UNIVERSITÄT
HEIDELBERG
ZUKUNFT
SEIT 1386

Universität Heidelberg

Fakultät für Mathematik und Informatik

Winter Semester 2019 / 2020

Course: **Mathematical Structure of Complex Systems**

Supervisor: **Professor Markus Kirkilionis <m.a.kirkilionis@warwick.ac.uk>**

FoodWeb: Modelling Nutrition and Diet with Hierarchical Hypergraph

Student: Claire Zhao Sun

Matrikelnummer: 3630998

Studiengang: Scientific Computing (1. Fachsemester)

Email: pz237@stud.uni-heidelberg.de

Report Submitted on 24 June 2020

Contents

1. Introduction

Food is one of the most fundamental necessities of our life. It provides not only energy that our bodies need for survival, but also the essential nourishments for good health. Patterns in diet and nutrition at a population level are intrinsically complex, since food consumption interacts with so many other aspects of life. In addition, the increasing demand for food by humanity pressures our environment and ecosystem. We live in this interwoven web and our relationship to it is defined by food.

The primary objective of this project is to first apply the concepts of hierarchical hypergraphs to model the food & nutrition landscape, and to then model the interactions of people with the food system. This paper aims to achieve the following three goals:

1) Interpret the hierarchical hypergraph concepts in a real-life application
2) Implement a hierarchical hypergraph model to represent the food system
3) Demonstrate the FoodWeb model via problem solving in different scenarios

This paper is organized as follows: Section 2 reviews existing research on hypergraphs and their applications in various disciplines. Research trends applying a complex system approach to diet is also examined. Section 3 introduces the basic concepts from graph to hierarchical hypergraph, laying the theoretical ground for their applications in modelling the food system. Section 4 discusses data collection and system set-up before presenting FoodWeb Hierarchical Hypergraph Model, implemented in a Neo4j graph database. Section 5 showcases a few examples of making queries in FoodWeb model under different scenarios and discusses potential model extensions for further applications. Section 6 concludes by identifying future research areas.

2. Literature Review

In the lecture series, “Mathematical Structures of Complex Systems” [1], we have seen many applications of graph representation: chemical molecules, cell biology, reaction networks [2], financial markets, etc. Moreover, hypergraph applications can be found in various disciplines such as complex networks [3], robotic systems [4], software development [5], hardware systems [6], social networks [7,8], etc. However, research specifically focused on hierarchical hypergraphs and their applications is rather limited in the literature and there is currently no research relating hypergraphs to food and nutrition.

On the other hand, there has been an emerging trend of using the complex system approach to study diet. In a systematic review by Langellier, et al. [9], agent-based and system dynamics models have been identified to explore the mechanisms influencing diet at various levels, e.g., at the neighbourhood-level, at the interpersonal level and at the individual level. However, there is still much work to be done with respect to "better understand mechanisms driving population-level diet, increasing use of models for policy decision support, and leveraging the wide availability of epidemiologic and policy evaluation data to improve model validation."

3. Graph Representation of the FoodWeb System

This section serves as a comprehensive introduction to basic definitions of graph theory, from simple graphs to more complex multipartite graphs and hierarchical hypergraphs. It aims to interpret how these concepts can be applied to model the intricate web of information embedded in the food, nutrition and diet system. We also introduce our model, "FoodWeb Hierarchical Hypergraph", in this section.

3.1 Basic Definitions

Definition **[Graph]**

A graph is a set theoretic structure, $G = G(V,E)$, where V is the basic set consisting of all the vertices (or nodes) and E is the set of edges (or links) connecting pairs of vertices. In this context, E is a subset of the Cartesian product of the basic set V with itself, i.e., $E \subseteq V \times V$.

If the two vertices connected by each edge are an unordered pair, then we call the graph an undirected simple graph. If, on the other hand, the two vertices are an ordered pair, i.e. the edge has a specific orientation, then we call the graph a directed graph. In the latter case, the directed edge is sometimes indicated by an arrow pointing from the source node to the target node. We denote the edge by an ordered tuple (s, t) ∈ E and the two vertices s, t ∈ V.

Definition **[Relation]**

Graphs can be used to describe how entities are related to one another. Let R be a binary relation on two finite sets S and T, where S = $\{s_1, s_2, \ldots, s_n\}$, T = $\{t_1, t_2, \ldots, t_m\}$. R is then defined as a set of ordered pairs (s_i, t_j) where s_i ⊆ S and t_j ⊆ T. R is a subset of the Cartesian product of S and T, i.e., R ⊆ S × T.

We may describe the binary relation R by drawing a directed graph as follows: first represent all elements in S and T as nodes, then for every ordered pair (s_i, t_j) ∈ R, draw an arrow connecting node s_i to node t_j. Furthermore, we can encode more specific information about a particular binary relationship by annotating a graph with labeled nodes and weighted edges.

Definition **[Weighted Graph]**

Weighted graphs are graphs where each edge has a numerical value called weight. They can be used to represent structures in which pairwise connections have numerical values. A weighted graph is a special type of labeled graph, in which nodes and edges are annotated.

Definition **[Multi-Partite Graph]**

A set theoretic structure G(V^m, E) = G($V_1, V_2, \ldots, V_m$, E) is called a multi-partite graph of order m, with $\{V_1, V_2, \ldots, V_m\}$ the vertex set and E the set of edges, such that only connections between distinct vertex sets are allowed but not within any vertex set.

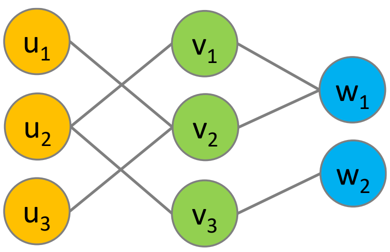

Figure 1. A multi-partite graph of order 3.

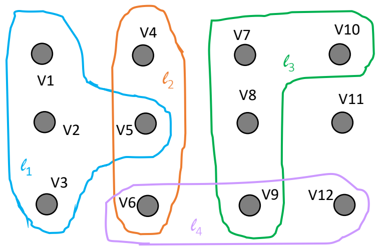

Figure 2. A finite hypergraph with 4 hyperlinks.

Definition **[Hypergraph]**

A hypergraph is set theoretic structure, H = H(V,L), which consists of two sets: the set of vertices, V, which is the basic set, and the set of hyperlinks (or hyperedges), L, which is a family of subsets of V, i.e., for each $l \in$ L also satisfies $l \subset$ V.

Definition **[Hierarchical Hypergraph]**

A **hierarchical hypergraph** of **order** m is a set theoretic structure $\mathrm{H}(\mathrm{V}, \mathrm{L}^{\mathrm{m}}) = \mathrm{H}(\mathrm{V}, \mathrm{L}_1, \mathrm{L}_2, \dots, \mathrm{L}_{\mathrm{m}})$, which consists of a basic set V (indexed by a set $\mathbb{I}_0$) and sets $\mathrm{L}_1, \mathrm{L}_2, \dots, \mathrm{L}_{\mathrm{m}}$ (indexed by sets $\mathbb{I}_1, \mathbb{I}_2, \dots, \mathbb{I}_{\mathrm{m}}$), m $\in \mathbb{N}$, if all of the following conditions hold:

(i) L_1 is a family of subsets of V, i.e., $\mathrm{L}_1 := \{\, l_1^{\mathrm{i}} \mid \mathrm{i} \in \mathbb{I}_1 \,\}$, such that each set $l_1^{\mathrm{i}} \subset$ V;

(ii) L_{k} , k = 2, …, m, is a family of sets $\mathrm{L}_{\mathrm{k}} := \{\, l_{\mathrm{k}}^{\mathrm{i}} \mid \mathrm{i} \in \mathbb{I}_{\mathrm{k}} \,\}$, such that any element $\lambda \in l$, with $l \in \mathrm{L}_{\mathrm{k}}$, satisfies either $\lambda \in$ V or $\lambda \in \mathrm{L}_{\varkappa}$, $1 \leq \varkappa < \mathrm{k}$.

If all index sets $\mathbb{I}_1, \mathbb{I}_2, \dots, \mathbb{I}_{\mathrm{m}}$ are countable, then the hierarchical hypergraph is **discrete**. If $\mathbb{I}_0$ is also countable, then the hierarchical hypergraph is **fully discrete**. If all index sets $\mathbb{I}_0, \mathbb{I}_1, \mathbb{I}_2, \dots, \mathbb{I}_{\mathrm{m}}$ are finite, then the hierarchical hypergraph is **finite**.

A hierarchical hypergraph, H = $\mathrm{H}(\mathrm{V}, \mathrm{L}^{\mathrm{m}})$, is said to use the **level preorder** $\leq_{\mathbb{L}}$, a binary relation, if and only if it satisfies the following preorder conditions for the set of level sets $\mathbb{L} := \{\mathrm{V}, \mathrm{L}_1, \mathrm{L}_2, \dots, \mathrm{L}_{\mathrm{m}}\}$ that for all L, L', L'' $\in \mathbb{L}$:

(i) L $\leq_{\mathbb{L}}$ L (**Reflexivity**);

(ii) if L $\leq_{\mathbb{L}}$ L' and L' $\leq_{\mathbb{L}}$ L'', then L $\leq_{\mathbb{L}}$ L'' (**Transitivity**).

The preorder $\leq_{\mathbb{L}}$ is a **partial level set order**, if it also satisfies that:

(iii) if L $\leq_{\mathbb{L}}$ L' and L' $\leq_{\mathbb{L}}$ L, then L = L'' (**Antisymmetry**).

In addition, ($\mathbb{L}$, $\leq_{\mathbb{L}}$) is called a **linear level set order**, if for all L, L' $\in \mathbb{L}$:

(iv) either L $\leq_{\mathbb{L}}$ L' or L' $\leq_{\mathbb{L}}$ L (**Comparability**).

We can then also write H = H($\mathbb{L}$, $\leq_{\mathbb{L}}$). For $\varkappa \in \mathrm{L}_{\mathrm{k}}$ and $\lambda \in \mathrm{L}_l$, we can also write $\varkappa \leq_{\mathbb{L}} \lambda$, if $\mathrm{L}_{\mathrm{k}} \leq_{\mathbb{L}} \mathrm{L}_l$, for k, $l \in$ {0, 1, . . . , m}. In this context, we can identify the basic set V ≡ L_0.

3.2 Interpretation & Application

Graphs are the most basic type of a relational mathematical model. In the simplest sense, nodes can be used to represent entities and edges indicate the pairwise relationships between them. In the case of hypergraphs, hyperedges can be used to represent a collection of objects with the same set membership, thus capturing higher order relationships that naturally occur in complex systems. The concept of order in multi-partite graphs and hierarchical hypergraphs can be used to differentiate distinct groups and denote the inherent structure of hierarchy in these systems.

In the context of food and nutrition, we propose a hierarchical hypergraph-based model called “FoodWeb Hierarchical Hypergraph Model”. Firstly, let’s consider a basic set consisting of all the essential nutrients, such as protein, fat, carbohydrates, vitamins, minerals, etc. as well as energy (commonly known as calories). Since these nutrients are the most fundamental building blocks of all foods, we can represent any particular food item as a hyperedge, i.e. a collection of basic nutrients according to its specific chemical composition. Furthermore, we can also represent any combination of food items as a hyperedge, for instance, a recipe consisting of various food ingredients, or a shopping basket containing groceries for a family’s weekly menu, and even the annual food consumption of a community or country. Thus, we can build up the order of levels in our FoodWeb Hierarchical Hypergraph Model.

In addition, we also introduce ecological cost factors in our hierarchical hypergraph in order to take into consideration various environmental impacts associated with food production and consumption. By adding ecological cost factors such as greenhouse gas emissions (GHG), land and water use, eutrophication, etc., in the basic set, we can track individual consumers’ consumption habits and account for the overall impact of certain diet patterns on the environment.

To make our FoodWeb Hierarchical Hypergraph have an internal level set order, we introduce edge weights. We can model each individual food item as a weighted hyperedge comprising all the essential nutrients and ecological factors in the basic vertex set. The edge weights encode the specific numerical quantity of each basic set node encapsulated by the hyperedge. Interestingly, these hyperedges can also be considered as internal nodes of a tree, with the basic set of vertices as leaf nodes. In this way, we can perceive FoodWeb

Hierarchical Hypergraph as a collection of trees with common shared leaf nodes. In our case, the leaf nodes correspond to the basic set. There is also an internal level set order for different trees, and it is easy to see that FoodWeb Hierarchical Hypergraph satisfies the level preorder conditions. More specifically, it can have a linear level set order, when the quantities of basic nutrients or ecological costs associated with each food item are compared numerically. However, if we define different levels to correspond to commonly applied concepts in real life, then a partial level set order is more appropriate. For example, we can define one level as “Food Groups” and another level as “Recipes”; these different levels need not to be compared explicitly, hence a partial order suffices here.

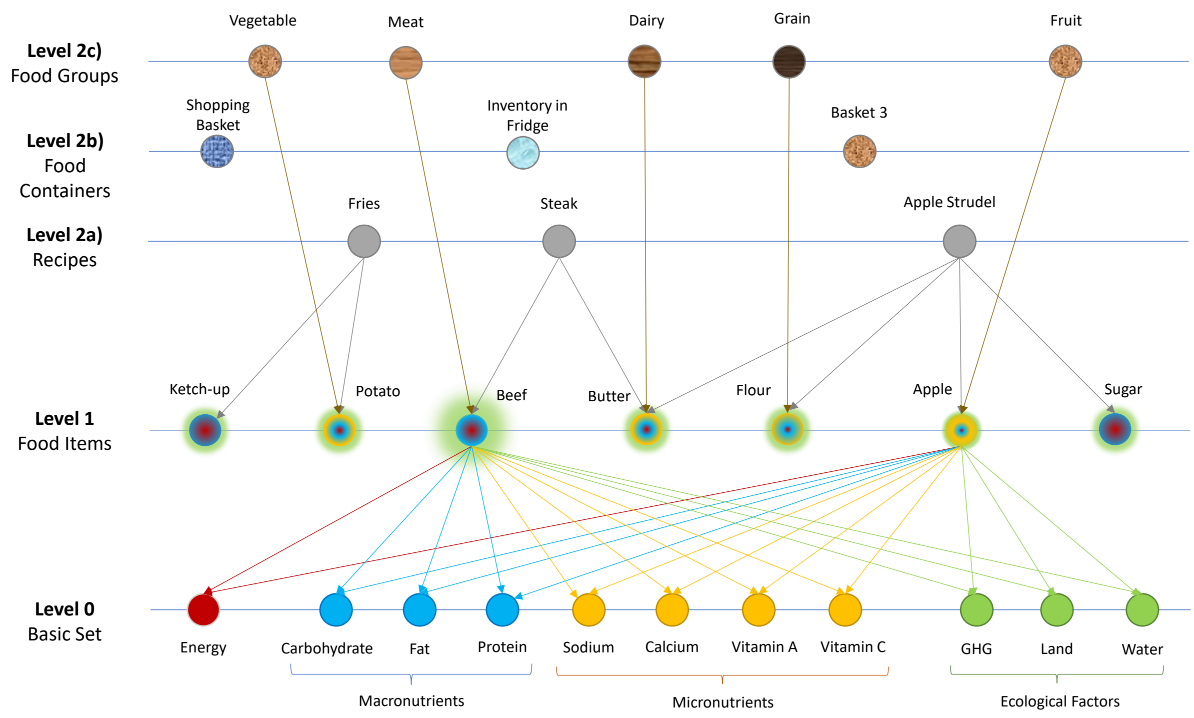

Figure 3. An illustration of FoodWeb Hierarchical Hypergraph. Level 0 represents the basic set consisting of nodes of energy, nutrients, and ecological cost factors. Level 1 represents food items, with each node encapsulating the food items specific composition in terms of energy, nutrients, and ecological impact. Level 2 represents combinations of food items; it can be interpreted as recipes or food groups, depending on the application in real life scenarios.

Furthermore, we can extend the idea of hierarchy and incorporate time evolution in our FoodWeb model. Hyperedges can be created with an explicit temporal component to represent individual consumer’s food consumption over a specific period of time. As such, we can model the aggregation of food, nutrition, and ecological costs over a time period for an individual consumer or for an entire population. By introducing rules to describe various diet habits and consumption patterns, we can run model simulations to study the potential impacts on population health as well as on the environment under different scenarios of policy changes in public food and nutrition advice. However, significantly more research is needed to develop this idea further and transform this FoodWeb Hierarchical Hypergraph into a dynamical model.

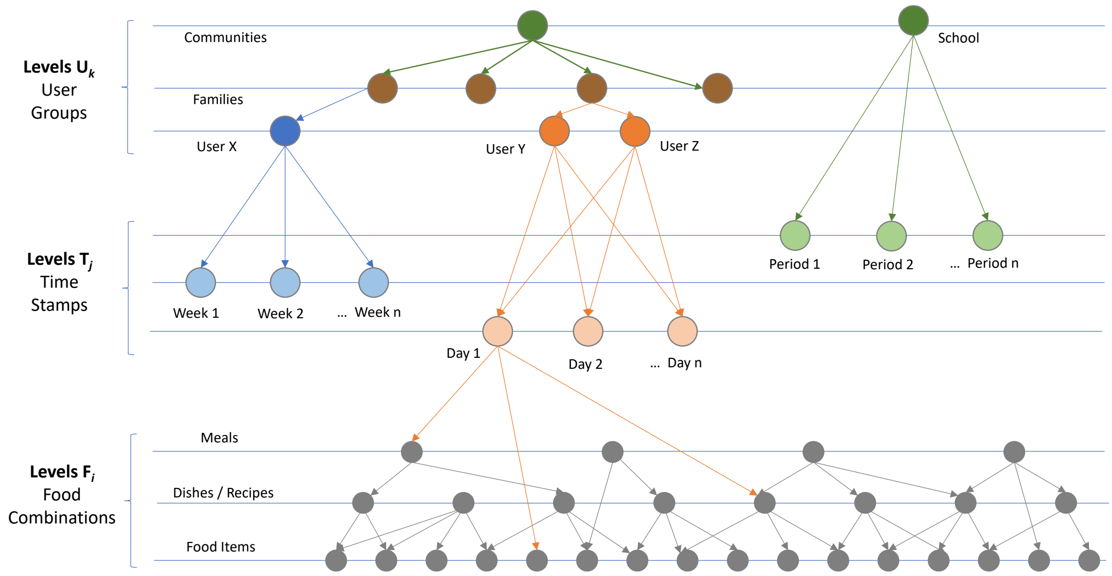

Figure 4. An illustration of hierarchical levels incorporating time stamps and user groups.

In this section, we have established the basic structure of FoodWeb Hierarchical Hypergraph. We model the energy, nutrients, and ecological costs as the elements in the basic set and we use hyperedges to model food items as a combination of various basic set elements. By introducing edge weights, we associated a hyperedge with its constituent nodes in a quantitative manner. We also demonstrate building up the hierarchical levels by applying the concept of encapsulation. Thus, we can use hyperedges to represent not only collections of food items, but also individuals as well as groups of individuals sharing certain consumption characteristics.

4. FoodWeb Hierarchical Hypergraph Model

In this section, we show the implementation of FoodWeb Hierarchical Hypergraph Model in Neo4j. We start by discussing the sourcing of data on food, nutrients and ecological factors from various databases and the challenges we have encountered. We then explain the system set-up for graph modelling in Neo4j and Python. We also introduce a special adaptation of using 'star-node' to represent hyperedges. Actual model implementations of food items, meals, individual consumers as well as system evolution will also be discussed in detail here.

4.1 Data Collection and Challenges

4.1.1 Food and Nutrition

With regards to data on food and nutrition, we have sourced from the US Department of Agriculture (USDA) Nutrient Data Laboratory website[1]. The latest version of the database, Standard Reference Release 28 (SR28), contains data on 8,789 food items and up to 150 food components. The food items include not only the raw ingredients but also common packaged food and restaurant meals. For the purposes of this project, we have incorporated all 8,789 food items from this database. However, for each food item, we have included only a limited subset of food components, i.e., 16 nutrient categories: calories, protein, fat, carbohydrate, fiber, sugar, cholesterol, calcium, iron, sodium, folic acid, vitamin A, vitamin B6, vitamin B12, vitamin C and vitamin D. These nutrients were chosen because they are the most commonly encountered in our everyday life; however, the extension to a more comprehensive coverage can be easily achieved in future studies, as needed.

4.1.2 Recipes and Meals

For recipe-related information, we find that the BBC Good Food website[2] serves as a comprehensive data source. However, it does not have a publicly available API, hence we did not collect data from the website directly. Fortunately, we managed to find on GitHub[3] a subset of the BBC recipe database, which had already been collected and processed into JSON format. We channeled this subset of recipe data into our model. One limitation worth mentioning is that the recipes in the current database do not contain quantitative information

[1] https://data.nal.usda.gov/ag-data-commons-hierarchy/food-nutrition
[2] https://www.bbcgoodfood.com/recipes
[3] https://github.com/mneedham/bbcgoodfood/

regarding the ingredients – only a list of all ingredients' names are provided, but not the amounts needed for each recipe. Thus, we model only the recipes as a preliminary demonstration of what can be done.

In order to overcome the shortcomings of the recipe data, we have enriched the dataset by manually collecting actual dietary samples and handcrafting 24 meals with complete information on food components based on the USDA Nutrient Database. Furthermore, we created these 24 personalized meals based on 4 distinct personas with different dietary styles: Viola is a vegetarian, Harold is diabetic and needs to avoid sugar, Sophie loves sweets and junk food, and Mike is a heavy meat-eater. We use these personas to compare the environmental impacts of various individual diet profiles. We also illustrate how to track a user's diet over a period of time, thus retaining state for the evolving FoodWeb system.

4.1.3 Ecological Cost

Data on ecological factors associated with food is sparse and this is a major area of challenge for our model. We have come across only a few relevant research papers (Poore & Nemecek 2018 [10], Dooren 2018 [11], Clark & Tilman 2017 [12]) which have published numerical values for greenhouse gas (GHG) emission, land usage, water usage, etc. on selected food items. However, consistency amongst different sources is an issue, as the methodologies applied to account for various ecological factors vary from research to research. We have decided to source from Poore and Nemecek (2018) for our model, because theirs is the latest published paper with the most comprehensive coverage of food groups (42 in total).

Recognizing the challenges in sourcing consistent ecological factor data, we have only incorporated two such factors: GHG emission (measured by kg of CO_2 equivalent) and land use (measured by m^2·year). We have used the mean values for each factor from Poore and Nemecek's paper, which covers the entire input-output process from the origin of production to the end point consumption. We also recognize that in order to more accurately capture the environmental costs, we need to adjust for each specific food item of its origin and for each consumer of his/her location. However, this level of precision is beyond the scope of this project.

4.2 System Setup

We choose to build our FoodWeb Hierarchical Hypergraph model using Neo4j, which works natively with graphs in terms of both processing and storage. We set up a Python driver to run Neo4j's graph query language, Cypher, to write, process, store and retrieve data from the graph database.

However, there is a limitation in Neo4j that it cannot be used to model hypergraphs directly. Neo4j is essentially a property graph database, in which edges are always required to have a start node and an end node. This poses a challenge for implementing a hyperedge which connects any number of given nodes in a hypergraph. We tackle this problem by transforming a hyperedge to a 'star-node', which has edges reaching out to all the basic nodes that the original hyperedge intends to encapsulate. In this manner, we can also represent the aggregate weight on the hyperedge by attributing the relevant components to the respective edges of the star-nodes. Thus, a hyperedge and a star-node are isomorphic.

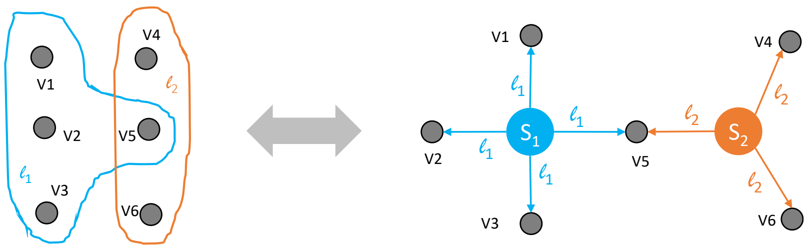

Figure 5. Using 'Star-Nodes' S_1 and S_2 to represent hyperedges l_1 and l_2, respectively. S_1 and l_1 are isomorphic and so are S_2 and l_2.

To model different levels of hierarchy, we can create 'star-nodes of start-nodes'. It offers flexibility to build up complexity in our FoodWeb model. We can store aggregated information as additional properties on the star-nodes for quick access and further processing. We can also attach a time stamp to the star-node to further implement system state for evolution and simulation purposes.

4.3 Model Implementation

4.3.1 Basic Set

As discussed in the last section, the basic set of our FoodWeb Hierarchical Hypergraph model includes nodes representing energy, nutrients and ecological factors. More concretely, we build 18 nodes in our basic vertex set. They represent Level 0 in the hierarchical structure. We distinguish each basic set node with a label specifying its category. We denote a basic set node as $b_i^L \in V_0$, where V_0 is the basic set, $L \in$ {Energy, Nutrient, EcoFactor}, i = 1, 2, …, 18.

18 Nodes in the Basic Set		Label
1	{name: Calorie, unit: kcal}	Energy
2	{name: Protein, unit: g }	Nutrient
3	{name: Fat, unit: g}	Nutrient
4	{name: Carbohydrate, unit: g}	Nutrient
5	{name: Fiber, unit: g }	Nutrient
6	{name: Sugar, unit: g }	Nutrient
7	{name: Cholesterol, unit: mg }	Nutrient
8	{name: Calcium, unit: mg }	Nutrient
9	{name: Iron, unit: mg}	Nutrient
10	{name: Sodium, unit: mg}	Nutrient
11	{name: Folic_Acid, unit: µg}	Nutrient
12	{name: Vit_A, unit: IU}	Nutrient
13	{name: Vit_B6, unit: mg}	Nutrient
14	{name: Vit_B12, unit: µg}	Nutrient
15	{name: Vit_C, unit: mg}	Nutrient
16	{name: Vit_D, unit: IU}	Nutrient
17	{name: GHG, unit: kg_CO2_eq}	EcoFactor
18	{name: Land, unit: sqm_year}	EcoFactor

Figure 6. Basic set nodes, with each assigned a label specifying its category

4.3.2 Food Items

We use ‘star-nodes’ to represent individual food items, which can be interpreted as Level 1 hyperedges in our FoodWeb Hierarchical Hypergraph model. The specific composition of the food item is modelled by establishing edges between the star-node and nodes in the basic set. The respective quantity of each food component in the food item is stored as a weight on the corresponding edge. We standardize all edge weights to the numerical amount of the basic set component per 100g of the food item. The unit of the edge weight is consistent with the unit specified on the basic set node. If the food item does not contain a specific food component, then there will not be an edge present between the star-node and that basic node representing that component.

Food Item	Contains per 100g	Basic Components
{name: APPLES}	{amount: 57.0}	{name: Calorie, unit: kcal}
	{amount: 13.68}	{name: Carbohydrate, unit: g}
	{amount: 0.25}	{name: Protein, unit: g}
	{amount: 0.12}	{name: Fat, unit: g}
	{amount: 2.3}	{name: Fiber, unit: g}
	{amount: 10.37}	{name: Sugar, unit: g}
	{amount: 7.0}	{name: Calcium, unit: mg}
	{amount: 0.12}	{name: Iron, unit: mg}
	{amount: 1.0}	{name: Sodium, unit: mg}
	{amount: 28.0}	{name: Vit_A, unit: IU}
	{amount: 0.049}	{name: Vit_B6, unit: mg}
	{amount: 0.04}	{name: GHG, unit: kg_CO2_eq}
	{amount: 0.06}	{name: Land, unit: sqm_year}

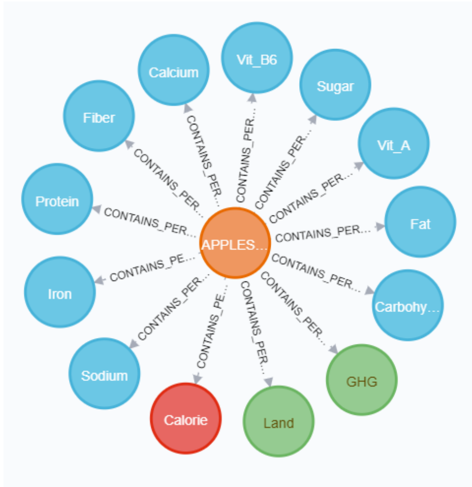

Figure 7. The table on the left shows the edge information between the star-node representing raw apples and its respective basic component nodes. The standard amount is per 100g of the food item. The chart on the right shows the responding visualization in Neo4j graph database.

We denote a food item node as $f_j^N \coloneqq \sum_{i=1}^{i=18} b_i w_i^j$, where N describes the node properties (such as name and description of the food item), j is the index of the food item in the database; b_i represents basic set node with i = 1, 2, …, 18; w_i^j is the edge weight associating food item f_j^N with each of its constituent basic nodes b_i.

4.3.3 Meals

We can use a 'star-node of star-nodes' to represent a meal which consists of a number of different food ingredients. Similar to how we constructed individual food items, we now extend edges from the star-node representing the meal to the star-nodes representing all of its ingredients. We add the edge weight to encode the constituent quantity, more specifically the amount is represented as a multiple of 100g of each food item. Each star-node representing a meal is therefore a Level 2 node, since it is connected to the basic set nodes through the Level 1 nodes. We can also collapse a Level 2 node to a Level 1 node by establishing direct links with the basic set nodes without going through other Level 1 nodes. This would be suitable only for meals that always have a fixed amount of each ingredient. In order to accommodate the flexibility in adjusting certain recipes, we suggest keeping the meal nodes at Level 2. Thus, we can personalize our favorite meals by editing the edge weights and update the node properties accordingly.

We denote a meal as $m_{\mathrm{k}}^{M} \coloneqq \sum_{j\in\{m\}} f_j^N w_j^{\mathrm{k}}$, where M describes the meal node properties (such as recipe name, description of the meal, special notes regarding food preparation, etc.), k is the index of the meal node in the database; f_j^N represents food items that the meal encompasses with $j \in \{m\}$ indicating the set membership; w_j^{k} is the edge weight associating the meal node m_{k}^{M} with each of its constituent food item nodes f_j^L.

The property on the star-node can contain summarized information with regards to the basic nodes. The information is aggregated by traversing its constituent food item nodes which connect with the basic nodes. Mathematically, we can write $m_{\mathrm{k}}^{M} = \sum_{j\in\{m\}} \sum_{i=1}^{i=18} b_i w_i^j \, w_j^{\mathrm{k}}$. The edge weights on each segment of the path to the specific basic node, w_i^j and w_j^{k}, are collected and processed, based on which the summarized information with respect to a certain basic node is then calculated. Depending on the ultimate purpose of the model, we can dynamically calculate only those that are of interest.

Meal	Contains_per_serving	Food_Items
{ name: Breakfast_1, type: Vege, kcal: 601.3, GHG: 0.135, Land: 0.285 }	{amount:2.65}	{name: BEVERAGES GREEN TEA}
	{amount:1.7}	{name: SILK VANILLA SOY YOGURT}
	{amount:0.1}	{name: CHIA SEEDS}
	{amount:0.1}	{name: PUMPKIN SEEDS}
	{amount:0.3}	{name: MIXED NUTS}
	{amount:1.5}	{name: BANANAS}
	{amount:0.7}	{name: BLACKBERRIES}

Figure 8. The table on the left shows the composition of a vegetarian breakfast. The node properties include the meal name, type, as well as other nutritional information, on a per serving basis. The exact amounts of calories, greenhouse gas emission, etc. associated with the meal are derived based on the quantities specified on the edges linking to food item nodes. For instance, the above breakfast includes 265g of green tea, 170g of vanilla soy yogurt, etc. The chart on the right shows the responding visualization in Neo4j graph database.

4.3.4 Individual Consumers & Time Stamps

By extending the concept of hyperedges, we can model an individual consumer as the collection of all the food and meals associated with him/her. We use an isomorphic 'star-node' to model the hyperedge that represents an individual. The node properties include the individual's personal information, such as name, diet style, daily calorie intake target, etc. We then connect this person node to all the meal nodes and food item nodes that he/she has. Like before, we can aggregate information for this person with regards to specific basic nodes. We can then compare two different individuals' diet style and the respective environmental impact.

In order to account for the evolution of time, we introduce 'time-stamp' as an edge weight from a person node to a meal node. The time stamps enable us to aggregate an individual's diet information over any period of time. This way, we can keep track of the adequacy of certain nutrient intake, monitor the cumulative diet pattern changes, and retain a system state for the overall model.

We denote a person as $p_{\mathrm{t}}^{D} \coloneqq \sum_{k \in \{t\}} m_k^M w_k^{\mathrm{t}}$, where D contains personal information regarding this individual, t indicates the time stamp associated with food consumption; m_k^M represents the meal that this individual has during the time period specified by the time-stamp, i.e., $\dot{k} \in \{t\}$; w_k^{t} = 1 indicating the time-stamp when the meal is consumed by the individual.

Person	Had	Meal
{name: Viola, diet_type: Vegetarian, daily_calorie_intake_target: 1800, special_notes: lactose intolerance}	{time_stamp: Day_1}	{name: Breakfast_1, kcal: 601.3, type: Vegetarian}
	{time_stamp: Day_1}	{name: Lunch_1, kcal: 872.0, type: Vegetarian}
	{time_stamp: Day_1}	{name: Dinner_1, kcal: 689.6, type: Vegetarian}
	{time_stamp: Day_2}	{name: Breakfast_2, kcal: 628.1, type: type: Vegetarian}
	{time_stamp: Day_2}	{name: Lunch_2, kcal: 895.4, type: Vegetarian}
	{time_stamp: Day_2}	{name: Dinner_2, kcal: 742.0, type: Vegetarian}

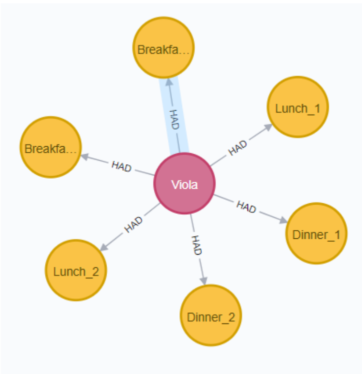

Figure 9. The table on the left shows a node representing an individual. The node properties include the person's name, diet type, daily calorie intake target, special notes on allergy and food intolerance, etc. The weight on the edges connecting the person node to various meal nodes denote the time stamps associated with the food consumption.

4.3.5 Group Aggregation

We can continue to build up hierarchies in the FoodWeb model by applying the same method of aggregating lower level nodes into higher level nodes. For example, we can represent a group by a 'star-node' with edges connecting to all of its members. We can store information specific to the group as a node property and information specific to each member as an edge property. The schema below demonstrates the overall structure of such an aggregator node.

We can visualize the hierarchical hypergraph model as a web, as shown in the diagram above, with the node representing the person at the center, surrounded by various meals and food item nodes, which in turn are derived from the basic set nodes consisting ultimately of energy, nutrients and ecological factors. Because of this web-like structure of the database, we name our project "FoodWeb".

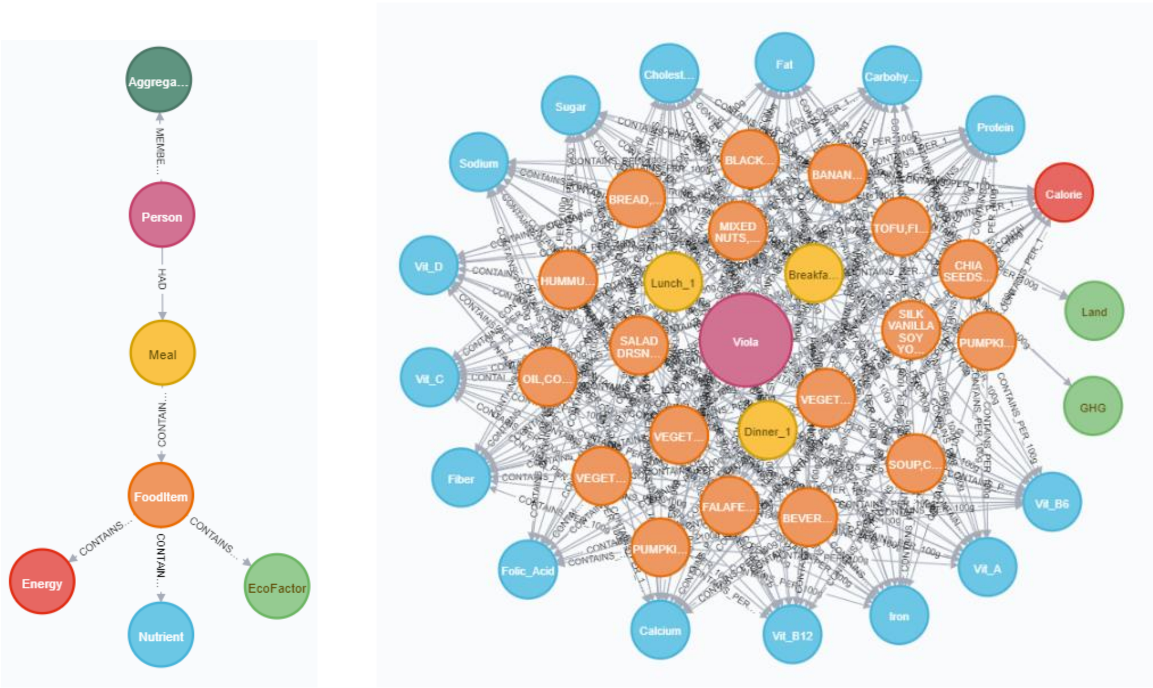

Figure 10. The diagram on the left shows the schema for an aggregator node. The diagram on the right demonstrates the web structure of the model with an individual person in the center surrounded by food items; the outer layer represents the basic set nodes of the hypergraph.

5. Applications and Further Research

In this section, we demonstrate two potential application scenarios using our FoodWeb Hierarchical Hypergraph model. We also discuss future research areas that may benefit from extending this model.

5.1 Scenario 1 – Food Intake Tracking

Imagine that Sophie has a sweet tooth. She reads about the potentially harmful effect associated with excessive sugar intake and she wants to monitor her sugar consumption level in order to stay healthy. She sets herself a target of 50g of sugar per day, which, according to the WHO's guideline, is equivalent to 10% of Sophie's daily total calorie intake. We can create an Aggregator node for Sophie to track her food intake for a specified period. A code snippet is provided below.

```
# Output warning message of breaching daily intake limit - Sophie Sugar
with driver.session() as session:
# Set a limit for daily sugar target: 10% of total calorie intake (Source: UN Guidelines); 1g sugar = 4kcal
    session.run("MATCH (p:Person {name: 'Sophie'}) SET p.sugar_target = p.daily_calorie_intake_target * 0.1/4 "
                )
# Create an Aggregator node to contain the summarized information
    session.run("CREATE (a:Aggregator {members: ['Sophie'], time_period:['Day_5','Day_6'], kcal:0, Sugar: 0}) "
                )
    session.run("MATCH (a:Aggregator {members: ['Sophie'], time_period:['Day_5','Day_6']}) "
                "UNWIND a.members AS name MATCH (p:Person {name: name}) "
                "MERGE (p)-[r:MEMBER_OF]->(a) "
                )
# Calculate the accumulative sugar intake by traversing the lower level nodes
    session.run("MATCH (a:Aggregator {members: ['Sophie'], time_period:['Day_5','Day_6']}) "
                "UNWIND a.members AS name UNWIND a.time_period AS time_stamp "
                "MATCH (p:Person {name: name})-[r:HAD {time_stamp: time_stamp}]-(m:Meal) "
                "WITH a, COLLECT(m) AS meals UNWIND meals as m "
                "MATCH (m)-[r1:CONTAINS_PER_Serving]->(f:FoodItem)-[r2:CONTAINS_PER_100g]->(v:Nutrient {name: 'Sugar'}) "
                "WITH a,r1, COLLECT([f,r2]) AS foods "
                "FOREACH ( f IN foods | SET a.Sugar = a.Sugar + r1.amount * (f[1]).amount ) "
                )
# Personalize warning message
    session.run("MATCH (p:Person {name: 'Sophie'})--(a:Aggregator {members: ['Sophie']}) "
                "WHERE a.Sugar > size(a.time_period) * p.sugar_target "
                "WITH a, round(a.Sugar - size(a.time_period) * p.sugar_target) AS overflow "
                "SET a.Warning = 'Sophie, you have consumed '+ toString(overflow) + "
                "'g more sugar than recommended in this period. Please consider reduction and stay healthy!'"
                )
```

Figure 11. Code snippet for tracking Sophie's sugar intake over a 2-day period.

We can then run the query below, which outputs a visual display of all the food containing sugar that Sophie has consumed during the specified 2-day period. It will alert Sophie when the amount of sugar in her food exceeds her pre-set target with a warning message: *“Sophie, you have consumed 151g more sugar than recommended in this time period. Please consider reduction and stay healthy!”* This will hopefully nudge Sophie to take better care of her diet and become more aware of food items that contain excessive sugar.

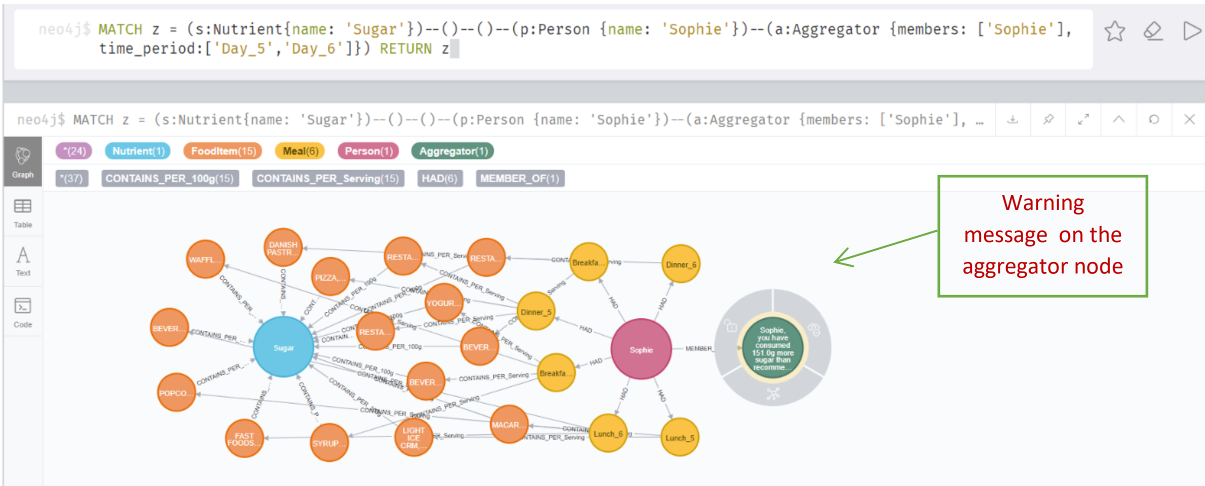

Figure 12. Query illustration for monitoring Sophie’s sugar intake. A warning message is issued on the aggregator node when her cumulative sugar intake level exceeds the pre-set target.

5.2 Scenario 2 – Recipe suggestion

Imagine Harold in the kitchen. He opens the fridge and discovers that there are only a few food items left: eggs, tomatoes, onions, and bacon. He wants to find a recipe which maximizes use of his existing inventory and minimizes the amount of extra ingredients. By running the below query, our FoodWeb model suggests suitable recipes as well as a list of the additional ingredients needed:

1. Greek salad omelette (additional ingredients: olive, feta cheese, olive oil & parsley)
2. Beef burger (additional ingredients: minced beef, burger bun and vegetables)

With this, a user could optimize his/her grocery inventory and reduce food waste.

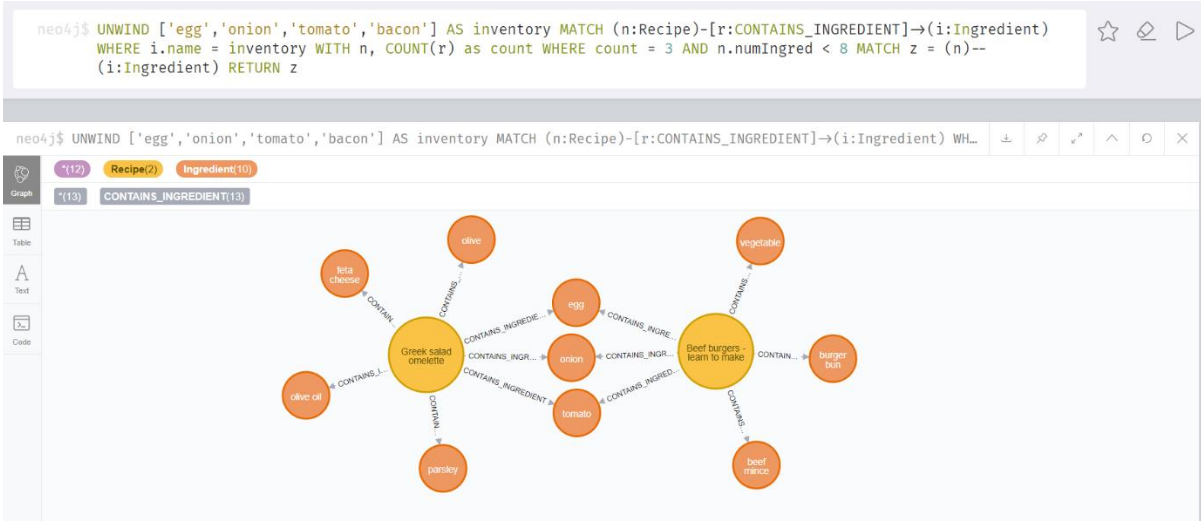

Figure 13. Query illustration for recipe recommendation based on limited available ingredients.

5.3 Potential Model Extensions

Due to the inherent flexibility in hierarchical hypergraphs, there are many potential extensions to FoodWeb. We explore a few of these below.

One application is to enlist the help of dietitians and nutritionists and collect data on real populations to study the dynamics of diet patterns vs. public health. One can then design meaningful attachment rules and use FoodWeb to run simulations of changing diet trends of a population. One can also extend our model further to study the Diet Problem [13] and try to find algorithms that optimize diets to be nutritionally balanced, environmentally sustainable and culturally acceptable.

Weight-loss is difficult to achieve in practice as plans often severely restrict the types of foods that can be eaten in any given day. The monotony of meal choices leads to frustration and cheating, which inhibits progress towards achieving weight targets. FoodWeb would enable patients to easily comprehend how each food item impacts their overall caloric balance and facilitate making smarter choices, e.g. replacing a high-calorie piece of cake with a more healthy piece of fruit.

Another possibility to commercially extend the model is to seek collaboration with food manufacturers and retailers in order to incorporate information on pricing, expiration dates, ecological costs, etc. into the database. This would allow supermarkets to help not only the health-conscious but also environmentally responsible consumers to make better informed purchasing decisions. Moreover, it could also help supermarkets do smart promotions. With algorithms such as the recipe suggestion query in FoodWeb, recommendations of complementary food items could be made to shoppers dynamically based on the existing contents of their shopping baskets. This would lead to better customer satisfaction and improved profitability.

6. Conclusion

This paper has explored the mathematical concepts of hierarchical hypergraphs through modelling a complex system, namely the food and nutrition system. FoodWeb Hierarchical Hypergraph model has been implemented in Neo4j to illustrate how to capture the multi-dimensionality and the intricacy of the food system by a hierarchical hypergraph structure. In addition, various use cases of how people can interact with FoodWeb to obtain relevant information at different levels of abstraction are demonstrated. Furthermore, potential model extensions for future research and commercialisation are also highlighted. In conclusion, hierarchical hypergraphs are a very promising tool for modelling the complexity of the food system. More research should be conducted, especially with regard to developing them further into dynamical systems.

References

1. Kirkilionis. *Mathematical Structures of Complex Systems (Lecture Notes),* Heidelberg University (2019)

2. Kirkilionis & Domijan, *Graph theory and qualitative analysis of reaction networks* (2008)

3. Estrada & Rodríguez-Velázquez. *Complex Networks as Hypergraphs,* arXiv preprint physics/0505137, (2005)

4. Scioni, et al. *Hierarchical Hypergraphs for Knowledge-centric Robot Systems,* Journal of Software Engineering for Robotics (2015)

5. Kahl. *Software Evolution via Hierarchical Hypergraphs with Flexible Coverage* (2001)

6. Ancona & Floriani. *A hypergraph-based hierarchical data structure and its applications,* Computational Mechanics Publications, Adv. Eng. Software (1989)

7. Wasserman & Faust. *Social Network Analysis,* Cambridge Univ. Press (1994)

8. Tan, et al. *Using Rich Social Media Information for Music Recommendation via Hypergraph Model,* ACM Trans. Multimedia Comput. Commun. (2010)

9. Langellier, et al. *Complex Systems Approaches to Diet: A Systematic Review.* Am J Prev Med. 2019

10. Poore & Nemecek. *Reducing Food's Environmental Impacts through Producers and Consumers,* Science (2018)

11. Dooren. *A Review of the Use of Linear Programming to Optimize Diets, Nutritiously, Economically and Environmentally*, Frontiers in Nutrition (2018)

12. Clark & Tilman. *Comparative Analysis of Environmental Impacts of Agricultural Production Systems, Agricultural Input Efficiency, and Food Choice,* Environ. Res. Lett. (2017)

13. Buttriss, et al. *Diet modelling: How it can Inform the Development of Dietary Recommendations and Public Health Policy,* British Nutrition Foundation Nutrition Bulletin (2014)