MEDINFO 2019: Health and Wellbeing e-Networks for All
L. Ohno-Machado and B. Séroussi (Eds.)

doi:10.3233/SHTI190178

Carnival: A Graph-Based Data Integration and Query Tool to Support Patient Cohort Generation for Clinical Research

David Birtwell[a], Heather Williams[a], Reed Pyeritz[b], Scott Damrauer[c], Danielle L. Mowery[a,d]

[a] *Institute for Biomedical Informatics, University of Pennsylvania, Philadelphia, PA, USA,*
[b] *Department of Medicine, University of Pennsylvania, Philadelphia, PA, USA*
[c] *Department of Vascular Surgery, University of Pennsylvania, Philadelphia, PA, USA*
[d] *Department of Biostatistics, Epidemiology, & Informatics, University of Pennsylvania, Philadelphia, PA, USA*

Abstract

Clinical research studies often leverage various heterogeneous data sources including patient electronic health record, online survey, and genomic data. We introduce a graph-based, data integration and query tool called Carnival. We demonstrate its powerful ability to unify data from these disparate data sources to create datasets for two studies: prevalence and incidence case/control matches for coronary artery disease and controls for Marfan syndrome. We conclude with future directions for Carnival development.

Keywords:

biomedical research, cohort studies, information storage and retrieval

Introduction

In biomedicine, clinical research studies are conducted to understand how best to prevent, diagnose, or treat disease in patients. A fundamental step to conducting a clinical research study is aggregating data and abstracting facts (i.e., clinical variables and treatment outcomes pertinent for defining a patient population for study). Patient data containing clinical facts (e.g., administrative, clinical, and genomic data) are generated at various points of care, and subsequently are stored across disparate, siloed resources. For example, large academic medical centers may store patient data within clinical registries, electronic health records, document stores, survey tools, and biobanks. Once study data have been integrated, the patient data must then be modeled to accurately represent and classify the patient's clinical case according to each study arm (i.e., case/control). For example, in a matched case/control study design, a clinical researcher might match patients based on age, biological sex, race, and genes, then their clinical data (disease, treatments, and outcomes) might be queried to understand disease progression and/or the effectiveness of therapeutic interventions. More complex match criteria can include periods of time, geographical locations, and environmental exposures. These clinical facts are stored in structured (hospital billing codes, laboratory, and medications) and/or unstructured (clinical notes) data formats.

Integrating such diverse data and aggregating patient clinical facts in a traditional relational database is challenging for several reasons. First, complex relationships of the data cannot easily be modelled into a sufficiently expressive relational schema. Second, complex relationship queries for generating and matching patient cohorts using relational databases could be hindered by suboptimal query responses (e.g., stalled or unfulfilled requests due to multiple-join statements) [1]. Graph databases such as Neo4j have been shown to support both semantic integration of disparate data [2] while improving query times over traditional relational databases such as MySQL [1]. Although graph databases have been used to integrate and represent biological disease networks (protein-protein interactions and drug-target pairs) and represent genotype-phenotype associations, few have demonstrated how graph databases might be leveraged to query heterogeneous data, clinical and genomic, to generate patient cohorts for clinical research studies [3–6].

In this work, we present ***Carnival***, a data unification technology that takes a novel approach — a strictly-formatted property graph database with a data model inspired by the Open Biological and Biomedical Ontology (OBO) Foundry ontologies — towards the integration of disparate data into a unified graph data resource. Carnival leverages this model to support the execution of common investigatory tasks, i.e., patient cohort identification, automated case/control matching, and the production of data sets for scientific analysis. For this work, we aim to 1) provide an overview of Carnival's infrastructure, 2) review a menu of predefined operations that can be combined and stacked to query, integrate, and reason over clinical and genomic data for creating case/control study populations, 3) present two case/control cohorts generated by Carnival, and 4) preview how Carnival will support semantic interoperability and intelligent queries using ontologies, leverage textual variables using natural language processing, and improve its usability with a graphical user interface for wider adoption by clinical research partners.

Methods

We describe the infrastructure, functionality, and utility of Carnival for supporting case/control studies from clinical and genomic data collected from the University of Pennsylvania Health System (UPHS). UPHS includes the first university-owned teaching hospital (Hospital of the University of Pennsylvania est. 1874) and the first hospital in the United States (Pennsylvania Hospital est. 1751). As part of the Penn Medicine BioBank (PMBB), over 60,000 UPHS patients have been consented for their clinical (EHR) and genomic (blood and tissue samples) data to be studied for clinical research. These biological specimens have been whole-exome sequenced by Regeneron Genetics Center and the context of their collection is represented using the Ontology for BioBanking (OBIB) [7]. Carnival leverages data from UPHS sources (e.g., Penn Data Store, a clinical data warehouse) and PMBB to generate datasets for clinical research studies.

Carnival

We describe our graph-based data unification tool named Carnival, so named because Carnival is a party of information and inspired by the squash. Carnival is a multi-layered, Groovy-powered application that includes data source adapters (*vines*) that extract lightly-processed data from their sources, a set of relational database utilities, and caching functions for querying source data, graph data writers (*reapers*) that attach extracted data to the Carnival data store (*graph)*, logical rule engines (*reasoners*) that modify and validate data within the graph, sample stratifiers (*algorithms*) to perform common tasks such as case-control matching, and data writers (*sowers*) that write data to external resources (see Figure 1).

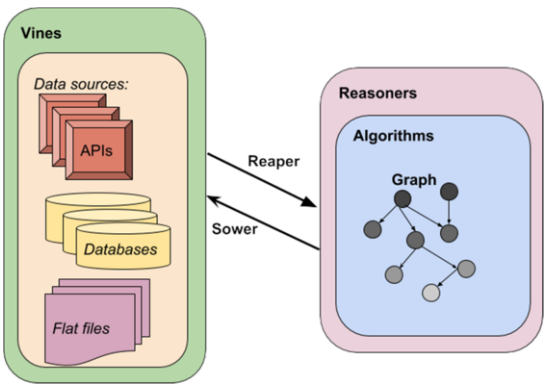

Figure 1– Carnival's Conceptual Framework

Vines

Data are pulled as needed from source systems (i.e., relational databases--Oracle, Microsoft SQL Server), application programming interfaces (API), or flat files (comma-separated value files). Typically, the vine contains methods (functions) that converse with the associated data source in its native language (e.g., SQL, Java, HTTP, etc.) to extract data. Each method produces a single matrix of data. Although there is nothing besides scale issues preventing the wholesale transfer of data from a source to the Carnival graph, vine functions are designed to extract the minimum data necessary for scientific study. For example, a vine for an EHR source might include an ICD Ever/Never method that accepts a set of ICD codes and patient identifiers then returns only the patient identifiers for patients whose medical record was assigned one or more of the ICD codes in the set.

Performing queries against large disparate data sources can be problematic for a variety of technical reasons: 1) connections can be dropped, 2) complex queries can take hours to return a result, and 3) queries that contain long lists of codes or identifiers can be cumbersome to compose. Carnival provides a suite of supportive classes and methods to address these difficulties, including SQL utilities for query composition, automated caching of vine method results, incremental caching to support restartable queries, and monitor threads that estimate time-to-completion of long-running queries.

Reapers

Reapers contain methods that extract data from source systems via vines and attach those data to the Carnival graph in a standardized way. A reaper method may be parameterized to limit its scope and call upon any number of vines to gather source data. The execution of the reaper method itself is recorded in the graph, including the inputs and outputs of the process. For example, an ICD Ever/Never reaper method that accepts a set of ICD codes and patient identifiers would execute a series of tasks: 1) create a vertex in the graph to represent the execution of the reaper method, 2) record the start time, finish time, and input ICD codes, 3) create links to the patient identifiers as inputs, 4) create vertices to represent the ICD Ever/Never status for each patient, 5) link those statuses to the appropriate patients, and 6) create links to those statuses from the execution vertex to mark them as outputs.

Graph

All data about a patient and metadata describing Carnival's processes are represented in Carnival's property graph structure. These highly-connected graphs contain meta information regarding the execution of the reapers, algorithms, and sowers queried and traversed using Tinkerpop-Gremlin and Cypher. We drew upon previous work in ontology modelling as inspiration for the Carnival graph data model: the Ontology for Biobanking (OBIB), the Ontology for Biomedical Investigations (OBI), and the Basic Formal Ontology (BFO) [7–9]. For example, BFO introduces the term Process, which denotes an event that occurs in a time and place. OBI introduces Planned Process, which extends Process to include a pre-defined plan, participants, inputs, and outputs. In the Carnival graph, healthcare encounters are modelled as planned processes, where participants include the patient and clinician and the outputs may be diagnoses and medications.

Reasoners

Reasoners apply logical rules to the graph to make modifications or additions to data. Reasoners execute their logic against the graph to validate that the graph is consistent with the reasoner logic. For example, a reasoner might assign a Boolean classification to patients as having or not having a disease state based on whether they have ever been assigned any of a set of ICD codes. The reasoner logic would check the ICD Ever/Never statuses of each patient and assign the appropriate Boolean disease state classification. The validation functionality might check that no patients have been assigned to both the have-disease and does-not-have-disease classes in the final dataset, which represents a logical inconsistency.

Operational Algorithms

Carnival leverages a graph-based, stratified-sampling case/control algorithm to match case patients with control patients for clinical research studies (i.e., case and control strata groups with equal numbers of males between the age of 35-40). Strata creation is managed by a strata manager class that takes as input the criteria for group partitioning, creates strata and strata group vertices in the graph, and assigns patient vertices to the appropriate groups.

First, the strata sampler groups patients by primary strata. In each stratum, patients are exhaustively grouped into disjoint sets. Strata can be defined along any singular value extracted from a patient: a *numeric* (e.g., current body mass index [BMI]), a *string* (race), a *Boolean* (e.g., FBN1 gene loss of function), or an *enumerated value set* (e.g., the existence of 2 or more ICD codes). Strata can be defined by a range (e.g., 20-29) for numerics and by multiple values grouped into the same strata group for enumerated values or strings. Each stratum also has a group for undefined or unknown values. Second, the cohort matcher then takes as input: patient cohorts for cases, candidate controls, and the primary strata that correspond to the desired matching criteria. The primary strata are combined to make a compound stratification that characterizes the cohorts for matching. For each strata group in the compound stratification, a number of patients in the candidate control cohort are selected as controls

corresponding to the number of patients in the case cohort from the same strata group. Additional criteria can also be provided to prioritize which of the controls within a group would be selected if there are more potential controls than necessary. For example, in a study where specimens associated with the patients will be expended, it may be advantageous to prioritize controls that have more specimens available to maximize the utility of PMBB specimens.

Sowers

Sowers extract data from the Carnival stratified case/control population graph and write it to external resources (e.g., databases, applications, and flat files). Data imported from disparate sources and linked in the Carnival graph may be queried producing data views useful for export to external systems. For example, an investigator may have an approved study that monitors patients who have a specific disease state, the data for which are tracked in a REDCap project. A Carnival sower can query the graph for disease states and write those data to the REDCap project via the REDCap API. In this way, Carnival supports extract, transform, and load (ETL) operations. This functionality can be leveraged to add new patient information (e.g., genomic facts into the EHR).

Clinical Data Elements for Case/Control Matching

Currently, case/control cohorts can be defined and analyzed using several clinical data elements shown in Table 1.

Table 1– FBN1 Controls

Clinical Data Element	Example
Demographics	current age, biological sex, race/ethnicity
Vital signs and risk factors	current BMI, weight, height, blood pressure, smoking status
Medications	administered at a given time or during time window
Specimen collection contexts	age or BMI at the time of specimen collection, medications administered before/after date of specimen collection
Hospital administration	ICD9/10, CPT procedures, DRG codes, and fee codes
Clinical status	death status
Genetic data	loss of function genes

Building a Cohort Using Operational Building Blocks

Carnival contains a number of operational algorithms or building blocks that can be combined to support cohort identification and case-control selection: *graph building, reasoners,* and *patient cohort algorithms*. *Graph building algorithms* aggregate data, via vines and reapers, and attach those data to the graph, examples include:

- Instantiating patients and encounters with identifiers
- Computing phecode assignments according to Phewas.org
- Computing BMI closest to the patient's PMBB recruitment date
- Obtaining the most recent and earliest healthcare encounter for each patient
- Calculating each patient's date of birth
- Gathering each patient's available specimens.

Reasoners operate over data in the graph. To support cohort building, Carnival contains a patient stratification algorithm that assigns patients to specific strata, which currently correspond to their current age and biological sex. A case-control matching algorithm creates a cohort of controls for a predefined set of cases based on a cohort of candidate controls and selected strata. *Patient cohort algorithms* facilitate the creation of patient cohorts and perform set operations on them. Current patient cohort algorithms include:

- Creating a cohort based on a set of identifiers, e.g., medical record number (MRN), enterprise master patient index (EMPI), or encounter identifier
- Creating a complement cohort that contains all patients not in an existing cohort
- Creating a cohort containing all patients who have been diagnosed with any of a set of ICD codes
- Creating a cohort containing all patients who have a loss of function mutation for a given gene.

We describe how Carnival uses clinical data elements and operational algorithms to support two case/control studies leveraging patient data from UPHS and PMBB.

Identifying Markers for Coronary Artery Disease

For the first case/control study, Carnival was leveraged to define the study populations for discovering biological markers for coronary artery disease. In the PMBB, blood specimens are collected at the time of enrollment. *Cases* are defined as patients without coronary artery disease (CAD) codes (410, I21) before enrollment into PMBB, and subsequently, have a myocardial infarction (MI), coronary revascularization, or another CAD event. *Controls* are defined as individuals without CAD codes who have had no MI or CAD events after PMBB enrollment.

Both cases and controls must have plasma and either buffy coat or DNA specimens available. A report of demographic and phenotype data relative to the date of enrollment was requested. Carnival was leveraged to generate the case/control list and report the following patient-specific information:

- Age at PMBB enrollment
- Current age
- Biological sex
- EHR race
- PMBB recruitment location
- Height, weight, and BMI measurement closest to PMBB enrollment
- Count, min, max, and median of lab results for: glucose, fasting glucose, hemoglobin A1C
- Count of distinct dates diabetes codes were assigned before PMBB enrollment
- Code and age of the patient for the first code matching the time filter for the most recent CAD codes before PMBB enrollment, most recent MI code before PMBB enrollment, first MI or CAD code after PMBB enrollment, and first revascularization code after PMBB enrollment.

The principal investigator (PI) manually reviewed the report and patient charts for the final case selection. When patient cases were chosen, Carnival generated two sets of controls. *Control group 1*: a 1:1 control instance matched to the case population on biological sex, current age +- 4 years, BMI closest to enrollment +- 6 points, and recruitment location. *Control group 2:* a frequency control matched to the case population on sex,

recruitment location, current age, and BMI closest to PMBB enrollment.

Defining Marfan Syndrome Study Controls

For this second case/control study, *Cases* were predefined by using the MRNs of patients provided by the PI. *Controls* were defined by Carnival. The Marfan Syndrome gene loss of function data value was critical for defining the control cohort. Candidate control matches were queried from PMBB based on age and biological sex. For each patient, Carnival determined whether a genetic assessment was conducted for the Marfan syndrome FBN1 gene and validated whether the outcome was negative to ensure that all patients within the control set were FBN1 negative.

Results

We leveraged Carnival to generate cohorts for two diseases: coronary artery disease and Marfan syndrome.

Identifying Markers for Coronary Artery Disease

Based on the CAD definitions for prevalence and incidence and specimen availability, Carnival identified 425 candidate case and 5,872 candidate control populations. The principal investigator conducted chart review to create the final 170 selected cases, then Carnival generated the final control groups 1 (n=170 cases) and 2 (n=60 cases) for these cases. Carnival identified a 1:1 case/control match (see Table 2).

Table 2– Coronary Artery Disease Cases and Controls

Sex	Age	Candi-date Cases	Selected Cases	Candi-date Controls	Control Group 1	Control Group 2
Female	<20	0	0	7	0	0
Female	20-29	0	0	486	0	1
Female	30-39	5	2	681	2	1
Female	40-49	8	7	599	6	2
Female	50-59	13	11	731	13	6
Female	60-70	39	20	814	23	3
Female	>70	68	31	450	27	10
Male	<20	0	0	1	0	0
Male	20-29	2	2	98	1	0
Male	30-39	3	2	156	2	2
Male	40-49	15	9	272	9	3
Male	50-59	40	24	529	23	5
Male	60-70	92	27	682	30	18
Male	>70	140	35	366	34	9
Totals		**425**	**170**	**5,872**	**170**	**60**

Defining Marfan Syndrome Study Controls

Using the exclusion criteria of patients who are not cases (case distribution not shown) and who do not have a loss of function mutation in the FBN1 gene, Carnival identified a 2:1 control/case match resulting in 146 selected controls (see Table 3).

Table 3– Marfan Syndrome Controls

Sex	Age	Selected Controls	Candidate Controls
Female	20-40	2	294
Female	40-50	12	300
Female	50-60	10	667
Female	60-70	18	1154
Female	>70	20	1718
Male	20-40	4	175
Male	40-50	12	343
Male	50-60	12	1004
Male	60-70	26	1892
Male	>70	30	2627
Totals		**146**	**10,174**

Discussion and Future Work

We demonstrated how Carnival's multi-layered framework, OBO foundry-inspired data model, and graph-based functionality can be leveraged to generate case/control matches and resulting datasets using PMBB and UPHS sources for two clinical research studies. We envision expanding Carnival's functionality 1) to improve semantic integration and intelligent query of data using ontologies, 2) to incorporate clinical data elements generated from clinical texts, and 3) to create a user-friendly interface to promote wider adoption of this tool by clinical research partners.

Improve Semantic Integration and Query using Ontologies

Carnival is a graph-based query tool that operates at the data level. We have partnered Carnival with TURBO technologies to provide richer semantic integration and reasoning among clinical data elements [10]. For example, patient diagnoses are currently defined using logical rules that operate over discrete diagnosis billing codes (e.g., ICD9 or ICD10). A single code rarely encapsulates a diagnosis for investigative purposes. Furthermore, compiling a list of diagnosis codes relevant to a particular disease or disease classification can be a daunting task. To accurately identify a diagnosis, TURBO has integrated ontologies (e.g., Monarch Disease Ontology) [11], to semantically link ICD codes to disease concepts. To provide semantic information services (i.e., returning a set of ICD diagnosis codes for a given disease classification), we have integrated Drivetrain [12], a TURBO technology that uses an RDF triple store and OBO Foundry ontologies [13]. This integration has permitted Carnival to operate at both the disease classification and diagnosis code levels.

Medication prescriptions have similarly benefitted from integration with Drivetrain. The Chemical Entities of Biological Interest (ChEBI) ontology [14,15] contains a rich semantic network of medication names, ingredients, and roles that Drivetrain links with medication order names in the UPHS EHR. Carnival operates over these elements, obviating the need for investigators to spend time deciphering individual medication order names. We aim to provide semantic integration over laboratory test results as well, when Drivetrain services for lab results become available.

Incorporate Clinical Data Elements from Clinical Notes

A wealth of clinical facts are locked within clinical free-text notes (e.g., discharge summaries, progress notes, radiology exams, and surgical pathology reports) [16]. We aim to integrate outputs from natural language processing tools including symptoms, signs, treatments, outcomes as well as their associated contexts

(negation, subject, temporality, uncertainty) to provide a richer clinical profile to infer each patient's disease state [17–21]. We will leverage the Observational Medical Outcomes Partnership (OMOP) Common Data Model (CDM) to promote interoperability with entities both within and outside UPHS.

Create a User-Friendly Interface to Promote Adoption

The current implementation of Carnival leverages server-side technology. There is currently a text-based command line interface that supports basic operations. Future work will include the addition of a web-services API and a browser-based Javascript user interface [22]. We will integrate Neo4j-based user interfaces to the graph (e.g., Neo4j Browser) [23]. We will also employ Javascript libraries (e.g., D3) [24] for graph visualization and user-driven data exploration.

Conclusions

We conclude that graph-based technologies can be utilized to integrate and query disparate patient health data to support complex, clinical research studies.

References

[1] B.-H. Yoon, S.-K. Kim, and S.-Y. Kim, Use of Graph Database for the Integration of Heterogeneous Biological Data, *Genomics Inform.* **15** (2017) 19–27.

[2] H. Ulrich, A.-K. Kock-Schoppenhauer, P. Duhm-Harbeck, and J. Ingenerf, Using Graph Tools on Metadata Repositories, *Stud. Health Technol. Inform.* **253** (2018) 55–59.

[3] A. Lysenko, I.A. Roznovăț, M. Saqi, A. Mazein, C.J. Rawlings, and C. Auffray, Representing and querying disease networks using graph databases, *BioData Min.* **9** (2016) 23.

[4] V. Touré, A. Mazein, D. Waltemath, I. Balaur, M. Saqi, R. Henkel, J. Pellet, and C. Auffray, STON: exploring biological pathways using the SBGN standard and graph databases, *BMC Bioinformatics*. **17** (2016) 494.

[5] W.S. Campbell, J. Pedersen, J.C. McClay, P. Rao, D. Bastola, and J.R. Campbell, An alternative database approach for management of SNOMED CT and improved patient data queries, *J. Biomed. Inform.* **57** (2015) 350–357.

[6] S. Mughal, I. Moghul, J. Yu, UKIRDC, T. Clark, D.S. Gregory, and N. Pontikos, Pheno4J: a gene to phenotype graph database, *Bioinformatics*. **33** (2017) 3317–3319.

[7] M. Brochhausen, J. Zheng, D. Birtwell, H. Williams, A.M. Masci, H.J. Ellis, and C.J. Stoeckert Jr, OBIB-a novel ontology for biobanking, *J. Biomed. Semantics*. **7** (2016) 23.

[8] Ontology for Biobanking | NCBO BioPortal, (n.d.). https://bioportal.bioontology.org/ontologies/OBIB (accessed November 25, 2018).

[9] Basic Formal Ontology (BFO) | Home, (n.d.). http://basic-formal-ontology.org (accessed November 25, 2018).

[10] C. Stoeckert, D. Birtwell, H. Freedman, M. Miller, and H. Williams, ICBO_2018_12: Transforming and Unifying Research with Biomedical Ontologies: The Penn TURBO project, in: International Conference on Biomedical Ontology (ICBO 2018), International Conference on Biological Ontology, 2018. http://icbo2018.cgrb.oregonstate.edu/.

[11] O.T. Wg, Monarch Disease Ontology. http://www.obofoundry.org/ontology/mondo.html (accessed November 25, 2018).

[12] The TURBO Ontology, *Turbo-Documentation*. https://pennturbo.github.io/Turbo-Documentation/turbo-ontology.html (accessed November 25, 2018).

[13] J.M. Hancock, OBO Foundry, in: Dictionary of Bioinformatics and Computational Biology, 2004.

[14] ChEBI (chemical entities of biological interest data based), in: Encyclopedia of Genetics, Genomics, Proteomics and Informatics, 2008: pp. 321–321.

[15] P. de Matos, R. Alcántara, A. Dekker, M. Ennis, J. Hastings, K. Haug, I. Spiteri, S. Turner, and C. Steinbeck, Chemical Entities of Biological Interest: an update, *Nucleic Acids Res.* **38** (2009) D249–D254.

[16] S. Velupillai, D. Mowery, B.R. South, M. Kvist, and H. Dalianis, Recent Advances in Clinical Natural Language Processing in Support of Semantic Analysis, *Yearb. Med. Inform.* **10** (2015) 183–193.

[17] OHDSI, OHDSI/CommonDataModel, *GitHub*. https://github.com/OHDSI/CommonDataModel (accessed November 25, 2018).

[18] D.L. Mowery, S. Velupillai, B.R. South, L. Christensen, D. Martinez, L. Kelly, L. Goeuriot, N. Elhadad, S. Pradhan, G. Savova, and W. Chapman, Task 2: ShARe/CLEF eHealth evaluation lab 2014, *CLEF 2014 Online Working Notes*. **1180** (2014) 31–42.

[19] D.L. Mowery, P. Jordan, J. Wiebe, H. Harkema, J. Dowling, and W.W. Chapman, Semantic annotation of clinical events for generating a problem list, *AMIA Annu. Symp. Proc.* **2013** (2013) 1032–1041.

[20] D.L. Mowery, B.E. Chapman, M. Conway, B.R. South, E. Madden, S. Keyhani, and W.W. Chapman, Extracting a stroke phenotype risk factor from Veteran Health Administration clinical reports: an information content analysis, *J. Biomed. Semantics*. **7** (2016) 26.

[21] D.L. Mowery, S. Velupillai, and W.W. Chapman, Medical diagnosis lost in translation: analysis of uncertainty and negation expressions in English and Swedish clinical texts, in: 2012 Workshop on Biomedical Natural Language Processing, Association for Computational Linguistics, 2012: pp. 56–64.

[22] M. Löbe, User Expectations of Metadata Repositories for Clinical Research, *Stud. Health Technol. Inform.* **253** (2018) 60–64.

[23] The Neo4j Browser: A User Interface Guide for Beginners, *Neo4j Graph Database Platform*. (n.d.). https://neo4j.com/developer/guide-neo4j-browser/ (accessed November 25, 2018).

[24] M. Bostock, D3.js - Data-Driven Documents, (n.d.). https://d3js.org/ (accessed November 25, 2018).

Address for Correspondence

David Birtwell: birtwell@pennmedicine.upenn.edu