Graphs and knowledge graphs

Dave Dubin

February, 2019

Scheduling Problem

Adapted from Aldous and Wilson (2000).

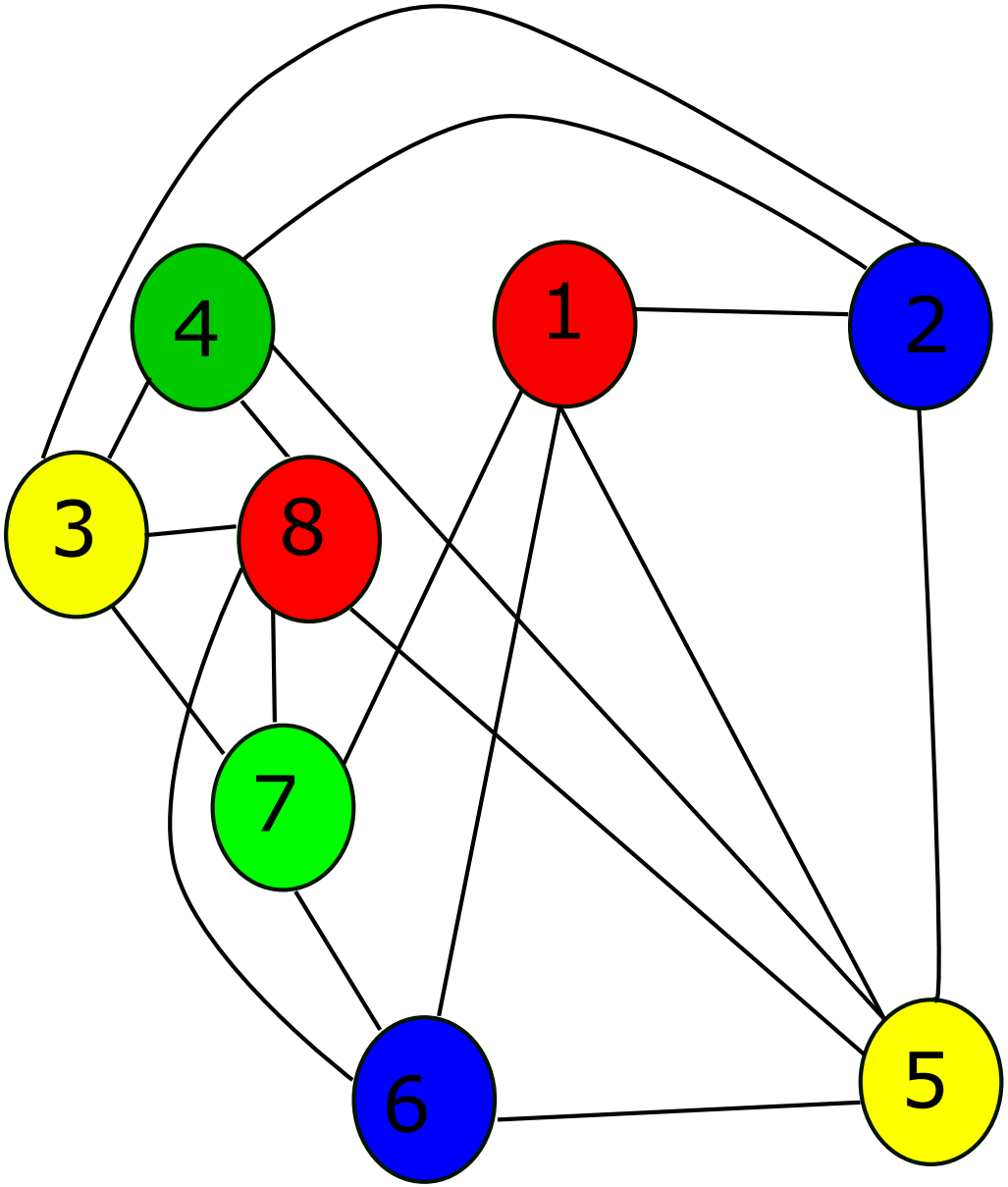
Students assigned to project groups need to meet together. We wish to book the smallest number of meeting rooms and times that we can, which means scheduling meetings at the same time if possible.

If the students are identified by letter, and the meetings by number, what's the smallest number of rooms and times that we can book while ensuring that no student will miss a meeting? Produce a timetable of meeting times and places.

Scheduling table

	1	2	3	4	5	6	7	8
A	X	X			X			
B	X				X	X		
C	X					X	X	
D						X	X	X
E			X				X	X
F			X	X				X
G		X	X	X				
H		X		X	X			
I				X	X			X
J					X	X		X

Graph coloring



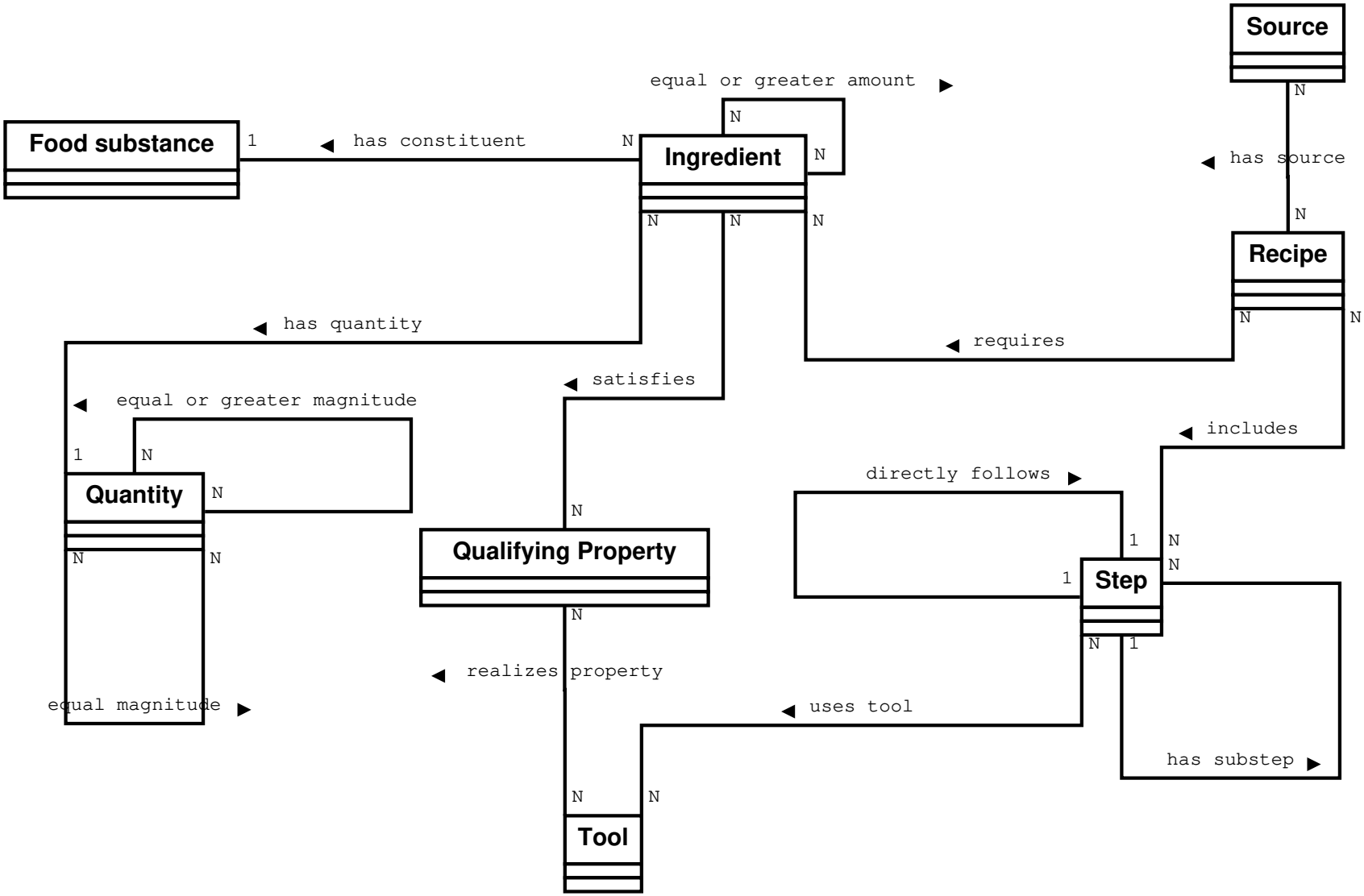
Formalization from this week's reading

Rahman's formalization:

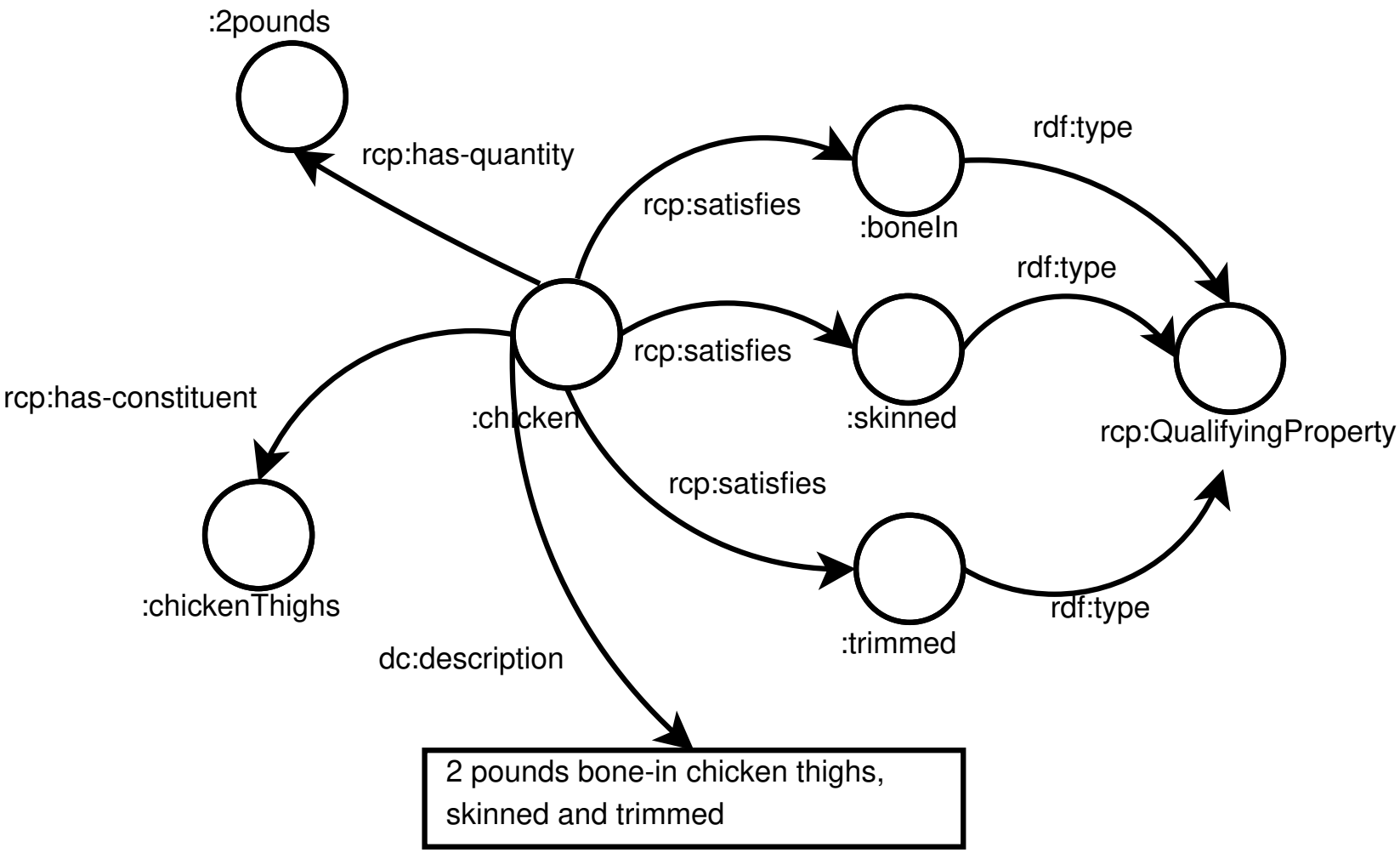
A *graph* G is a tuple (V, E) which consists of a finite set V of *vertices* and a finite set E of *edges* ; each edge is an unordered pair of vertices. The two vertices associated with an edge e are called the *end-vertices* of e . We often denote by (u, v), an edge between two vertices u and v. We also denote the set of vertices of a graph G by $V(G)$ and the set of edges of G by $E(G)$. Let $e = (u, v)$ be an edge of a graph G. Then the two vertices u and v are said to be *adjacent* in G and the edge e is said to be *incident* to the vertices u and v. The vertex u is also called a *neighbor* of v in G and vice versa.

The number of edges incident to a vertext is the *degree* of that vertex (where the edge is counted twice if both ends connect to the same vertex). In a *directed graph* the edges are ordered pairs.

UML Class Diagram



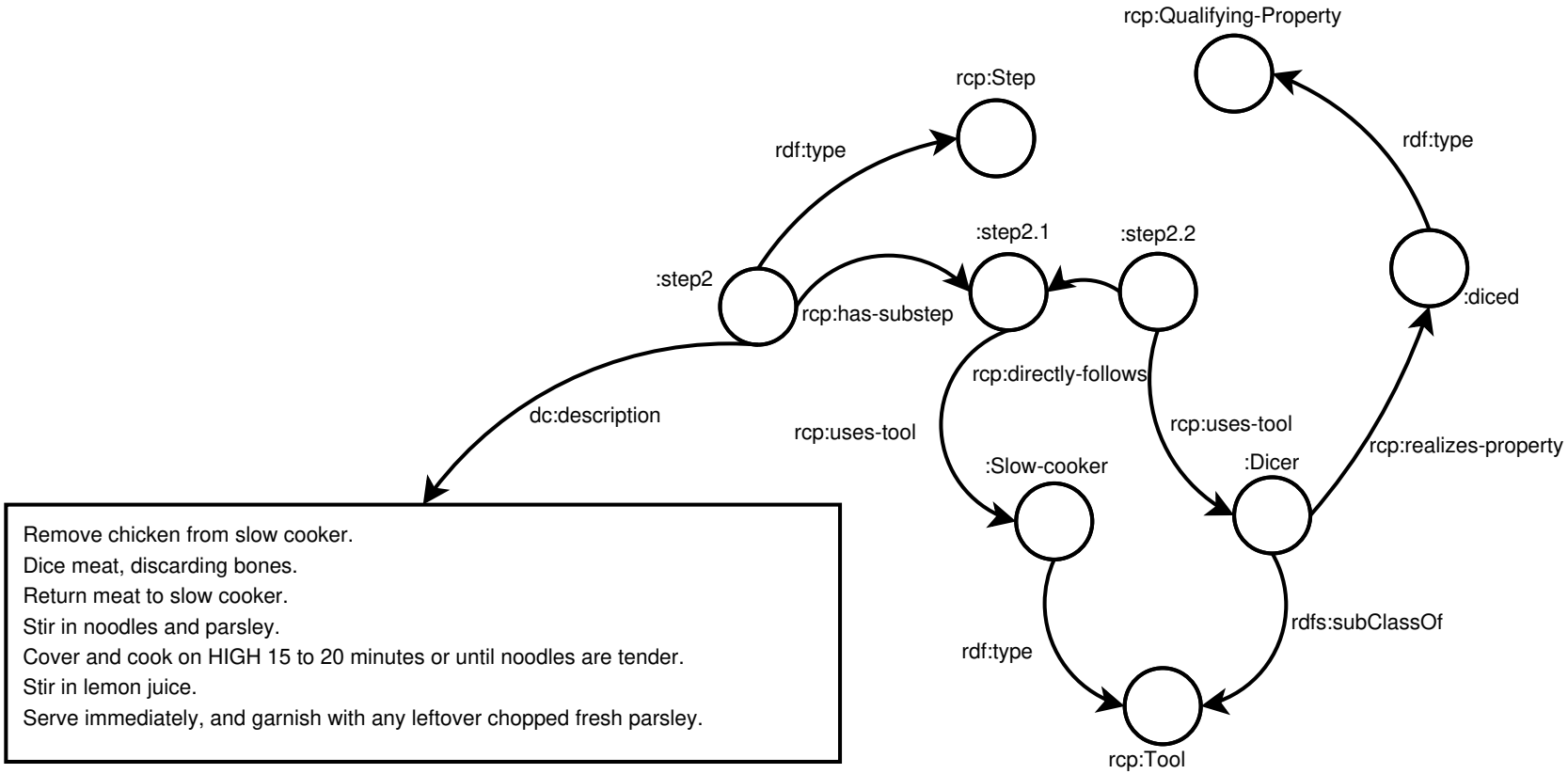
Recipe ingredient



Digression: inference and missing links

How many missing edges on the previous diagram could be deduced using the information in my list of entity classes and relationships?

Recipe substeps and tools



RDF's graph model

- A statement in RDF consists of three parts: a *subject*, a *predicate*, and an *object*.

RDF's graph model

- A statement in RDF consists of three parts: a *subject*, a *predicate*, and an *object*.
  - The subject: what's being talked about;

RDF's graph model

- A statement in RDF consists of three parts: a *subject*, a *predicate*, and an *object*.
  - The subject: what's being talked about;
  - The predicate: a property it has or a binary relationship it stands in;

RDF's graph model

- A statement in RDF consists of three parts: a *subject*, a *predicate*, and an *object*.
  - The subject: what's being talked about;
  - The predicate: a property it has or a binary relationship it stands in;
  - The object: the value of the property, or the other participant in the relationship.

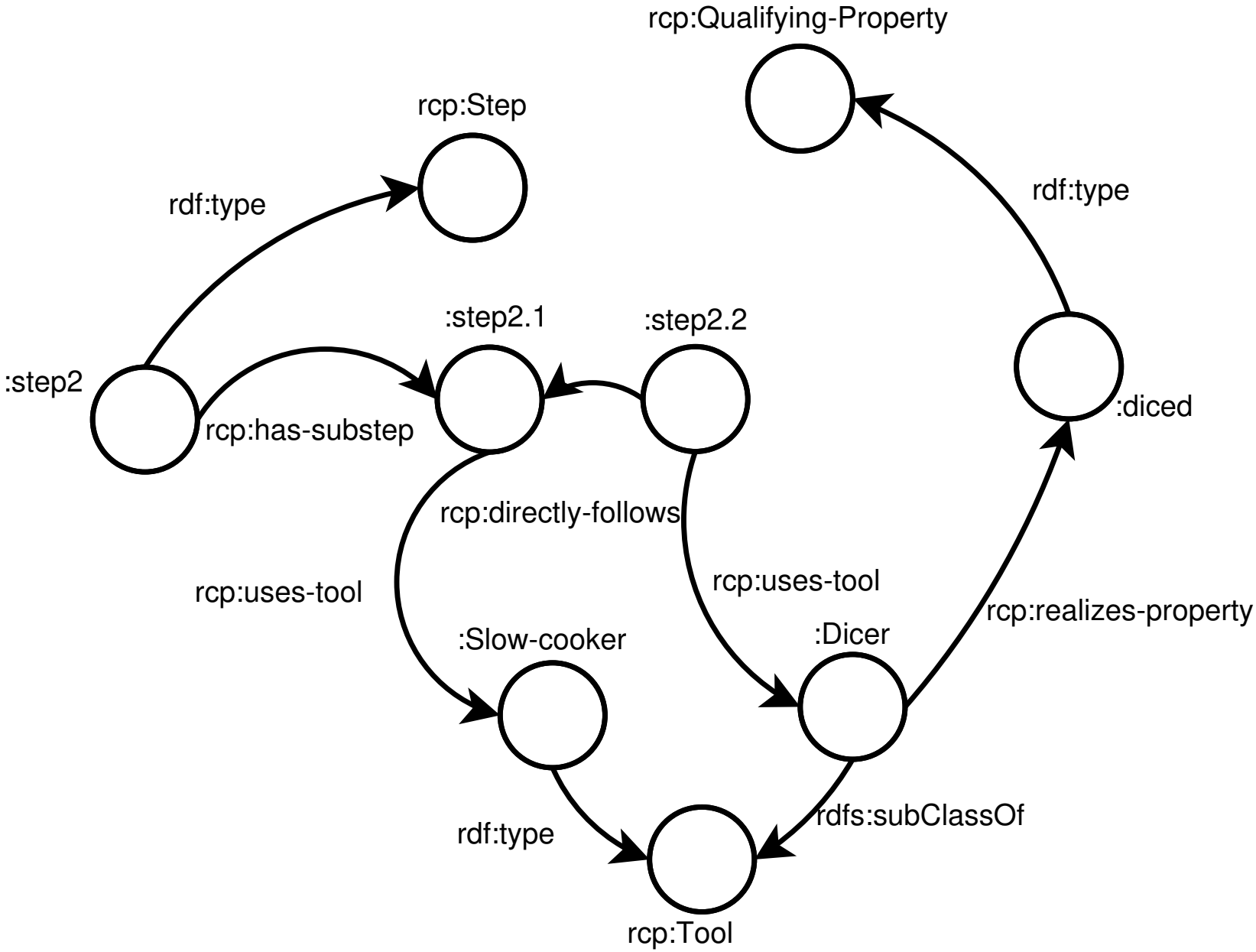
RDF's graph model

- A statement in RDF consists of three parts: a *subject*, a *predicate*, and an *object*.
  - The subject: what's being talked about;
  - The predicate: a property it has or a binary relationship it stands in;
  - The object: the value of the property, or the other participant in the relationship.
- In RDF, all properties are *reduced* to binary relationships.

RDF's graph model

- A statement in RDF consists of three parts: a *subject*, a *predicate*, and an *object*.
  - The subject: what's being talked about;
  - The predicate: a property it has or a binary relationship it stands in;
  - The object: the value of the property, or the other participant in the relationship.
- In RDF, all properties are *reduced* to binary relationships.
- So just as suggested by the diagram, subjects and objects can be understood as vertices in a directed graph, where relationships serve as the graph edges.

Recipe substeps and tools



Choices for modeling cooking tools

1. *Tools* class instances are every individual physical cooking tool.

Choices for modeling cooking tools

1. *Tools* class instances are every individual physical cooking tool.
   - $S \subseteq T$ and $K \subseteq D \subseteq T$

Choices for modeling cooking tools

1. *Tools* class instances are every individual physical cooking tool.
   - $S \subseteq T$ and $K \subseteq D \subseteq T$
   - $s \in T$ and $k \in K$

Choices for modeling cooking tools

1. *Tools* class instances are every individual physical cooking tool.
   - $S \subseteq T$ and $K \subseteq D \subseteq T$
   - $s \in T$ and $k \in K$
   - So $k \in D$ and $k \in T$

Choices for modeling cooking tools

1. *Tools* class instances are every individual physical cooking tool.
   - $S \subseteq T$ and $K \subseteq D \subseteq T$
   - $s \in T$ and $k \in K$
   - So $k \in D$ and $k \in T$
2. *Tools* class instances are classes of specific tools, such as knives and slow cookers.

Choices for modeling cooking tools

1. *Tools* class instances are every individual physical cooking tool.
   - $S \subseteq T$ and $K \subseteq D \subseteq T$
   - $s \in T$ and $k \in K$
   - So $k \in D$ and $k \in T$
2. *Tools* class instances are classes of specific tools, such as knives and slow cookers.
   - $S \in T$ and $K \in T$

Choices for modeling cooking tools

1. *Tools* class instances are every individual physical cooking tool.
   - $S \subseteq T$ and $K \subseteq D \subseteq T$
   - $s \in T$ and $k \in K$
   - So $k \in D$ and $k \in T$
2. *Tools* class instances are classes of specific tools, such as knives and slow cookers.
   - $S \in T$ and $K \in T$
   - $K \in D$ and $D \subseteq T$

Choices for modeling cooking tools

1. *Tools* class instances are every individual physical cooking tool.
   - $S \subseteq T$ and $K \subseteq D \subseteq T$
   - $s \in T$ and $k \in K$
   - So $k \in D$ and $k \in T$
2. *Tools* class instances are classes of specific tools, such as knives and slow cookers.
   - $S \in T$ and $K \in T$
   - $K \in D$ and $D \subseteq T$

- Considerations:

Choices for modeling cooking tools

1. *Tools* class instances are every individual physical cooking tool.
   - $S \subseteq T$ and $K \subseteq D \subseteq T$
   - $s \in T$ and $k \in K$
   - So $k \in D$ and $k \in T$
2. *Tools* class instances are classes of specific tools, such as knives and slow cookers.
   - $S \in T$ and $K \in T$
   - $K \in D$ and $D \subseteq T$

- Considerations:
  - What I know about tools and how I want my system to answer questions;

Choices for modeling cooking tools

1. *Tools* class instances are every individual physical cooking tool.
   - $S \subseteq T$ and $K \subseteq D \subseteq T$
   - $s \in T$ and $k \in K$
   - So $k \in D$ and $k \in T$
2. *Tools* class instances are classes of specific tools, such as knives and slow cookers.
   - $S \in T$ and $K \in T$
   - $K \in D$ and $D \subseteq T$

- Considerations:
  - What I know about tools and how I want my system to answer questions;
  - Expressive limitations of my modeling language.