Network Provenance History

The provenance history aspect of an NDEx network is used to document the workflow of events and information sources that produced the current network. API operations that create or update networks add default events to the provenance history. Applications can also explicitly modify the provenance history in order to customize events, controlling the granularity of events recorded and the level of detail captured.

Motivation

A network can represent assertions of biological relationships that are the results of experimental, analytic, or curation processes. Networks may in turn serve as inputs to further processes of analysis and model creation. If the workflow and dependencies on information sources are clearly documented, researchers may better understand the meaning of the relationships in the network and are better empowered if they wish to reproduce the analyses leading to the network. To achieve these goals, networks stored in NDEx can optionally include a provenance history aspect that can be accessed and managed via the NDEx API.

For example, a network might be derived by an algorithm which finds subnetworks based on experimental data mapped to entities in a reference network; in this case the application performing the analysis should record the analysis event in the provenance history of the output network, including references or descriptions of the algorithm used, the input experimental data, and a description of the input reference network.

For robustness, the provenance history stores descriptions of ‘ancestor' networks and other information sources, not just links to those resources. This preserves the utility of the provenance history in situations in which some or all of the input information sources are unavailable or have been modified since they were used in the workflow. Researchers (or algorithms) can inspect the provenance history of the current network to address questions about the status of all of the inputs to the workflow.

Related Work

NDEx network provenance history is similar in intent to Synapse Analytical Provenance (https://www.synapse.org/#!Wiki:syn2305384/ENTITY/62865)

Provenance History Structure

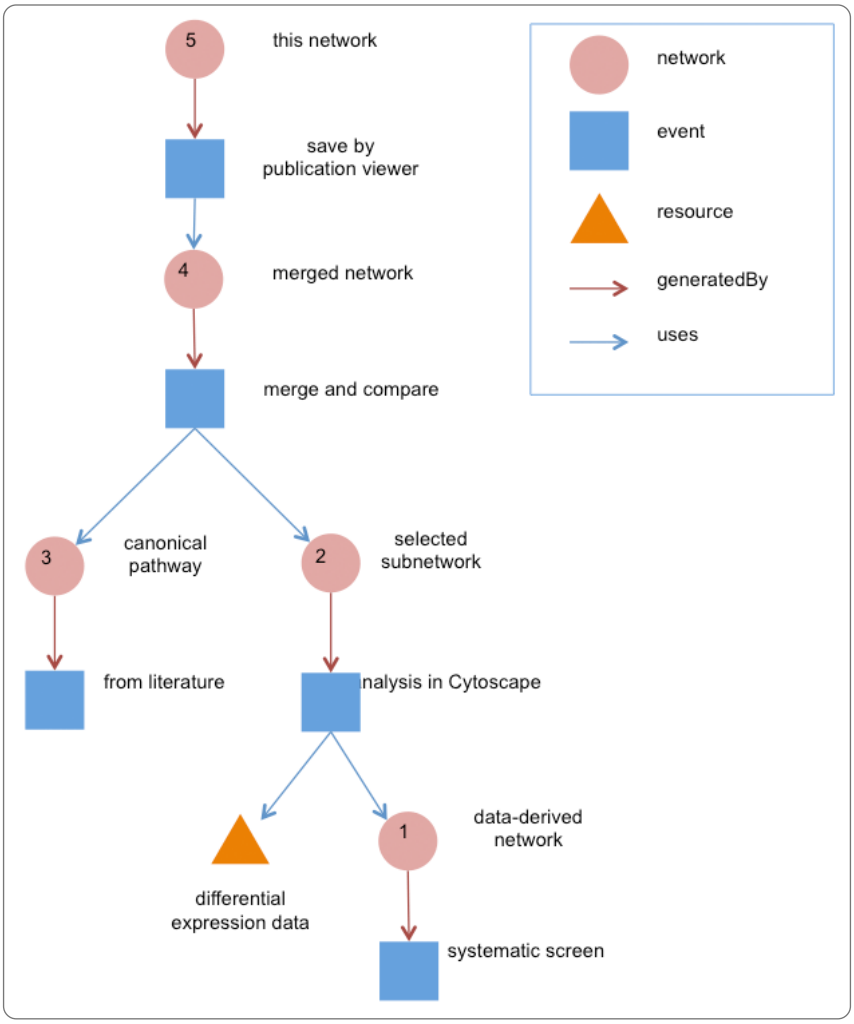
A provenance history is a tree structure containing ProvenanceEntity and ProvenanceEvent objects (Figure 1). It is serialized as a JSON structure by the NDEx API. The root of the tree structure is a ProvenanceEntity object representing the current state of the network. Each ProvenanceEntity may have a single ProvenanceEvent object that represents the immediately prior event that produced the ProvenanceEntity. In turn, linked to network of ProvenanceEvent and ProvenanceEntity objects representing the workflow history that produced the current state of the Network. The provenance history records significant events as Networks are copied, modified, or created, incorporating snapshots of information about "ancestor" networks.

ProvenanceEntity

- uri >> URI of the resource described by the ProvenanceEntity. This field will not be set in some cases, such as a file upload or an algorithmic event that generates a network without a prior network as input.
- creationEvent – ProvenanceEvent >> Has semantics of PROV:wasGeneratedBy
- properties >> array of SimplePropertyValuePair objects

ProvenanceEvent

- endingAtTime – timestamp >> Has semantics of PROV:endingAtTime
- startingAtTime – timestamp >> Has semantics of PROV:endingAtTime
- inputs – array of ProvenanceEntity objects >> Has semantics of PROV:used
- properties >> array of SimplePropertyValuePair objects



Provenance History and Network Equivalence

The provenance history can be used to infer network equivalence, whether a given network stored in NDEx has the same content as another network or an external resource. This is valuable since in the general case, computing equivalence by algorithm may be computationally expensive or could require network format-specific knowledge.

Two networks on NDEx servers may be inferred to be equivalent if the following conditions are met:

- one is the ancestor of the other in their provenance histories
- the events between the ancestor and descendent are all information preserving COPY operations.
- the ancestor has not been modified since the initial COPY operation.

Similarly, a network may be considered equivalent to an external source in either of the following cases:

- the network is the output of an UPLOAD event of a file derived from the external source and the external source has not been modified since the time of the upload.
- the network is an unmodified copy of a network meeting the above criteria.

Provenance Updates by NDEx API Operations

Seven REST API methods perform default updates to the provenance history of a network. They record basic information about the network (e.g. number of edges, nodes, title, description, version) and the event (e.g. type, time, username, first name, last name).

addNamespace /network/{networkId}/namespace

Records the added namespace name and value as an event property.

setNetworkProperties /network/{networkId}/properties

Records properties as event properties.

setNetworkPresentationProperties /network/{networkId}/presentationProperties

Records presentation properties as event properties.

updateNetworkProfile /network/{networkId}/summary

Records network name, description, and version as event properties and displays the current state as node properties.

createNetwork /network/asNetwork

No additional information recorded.

createNetwork /network/asPropertyGraph

No additional information recorded.

uploadNetwork /network/upload

Records filename as an event property.

Network Updates that do NOT Modify Provenance History

The provenance history is NOT updated when:

- Network membership information is changed or when network visibility (e.g. PUBLIC or PRIVATE) is changed.
- The provenance history is explicitly updated by the API.

Reading and Setting Provenance History

An application can read and alter the provenance history without adding any additional event to the provenance history. The API methods are:

getProvenance /network/{networkId}/provenance

setProvenance /network/{networkId}/provenance

Provenance Events vs. Network Modification Times

Any operation that modifies the network, including changes to visibility or provenance also changes the last modification date of the network.

Changes to network membership – what users have access to the network – do not modify the network itself and so do not change either the modification date or provenance history.

Properties of ProvenanceEntity and ProvenanceEvent objects

The standard fields in ProvenanceEntity and ProvenanceEvent objects correspond to relationships defined in the PROV-O ontology. Other property-value pairs can annotate these objects to provide more information about the entities and events. Any ad hoc pair of strings can be added as a property-value pair, and the properties used may be idiosyncratic to the recorded events and entities. However, the use of properties defined in the Dublin Core (DC) metadata annotations and the Provenance, Authoring and Versioning ontology (PAV) are preferred when applicable..

It is important to note a difference in the use of these ontologies in an NDEx provenance structure and the original intent. A ProvenanceEntity is a description of the referenced object, not the object itself. Therefore, a property such as "dc:title" that is asserted for a ProvenanceEntity refers to the original entity that the ProvenanceEntity represents. The provenance history references ancestor networks and other data sources but can also include self-contained descriptions of those objects that capture their state at the time they were used.

Dublin Core (DC) Properties

- dc:title
- dc:description
- dc:rights
- dc:rightsHolder
- dc:format

PAV Properties

- pav:retrievedFrom >> Direct retrieval – a COPY of the source network with no transformation of the content.
- pav:importedFrom >> Import with some transformation, as in a file UPLOAD where the source data is processed to create the network. The content reflects the external source but potentially has differences dependent on the import method.
- pav:derivedFrom >> The network was generated by an operation that transforms the content of the source.
- pav:sourceAccessedAt >> The network was generated by a transformation operation that consulted the source as part of the transformation.
- pav:version >> The version of the current network.
- pav:previousVersion >> The previous version. Note that this might be the version of a network that is not in the provenance history – a version could be created from new sources, not necessarily as a transformation of an earlier version.

Provenance History Use Cases

Copying Networks

In a copy operation, an application / utility creates a new network (the target) that encodes the same content as an existing network (the source).

In the resulting target provenance history, the root ProvenanceEntity represents the target and the copy operation is represented as a ProvenanceEvent of type COPY in which the output is the root entity and the input is a ProvenanceEntity representing the source.

The ProvenanceEntity representing the source and all of its prior entities and events are copied from the provenance history of the source.

Information stored in the provenance history about the source is intended to reflect the state of the source at the time of the copyand should not be updated to reflect subsequent changes in the source. Information about the source stored in the provenance history is thereby preserved, regardless of whether the source is later modified or deleted.

Upload / Import Network File

Upload is a special type of import, where the ProvenanceEntity for the source should store information about the uploaded file in the properties, such as the filename, file type, or data size.

Network Query / Filter

A network created by a query or other operation that retrieves part of the source is a common type of transformation operation. The new network is derived from the source.

Editing Operations

In any case where the source network has the same UUID as the target, the ProvenanceEvent is an edit of some type. Because the event can have both startingAt and endingAt properties, the editing process can span an arbitrary amount of time. The application managing the editing process can therefore control the granularity of the provenance history. For example, an editing application could represent a long sequence of edits in a verbose chain of events and intermediate states or it could simply keep updating the endingAt time as the edits continued. In both cases, the resulting provenance history would be a valid representation of the workflow, although one would capture greater detail than the other.

Translation of Network Identifiers

In the case where a utility creates a network that has content equivalent or homologous to the source but described in a different identifier system (such as gene ids replaced with corresponding gene symbols), an additional resource describing the identifier mapping is typically involved. In this case, the mapping resource is also an input to the ProvenanceEvent, and it is appropriate to use the property pav:sourceAccessedAt to describe the relationship.

Merging Networks

"Merging" in this context means a modification operation in which the information in network A is augmented by information coming from network B, or where a new network is created from both A and B. This creates a branched provenance history in which the ProvenanceEvent for the merge has two inputs, both network A and network B.