form connections.

In this chapter, we have come up with a theoretical justification of the known semi-heuristic method, and we have come up with a new, more flexible definition of almost-granules. However, a lot of work is still ahead: there is still a lot of room for improvement in how we can effectively process big data to find such almost-granules and to compute their degree of granule-ness.

Acknowledgments. This work was supported in part by the National Science Foundation (NSF) grants HRD-0734825 and HRD-1242122 (Cyber-ShARE Center of Excellence), NSF grant DUE-0926721, and by M. S. Hossain's startup grant at UTEP.

References

[1] J. Aczel, *Functional Equations and Their Applications*, Academic Press, New York, 1966.

[2] J.-P. Brassard and J. Gecsei, "Path Building in Cellular Partitioning Networks", *ACM SIGARCH Computer Architecture News*, 1980, Vol. 8, No. 3, pp. 44–50.

[3] A. Di Ciaccio, M. Coli, and J. M. Angulo Ibanez (Eds.), *Advanced Statistical Methods for the Analysis of Large Data*, Springer Verlag, Berlin, Heidelberg, 2012.

[4] C. Faloutsos, K. S. McCurley, and A. Tomkins, "Fast Discovery of Connection Subgraphs", *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining KDD'04*, Seattle, Washington, August 22–25, 2004, pp. 118–127.

[5] L. Fang, A. D. Sarma, C. Yu, and P. Bohannon, "Rex: explaining relationships between entity pairs," *Proceedings of the VLDB Endowment*, 2011, Vol. 5, No. 3, pp. 241–252.

[6] K. Heath, N. Gelfand, M. Ovsjanikov, M. Aanjaneya, and L. Guibas, "Image Webs: Computing and Exploiting Connectivity in Image Collections," *Proceedings of the 23th IEEE Conference on Computer Vision and Pattern Recognition CVPR'2010*, San Francisco, California, June 13–18, 2010, pp. 3432–3439.

[7] M. S. Hossain, M. Akbar, and N. F. Polys, "Narratives in the Network: Interactive Methods for Mining Cell Signaling Networks", *Journal of Computational Biololy*, 2012, Vol. 19, No. 9, pp. 1043–1059.

[8] M. S. Hossain, P. Butler, A. P. Boedihardjo, and N. Ramakrishnan, "Storytelling in Entity Networks to Support Intelligence Analysts", *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining KDD'12*, Beijing, China, August 12–16, 2012, pp. 1375–1383.

[9] M. S. Hossain, J. Gresock, Y. Edmonds, R. Helm, M. Potts, and N. Ramakrishnan, "Connecting the Dots between PubMed Abstracts", *PLoS ONE*, 2012, Vol. 7, No. 1, Paper e29509.

[10] G. Klir and B. Yuan, *Fuzzy Sets and Fuzzy Logic*, Prentice Hall, Upper Saddle River, New Jersey, 1995.

[11] D. Kumar, N. Ramakrishnan, R. Helm, and M. Potts, "Algorithms for Storytelling", *IEEE Transactions on Knowledge and Data Engineering*, 2008, Vol. 20, No. 6, pp. 736–751.

[12] C. D. Manning, P. Raghavan, and H. Schütze, *Introduction to Information Retrieval*, Cambridge University Press, Cambridge, Massachusetts, 2008.

[13] H. T. Nguyen and E. A. Walker, *A First Course in Fuzzy Logic*, Chapman and Hall/CRC, Boca Raton, Florida, 2006.

[14] F. J. Ohlhorst, *Big Data Analytics*, John Wiley & Sons, 2012.

[15] A. Rajaraman and J. D. Ullman, *Mining of Massive Datasets*, Cambridge University Press, Cambridge, Massachusetts, 2011.

[16] R. Roy and D. W. J. Olver, "Lambert W function", In: W. J. Olver, D. M. Lozier, R. F. Boisvert, and C. F. Clark, *NIST Handbook of Mathematical Functions*, Cambridge University Press, Cambridge, Massachusetts, 2010.

[17] D. J. Sheskin, *Handbook of Parametric and Nonparametric Statistical Procedures*, Chapman and Hall/CRC Press, Boca Raton, Florida, 2011.

[18] S. Srinivasa and V. Bhatnagar (Eds.), *Big Data Analytics*, Proceedings of the First International Conference on Big Data Analytics BDA'2012, New Delhi, India, December 24–26, 2012, Springer Lecture Notes in Computer Science, Vol. 7678, 2012.

[19] D. R. Swanson, "Complementary Structures in Disjoint Science Literatures", In: A. Bookstein, Y. Chiaramella, G. Salton, and V. V. Raghavan (Eds.), *Proceedings of the 14th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval SIGIR'91*, Chicago, Illinois, October 13–16, 1991, pp. 280–289.

[20] L. A. Zadeh, "Fuzzy sets", *Information and Control*, 1965, Vol. 8, pp. 338–353.