May 28, 2021

This document is part of the paper "RDF^M: An Alternative Approach for representing and maintaining meta-knowledge in Web of Data". It presents the matching algorithm which is used in the paper.

1 Algorithm

Algorithm Step 3.2: matching_function

Input: line send by Algorithm 1.
Output: Result send back to Algorithm 1.

```
1: procedure EXECUTE line containing user_ query
2:     for each line do
3:         if subject is variable, object is variable then.
4:             if subject variable is new then
5:                 Append the contents of subject-dict of result dict with that of subject-dict of
   current dict. And Add variable to list in vars['subject']
6:             end if
7:             if subjectis variable and is previously encountered then
8:                 Find the position of previous encounter and select the associated dictionary.
9:                 for each key in associated-dict do
10:                    if key exists in current dict's subject-dict then
11:                        Make union of current dict's subject-dict[key] and result dict's associated-
   dict[key].
12:                        Store it as value of associated-dict[key].
13:                    end if
14:                    if key not found in subject-dict of current dict then
15:                        Delete the key from associated-dict of result dict.
16:                    end if
17:                end for
18:            end if
19:            if object variable is new then
20:                Append the contents of object-dict of result dict with that of object-dict of current
   dict.
21:                Add variable to list in vars['object']
22:            end if
23:            if object is variable and is previously encountered then
24:                Find the position of previous encounter and select the associated dictionary.
25:                for each key in associated-dict do
26:                    if key exists in current dict's object-dict then
27:                        Make union of current dict's object-dict[key] and result dict's associated-
   dict[key].
28:                        Store it as value of associated-dict[key].
29:                    end if
30:                    if key not found in object-dict of current dict then
31:                        Delete the key from associated-dict of result dict.
32:                    end if
33:                end for
34:            end if
35:        end if
36:        if subject is given, object is variable then
37:            Keep the matching given subjectin result dict's subject-dict.
38:            for each non-matching key in subject-dict do
39:                Remove subject from object-dict of result dict.
40:                Remove object from uid-dict of result dict.
41:            end for
42:            Remove the non-matching keys.
43:            if object variable is new then
44:                Append the contents of object-dict of result dict with that of object-dict of current
   dict and add variable to list in vars['object']
45:            end if
46:            if object is variable and is previously encountered then
47:                Find the position of previous encounter and select the associated dictionary.
48:                for each key in associated-dict do
49:                    if key exists in current dict's object-dict then
50:                        Make union of current dict's object-dict[key] and result dict's associated-
   dict[key].
51:                        Store it as value of associated-dict[key].
52:                    end if
53:                    if key not found in object-dict of current dict then
54:                        Delete the key from associated-dict of result dict.
55:                    end if
56:                end for
57:            end if
58:        end if
```

```
59:         if subject is variable, object is given then
60:             Keep the matching given object in result dict's subject-dict.
61:             for each non-matching key in object-dict do
62:                 Remove subject from subject-dict of result dict.
63:                 Remove object from uid-dict of result dict.
64:             end for
65:             Remove the non-matching object keys.
66:             if subject variable is new then
67:                 Append the contents of subject-dict of result dict with that of subject-dict of
    current dict.
68:                 Add variable to list in vars['subject']
69:             end if
70:             if subjectis variable and is previously encountered then
71:                 Find the position of previous encounter.
72:                 Previously at subject, object or uid position, select the associated dictionary.
73:                 for each key in associated-dict do
74:                     if key exists in current dict's subject-dict then
75:                         Make union of current dict's subject-dict[key] and result dict's associated-
    dict[key].
76:                         Store it as value of associated-dict[key].
77:                     end if
78:                     if key not found in subject-dict of current dict then
79:                         Delete the key from associated-dict of result dict.
80:                     end if
81:                 end for
82:             end if
83:         end if
84:         if subject is given, object is given then
85:             Keep the matching given object in result dict's subject-dict.
86:             for each non-matching key in object-dict do
87:                 Remove subject from subject-dict of result dict.
88:                 Remove object from uid-dict of result dict.
89:             end for
90:             Remove the non-matching keys.
91:             Keep the matching given subjectin result dict's subject-dict.
92:             for each non-matching key in subject-dict do
93:                 Remove subject from object-dict of result dict.
94:                 Remove object from uid-dict of result dict.
95:             end for
96:             Remove the non-matching keys.
97:         end if
98:         Start UID1 processing.
99:         if UID is variable in user query then check:
100:             if UID variable is new then
101:                 Append the contents of uid-dict of result dict with that of uid-dict of current
    dict.
102:                 Add variable to list in vars['uid']
103:             end if
104:             if UID is variable and is previously encountered then
105:                 Find the position of previous encounter.
106:                 Previously at sub, object or uid position, select the associated dictionary.
107:                 for each key in associated-dict do
108:                     if key exists in current dict's uid-dict then
109:                         Make union of current dict's uid-dict[key] and result dict's associated-
    dict[key].
110:                         Store it as value of associated-dict[key].
111:                     end if
112:                     if key not found in uid-dict of current dict then
113:                         Delete the key from associated-dict of result dict.
114:                     end if
115:                 end for
116:             end if
117:         end if
```

```
118:          if UID1 is given then
119:              Keep the matching given UID in result dict's uid-dict.
120:              for each non-matching key in uid-dict do
121:                  Remove subject from subject-dict of result dict.
122:                  Remove object from object-dict of result dict.
123:              end for
124:              Remove the non-matching keys.
125:          end if
126:          Predicate parameters processing.
127:          for each parameter do
128:              if parameter if variable then
129:                  Check if parameter is new OR Check if parameter matches with previous parameter variables and match
130:              end if
131:              if parameter is given then
132:                  Match given parameter accordingly with result dict.
133:              end if
134:              if parameter is not required then
135:                  No processing required.
136:              end if
137:          end for
138:          Start UID2 processing.
139:          if UID2 is variable then
140:              Check if UID2 is new OR
141:              Check if UID2 matches with previous variables and match accordingly
142:          end if
143:          if UID2 is given then
144:              Match given UID2 with result dict
145:          end if
146:      end for
147:  end procedure
```