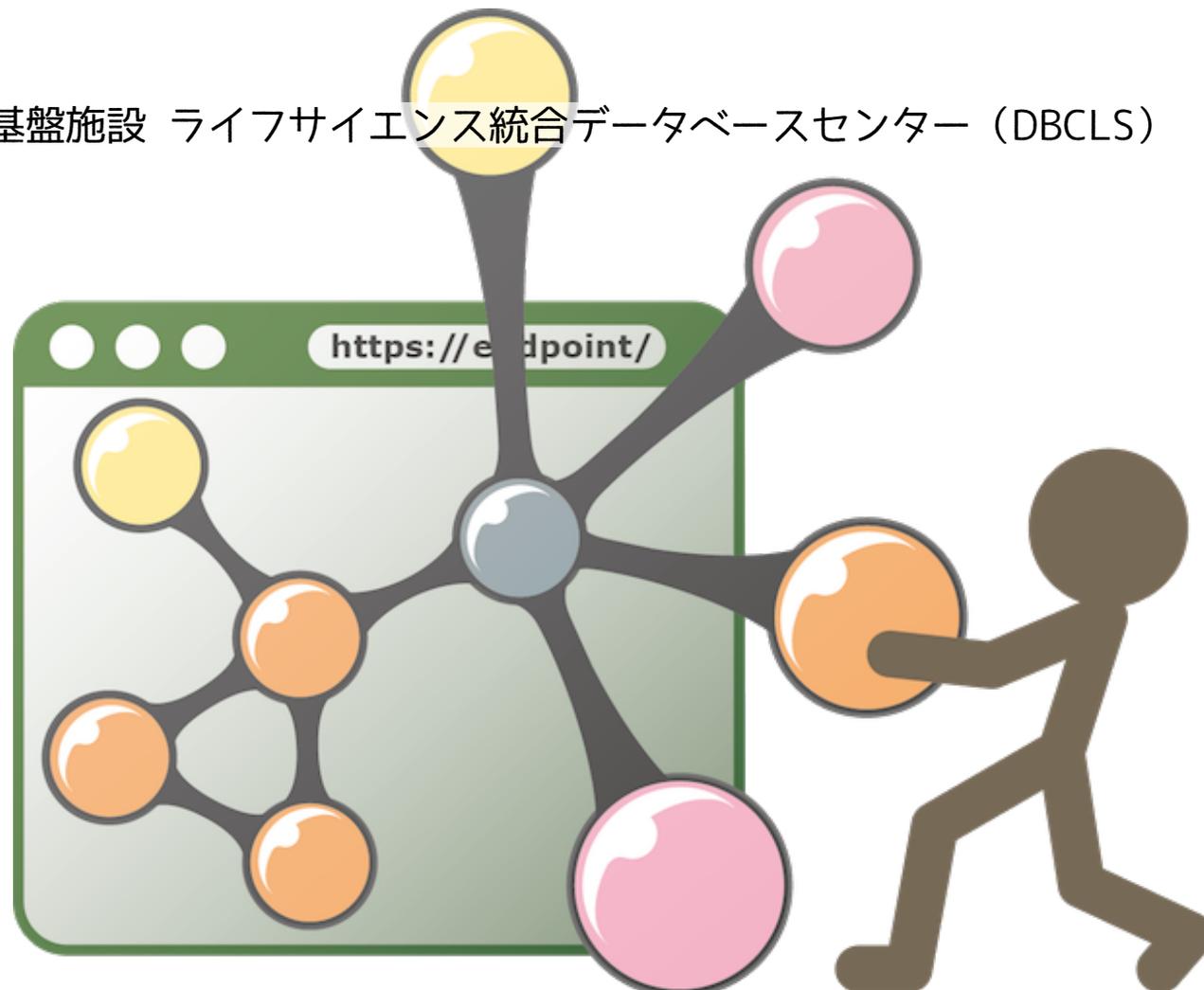


3 Endpoint browser

トーゴの日シンポジウム2020

守屋 勇樹

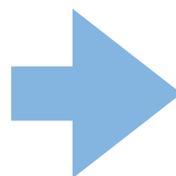
情報・システム研究機構 データサイエンス共同利用基盤施設 ライフサイエンス統合データベースセンター (DBCLS)



概要

- 様々なデータがRDF化されてきている
- けど、SPARQLを書くのは大変

SPARQLクエリの例



```
## endpoint https://www.genome.jp/oc/proxy/sparql
BASE <http://rdf.genome.jp/keggoc/2019-02-28/>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX orth: <http://purl.org/net/orth#>
PREFIX void: <http://rdfs.org/ns/void#>

SELECT ?size (COUNT (?size) AS ?num)
WHERE {
  {
    SELECT ?oc (COUNT (?genes) AS ?size)
    WHERE {
      ?oc void:inDataset/rdfs:label "Cellular_organisms" ;
      orth:hasHomologous+ ?genes .
      ?genes a orth:Gene .
    }
  }
}
ORDER BY DESC (?num)
```

- URIを打ち込むのも面倒
- RDFやクエリによって使うURIは様々なのでコピーすらも面倒
- 使われてる述語も覚えてない

概要

- そこで SPARQL support (<https://sparql-support.dbcls.jp/>) を開発

```
1 ## endpoint https://www.genome.jp/oc/proxy/sparql
2 BASE <http://rdf.genome.jp/keggoc/2019-02-28/>
3 PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
4 PREFIX orth: <http://purl.org/net/orth#>
5 PREFIX void: <http://rdfs.org/ns/void#>
6
7 SELECT ?size (COUNT (?size) AS ?num)
8 WHERE {
9   {
10    SELECT ?oc (COUNT (?genes) AS ?size)
11    WHERE {
12      ?oc void:inDataset/rdfs:label "Cellular_organisms" ;
13         orth:hasHomologous+ ?genes .
14      ?genes a orth:Gene .
15    }
16   }
17 }
18 ORDER BY DESC (?num)
```

size	num
1	1291446
2	240201
3	105026
4	73087
5	48982

- 様々な自動補完
- クエリ管理
- 検索時間表示
- 述語検索
- 複数行コメントアウト

などの機能を持った
SPARQLクライアント

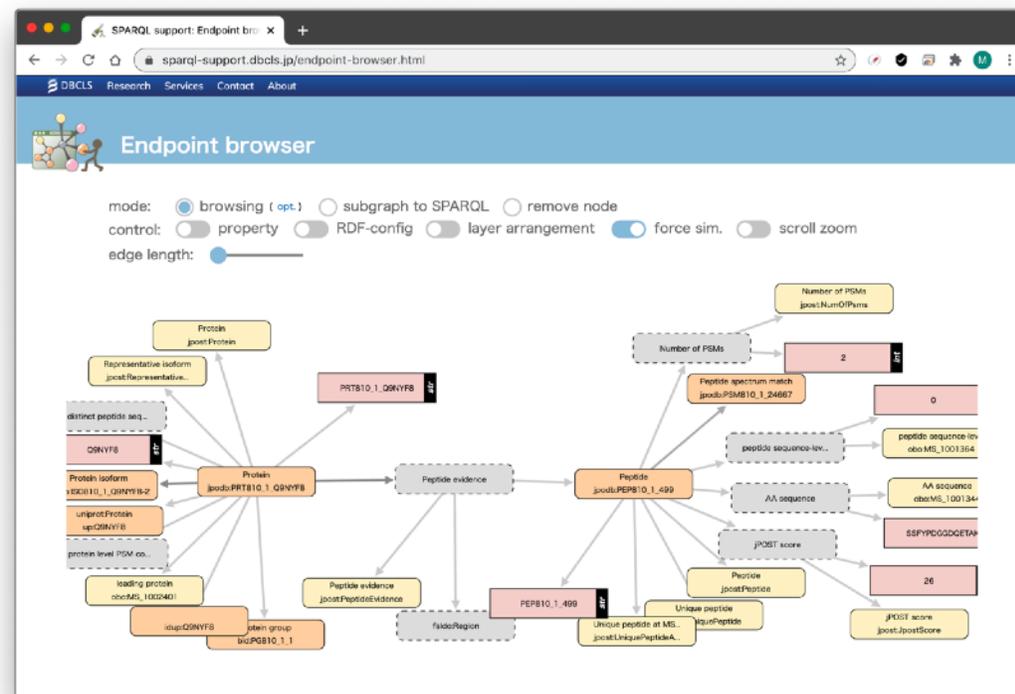
詳細は[こちら](#)

概要

- でもSPARQLクエリを書くにはデータ構造を考えないといけないが
 - RDFの構造が複雑
 - そもそも構造を知らない、覚えてない
 - 構造を記述したSchema図が提供されていない
 - ver. upでRDFの構造が変わった、オントロジーが変わった……

そこで Endpoint browser を開発

(<https://sparql-support.dbcls.jp/endpoint-browser.html>)



Endpoint browserでできること

- エンドポイントに入ったRDFデータで、
 - 段階的にネットワークグラフ状に可視化
 - クリック操作だけでRDFデータの探索
 - クリック操作で簡単なSPARQLクエリを実行
 - エンドポイントを跨いだ検索
 - RDF-configの生成



- 用意するものは、
 - エンドポイントのURL
 - 最初のノードのURI (またはリテラル)

Endpoint browser

endpoint

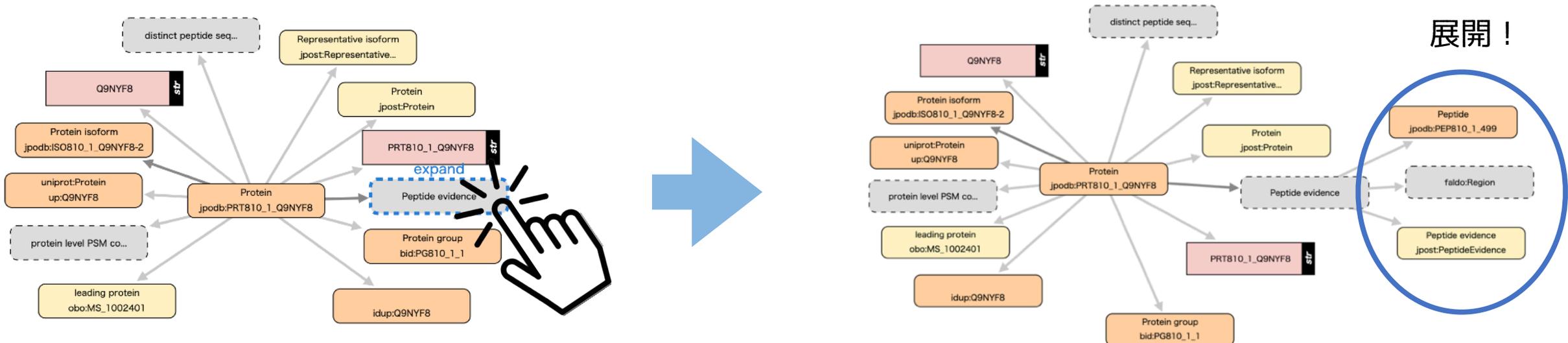
start node

* 開始画面

Endpointの探索

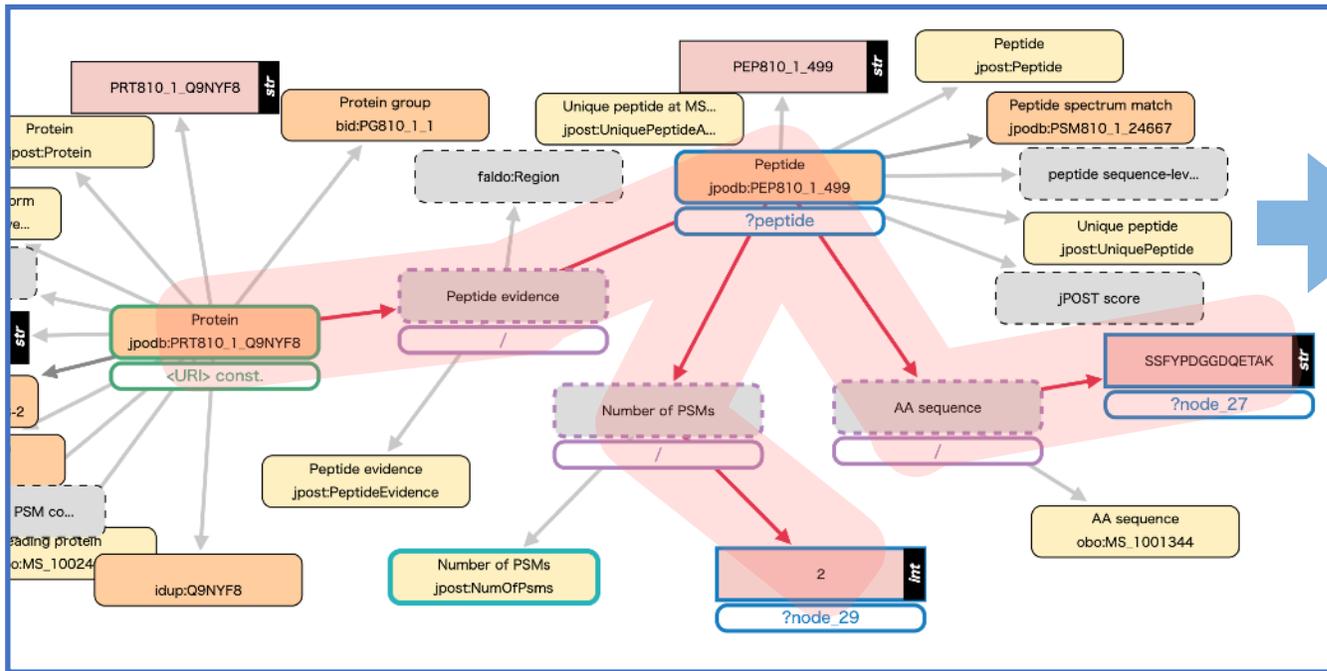


- 最初のノードを主語とする、複数のトリプルから構築されたネットワークグラフが表示
- ノードをクリックしていくことで、ネットワークが展開



ノードは class instance literal type blank で色が分かれています

SPARQLクエリの発行と実行



```
# @endpoint https://db-dev.jpostdb.org/proxy/sparql
PREFIX sio: <http://semanticscience.org/resource/>
PREFIX jpost: <http://rdf.jpostdb.org/ontology/jpost.owl#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX jpostdb: <http://rdf.jpostdb.org/entry/>
SELECT DISTINCT ?peptide ?node_29 ?node_27
WHERE {
  jpostdb:PRT810_1_Q9NYF8 jpost:hasPeptideEvidence/jpost:hasPeptide ?peptide .
  ?peptide jpost:hasSequence/rdf:value ?node_27 .
  ?peptide sio:SIO_000216/sio:SIO_000300 ?node_29 .
}
LIMIT 100
```

run

The screenshot shows the SPARQL support web interface. The query is displayed in a text area, and the results are shown in a table below. The table has three columns: 'peptide', 'node_29', and 'node_27'. The results are as follows:

peptide	node_29	node_27
jpostdb:PEP810_1_499	26	"SSFYDGGDQETAK"
jpostdb:PEP810_1_499	0	"SSFYDGGDQETAK"
jpostdb:PEP810_1_499	2	"SSFYDGGDQETAK"
jpostdb:PEP810_1_3846	23	"EEEWDPETPK"
jpostdb:PEP810_1_3846	0	"EEEWDPETPK"

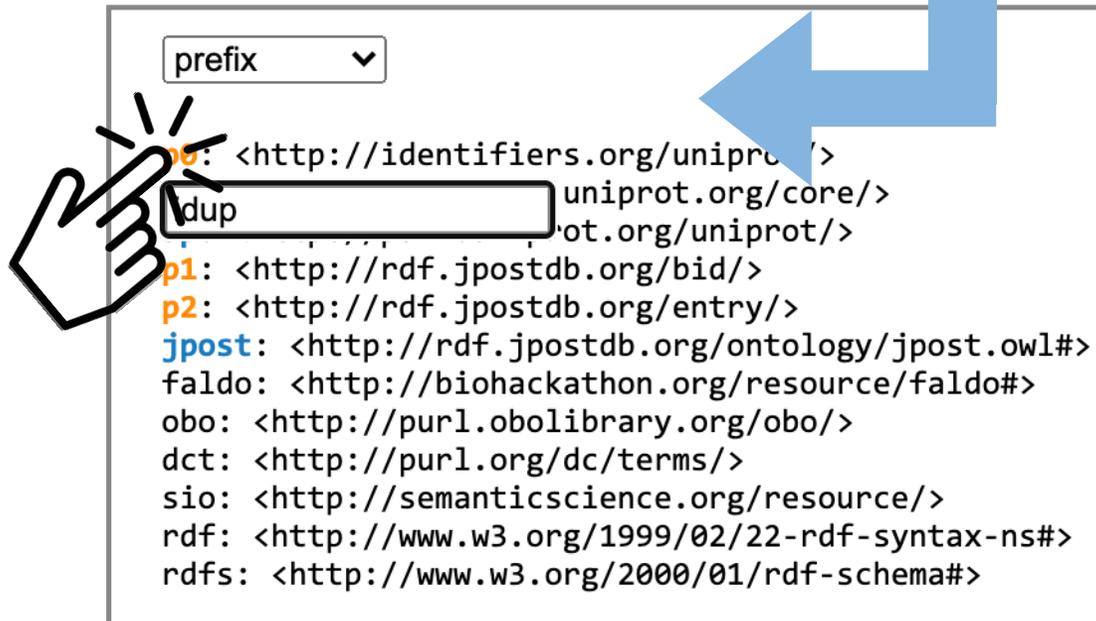
- ノードをクリックして部分グラフを選択するだけでSPARQLクエリが構築
- そのままSPARQL supportで実行

クリックしていくと変数、定数、空白ノードなどの型を選択できます

prefixや変数名の自動提案と変更

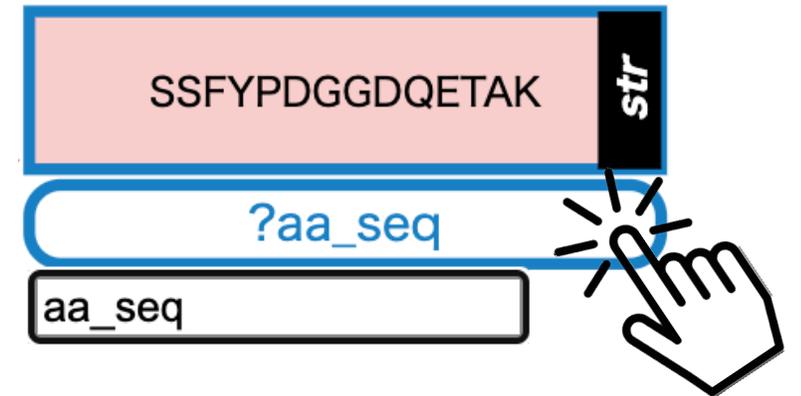
- prefixや変数名は、ある程度自動で提案
- 自動でつけられたものは分かりにくい場合もあるので変更

mode: browsing (opt.) subgraph to SPARQL
control: property RDF-config laye
edge length:



prefix ▼

- id: <http://identifiers.org/uniprot/>
- dup: <http://uniprot.org/core/>
- ot: <http://uniprot.org/uniprot/>
- p1: <http://rdf.jpostdb.org/bid/>
- p2: <http://rdf.jpostdb.org/entry/>
- jpost: <http://rdf.jpostdb.org/ontology/jpost.owl#>
- faldo: <http://biohackathon.org/resource/faldo#>
- obo: <http://purl.obolibrary.org/obo/>
- dct: <http://purl.org/dc/terms/>
- sio: <http://semanticscience.org/resource/>
- rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
- rdfs: <http://www.w3.org/2000/01/rdf-schema#>



SSFYPDGGDQETAK str

?aa_seq

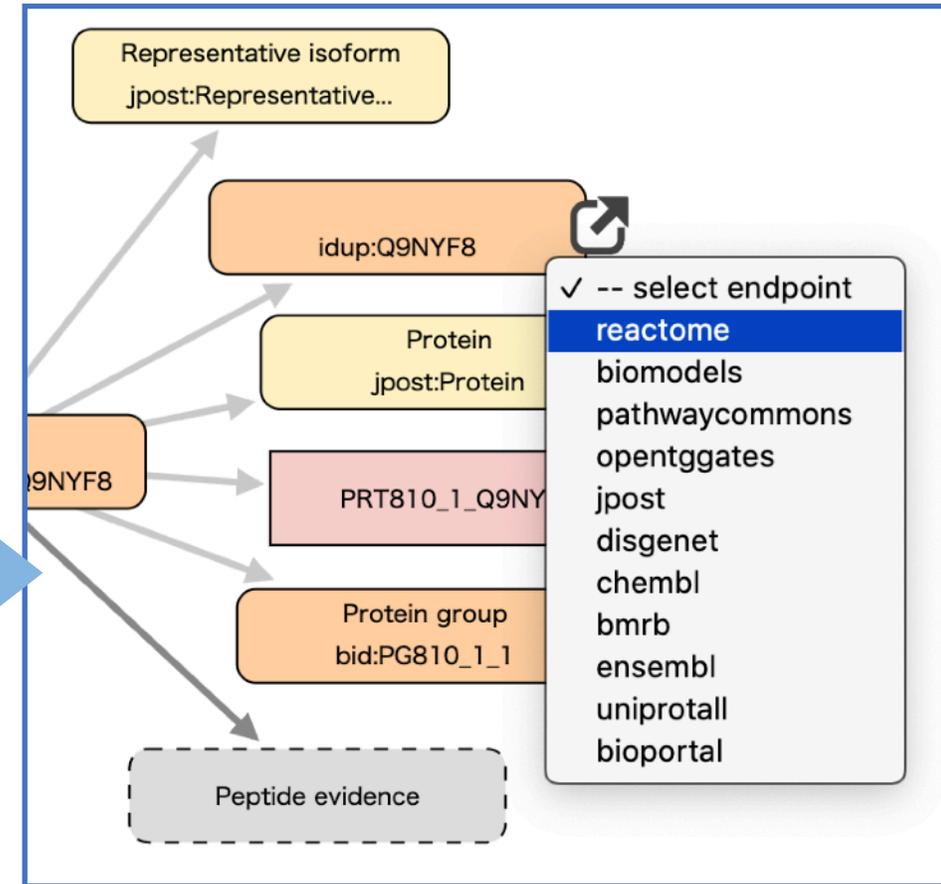
aa_seq

エンドポイントを跨いだ検索

mode: browsing (opt.) subgraph to SPARQL remove node

inverse link federated search endpoint: <https://anpther.endpoint/>

- 他のエンドポイントに切り替えてネットワークグラフを展開することで、エンドポイントを跨いだ検索
- 色々なエンドポイントで使われている identifiers.org や purl の URI ノードからはエンドポイントの候補リストから検索可能

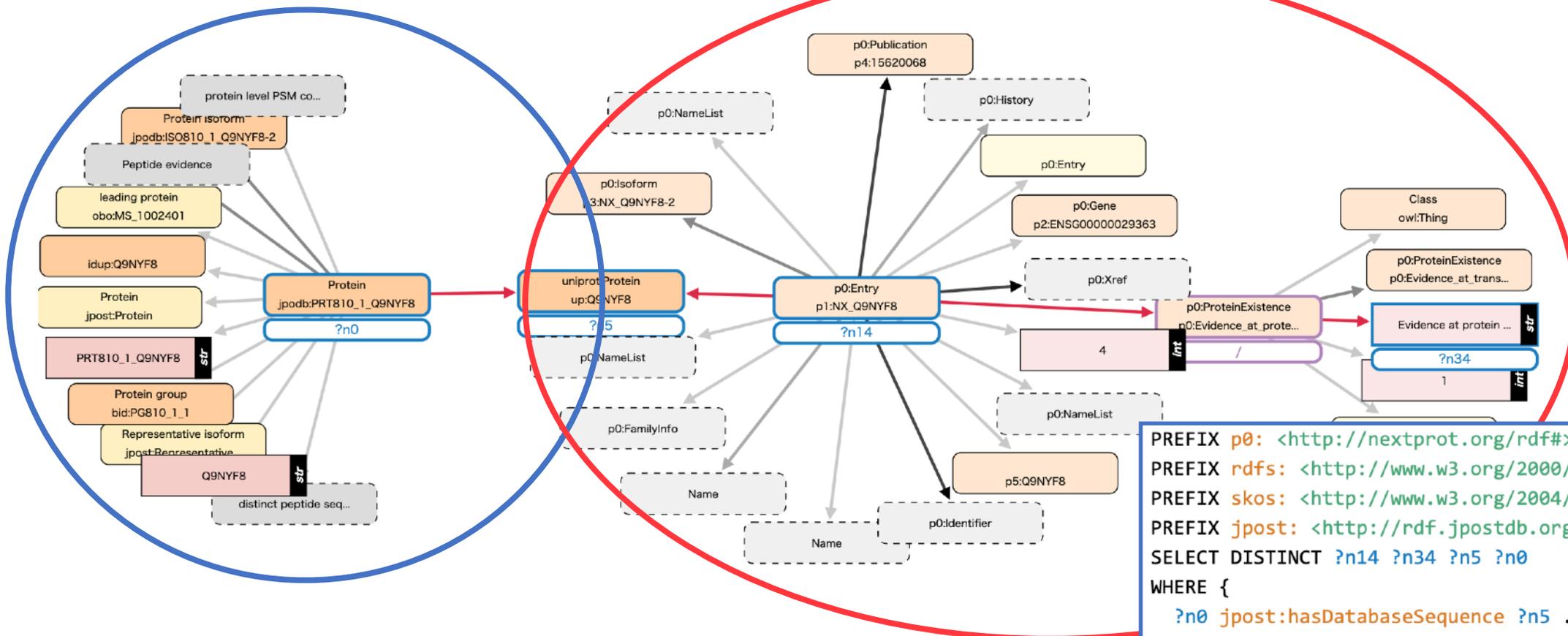


Federatedクエリの発行

フォーカスしているエンドポイントのノード以外は色が薄くなります

Endpoint 1

Endpoint 2



- エンドポイントを跨いでSPARQLクエリを発行

```
PREFIX p0: <http://nextprot.org/rdf#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX skos: <http://www.w3.org/2004/02/skos/core#>
PREFIX jpost: <http://rdf.jpostdb.org/ontology/jpost.owl#>
SELECT DISTINCT ?n14 ?n34 ?n5 ?n0
WHERE {
  ?n0 jpost:hasDatabaseSequence ?n5 .
  SERVICE <https://sparql.nextprot.org/> {
    ?n5 ^skos:exactMatch ?n14 .
    ?n14 p0:existence/rdfs:label ?n34 .
  }
}
```

RDF-configの生成

mode: browsing (opt.) subgraph to
control: property RDF-config
edge length:

model

- **Protein** **jpost:PRT810_1_Q9NYF8:**
 - a:
 - **jpost:Protein**
 - **obo:MS_1002401 # leading protein**
 - **jpost:RepresentativeIsoform**
 - **dct:identifier:**
 - **protein_id:** "PRT810_1_Q9NYF8"
 - **jpost:hasDatabaseSequence:**
 - **uniprot: up:Q9NYF8**
 - **jpost:hasIsoform+:**
 - **protein_isoform:** **jpost:IS0810_1_Q9NYF8-2**
 - **jpost:hasPeptideEvidence+:**
 - []:
 - a: **jpost:PeptideEvidence**
 - **faldo:location:**
 - []:
 - a: **faldo:Region**
 - **faldo:begin:**
 - []:
 - a: **faldo:ExactPosition**
 - **faldo:position:**

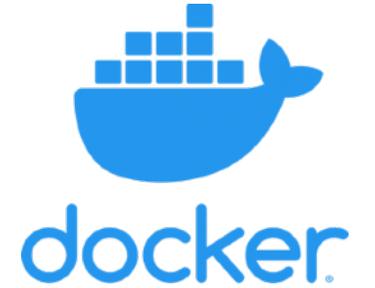
- RDF-configのyamlファイルを生成
 - ラベルから変数名の自動提案
 - prefix、変数名、出現回数の制約などを簡単に編集
 - スキーマ図などのRDF-config出力チェック機能

RDF-configについては4番のポスターを見て下さい

作成チュートリアルは[こちら](#)

Docker

- 公開されたエンドポイント向けのサービスだが、
Dockerを使ってローカルでサービスを構築することで
非公開エンドポイントでも利用可能



まとめ

- クリックで、
RDFデータを探索できる
SPARQLクエリを生成できる
RDF-configを作れる

