ContFed's Labelling approach

Algorithm 1: CostFed Hybrid (ASK+Index) algorithm for labelling all hyperedges of each disjunctive hypergraph of a SPARQL query

```
Require: $\mathcal{D}$= $\{D_1, \ldots, D_n\}$; $DHG$ = $\{HG_1, \ldots, HG_x\}$; $CostFed_{\mathcal{D}}$ //sources, disjunctive
    hypergraphs of a query, CostFed summaries of sources
 1: for each $HG_i \in DHG$ do
 2:   E = hyperedges ($HG_i$)
 3:   for each $e_i \in E$ do
 4:     $s$ = subjvertex($e_i$); $p$ = predvertex($e_i$); $o$ = objvertex($e_i$);
 5:     $sp$ = sbjPrefix($s$); $op$ = objPrefix($o$); //can be null i.e. for unbound s, o
 6:     if !bound($s$) $\wedge$ !bound($p$) $\wedge$ !bound($o$) then
 7:       $\lambda_e(e_i)$ = $\mathcal{D}$
 8:     else if bound($p$) then
 9:       if $p = rdf : type$ $\wedge$ bound($o$) then
10:         $\lambda_e(e_i)$ = $CostFed_{\mathcal{D}}lookup$($p$, $o$)
11:       else if !commonpredicate($p$) $\vee$ (!bound($s$) $\wedge$ !bound($o$)) then
12:         $\lambda_e(e_i)$ = $CostFed_{\mathcal{D}}lookup$($sp$, $p$, $op$)
13:       else
14:         if cachehit($s$, $p$, $o$) then
15:           $\lambda_e(e_i)$ = $cachelookup$($s$, $p$, $o$)
16:         else
17:           $\lambda_e(e_i)$ = ASK($s$, $p$, $o$, $D$)
18:         end if
19:       end if
20:     else
21:       Call Lines 14-18
22:     end if
23:   end for
24: end for
25: return $DHG$ //Set of labelled disjunctive hypergraphs
```

The pseudo-code of labelling all the hyperedges of each disjunctive hypergraph of a SPARQL query is shown in Algorithm 1. It takes the set of all sources $\mathcal{D}$, the set of all disjunctive hypergraphs DHG of the input query Q and the data summaries $CostFed_{\mathcal{D}}$ of all sources in $\mathcal{D}$ as input. It returns a set of *labelled* disjunctive hypergraphs as output. For each hypergraph and each hyperedge, the subject, predicate, object, subject prefix, and object prefix are collected (Lines 2-5 of Algorithm 1). Edges with unbound subject, predicate, and object vertices (e.g e = (?s, ?p, ?o)) are labelled with the set of all possible sources $\mathcal{D}$ (Lines 6-7 of Algorithm 1). A data summary lookup is performed for edges with the predicate vertex `rdf:type` that have a bound object vertex. All sources with matching capabilities are selected as label of the hyperedge (Lines 9-10 of Algorithm 1).

Our algorithm makes use of the notion of *common predicates*. A common predicate is a predicate that is used in a number of sources above a specific threshold value θ specified by the user. A predicate is then considered a common predicate if it occurs in at least $\theta|\mathcal{D}|$ sources. We make use of the ASK queries for triple patterns with common predicates. Here, an ASK query is sent to all of the available sources to check whether they contain the common predicate *cp*. Those sources which return `true` are selected as elements of the set of sources used to label that triple pattern. The results of the ASK operations are stored in a cache. Therefore, every time we perform a cache lookup before SPARQL ASK operations (Lines 14-18).