NATIONAL AND KAPODISTRIAN UNIVERSITY OF ATHENS

SCHOOL OF SCIENCES
DEPARTMENT OF INFORMATICS & TELECOMMUNICATIONS

MASTER'S PROGRAM
"DATA SCIENCE AND INFORMATION TECHNOLOGIES (DSIT)"
SPECIALIZATION: BIOINFORMATICS BIOMEDICAL
DATA SCIENCE

DIPLOMA THESIS

Data Exploration and Retrieval for Hematological Markers Networks

Michael A. Batskinis

Supervisor: **Theodore Dalamagas,** Research Director, Information Management Systems Institute, Athena Research Center

ATHENS

OCTOBER 2021

ΕΘΝΙΚΟ ΚΑΙ ΚΑΠΟΔΙΣΤΡΙΑΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

ΔΙΙΔΡΥΜΑΤΙΚΟ ΠΡΟΓΡΑΜΜΑ ΜΕΤΑΠΤΥΧΙΑΚΩΝ ΣΠΟΥΔΩΝ
“DATA SCIENCE AND INFORMATION TECHNOLOGIES (DSIT)”
ΕΙΔΙΚΟΤΗΤΑ: ΒΙΟΠΛΗΡΟΦΟΡΙΚΗ- ΕΠΙΣΤΗΜΗ ΒΙΟΛΟΓΙΚΩΝ ΔΕΔΟΜΕΝΩΝ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Εξερεύνηση και Ανάκτηση Δεδομένων για Δίκτυα Αιματολογικών Δεικτών

Μιχαήλ. Α. Μπατσκίνης

Επιβλέπων: **Θεόδωρος Δαλαμάγκας,** Διευθυντής Ερευνών, Ινστιτούτο Πληροφοριακών Συστημάτων, Ερευνητικό Κέντρο Αθηνά

ΑΘΗΝΑ

ΟΚΤΩΒΡΙΟΣ 2021

DIPLOMA THESIS

Data Exploration and Retrieval for Hematological Markers Networks

Michael A. Batskinis
DS2190011

SUPERVISOR: **Theodore Dalamagas,** Research Director, Information Management Systems Institute, Athena Research Center

EXAMINATION COMMITTEE

Theodore Dalamagas, Research Director, Information Management Systems Institute, Athena Research Center

Marianna Antonelou, Assistant Professor, Department of Biology, National and Kapodistrian University of Athens

Dimitrios Gunopulos, Professor, Department of Informatics and Telecommunications, National and Kapodistrian University of Athens

October 2021

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Εξερεύνηση και Ανάκτηση Δεδομένων για Δίκτυα Αιματολογικών Δεικτών

Μιχαήλ. Α. Μπατσκίνης
DS2190011

ΕΠΙΒΛΕΠΩΝ: **Θεόδωρος Δαλαμάγκας,** Διευθυντής Ερευνών, Ινστιτούτο Πληροφοριακών Συστημάτων, Ερευνητικό Κέντρο Αθηνά

ΕΞΕΤΑΣΤΙΚΗ ΕΠΙΤΡΟΠΗ **Θεόδωρος Δαλαμάγκας,** Διευθυντής Ερευνών, Ινστιτούτο Πληροφοριακών Συστημάτων, Ερευνητικό Κέντρο Αθηνά

Μαριάννα Αντωνέλου, Επίκουρη Καθηγήτρια, Τμήμα Βιολογίας, Εθνικό και Καποδιστριακό Πανεπιστήμιο Αθηνών

Δημήτριος Γουνόπουλος, Καθηγητής, Τμήμα Πληροφορικής και Επικοινωνιών, Εθνικό και Καποδιστριακό Πανεπιστήμιο Αθηνών

Οκτώβριος 2021

ABSTRACT

Hematological data graphs is a widely used tool to capture pathophysiological aspects of red blood cells for patient clinical studies. Those graphs model quantitative and qualitative characteristics of several hematological markers. A critical challenge is to be able to provide effective graph processing methods to assist data exploration and retrieval for those graphs, at various levels of granularity (raw data vs aggregate data). This study will adopt graph database technologies to develop a system to store, retrieve and explore hematological markers data. The thesis will carry out the following tasks: (a) graph data collection and preparation, (b) surveying state-of-the-art graph databases, (c) designing and developing a graph database for hematological markers graph data (e.g., based on Neo4j), (d) develop method for data exploration and retrieval based on real user examples.

SUBJECT AREA: Graph analytics
KEYWORDS: hematological markers networks, graph database, data exploration, centrality analysis, community detection

ΠΕΡΙΛΗΨΗ

Οι γράφοι αιματολογικών δεδομένων είναι ένα ευρέως χρησιμοποιούμενο εργαλείο για την καταγραφή παθοφυσιολογικών πτυχών των ερυθρών αιμοσφαιρίων σε κλινικές μελέτες ασθενών. Τέτοιου είδους δίκτυα μοντελοποιούν ποσοτικά και ποιοτικά χαρακτηριστικά αρκετών αιματολογικών δεικτών. Μια σημαντική πρόκληση είναι η δημιουργία αποτελεσματικών μεθόδων επεξεργασίας τέτοιου είδους γράφων με σκοπό την εξερεύνηση και την ανάκτηση δεδομένων. Η παρούσα διπλωματική εργασία έχει σκοπό να αξιοποιήσει τεχνολογίες βάσεων δεδομένων γράφων με σκοπό την ανάπτυξη ενός συστήματος αποθήκευσης, ανάκτησης και διερεύνησης δεδομένων αιματολογικών δεικτών. Η διατριβή θα εκτελέσει τις ακόλουθες εργασίες: (α) τη συλλογή και την προετοιμασία δεδομένων γράφων, (β) την εξερεύνηση πρότυπων βάσεων δεδομένων γράφων, (γ) τον σχεδιασμό και την ανάπτυξη βάσης δεδομένων γράφων για δεδομένα γράφων σχετικά με αιματολογικούς δείκτες (π.χ. στο Neo4j) και (δ) την ανάπτυξη μεθόδου για την εξερεύνηση και την ανάκτηση δεδομένων σχετιζόμενων με πραγματικά δεδομένα.

ΘΕΜΑΤΙΚΗ ΠΕΡΙΟΧΗ: Graph analytics
ΛΕΞΕΙΣ ΚΛΕΙΔΙΑ: δίκτυα αιματολογικών δεικτών, βάσεις δεδομένων γράφων, εξερεύνηση δεδομένων, ανάλυση κεντρικότητας, εντοπισμός κοινωνιών

CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

Red blood cells (RBCs) or erythrocytes are the most common type of blood cell. They have a flattened biconcave disk shape depressed in the center and no nucleus or organelles (e.g. mitochondria) [1]. The most important feature of RBCs is their three-layered membrane, to which they owe their increased flexibility and endurance [2]. The main function of RBCs is related to the process of gas exchange, which is carried by hemoglobin (Hb), a protein of RBCs. Besides that, they use glycolysis to generate energy carriers and they are closely related to Pentose Phosphate Pathway (PPP) [1]. A relative example is the case of Glucose 6-Phosphate Dehydrogenase (G6PD) deficiency, an inborn error of metabolism that results to reduced antioxidant capacity and increased susceptibility of RBC breakdown [3]. Since PPP is the only pathway that preserves their antioxidant capacity, any abnormality in that path is associated with increased levels of oxidative stress and eventually hemolysis [4].

Moreover, RBCs from G6PD deficient donors are more susceptible to the events that occur during the time they are stored in blood banks. That said, normally, erythrocytes can be converged in mannitol-containing storage solutions up to 42 days in 1^o-6^o C [5], while donor-related parameters such as the age, sex and ethnicity seem to play an important role as well [3]. During the time they remain in storage several metabolic and morphological alterations might occur, and while metabolic alterations are usually reversable, this does not apply for morphological changes [6]. Therefore, extensive stay in storage systems result to a proportion of deformed RBCs that are incompatible for transfusion, since they are prone to removal from the circulation [3]. That said, in the past years, several techniques have been developed to measure physiological parameters of RBCs, such as their mechanical fragility and the levels of radical products, that could give insight about their consistency during storage.

Additionally to that, the quantitative analysis of biological parameters (e.g. metabolites, proteins etc.) of RBCs during storage is another efficient approach towards the understanding of storage effect. A quite recent and very informative method for assessing the significance of the findings from such approaches is the construction of hematological networks that underline potential associations between biological components. The fundamentals of analyzing hematological data using correlation networks lay on the basic aspects of graph theory. A key issue for the hematology research community is to be able to illustrate and analyze hematological data using hematological graphs since, if designed carefully, can be a great asset for the community, providing answers to complex biological issues and potentially cut down the time spent in the lab. Major challenges to cope with, in order to efficiently manipulate such data, are related to:

- the modelling and construction of a conclusive graph that highlights homologous and heterologous associations between different biomedical parameters in a hematological network,
- the ability to organize and store graph entities so that they can be accessed and manipulated efficiently,

- the transferability of the graph through different graph databases and applications, and
- the analysis of hematological graphs utilizing complex techniques that could give insight about their structural characteristics, reveal their most crucial components, and help to better understand the complexity of the problem.

This Thesis presents a framework to support data exploration and retrieval for hematological marker networks. It adopts graph database technologies to develop a system to store, retrieve and explore hematological markers data, carry out the following tasks: (a) graph data collection and preparation, (b) surveying state-of-the-art graph databases, (c) designing and developing a graph database for hematological markers graph data (e.g., based on Neo4j) and (d) development of method for data exploration and retrieval based on real user examples. Next, we overview the key concepts of this work and results produced.

The main aspects of this work concern the analysis of biochemical and hematological data of G6PD deficient (G6PD-) donors using graph analytics. To construct of a conclusive hematological marker network an efficient amount of data was collected. The final dataset consisted of real user data of G6PD- donors and computationally verified data regarding the case-study biological problem that were retrieved from available sources. The next step towards the construction of the graph was the preprocessing and refinement of the dataset, in the sense of removing duplicate and missing values. Following that, a set of biological queries to which the final graph model had to be able to answer was collected. Some important biological scenarios were related to the identification of biologically converged parameters and significant intra-and inter-parameter associations, as well as the characterization of the most popular components of the case-study system.

Once a considerable number of queries was defined, the outline of the graph model was designed. Throughout this purpose all node and relationship types were defined based the biological group they represented (e.g. amino acids, proteins, physiological parameters etc.) or the association type (e.g. compound-physiological correlations), accordingly. The construction of the hematological markers network took place in the open-source browser-based version of Neo4j, a graph database that stores and presents, efficiently, accurately and with high speed, relational data in the form of nodes, edges, and properties. The implementation was conducted in Cypher query language. Using a set of predefined functions and algorithms we were able to perform several types of analyzes, identify homologous and heterologous correlations between components of same and different node types and compare in vivo (fresh RBCs) and in vitro systems (packed, stored RBCs). The most popular ones amongst them were Pearson's and Cosine similarity algorithms. Following that, in each type of correlation a different threshold was set, so that the most insignificant associations would be excluded from the final graph model. Moreover, the quality of intra- and inter- parameters correlations was further evaluated in terms of repeatability, in the sense of consistent is a relationship throughout the duration of storage.

Despite the fact the Neo4j was suitable for the development of the hematological markers network, it has limited visualization capabilities, especially for users without an IT background. Therefore, a different browser-based open-source tool was used as interface for the visualization and analysis of our graph model. The proposed tool was GraphXR, a web application that allows the user the ability to explore any graph data in 2D and 3D space and interact with ease, since it provides a set of predefined tools and algorithms that are necessary for any kind of graph analysis. Another great asset of GraphXR is the fact that interacts with Neo4j, in the sense that the user can easily load a copy of any working project from Neo4j to GraphXR. That said, we used GraphXR to perform centrality analysis by applying betweenness, degree and closeness centrality algorithms to identify the most popular nodes the graph, connectivity analysis to investigate the complexity and density of intra- and inter- parameter associations and community detection analysis to find cluster of nodes and gain insight about hub nodes and their neighbors.

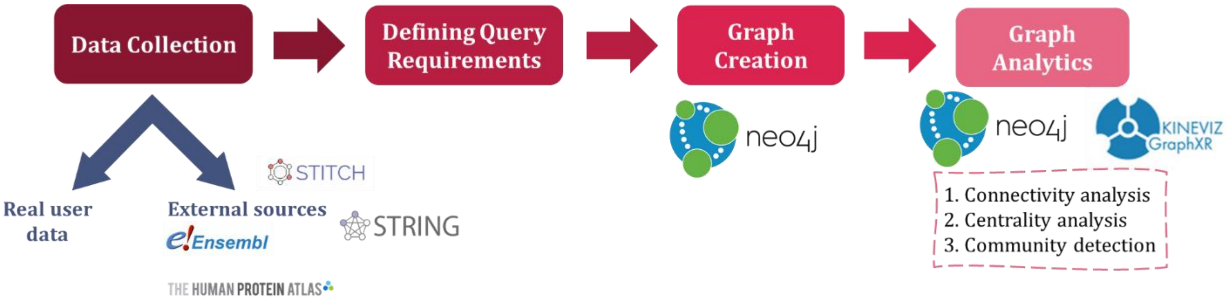


Figure 1. Workflow

Contributions.

The contributions of this Thesis are:

- the proposal and construction of a graph model that is related to hematological data from G6PD^{-} donors
- surveying available graph DBs and addressing the one that is more suitable for the current work
- understanding the biological aspects of the problem and defining a set of query requirements for hematological markers networks
- designing a graph DB for hematological data that could give insight about the inter- and intra- parameter correlations between graph entities
- addressing suitable graph analytics-related methods and demonstrating effective query solutions from hematological graphs
- providing detailed description of the programming that resulted to the creation of the final network by addressing the implementation in Cypher query language
- projecting the final graph model in an open-source browser-based user-friendly graph analytics-related visualization tool.

Outline.

The following Chapter refers to the biological and technical background of this work, while previous research and related work are also mentioned. Chapter 3 describes the process of data collection and pre-processing that leads to the final dataset, that was used for the construction of the graph. In Chapter 4, the query requirements, the setup of an appropriate graph model and the statistical analysis that led to the creation of the hematological markers network in Neo4j are described in detail. Chapter 5 addresses the appropriate tools for the visualization and analysis of relative graph models, the graph-related algorithms that are used throughout the analysis of our model and concludes with the demonstration of several biological scenarios that can be answered using the application GraphXR. Finally, in Chapter 6 the conclusions and some future ideas regarding the current work are mentioned.

2. BACKGROUND AND RELATED WORK

2.1 Background

2.1.1 Main features of Red Blood Cells (RBCs)

Red blood cells (RBCs) or erythrocytes are produced from pluripotent haemopoietic cells which are originated from the bone marrow. These stems cells after a series of events differentiate to RBCs. In humans, RBCs have a flattened biconcave disk shape depressed in the center [1]. A physiological erythrocyte usually has a diameter of 6.2 – 8.2 μm and a thickness of approximately 2 – 2.5 μm at its thickest part and 0.8 – 1 μm at its more flattened point (the center) [7]. Additionally, mature red cells have neither a nucleus or organelles (e.g. mitochondria), thus they do not contain any nucleic acid (DNA or RNA) and cannot divide or carry out protein synthesis and they have limited repair mechanisms [8]. A very interesting structural feature of RBCs is their membrane, to which they owe their increased flexibility and endurance. That said, RBC's membrane consists of three layers: the exterior one which is rich in carbohydrates, the lipid layer which besides the lipidic components (mainly phospholipids and cholesterol) contains many transmembrane, integral proteins and the membrane skeleton in the inner side of the lipid bilayer. Additionally, it is noteworthy the fact that in a typical human red cell half of the membrane mass consists of proteins [2].

The most important function of RBCs is related to the process of gas exchange [1]. In vertebrates, gas exchange is conducted with the transfer of O_2 and CO_2 between the blood system and the lungs. This process is carried by hemoglobin (Hb), a protein of RBCs. Typically, Hb consists of four – per two identical – globular subunits and a heme molecule which contains an iron ion and is the binding place of the O_2. The iron comes in two states, the ferrous (Fe^{+2}) and the ferric (Fe^{+3}) states. That said, when the iron ion is in its ferrous state, the Hb is capable of binding O_2, while in ferric state Hb is not able to transfer O_2 (methemoglobin). In such cases, an enzyme called methemoglobin reductase catalyzes the reduction of Fe^{+3} to Fe^{+2} [9]. While there are several types of Hb in humans, depending on the age, in adults the most common is formed by two alpha chains and two beta chains ($\alpha_2\beta_2$) [10].

Besides gas exchange, RBCs participate in the immune response of the body by, indirectly, killing pathogens, that have infected them, with free radicals that are released from the Hb of lysed red cells [11]. Moreover, as it was mentioned, erythrocytes lack of mitochondria, so they make up for the energy they lose through the glycolysis of glucose and the subsequent lactic acid fermentation of the produced pyruvate [12]. They are, also, closely associated with the Pentose Phosphate Pathway (PPP). A relative example is the case of **Glucose 6-Phosphate Dehydrogenase (G6PD) deficiency.** G6PD deficiency is a recessive X-linked inborn error of metabolism that results in reduced antioxidant capacity and increased susceptibility of RBC breakdown [3]. Typically, G6PD is an enzyme of the PPP and conducts the conversion of glucose 6-phosphate to 6-phosphoglucono-δ-lactone and maintains the levels of antioxidant equivalents, such as NADPH. In RBCs is the only pathway that preserves their antioxidant capacity.

Therefore, in the case of G6PD deficiency, erythrocytes are exposed to free radicals that in the event of extensive oxidative stress result to hemolysis [13].

2.1.2. RBCs' parameters measured during storage in blood bank conditions

Erythrocytes are the most commonly transfusable and highly demanded cells worldwide, a fact that can be verified by the hundreds of millions of red cell units that are being stored in blood banks and transfused every year. Donors' biological profile along with the effect of the extensive storage are parameters that affect the homeostasis of RBCs, making a proportion of them insufficient for transfusion [5]. Parameters such as the age, the sex and the ethnicity of the donor seem to play an important role in the consistency of red cells during storage [3]. On top of that studies regarding the storability of RBCs have shown that erythrocytes can be conserved in mannitol-containing storage solutions up to 42 days at 1-6 degrees [5]. Several alterations regarding metabolic and morphological features of RBCs are likely to occur during the period that they remain in blood banks, and while the effects in metabolic parameters are most of the times reversable, this is not the case for their morphology [3]. It is known that the membrane of erythrocytes can shapeshift in cases of applied stress (e.g. mechanical stress) [14] and while this feature is quite important during the microcirculatory blood flow, it has been related to several pathological conditions, one of which is the outcome of extensive stay of RBCs in storage [15].

Deformed packed red blood cells (pRBCs) are likely to cause harmful effects and result to an ineffective blood transfusion. That said, there are several techniques that have been developed in the past years with which we can measure significant physiological parameters of RBCs in the circulation that could give insight about the vitality of erythrocytes during storage. Some of the most important ones will be described below in more detail:

- Erythrocyte osmotic fragility **(Mean Corpuscular Fragility, MCF)** is the degree of hemolysis that comes from subjecting RBCs to osmotic stress by putting them in hypotonic solution. As a metric MCF has been used for the diagnosis of diseases related to genetic abnormalities to the membrane of erythrocytes [16], such as hereditary spherocytosis.
- Erythrocyte mechanical fragility **(MFI)** is related to the part of hemolysis that results by applying mechanical stress to RBCs, such as the kind of stress that occurs during the microcirculatory blood flow. While there are several approaches regarding the measurement of MFI, there is not a commonly acceptable practice. However, it is an important parameter to gain insight about the health of RBC membrane and submembrane skeleton and, additionally, can be very handy in cases of evaluating the quality of pRBCs via *in vitro* testing [17].
- **Cell free Hb** is another hematological parameter that is measurement in several diagnostic experiments regarding the consistency of RBC. As it was mentioned before, typically, Hb is a protein inside the RBCs and plays an important role during gas exchange. However, in cases of hemolysis Hb is released from erythrocytes and flows freely in the vascular system causing outspread oxidative damage. That said,

by quantifying the concentration of free Hb in the supernatant of pRBCs one might gain insight about the health of erythrocytes or indication about storage lesion [18].

- Reactive Oxygen Species **(ROS)** are highly reactive molecules that under extensive amounts can induce oxidative stress causing serious damage to cells and their components, while they play an important role in cell ageing [19]. Accumulation of ROS in RBCs during storage is a common cause. However, since erythrocytes of G6PD deficient donors lack of antioxidant equivalents, their intracellular environment tends to produce more ROS, making them more susceptible to hemolysis [20]. Closely related to the quantification of ROS is the measurement of the **antioxidant capacity** of the plasma or supernatant for the same reasons that were mentioned before.
- Another physiological parameter that could give insight about the vitality of erythrocytes is the characterization of the levels of their **deformability** since irreversible change in their morphology may cause to ineffective transfusions.
- Mean corpuscular volume **(MCV)** is the ratio of hematocrit and the total number of RBCs in the blood and it is used as aid for characterization of microcytic anemia (MCV lower than normal) and macrocytic anemia (MCV greater than normal) [21].
- Another parameter that is particularly important for this work, since we study the hematological profile of G6PD deficient donors, is the **activity of G6PD** during storage. Previous studies have shown that the activity of G6PD in pRBCs tends to decrease during storage compared to fresh blood [3].

2.1.3 Data Analysis using Knowledge Graphs

<u>The concept of a graph</u>

According to graph theory, a graph is a set of entities, some of which form pairs of connections. The entities of a graph are called nodes or based on discrete mathematics *vertices*, and the pairs of connections are known as relationships or edges [22]. Additionally, if the nodes and edges of a graph demonstrate real data, such as the connections of a person in social media or the metabolic pathways of cancer cells, then that graph is called a **knowledge graph** [23]. Based on the type of relationships that are formed between the nodes, we can distinguish four different types of knowledge graphs. If the edges of a graph have orientation, then it is called a *directed graph*. In the case that the links between the nodes have not a direction, the graph is called *undirected*, while if some edges have orientation and some not, the graph is called a *mixed graph*. The last type is called *weighted graph* and refers to the fact that to each edge a number (weight) is assigned [24]. Depending on the data type this weight can either demonstrate the cost, the length (e.g. world map network) or the strength of the connection (e.g. protein-protein interaction networks).

In Figure 2, the different categories of networks according to their relationship types are presented. On the top left of the figure we can observe a toy example of a directed graph (1), while next to it the representation of an undirected graph is shown (caption 2). An example of a mixed graph is available on caption 3 and the example concerning the weighted graph is presented on bottom right (caption 4).

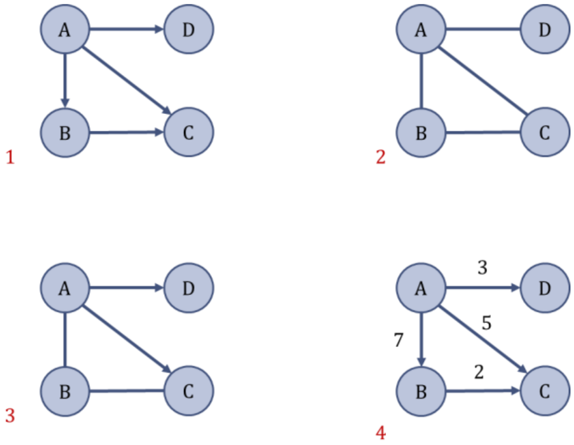


Figure 2. Network styles according to edge types.

Some other major characteristics that are significant for each network, besides the type of edges, are degree of nodes and the diameter of the graph. The degree of a node is the number of edges that forms with the rest of the nodes [22]. More specifically, in a directed graph one can distinguish the indegree, which is the total number of incoming relationships, and the outdegree, which, as its name denotes, is the total number of outgoing edges [25]. By computing of edges of each node, one can identify those with high-degree and, therefore, are more central in the graph. These nodes are characterized as hub nodes and they usually have a significant impact in the consistency and robustness of a graph, since if we remove them the network will collapse [26]. On the other hand, the diameter of a graph is the maximum distance between a pair of nodes. The denser the connections in a graph the smaller its diameter is. Its value is bound in [1,infinite) with infinite to be when the graph is totally disconnected or else it has no edges [22].

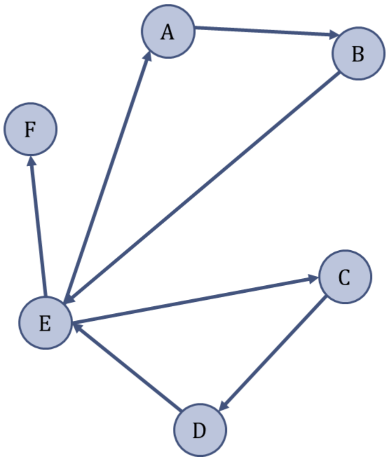


Figure 3. Explaining the terms of degree, hub, and diameter.

A toy example that describes the terms of degree, hub and diameters is shown in Figure 3. Since the degree of a node is its number of relationships it has, one can easily notice that the degree of A, B, C and D is 2, while for node F is one. At last, node E has 5 relationships, meaning that is the node with the higher degree. Moreover we can observe that by removing node E from the network, it immediately falls apart. Therefore, we can assume that node E is a hub node. On what matters the diameter of this toy example we can see that the longest path is the one from node C to node B (C→D→E→A→B). That said, the diameter of this network is 4, as the number of steps it takes to go from C to B. So far, we have described the concept of a knowledge graph and its entities along with some of the most significant characteristics of a network. Following that, it is important to discuss the aspects of graph analytics and the impact each one has. **Graph analytics** or else network analysis is the analysis of associations between different elements of a graph. There are several approaches when it comes to explore graph data, such as identifying the most important nodes of the network or else the ones that have more influence to it (Centrality analysis), investigating the density of connections between the entities of the graph (Connectivity analysis) or their classification into strongly connected groups or modules with similar characteristics (Community detection) or inspecting the reachability from one to node to another (Path analytics) [27]. For instance, estimating the influence of a person in a social network could be a good example of the Centrality analysis, while finding the best path in a weighted graph (also known as shortest path in graph theory) that connects two cities in a world map network could be a good application of Path analytics.

In general, graph analytics can be applied in a wide range of operations, such social networks (e.g. identifying people with great influence in social media), national security (e.g. detecting fraud in e-commerce businesses) or healthcare (e.g. spreadability of a COVID-19 virus) For each approach there are several graph analytics-related algorithms that help us get insight about the associations between nodes or relationships, and most of them will be discussed later in more detail.

Neo4j: working with graph databases

Given the fact that the volume of data is constantly increasing, it is quite important to work with or develop tools that can manipulate extensive amounts of information with a considerably high performance. In the case of graph analytics there are several web-based data storages that can implement such tasks with high speed and accuracy, and they are called graph databases. A **graph database** is a NoSQL database that stores and represent data in the form of nodes, edges/relationships, and properties [28].

A great asset of any graph database is the fact that edges are collected in such way, so that they can be retrieved or represented, usually, with a few high-speed operations [29]. Another major characteristic of the graph databases, closely related to their high performance, is the fact they consider relationships as their top priority in terms of storing and manipulating data. Thus relationships can be stored, separately, under specific labels and take additional information (properties), similarly to nodes, which gives the ability to the user to display efficiently any relationship type along with the connected nodes [30]. Up to now there have been reported several graph databases. A list with some of the most noteworthy ones is available on Table 1.

Table 1. A list with the most remarkable graph databases.

Name	**Latest version**	**Details**
Amazon Neptune	7.0.0 (April 2020)	A graph database established by Amazon and part of Amazon Web Services. Supports Apache, TinkerPop, Gremilin and SPARQL query languages

ArangoDB	3.7.2 (August 21, 2020)	A NoSQL database. Supports three different data structures (key/value, graphs and documents) and AQL (Arango Query Language)
Cayley	0.7.7 (October 15, 2019)	An open-source graph database inspired by Google's Knowledge Graph database. Supports three query languages, namely, Gizmo, GraphQL and MQL [31]
DataStax	6.0.1 (June 2018)	An enterprise graph database supporting TinkerPop and unifying with Cassandra
FlockDB	1.8.5 (February 23, 2012)	An open-source graph database that works with wide yet shallow networks. Performs well with rapid set operations [32]
Neo4j	4.3.3 (August 2021)	A graph database with open source and enterprise editions. Provides both server and desktop versions. It is accessible from most of programming languages through its REST API interface [33]
OrientDB	3.0.28 (Feb 2020)	Similarly to Neo4j, it provides both a community and enterprise edition. Supports a query language like SQL and it can be accessible through its REST or JSON API [34]

It is important to mention that for this work the browser-based open-source version of Neo4j was selected as the environment for the development of the hematological markers network. As it mentioned in Table 1 Neo4j is graph database which comes in a free and a commercial edition. For any implementation the Cypher query language is used. However, it can be accessible by many programming languages through the API interface [33]. Any graph data is stored in Neo4j under the form of a node, relationship, or property. Nodes and relationships can have multiple properties and, additionally, they can be categorized in groups under specific labels, so that they can be easily accessible. Relationships can either be have one orientation or be bidirectional (also known as undirected) or start and end to the same node (self-loops) [33]. Properties can be stored in several formats such as string, integer, float or boolean.

In Figure 4, a infographic example that describes the representation of graph entities in Neo4j is presented. In this toy-example one can notice that there is one node type under the label "User" and one relationship type under the label "KNOWS". Moreover, the nodes have two properties, one string-type attribute which gives insight about the name of the user and one number-type which provides information about the year of birth. Regarding the relationships, one can observe that all edges have one orientation and one property, which represents the relationship the connected users have.

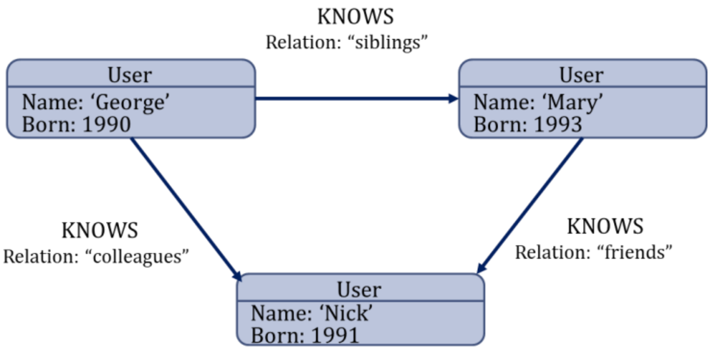


Figure 4. An infographic representation of graph entities in Neo4j.

Prerequisites to use the browser-based open-source edition of Neo4j is to have a compatible of JAVA and download and install several libraries with a set of predefined graph-related functions available by Neo4j that are necessary for most of the graph development issues. The first library is called the APOC library (Awesome Procedures On Cypher library) and consists a set of operations that are useful for manipulating graph data, such as importing/exporting data in CSV format, simplified vector-related functions, or reforming graph entities. Another mandatory library is the GDS library (Graph Data Science library) which contains a list of algorithms related to graph analytics. More specifically, all operations regarding centrality analysis, community detection, path analytics or comparing different networks are provided by this library. All information about the installation of these libraries and the use of each algorithm are fully described in the documentation of Neo4j platform. However, to use any function of the above one needs to be familiarized with Cypher query language.

Cypher as a query language was designed to be used within Neo4j system and, even though presents some similarities with SQL, serves the needs of graph database system. That said, Cypher is built according to the concepts of graph theory [35] and is mainly based on patterns of nodes and relationships, which are further filtered by their properties. Therefore, nodes are presented with parentheses surrounding their label and properties, while relationship types are depicted with square brackets. Undirected relationships are shown with dashes, while arrows are used to present the directed ones [33]. Like most of the query languages, Cypher includes a set of keywords to specify patterns of nodes and edges, filter graph entities and return results in the form of tables or graphs. Some of the most used keywords will be described below:

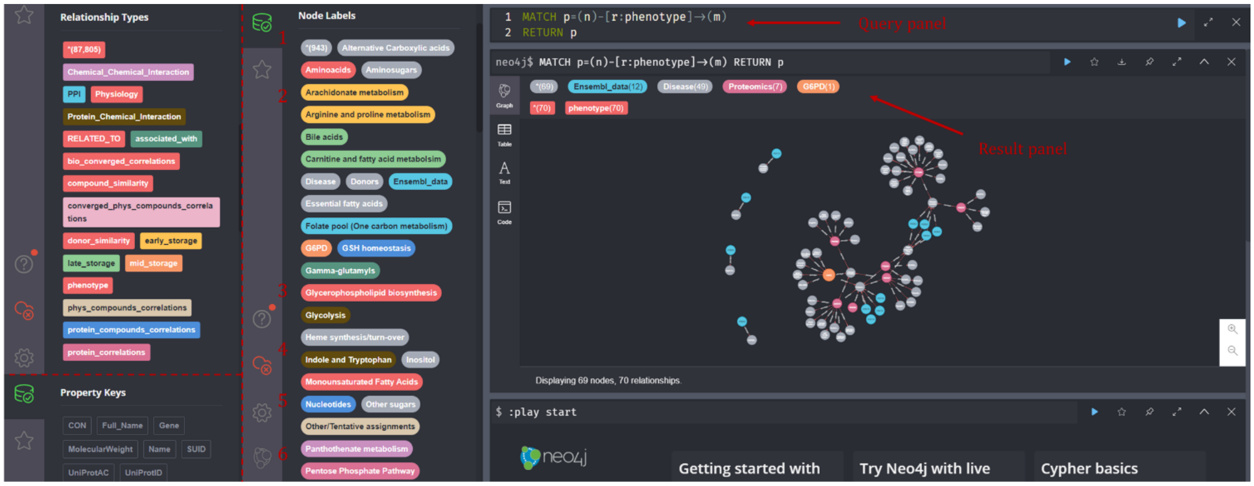
- the LOAD CSV (optional: WITH HEADERS) FROM [path] query is used to import data from csv files. The path of the file needs to be declared
- the MATCH keyword specifies the pattern of nodes and relationships to look at in the graph database
- the MERGE keyword is used to create new nodes and relationships without generating duplicates
- the WHERE clause filters entities of the pattern that specified with MATCH query
- the WITH statement gives the ability to concatenate query parts, using the result of one part as the starting point to the next one
- the RETURN query states what will be presented in the result panel [33].

Following that, an example using Cypher queries will be described in detail:

```
LOAD CSV WITH HEADERS FROM
"file:///C:/Users/mbats/OneDrive/Desktop/Ensembl_interactions.csv" AS data
WITH data
MERGE (n:Ensembl_data{UniProtID:data.UniprotID})-[r:phenotype]-
(m:Disease{Name:data.disease,source:data.source})
WITH n, r, m
MATCH (p)
WHERE (p:Proteomics or p:G6PD) AND p.UniProtID = n.UniProtID
DELETE n, r
MERGE (p)-[:phenotype]->(m)
```

This example starts by importing data from “Ensembl_interactions.csv” using the LOAD CSV clause. After the that, with the use of the MERGE keyword a path regarding diseases and proteins is passed to the graph and by using the WITH statement resulting graph data are passed to the next part of the query. Following that, the MATCH keyword is applied to look at all nodes of the graph and by using the WHERE statement a filtering process starts which concludes with deleting duplicate entities using the DELETE keyword and connect existing non Ensembl disease-related proteins with diseases/phenotypes.

This section concludes with a presentation view of the server edition of Neo4j (Figure 5). The Neo4j platform consists of three parts. The first one is the Tools panel on the of the display screen which contains a set of shortcuts such as displaying labels and property names in lists (1), saving favorite queries (2), help about Neo4j-related keywords and functions (3), connecting to cloud (4), browser settings (5) and general information about Neo4j (6). The most common of those options is the first one since it gives the ability to the user to navigate through different node and relationship types and display desired ones with ease. In this Figure part of node labels, relationship types and property names are displayed on the left. The second part of the Neo4j browser is the Query panel which is the space to write any Cypher query you want to be executed by Neo4j, the results of which will be displayed in the Result panel in the form of a graph or a table. The result panel is quite informative since it shows the type of nodes and relationships that are currently displayed and it gives the ability to export the output in CSV, JSON, PNG or SVG format (the last two options are available only in the case the result is graph). Moreover, nodes and relationships are presented in different colors according to the type they belong to and the user can select manually the size and color of graph entities, as well as the desired node property name to be displayed.



```
MATCH p=(n)-[r:phenotype]→(m)
RETURN p
```

```
neo4j$ MATCH p=(n)-[r:phenotype]→(m) RETURN p
```

Displaying 69 nodes, 70 relationships.

```
$ :play start
```

Figure 5. The interface of browser-based edition of Neo4j.

2.2. Related Work

Antonelou *et al.* 2018 [3]

The authors of this work studied the metabolic and physiological correlations in erythrocytes from G6PD deficient donors in both fresh blood and packed, stored cells. For this work RBCs from the venous blood of six male G6PD deficient donors(G6PD$^-$) and three male control donors (G6PD$^+$) of same age were subjected to analysis. The samples were stored up to 42 days and in the meanwhile weekly samplings took place, starting from the day RBCs were collected. That said, blood samples could be divided in two systems regarding the sampling stage. Thus, erythrocytes that were retrieved the first day of the experiment (day 0) are characterized as the *in vivo* system, while packed RBCs will be referred as the *in vitro* system concerning the samplings of days 7, 14, 21, 28, 35 and 42. During each sampling the measure of several physiological (e.g. MFI, MCV, G6PD activity), metabolic (e.g. amino acids, nucleotides) and proteomic parameters were estimated.

Once all data were collected, they used them for the construction of hematological networks with which correlations between parameters of the *in vivo* and *in vitro* system were estimated. Moreover, to increase the significance of their findings, the authors compared the data retrieved from fresh blood cells with data from every sampling of the *in vitro* system (e.g. D0 vs D7, D0 vs D14 etc.) and they considered as converged correlations those that were observed repeatedly at multiple sampling stages. Pearson's correlation analysis was performed for the estimation of any potential correlations. The creation of each network was conducted in Cytoscape and inverse Pearson's coefficient was used as a metric for defining the length of an edge, in a sense that the greater the Pearson's r value, the stronger the connection between two components was.

Findings regarding the analysis of the metabolic profile of G6PD$^-$ donors highlighted bioactive lipids, free fatty acids, bile acids, glycolytic metabolites, purines, and amino acids as top discriminative metabolic parameters for G6PD$^-$ donors. On top of that, from the comparative analysis of G6PD$^-$ and control donors, parameters related to one carbon or sulfur metabolism (e.g. methionine), antioxidant capacity (e.g. NADPH) or glutathione homeostasis were characterized with significantly decreased levels compared to control donors. Equally notable were the results regarding changes in the concentration of compounds related to lipid metabolism. Another interesting finding of this study was the storage effect on both G6PD$^-$ and control samples. It seems that, despite differences in the genetic and metabolic background, in both cases extensive stay of erythrocytes in storage leads reduced antioxidant capacity, decreased levels of G6PD activity, followed by increased levels of glycated Hb and osmotic hemolysis (intra-parameter relationships).

Regarding the network analysis of inter-parameter associations several clusters were identified and those with higher density of connections and greater impact to the consistency of the hematological network included parameters related to a) in-bag hemolysis, susceptibility to hemolysis, 2,3-biphosphoglycerate (2,3-BPG) and dehydroascorbate, b) G6PD activity, c) fatty/bile acids, d) redox (e.g. ROS, antioxidant capacity etc.) and e) hematological/physiological features (e.g. MCV, MFI etc.). A subsequent analysis on the G6PD activity subnetwork highlighted amino acids and 2-OH-glutarate as compounds positively correlated to G6PD, while parameters related to PPP, bile acids, oxidized lipids and monosaturated fatty acids were negatively associated with G6PD. At last, of great interest was ta subnetwork that emphasized in the complexity of the hemolysis phenotype, which as described in their work it is a multivariate phenomenon that is possibly affected by the donor’s profile, besides the effect of storage itself.

Overall, this work pointed out that even though there are some significant differences between the biological profiles of G6PD$^-$ and control donors, the effect of storage was quite similar in both cases. Additionally, the authors highlighted the multivariate character of hemolysis, while they addressed some crucial parameters that contribute to that. Moreover, it is important to mention that this work was a starting point for our work, as data regarding the biochemical and physiological profile of all donors were used as part of our final dataset.

Kowsar *et al.* 2020 [36]

The authors of this review studied the hematological and biochemical characteristics of COVID-19 non-survivor subjects. For this project data related to COVID-19 cases were extracted from the literature, excluding under 19 years old cases. The final dataset consisted of 14,359 cases that survived from the infection and 4,655 non-survivors. All extracted data converted in such way, so that they have the same format and units. From each data source information regarding the country, sex, age, blood parameters and pre-existing health issues were collected. The final dataset was subjected to normality check analysis using Anderson-Darling test. Network analysis and any further meta-analysis were conducted in PAST and META-MAR applications, respectively. On top of that, several statistical methods were applied to check the heterogenicity of the dataset (e.g. Cochran’s Q test). Regarding network analysis Pearson similarity algorithm was used as a method for defining relationships between different entities.

The outcome from gathering blood test results showed an increased number of neutrophils and white blood cells on patients that did not survive from the infection, which was not the case for the number of lymphocytes and platelets. They, also, mentioned several complications from which non-survivors suffered due to COVID-19 infection, such as acute kidney injury, heart failure or septic shock. Results from the meta-analysis of collected data pointed out that mortality rates increase as the age of the diseased increases. Prior health complications, such as diabetes or cerebrovascular disease, have a similar effect. These findings were further supported by the correlation networks regarding hematological parameters. More specifically, platelets and lymphocytes had a correlation of 0.72 with COVID-19, while neutrophils were associated with evidence of infection by the virus with a rate of 0.93. At last, correlation networks regarding patients' profile and evidence of infection, also, confirmed results of the meta-analysis since age and prior health issues had a Pearson's coefficient of 0.79.

Goodman *et al.* 2013 [37]

In this review the authors describe the proteomics and interactomics of human erythrocytes. At first, they state the total number of proteins that have been characterized in RBCs. Following that, they extensively reported the methods and findings of previous studies that led to the current data about the RBC proteomics. Moreover, they comment about the changes that occur in transmembrane proteins of erythrocytes during several health issues, such as malaria, Alzheimer's disease, or chronic kidney disease. On top of that, they describe the current and previous findings regarding the proteomic analysis of individuals that suffered from sickle cell disease (SCD). SCD is a recessive autosomal disease that occurs due to a point mutation in the beta chain of Hb. Because of this mutation Hb polymerize in such way, so that erythrocytes eventually take a characteristic sickled shape. In its severe form SCD leads to vasoocclusive crises that, if not handled properly, could have serious effects on the survival of the patient. After that, they state several findings regarding measurements of hematological parameters and proteomics of SCD patients. More specifically, they mention that proteins related to repair mechanism, lipid raft components, membrane skeletal proteins and radical oxygen scavengers play an important role in the development of the disease.

In the attempt of finding potential evidence about the existence of biomarkers related to SCD, the authors of this work proceed with the construction of (Protein-protein interaction) PPI network. Data regarding the nodes of the graph collected experimentally, while information about the edges was retrieved from related databases. The significance of the relationship between two components was defined by Spearman's coefficient. According to the study, edges with Spearman's coefficient less than 0.3 were subtracted from any further analysis to minimize the chance of introducing false positive interactions to the network. Following the construction of the PPI network, the application of several graph-analytics methods took place. Three centrality metrics were estimated for the entities of the graph, namely, betweenness, degree and closeness centrality. The authors stated the significance, performance, and drawbacks of each metric. On top of that, they report the most significant findings, as they derived from centrality analysis, regarding proteins correlated with SCD.

At last, they refer to other approaches of previous works that attempted to perform community detection using Voronoi diagrams in graphs. According to the authors, Voronoi diagrams are used to cluster nodes according to the distance of the member of a predefined subset from the center of the cluster, which in this case they are called Voronoi sites. To further expand the findings of this study they applied the method for the case of the PPI network of patients with SCD. That said, for the purpose of this analysis, proteins directly affected by the SCD were used as Voronoi sites, while components of the graph that belong to a cluster are considerably more likely to be affected by the SCD-affected protein. In total, 22 proteins were characterized as Voronoi sites, meaning they altered by SCD, and 16 clusters were marked through this analysis.

De León *et al.* 2014 [38]

In this work the authors developed a vascular network model to illustrate molecular paths related to atherogenesis-oriented processes. They applied their method in human and murine datasets. At first, they address some crucial risk factors that contribute to the development of atherogenesis (e.g. such as cigarette smoking). Following that, to construct the Vascular Inflammatory Processes Network (V-IPN), they used available data from literature and data concerning computationally predicted associations regarding health issues, such as inflammation or vascular disease. On top of that they developed a computational reasoning method, called Reverse Causal Reasoning (RCR), to preprocess the concatenated dataset, in a sense of finding statistically significant hypotheses regarding the graph entities. Once they retrieved and preprocessed all retrieved data, they proceed with manually reviewing and refining the graph model, so that only edges related to vascular inflammation processes would remain. On top of that, they tried to further evaluate the integrity of the graph model by utilizing information from Gene Expression Omnibus (GEO).

Using this RCR method the authors were able to explore the graph entities and reveal potential associations that were not stated in the sources from which they retrieved the original data. The V-IPN could be divided into six communities, according to the key pathological aspects of vascular disease. Five of those clusters were related to primary atherogenic vascular-oriented mechanisms, while the sixth cluster was concerning events occurring during atherosclerotic lesions. Moreover, despite the discrimination of these subnetworks, there were, still, some inter-community connections between components of different clusters. On top of that, by utilizing once more the RCR method the authors could evaluate the significance of the connections between the genes of each community and find those with more significantly changed expression levels (increased or decreased). Through this process they identified common and discriminative factors in human and murine cases through in different stages (early or progressed) of inflammation, vascular disease and atherogenesis.

Amanatidou *et al.* 2020 [39]

In this project the authors developed a method for investigating the PPI network of proteins related to Blood-cell Autoimmune Diseases (BLADs). The proposed graph model that will be described below includes information regarding PPI and terminologies concerning autoimmune diseases. At first, some of the most known BLADs are reported, while it is briefly explained that most autoimmune diseases are multifactorial. The aim of this study was to state that reporting proteins directly related to disease-associated proteins could give insight about the nature of the disease.

To construct the PPI network and BLADs, they, initially, collected data regarding autoimmune diseases from ICD-10, Orphanet, Mesh and NIH-NHLBI databases. This dataset was further evaluated to avoid listing duplicates with alternative names. Data concerning proteins associated with the recorded BLADs were retrieved from OMIM and DisGENET databases. Following that, the authors used IntAct to find proteins experimentally correlated with the collected BLADs-associated proteins. The construction of the network was conducted in Cytoscape. Topological and functional enrichment analysis were the first steps for exploring the graph entities. On top of that, they performed centrality analysis, by computing betweenness, degree and closeness centrality. Moreover, proteins with high scores in functional enrichment analysis and/or the centrality analysis were further investigated by adding more GO terms. The final list of candidate proteins related to BLADs derived from the intersection of proteins with top score in centrality analysis, proteins related to BLADs and proteins with significant GO terms about autoimmune disease.

Detecting communities was the next step of this analysis and clusters with more than two components were subjected to further evaluation. Throughout this process, they identified for each cluster proteins directly correlated to BLADs, proteins with at least on common 1st neighbor, proteins whose 1st neighbors are connected and proteins with at least one common 2nd neighbor. In total 32 out of the 42 clusters were investigated (the other 10 had only two components). Hub proteins, as well as .the rest of candidate BLADs-related proteins of each cluster were also subjected gene expression analysis using the GEO2R function of GEO. The threshold of p-value or excluding insignificant results was set at 0.05. By combining the results of this analysis the authors managed to discriminate 14 proteins that are most likely related to one or more BLADs, 7 of which they were confirmed in the literature. In addition, with the cluster analysis they were able to distinguish 17 more proteins that play a connecting role between clusters of different BLADs, indicating possible interconnections between them.

Marzec *et al.* 2021 [40]

The authors of this work focused on a sex-dependent aspect of the storage effect in the membrane of stored RBCs. Their dataset consisted of venous blood from 24 men and 24 women of varied ages. To collect enough data they performed weekly samplings up to the 42nd day, starting from the day they retrieved the blood samples (fresh RBCs). Throughout the analysis, they observed significant differences between male and female blood samples regarding RBC's lipidomics. More specifically, levels of cholesterol and triglycerides were more elevated in erythrocytes from female donors. That applies to the values of free iron, as well, indicating a higher level of hemolysis. Results regarding several hematological parameters (e.g. MCV, hematocrit etc.) confirmed that values of membrane's deformability were greater in males, though in both sexes there was a decreasing tendency. Alterations in RBC's metabolism due to storage effect were found to be sex independent. To check the significance of their results, the authors applied one-way ANOVA followed by post-hoc tests depending on the parameters they examined (e.g. for estimating the significance from the analysis of biochemical parameters Tukey's post hoc was performed after the one-way ANOVA).

Table 2. Similarities and differences of related work with our work

Work	Similarities	Differences
Antonelou *et al.* 2018	1. Both projects handle the same biological problem. 2.. Moreover, the dataset that was used in the work Antonelou *et al.* was part of the final dataset of our project. 3. Network analysis was conducted in both cases. Pearson's coefficient was used as a metric to define the significance of relationships in intra- and inter-parameters association of connected components.	1. Centrality analysis was not applied in the Antonelou *et al* work 2. Their approach of the network analysis was quite static, and it was conducted in Cytoscape 3.Besides the experimental data, external sources were used, in our work, for the enrichment of the hematological markers network 3. No use of any graph database system was made in their project.
Kowsar *et al.* 2020	1. Both works are related with the analysis of health-related issues using network analysis 2. For defining the significance of the relationships Pearson's similarity algorithm was used in both cases	1. The main biological aspect is different in the two projects 2. No graph analytics methods are utilized in the work of Kowsar *et al* 3. Also, they did not use any graph database system for the construction of the network 4. A more statistical approach took place for the meta-analysis of the resulting graphs.
Goodman *et al.* 2013	Both projects are related to the analysis of proteomics and interactomics of erythrocytes through graph analytics	1. Goodman *et al*. focus mostly on the analysis of PPI networks and reviews previous works in that matter 2. To construct the PPI network they use Spearman's similarity algorithm (Pearson's similarity algorithm was used in our project) 3. For detecting communities the Voronoi diagrams were utilized, while in our case was made use of several algorithms (Louvain method, Strongly Connected Components etc.)

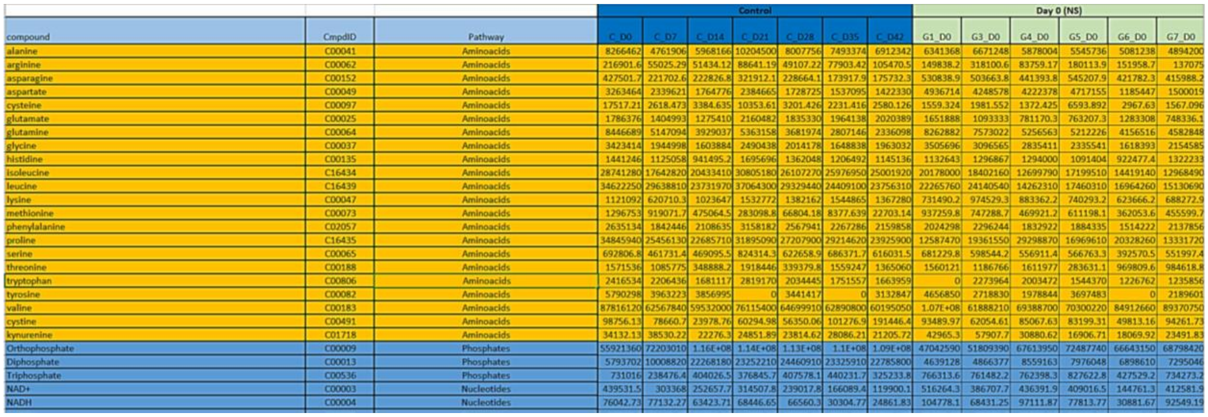
		4. Since their project is a review of the current methods and knowledge there is not a novelty in their results 5. They do not make use of any graph database system or interface to manipulate the constructed graph.
De León *et al*. 2014	1. Both projects focus on blood-related health issues 2. In both cases community detections algorithms/methods are utilized	1. In De León *et al*. the final dataset consists only of predicted and literature-related data 2. They mostly focus on the development of the RCR method for predicting and refining relationships between graph entities and so much on applying graph analytics 3. Their approach does not utilize any graph database system and is quite static.
Amanatidou *et al*. 2020	1. Both projects focus on graph analytics of health issues 2. Both projects use experimentally and computationally verified data	1. In Amanatidou *et al* the biological aspect is not related to RBCs 2. A different statistical approach is applied in their case. The relationships of the connected components is based on the functional enrichment analysis, while in our case similarity algorithms provided by the graph database are utilized 3. Though the graph-related analysis they performed was extensive, it is quite static, and they do not make use of any graph database system. The construction of their graph was made in Cytoscape.
Marzec *et al*. 2021	The main biological aspect of both projects is the effect of storage in RBCs	1. Marzec *et al.* use simpler statistical approaches (e.g. ANOVA) and no graph analytics 2. Though they used more donors, the parameters they analyzed were fewer. Also, it consisted only of experimental data.

3. DATA COLLECTION

3.1. Experimental Data

Metabolic and Physiological Data

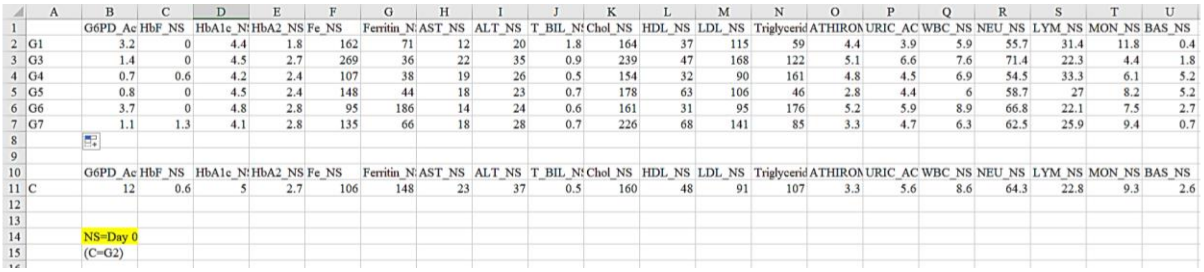
For this Thesis real user data as well as data from external sources were used. More specifically, experimental data – concerning the metabolic, physiological, proteomic and vesicular profile – retrieved from six different (G1, G3, G4, G5, G6, G7) G6PD deficient individuals (G6PD$^-$) and one control (C/G2) individual (G6PD$^+$). Regarding the metabolic and physiological data, each donor participated in 7 weekly samplings based on the storage stage of his/her RBCs (D0, D7, D14, D21, D28, D35 and D42). For each donor the concentration of 295 distinct metabolites in RBCs from several metabolic pathways (e.g. glycolysis, pentose phosphate pathway etc.) was estimated, while for the case of physiological data 83 parameters related to the physiology of RBCs, such as cell's fragility and reactive oxygen species (ROS), were measured.



			Control							Day 0 (NS)					
compound	CmpdID	Pathway	C_D0	C_D7	C_D14	C_D21	C_D28	C_D35	C_D42	G1_D0	G3_D0	G4_D0	G5_D0	G6_D0	G7_D0
alanine	C00041	Aminoacids	8266462	4761906	5968166	10204500	8007756	7493374	6912342	6341368	6671248	5878004	5545736	5081238	4894200
arginine	C00062	Aminoacids	216901.6	55025.29	51434.12	88641.19	49107.22	77903.42	105470.5	149838.2	318100.6	83759.17	180113.9	151958.7	137075
asparagine	C00152	Aminoacids	427501.7	221702.6	222826.8	321912.1	228664.1	173917.9	175732.3	530838.9	503663.8	441393.8	545207.9	421782.3	415988.2
aspartate	C00049	Aminoacids	3263464	2339621	1764776	2384665	1728725	1537095	1422330	4936714	4248578	4222378	4717155	1185447	1500019
cysteine	C00097	Aminoacids	17517.21	2618.473	3384.635	10353.61	3201.426	2231.416	2580.126	1559.324	1981.552	1372.425	6593.892	2967.63	1567.096
glutamate	C00025	Aminoacids	1786376	1404993	1275410	2160482	1835330	1964138	2020389	1651888	1093333	781170.3	763207.3	1283308	748336.1
glutamine	C00064	Aminoacids	8446689	5147094	3929037	5363158	3681974	2807146	2336098	8262882	7573022	5256563	5212226	4156516	4582848
glycine	C00037	Aminoacids	3423414	1944998	1603884	2490438	2014178	1648838	1963032	3505696	3096565	2835411	2335541	1618393	2154585
histidine	C00135	Aminoacids	1441246	1125058	941495.2	1695696	1362048	1206492	1145136	1132643	1296867	1294000	1091404	922477.4	1322233
isoleucine	C16434	Aminoacids	28741280	17642820	20433410	30805180	26107270	25976950	25001920	20178000	18402160	12699790	17199510	14419140	12968490
leucine	C16439	Aminoacids	34622250	29638810	23731970	37064300	29329440	24409100	23756310	22265760	24140540	14262310	17460310	16964260	15130690
lysine	C00047	Aminoacids	1121092	620710.3	1023647	1532772	1382162	1544865	1367280	731490.2	974529.3	883362.2	740293.2	623666.2	688272.9
methionine	C00073	Aminoacids	1296753	919071.7	475064.5	283098.8	66804.18	8377.639	22703.14	937259.8	747288.7	469921.2	611198.1	362053.6	455599.7
phenylalanine	C02057	Aminoacids	2635134	1842446	2108635	3158182	2567941	2267286	2159858	2024298	2296244	1832922	1884335	1514222	2137856
proline	C16435	Aminoacids	34845940	25456130	22685710	31895090	27207900	29214620	23925900	12587470	19361550	29298870	16969610	20328260	13331720
serine	C00065	Aminoacids	692806.8	461731.4	469095.5	824314.3	622658.9	686371.7	616031.5	681229.8	598544.2	556911.4	566763.3	392570.5	551997.4
threonine	C00188	Aminoacids	1571536	1085775	348888.2	1918446	339379.8	1559247	1365060	1560121	1186766	1611977	283631.1	969809.6	984618.8
tryptophan	C00806	Aminoacids	2416534	2206436	1681117	2819170	2034445	1751557	1663959	0	2273964	2003472	1544370	1226762	1235856
tyrosine	C00082	Aminoacids	5790298	3963223	3856995	0	3441417	0	3132847	4656850	2718830	1978844	3697483	0	2189601
valine	C00183	Aminoacids	87816120	62567840	59532000	76115400	64699910	62890800	60195050	1.07E+08	61888210	69388700	70300220	84912660	89370750
cystine	C00491	Aminoacids	98756.13	78660.7	23978.76	60294.98	56350.06	101276.9	191446.4	93489.97	62054.61	85067.63	83199.31	49813.16	94261.73
kynurenine	C01718	Aminoacids	34132.13	38530.22	22276.3	24851.89	23814.62	28086.21	21205.72	42965.3	57907.7	30880.62	16906.71	18069.92	23491.83
Orthophosphate	C00009	Phosphates	55921360	72203010	1.16E+08	1.14E+08	1.13E+08	1.1E+08	1.09E+08	47042590	51809390	67613950	72487740	66643150	68798420
Diphosphate	C00013	Phosphates	5793702	10008820	22268180	23252210	24460910	23325910	22785800	4639128	4866377	8559163	7976048	6898610	7295046
Triphosphate	C00536	Phosphates	731016	238476.4	404026.5	376845.7	407578.1	440231.7	325233.8	766313.6	761482.2	762398.3	827622.8	427529.2	734273.2
NAD+	C00003	Nucleotides	439531.5	303368	252657.7	314507.8	239017.8	166089.4	119900.1	516264.3	386707.7	436391.9	409016.5	144761.3	412581.9
NADH	C00004	Nucleotides	76042.73	77132.27	63423.71	68446.65	66560.3	30304.77	24861.83	104778.1	68431.25	97111.87	77813.77	30881.67	92549.19

Figure 6. Sample of metabolic data. In this figure the abundances of all amino acids in RBCs of G6PD$^+$ donor, as well as the first sampling (D0) of all G6PD$^-$ donors are shown.

In Figure 6 a sample of the metabolic data is presented. For each metabolite/compound information about the name, the ID in KEGG database, the pathway in which is a part of and the abundances in RBCs of G6PD$^-$ and G6PD$^+$ donors were collected.



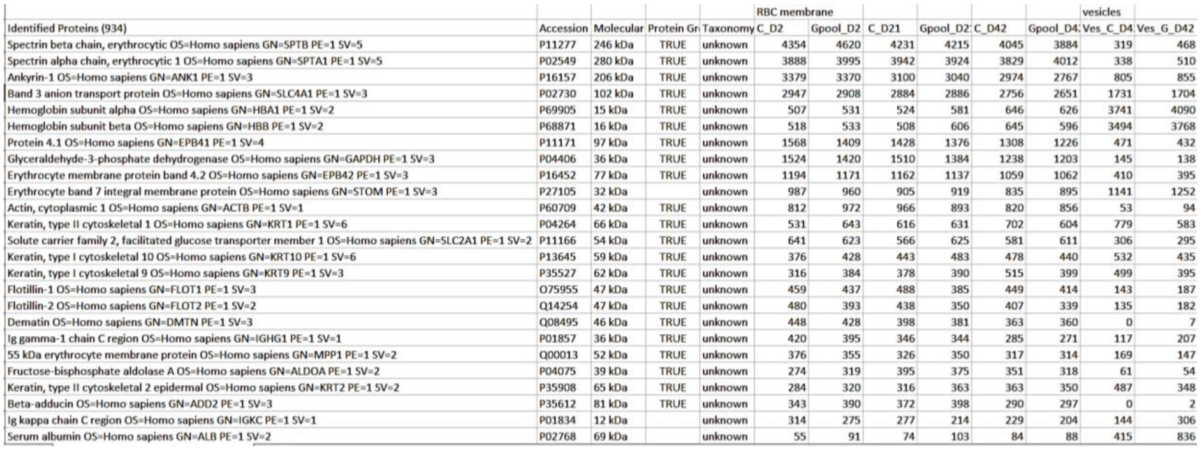
	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
1		G6PD_Ac	HbF_NS	HbA1c_N:	HbA2_NS	Fe_NS	Ferritin_N:	AST_NS	ALT_NS	T_BIL_N:	Chol_NS	HDL_NS	LDL_NS	Triglycerid	ATHIRON	URIC_AC	WBC_NS	NEU_NS	LYM_NS	MON_NS	BAS_NS
2	G1	3.2	0	4.4	1.8	162	71	12	20	1.8	164	37	115	59	4.4	3.9	5.9	55.7	31.4	11.8	0.4
3	G3	1.4	0	4.5	2.7	269	36	22	35	0.9	239	47	168	122	5.1	6.6	7.6	71.4	22.3	4.4	1.8
4	G4	0.7	0.6	4.2	2.4	107	38	19	26	0.5	154	32	90	161	4.8	4.5	6.9	54.5	33.3	6.1	5.2
5	G5	0.8	0	4.5	2.4	148	44	18	23	0.7	178	63	106	46	2.8	4.4	6	58.7	27	8.2	5.2
6	G6	3.7	0	4.8	2.8	95	186	14	24	0.6	161	31	95	176	5.2	5.9	8.9	66.8	22.1	7.5	2.7
7	G7	1.1	1.3	4.1	2.8	135	66	18	28	0.7	226	68	141	85	3.3	4.7	6.3	62.5	25.9	9.4	0.7
8																					
9																					
10		G6PD_Ac	HbF_NS	HbA1c_N:	HbA2_NS	Fe_NS	Ferritin_N:	AST_NS	ALT_NS	T_BIL_N:	Chol_NS	HDL_NS	LDL_NS	Triglycerid	ATHIRON	URIC_AC	WBC_NS	NEU_NS	LYM_NS	MON_NS	BAS_NS
11	C	12	0.6	5	2.7	106	148	23	37	0.5	160	48	91	107	3.3	5.6	8.6	64.3	22.8	9.3	2.6
12																					
13																					
14		NS=Day 0																			
15		(C=G2)																			

Figure 7. Sample of physiological data. In this figure a sample of physiological data is displayed. The first part concerns G6PD$^-$ donors, while the second part shows information about the physiological profile of the control donor.

In Figure 7 a sample of the physiological data is shown. More specifically, each row represents the physiological profile of RBCs of each donor, while columns state the abundances of each physiological parameter in RBCs and the sampling stage. It is important to mention that in this case a different reference code – compared to metabolic data – for the description of sampling stages was used. For instance, instead of referring to the first sampling as D0, the term NS (no storage) or D2 was used. Therefore, as described in the section “Data pre-processing and curation” some modification had been made to adapt a common reference code.

Proteomic Data

For the case of proteomic data three pooled − storage based − samplings took place for both G6PD$^-$ and G6PD$^+$ donors (D0, D21 and D42), while for the collection of vesicular data, donors participated only in one sampling at the 42nd day of the experiment. The initial proteomic dataset consisted of 934 unique proteins. For each protein information about the official protein name, the gene it is expressed from, the molecular weight, the accession number (AC) in UniProtKB/SwissProt, as well as the abundances in RBCs and vesicles of G6PD$^-$ and G6PD$^+$ donors were collected. Measurements of G6PD$^-$ donors are denoted with the extension “Gpool” or “_G_” in their name, while the control donor is marked as “C_” (Figure 8).

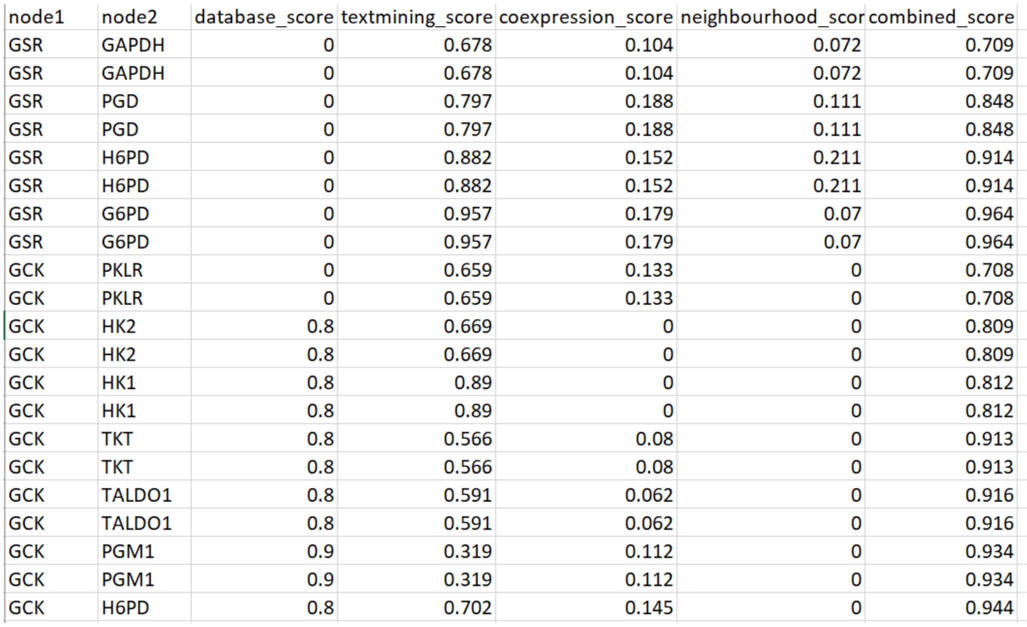


					RBC membrane						vesicles	
Identified Proteins (934)	Accession	Molecular	Protein Gr	Taxonomy	C_D2	Gpool_D2	C_D21	Gpool_D2	C_D42	Gpool_D4	Ves_C_D4	Ves_G_D42
Spectrin beta chain, erythrocytic OS=Homo sapiens GN=SPTB PE=1 SV=5	P11277	246 kDa	TRUE	unknown	4354	4620	4231	4215	4045	3884	319	468
Spectrin alpha chain, erythrocytic 1 OS=Homo sapiens GN=SPTA1 PE=1 SV=5	P02549	280 kDa	TRUE	unknown	3888	3995	3942	3924	3829	4012	338	510
Ankyrin-1 OS=Homo sapiens GN=ANK1 PE=1 SV=3	P16157	206 kDa	TRUE	unknown	3379	3370	3100	3040	2974	2767	805	855
Band 3 anion transport protein OS=Homo sapiens GN=SLC4A1 PE=1 SV=3	P02730	102 kDa	TRUE	unknown	2947	2908	2884	2886	2756	2651	1731	1704
Hemoglobin subunit alpha OS=Homo sapiens GN=HBA1 PE=1 SV=2	P69905	15 kDa	TRUE	unknown	507	531	524	581	646	626	3741	4090
Hemoglobin subunit beta OS=Homo sapiens GN=HBB PE=1 SV=2	P68871	16 kDa	TRUE	unknown	518	533	508	606	645	596	3494	3768
Protein 4.1 OS=Homo sapiens GN=EPB41 PE=1 SV=4	P11171	97 kDa	TRUE	unknown	1568	1409	1428	1376	1308	1226	471	432
Glyceraldehyde-3-phosphate dehydrogenase OS=Homo sapiens GN=GAPDH PE=1 SV=3	P04406	36 kDa	TRUE	unknown	1524	1420	1510	1384	1238	1203	145	138
Erythrocyte membrane protein band 4.2 OS=Homo sapiens GN=EPB42 PE=1 SV=3	P16452	77 kDa	TRUE	unknown	1194	1171	1162	1137	1059	1062	410	395
Erythrocyte band 7 integral membrane protein OS=Homo sapiens GN=STOM PE=1 SV=3	P27105	32 kDa		unknown	987	960	905	919	835	895	1141	1252
Actin, cytoplasmic 1 OS=Homo sapiens GN=ACTB PE=1 SV=1	P60709	42 kDa	TRUE	unknown	812	972	966	893	820	856	53	94
Keratin, type II cytoskeletal 1 OS=Homo sapiens GN=KRT1 PE=1 SV=6	P04264	66 kDa	TRUE	unknown	531	643	616	631	702	604	779	583
Solute carrier family 2, facilitated glucose transporter member 1 OS=Homo sapiens GN=SLC2A1 PE=1 SV=2	P11166	54 kDa	TRUE	unknown	641	623	566	625	581	611	306	295
Keratin, type I cytoskeletal 10 OS=Homo sapiens GN=KRT10 PE=1 SV=6	P13645	59 kDa	TRUE	unknown	376	428	443	483	478	440	532	435
Keratin, type I cytoskeletal 9 OS=Homo sapiens GN=KRT9 PE=1 SV=3	P35527	62 kDa	TRUE	unknown	316	384	378	390	515	399	499	395
Flotillin-1 OS=Homo sapiens GN=FLOT1 PE=1 SV=3	O75955	47 kDa	TRUE	unknown	459	437	488	385	449	414	143	187
Flotillin-2 OS=Homo sapiens GN=FLOT2 PE=1 SV=2	Q14254	47 kDa	TRUE	unknown	480	393	438	350	407	339	135	182
Dematin OS=Homo sapiens GN=DMTN PE=1 SV=3	Q08495	46 kDa	TRUE	unknown	448	428	398	381	363	360	0	7
Ig gamma-1 chain C region OS=Homo sapiens GN=IGHG1 PE=1 SV=1	P01857	36 kDa	TRUE	unknown	420	395	346	344	285	271	117	207
55 kDa erythrocyte membrane protein OS=Homo sapiens GN=MPP1 PE=1 SV=2	Q00013	52 kDa	TRUE	unknown	376	355	326	350	317	314	169	147
Fructose-bisphosphate aldolase A OS=Homo sapiens GN=ALDOA PE=1 SV=2	P04075	39 kDa	TRUE	unknown	274	319	395	375	351	318	61	54
Keratin, type II cytoskeletal 2 epidermal OS=Homo sapiens GN=KRT2 PE=1 SV=2	P35908	65 kDa	TRUE	unknown	284	320	316	363	363	350	487	348
Beta-adducin OS=Homo sapiens GN=ADD2 PE=1 SV=3	P35612	81 kDa	TRUE	unknown	343	390	372	398	290	297	0	2
Ig kappa chain C region OS=Homo sapiens GN=IGKC PE=1 SV=1	P01834	12 kDa		unknown	314	275	277	214	229	204	144	306
Serum albumin OS=Homo sapiens GN=ALB PE=1 SV=2	P02768	69 kDa		unknown	55	91	74	103	84	88	415	836

Figure 8. Sample of proteomic data.

3.2. External Data Sources

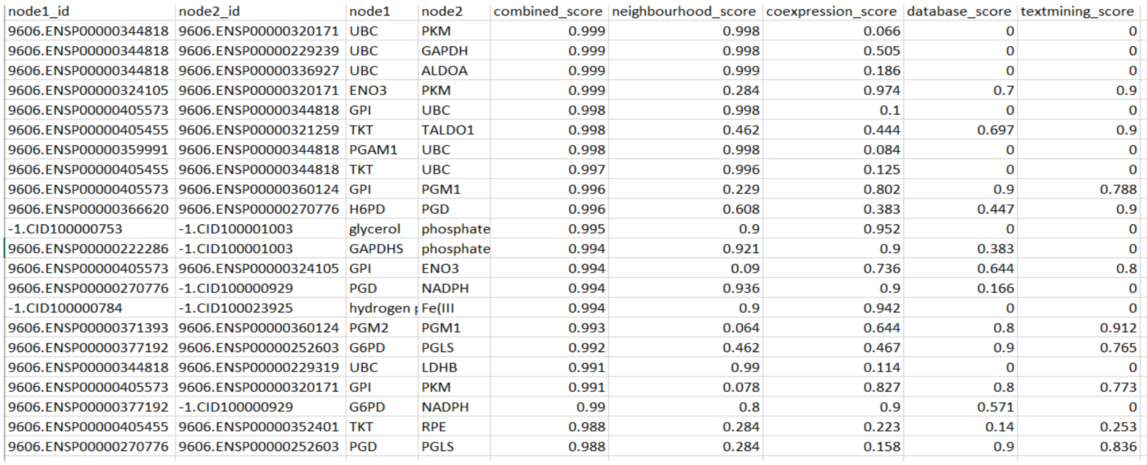
To enrich the size of the final dataset several open access databases with information relative to G6PD were used. More specifically, data about protein interactions between G6PD and other proteins were retrieved from the API (Application Programming Interface) of String database. String is a database of experimentally proven and predicted interactions − physical and functional − between proteins [41]. For this analysis only functional – direct and indirect – relationships of G6PD with other proteins were collected. In Figure 9 a sample of the dataset that retrieved from String is shown. Each row presents information about protein interactions. For each interaction, the protein interactors (node1 and node2) and several types of metrics, such as the prediction score applied by the database, the text mining score of the interaction, the co-expression and neighborhood score of the two interactors and the combined score are recorded.



node1	node2	database_score	textmining_score	coexpression_score	neighbourhood_scor	combined_score
GSR	GAPDH	0	0.678	0.104	0.072	0.709
GSR	GAPDH	0	0.678	0.104	0.072	0.709
GSR	PGD	0	0.797	0.188	0.111	0.848
GSR	PGD	0	0.797	0.188	0.111	0.848
GSR	H6PD	0	0.882	0.152	0.211	0.914
GSR	H6PD	0	0.882	0.152	0.211	0.914
GSR	G6PD	0	0.957	0.179	0.07	0.964
GSR	G6PD	0	0.957	0.179	0.07	0.964
GCK	PKLR	0	0.659	0.133	0	0.708
GCK	PKLR	0	0.659	0.133	0	0.708
GCK	HK2	0.8	0.669	0	0	0.809
GCK	HK2	0.8	0.669	0	0	0.809
GCK	HK1	0.8	0.89	0	0	0.812
GCK	HK1	0.8	0.89	0	0	0.812
GCK	TKT	0.8	0.566	0.08	0	0.913
GCK	TKT	0.8	0.566	0.08	0	0.913
GCK	TALDO1	0.8	0.591	0.062	0	0.916
GCK	TALDO1	0.8	0.591	0.062	0	0.916
GCK	PGM1	0.9	0.319	0.112	0	0.934
GCK	PGM1	0.9	0.319	0.112	0	0.934
GCK	H6PD	0.8	0.702	0.145	0	0.944

Figure 9. Sample of data retrieved from String database.

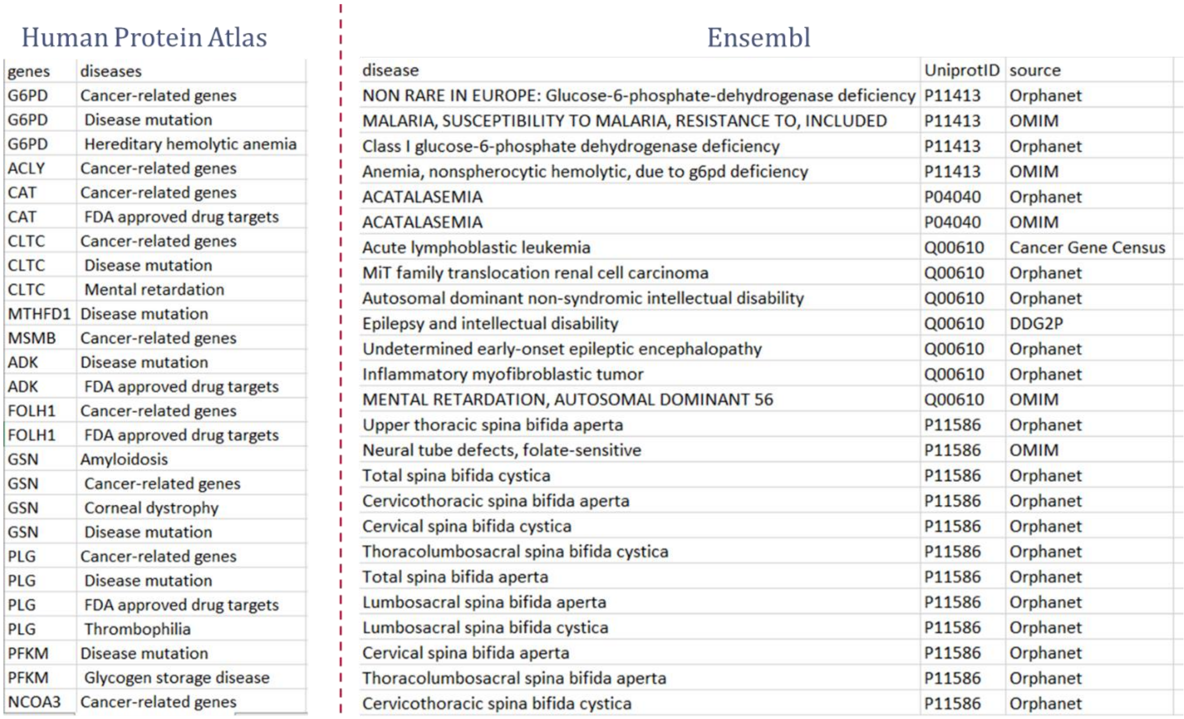
Additionally, data regarding protein-chemical or chemical-chemical interactions retrieved from STITCH database. To understand the type of interaction between two interactors, one needs to record the identity of each interactor. Therefore, all chemicals/compounds have a unique reference code that has the initials "CID", while all proteins are identified by their protein id, which has the initials "ENSP". The confidence of each relationship derives from the combination of several metrics, such as the co-expression levels of the two interactors, the text mining score or the prediction score that was applied by the database itself. A sample of the data that were retrieved from STITCH database is available in Figure 10 [42]. In total, 241 interactions were collected from String database and 453 additional interaction retrieved from STITCH database.



node1_id	node2_id	node1	node2	combined_score	neighbourhood_score	coexpression_score	database_score	textmining_score
9606.ENSP00000344818	9606.ENSP00000320171	UBC	PKM	0.999	0.998	0.066	0	0
9606.ENSP00000344818	9606.ENSP00000229239	UBC	GAPDH	0.999	0.998	0.505	0	0
9606.ENSP00000344818	9606.ENSP00000336927	UBC	ALDOA	0.999	0.999	0.186	0	0
9606.ENSP00000324105	9606.ENSP00000320171	ENO3	PKM	0.999	0.284	0.974	0.7	0.9
9606.ENSP00000405573	9606.ENSP00000344818	GPI	UBC	0.998	0.998	0.1	0	0
9606.ENSP00000405455	9606.ENSP00000321259	TKT	TALDO1	0.998	0.462	0.444	0.697	0.9
9606.ENSP00000359991	9606.ENSP00000344818	PGAM1	UBC	0.998	0.998	0.084	0	0
9606.ENSP00000405455	9606.ENSP00000344818	TKT	UBC	0.997	0.996	0.125	0	0
9606.ENSP00000405573	9606.ENSP00000360124	GPI	PGM1	0.996	0.229	0.802	0.9	0.788
9606.ENSP00000366620	9606.ENSP00000270776	H6PD	PGD	0.996	0.608	0.383	0.447	0.9
-1.CID100000753	-1.CID100001003	glycerol	phosphate	0.995	0.9	0.952	0	0
9606.ENSP00000222286	-1.CID100001003	GAPDHS	phosphate	0.994	0.921	0.9	0.383	0
9606.ENSP00000405573	9606.ENSP00000324105	GPI	ENO3	0.994	0.09	0.736	0.644	0.8
9606.ENSP00000270776	-1.CID100000929	PGD	NADPH	0.994	0.936	0.9	0.166	0
-1.CID100000784	-1.CID100023925	hydrogen p	Fe(III	0.994	0.9	0.942	0	0
9606.ENSP00000371393	9606.ENSP00000360124	PGM2	PGM1	0.993	0.064	0.644	0.8	0.912
9606.ENSP00000377192	9606.ENSP00000252603	G6PD	PGLS	0.992	0.462	0.467	0.9	0.765
9606.ENSP00000344818	9606.ENSP00000229319	UBC	LDHB	0.991	0.99	0.114	0	0
9606.ENSP00000405573	9606.ENSP00000320171	GPI	PKM	0.991	0.078	0.827	0.8	0.773
9606.ENSP00000377192	-1.CID100000929	G6PD	NADPH	0.99	0.8	0.9	0.571	0
9606.ENSP00000405455	9606.ENSP00000352401	TKT	RPE	0.988	0.284	0.223	0.14	0.253
9606.ENSP00000270776	9606.ENSP00000252603	PGD	PGLS	0.988	0.284	0.158	0.9	0.836

Figure 10. Sample of data retrieved from STITCH database.

Data, regarding diseases related to G6PD or proteins closely associated with it, retrieved from Human Protein Atlas (HPA) and Ensembl. HPA contains information about most human proteins and gives insight about the tissue their expressed (e.g. brain, blood etc.), the method they were extracted (e.g. spectrometry, antibody-based method etc.) and potential pathogenicity status [43]. For this analysis, data related to the pathogenicity status of G6PD, and the most statistically significant proteins of the hematological markers network were retrieved from the API server of HPA (Figure 11). Ensembl is an open genome browser with genomic information about vertebrates. Specifically, each human gene is fully annotated, and data related to chromosome position, variations, phenotypes, diseases, and external sources are available [44]. In this case, too, data about G6PD and the most statistically significant proteins of the network were collected. In more detail for each protein information, about its phenotype or implication in disease, as well as the external source from which the information about the disease was provided, were retrieved (Figure 11). In total, 39 records about diseases were collected from Ensembl database and 27 additional diseases were retrieved from HPA.



Human Protein Atlas

genes	diseases
G6PD	Cancer-related genes
G6PD	Disease mutation
G6PD	Hereditary hemolytic anemia
ACLY	Cancer-related genes
CAT	Cancer-related genes
CAT	FDA approved drug targets
CLTC	Cancer-related genes
CLTC	Disease mutation
CLTC	Mental retardation
MTHFD1	Disease mutation
MSMB	Cancer-related genes
ADK	Disease mutation
ADK	FDA approved drug targets
FOLH1	Cancer-related genes
FOLH1	FDA approved drug targets
GSN	Amyloidosis
GSN	Cancer-related genes
GSN	Corneal dystrophy
GSN	Disease mutation
PLG	Cancer-related genes
PLG	Disease mutation
PLG	FDA approved drug targets
PLG	Thrombophilia
PFKM	Disease mutation
PFKM	Glycogen storage disease
NCOA3	Cancer-related genes

Ensembl

disease	UniprotID	source
NON RARE IN EUROPE: Glucose-6-phosphate-dehydrogenase deficiency	P11413	Orphanet
MALARIA, SUSCEPTIBILITY TO MALARIA, RESISTANCE TO, INCLUDED	P11413	OMIM
Class I glucose-6-phosphate dehydrogenase deficiency	P11413	Orphanet
Anemia, nonspherocytic hemolytic, due to g6pd deficiency	P11413	OMIM
ACATALASEMIA	P04040	Orphanet
ACATALASEMIA	P04040	OMIM
Acute lymphoblastic leukemia	Q00610	Cancer Gene Census
MiT family translocation renal cell carcinoma	Q00610	Orphanet
Autosomal dominant non-syndromic intellectual disability	Q00610	Orphanet
Epilepsy and intellectual disability	Q00610	DDG2P
Undetermined early-onset epileptic encephalopathy	Q00610	Orphanet
Inflammatory myofibroblastic tumor	Q00610	Orphanet
MENTAL RETARDATION, AUTOSOMAL DOMINANT 56	Q00610	OMIM
Upper thoracic spina bifida aperta	P11586	Orphanet
Neural tube defects, folate-sensitive	P11586	OMIM
Total spina bifida cystica	P11586	Orphanet
Cervicothoracic spina bifida aperta	P11586	Orphanet
Cervical spina bifida cystica	P11586	Orphanet
Thoracolumbosacral spina bifida cystica	P11586	Orphanet
Total spina bifida aperta	P11586	Orphanet
Lumbosacral spina bifida aperta	P11586	Orphanet
Lumbosacral spina bifida cystica	P11586	Orphanet
Cervical spina bifida aperta	P11586	Orphanet
Thoracolumbosacral spina bifida aperta	P11586	Orphanet
Cervicothoracic spina bifida cystica	P11586	Orphanet

Figure 11. Sample of data retrieved from HPA (left) and Ensembl (right). For each record in the HPA dataset (left) there are information about genes and their relation with diseases, while for the case of the Ensembl dataset (right) the name of each disease, the UniProtKB/SwissProt of the related protein and the original source of the information are reported.

3.3. Data pre-processing and curation

For the cases of metabolic and physiological data parameters with missing values, to at least one subject, were subtracted from any further analysis. Moreover, due to some ambiguities in some terminologies of physiological data and to increase the accuracy of the method, a common reference code for time/storage characterization was used. Thus, all labels in physiological data with the extension “NS” or “D2” will be replaced with “D0”. Furthermore, proteins with sampling measurements less than 10 units at all sampling stages were excluded from the analysis, while values of those who passed the filtering process were normalized for statistical purposes. After finishing with the pre-processing, the final experimental dataset consisted of 295 metabolites/compounds, 58 physiological parameters and 465 proteins.

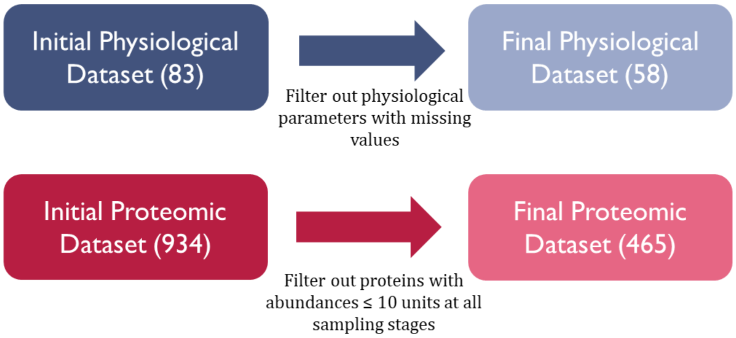

Initial Physiological Dataset (83)
Final Physiological Dataset (58)
Filter out physiological parameters with missing values
Initial Proteomic Dataset (934)
Final Proteomic Dataset (465)
Filter out proteins with abundances ≤ 10 units at all sampling stages

Figure 12. Schematic presentation of the pre-processing of physiological and proteomic data.

In addition, throughout the introduction of data from external sources to the hematological markers network, a cross evaluation was made to avoid data duplication. Therefore, data represented in both experimental dataset and dataset from external sources were introduced only once to the network keeping as properties information or parameters that were unique in each of the two datasets, while common information were imported only once (Figure 13).

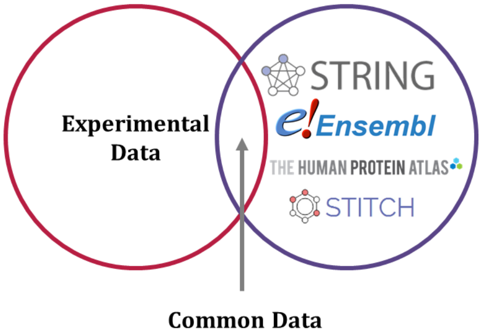


Figure 13. Schematic representation of the composition of the final dataset. Data in common between the experimental dataset and the dataset from external sources were parsed only once keeping all unique properties from each component, but only once the properties in common.

4. GRAPH DATABASE FOR HEMATOLOGICAL MARKERS NETWORKS

4.1. Query Requirements

The next step towards the construction of the hematological markers networks was to determine user requirements in terms of a set of biological queries (to be posed on such networks) that could be useful for the better understanding of biological aspects of the problem. Those requirements will drive the construction of a knowledge graph that could explain interactions or, better yet, reveal potential associations between different parameters. Next, we present these queries arranged in three groups based on the general concept of each query. Each group will be presented in more detail below.

The first group of queries is related to "Biologically Converged Parameters". The following biological questions are part of this group:

1. Inter- and intra- parameter associations in all possible combinations: This query intends to give insight about interactions between one or more different data types. A good example of that could be the gathering of all biologically converged relationships between a group of metabolites (e.g. amino acids) or between metabolites and physiological parameters or even the correlations of statistically significant proteins of the hematological network with diseases. Another aspect of this query could be the association of the metabolic profile of the first sampling (D0 or *in vivo)* with the rest of the samplings (D7 – D42 or *in vitro*).

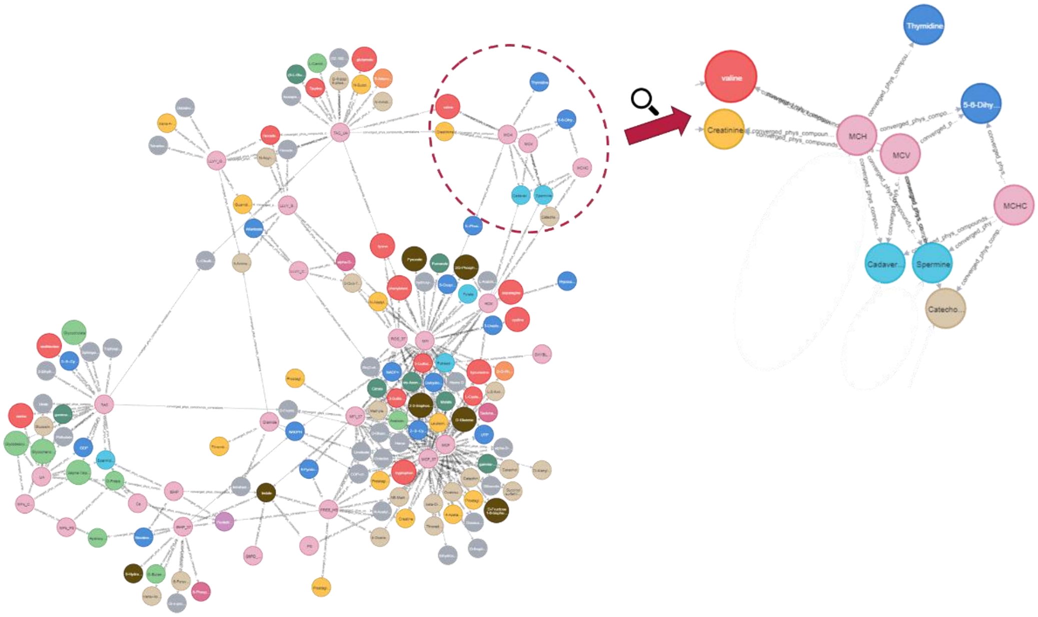


Figure 14. Correlation network of converged relations between physiological parameters and metabolites. This network depicts biologically converged correlations between physiological parameters and metabolites and presents an aspect of the inter-parameter associations. Dashed in pink color the physiological parameters are shown, while metabolites are marked with different colors depending the metabolic pathway they belong.

2. *<u>Determination of crucial parameters:</u>* In substance this biological question refers to the characterization of the most popular nodes of the – case study – system (hub nodes). To answer this question, one could use several centrality algorithms (e.g. betweenness centrality algorithm or PageRank) and detect the ones with the highest score. In graph analytics of biological networks, the most used centrality algorithms are a) betweenness centrality algorithm, b) degree centrality algorithm and c) closeness centrality algorithm [45]. For instance, to find out the most crucial physiological parameters of RBCs one could compute the betweenness and degree centrality of all nodes in the correlation network of biologically converged parameters and later filter out those with high scores in at least one of the two centralities.

3. *<u>Identification of converged metabolites based on the storage timeline of RBCs</u>*: To answer this query one needs to divide the seven samplings into three – storage based – groups (early, mid, and late storage). Early storage refers to the first two samplings (D0 and D7), mid storage to the next three samplings (D14, D21 and D28) and last storage concerns the last two sampling stages (D35 and D42). For each group, one could perform correlation analysis (e.g. Pearson's correlation analysis) to identity interactions that are formed between different metabolites across different storages. Later, to enrich the outcome of this approach the percentage of identity between different – storage -based – networks could be estimated, to get insight about the homogeneity of the system across time.

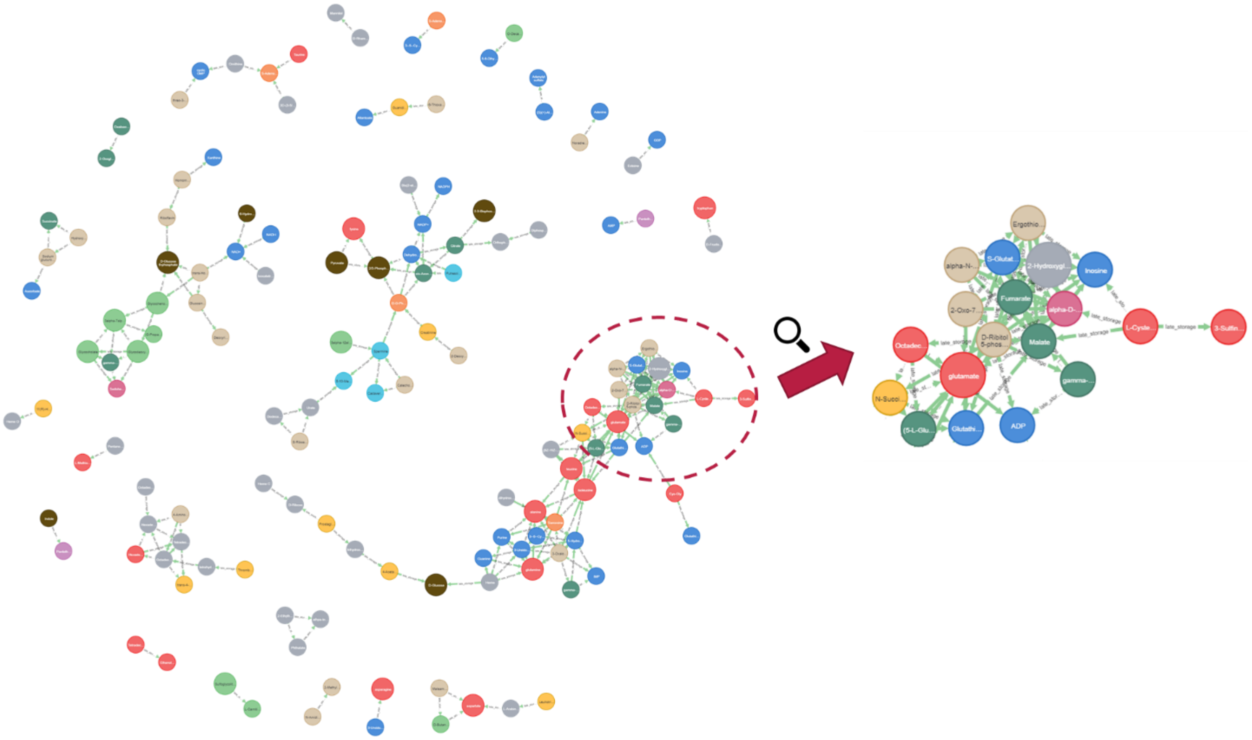


Figure 15. Late storage metabolic correlation network. The presented network depicts the associations of metabolites regarding the last two samplings (D35, D42) or else the late storage after performing Pearson's correlation analysis. Similar network could be derived from the analysis of the other two timelines (early and late storage). Nodes are dashed in different colors according to the metabolic pathway they belong.

The second group of queries is related to “Data Visualization and Subnetworks Representation”. More specifically, with this set of queries, a method to represent/manipulate specific graph data, once the graph has been created, is suggested. The following biological questions are part of this group:

1. <u>*Graph representation based on specific properties of the case study system:*</u> As it is pointed out by the name of this query, one can focus on specific subgroups of the network based on specific properties of nodes or relationships. For instance, nodes could be filtered out based on their degree centrality score or the pathway to which they belong, and relationships could be processed based on the correlation value of the connected nodes or the sampling stage of that.

2. <u>*Data representation in descending/ascending order*</u>: In this case, too, once the hematological marker network is fully created, one can extract data of interest in tables and later present them in diagrams, such as heatmaps or bar graphs. For instance, in Figure 16 the heatmap of betweenness centrality (BC) scores of all converged metabolites across all sampling stages is shown. One can distinguish metabolites with high BC scores based on color differences.

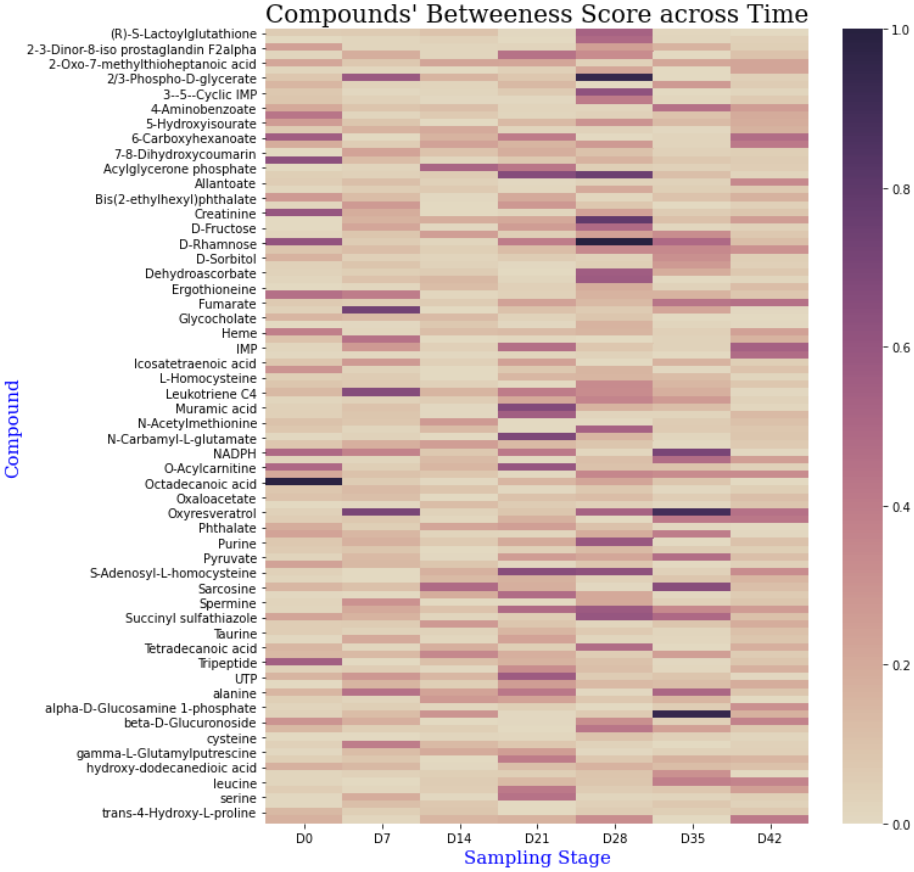


Figure 16. Betweenness centrality (BC) score of converged metabolites across different samplings. Blocks dashed in light color indicate a small BC score, while those dashed in dark colors are supposed to present a higher BC score.

3. <u>*Detection of clusters*</u>: This query refers to the detection of communities on different relationship types, such as converged metabolites, converged physiological parameters, statistically significant proteins, or combinations of those, using relative algorithms. One of the most used algorithms for detection of clusters is the Louvain algorithm. Louvain algorithm is a hierarchical clustering algorithm, that recurrently identifies communities of nodes, by comparing the number of edges within the cluster with the expected number of edges that could be found in it, on highly connected graphs [46]. In Figure 17 an example of the detection of such clusters using the Louvain algorithm is shown. The algorithm was applied on storage-based graphs (early, mid, and late storage) that were described above.

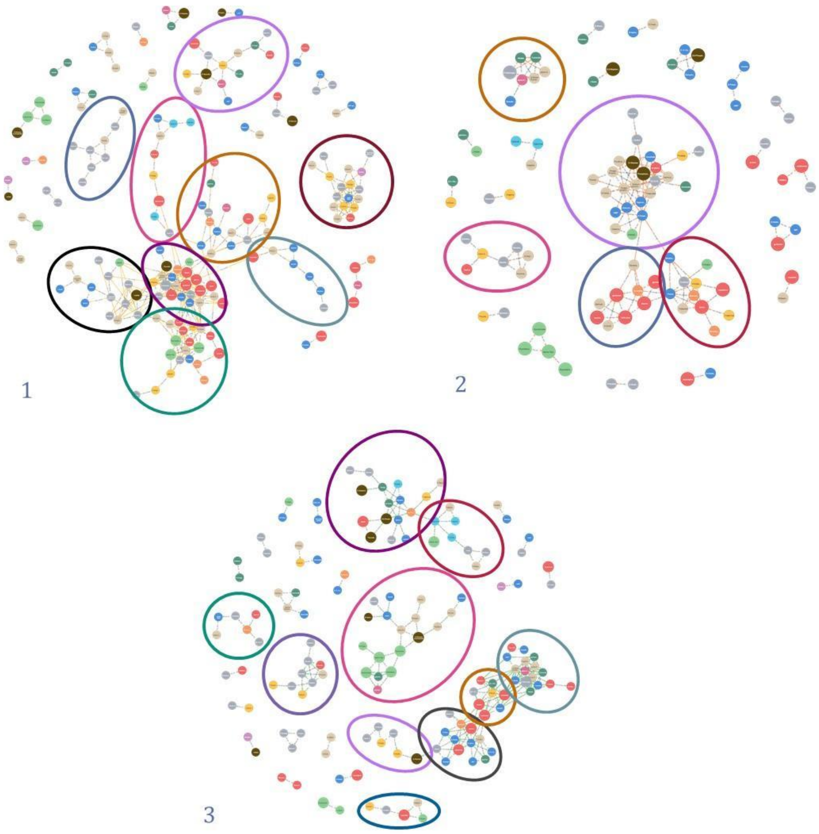


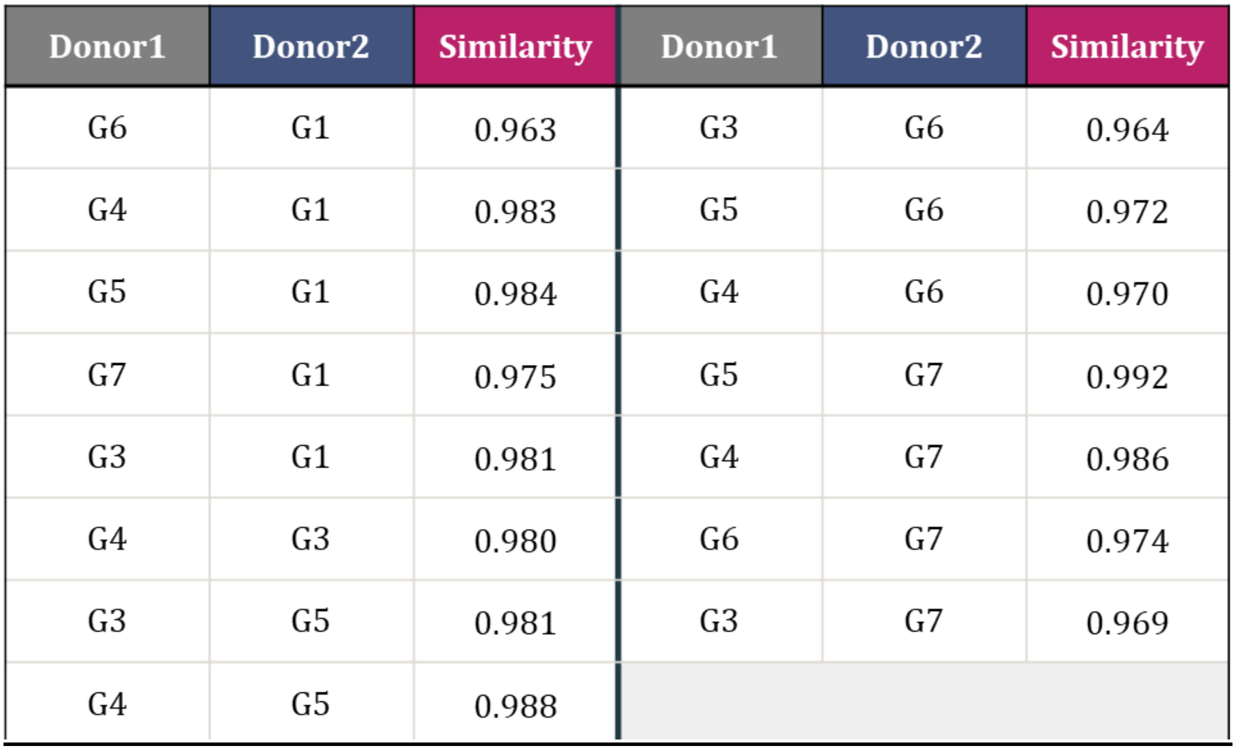
Figure 17. Detection of communities with highly connected components on storage-based graphs. These figures show the clusters that were detected using Louvain algorithm on storage-based graphs (early, mid, and late storage). In caption 1 (left) clusters of the early-storage metabolic correlation network are presented. In caption 2 (right) clusters of metabolites from the mid-storage metabolic correlation network are shown, and clusters of late-storage metabolic correlation network are marked in caption 3 (center).

4. *Focusing on clusters/subnetworks*: This query leads to a subsequent analysis after the detection of communities/clusters. Therefore, by focusing on specific clusters one could extract useful theoretical information that, potentially, could be further investigated through experimental procedures. For instance, it could be of major importance to explore the association of metabolites or other compounds that are highly correlated with the fragility of RBCs or regeneration of ROS, since both are physiological parameters with high impact on the vitality of RBCs. Another interesting approach, that could, potentially, highlight converged relationships between metabolites across time, could be estimating the percentage of identity between clusters of different networks, such as the early-storage metabolic correlation network and late-storage metabolite correlation network.

The final group of queries is related to “Comparative Analysis of Donors’ Metabolic Profile”. This group consists of the following biological questions:

1. *Comparing donors’ metabolic profile in pairs*: Answering to this question could highlight either the homogeneity or heterogeneity of the system, since all donors were tested under the same conditions. Pairwise comparison of metabolic profiles of G6PD$^-$ donors could shed light on this matter. In Table 3 Cosine Similarity of all possible combinations of donors are presented. The closer the similarity score to one the more similar the metabolic profiles of the compared donors are.

Table 3. Pairwise comparison of G6PD$^-$ donors’ metabolic profile. Cosine Similarity was used as metric for the comparison of donors. The closer the similarity score to one the more similar the metabolic profile of donors are.

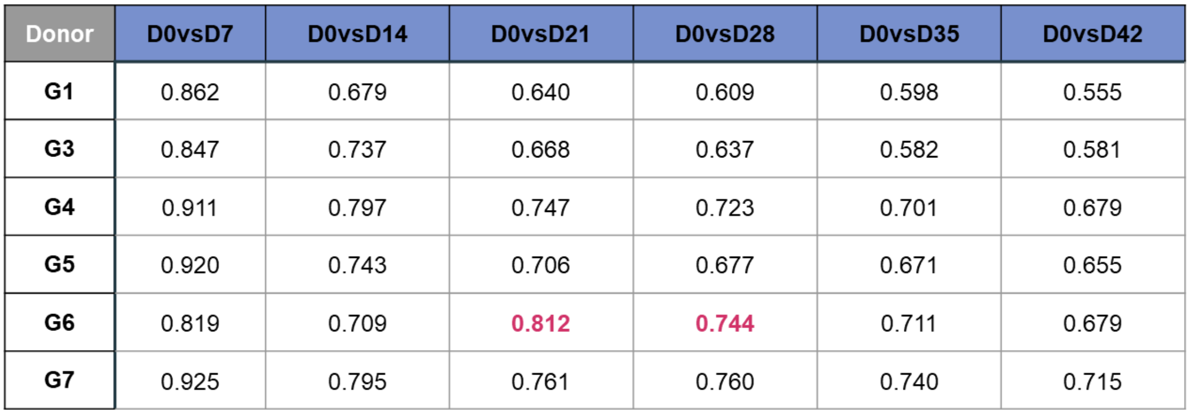


Donor1	Donor2	Similarity	Donor1	Donor2	Similarity
G6	G1	0.963	G3	G6	0.964
G4	G1	0.983	G5	G6	0.972
G5	G1	0.984	G4	G6	0.970
G7	G1	0.975	G5	G7	0.992
G3	G1	0.981	G4	G7	0.986
G4	G3	0.980	G6	G7	0.974
G3	G5	0.981	G3	G7	0.969
G4	G5	0.988			

2. *Investigate the impact of storage to RBCs’ metabolic profile*: The purpose of this query is to gain insight about the effect of storage to RBCs’ vitality and functionality. Comparing the *in vivo* system of each donor (D0) with the *in vitro* system (D7 – D42) could reveal the critical storage period at which the functionality of RBCs starts to disrupt. The higher the similarity score between the two systems the lower the disturbance of RBCs’ function is. In Table 4 Cosine Similarity of *in vivo* system of metabolic profile of each donor with the rest of storage stages (*in vitro* system) is shown. . Each row represents the metabolic profile of G6PD- donor and columns indicate the compared systems. For instance, the column with header “D0vsD7” presents the Cosine Similarity

of in vivo system with the 7th day of storage. One can easily notice a decreasing affinity between the two systems as time passes by.

Table 4. Impact of storage on RBCs' functionality at $G6PD^-$ donors.



Donor	D0vsD7	D0vsD14	D0vsD21	D0vsD28	D0vsD35	D0vsD42
G1	0.862	0.679	0.640	0.609	0.598	0.555
G3	0.847	0.737	0.668	0.637	0.582	0.581
G4	0.911	0.797	0.747	0.723	0.701	0.679
G5	0.920	0.743	0.706	0.677	0.671	0.655
G6	0.819	0.709	**0.812**	**0.744**	0.711	0.679
G7	0.925	0.795	0.761	0.760	0.740	0.715

4.2. Data Model

After defining all necessary queries that would help to better set up the final knowledge graph, we developed the graph data model. Throughout this process node and relationship types of the final graph were specified. For this reason, all data were grouped into categories. In total, the hematological markers network consists of 950 nodes, divided in 41 groups and 87,799 relationships, arranged in 17 distinct types. Following up, all node and relationship types will be presented in tables, alongside with their properties and any additional information that would help to better understand the outline of the knowledge graph.

Starting with, in Table 5 node types of hematological markers network with their properties are presented.

Table 5. Node types of hematological markers network along with their properties.

Node Type	Properties	Additional Information
Compounds (295)	1. early mean [avg (D0, D7)] 2. mid mean [avg (D14, D21, D28)] 3. late mean [avg (D35, D42)] 4. Name 5. molecule type	295 compounds categorized in 33 groups based on the metabolic profile they belong to
Physiological Parameters (58)	1. Name (acronym) 2. Full_name (official name)	Physiological parameters with missing values were excluded from the analysis

Proteomics (465)	1. Name 2. Gene 3. UniProtID 4. Molecular Weight 5. early control 6. mid control 7. late control 8. early G6PD 9. mid G6PD 10. late G6PD 11. abs_early logFC 12. abs_mid logFC 13. abs_late logFC 14. molecule type	The following properties refer to the control samplings: early, mid & late control, while early, mid & late G6PD refer to G6PD$^-$ samplings Properties abs_early, abs_mid & abs_late logFC refer to the absolute value of logFC (logarithmic Fold Change) value between diseased and control samples of the corresponding sampling stage
Donors (7)	1. Name	1 control & 6 G6PD$^-$ donors Hematological data regarding control donor are not included in any further analysis, due to lack of data
Stitch Data (40)	1. Name 2. molecule type	Additional data related to G6PD that were retrieved from STITCH database
String Data (15)	1. Gene 2. molecule type	Additional proteomic data related to G6PD retrieved from String database
Ensembl Data (4)	1. UniProtID	External proteomic data related to G6PD that were collected from Ensembl rest API
Disease (49)	1. Name 2. source	Data related to diseases or generic pathogenic phenotypes. Information about such data were retrieved from Ensembl and HPA
G6PD (1)	1. Name 2. Gene 3. UniProtID 4. molecule type	—

Following up, in Table 6 relationship types of the knowledge graph alongside with their properties and additional explanatory information are presented.

Table 6. Relationship types of hematological markers network along with their properties

Relationship Type	Properties	Additional Information
RELATED TO (12,390 w/o control donor 14,455 with control donor)	1. CON 2. timestamp	Connects donors with compounds. Each relationship presents the concentration (CON) of each compound at a specific sampling stage (timestamp)
Physiology (1,025 w/o control donor 1,435 with control donor)	1. value 2. timestamp	Connects donors with physiological parameters. Measurement (value) of each physiological parameter along with the sampling stage (timestamp) of it are recorded
Associated with (32)	—	Filters G6PD-related compounds at most of sampling stages
Compound Similarity (12,360)	1. similarity 2. timestamp 3. correlation type	Connects highly correlated compounds. Threshold: abs(Pearson’s R) ≥ 0.85
Bio converged Correlations (134)	1. correlation values 2. correlation type	Filters pairs of compounds with significant correlation in at least 4 sampling stages
Early storage (357) Mid storage (158) Late storage (236)	1. similarity 2. correlation type	Each relationship type represents significant correlations between compounds after grouping them based on storage stage. That said, early storage refers to the first two samplings (D0, D7), the next three samplings (D14, D21 and D28) are characterized as mid storage and late storage refers to the last two samplings (D35 and D42)
Phys - compounds correlations (42,351)	1. time pair 2. similarity 3. correlation type	Correlations between physiological parameters and compounds at all possible time-based correlations (homologous or heterologous). Threshold: abs(Pearson’s R) ≥ 0.80

Converged phys - compounds correlations (312)	1. times of occurrence	Filters pairs of physiological parameters and compounds that are correlated in at least 25% of theoretically possible combinations*. *Theoretically possible combinations: Since there are 7 sampling stages for both metabolic and physiological data, there are 49 (7*7) possible correlations between physiological parameters and compounds.
Protein correlations (6,704)	1. correlation type 2. similarity	Correlations between proteins of G6PD$^-$ donors Threshold: abs(Pearson's R) ≥ 0.99 The threshold in this case was stricter due to lack of proteomic data (only 3 samplings took place)
Protein compounds correlations (8,515)	1. correlation type 1. similarity	Correlations between proteins and biologically converged compounds. Threshold: abs(Pearson's R) ≥ 0.85
Donor similarity (15)	2. similarity	Pairwise comparison of G6PD$^-$ donors' metabolic profile. Cosine similarity was used as metric
PPI (445)	1. source 2. textmining score 3. neighborhood score 4. database score 5. coexpression 3. combined score	Protein-protein interactions of proteins related – directly or indirectly – to G6PD. STITCH database and String database are the sources of these interactions
Protein Chemical Interaction (137)	1. source 2. textmining score 3. neighborhood score 4. database score 5. coexpression 6. combined score	Protein-chemical interactions of proteins related – directly or indirectly – to G6PD. STITCH database is the source of these interactions
Chemical Chemical Interaction (89)	1. source 2. textmining score 3. neighborhood score 4. database score 5. coexpression 6. combined score	Introduces chemical-chemical interactions to knowledge graph. Data regarding to such relationships were retrieved from STITCH database
Phenotype (70)	—	Connects proteins to diseases or pathological phenotypes that are related to. Ensembl and HPA are the sources of such interactions

4.3. Neo4j Design and Setup

4.3.1. Importing Data to Neo4j

There are several ways to import data into Neo4j depending on the data source. For API data the most common way is through prefixed algorithms (e.g. APOC standard extension library) that are available in Neo4j, while for remote or local files Cypher queries are preferred [33]. In this section, we are going to focus on importing local data using Cypher queries.

Before importing local data sources to Neo4j is mandatory to check the format of the file, since only CSV (comma-separated values) files can be processed with Neo4j. Following that, the LOAD CSV command should be used to read CSV files. To use this clause properly, one needs to specify the exact path of the location of the file including the prefix “file:///” to the query. An example of this command is shown below.

```
LOAD CSV WITH HEADERS FROM
"file:///C:/Users/mbats/OneDrive/Desktop/metabolomic_data.csv" AS row
```

In this example each row of the file “metabolomic_data.csv” is passed to the Neo4j platform. Since the extension “WITH HEADERS” was used, all values of the first line of the file will be considered as column names.
In addition, it is important to mention that each value is passed to the platform in the format of a string, while null or empty values are not stored in Neo4j. Therefore, several transformations, such replacing missing values or transforming string data to integers or floats, might be necessary while processing. Besides that, the most practical part of dealing with CSV files in Neo4j platform is the fact that the user can convert any data into graph-related data types, such as nodes or relationships. Thus, the performance during data loading is increasing and the handling of large amounts of data is more manageable. Prerequisites for such procedures are the understanding of graph database systems and basic Cypher commands. An example of transforming text data into graph data is shown below.

```
LOAD CSV WITH HEADERS from
"file:///C:/Users/mbats/OneDrive/Desktop/proteomics_data.csv" AS data
WITH data
WHERE NOT ALL(x IN
[data.Gpool_D2,data.Gpool_D42,data.Gpool_D21,data.C_D2, data.C_D42,
data.C_D21,data.Ves_C_D42,data.Ves_G_D42] WHERE toFloat(x) <= 10.0)
MERGE (n:Proteomics{Name:data.`Identified Proteins
(934)`,UniProtID:data.`AccessionNumber`,MolecularWeight:data.`Molecular
Weight`,early_control:toFloat(data.C_D2),early_G6PD:toFloat(data.Gpool_D2),mid_control:toFloat(data.C
_D21),mid_G6PD:toFloat(data.Gpool_D21),late_control:toFloat(data.C_D42),late_G6PD:toFloat(data.Gp
ool_D42),Ves_C_D42:toFloat(data.Ves_C_D42),Ves_G_D42:toFloat(data.Ves_G_D42)})
```

In this example each row of the file “proteomics_data.csv” is passed to the Neo4j platform. Once again, the first row of the file is used as column names. Following that, a filtering process takes place using the WHERE clause. The query concludes by transforming initial data into graph data and more specifically into nodes under the label “Proteomics”.

One can easily notice that several columns of proteomics data are passed as node properties and some of them are transformed to float values.

This section concludes with presenting the whole process that was followed to pass all available data regarding biomedical/hematological markers related to the issue that was studied. The process of importing the data was divided in four parts depending on the data type/source:

A. Donor Names & Metabolic Data

```
LOAD CSV WITH HEADERS FROM
"file:///C:/Users/mbats/OneDrive/Desktop/metabolomic_data.csv" AS row
UNWIND keys(row) AS head
WITH DISTINCT(head) AS heads ORDER BY toUpper(head) ASC
WHERE heads =~ 'G.*' OR heads =~ 'C_.*'
WITH apoc.text.replace(heads,'_D[0-9]*','') as names
WITH DISTINCT names
MERGE (n:Donors{Name:names});
```

The above clause extracts information regarding G6PD donors from the “metabolomic_data.csv” file. Each donor (including control) was passed as a distinct node under the label “Donors”. Following that, information about each metabolite/compound from the aforementioned file was introduced as a distinct node to the knowledge graph. All compounds were grouped based on the metabolic path they belong to. Additionally, several columns were used as properties for each node. The query that was used to pass metabolites to the network is shown below.

```
LOAD CSV WITH HEADERS FROM
"file:///C:/Users/mbats/OneDrive/Desktop/metabolomic_data.txt" AS record
CALL apoc.create.node([record.Pathway],{Name: record.compound, pvalue:toFloat(record.pvalue)})
YIELD node
WITH record,node
MATCH (n:Donors)
WITH record,node,n, ["D0","D7","D14","D21","D28","D35","D42"] AS timestamps
UNWIND range(0,size(timestamps)-1) AS id
MERGE(n)-
[:RELATED_TO{CON:toFloat(record[n.Name+"_"+timestamps[id]]),timestamp:timestamps[id]}]->(node);
```

B. Physiological Data

The following queries describe the process of importing physiological data to the hematological markers network. At first, physiological data of G6PD$^-$ donors were introduced to the network. Each column name was passed as distinct node under the label “Physiological_Parameters” and the value of each parameters alongside with the sampling stage were passed as properties for each node. This procedure was repeated for control data since they were stored in a different file. This process concludes by passing the biomedical explanation of each physiological parameter as a property.

```
LOAD CSV WITH HEADERS FROM
"file:///C:/Users/mbats/OneDrive/Desktop/physiological_data_refined.txt" AS lines
UNWIND keys(lines) AS parms
WITH apoc.text.replace(parms,'_D[0-9]*','') AS names, lines
WITH distinct(names), lines
```

```
MERGE (p:Physiological_Parameters{Name:names})
WITH distinct(names), lines, p,
["D0","D7","D14","D21","D28","D35","D42"] AS timestamps ORDER BY names ASC
UNWIND range(0,size(timestamps)-1) AS id
WITH p, timestamps[id] AS time,
collect(lines[names+"_"+timestamps[id]]) AS values
WHERE size(values) > 0
MATCH (n:Donors)
WITH collect(distinct n.Name) AS source, time, p, values
UNWIND range(0,size(values)-1) as vector
MATCH (m:Donors)
WHERE m.Name =~ source[vector]
MERGE (m)-[:Physiology{timestamp:time,value:toFloat(values[vector])}]->(p);

LOAD CSV WITH HEADERS FROM
"file:///C:/Users/mbats/OneDrive/Desktop/physiological_data_control.txt" AS data
WITH data, ["D0","D7","D14","D21","D28","D35","D42"] AS
timestamps
MATCH (n:Physiological_Parameters), (m:Donors{Name:'C'})
UNWIND range(0,size(timestamps)-1) AS id
WITH m, n, timestamps[id] AS time, data[n.Name+"_"+timestamps[id]] AS value
WHERE value IS NOT NULL
MERGE (m)-[r:Physiology{timestamp:time,value:value}]->(n);

LOAD CSV FROM
"file:///C:/Users/mbats/OneDrive/Desktop/physiological_abbreviations.txt" AS data
WITH data
MATCH (n:Physiological_Parameters)
WHERE n.Name = data[0]
SET n.Full_Name =  data[1];
```

C. Proteomic Data

Before introducing proteomic data to the knowledge graph a filtering process took place. Therefore, all proteins with concentration less than 10 units at all samplings were excluding from the analysis for normality issues. Following that, each of the rest proteins was passed as a distinct node under the label “Proteomics”. Moreover, several columns were added as properties for each protein.

```
LOAD CSV WITH HEADERS FROM
"file:///C:/Users/mbats/OneDrive/Desktop/proteomics_data.txt" AS data
WITH data
WHERE NOT ALL(x IN [data.Gpool_D2, data.Gpool_D42, data.Gpool_D21, data.C_D2, data.C_D42,
data.C_D21, data.Ves_C_D42, data.Ves_G_D42] WHERE toFloat(x) <= 10.0)
MERGE (n:Proteomics{Name:data.`Identified Proteins (934)`, UniProtID:data.`Accession Number`,
MolecularWeight:data.`Molecular Weight`, early_control:toFloat(data.C_D2),
early_G6PD:toFloat(data.Gpool_D2), mid_control:toFloat(data.C_D21),
mid_G6PD:toFloat(data.Gpool_D21), late_control:toFloat(data.C_D42),
late_G6PD:toFloat(data.Gpool_D42),Ves_C_D42:toFloat(data.Ves_C_D42),Ves_G_D42:toFloat(data.Ve
s_G_D42)})
WITH n,apoc.text.regexGroups(n.Name, 'GN=[A-Z]*')[0][0] AS name
SET n.Gene = apoc.text.replace(name, 'GN=','');
```

D. External Sources

At last, introducing data from external sources to the network was the final part of the process. Besides some differences in the context of their data, the main idea of introducing each data source to the network was somewhat the same. To begin with, a comparison with the existing graph data took place, to identify which data were common and which one were not present in the network. Common data were updated, in the sense of introducing some extra properties to existing nodes, while new data were passed as nodes under the label of the data source from which they retrieved from. Moreover, information about relationships between nodes were introduced to the network as well. Following that, the query that was used for each external source is presented below.

STITCH database

```
LOAD CSV WITH HEADERS FROM
"file:///C:/Users/mbats/OneDrive/Desktop/stitch_interactions.csv" AS data
WITH apoc.coll.union(collect([data.node1,data.node1_id]),collect([data.node2,data.node2_id])) AS
list_of_names
UNWIND range(0,size(list_of_names)-1) AS i
match (m)
WHERE (labels(m) IN [["Proteomics"],["Physiology"],["G6PD"]] AND m.Name contains
apoc.text.capitalize(list_of_names[i][0])) OR (NOT labels(m) IN [["Proteomics"],["Physiology"],["G6PD"]]
AND m.Name = list_of_names[i][0])
SET m.molecule_type =
CASE
WHEN
list_of_names[i][1] CONTAINS 'ENSP' THEN 'Protein'
WHEN
list_of_names[i][1] CONTAINS 'CID' THEN 'Chemical'
END
WITH list_of_names, COLLECT(list_of_names[i]) AS names
WITH apoc.coll.subtract(list_of_names, names) AS external_sources
UNWIND range(0,size(external_sources)-1) AS j
MERGE(k:Stitch_data{Name:external_sources[j][0], molecule_type:
CASE WHEN external_sources[j][1] CONTAINS 'ENSP' THEN 'Protein'
ELSE 'Chemical' END});
LOAD CSV WITH HEADERS FROM
"file:///C:/Users/mbats/OneDrive/Desktop/stitch_interactions.csv" AS data
WITH data
MATCH (n)
MATCH (m)
WHERE (NOT labels(n) IN [['Donors'],['Physiological_Parameters']] AND NOT labels(m) IN
[['Donors'],['Physiological_Parameters']]) AND (apoc.text.capitalize(n.Name) CONTAINS
apoc.text.capitalize(data.node1) AND apoc.text.capitalize(m.Name) CONTAINS
apoc.text.capitalize(data.node2)) AND (n) <> (m)
MERGE (n)-[r:interaction{source: "STITCH", textmining_score:toFloat(data.textmining_score),
coexpression:toFloat(data.coexpression_score),neighbourhood_score:toFloat(data.neighbourhood_score
),database_score:toFloat(data.database_score),combined_score:toFloat(data.combined_score)}]->(m)
WITH n, r, m
CALL apoc.refactor.setType(r, CASE
WHEN n.molecule_type = 'Protein' and m.molecule_type = 'Protein' then 'PPI'
WHEN n.molecule_type = 'Chemical' AND m.molecule_type = 'Chemical' THEN
'Chemical_Chemical_Interaction'
WHEN (n.molecule_type = 'Chemical' AND m.molecule_type = 'Protein') OR (m.molecule_type = 'Chemical'
and n.molecule_type = 'Protein') THEN
'Protein_Chemical_Interaction'
END)
YIELD INPUT, OUTPUT
WHERE type(r) = 'interaction'
DELETE r
```

String database

```
LOAD CSV WITH HEADERS FROM
"file:///C:/Users/mbats/OneDrive/Desktop/string_interactions.csv" AS data
WITH data
MATCH (n)
WHERE labels(n) IN [["Proteomics"],["Stitch_data"],["G6PD"]] AND n.Gene = data.node1 or n.Gene =
data.node2
WITH apoc.coll.union(COLLECT(DISTINCT data.node1), COLLECT(DISTINCT data.node2)) AS
listOFnames, COLLECT(DISTINCT n.Gene) AS common_names
WITH apoc.coll.subtract(listOFnames, common_names) AS string_data
UNWIND range(0,size(string_data)-1) as j
MERGE (m:String_data{Gene:string_data[j],molecule_type:"Protein"})
WITH m
LOAD CSV WITH HEADERS FROM
"file:///C:/Users/mbats/OneDrive/Desktop/string_interactions.csv" as data
MATCH (n:String_data)
WHERE m.Gene = data.node1 AND n.Gene = data.node2
MERGE (m)-[:PPI{source:"String", database_score:data.database_score,
textmining_score:data.textmining_score, coexpression_score:data.coexpression_score,
neighbourhood_score:data.neighbourhood_score,
combined_score:data.combined_score}]->(n);
```

Ensembl database

```
LOAD CSV WITH HEADERS FROM
"file:///C:/Users/mbats/OneDrive/Desktop/Ensembl_interactions.csv" AS data
WITH data
MERGE (n:Ensembl_data{UniProtID:data.UniprotID})-[r:phenotype]-
(m:Disease{Name:data.disease,source:data.source})
WITH n, r, m
MATCH (p)
WHERE (p:Proteomics or p:G6PD) AND p.UniProtID = n.UniProtID
DELETE n, r
MERGE (p)-[:phenotype]->(m);
```

Human Protein Atlas database

```
LOAD CSV WITH HEADERS FROM
"file:///C:/Users/mbats/OneDrive/Desktop/HPA_interactions.csv" AS data
WITH data
MERGE (m:Disease{Name:data.diseases,source:"HPA"})
with data,m
MATCH (p)
WHERE p.UniProtID = data.UniProtID and m.Name = data.diseases
MERGE (p)-[:phenotype]->(m)
```

4.3.2. Hematological Data Analysis

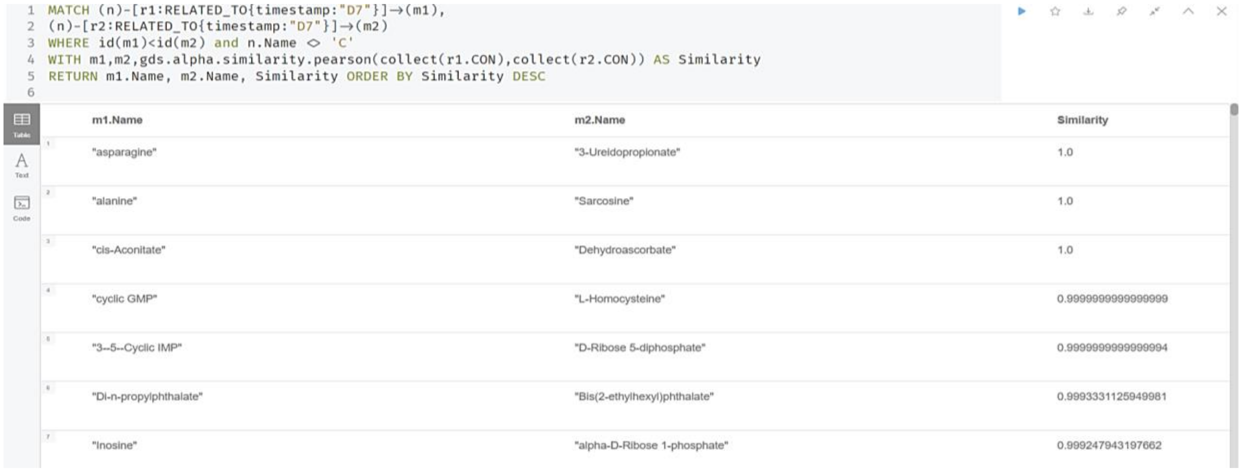
The next step, after importing all necessary data to the network, included statistical analysis using graph-related algorithms, to filter the most statistically significant parameters of the network. The process that was followed starts with finding a suitable approach to explore the data that were available, continues with setting a proper threshold, so that the outcome would be accurate enough and concludes with filter out biologically converged intra- and inter- parameter relationships.

<u>Approach</u>

Starting with, two algorithms were applied during the statistical analysis: Pearson Similarity algorithm and Cosine Similarity algorithm. The first one was used for the characterization of significant intra- and inter- parameters associations between different data types (Compound Similarities, Physiological Parameter – Compound Similarities, Protein Similarities, Protein – Compounds Similarities) and the second one was used for the identification of percent of identity between metabolic profiles of different users (Donor similarities) or different storage stages (early, mid, and late storage).

Pearson Similarity algorithm estimates the similarity between two lists of numbers. It can be characterized as a symmetrical, since calculating the similarity of item A with item B would be the same as the computation of similarity between item B and item A. In practice, Pearson Similarity is the covariance matrix of two variables divided by the product of their standard deviation [47]. The outcome is bounded in [-1,1]. The closer to -1 or 1 the similarity of two items, the more negative or positive, respectively, associated they are. Two variables are negative correlated as the one variable increases the other decreases, and vice versa, while positive correlation indicates that both variables move in tandem [48]. The mathematical equation that describes Pearson Similarity algorithm is presented below:

$$\text{similarity}(A,B)=\frac{\text{cov}(A,B)}{\sigma_A \cdot \sigma_B}=\frac{\sum_{i=1}^{n}(A_i-\underline{A})(B_i-\underline{B})}{\sqrt{\sum_{i=1}^{n}(A_i-\underline{A})^2(B_i-\underline{B})^2}}$$



```
MATCH (n)-[r1:RELATED_TO{timestamp:"D7"}]→(m1),
(n)-[r2:RELATED_TO{timestamp:"D7"}]→(m2)
WHERE id(m1)<id(m2) and n.Name <> 'C'
WITH m1,m2,gds.alpha.similarity.pearson(collect(r1.CON),collect(r2.CON)) AS Similarity
RETURN m1.Name, m2.Name, Similarity ORDER BY Similarity DESC
```

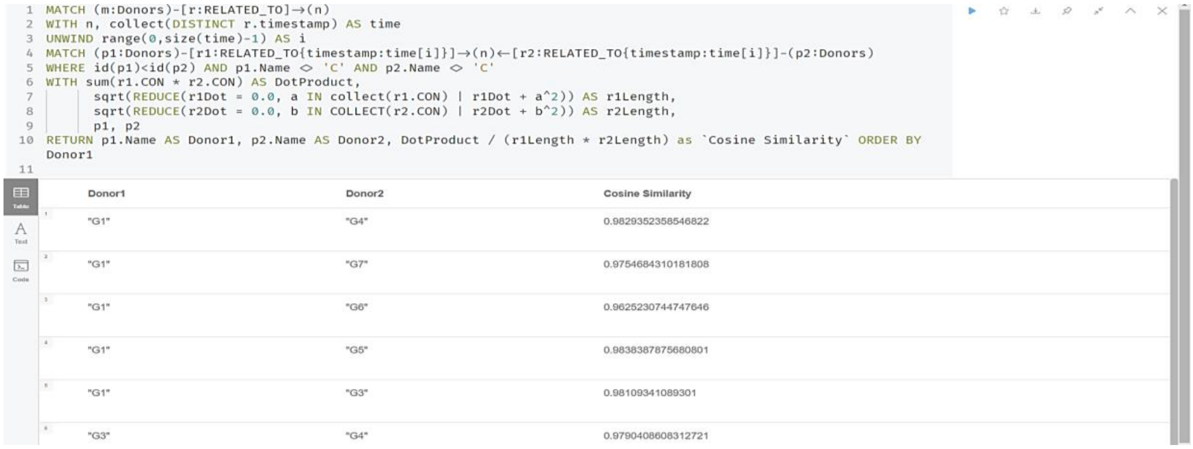
	m1.Name	m2.Name	Similarity
1	"asparagine"	"3-Ureidopropionate"	1.0
2	"alanine"	"Sarcosine"	1.0
3	"cis-Aconitate"	"Dehydroascorbate"	1.0
4	"cyclic GMP"	"L-Homocysteine"	0.9999999999999999
5	"3--5--Cyclic IMP"	"D-Ribose 5-diphosphate"	0.9999999999999994
6	"Di-n-propylphthalate"	"Bis(2-ethylhexyl)phthalate"	0.9993331125949981
7	"Inosine"	"alpha-D-Ribose 1-phosphate"	0.999247943197662

Figure 18. An example of the use of Pearson Similarity algorithm in NEO4J.

In Figure 18 an example of the use of Pearson Similarity algorithm along with part of the output is shown. More specifically the Cypher query that is presented calls all metabolites and their concentration at the 7th day of the experiment and returns the Pearson coefficient of all possible pairs of metabolites. Each metabolite is considered as vector whose elements are the concentration of the metabolite from each of the donors.

Cosine Similarity algorithm estimates the similarity between two non-zero vectors, by computing the cosine of their angle. The outcome is bounded in [0,1]. When the outcome is zero the two vectors are diametrically opposed, thus there is not association between them. On the other hand, the closer to one the cosine similarity of two variables, the more identical they are [49]. The mathematical expression that describes Cosine Similarity is available below:

$$\text{similarity}(A, B) = \frac{A \cdot B}{\|A\| \cdot \|B\|} = \frac{\sum_{i=1}^{n} A_i \cdot B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \cdot \sqrt{\sum_{i=1}^{n} B_i^2}}$$



```
MATCH (m:Donors)-[r:RELATED_TO]->(n)
WITH n, collect(DISTINCT r.timestamp) AS time
UNWIND range(0,size(time)-1) AS i
MATCH (p1:Donors)-[r1:RELATED_TO{timestamp:time[i]}]->(n)<-[r2:RELATED_TO{timestamp:time[i]}]-(p2:Donors)
WHERE id(p1)<id(p2) AND p1.Name <> 'C' AND p2.Name <> 'C'
WITH sum(r1.CON * r2.CON) AS DotProduct,
     sqrt(REDUCE(r1Dot = 0.0, a IN collect(r1.CON) | r1Dot + a^2)) AS r1Length,
     sqrt(REDUCE(r2Dot = 0.0, b IN COLLECT(r2.CON) | r2Dot + b^2)) AS r2Length,
     p1, p2
RETURN p1.Name AS Donor1, p2.Name AS Donor2, DotProduct / (r1Length * r2Length) as `Cosine Similarity` ORDER BY
Donor1
```

	Donor1	Donor2	Cosine Similarity
1	"G1"	"G4"	0.9829352358546822
2	"G1"	"G7"	0.9754684310181808
3	"G1"	"G6"	0.9625230744747646
4	"G1"	"G5"	0.9838387875680801
5	"G1"	"G3"	0.98109341089301
6	"G3"	"G4"	0.9790408608312721

Figure 19. Applying Cosine Similarity algorithm in NEO4J to identify the identity of donors' metabolic profile.

In Figure 19 one can observe the clause that was used to calculate the cosine similarity of all pairs of donors, along with part of the output.

Setting the threshold

After properly estimating Pearson Similarity scores, the filtering of the most significant intra- and inter- parameter correlations took place. The first step, to achieve that, was to set a threshold, so that statistically significant associations will be distinguished. The value of the threshold varied in each case, depending on the size of the case study dataset or the number of samplings. For instance, in the case of proteomic data, a stricter threshold was applied since fewer samplings took place (three samplings in total: D0, D21 and D42) compared to the rest of the experimental data (seven weekly samplings in total). It's important to mention that this step was applied only in cases where Pearson Similarity algorithm were used, since Cosine Similarity was used only for purposes of identity characterization between compared groups. In Table 7 thresholds of all intra- and -inter- parameter associations are available.

Table 7. Thresholds applied on intra- and inter- parameter correlations for filtering purposes.

Correlation Type	Threshold (absolute Pearson Similarity)
Compound Similarities	0.85
Physiological Parameter – Compound Similarities	0.80
Protein Similarities	0.99
Protein – Compound Similarities	0.85

Filtering biologically converged correlations

By applying the threshold that was mentioned above the most insignificant associations between different node types were excluded from any further analysis. However a stricter approach was necessary, to proceed with the filtering of biologically converged correlations. For this reason, the repeatability score was applied. As its name suggests, repeatability score explores the times an event occurs. In our case, the event, that was tested, was the correlation between two variables. Therefore, if a case study pair of variables passed the repeatability score, the relationship that is formed between them would be considered biologically converged.

From this process metabolites related to G6PD, biologically converged correlations between metabolites and biologically converged relationships between metabolites and physiological parameters were identified. For the first two cases the repeatability score was described as the occurrence of the relationship between each pair in at least 4 out of the 7 samplings, while for the case of converged correlations between physiological parameters and metabolites the repeatability score was expressed as the occurrence of the relationship between two variables in at least 25% of the theoretically possible combinations (see section 4.2).

Address queries

In this section the cypher queries that were used throughout the statistical analysis will be addressed.

Compound Similarities

```
MATCH (n)-[r1:RELATED_TO]->(m)
WITH COLLECT(DISTINCT r1.timestamp) as timepoints
UNWIND range(0,size(timepoints)-1) as time
MATCH (n:Donors)-[r:RELATED_TO{timestamp:timepoints[time]}]->(m1),
(n)-[r2:RELATED_TO{timestamp:timepoints[time]}]->(m2)
WHERE m1 <> m2 AND n.Name <> 'C'
WITH m1,m2,r2.timestamp AS timepoint,
gds.alpha.similarity.pearson(collect(r.CON),collect(r2.CON)) as Similarity
WHERE abs(Similarity) >=0.85
MERGE (m1)-[r3:compound_similarity{similarity:Similarity, timestamp:timepoint}]-(m2)
SET r3.correlation_type = CASE WHEN r3.similarity > 0 THEN "positive" else "negative" END;
```

Biologically Converged Correlations between Metabolites

```
MATCH (m1)-[r:compound_similarity]-(m2)
WITH DISTINCT m1,m2,[R IN COLLECT(r.similarity) WHERE abs(R)>=0.85] AS true_values
WHERE size(true_values)>=4 AND id(m1)<id(m2)
MERGE (m1)-[r:bio_converged_correlations{correlation_values:true_values}]->(m2)
SET r.correlation_type = CASE WHEN ALL(x IN r.correlation_values WHERE x < 0) THEN "negative" END
SET r.correlation_type = CASE WHEN ALL(x IN r.correlation_values WHERE x > 0) THEN "positive" END;
```

Metabolites associated with G6PD

```
CREATE (n:G6PD{Name:"G6PD",Gene:'G6PD,UniProtID:'P11413'});
MATCH (n:Donors)
WHERE n.Name <> 'C' //Does not include control donor
WITH COLLECT(n.Name) AS samples,
["D7","D14","D21","D28","D35","D42"] AS timestamps
UNWIND range(0,size(samples)) AS id
UNWIND range(0,size(timestamps)) as time
MATCH (n:Donors{Name:samples[id]})-[r:RELATED_TO{timestamp:"D0"}]->(m),
(n2:Donors{Name:samples[id]})-[r2:RELATED_TO{timestamp:timestamps[time]}]->(m)
WHERE n.Name <> 'C' AND n2.Name <> 'C'
WITH m, r2.timestamp AS pair,
gds.alpha.similarity.pearson(COLLECT(r.CON), COLLECT(r2.CON)) AS similarity
WITH m, COLLECT(similarity) AS allPearsons
WITH m,[R IN allPearsons WHERE abs(R)>=0.80] AS true_values
MATCH (n:G6PD)
WHERE size(true_values)>=4
MERGE (m)-[r:associated_with]->(n)
```

Storage-based Metabolic Correlation Networks (early, mid and late storage)

```
WITH ["D0","D7"] AS time
UNWIND range(0,size(time)-1) AS id
MATCH (p)<-[r2:RELATED_TO{timestamp:time[id]}]-(n)-[r1:RELATED_TO{timestamp:time[id]}]->(m)
WHERE id(p)<id(m) AND NOT n.Name = 'C'
WITH p,m,gds.alpha.similarity.pearson(COLLECT(r1.CON),COLLECT(r2.CON)) AS Similarity
WHERE abs(Similarity)>=0.85
MERGE (m)-[r:early_storage{similarity:Similarity}]-(p)
SET r.correlation_type = CASE WHEN r.Similarity > 0 THEN "positive" ELSE "negative" END
UNION
WITH ["D14","D21","D28"] AS time
UNWIND range(0,size(time)-1) AS id
MATCH (p)<-[r2:RELATED_TO{timestamp:time[id]}]-(n)-[r1:RELATED_TO{timestamp:time[id]}]->(m)
WHERE id(p)<id(m) AND NOT n.Name = 'C'
WITH p,m,gds.alpha.similarity.pearson(COLLECT(r1.CON),COLLECT(r2.CON)) AS Similarity
WHERE abs(Similarity)>=0.85
MERGE (m)-[r:mid_storage{similarity:Similarity}]-(p)
SET r.correlation_type = CASE WHEN r.Similarity > 0 THEN "positive" ELSE "negative" END
UNION
WITH ["D35","D42"] AS time
UNWIND range(0,size(time)-1) AS id
MATCH (p)<-[r2:RELATED_TO{timestamp:time[id]}]-(n)-[r1:RELATED_TO{timestamp:time[id]}]->(m)
WHERE id(p)<id(m) AND NOT n.Name = 'C'
WITH p,m,gds.alpha.similarity.pearson(COLLECT(r1.CON),COLLECT(r2.CON)) AS Similarity
WHERE abs(Similarity)>=0.85
MERGE (m)-[r:late_storage{similarity:Similarity}]-(p)
```

```
SET r.correlation_type = CASE WHEN r.Similarity > 0 THEN "positive" ELSE "negative" END;
```

Protein Correlations

```
MATCH (n:Proteomics)
WITH n
MATCH (m:Proteomics)
WHERE n.Name <> m.Name AND id(n)<id(m)
WITH n, m,
gds.alpha.similarity.pearson([n.early_G6PD,n.mid_G6PD,n.late_G6PD],[m.early_G6PD,m.mid_G6PD,m.late_G6PD]) AS similarity
WHERE abs(similarity)>=0.99
MERGE (n)-[r:protein_correlations{similarity:similarity}]->(m)
SET r.correlation_type = CASE WHEN abs(similarity) > 0 THEN "positive" ELSE "negative" END;
```

Physiological Parameters – Compounds Correlations

```
MATCH (p)<-[r1:Physiology]-(n)-[r2:RELATED_TO]->(m)
WHERE NOT n.Name = 'C'
WITH p, m, r1.timestamp AS time1, r2.timestamp AS time2,
gds.alpha.similarity.pearson(COLLECT(r1.value),COLLECT(r2.CON)) AS similarity
WHERE abs(similarity)>=0.80
MERGE (p)-[r:phys_compounds_correlations{time_pair:time1+"-"+time2,similarity:toFloat(similarity)}]-(m)
SET r.correlation_type = CASE WHEN r.similarity > 0 THEN "positive" ELSE "negative" END;
```

Converged Correlations between Physiological Parameters and Compounds

```
MATCH (n)-[r:phys_compounds_correlations]->(m)
WITH n, m, COLLECT(r.similarity) AS values, count(r) AS rel_counts
WHERE rel_counts > 12 //25% of theoretically possible combinations
MERGE (n)-[r:converged_phys_compounds_correlations{times_of_occurance:rel_counts}]-(m)
SET m.correlation_type = CASE WHEN ALL(x IN values WHERE x>0) THEN "positive" END
SET m.correlation_type =CASE WHEN ALL(x IN values WHERE x<0) THEN "negative" END;
```

Donor Similarity

```
MATCH (m:Donors)-[r:RELATED_TO]->(n)
WITH n, collect(DISTINCT r.timestamp) AS time
UNWIND range(0,size(time)-1) AS i
MATCH (p1:Donors)-[r1:RELATED_TO{timestamp:time[i]}]->(n)<-[r2:RELATED_TO{timestamp:time[i]}]-(p2:Donors)
WHERE id(p1)<id(p2) AND p1.Name <> 'C' AND p2.Name <> 'C'
WITH sum(r1.CON * r2.CON) AS DotProduct,
sqrt(REDUCE(r1Dot = 0.0, a IN collect(r1.CON) | r1Dot + a^2)) AS r1Length,
sqrt(REDUCE(r2Dot = 0.0, b IN COLLECT(r2.CON) | r2Dot + b^2)) AS r2Length,
p1, p2
MERGE (p1)-[s:donor_similarity]-(p2)
SET s.similarity = DotProduct / (r1Length * r2Length)
```

The outcome of the whole process leads to the introduction of some relationship types to the knowledge graph, that were described in section 4.2. More specifically, the following relationship types were generated through this process: “compound_similarity”, “bio_converged_correlations”, “associated_with”, “early_storage”, “mid_storage”, “late_storage”, “protein_correlations”, “protein_compounds_correlations, “phys_compounds_correlations”, “converged_phys_compounds_correlations” and “donor_similarity”.

4.3.3. Constructing the Knowledge Graph

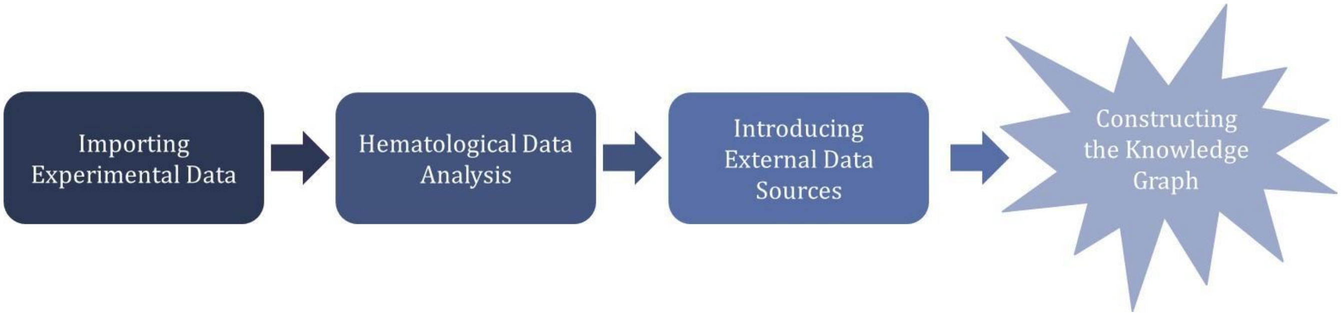

Importing Experimental Data
Hematological Data Analysis
Introducing External Data Sources
Constructing the Knowledge Graph

Figure 20. Schematic representation of the creation of the hematological markers network

By assembling the outcome of what was described in sections 4.2, 4.3.1 and 4.3.2 the final knowledge graph can be generated. We could describe the hematological markers network as a network of two layers. The first layer consists of the preprocessed experimental data along with all correlations that were mentioned in section 4.3.2, while the second layer includes external data sources (nodes, relationships, and properties) that enrich the length and depth of the knowledge graph by adding more detailed information regarding proteins and metabolites related – directly or indirectly – to G6PD.

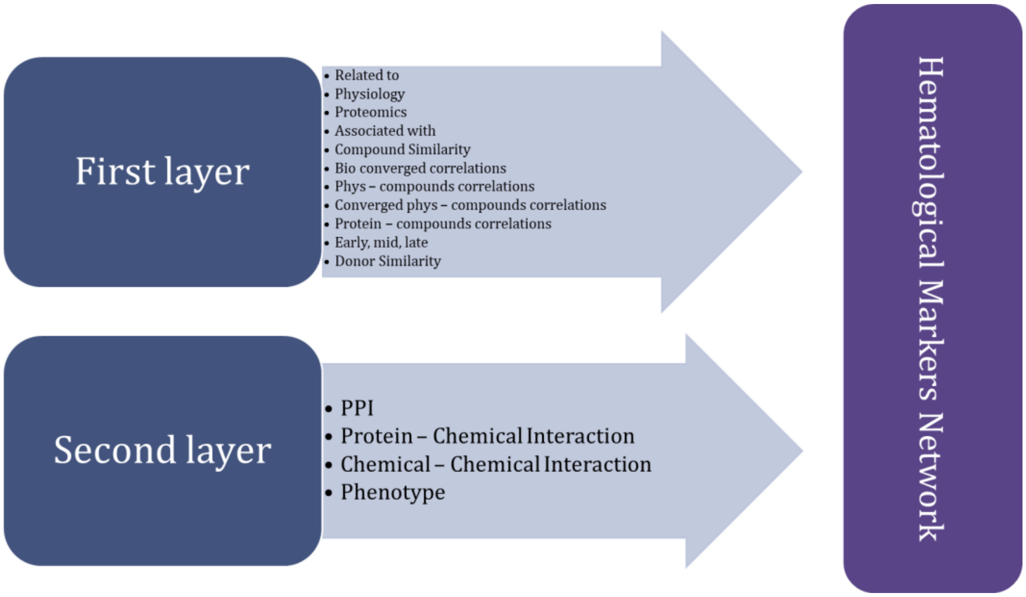

First layer
• Related to
• Physiology
• Proteomics
• Associated with
• Compound Similarity
• Bio converged correlations
• Phys – compounds correlations
• Converged phys – compounds correlations
• Protein – compounds correlations
• Early, mid, late
• Donor Similarity
Second layer
• PPI
• Protein – Chemical Interaction
• Chemical – Chemical Interaction
• Phenotype
Hematological Markers Network

Figure 21. Schematic representation of hematological markers network’s two layers. The first layer includes all relationship types related to the experimental dataset and the second layer concerns all graph data the introduced to the knowledge graph from the external sources that was mentioned in Chapter 3.

5. DATA EXPLORATION AND RETRIEVAL

5.1. Data Analysis and Visualization

Though Neo4j is a suitable tool for the development and exploration of graph data even in large scale, it has limited visualization capabilities, especially for users without an IT background. Therefore, it was necessary to find another browser-based tool to use as the interface for our graph with which any mainstream user could interact with ease. GraphXR was proposed as a suitable tool for this purpose.

GraphXR is a web application specialized in the analysis and visualization of graph data in 2D and 3D space. As a browser-based tool, GraphXR gives the user the ability to navigate and explore any set of graph data through its environment and provides a set of tools and predefined algorithms , that are necessary for the analysis of graph data, and will be described in more detail in the next sections. One of the many assets of GraphXR, besides being user friendly, is the connection it has with Neo4j. That said, the user can link a copy of any working project in Neo4j to GraphXR, without disrupting the original project. Moreover, any new elements passed to the existing network can be saved back to the original project inNeo4j [50].

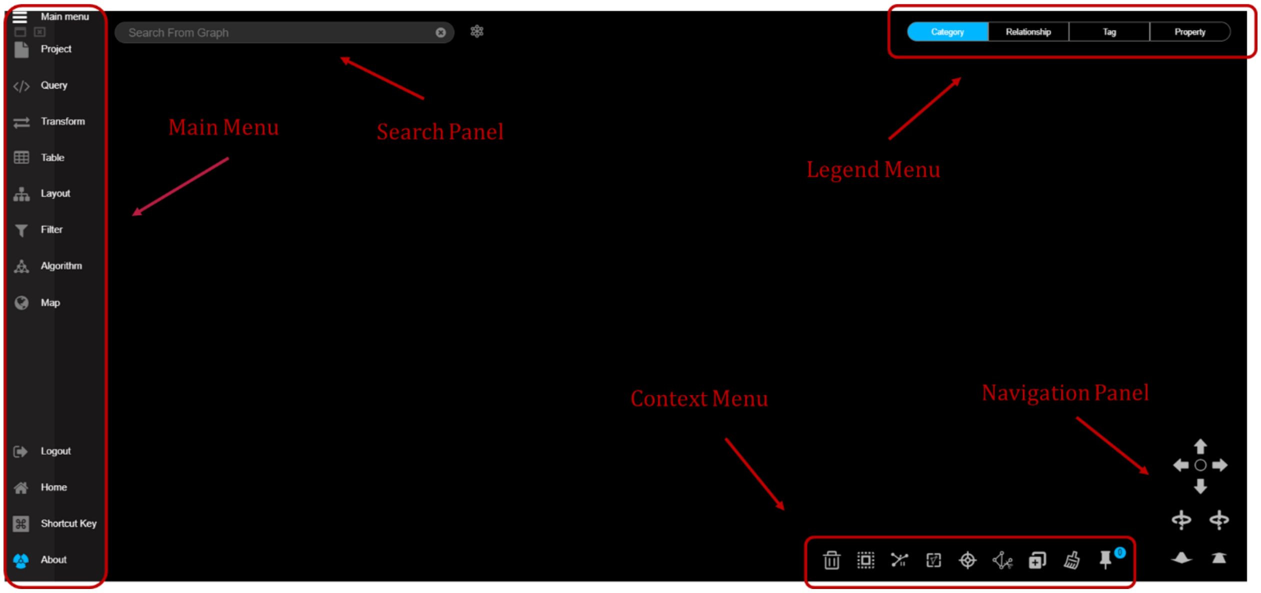


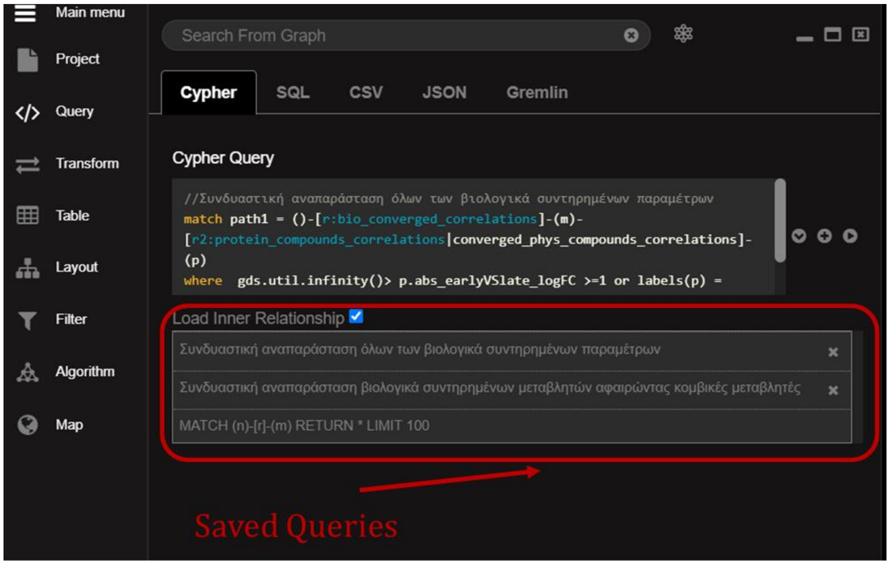
Figure 22. Display screen of GraphXR

Moving on, the display screen along with all possible options that are available to the user will be described in detail. Starting with, in Figure 22 a screenshot of the display screen of GraphXR is shown. On the left of the display screen the Main menu is presented with a set of options that include opening panels, importing, transforming, and displaying data. On the right of the display screen Legend menu (top right), Context menu (bottom right) and Navigation panel (bottom right) are available. Through the Legend menu one can select nodes based on their category, tags or properties and relationships by relationship type, while on the Context menu there are several tools to use for data selection and manipulation in graph space. Finally, through the Navigation panel one can navigate within the graph space in 3D mode, rotate and zoom in/out the graph and with the search panel one can search for nodes or relationships of the graph using specific keywords (e.g. property names or node labels).

Main Menu

The main menu panel is the part of GraphXR that includes most of the tools that are necessary for any kind of data manipulation or visualization in graph scape. It consists the following tabs:

- **Project** panel, which includes Category and Relationship tabs to give the ability to the user to select any node or relationship type, respectively, to be displayed in the graph space. Moreover, it includes the Settings tab with which the user can control the display size of nodes and the width of edges and the final tab of Project Panel comes under the name "Data" and allows the user to import/export data in standard formats (e.g. CSV and GXRF) or save data to Neo4j
- **Query** panel, which enables the use of Cypher and SQL queries or manipulating and saving mappings of CSV files. It is noteworthy that favorite queries can be saved for later use, providing an alternative method for filtering specific nodes or relationships. In Figure 23 an example of the Query panel is available



Cypher Query

```
//Συνδυαστική αναπαράσταση όλων των βιολογικά συντηρημένων παραμέτρων
match path1 = ()-[r:bio_converged_correlations]-(m)-
[r2:protein_compounds_correlations|converged_phys_compounds_correlations]-
(p)
where  gds.util.infinity()> p.abs_earlyVSlate_logFC >=1 or labels(p) =
```

Load Inner Relationship ☑

Συνδυαστική αναπαράσταση όλων των βιολογικά συντηρημένων παραμέτρων ✖

Συνδυαστική αναπαράσταση βιολογικά συντηρημένων μεταβλητών αφαιρώντας κομβικές μεταβλητές ✖

MATCH (n)-[r]-(m) RETURN * LIMIT 100

Figure 23. An example of the Query Panel. Within the red rectangular shape some saved/favourite Cypher queries are shown.

- **Transform** panel, which consists of a set of formulas and data operators that are useful for data transformation. Some of them are responsible for merging nodes with same properties under one node or connecting nodes with same properties values or even providing access to external applications for data gathering and transformation.
- **Table** panel, which presents data that are displayed in graph space, in tables. There are two separate tabs, one for presenting node graph data and another one for displaying relationship data along with their properties. Moreover, the user can

manipulate these tables by selecting, reformatting, or even removing property values and can export the edited tables in CSV format

- **Layout** panel, which provides a set of options in terms of data visualization. That said, graph data can be displayed in “Force”, “Parametric”, “Geometric” or “Tree” layout. Force layout applies a non-canonical shape to the graph and lets the user manipulate the length, the strength and possibly the 3D representation of relationships. Parametric layout shifts the shape of the graph by determining the 3D space using specific node properties. Geometric layout forces the network to shapeshift by applying geometric shapes, such as circular or cubic shape and at last, tree layout applies a hierarchical shape to the graph, making it easier to distinguish root and leaf nodes.
- **Filter** panel, which hides graph data by setting thresholds to one or more node or relationship properties.
- **Algorithm** panel, which contains a set of the most popular graph-related algorithms, such as Degree, Closeness, Betweenness or Community Detection. Each one of them will be discussed in more detail in section 5.2.
- **Map** panel, which is useful for cases of analysing geospatial data on a world map.

Legend Menu

As it is already mentioned the legend menu, which is displayed in the top right of the display screen, allows the user to select specific graph data based on node label (category) or property values of choice and relationship by relationship type. Doing that, one can easily subtract selected data or hide the rest of them to focus on specific regions of the graph.

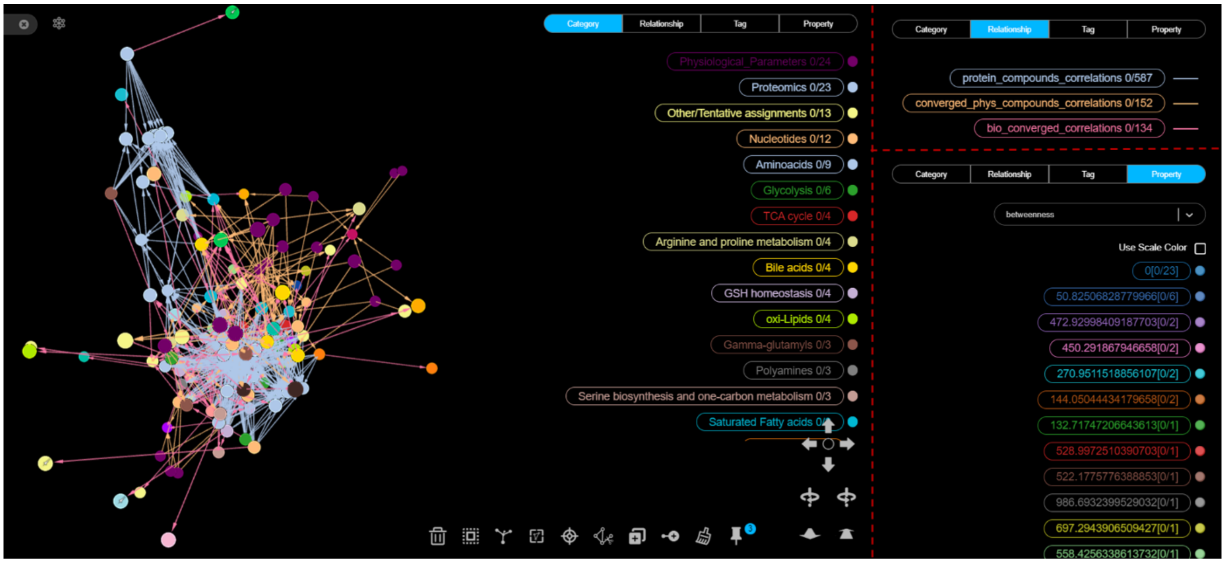


Figure 24. An example of the tools that are available in the Legend menu.

In Figure 24 an example of the options that are available in the Legend menu is presented. At first, by selecting the Category tab, a list with all node labels is expanded and each node type is dashed with a different color. Moreover, the population of each node type is recorded as well. The Relationship tab shows the relationship types that are presented in the displayed graph. Edges of each relationship type are dashed with different colors. At last, the Property tab allows to group nodes by the property of choice (in this case betweenness score). That said, nodes with the same property values will be marked with the same color. Additionally, in each of the cases presented in this example the user can select one or more groups to hide or display just by clicking on them and subsequently use some of the filters in the Context menu, that will be described later in detail.

Context Menu

The context menu, as it was mentioned above, is located on the bottom right of the display screen and contains a set of tools for manipulating data displayed on the graph space. In Figure 25 the tag name of each tool of the Context menu is presented. The use of each tool is described below:

- By clicking on the **Info** tag information about selected nodes will pop up
- **Trace Neighbor** allocates up to the n-th neighbor of a selected node
- With option **Tag** the user may add new properties to selected nodes
- **Delete** erases selected nodes from the graph space
- By clicking **Expand** more existing – but not currently visible – relationships related to selected nodes are introduced to the graph space
- With **Inverse**, nodes, currently unselected, are selected. The number on the top of the icon shows how many nodes are currently selected.
- **Hide selection** hides selected nodes along with their edges
- **Select Visible Nodes** can be applied while no nodes are selected. On that occasion, all displayed nodes will be selected
- **Enable/Disable Force Layout** applies or removes the force layout
- **Fly Out/Center To** zooms in or out on the center if no nodes are selected. Otherwise zooms in or out in respect to the selected nodes
- With **Leaf Trimming** nodes with zero or one relationship are deleted from the graph space
- By clicking **Add Node/Edge** one can introduce new nodes/edges to the displayed graph
- **Clear** removes all graph data from the display screen
- With **Pin** selected nodes freeze to their current 3D location and the graph can be rotated around them. **Release** clears pinned nodes. The number on the top of the icon displays how many nodes are pinned.

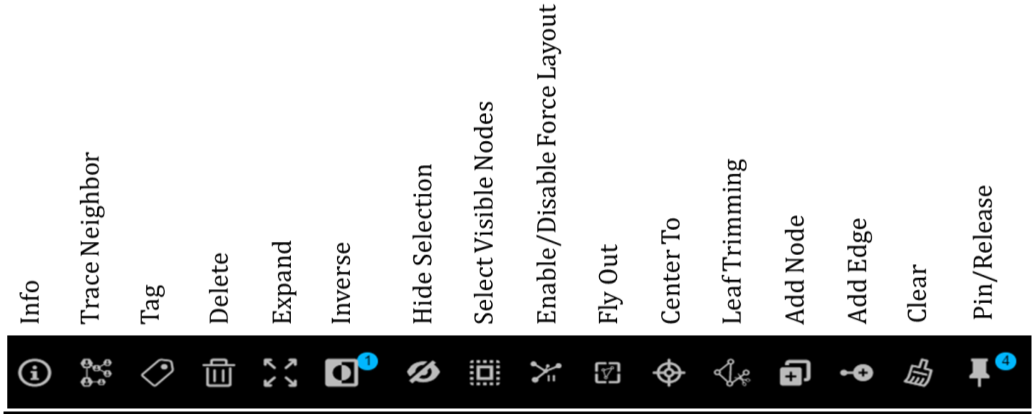
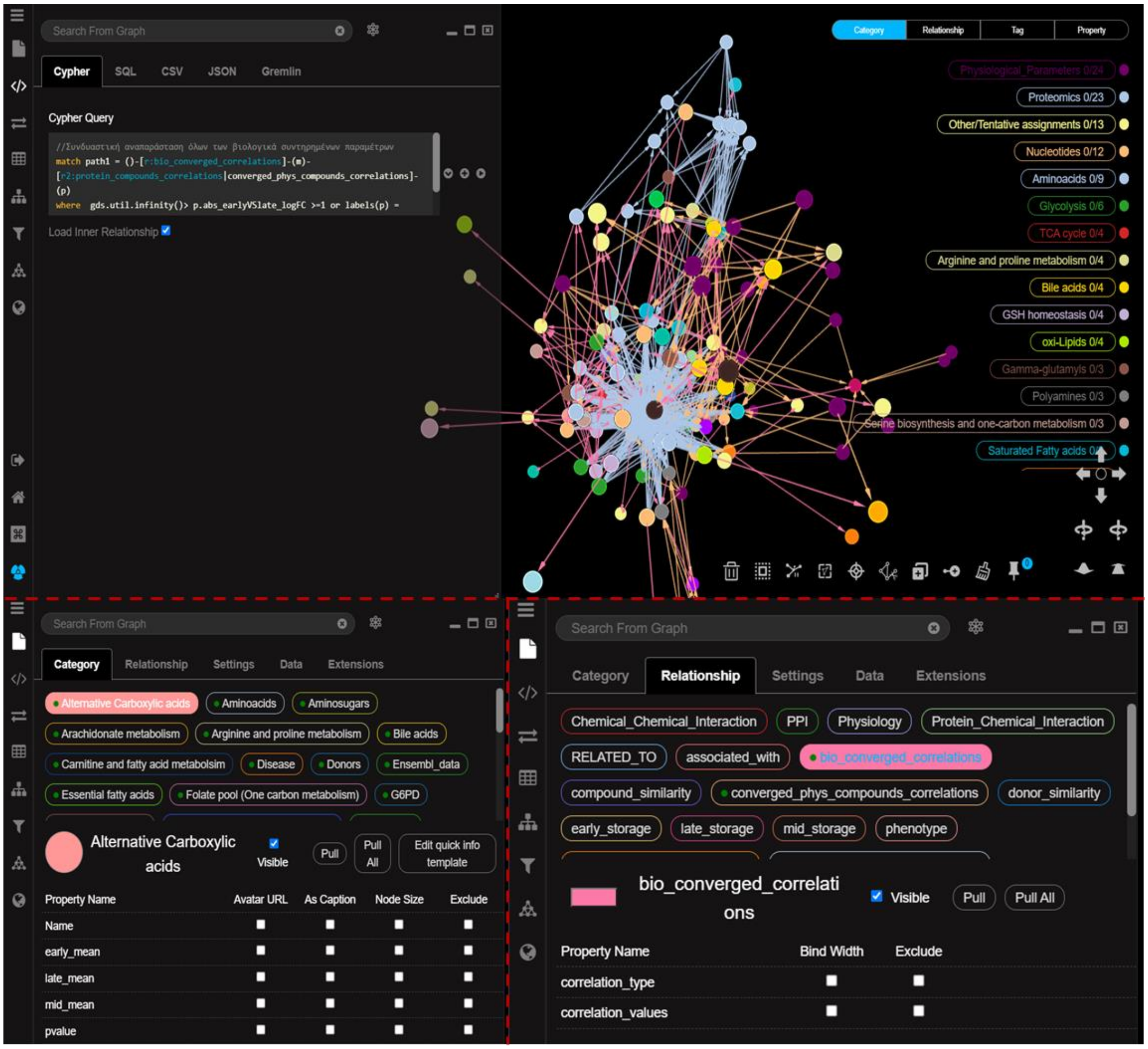


Figure 25. The Context menu of GraphXR.

Taking under consideration all available tools that described above one can proceed with the visualization and further analysis of any graph data. In GraphXR nodes are presented as circles and relationships as edges that connect two nodes. Detailed information about properties of any node is accessible through the Legend menu, as described above, or by double-clicking the node of interest. That said, there are two possible ways to display a network on the graph space. The first one is through the Query panel of the Main menu by executing a query that returns the desired network. The second option includes the use of the Project panel of the Main menu.

In Figure 26 an example of how to display a network in graph space is shown*.* On the top of the figure one can observe the network of biologically converged parameters (metabolites, physiological parameters, and proteins), as it was generated using the Query tab of the Main menu. Part of the query that results in the displayed graph is also shown on the Cypher Query panel. Nodes are dashed in different colors according to the node type they belong to. The same goes for relationships too. On the bottom of the figure one can observe the way to display this network using the Project panel of the Main menu. At first by clicking on the Category tab the user can specify the node types that want to be displayed on the graph space. That said, the user needs to click any node type and be sure to check the box with the description “Visible”. By doing that a small green circle will be displayed on the left of the selected node type, which means that it can be presented on the graph space. Next, by clicking the Pull or Pull All button the selected node types will be displayed on the graph space (bottom left). To show any relationship regarding the selected nodes one needs to work accordingly (bottom right).



```
//Συνδυαστική αναπαράσταση όλων των βιολογικά συντηρημένων παραμέτρων
match path1 = ()-[r:bio_converged_correlations]-(m)-
[r2:protein_compounds_correlations|converged_phys_compounds_correlations]-
(p)
where  gds.util.infinity()> p.abs_earlyVSlate_logFC >=1 or labels(p) =
```

Property Name	Avatar URL	As Caption	Node Size	Exclude
Name	□	□	□	□
early_mean	□	□	□	□
late_mean	□	□	□	□
mid_mean	□	□	□	□
pvalue	□	□	□	□

Property Name	Bind Width	Exclude
correlation_type	□	□
correlation_values	□	□

Figure 26. An example of the way to visualize a network in the graph space of GraphXR.

5.2. Networks' Centralities and Communities

One of the most common – yet of major importance – procedures in graph analytics, once a network is fully established, concerns the calculation of several centrality metrics, as well as the estimation of community formations (also known as clusters). At first, using centrality algorithms to any kind of graph could highlight the most important nodes and give insight about the dynamics of the network, such as its spreadability, consistence and credibility. On the other hand, community detection algorithms could help us identify strongly connected nodes, discriminate those that are more isolated and subsequently focus on specific clusters based on our interest and design a more detailed analysis about them [51].

Calculating Centralities

Former analyzes regarding RBC interaction networks suggest that the following centrality algorithm are particularly important to identify the most popular nodes of the graph: Betweenness Centrality, Degree Centrality and Closeness Centrality algorithm [37]. Taking that into account we proceeded with the calculation of centrality metrics for the hematological markers network.

To begin with, by estimating the **Betweenness Centrality (BC)** of a network one can get insight about the influence of a node over the spreadability of the information in a network. In practice, the power of a node is estimated as the number of shortest paths, between all possible groups of nodes, in which a node is part of. With the term "shortest path" we refer to the best path that connects two nodes in a graph by minimizing the cost [51]. In Figure 27 a toy example explaining the term of shortest path is presented. In this figure one can easily notice that there are two alternative paths from node A to node D. The 1st path includes nodes A, B and D and the 2nd path consists of nodes A, C and D. However, to identify the optimal path, or else the shortest path, one needs to take under consideration the weight of the edges. That said, on the 1st path the total weight is 15, while on the 2nd path the total weight is 12. Therefore, the 2nd path can be characterized as the shortest path from node A to node D.

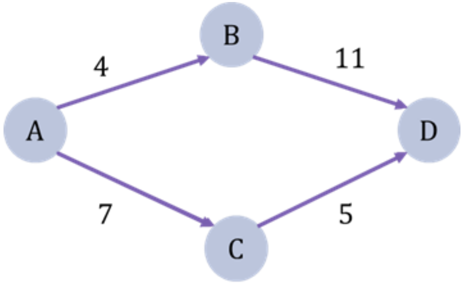


Figure 27. Explaining the term of shortest path.

Taking that into account, the mathematical equation that best describes the Betweenness Centrality algorithm is the following:

$BC(u)=\sum_{s\neq v\neq t}\frac{\sigma_{st}(u)}{\sigma_{st}}$, where σ_{st} is the total number of shortest paths from node s to node t and $\sigma_{st}(u)$ is the number of shortest paths from node s to node t to which node u is part of. The greater the BC measure of a node the more influence the network has.

Figure 28 shows an infographic example for better understanding the BC algorithm. More specifically, the figure describes the calculation of BC of node E in respect to the shortest paths from node A to node F. To calculate the BC of node E we need to take under consideration all shortest paths that start from node A pass-through node E and reach node F and divide them by the total amount of shortest paths from node A to node F. The total amount of shortest paths from node A to node F is 4, and those that pass-through are 3. Therefore, the BC score of node E is $\frac{3}{5}$, or else 0.60.

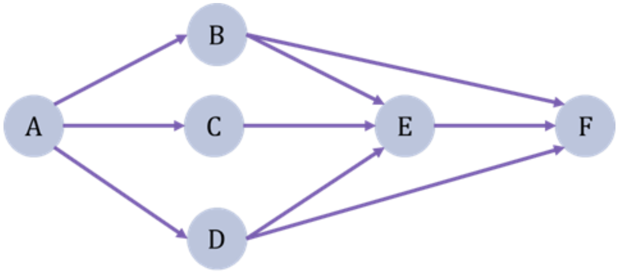


Figure 28. Toy example: BC of node E in respect to the shortest paths from node A to node F.

Degree Centrality (DC) of a node is just the number of relationships concerning that node. Therefore, if a node has five relationships, its DC score will be five. Sometimes for normality reasons we tend to divide the DC score with the largest DC score that occurs in a network, which belongs to the node with the most relationships [51]. Though, DC is probably the simplest and fastest centrality algorithm, the importance of the results is not always significant. For instance, a node can still be disconnected from an important part of the network, besides the fact that it might have a high DC score [37].

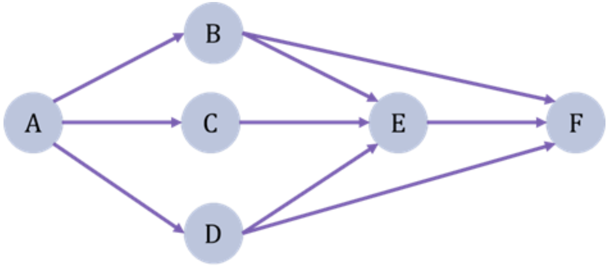


Figure 29. Toy example: DC of node F. By taking a close look to the figure on the left one can notice that node F has three incoming and no outgoing relationships. The total number of relationships in this network is nine. Therefore, the DC score of node F is DC(F)=3.

At last, the **Closeness Centrality (CC)** algorithm detects those nodes that are related with increasing the spreadability of information to the network, or else they have the shortest distances to all other nodes. In practice, a node with a high CC score is more central to the graph and "closer" to other nodes. As a measure, Closeness Centrality far more accurate than Degree Centrality since CC compares the relationships of node with the entire network [37]. The mathematical equation that best describes CC algorithm is shown below:

$CC(u)=\frac{1}{\sum_{i=1}^{n} d(u,y_i)}$, where n stands for the number of nodes of a network and $\Sigma d(u,y_i)$ is the sum of distances of node u to the rest of the nodes (y_i) and $u \neq y_i$. The outcome is bounded in [0,1]. In many cases it is quite common to use the normalized version of the algorithm, which represents the average length of shortest paths rather than the sum. This modification also allows the comparison of CC scores of nodes of graphs of different sizes. That said, the updated mathematical formula regarding CC algorithm is the following:

$$\mathrm{normCC}(u)=\frac{\mathrm{n-1}}{\sum_{\mathrm{i=1}}^{\mathrm{n}}\mathrm{d}\left(\mathrm{u,y_i}\right)}$$

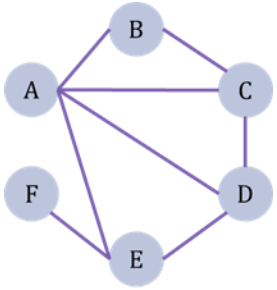
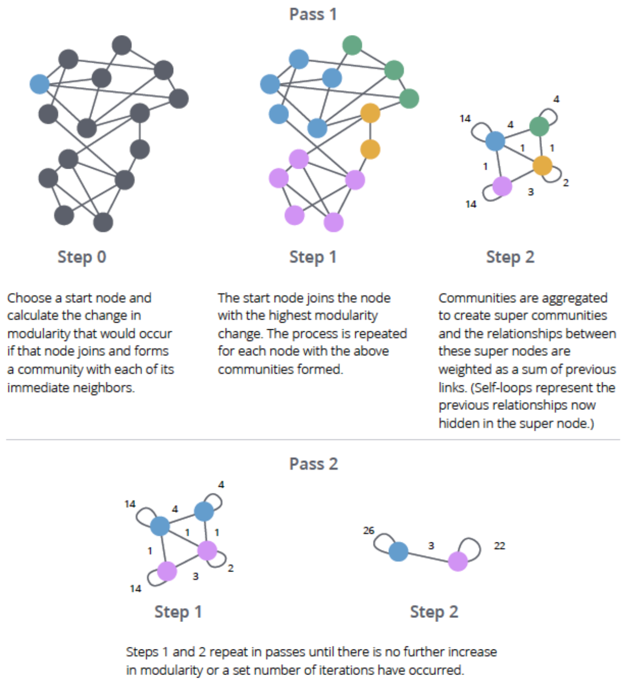


Figure 30. Toy example: normalized CC of node A. In the figure on the left one can easily observe that the distance of node A to the rest of nodes is one with the only exception being the distance from node A to node F, which is two since they are connected through an intermediate component (node E). That said the normalized CC of node A is: $\boldsymbol{normCC\ (A) = \frac{6-1}{1\cdot4+2} = 5/6 \approx 0.83}$

Detecting Communities

The concept of community is quite regular in data analysis and it is related to the classification of data in groups for characterization purposes or retrieving additional information. In graph analytics a community is defined as a subset of nodes inside a network with more dense connections between them than those formed with the rest of the graph [52]. In biological networks the identification of metabolite communities could correspond to metabolic pathways, while clusters of proteins could highlight potential biological interactions or effect on the same biological process [53].

One of the most popular algorithms for detecting communities is the **Louvain method.** The concept of the method is an optimization of the modularity metric. Modularity as a measure is related with the capability of a network to form clusters. Therefore, networks with high modularity form highly connected communities with sparse connections between nodes of other modules [54]. Modularity measure is bounded in [-0.5, 1]. Since the application of modularity measure is quite expensive in large networks, a more heuristic approach is used in the Louvain method to optimize the modularity score of each cluster. That said, the algorithm starts by computing the modularity measure of small communities. Next each small cluster is grouped into one node and the process is repeated until the modularity of each module is maximized [55]. Overall, the Louvain method for detecting communities is one of the fastest modularity-based algorithms with high performance even in large networks.



Choose a start node and calculate the change in modularity that would occur if that node joins and forms a community with each of its immediate neighbors.

The start node joins the node with the highest modularity change. The process is repeated for each node with the above communities formed.

Communities are aggregated to create super communities and the relationships between these super nodes are weighted as a sum of previous links. (Self-loops represent the previous relationships now hidden in the super node.)

Steps 1 and 2 repeat in passes until there is no further increase in modularity or a set number of iterations have occurred.

Figure 31. Explaining the Louvain method for detecting communities. [56]

Another well-known algorithm that is widely used for detecting communities in graphs is the **Connected Components (CC)** algorithm. In practice, this algorithm is used for the identification of clusters in *undirected* graphs and considers as a set of connected components a subgraph in which there is a path to every pair of nodes inside the subset [51]. In Figure 32, an infographic example that explains the idea of CC algorithm is presented. To begin with, nodes dashed in bordeaux indicate that they are not visited by the algorithm, while those marked with purple have been visited by the algorithm. The process starts randomly from any node of the graph and the component counter is set to zero (1). In this case and for simplicity reasons we will start from node A and continue in ascending order. At first, the algorithm visits node A and all adjacent nodes are also considered as visited. After checking all adjacent nodes, the component counter is set to one (2). Next the algorithm checks if node D is already visited or not. Since it was not visited the algorithm repeats the process of step 2 and checks all adjacent nodes of node D and sets the component counter to two (3). Moving on, the next node that is checked by the algorithm node F. By repeating the process, the component counter is set to 3 (4). The last node that is visited by the algorithm is node H. Node H has no adjacent nodes, thus it will be considered as a separate component. The component counter concludes to 4 (5). Once all nodes have been visited the algorithm a component id is applied to each node based on the community they belong to, and they are dashed accordingly (6).

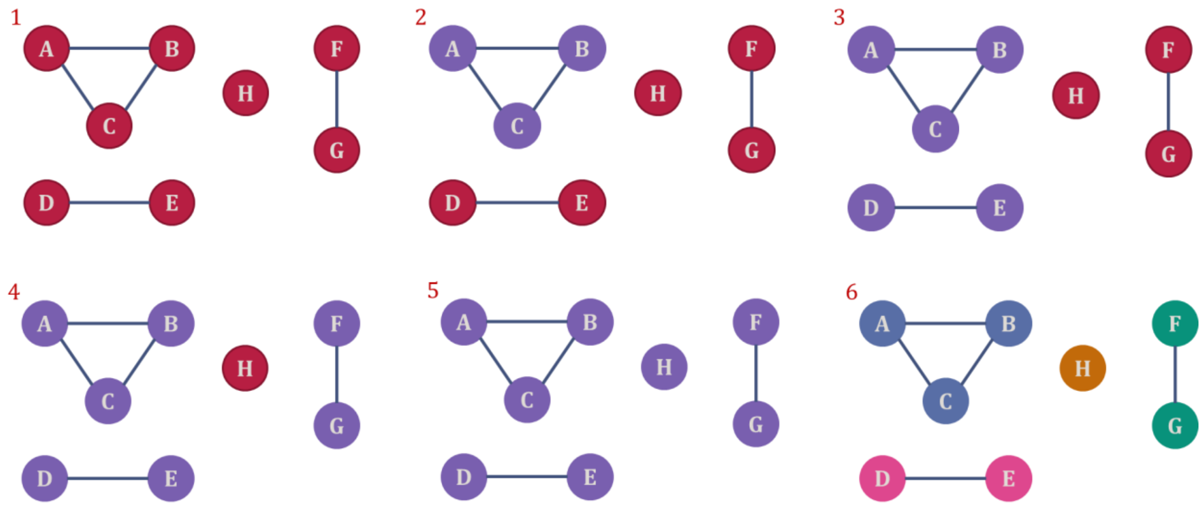


Figure 32. Explaining Connected Components algorithm.

An alternative method that is mostly applied to *directed* graphs is the **Strongly Connected Components (SCC)** algorithm. Similarly to the Connected Components algorithm a group of nodes is considered strongly connected if there is a path between each pair of the group. However, in this case the path needs to be directed.

The last algorithm that is going to be discussed in this section is the **Label Propagation algorithm (LPA)**. Label Propagation is a fast algorithm for detecting communities in graphs. In this case communities are identified based on the structure of the graph and without having any prior knowledge about them. However, it can be used also in a semi-supervised manner by assigning initial labels to some nodes to reduce the proposed solutions. Though LPA performs very well on densely connected graphs, it seems that detecting communities in sparsely connected graphs is quite troublesome for this algorithm, since some nodes will tend to be trapped inside a densely connected group, resulting to mislabeling them [51].

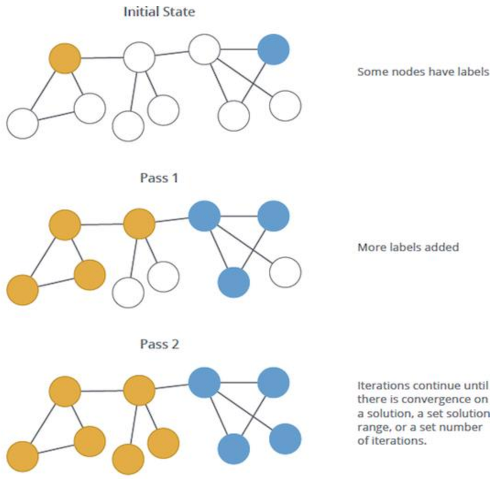

Initial State
Some nodes have labels
Pass 1
More labels added
Pass 2
Iterations continue until there is convergence on a solution, a set solution range, or a set number of iterations.

Figure 33. Explaining Label Propagation algorithm [57].

Case study using GraphXR:

In this section a complete walkthrough regarding the calculation of centralities, the detection of communities, and the representation of results related to this analysis will be presented. Starting with, we need to define the graph with which we will work on. For this case the network of biologically converged components will be used as the case study. The part of displaying the graph has been described in detail in section 5.1. That said, we can move with computing some centrality metrics for this graph. Thus, we need to choose the Algorithm tag from the Main menu and then choose the desired centrality algorithm. The computation is automated, and the result will be presented as a new property for each node displayed on the graph space.

In Figure 34 a case study of computing BC scores in GraphXR is presented. This example focuses on estimating the BC of biologically converged components. To calculate the BC for each node one needs to start by clicking the button under the name Betweenness, which is available on the Centrality tab of the Algorithm panel of the Main menu. Once this is done, the computation of BC scores starts. The appearance of a green box with the message “Calculation finished” will be shown in the center of the display once the computation is done. The BC measure of each node has been passed as a new property under the name of the algorithm (in this case “betweenness”) and it is easily accessible through the Legend Menu by clicking the Property tab and then selecting the property related to betweenness scores. As we can see on the right of the figure above, nodes have been grouped and marked under different colors according to their BC value. This is a quite important feature of GraphXR since it gives the user the option to display nodes with specific BC scores and thus distinguish t the ones that may seem more significant.

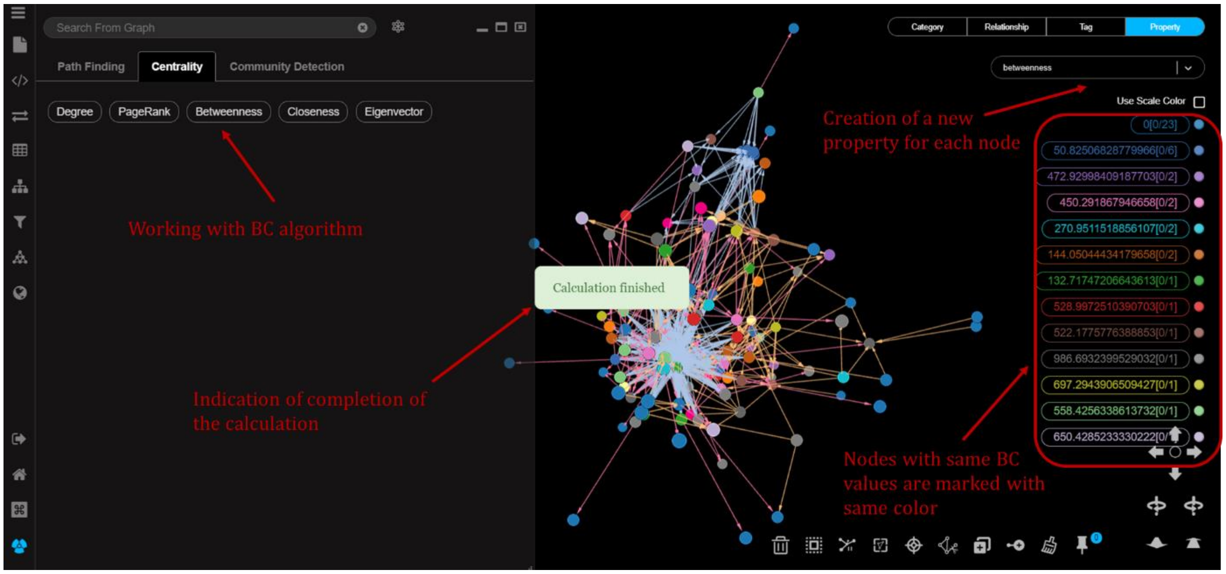
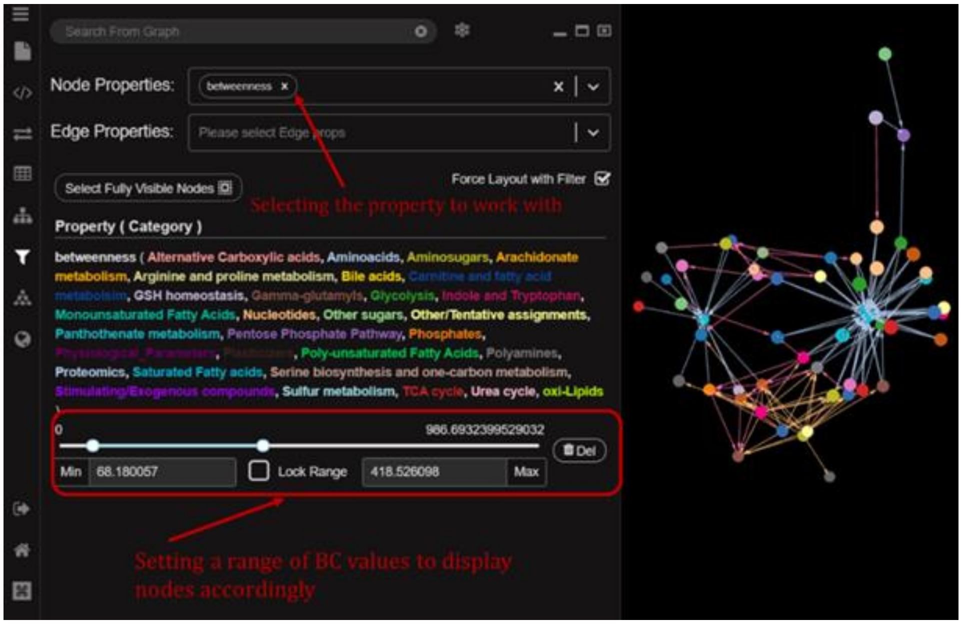


Figure 34. Computing BC scores for biologically converged components.

Once the computation is finished one can choose to display only nodes with a specific range of BC score and exclude the rest of them. This process can be easily executed in GraphXR with two different ways, either using the Filter panel of the Main menu or by combining the Property tab of the Legend menu and the Context menu.

In Figure 35 an example of the first approach is explained in detail. Starting with, Through the Filter panel one can choose to display only the nodes or relationships with a specific range of values of a chosen property. For this case the property of interest is the one concerning the BC scores. By specifying the property name a scale bounded with the minimum and maximum values of the selected property appears. The user can manipulate the limits of this scale. The outcome of this process is to display only nodes with the specified range of BC values. In this case the limits have been set from 68.18 to 418.52.


Search From Graph
Select Fully Visible Nodes
Del
Selecting the property to work with
Setting a range of BC values to display nodes accordingly

Node Properties: betweenness

Edge Properties: Please select Edge props

Force Layout with Filter ☑

Property (Category)

betweenness (Alternative Carboxylic acids, Aminoacids, Aminosugars, Arachidonate metabolism, Arginine and proline metabolism, Bile acids, Carnitine and fatty acid metabolism, GSH homeostasis, Gamma-glutamyls, Glycolysis, Indole and Tryptophan, Monounsaturated Fatty Acids, Nucleotides, Other sugars, Other/Tentative assignments, Panthothenate metabolism, Pentose Phosphate Pathway, Phosphates, Physiological_Parameters, Plasticizers, Poly-unsaturated Fatty Acids, Polyamines, Proteomics, Saturated Fatty acids, Serine biosynthesis and one-carbon metabolism, Stimulating/Exogenous compounds, Sulfur metabolism, TCA cycle, Urea cycle, oxi-Lipids)

0 986.6932399529032

Min 68.180057 □ Lock Range 418.526098 Max

Figure 35. Displaying selected nodes according BC values using the Filter panel of the Main menu.

In Figure 36, an example of the second approach is presented. In this case to present the final output on the display screen a three-step process takes place. At first, through the Legend menu the user, by choosing the property related to BC scores, reveals a list of groups of nodes categorized by their BC measure. By browsing to this list one can select groups of nodes with a desired betweenness score (1). The next step includes the use of Context menu. Through the Context menu the user needs to click on the Inverse tag to select the rest of the nodes (2). The process concludes by clicking the tag Hide Selection of the Context menu. By doing that, the nodes that were selected in the previous step are now hidden from the graph space and only initially selected nodes with the desired BC measures are displayed on it (3). The advantage of the second way of filtering nodes according to BC values is the fact that the user can manually select groups of nodes with desired property values and display only them.

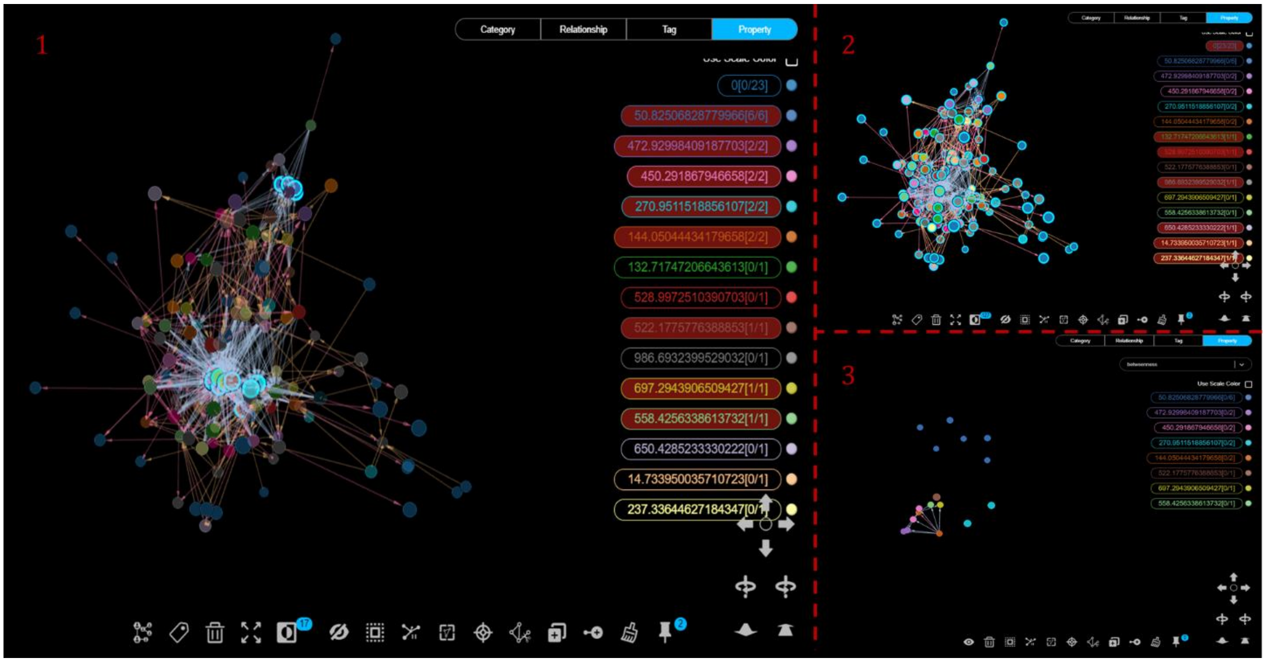


Figure 36. Displaying selected nodes according BC values using the Legend Context menu.

Both approaches that were described above point out a series of events that need to take place for the calculation of any centrality measure or the detection of communities. The only difference is that in the case of detecting communities in a graph, the resulting node properties correspond to the community to which the node belongs.

5.3. Exploratory Analysis

In this section a step-by-step representation of several data exploration cases will be provided. The examples that will be presented are directly related to the biological questions that were introduced in chapter 4. The main objective of this section is to gain insight about the applications of what was mentioned in the previous sections of this chapter, in the biological problem that is studied during this work.

A. Exploring inter- and intra- parameter associations

Since the hematological markers network was set up to investigate homologous and heterologous correlations between different components and to answer to a set of biological questions related to this biological problem, a first approach regarding the exploration analysis could be to spectate specific relationships of the graph at will, depending on the question we want to answer. That said, a good example to start with could be the discovery of G6PD-related metabolites along with compounds that are highly correlated with (also known as first neighbors in graph analytics). An explanatory walkthrough of that is available on Figure 37. The first step towards the identification of metabolites associated with G6PD and the compounds they are highly correlated with is to select to display via Project panel of the Main menu only relationship types regarding: a) G6PD-related components (relationship type: associated with) and b) biologically converged metabolites (relationship type: bio converged compounds).

By doing that only connections of those two types will be presented on graph space (1). Following that, through the Relationship tab of the Legend menu we select the relationship type of G6PD-related components and then we click on Select Visible Nodes of the Context menu. This step concludes by setting the value on the Trace Neighbor tag to one (2), so that we can also pick compounds highly correlated to G6PD-related compounds (3). Once all necessary components and edges of the network have been selected, we move on with removing the rest of the data from the graph space. To do that, we need to click on the Inverse tag of the Context menu, so that data disregarding G6PD-related components and their associates will be chosen (4). The process concludes by using the Hide Selection tag of the Context menu. By doing that only compounds related to G6PD along with their first neighbors are shown on the graph space. All nodes are dashed with different colors according to the node type to which they belong. Moreover, the shorter the length of an edge between two components, the more significant their connection is (5). An alternative option of what was described above could be to use the Query panel of the Main menu and display the desired graph using Cypher queries.

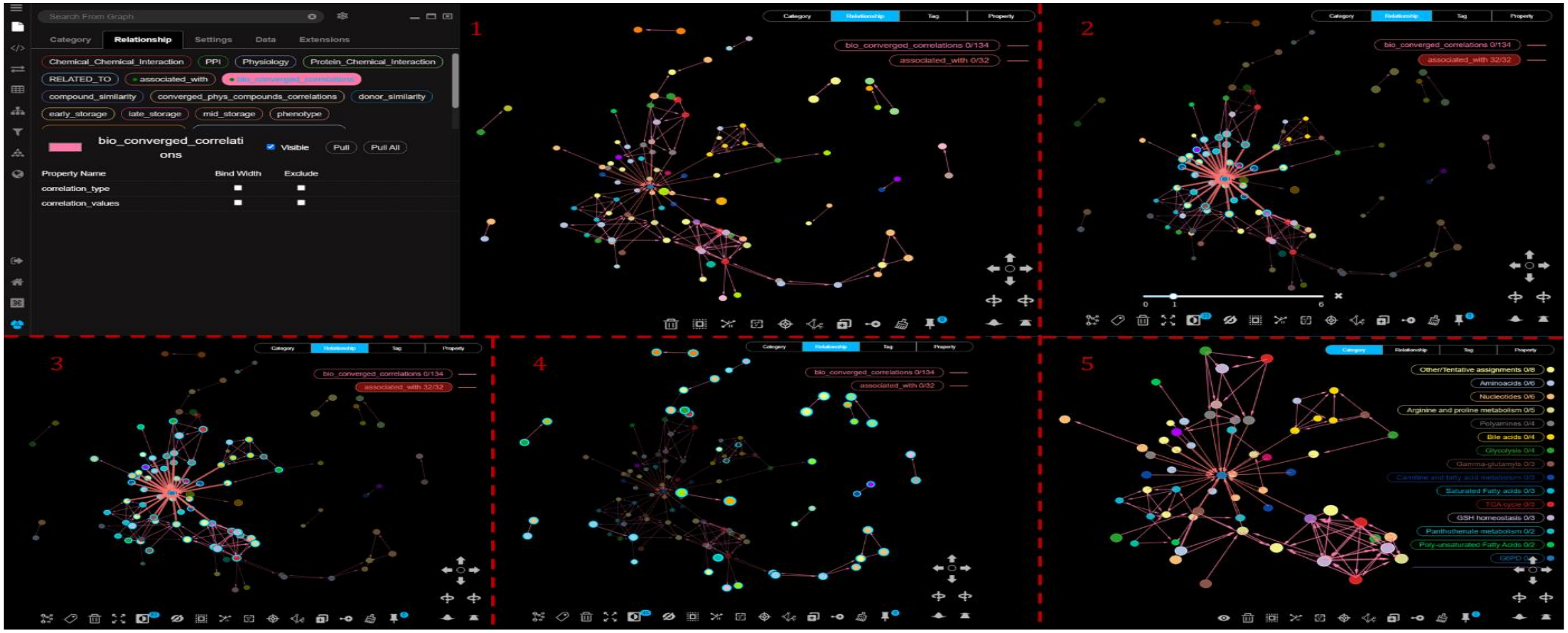


Figure 37. Discovery of G6PD-related metabolites along with their first neighbors.

Another interesting approach, that is displayed in Figure 38, could be to display the map of G6PD-related proteins and diseases or any pathological phenotype with which they are associated, since it might shed some light on their contribution to the development of a disease or highlight potential functional relationships. To display this network, we need to select the relationship types phenotype and protein correlations from the Project panel of the Main menu. Since the displayed graph is relatively small, we can choose a different layout for this case to which we can add some additional information. That said, the specific network is presented in Circular layout and moreover for all nodes under the label Disease the name of the disease is presented, while for proteins (nodes types: Ensembl_data, Proteomics, G6PD) the UniProt Accession Number (UniProtAC) is displayed. Edges marked in orange color indicate association with disease, while blue-colored edges suggest protein correlations.

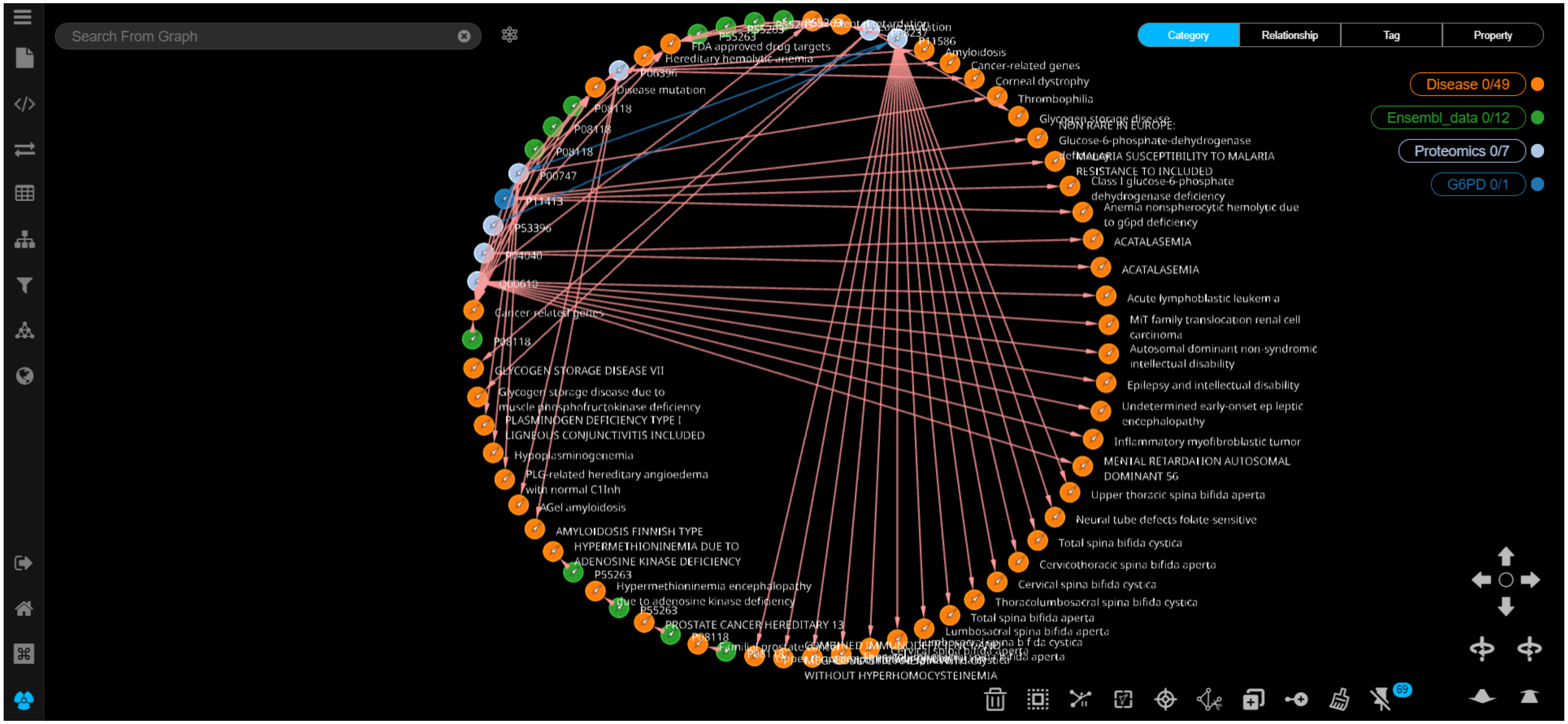
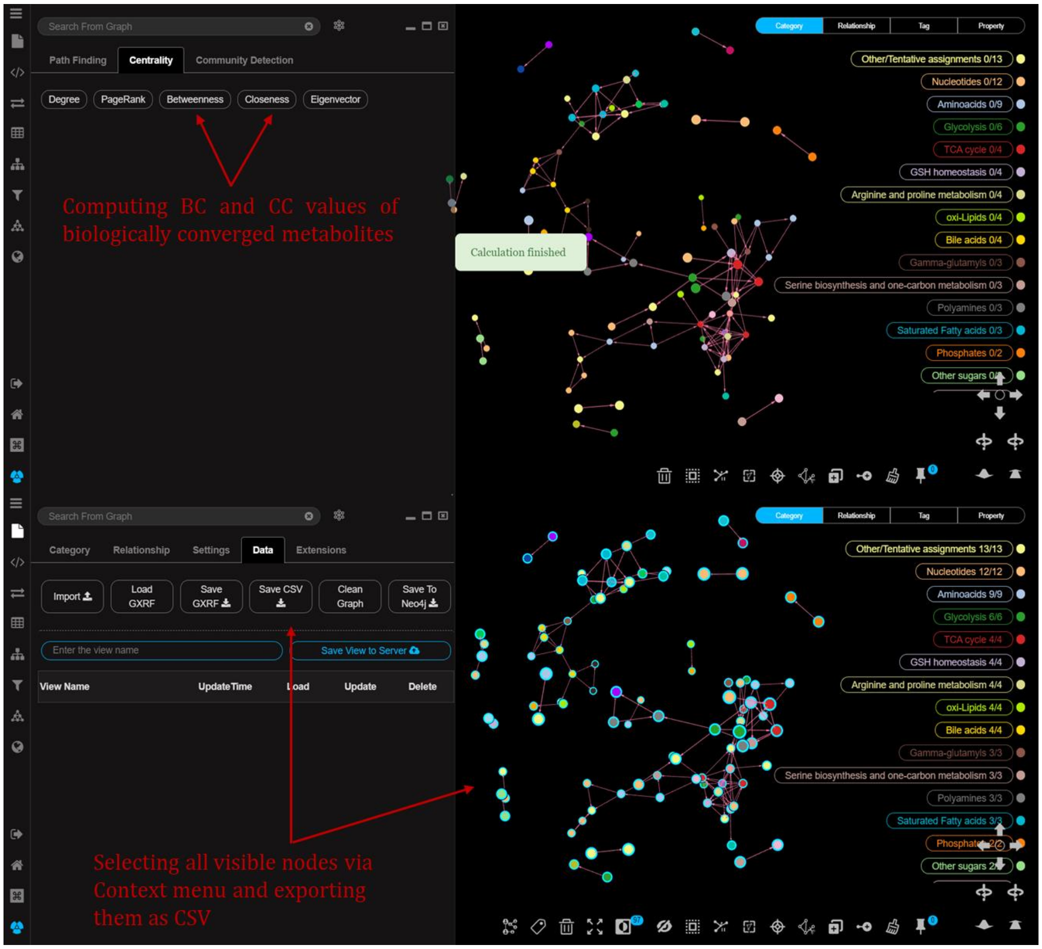

Search From Graph
Category
Relationship
Tag
Property
Disease 0/49
Ensembl_data 0/12
Proteomics 0/7
G6PD 0/1
P55263
FDA approved drug targets
Hereditary hemolytic anemia
P06396
Disease mutation
P08118
P08118
P08118
P00747
P11413
P53396
P04040
Q00610
Cancer-related genes
P08118
GLYCOGEN STORAGE DISEASE VII
Glycogen storage disease due to muscle phosphofructokinase deficiency
PLASMINOGEN DEFICIENCY TYPE I
LIGNEOUS CONJUNCTIVITIS INCLUDED
Hypoplasminogenemia
PLG-related hereditary angioedema with normal C1Inh
AGel amyloidosis
AMYLOIDOSIS FINNISH TYPE
HYPERMETHIONINEMIA DUE TO ADENOSINE KINASE DEFICIENCY
P55263
Hypermethioninemia encephalopathy due to adenosine kinase deficiency
P55263
PROSTATE CANCER HEREDITARY 13
P08118
WITHOUT HYPERHOMOCYSTEINEMIA
P11586
Amyloidosis
Cancer-related genes
Corneal dystrophy
Thrombophilia
Glycogen storage disease
NON RARE IN EUROPE:
Glucose-6-phosphate-dehydrogenase
MALARIA SUSCEPTIBILITY TO MALARIA
RESISTANCE TO INCLUDED
Class I glucose-6-phosphate dehydrogenase deficiency
Anemia nonspherocytic hemolytic due to g6pd deficiency
ACATALASEMIA
ACATALASEMIA
Acute lymphoblastic leukemia
MiT family translocation renal cell carcinoma
Autosomal dominant non-syndromic intellectual disability
Epilepsy and intellectual disability
Undetermined early-onset epileptic encephalopathy
Inflammatory myofibroblastic tumor
MENTAL RETARDATION AUTOSOMAL DOMINANT 56
Upper thoracic spina bifida aperta
Neural tube defects folate-sensitive
Total spina bifida cystica
Cervicothoracic spina bifida aperta
Cervical spina bifida cystica
Thoracolumbosacral spina bifida cystica
Total spina bifida aperta
Lumbosacral spina bifida aperta
69

Figure 38. Associations of G6PD-related proteins with pathological phenotypes.

B. Determination of crucial parameters

The purpose of this analysis is to highlight the most popular components of any case-study network displayed on graph space. To achieve that, we need to work with several centrality measures of the network, so that any finding, that might be derived, would be more trustworthy. A good example could be to identify the most crucial components concerning the metabolic profile or their interconnections with the physiological and proteomic profile of G6PD^{-} donors. That said, a complete walkthrough of the identification of the most significant parameters would be presented below. For the characterization of such components the betweenness (BC) and closeness centrality (CC) metrics were used as a guide. Resulting BC and CC values of the case-study network will be further investigated by exporting the findings of this analysis and visualizing them using more responsive techniques, such as heatmaps.

To begin with, a step-by-step representation of retrieving BC and CC measures of biologically converged metabolites and subsequently identifying the most significant ones via visualization techniques is available below.



Computing BC and CC values of biologically converged metabolites

Calculation finished

Selecting all visible nodes via Context menu and exporting them as CSV

Figure 39. Identifying the most significant metabolites – Part I.

Data retrieved from the procedure described in Figure 39 are stored in CSV files. It is important to mention that each node type is stored in a different file. Therefore, before proceeding with the visualization of results it is necessary to combine all CSV files into one. After that, some preprocessing took place such as normalizing the scale of BC values, so that is bound to [0,1], and setting a threshold of 0.05 to centrality measures to exclude components with insignificant betweenness and closeness values. That said, in Figure 40 the output of most significant metabolites is presented. Metabolites are considered crucial for the network since it has relatively high BC and CC scores. Such components could be characterized as central nodes of the biologically converged compounds, indicating that they might play some role in the metabolic profile of $G6PD^-$ donors. However, to prove such findings more experiments need to take place.

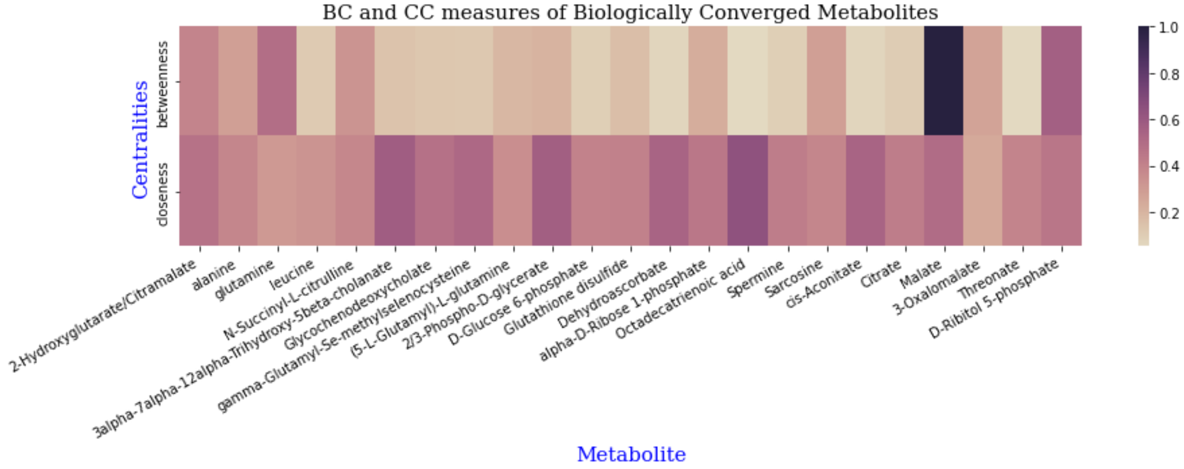


Figure 40. Identifying the most significant metabolites – Part II.

A more informative example of such an exploration analysis is the characterization of crucial parameters amongst metabolites and physiological parameters. Since the parts of displaying the graph, computing centralities, retrieving data, pre-processing, and preparing for visualization via heatmaps are similar to the first case, we will focus and subsequently discuss the outcome of the analysis.

In Figure 41 the outcome of the exploration analysis, that was conducted for the characterization of the most significant G6PD-related components, is presented. One can easily notice that even though most of the displayed parameters have similar closeness values, some of them can be distinguished as more noteworthy due to their high betweenness measure. More specifically, mechanical fragility (MFI and MFI_37), osmotic fragility (MCF and MCF_37) and antioxidant capacity (TAC and TAC_UA) of RBCs seem to be these parameters that are more central to the network. This finding depicts some of the primary characteristics of RBCs, which are related to their sustainability to mechanical and oxidative stress. At any time these markers can give insight about the RBC’s integrity since high levels of MFI or MFC are related with RBC aging and subsequently hemolysis [58].

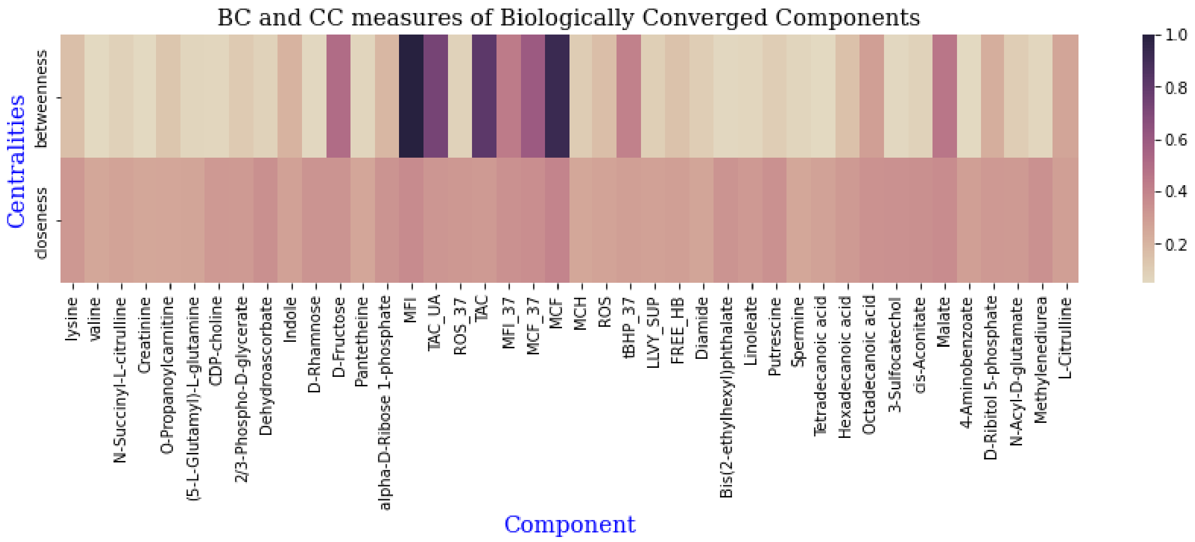
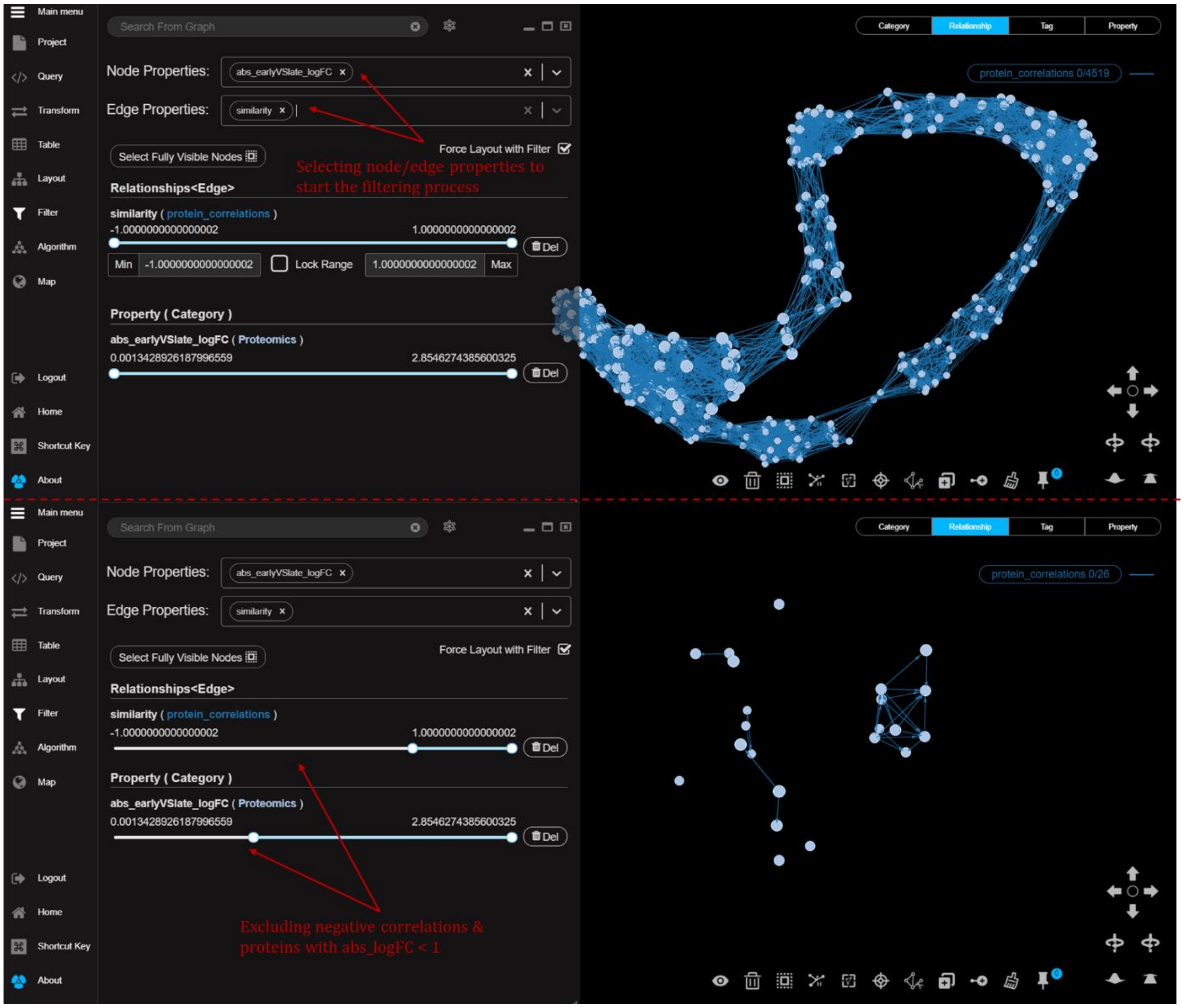


Figure 41. Identifying crucial components related to G6PD.

C. Presenting graph according to specific node or relationship properties

Two aspects will be discussed in this section. The first one is related to displaying only those nodes or edges that pass a filter set by the user and the second concerns the representation of nodes in terms of coloring based on properties values. To explain properly the first case we chose to work on a very dense network, that of protein correlations, and extract some valuable information from it.

In Figure 42 the filtering of nodes using the Filter panel of GraphXR is presented. At first through the Filter panel of the Main menu we can select the node or edge properties to which we will apply a filter (top). Following that, we proceed with setting the desired threshold to each of the selected properties. The first filter concerns the edge property regarding the correlation between the connected proteins. Its value is bound in [0,1] after the filtering process, so that correlations with negative measures will be excluded. The second filter is about the statistical significance of displayed proteins. To measure that the absolute value of log2 of Fold Change (logFC) was used as a metric. Fold Change is the ratio between two different states [59]. In our case the first state (numerator) is the concentration of a protein in the in vivo system, while the second case (denominator) is the concentration of the same protein during the last sampling (in vitro system). The logFC of each protein was computed during the setup of the hematological markers network. LogFC is significant if greater than 1 (numerator = 2denominator) or smaller than -1(denominator = 2nominator). In this case we used the absolute value of logFC measure and for this reason it is bound in [1,infinite) (bottom). Once the filtering process is over the user can navigate through the Table panel and see in more detail each one of the proteins that satisfy the criteria.



Node Properties: abs_earlyVSlate_logFC

Edge Properties: similarity

Force Layout with Filter ☑

Relationships<Edge>

similarity (protein_correlations)

-1.0000000000000002 1.0000000000000002

Min -1.0000000000000002 □ Lock Range 1.0000000000000002 Max

Property (Category)

abs_earlyVSlate_logFC (Proteomics)

0.0013428926187996559 2.8546274385600325

Selecting node/edge properties to start the filtering process

Node Properties: abs_earlyVSlate_logFC

Edge Properties: similarity

Force Layout with Filter ☑

Relationships<Edge>

similarity (protein_correlations)

-1.0000000000000002 1.0000000000000002

Property (Category)

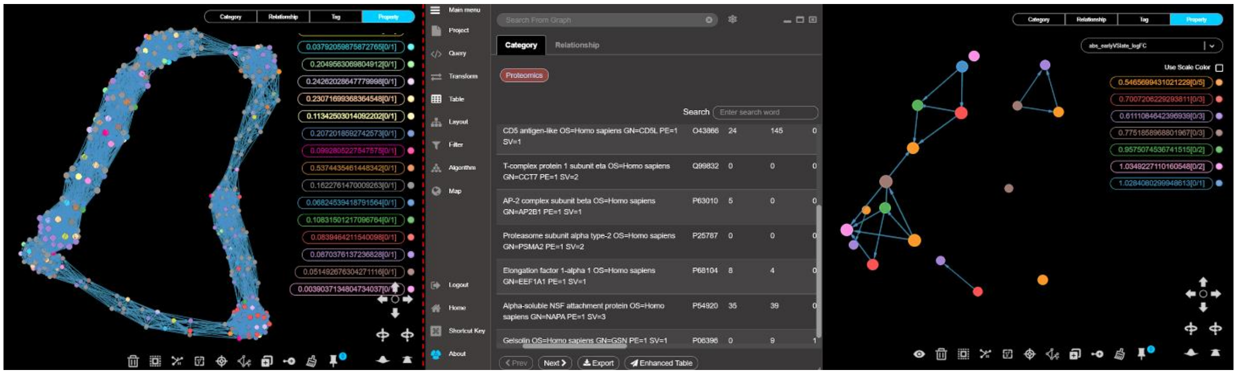
abs_earlyVSlate_logFC (Proteomics)

0.0013428926187996559 2.8546274385600325

Excluding negative correlations & proteins with abs_logFC < 1

Figure 42. Filtering statistically significant proteins and their positive correlations.

In Figure 43 an example of the second aspect of this section is shown. Once again, the network of protein correlations is used as a template, but in this case, we will focus on selecting nodes according to a specific value using the Property tab of the Legend menu. Moreover, we will refer to the coloring system that is provided to color nodes by their property values. One the left part of the figure one can observe the protein correlation network in which all nodes are dashed according to the selected property value (logFC of in vivo versus late in vitro in this case). Nodes of the same color have the same property value. On the right part of the figure, we can notice those nodes that were filtered manually, along with their relationships. Moreover, by using the Table panel we can see more information regarding those nodes.

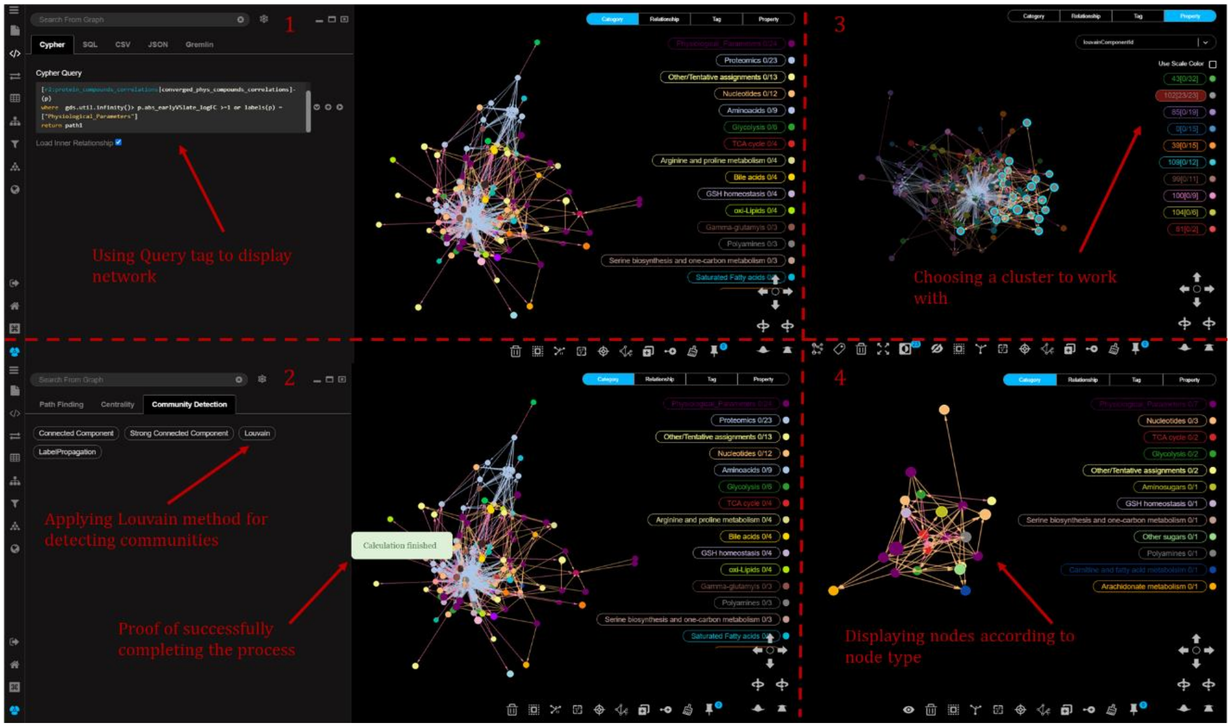


CD5 antigen-like OS=Homo sapiens GN=CD5L PE=1 SV=1	O43866	24	145	0
T-complex protein 1 subunit eta OS=Homo sapiens GN=CCT7 PE=1 SV=2	Q99832	0	0	0
AP-2 complex subunit beta OS=Homo sapiens GN=AP2B1 PE=1 SV=1	P63010	5	0	0
Proteasome subunit alpha type-2 OS=Homo sapiens GN=PSMA2 PE=1 SV=2	P25787	0	0	0
Elongation factor 1-alpha 1 OS=Homo sapiens GN=EEF1A1 PE=1 SV=1	P68104	8	4	0
Alpha-soluble NSF attachment protein OS=Homo sapiens GN=NAPA PE=1 SV=3	P54920	35	39	0
Gelsolin OS=Homo sapiens GN=GSN PE=1 SV=1	P06396	0	9	1

Figure 43. Filtering nodes manually via Legend menu.

D. Working with communities

The process that is followed to identify and subsequently work with specific communities of a graph is like the case study that was described in section 5.2. Here, a similar case will be investigated, but this time, we will work with detecting communities of biological converged components and after we will focus on some of them. A six-step exemplary case will be described below in detail.

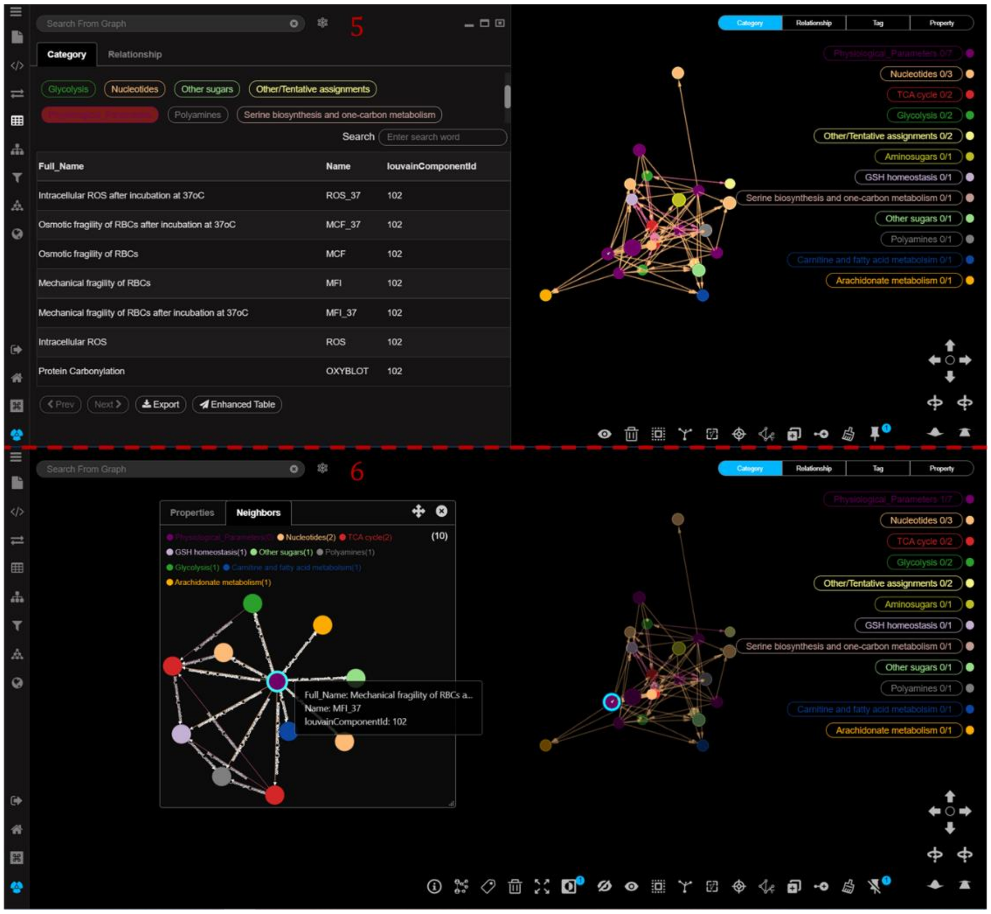

1
Search From Graph
Cypher
SQL
CSV
JSON
Gremlin
Cypher Query
Load Inner Relationship ☑
Using Query tag to display network
Category
Relationship
Tag
Property
Physiological_Parameters 0/24
Proteomics 0/23
Other/Tentative assignments 0/13
Nucleotides 0/12
Aminoacids 0/9
Glycolysis 0/8
TCA cycle 0/4
Arginine and proline metabolism 0/4
Bile acids 0/4
GSH homeostasis 0/4
oxi-Lipids 0/4
Gamma-glutamyls 0/3
Polyamines 0/3
Serine biosynthesis and one-carbon metabolism 0/3
Saturated Fatty acids 0/2
3
louvainComponentId
Use Scale Color □
43[0/32]
102[23/23]
85[0/19]
0[0/15]
39[0/15]
109[0/12]
99[0/11]
100[0/9]
104[0/6]
81[0/2]
Choosing a cluster to work with
2
Path Finding
Centrality
Community Detection
Connected Component
Strong Connected Component
Louvain
LabelPropagation
Applying Louvain method for detecting communities
Calculation finished
Proof of successfully completing the process
4
Physiological_Parameters 0/7
Nucleotides 0/3
TCA cycle 0/2
Glycolysis 0/2
Other/Tentative assignments 0/2
Aminosugars 0/1
GSH homeostasis 0/1
Serine biosynthesis and one-carbon metabolism 0/1
Other sugars 0/1
Polyamines 0/1
Carnitine and fatty acid metabolism 0/1
Arachidonate metabolism 0/1
Displaying nodes according to node type

```
[r1:protein_compounds_correlations|converged_phys_compounds_correlations]-(p)
where  gds.util.infinity()> p.abs_earlyVSlate_logFC >=1 or labels(p) = ["Physiological_Parameters"]
return path1
```

Figure 44. Working with communities – Part I.

The first part of this example starts by using the Query panel and by choosing the favorite query that returns all biologically converged relationships the relative network appears on the graph space . Next, through the Algorithm panel of the Main menu we can browse to the Community Detection tab and select the Louvain method for detecting communities. Following that, by searching the Property tab of the Legend menu we can find one or more communities based on the property value created by the Louvain algorithm. The first part of this case concludes by displaying nodes of the selected community based on the category they belong (Figure 44).

The second part of this process is related to exploration analysis that can take place once we have selected a cluster to work with. One of the options that are available in GraphXR includes the Table panel of the Main menu from which the user can display all properties of any node type that is part of the selected community. Another aspect could be to choose manually any node of the presented graph we are interested in and display its neighbors. Figure 45 presents in detail both aspects. On the top left of the figure the option of working with the Table panel is presented. Here the property table of the category concerning physiological parameters has been selected (5). The lower part of the figure focuses on displaying neighbors of manually selected nodes from the cluster. More specifically, in this case the neighbors of the physiological parameter "Mechanical Fragility of RBCs after incubation at 37o C" are shown. Each neighbor is dashed according to the node type it belongs to (6). Following such exploratory approaches one can gain insight about potential effects between connected components. Of course, further investigation is required to drive to any accurate result.



Full_Name	Name	louvainComponentId
Intracellular ROS after incubation at 37oC	ROS_37	102
Osmotic fragility of RBCs after incubation at 37oC	MCF_37	102
Osmotic fragility of RBCs	MCF	102
Mechanical fragility of RBCs	MFI	102
Mechanical fragility of RBCs after incubation at 37oC	MFI_37	102
Intracellular ROS	ROS	102
Protein Carbonylation	OXYBLOT	102

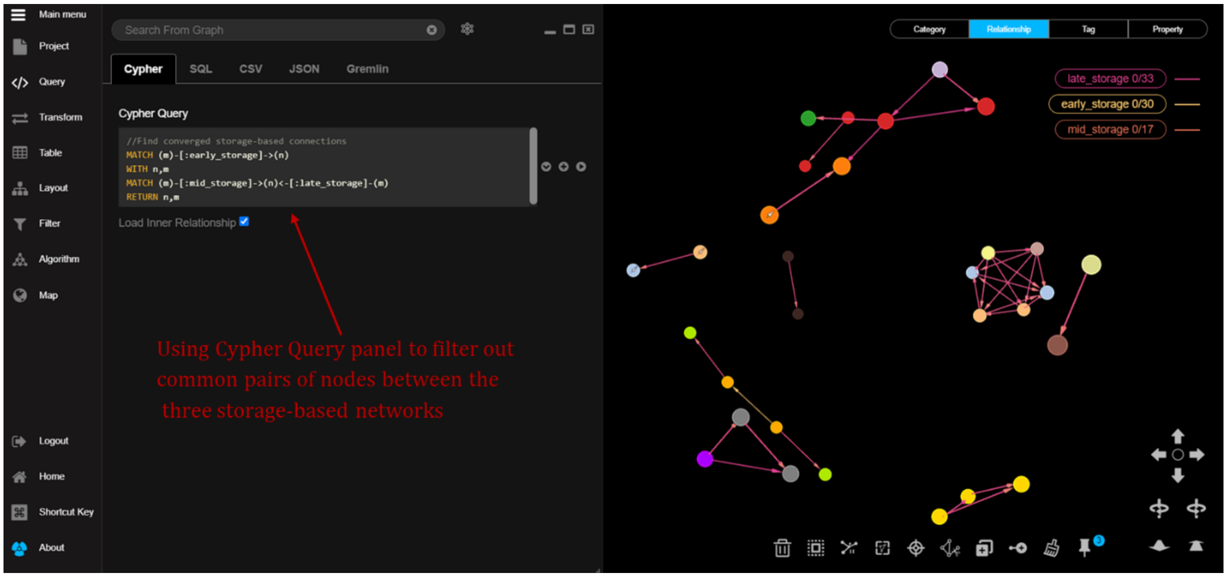
Full_Name: Mechanical fragility of RBCs a...
Name: MFI_37
louvainComponentId: 102

5
6
Search From Graph
Category
Relationship
Glycolysis
Nucleotides
Other sugars
Other/Tentative assignments
Polyamines
Serine biosynthesis and one-carbon metabolism
Search
Enter search word
Prev
Next
Export
Enhanced Table
Tag
Property
Physiological_Parameters 0/7
Nucleotides 0/3
TCA cycle 0/2
Glycolysis 0/2
Other/Tentative assignments 0/2
Aminosugars 0/1
GSH homeostasis 0/1
Serine biosynthesis and one-carbon metabolism 0/1
Other sugars 0/1
Polyamines 0/1
Carnitine and fatty acid metabolism 0/1
Arachidonate metabolism 0/1
Properties
Neighbors
Physiological_Parameters(0)
Nucleotides(2)
TCA cycle(2)
(10)
GSH homeostasis(1)
Other sugars(1)
Polyamines(1)
Glycolysis(1)
Carnitine and fatty acid metabolism(1)
Arachidonate metabolism(1)
Physiological_Parameters 1/7

Figure 45. Working with communities – Part II.

E. Storage-based analysis of the metabolic profile

The last case study that will be presented concerns the comparative analysis of metabolic networks through different storage-based time periods. Therefore, for this example three metabolic correlation networks will be used namely “early storage”, “mid storage” and “late storage”. More information about each one of them is available in chapter 4. The process of this exploration analysis starts with detecting pair of nodes that maintain a strong connection throughout all stages (Figure 46) and concludes with computing the percentage of identity between those three networks (Table 8). For the execution of both procedures the Query panel of the Main menu was used.



```
//Find converged storage-based connections
MATCH (m)-[:early_storage]->(n)
WITH n,m
MATCH (m)-[:mid_storage]->(n)<-[:late_storage]-(m)
RETURN n,m
```

Load Inner Relationship ☑

Using Cypher Query panel to filter out common pairs of nodes between the three storage-based networks

Figure 46. Identifying common pairs of nodes between all storage-based metabolic correlation networks.

Table 8. Percentage of identity between storage-based metabolic correlation networks. Each percentage was computed via Cypher queries.

Compared Networks	Percentage (%) of Identity
early storage VS mid storage	13.924
early storage VS late storage	15.678
mid storage VS late storage	27.848

The Cypher query that was used for the computation of each percentage is available below:

```
MATCH (n)-[r1:early_storage]-(m),
(n)-[r:mid_storage]-(m)
WITH COLLECT(DISTINCT [n.Name,m.Name]) AS pair, type(r1) AS type,COUNT(DISTINCT r) AS
`common pairs`
MATCH (m1)-[r2:mid_storage]-(m2)
RETURN type+" VS "+type(r2) AS `compared timelines`,(toFLoat(`common pairs`)/COUNT(DISTINCT
r2))*100 AS `% network identity`
UNION
MATCH (n)-[r1:early_storage]-(m),
(n)-[r:late_storage]-(m)
```

```
WITH COLLECT(DISTINCT [n.Name,m.Name]) AS pair, type(r1) AS type,COUNT(DISTINCT r) AS
`common pairs`
MATCH (m1)-[r2:late_storage]-(m2)
RETURN type+" VS "+type(r2) AS `compared timelines`,(toFLoat(`common pairs`)/COUNT(DISTINCT
r2))*100 AS `% network identity`
UNION
MATCH (n)-[r1:late_storage]-(m),
(n)-[r:mid_storage]-(m)
WITH COLLECT(DISTINCT [n.Name,m.Name]) AS pair, type(r1) AS type,COUNT(DISTINCT r) AS
`common pairs`
MATCH (m1)-[r2:mid_storage]-(m2)
RETURN type+" VS "+type(r2) AS `compared timelines`,(toFLoat(`common pairs`)/COUNT(DISTINCT
r2))*100 AS `% network identity`
```

Each of the examples presented above is supposed to give some insight about the potential anyone has in terms of exploring graph data using GraphXR. Depending on the question one needs to answer a different combination of the above methods might be in handy.

6. CONCLUSIONS AND FURTHER WORK

Understanding the complexity of biochemical and physiological events that occur during the storage of erythrocytes could give insight about the most crucial parameters that are affected by or related to storage lesion, especially for samples retrieved from donors with prior blood-related health issues, such as G6PD deficiency. Designing a conclusive hematological markers network using both experimental and computationally verified data and subsequently utilizing graph analytics is a very efficient way to look into and highlight intra- and inter-parameter associations between different biochemical and hematological components, and potentially reveal new correlations that might not been extensively investigated before. However, to develop a method that best describes and illustrates the case-study biological problem is necessary to collect several biological scenarios to which the graph needs to be able to answer. Moreover, the appropriate tools and functions need to be exploited, so that the final graph model consists of well-structured relationships that highlight the significant parameters of the network, that are related with the biological issue, and their closely associates.

Utilizing graph database systems, such as Neo4j, to develop and evaluate the graph model, that has been designed, provides the asset of handling large amounts of relational data efficiently, accurately and with high speed. Neo4j is quite ideal for this purpose since it provides a wide variety of predefined tools and algorithms that can be of great use during the construction of the hematological markers network. Moreover, it can be easily accessed by many programming languages through its REST API, while it gives a more dynamic approach to the process of graph analytics and visualization by been able to directly link a copy of any working project to other browser-based graph-related visualization tools, such as GraphXR, that can be used easily from users without an IT background. However, prior knowledge of the basic aspects of graph theory and programming with Cypher query language are prerequisites to use Neo4j.

Amongst the most important findings of this study were the construction of a conclusive graph that depicts the associations of hematological parameters throughout the duration of RBCs in storage and highlights the most of popular graph entities. More specifically, the current graph model can give insight to several biological questions related to the storage effect on erythrocytes from G6PD deficient donors. Some of the most important biological scenarios that have been answered with the specific graph model are related to finding differences between *in vivo* and *in vitro* systems, identifying biologically converged metabolic and physiological parameters of the network, detecting highly connected communities of converged components, understanding the effect of storage on RBCs through comparative analyzes of storage- and time- based metabolic correlation networks (e.g. early storage vs late storage) or collecting phenotypic information regarding G6PD and G6PD-related proteins. Of course, further investigation through experimental procedures is required to evaluate the integrity of these results.

Further Work

Up to now the development of hematological markers network is fully established and graph analytics of the current graph model have been concluded as well. However, as more data become available new challenges may rise that could expand the current version the graph. Another task that could be further investigated, could be the automation of the construction of the network. So far, all data sources are loaded manually to Neo4j via Cypher queries and they were manipulated from there. However, as it was already mentioned Neo4j provides a REST API interface that can be accessible from most programming languages. That said, an interesting approach could be the development of an automated method (e.g. in Python) that would start from retrieving all data that may be loaded to the graph, then deal with any kind of preprocessing and data cleaning and, finally, conclude with the construction of the final graph model.

BIBLIOGRAPHY

[1] S. Peter Klinken, “Red blood cells,” *The International Journal of Biochemistry & Cell Biology*, vol. 34, no. 12, Dec. 2002, doi: 10.1016/S1357-2725(02)00087-0.

[2] K. Yazdanbakhsh, C. Lomas-Francis, and M. E. Reid, “Blood groups and diseases associated with inherited abnormalities of the red blood cell membrane,” *Transfusion Medicine Reviews*, vol. 14, no. 4, Oct. 2000, doi: 10.1053/tmrv.2000.16232.

[3] J. A. Reisz *et al.*, “Metabolic Linkage and Correlations to Storage Capacity in Erythrocytes from Glucose 6-Phosphate Dehydrogenase-Deficient Donors,” *Frontiers in Medicine*, vol. 4, Jan. 2018, doi: 10.3389/fmed.2017.00248.

[4] M. CHEVION, T. NAVOK, G. GLASER, and J. MAGER, “The Chemistry of Favism-Inducing Compounds. The Properties of Isouramil and Divicine and Their Reaction with Glutathione,” *European Journal of Biochemistry*, vol. 127, no. 2, Oct. 1982, doi: 10.1111/j.1432-1033.1982.tb06886.x.

[5] M. García-Roa *et al.*, “Red blood cell storage time and transfusion: current practice, concerns and future perspectives.,” *Blood transfusion = Trasfusione del sangue*, vol. 15, no. 3, May 2017, doi: 10.2450/2017.0345-16.

[6] S. Chien, “Red Cell Deformability and its Relevance to Blood Flow,” *Annual Review of Physiology*, vol. 49, no. 1, Mar. 1987, doi: 10.1146/annurev.ph.49.030187.001141.

[7] M. L. Turgeon, *Clinical Hematology: Theory and Procedures*. Jones & Bartlett Learning, 2004.

[8] S. Kabanova, P. Kleinbongard, J. Volkmer, B. Andrée, M. Kelm, and T. W. Jax, “Gene expression analysis of human red blood cells,” *International Journal of Medical Sciences*, 2009, doi: 10.7150/ijms.6.156.

[9] R. Linberg, C. D. Conover, and K. L. Shum, “Hemoglobin Based Oxygen Carriers: How Much Methemoglobin is too Much?,” *Artificial Cells, Blood Substitutes, and Biotechnology*, vol. 26, no. 2, Jan. 1998, doi: 10.3109/10731199809119772.

[10] R. C. Hardison, “Evolution of Hemoglobin and Its Genes,” *Cold Spring Harbor Perspectives in Medicine*, vol. 2, no. 12, Dec. 2012, doi: 10.1101/cshperspect.a011627.

[11] N. Jiang, N. S. Tan, B. Ho, and J. L. Ding, “Respiratory protein–generated reactive oxygen species as an antimicrobial strategy,” *Nature Immunology*, vol. 8, no. 10, Oct. 2007, doi: 10.1038/ni1501.

[12] Y. Ogasawara, M. Funakoshi, and K. Ishii, “Glucose metabolism is accelerated by exposure to t-butylhydroperoxide during NADH consumption in human erythrocytes,” *Blood Cells, Molecules, and Diseases*, vol. 41, no. 3, Nov. 2008, doi: 10.1016/j.bcmd.2008.05.007.

[13] M. CHEVION, T. NAVOK, G. GLASER, and J. MAGER, “The Chemistry of Favism-Inducing Compounds. The Properties of Isouramil and Divicine and Their Reaction with Glutathione,” *European Journal of Biochemistry*, vol. 127, no. 2, Oct. 1982, doi: 10.1111/j.1432-1033.1982.tb06886.x.

[14] S. Chien, “Red Cell Deformability and its Relevance to Blood Flow,” *Annual Review of Physiology*, vol. 49, no. 1, Mar. 1987, doi: 10.1146/annurev.ph.49.030187.001141.

[15] S. M. Frank *et al.*, “Decreased Erythrocyte Deformability After Transfusion and the Effects of Erythrocyte Storage Duration,” *Anesthesia & Analgesia*, vol. 116, no. 5, May 2013, doi: 10.1213/ANE.0b013e31828843e6.

[16] E. Keohane, C. Otto, and J. Walenga, *Hematology: clinical principles and applications*, 6th ed. 2007.

[17] L. A. Ziegler, S. E. Olia, and M. v. Kameneva, “Red Blood Cell Mechanical Fragility Test for Clinical Research Applications,” *Artificial Organs*, vol. 41, no. 7, Jul. 2017, doi: 10.1111/aor.12826.

[18] D. J. Schaer, P. W. Buehler, A. I. Alayash, J. D. Belcher, and G. M. Vercellotti, “Hemolysis and free hemoglobin revisited: exploring hemoglobin and hemin scavengers as a novel class of therapeutic proteins,” *Blood*, vol. 121, no. 8, Feb. 2013, doi: 10.1182/blood-2012-11-451229.

[19] M. Valko, C. J. Rhodes, J. Moncol, M. Izakovic, and M. Mazur, “Free radicals, metals and antioxidants in oxidative stress-induced cancer,” *Chemico-Biological Interactions*, vol. 160, no. 1, Mar. 2006, doi: 10.1016/j.cbi.2005.12.009.

[20] S. R. Richardson and G. F. O’Malley, *Glucose 6 Phosphate Dehydrogenase Deficiency*. 2021.

[21] S. Schrier and S. Landaw, “Mean corpuscular volume.” https://somepomed.org/articulos/contents/mobipreview.htm?28/19/28991/contributors (accessed Oct. 08, 2021).

[22] V. K. Balakrishnan, *Graph Theory*, 1st ed. 1997.

[23] L. Ehrlinger and W. Wöß, “Towards a Definition of Knowledge Graphs,” 2016. [Online]. Available: http://www.semantic-web-journal.net/content/

[24] I. Robinson, J. Webber, and E. Eifrem, “Graph Databases.”

[25] S. Bhavanari and P. Kuncham Syam, *Discrete Mathematics and Graph Theory*. Prentice Hall of India, 2009.

[26] M. Saberi, R. Khosrowabadi, A. Khatibi, B. Misic, and G. Jafari, “Topological impact of negative links on the stability of resting-state brain network,” *Scientific Reports*, vol. 11, no. 1, Dec. 2021, doi: 10.1038/s41598-021-81767-7.

[27] K. Aasavari, “What Is Graph Analytics?” https://medium.com/swlh/what-is-graph-analytics-9223d71c26d8 (accessed Oct. 08, 2021).

[28] N. Bourbakis, *Artificial Intelligence and Automation*, vol. 3. World Scientific, 1998.

[29] B.-H. Yoon, S.-K. Kim, and S.-Y. Kim, “Use of Graph Database for the Integration of Heterogeneous Biological Data,” *Genomics & Informatics*, vol. 15, no. 1, 2017, doi: 10.5808/GI.2017.15.1.19.

[30] R. Angles and C. Gutierrez, “Survey of graph database models,” *ACM Computing Surveys*, vol. 40, no. 1, Feb. 2008, doi: 10.1145/1322432.1322433.

[31] “Cayley.” https://github.com/cayleygraph/cayley (accessed Oct. 08, 2021).

[32] N. Kallen, R. Pointer, J. Kalucki, and E. Ceaser, “FlockDB,” 2010. https://github.com/twitter-archive/flockdb (accessed Oct. 08, 2021).

[33] “The Neo4j Getting Started Guide v4.3.”

[34] “Overview of OrientDB.” https://orientdb.org/docs/3.0.x/misc/Overview.html (accessed Oct. 08, 2021).

[35] N. Francis *et al.*, “Cypher,” May 2018. doi: 10.1145/3183713.3190657.

[36] R. Kowsar, K. Sadeghi, S. Farshad Kateb Yektadam Persian Co Elham Bonakdar, A. Hossein Mahdavi, A. Mohammad Rahimi, and M. Sroka, “Hematological and biochemical characteristics of COVID-19 non-survivors: a meta-and network analysis,” 2020, doi: 10.21203/rs.3.rs-130151/v1.

[37] S. R. Goodman, O. Daescu, D. G. Kakhniashvili, and M. Zivanic, “The proteomics and interactomics of human erythrocytes,” *Experimental Biology and Medicine*, vol. 238, no. 5. pp. 509–518, Aug. 2013. doi: 10.1177/1535370213488474.

[38] H. de León *et al.*, “A vascular biology network model focused on inflammatory processes to investigate atherogenesis and plaque instability,” 2014. [Online]. Available: http://www.translational-medicine.com/content/12/1/185

[39] A. I. Amanatidou, K. C. Nastou, O. E. Tsitsilonis, and V. A. Iconomidou, “Visualization and analysis of the interaction network of proteins associated with blood-cell targeting autoimmune diseases,” *Biochimica et Biophysica Acta - Molecular Basis of Disease*, vol. 1866, no. 5, May 2020, doi: 10.1016/j.bbadis.2020.165714.

[40] E. Szczesny-Malysiak, T. Mohaissen, K. Bulat, M. Kaczmarska, A. Wajda, and K. M. Marzec, “Sex-dependent membranopathy in stored human red blood cells.”

[41] L. J. Jensen *et al.*, “STRING 8--a global view on proteins and their functional interactions in 630 organisms,” *Nucleic Acids Research*, vol. 37, no. Database, Jan. 2009, doi: 10.1093/nar/gkn760.

[42] M. Kuhn, C. von Mering, M. Campillos, L. J. Jensen, and P. Bork, “STITCH: interaction networks of chemicals and proteins,” *Nucleic Acids Research*, vol. 36, no. Database, Dec. 2007, doi: 10.1093/nar/gkm795.

[43] F. Pontén, K. Jirström, and M. Uhlen, “The Human Protein Atlas—a tool for pathology,” *The Journal of Pathology*, vol. 216, no. 4, Dec. 2008, doi: 10.1002/path.2440.

[44] T. Hubbard, “The Ensembl genome database project,” *Nucleic Acids Research*, vol. 30, no. 1, Jan. 2002, doi: 10.1093/nar/30.1.38.

[45] G. A. Pavlopoulos *et al.*, “Using graph theory to analyze biological networks,” *BioData Mining*, vol. 4, no. 1, Dec. 2011, doi: 10.1186/1756-0381-4-10.

[46] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, “Fast unfolding of communities in large networks,” *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2008, no. 10, Oct. 2008, doi: 10.1088/1742-5468/2008/10/P10008.

[47] Kent State University Libraries., “SPSS Tutorials: Pearson Correlation,” May 15, 2017. http://libguides.library.kent.edu/SPSS/PearsonCorr (accessed Oct. 08, 2021).

[48] N. Steven, “What Do Correlation Coefficients Positive, Negative, and Zero Mean?,” 2021. https://www.investopedia.com/ask/answers/032515/what-does-it-mean-if-correlation-coefficient-positive-negative-or-zero.asp#positive-correlation (accessed Oct. 08, 2021).

[49] A. Singhal, “Modern Information Retrieval: A Brief Overview,” 2001. [Online]. Available: http://trec.nist.gov

[50] “GraphXR User Guide.”

[51] “Neo4j_Graph_Algorithms_r3”.

[52] F. Radicchi, C. Castellano, F. Cecconi, V. Loreto, and D. Parisi, “Defining and identifying communities in networks,” 2004. [Online]. Available: www.pnas.orgcgidoi10.1073pnas.0400054101

[53] M. Girvan and M. E. J. Newman, “Community structure in social and biological networks.” [Online]. Available: www.pnas.orgcgidoi10.1073pnas.122653799

[54] M. E. J. Newman, “Modularity and community structure in networks,” 2006. [Online]. Available: www.pnas.orgcgidoi10.1073pnas.0601602103

[55] A. Clauset, M. E. J. Newman, and C. Moore, “Finding community structure in very large networks,” *Physical Review E*, vol. 70, no. 6, Dec. 2004, doi: 10.1103/PhysRevE.70.066111.

[56] M. Needham and A. E. Hodler, “Graph Algorithms in Neo4j: Louvain Modularity,” 2019. https://neo4j.com/blog/graph-algorithms-neo4j-louvain-modularity/ (accessed Oct. 08, 2021).

[57] M. Needham and A. E. Hodler, “Graph Algorithms in Neo4j: Label Propagation,” 2019. https://neo4j.com/blog/graph-algorithms-neo4j-label-propagation/ (accessed Oct. 08, 2021).

[58] A. Orbach, O. Zelig, S. Yedgar, and G. Barshtein, “Biophysical and Biochemical Markers of Red Blood Cell Fragility,” *Transfusion Medicine and Hemotherapy*, vol. 44, no. 3, 2017, doi: 10.1159/000452106.

[59] M. D. Robinson and G. K. Smyth, “Small-sample estimation of negative binomial dispersion, with applications to SAGE data,” *Biostatistics*, vol. 9, no. 2, Jul. 2007, doi: 10.1093/biostatistics/kxm030.