Crowd-based Ontology Engineering with the uComp Protégé Plugin

Editor(s): Krzysztof Janowicz, STKO Lab, University of California, Santa Barbara, USA; Stefan Schlobach, Vrije Universiteit Amsterdam, The Netherlands
Solicited review(s): Benjamin Adams, The University of Auckland, New Zealand; three anonymous reviewers

Gerhard Wohlgenannt [a,*], Marta Sabou [b,c] and Florian Hanika [a]
[a] *Vienna University of Economics and Business (WU), Welthandelsplatz 1, 1020 Vienna, Austria*
E-mail: gerhard.wohlgenannt@wu.ac.at, florian.hanika@gmail.com
[b] *Vienna University of Technology (TUW), Inst. of Software Technology, CDL-Flex, Favoritenstrasse 9-11/188, 1040 Vienna, Austria*
E-mail: marta.sabou@ifs.tuwien.ac.at
[c] *MODUL University Vienna, Am Kahlneberg 1, 1190 Vienna, Austria*

Abstract. Crowdsourcing techniques provide effective means for solving a variety of ontology engineering problems. Yet, they are mainly used as external support to ontology engineering, without being closely integrated into the work of ontology engineers. In this paper we investigate how to closely integrate crowdsourcing into ontology engineering practices. Firstly, we show that a set of basic crowdsourcing tasks are used recurrently to solve a range of ontology engineering problems. Secondly, we present the *uComp Protégé plugin* that facilitates the integration of such typical crowdsourcing tasks into ontology engineering from within the Protégé ontology editor. An evaluation of the plugin in a typical ontology engineering scenario where ontologies are built from automatically learned semantic structures, shows that its use reduces the working times for the ontology engineers 11 times, lowers the overall task costs by 40% to 83% depending on the crowdsourcing settings used and leads to data quality comparable with that of tasks performed by ontology engineers. Evaluations on a large anatomy ontology confirm that crowdsourcing is a scalable and effective method: good quality results (accuracy of 89% and 99%) are obtained while achieving cost reductions of 75% from the ontology engineer costs and providing comparable overall task duration.

Keywords: crowdsourcing, ontology engineering, ontology learning, Protégé plugin

1. Introduction

The advent of the Web has significantly changed the context of systems that rely on formally specified knowledge (e.g., ontologies). These systems became distributed, cater for many users with different levels of expertise and integrate knowledge sources of varying quality [11]. As a consequence, an important change was required in the knowledge acquisition methods that enable these systems by gradually extending knowledge creation processes to include groups of users with diverse levels of expertise and training [11]. This change started by opening up knowledge creation tools to wider groups of contributors. Indeed, WebProtégé [30] is an extension of the Protégé ontology editor that allows the ontology engineering process to be performed by a larger, distributed group of contributors. Similarly, in the area of natural language processing, GATE Teamware extends the GATE linguistic annotation toolkit with distributed knowledge creation capabilities [1]. While these extensions support the collaborative and distributed work of knowledge experts (ontology engineers and linguists),

[*]Corresponding author. E-mail: gerhard.wohlgenannt@wu.ac.at, Tel.no.: +43 1 31336 5228

recent approaches have tried further broadening this process by allowing large populations of *non-experts* to create knowledge through the use of crowdsourcing techniques such as games or mechanised labour platforms. Crowdsourcing methods provide effective means to solve a variety of knowledge acquisition tasks [24] by outsourcing these to “an undefined, generally large group of people in the form of an open call" [12].

A crucial knowledge acquisition process in the area of the Semantic Web is ontology engineering. Ontology engineering is the process spanning the creation and maintenance of ontologies during their entire lifecycle. Similarly to other knowledge creation tasks, ontology engineering is traditionally performed by ontology experts and therefore it tends to be a complex, costly and, above all, time-consuming process.

Let’s consider the task of ontology creation. To reduce its complexity, ontology construction is often bootstrapped by re-using existing ontologies or automatically derived ones. Ontology learning methods, for example, automatically extract ontologies from (a combination of) unstructured and structured resources. Although the automatically extracted ontologies already provide a good basis for building the ontology, they typically contain questionable or wrong ontological elements and require a phase of verification and redesign (especially pruning) by the ontology engineer. The ontology verification phase involves, among others, checking that the ontology concepts are relevant to the domain of interest and that the extracted subsumption relations are correct. As detailed in Section 2, crowdsourcing has been used effectively to solve a range of such ontology verification tasks and therefore it could be employed to support the ontology engineer in performing such tasks.

Unfortunately, crowdsourcing techniques require high upfront investments (understanding the techniques, creating appropriate tasks) and therefore, despite their proven usefulness, these techniques remain outside the reach of most ontology engineers. Therefore, in this paper we investigate how to more closely embed crowdsourcing into ontology engineering in line with the current trends of open and extended knowledge acquisition processes. In the area of Natural Language Processing (NLP), where the use of crowdsourcing is highly popular [22], there already exists an effort towards supporting easy integration of crowdsourcing methods into linguists’ work: the GATE Crowdsourcing Plugin is a component of the popular GATE NLP platform that allows inserting crowdsourcing tasks into larger NLP workflows, from within GATE’s user interface [2]. In the Semantic Web area, recent approaches, such as ZenCrowd [8] and CrowdMap [25], involve large groups of non-experts to solve specific knowledge acquisition tasks such as entity linking or ontology matching, respectively. There has been however no work in including crowdsourcing to support a broad range of ontology engineering tasks, although Noy and colleagues [19] introduce a vision for tool support to facilitate the integration of crowdsourcing into ontology engineering after experimentally proving that crowd-workers are a viable alternative for verifying subclass-superclass relations.

To achieve our goal we seek answers to the following research questions:

Which tasks can be crowdsourced? Based on an extensive literature review, we distill a set of crowdsourcing tasks that are likely to be common to solving a variety of ontology engineering problems and which should be implemented by the desired tool support (Section 2).

How to implement crowdsourcing enabled ontology engineering? We present a tool, the uComp Protégé plugin, which allows ontology engineers to crowdsource tasks directly from within the popular ontology engineering tool and as part of their ontology engineering work (Section 3).

Is crowd-based ontology engineering feasible and scalable? We conduct a variety of experimental evaluations focusing both on understanding the feasibility and the scalability of crowd-based ontology engineering. Feasibility evaluations focus on assessing the extent to which the plugin leads to improvements over manually solving a set of tasks in terms of time and cost reductions, while maintaining good data quality. Scalability evaluations assess the plugin’s effect on the time, cost and quality of the ontology engineering process for large, real-life and domain specific ontologies. The details of the evaluation setup are described in Section 4 while the results of the feasibility and scalability evaluations are detailed in Sections 5 and 6 repectively.

Our findings show that in a scenario where automatically extracted ontologies are verified and pruned, the use of the plugin significantly reduces the time spent by the ontology engineer (11 times) and leads to important cost reductions (40% to 83% depending on the crowdsourcing settings used) without a loss of quality with respect to a manual process. Experimental evaluations on a large and domain specific ontology has lead to comparable task durations and significant cost reductions (of 75%) while obtaining good quality results that are in line with results obtained by other studies [9,19].

2. Use of crowdsourcing for knowledge acquisition

Crowdsourcing methods are usually classified in three major genres depending on the motivation of the human contributors (i.e., payment vs. fun vs. altruism).

Mechanised labour (MLab) is a type of paid-for crowdsourcing, where contributors choose to carry out small tasks (or micro-tasks) and are paid a small amount of money in return. Popular crowdsourcing marketplaces include Amazon's Mechanical Turk (MTurk) and CrowdFlower (CF). MTurk allows requesters to post their micro-tasks in the form of Human Intelligence Tasks (or HITs) to a large population of micro-workers (often referred to as turkers). Games with a purpose (GWAPs) enable human contributors to carry out computation tasks as a side effect of playing online games [32]. An example from the area of computational biology is the Phylo game (phylo.cs.mcgill.ca) that disguises the problem of multiple sequence alignment as a puzzle like game thus intentionally decoupling the scientific problem from the game itself [13]. The challenges in using GWAPs in scientific context are in designing appealing games and attracting a critical mass of players. Finally, in altruistic crowdsourcing a task is carried out by a large number of volunteer contributors, such as in the case of the Galaxy Zoo (www.galaxyzoo.org) project where over 250K volunteers willing to help with scientific research classified Hubble Space Telescope galaxy images (150M galaxy classifications).

Crowdsourcing methods have been used to support several knowledge acquisition and, more specifically, ontology engineering tasks. To provide an overview of these methods we will group them along the three major stages of the Semantic Web Life-cycle as identified by Siorpaes in [27] and sum them up in Table 1.

Stage 1: Build and maintain Semantic Web vocabularies. Acquisition of terminological knowledge has been addressed through games from as early as 2006 when von Ahn built Verbosity [33], a GWAP inspired from the Taboo game (where a narrator offers hints and a guesser must guess the concept). Verbosity collects a database of common-sense facts and employs the cards metaphor to guide the types of hints that the narrator gives. For example, the Type card allows providing hints about the super-classes of the concept, by generating appropriate natural language templates to be filled by the narrator. This approach ensures that the game collects a broad range of relations, such as type, purpose, is related, is opposite, and a variety of spatial relations. Built one year after Verbosity, the Common Consensus GWAP focuses on acquiring a particular type of knowledge, namely goals [17]. Inspired from the Family Feud TV show, the game asks players to answer questions such as "What are some things you would use to watch a movie?" in order to elicit common-sense knowledge about goals. The game relies on a handful of question templates which allow acquiring different types of knowledge about goals (e.g., parent and children goals or orthogonal connections between goals). As a multi-player game, players receive points in real time for all their answers that are also given by other players. Vickrey and colleagues [31] report on three games (inspired from the Scattegories and Taboo games) that aim to collect semantically related words: Categorilla (players must supply a phrase fitting a specific category, e.g., things that fly, and starting with a given letter), Categodzilla (same as Categorilla but without the letter restriction) and Free Association, where players must type words related to a given seed word.

The OntoPronto game [27] aims to support the creation and extension of Semantic Web vocabularies. Players are presented with a Wikipedia page of an entity and they have to (1) judge whether this entity denotes a concept or an instance; and then (2) relate it to the most specific concept of the PROTON ontology, therefore extending PROTON with new classes and instances. Climate Quiz [26] is a Facebook game where players evaluate whether two concepts are related (e.g. environmental activism, activism), and which label is the most appropriate to describe their relation. The possible relation set contains both generic (*is a sub-category of*, *is identical to*, *is the opposite of*) and domain-specific (*opposes*, *supports*, *threatens*, *influences*, *works on/with*) relations. Two further relations, *other* and *is not related to* were added for cases not covered by the previous eight relations. The game's interface allows players to switch the position of the two concepts or to skip ambiguous pairs. Guess What?! [18] goes beyond eliciting or verifying relations between concepts to creating com-

SW Life-cycle Stage	Approach	Genre	Solved Task
Stage 1: Build and maintain Semantic Web vocabularies	Verbosity [33]	GWAP	(T3) Specification of Relation Type
	Common Consensus [17]	GWAP	Provide goal related information
	Categorilla [31]	GWAP	(T3) Specification of Relation Type
	Free Association [31]	GWAP	(T1) Specification of Term Relatedness
	InPho [9]	MLab	(T3) Specification of Relation Type (subs)
			(T1) Specification of Term Relatedness
	Noy [19]	MLab	(T2) Verification of Relation Correctness (subs)
	OntoPronto [27]	GWAP	Class vs. instance decisions
			(T3) Specification of Relation Type (subs/instOf)
	Climate Quiz [26]	GWAP	(T3) Specification of Relation Type (8 relations)
	Guess What?! [18]	GWAP	Verify complex class definitions
			Generate class names for complex defs
Stage 2: Align Semantic Web vocabularies	CrowdMap [25]	MLab	(T2) Verification of Relation Correctness (subs/eqv)
			(T3) Specification of Relation Type (subs/eqv)
	SpotTheLink [28]	GWAP	(T1) Specification of Term Relatedness
			(T3) Specification of Relation Type (subs/eqv)
	Cheatham [5]	MLab	(T2) Verification of Relation Correctness (eqv)
Stage 3: Annotate content, maintain annotations	ZenCrowd [8]	MLab	Text to URL mapping (annotation)
	WhoKnows? [34]	GWAP	Answering quiz questions
	RISQ! [36]	GWAP	Answering quiz questions
	UrbanMatch [4]	GWAP	Associate LOD concepts to images

Table 1

Overview of approaches addressing problems in various stages of the Semantic Web life-cycle [27], their genres and the type of crowdsourcing tasks that they employ.

plex concept definitions. The game explores instance data available as linked open data. Given a seed concept (e.g., banana), the game engine collects relevant instances from DBpedia, Freebase and OpenCyc and extracts the main features of the concept (e.g., fruit, yellowish) which are then verified through the collective process of game playing. The tasks performed by players are: Players (1) assign a class name to a complex class description (e.g., assign $Banana$ to $fruit\&yellow\&grows\ on\ trees$) and (2) verify such class definitions.

Eckert and colleagues [9] relied on MTurk microworkers to build a concept hierarchy in the philosophy domain. Crowdsourcing complemented the output of an automatic hierarchy learning method in: a) judging the relatedness of concept pairs (on a 5-points scale between unrelated and related) and b) specifying the level of generality between two terms (more/less specific than). Noy and colleagues [19] focused on verifying the correctness of taxonomic relations as a critical task while building ontologies.

Stage 2: Align Semantic Web vocabularies. The CrowdMap system enlists micro-workers to solve the ontology alignment task [25] by asking them to 1) verify whether a given relation is correct (e.g., “Is conceptA the same as conceptB? yes/no ") and 2) specify how two given terms are related, in particular by choosing between *sameAs*, *isAKindOf* and *notRelated*. CrowdMap is designed to allow sameAs, subsumption or generic mappings between classes, properties and axioms, but currently it only supports equivalence and subsumption mappings between classes. SpotTheLink has been instantiated to align the eCl@ss and UNSWPC [27] as well as the DBpedia and PROTON ontologies [28]. The final version of the game solves ontology alignment through two atomic tasks: (1) choosing a related concept – given a DBpedia concept players choose and agree upon a related PROTON concept; (2) specifying the type of relation between two concepts in terms of equivalence or subsumption. More recently, Cheatham and Hitzler [5] made use of MTurk to generate a mapping for the Conference track of the Ontology Alignment Evaluation Initiative (OAEI) thus pioneering the use of crowdsourcing for generating benchmark data in the Semantic Web research area. They conclude that crowdsourcing offers a scalable, cost-effective method for generating benchmarks that highly agree with expert opinion.

Stage 3: Annotate content and maintain annotations. In ZenCrowd [8] crowd-workers verify the output of automatic entity linking algorithms. Concretely, given a named entity, e.g., “Berlin", and a set of DBpedia URLs generated automatically, crowd-workers choose all the URLs that represent that entity or “None of the above" if no URL is suitable. In essence, this is an annotation task. WhoKnows? [34] and RISQ! [36] are GWAPs which rely on similar mechanisms: they use LOD facts to generate questions and use the answers to (1) evaluate property rankings (which property of an instance is the most important/relevant); (2) detect inconsistencies; and (3) find doubtful facts. The obtained property rankings reflect the wisdom of the crowd and are an alternative to semantic rankings generated algorithmically based on statistical and linguistic techniques. The games differ in the gaming paradigm they adopt. While WhoKnows?! uses a classroom paradigm and aims towards being an educational game, RISQ! is a Jeopardy-style quiz game. UrbanMatch [4] relies on players’ mobility to link LOD concepts to representative images from an image database.

2.1. *Typical crowdsourcing tasks in ontology engineering*

Based on the analysis above, we distill a set of recurrent basic crowdsourcing task types used to solve a variety of ontology engineering problems, as follows.

T1. Specification of Term Relatedness. Crowd-workers judge whether two terms (typically representing ontology concepts) are related. In some cases they are presented with pairs of terms [9] while in others they might need to choose a most related term from a set of given terms [28]. This type of crowdsourcing task is suitable both in ontology creation [9] and in ontology alignment scenarios [28].

T2. Verification of Relation Correctness. Presented with a pair of terms (typically representing ontology concepts) and a relation between these terms, crowd-workers judge whether the suggested relation holds. Frequently verified relations include generic ontology relations such as equivalence [5,25] and subsumption [19,25], which are relevant both in ontology evaluation [19] and ontology alignment scenarios [25].

T3. Specification of Relation Type. In these tasks, crowd-workers are presented with two terms (typically corresponding to ontology concepts) and choose an appropriate relation from a set of given relations. Most efforts focus on the specification of generic ontology relations such as equivalence [25,26,28], subsumption [9,25,26,27,28], disjointness [26] or instanceOf [26,27]. The verification of domain-specific named relations such as performed by Climate Quiz [26] is less frequent.

T4. Verification of Domain Relevance. For this task, the crowdworkers confirm whether a given term is relevant for a domain of discourse. This task is mostly needed to support scenarios where ontologies are extracted using automatic methods, for example, through ontology learning.

The core crowdsourcing tasks above have been used by several approaches and across diverse stages of ontology engineering, thus being of interest in a wide range of ontology engineering scenarios. As such, they guided the development of our plugin, which currently supports tasks T2, T4, and partially T3.

3. The uComp Protégé plugin

In order to support ontology engineers to easily and flexibly integrate crowdsourcing tasks within their work, we implemented a plugin in Protégé, one of the most widely used ontology editors. The typical workflow of using the plugin involves the following main stages (as also depicted in Figure 1).

1. Task Specification. An ontology engineer using Protégé can invoke the functionalities of the plugin from within the ontology editor at any time within his current work. The plugin allows specifying some well defined ontology engineering tasks, such as those discussed in Section 3.2 above. The view of the plugin that is appropriate for the task at hand is added to the editor’s user interface via the *Window* $\rightarrow$ *Views* menu.

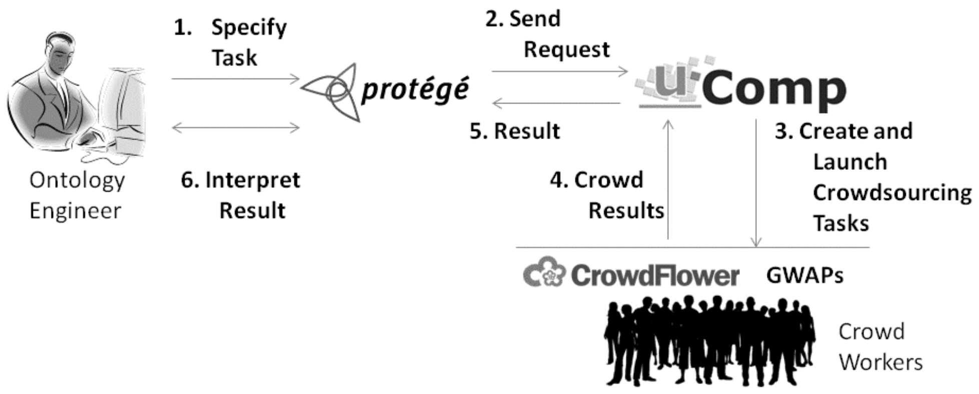


Fig. 1. Main stages when using the uComp plugin.

The ontology engineer then specifies the part of the ontology to verify (e.g., a specific class or all classes in the ontology), provides additional information and options in the plugin view and then starts the crowdsourcing deployment process. Crowdsourced tasks can be canceled (or paused) anytime during the crowdsourcing process. We further detail the plugin's functionality in Section 3.1.

2. **Task Request.** The plugin uses the uComp API[1] to request the processing of the task by the crowd.
3. **Creation of Crowdsourcing Tasks.** The process of crowdsourcing happens through the uComp platform, a hybrid-genre crowdsourcing platform which facilitates various knowledge acquisition tasks by flexibly allocating the received tasks to GWAPs and/or mechanised labour platforms alike (in particular, CrowdFlower) [24] depending on user settings. In Section 3.2 we present the crowdsourcing tasks created by the uComp platform (developed in the uComp project, `http://www.ucomp.eu/`).
4. **Collection of Crowd Results.** The uComp platform collects crowd-work harvested by individual genres (GWAPs and micro-task crowdsourcing). Mechanisms for evaluating the quality of each contribution are in place, which aim to filter out potential spammers.
5. **Combination of Crowd Results.** When all crowdsourcing tasks of a job have been completed, the platform combines the results and provides them to the plugin. If the task was crowdsourced to different genres, then the results provided by different platforms must be merged. Subsequently, the individual judgements can be aggregated using appropriate aggregation mechanisms (e.g., majority voting, weighted majority voting, average, collection).
6. **Result Presentation and Interpretation.** As soon as available, the plugin presents the results to the ontology engineer and saves them in the ontology. All data collected by the plugin is stored in the ontology in `rdfs:comment` fields, for example information about the ontology domain, the crowdsourcing job ID, and the crowd-created results. Depending on the result, the ontology engineer will perform further actions such as deleting parts of the ontology which have been validated as non-relevant.

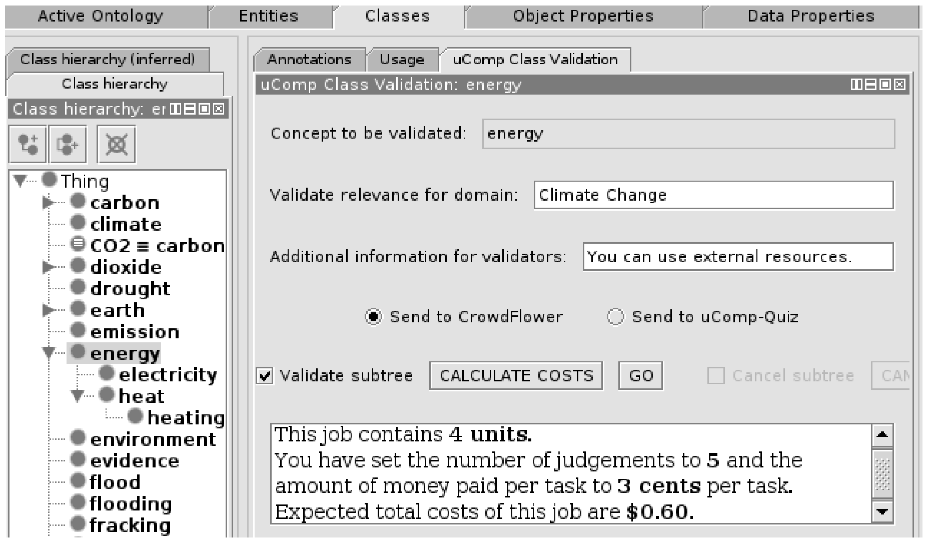
[1]`http://tinyurl.com/uCompAPI`

3.1. Plugin functionality

The plugin provides a set of views for crowdsourcing the following tasks:

– Verification of Domain Relevance (T4)
– Verification of Relation Correctness - Subsumption (T2)
– Verification of Relation Correctness - InstanceOf (T2) - the verification of *instanceOf* relations between an individual and a class.
– Specification of Relation Type (T3) is a Protégé view component that collects suggestions for labeling unlabeled relations by assigning to them a relation type from a set of relation types specified by the ontology engineer.
– Verification of Domain and Range where crowdworkers validate whether a property's *domain* and *range* axioms are correct.

The following subsections contain detailed descriptions of all plugin functionalities.



Concept to be validated: energy

Validate relevance for domain: Climate Change

Additional information for validators: You can use external resources.

[✓ Send to CrowdFlower] Send to uComp-Quiz

[✓] Validate subtree

This job contains **4 units.**
You have set the number of judgements to **5** and the amount of money paid per task to **3 cents** per task.
Expected total costs of this job are **$0.60.**

Fig. 2. The interface of the uComp Class Validation view used to create a Verification of Domain Relevance (T4) task.

Verification of Domain Relevance (T4) is supported by the *uComp Class Validation* view of the plugin and crowdsources the decision of whether a concept (class) is relevant for a domain. Figure 2 shows the screenshot of this view for the class "energy" before initiating the verification. The plugin view's interface contains the following information:

Task specific information contains elements such as the concept selected by the user for validation. This part of the view is diverse among different plugin functionalities.

Generic information such as the *domain* of the ontology, i.e., the field of knowledge which the ontology covers, is present in all views of the plugin. If entered once, the domain will be stored in the ontology (as `rdfs:comment`) and be pre-filled subsequently, but it can also be changed at any time.

Additional information For every task, the plugin contains a predefined task description (typically including examples) which is presented to the crowd-worker. If the ontology engineer wants to extend this task description, (s)he can provide more guidelines in the *additional information* field. This functionality is present in all the views of the plugin.

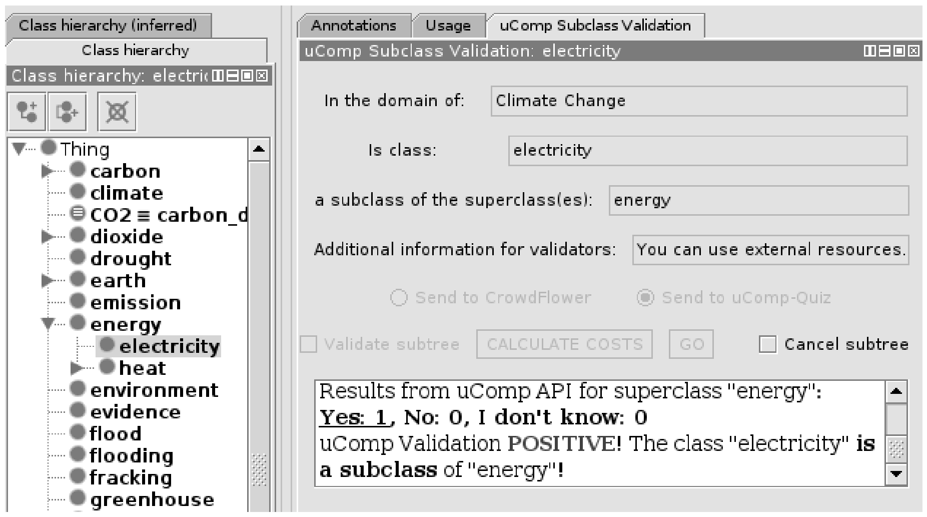
Recursive control allows performing a task (e.g., domain relevance validation) not only for the current class, but for a larger part of or even the entire ontology. If the *Validate subtree* option is selected, the plugin crowdsources the specified task for the current concept and all its subconcepts recursively. To apply the functionality to the entire ontology, the plugin is invoked from the uppermost class, i.e., (*Thing*). For example, in Figure 2, the class "energy" has 3 subconcepts, which adds up to 4 units being verified if *recursive control* is activated.

Calculate costs is a button that computes and displays expected cost of HC verification before actually starting the job. The button is only available in the user interface if CrowdFlower has been selected as crowdsourcing method (and not the GWAP).

GO button to start the crowdsourcing process.

Verification of Relation Correctness - Subsumption (T2) is achieved with the *uComp SubClass Validation*. When selecting a class in Protégé, the plugin automatically detects its superclasses (if any) and fills the boxes in the plugin user interface (UI). As with any plugin functionality, the elements of the user interface are described in the plugin documentation, and additional information is also given interactively as mouse-over overlays. As soon as results are available these are presented in the UI, as shown in Figure 3. The screenshot gives an example with one evaluator, who rated the *IS-A* relation between "electricity" and "energy" as valid (positive) in the domain of "Climate Change". If the majority of judgements is negative, a button to remove the relation is displayed.

Verification of Relation Correctness - InstanceOf (T2) validates the instanceOf (*rdf:type*) between an individual and a class. It is performed with the *uComp*



In the domain of: Climate Change

Is class: electricity

a subclass of the superclass(es): energy

Additional information for validators: You can use external resources.

Results from uComp API for superclass "energy":
Yes: 1, No: 0, I don't know: 0
uComp Validation **POSITIVE**! The class "electricity" **is a subclass** of "energy"**!**

Fig. 3. Screenshot showing the interface for subClassOf relation validation, including the display of results.

Individual Validation view. The user simply selects an individual, and the plugin automatically displays the individual's type in the user interface. The user can either validate single individuals, or all individuals of a certain type (e.g., all individuals of type *Region* as in Figure 4) at once.

Specification of Relation Type (T3) is a Protégé view named *uComp Relation Label Suggestion*, found in the *Object property* views. This view is used to assign relation types to unlabeled object properties. As a convention, unlabeled object properties are all existing properties with the label "relation". This task type is relevant in systems that learn association between classes, but not a specific relation type, for example in ontology learning scenarios. The Protégé user selects relation type candidates from the existing object properties. Figure 5 gives an example with 9 candidates selected. Crowd workers choose one option from the list of candidates (or the option "Other"). As in other tasks, either a single relation can be specified by the crowd, or all unlabeled relations at once, by marking "Suggest for all relations" in the user interface.

Verification of Domain and Range is another view applied to object properties, named *uComp Object Property Validation*. This view helps to verify the correctness of domain and range restrictions. For each property, two validation units are generated, one for the property's domain, and one for the range axiom.

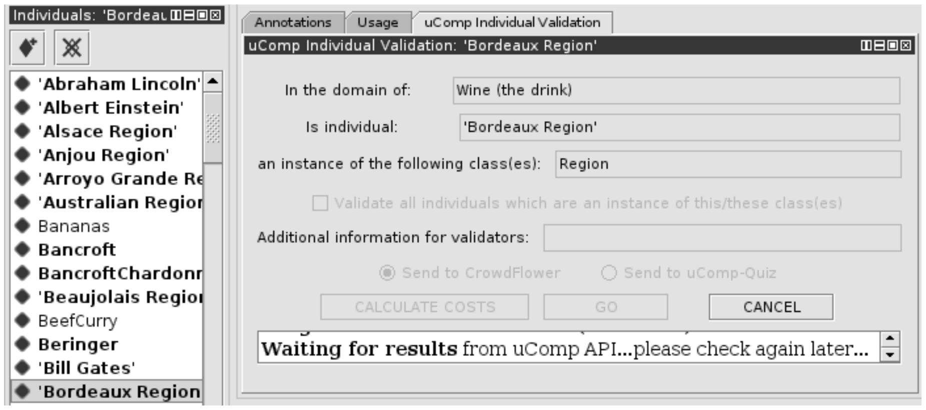
3.2. *Crowdsourcing task interfaces*

Upon receiving the request from the Protégé plugin, the uComp API selects the appropriate crowdsourcing genre and creates the relevant crowd-jobs. Currently the platform can crowdsource tasks either to GWAPs such as Climate Quiz [26] or to CrowdFlower, with a hybrid-genre strategy currently being developed. In this paper, we test the plugin by crowdsourcing only through CrowdFlower.

Figure 6 depicts the crowdsourcing interfaces created automatically by the uComp platform for the verification of domain relevance (part a) and the validation of subsumption relations (part b) while Figure 7 shows the interface of instance verification tasks. Figure 8 displays the automatically generated crowdsourcing interface for the relation specification task.

The uComp platform requires only the task data from the Protégé plugin and it provides relevant instructions as well as gold units to all tasks. Additionally, each crowdsourcing interface is extended with straightforward verification questions (i.e., typing some letters of the input terms). It has been shown experimentally (e.g., [14,16]), that extending task interfaces with explicitly verifiable questions forces workers to process the content of the task and also signals that their answers are being scrutinized. This seemingly simple technique had a significant positive effect on the quality of the collected data, as reported in [14,16].

To ensure a good quality output, by default all created jobs are assigned to Level 3 CrowdFlower contributors which are the contributors delivering, on average, the highest quality work. Also, for the moment we assume that the (labels of the) verified ontologies will be in English and therefore we restrict contribu-


Individuals: 'Bordeau

'Abraham Lincoln'
'Albert Einstein'
'Alsace Region'
'Anjou Region'
'Arroyo Grande Re
'Australian Regior
Bananas
Bancroft
BancroftChardonr
'Beaujolais Regio
BeefCurry
Beringer
'Bill Gates'
'Bordeaux Region

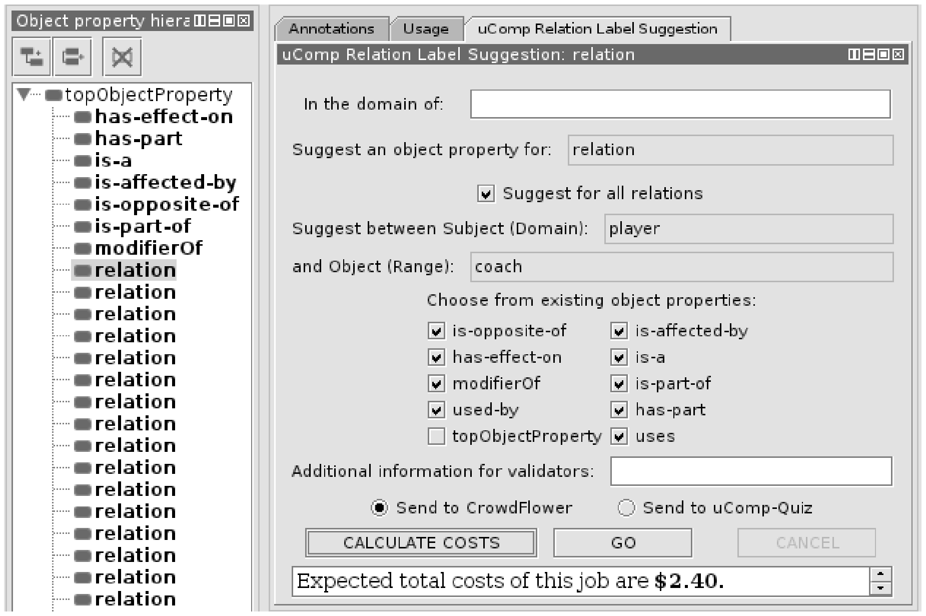
Annotations | Usage | uComp Individual Validation

uComp Individual Validation: 'Bordeaux Region'

In the domain of: Wine (the drink)
Is individual: 'Bordeaux Region'
an instance of the following class(es): Region
Validate all individuals which are an instance of this/these class(es)
Additional information for validators:
[✓ Send to CrowdFlower] Send to uComp-Quiz
CALCULATE COSTS | GO | CANCEL

Waiting for results from uComp API...please check again later...

Fig. 4. Screenshot showing the interface for individual validation, currently waiting for results from the uComp API.


Object property hiera

topObjectProperty
has-effect-on
has-part
is-a
is-affected-by
is-opposite-of
is-part-of
modifierOf
relation
relation
relation
relation
relation
relation
relation
relation
relation
relation
relation
relation
relation
relation
relation
relation

Annotations | Usage | uComp Relation Label Suggestion

uComp Relation Label Suggestion: relation

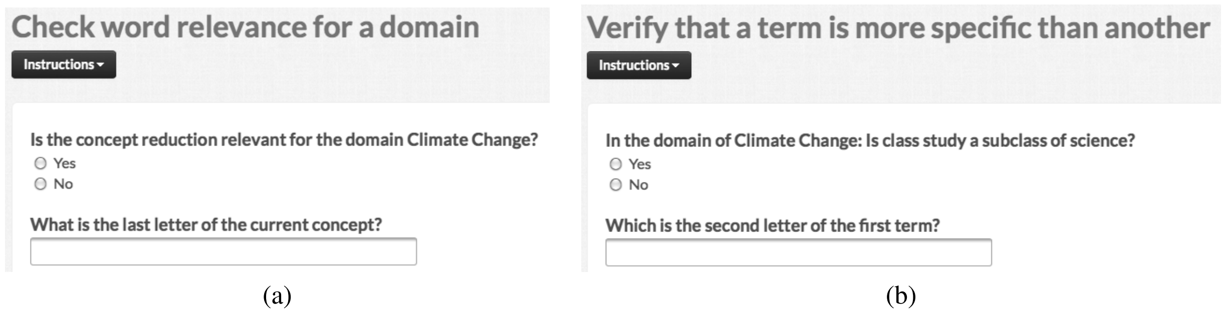
In the domain of:
Suggest an object property for: relation
✓ Suggest for all relations
Suggest between Subject (Domain): player
and Object (Range): coach
Choose from existing object properties:

✓ is-opposite-of	✓ is-affected-by
✓ has-effect-on	✓ is-a
✓ modifierOf	✓ is-part-of
✓ used-by	✓ has-part
topObjectProperty	✓ uses

Additional information for validators:
[✓ Send to CrowdFlower] Send to uComp-Quiz
CALCULATE COSTS | GO | CANCEL

Expected total costs of this job are **$2.40.**

Fig. 5. The interface of relation type suggestion, the user selects candidates from all existing properties.


Check word relevance for a domain
Instructions

Is the concept reduction relevant for the domain Climate Change?
Yes
No

What is the last letter of the current concept?

(a)

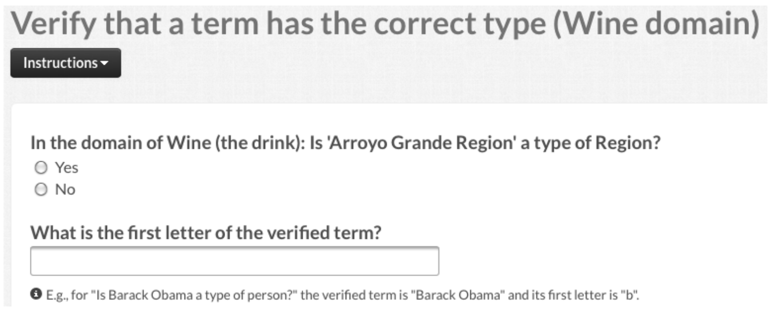
Verify that a term is more specific than another
Instructions

In the domain of Climate Change: Is class study a subclass of science?
Yes
No

Which is the second letter of the first term?

(b)

Fig. 6. Generated CrowdFlower job interface for (a) the Verification of Domain Relevance (T4) and (b) the Verification of Subsumption (T2) tasks.


Verify that a term has the correct type (Wine domain)

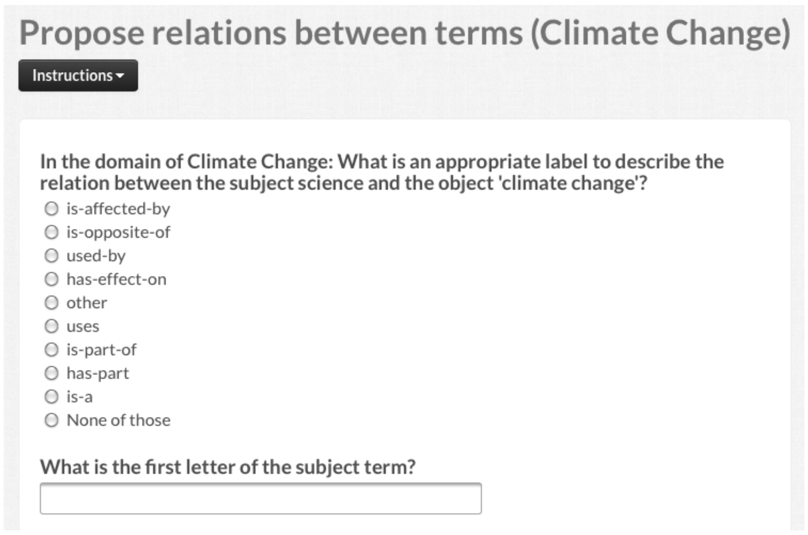
In the domain of Wine (the drink): Is 'Arroyo Grande Region' a type of Region?

- Yes
- No

What is the first letter of the verified term?

E.g., for "Is Barack Obama a type of person?" the verified term is "Barack Obama" and its first letter is "b".

Fig. 7. Generated CrowdFlower job interface for the Verification of InstanceOf (T2) task.


Propose relations between terms (Climate Change)

In the domain of Climate Change: What is an appropriate label to describe the relation between the subject science and the object 'climate change'?

- is-affected-by
- is-opposite-of
- used-by
- has-effect-on
- other
- uses
- is-part-of
- has-part
- is-a
- None of those

What is the first letter of the subject term?

Fig. 8. Generated CrowdFlower job interface for the Specification of Relation Type (T3) task.

tors to the main English speaking countries: Australia, United Kingdom and United States. In each created job we present 5 units per task and for each unit we collect 5 individual judgements. A price per task of \$0.05 was specified for all jobs. A task is complete when all requested judgments have been collected.

The plugin is available from Protégé's central registry as the *uComp Crowdsourcing Validation plugin*. The plugin has been tested with Protégé versions 4.2 and 4.3, as well as the recent version 5.0 (beta). A local configuration file contains the uComp-API key[2] and various adaptable settings (e.g., judgements per unit, price per unit).

[2]Request a key from the uComp team, see `http://tinyurl.com/uCompAPI`

4. Evaluation setup

We performed two main groups of experimental evaluations of the uComp Plugin. Table 2 displays an overview of the various experiments performed including their focus and the various experimental settings, described in what follows.

The first set of evaluations focuses on assessing the **feasibility** of using the uComp plugin as part of various tasks performed during ontology engineering. The primary goal of these feasibility evaluations is understanding the time and cost savings made possible by the use of the plugin in comparison to scenarios where an ontology engineer performs the tasks. As such, they are performed on small ontologies from various domains. The setup involves a group of 8 (or 5) ontology engineers which perform the same tasks over the same datasets but using two different approaches. In the first setting (**S_Manual**), all ontology engineers

Eval.	Evaluation	Data			Manual	CF Settings	
Type	Task	Ontology	Evaluated Feature	No. of Features	Evalu- ators	U,P,J/ Task	Worker selection
Fea- sibi- lity	T_DomRel	Cl. Change	Class	101	8	5,0.05,5	Level 3, Australia, UK, USA
	T_DomRel	Finance	Class	77	8		
	T_SubsCorr	Cl. Change	Subs. Rel.	43	8		
	T_SubsCorr	Finance	Subs. Rel.	20	8		
	T_InstOfCorr	Wine	InstOf Rel.	116	8		
	T_RelTypeSpec	Cl. Change	Unnamed Rel.	24	5		
	T_RelTypeSpec	Tennis	Unnamed Rel.	24	5		
Scala- bility	T_DomRel	Human	Class	4304	0	5,0.03,3	
	T_SubsCorr	Human	Subs. Rel.	3761	0	5,0.05,3	

Table 2

Overview of evaluation tasks performed, including used datasets and settings. U,P,J = Unit, Price ($), Judgements.

used the traditional (that is manual) approach to perform the ontology engineering tasks. In the second setting (**S_Crowd**), four of the eight ontology engineers used the Plugin to crowdsource (that is, create and launch) the same ontology engineering tasks, after being given a brief tutorial about the plugin (30 minutes). The two settings were then compared along the time, cost and quality dimensions. Time was measured as number of minutes to complete a task.

We evaluate the Plugin in the context of an ontology learning scenario as described in the introduction because i) bootstrapping ontology engineering by extracting an initial ontology automatically is a feasible and frequent ontology engineering approach and ii) automatically generated ontologies present errors that are best solved through human intervention. After the verification step with the uComp plugin, the resulting ontologies are used as part of a media monitoring tool[3] with the purpose of visualising information extracted from text corpora in a way that is meaningful to the web (i.e., non-specialised) audience. Therefore, agreement on the elements of these ontologies by the general public is, in this particular use case scenario, very important.

To estimate the cost, time and quality aspects when working on large, real-life ontologies we perform a set of **scalability** evaluations. Given the large scale of these experiments it is not feasible to perform them with domain experts as well. Therefore the focus is not on understanding the time/cost savings with respect to manual work, but rather to measure the time and cost involved when running such large-scale tasks.

[3]`http://www.ecoresearch.net/climate/`

4.1. Evaluation tasks

Plugin evaluations were performed over four different ontology engineering tasks in order to 1) test different functionalities of the plugin; and 2) to obtain evaluation results over a range of tasks. These tasks are:

T_DomRel: Verification of Domain Relevance (T4). For each concept of the ontology decide whether it is relevant for the domain in question (in our case, climate change and finance). In S_Manual, evaluators were asked to perform this task by assigning True/False values to a class level annotation property (called $uComp\_class\_relevance$) that we created for the purposes of our experiments.

T_SubsCorr: Verification of Relation Correctness – Subsumption (T2). For all subsumption relations in the ontology evaluators verified whether these relations were correct. In S_Manual, evaluators recorded their judgements in an annotation property at the relation level created for the experiments (called $uComp\_subclassof\_check$).

T_InstOfCorr: Verification of Relation Correctness – InstanceOf (T2). The evaluation of this task has been performed on the Wine ontology. The wine.owl ontology originally has 54 instances of the concepts $WineGrape$ and $Region$. We extended those concepts with 62 instances from other domains, namely famous persons and multinational companies. The ontology engineers were asked to verify all instances of these two classes and record their judgements with an annotation named *uComp_instanceOf_check* that we created specifically for the evaluation.

T_RelTypeSpec: Specification of Relation Type - (T3). For this task we focused on two automatically learned ontologies from the domains of Climate Change and Tennis. The domain engineers were asked to visit all unnamed relations and select from a set of relations.

In the scalability evaluations, we focus on evaluating two functionalities of the plugin, namely the domain relevance check and checking the correctness of subsumption relations.

4.2. Evaluation measures

The goal of the evaluation is to assess the improvements that the uComp Plugin could enable in an ontology engineering scenario in terms of typical project completion aspects such as time, cost and quality of output. The usability of the plugin is an additional criteria that should be evaluated. Concretely, the evaluation goals can be summarised into the following questions:

Time *How does the use of the plugin affect the time needed to perform ontology engineering tasks?* We distinguish the total task time (T_{tt}) as the time taken from the start of the ontology engineering task until its finalisation; and the time of the ontology engineer spent actively in the task (T_{oe}). In a crowdsourced scenario, $T_{oe} < T_{tt}$, because the ontology engineer is only actively working during the outsourcing of the task. In contrast, in a traditional scenario $T_{oe} = T_{tt}$.

Cost *Are there cost benefits associated with the use of the plugin?* We compute costs related to payments for the involved work-force, that is payments to ontology experts (C_{oe}) and payments to crowdworkers (C_{cs}). Ontology engineer costs are computed by multiplying the time they spend on the task (T_{oe}) with an average monthly wage. To allow comparison to other studies [20], the wage of a research scientist was assumed to be $54,000 per annum.

Quality *What are the implications on the quality of the resulting output when using the Plugin?* Several studies have shown that the quality of various knowledge acquisition tasks performed by crowdworkers is, in general, similar to (or even better than) the quality of tasks performed by ontology engineers [19,23,29]. While the quality of the obtained data is not the core focus of our evaluation, we expect to obtain similar results to previous studies.

Usability *Is the plugin usable?* As any end-user tool, the plugin should be easy to understand and use by the average ontology engineer already familiar with the Protégé environment.

4.3. Evaluation data

4.3.1. Feasibility track

The input to most evaluation tasks are ontologies generated by the ontology learning algorithm described in [35] (primarily) from textual sources. In the feasibility track, we evaluate the plugin over four ontologies covering four diverse domains (climate change, finance, tennis and wine). All four domains are more or less general knowledge domains, but some (climate change, tennis) require domain familiarity or interest.

The ontologies tested as part of this evaluation experiments are of small to medium size (see Table 3). The *Climate Change* ontology has 101 classes and 81 relations (out of which 43 are taxonomic relations, and 24 unnamed relations) while the *Finance* ontology has 77 classes and 50 relations (20 of which are taxonomic relations). The *Tennis* ontology was used for the relation suggestion task only, it contains 24 unnamed relations. The ontologies were used as generated by the ontology learning process.

The ontologies used in the evaluation process, the instructions given to the manual evaluators, and the results, are found online[4]. Additionally to automatically generated ontologies, we have made use of the *Wine* ontology[5] when evaluating the instanceOf verification tasks (T2).

4.3.2. Scalability track

For the scalability evaluation we chose the *Human* ontology which represents the human anatomy part of the NCI thesaurus and was made available as part of the Anatomy track of the Ontology Matching Initiative[6]. As shown in Table 3, this ontology is several degrees of multitude larger than the ontologies used for the feasibility evaluation. Additionally, it is a domain specific ontology and therefore allows evaluating the usefulness of the plugin for medium-sized to large domain specific ontologies, concretely for the anatomy domain.

[4]http://tinyurl.com/ucomp
[5]http://www.w3.org/TR/owl-guide/wine.rdf
[6]http://oaei.ontologymatching.org/2014/anatomy/index.html

	Ontology				
Nr. of	**Climate Change**	**Finance**	**Wine**	**Tennis**	**Human**
Classes	101	77	138	52	3304
Relations	77	50	-	67	-
IsA Relations	43	20	228	35	3761
Unnamed Relations	24	30	-	24	-
Instances	0	0	206	0	0

Table 3
Overview of the ontologies used in the feasibility and scalability evaluations.

Unlike in the previous experiments, this is an approved ontology (not a learned ontology) and therefore we can assume it as correct. This allows measuring the quality of the crowd work even without having a baseline created by domain experts. Our strategy is that of modifying the *Human* ontology by adding incorrect data to it and assess how effective the crowd is in filtering out the incorrect cases. Accordingly, we have performed the following changes to the ontology.

For experiments focusing on measuring the domain relevance of ontology concepts, the *Human* ontology already provides 3304 concepts. We have extended the ontology with 1000 additional classes with labels extracted using ontology learning techniques from corpora related to the domains of climate change and tennis. These 1000 labels have been manually verified to exclude labels that might refer to the human body. These 1000 classes have been added as random leaf classes to the ontology resulting in a total of 4304 concept labels to be verified for domain relevance. More precisely, for the 500 terms each from the domains of *climate change* and *tennis*, the insertion algorithm randomly selects a concept from the original ontology and then inserts a new sub-concept which has the term as concept label. This strategy guarantees that non-relevant concepts are evenly distributed in the ontology.

For experiments involving the verification of the correctness of subsumption relations, human.owl provides 3761 correct subsumption relations. To introduce incorrect relations, we identified 800 pairs of leaf concepts and swapped their places in the ontology, therefore creating 1600 incorrect subsumption relations. Again, concepts to be swapped are randomly selected, and marked with an *rdfs:comment* tag to easily find incorrect concepts in subsequent analysis.

4.4. Ontology engineers and crowd contributors

Regarding the evaluators, four were experienced Protégé users, the other four work in the Semantic Web area but have limited knowledge of Protégé and were shortly trained in Protégé. None of the ontology engineers involved had any strong expertise in a particular domain.

The crowdsourcing job settings are displayed in Table 2. For the feasibility evaluations we grouped 5 units/task, paid $0.05 per task, requested 5 judgement per unit and crowdsourced our jobs to Level 3 workers from English speaking countries, in particular, UK, USA and Australia. Level 3 workers are workers that consistently provide the highest quality work on CrowdFlower. For the scalability evaluations, the job settings were maintained similar to the feasibility level evaluations, with two notable changes. Firstly, we requested only 3 judgements per unit (as opposed to 5 originally) as this setting is more feasible for large-scale tasks where every extra judgement introduces important increases in job costs and duration. We paid $0.03 for each domain relevance verification task (ie., 5 units) and $0.05 for a task of verifying subsumption relations to cater for the varying difficulty levels of these two tasks. Secondly, we have selected the *"Units Should be Completed in the Order they were Uploaded"* option offered by CrowdFlower which is recommended for large scale projects. When enabled, this option crowdsources the data in batches of 1000 units at a time. Besides this job setting changes, the pricing model of CrowdFlower has changed from May 2014, when the feasibility experiments were run, by adding to the actual worker costs 20% transaction fees.

5. Feasibility evaluation results

Task Duration. Table 4 lists the task duration for the *Climate Change* and *Finance* ontologies and the two settings (*S_Manual* and *S_Crowd*), detailed in terms of the average time intervals spent by the ontology engineer (T_{oe}), by using crowdsourcing (T_{cs}) and the total time of the task ($T_{tt} = T_{oe} + T_{cs}$). In the case of *S_Crowd*, the time needed for the ontology engineers to create and launch the crowdsourcing task was on average between 1 and 2 minutes. To simplify calculations, we chose to take the average time as 2 minutes across all tasks. We notice that the time reduction ratio for the ontology engineer across the two settings (computed as the ratio of the ontology engineering time in *S_Manual* and *S_Crowd*) is significant and ranges from a 13.7 fold reduction to a 7.5 fold reduction, with an overall average of 11: thus ontology engineers need to spend 11 times less time on the task when using the Plugin than in the manual scenario. The duration of the overall task increases and varies between 2.4 (142 mins) and 4.7 (282 mins) hours. Note however, that the current evaluation setup maximizes quality rather than speed. Faster completion rates (possibly at the expense of data quality) could have been obtained by not restricting the geographical location and previous achievements of the crowd-workers.

Costs. For the cost analysis, we compute average costs for the total task (C_{tt}) as the sum of the average cost of the ontology engineer (C_{oe}) and the average cost of the crowd-sourced tasks (C_{cs}) as detailed in Table 5. Considering an annual salary of $54,000 and a corresponding $26 hourly wage[7], average ontology engineering costs were computed based on the average times shown in Table 4. Cost savings were then computed for each cost category.

Ontology engineer cost savings are high and range from 92.4% to 86.15%, averaged at 89.9%. For the entire task, cost savings are moderate (19.7% - 60.5%, Avg = 39%), with *S_Crowd* reducing *S_Manual* costs with 40%. Note, however, that task level cost savings will ultimately depend on the cost that ontology engineers decide to pay to crowd-workers. For example, choosing a cheaper task setting than currently (i.e., 3 judgements, with $0.01 per task vs. the current 5 judgements and $0.05 per task) will lead to average cost savings of 83.3% for the total task (*S_CrowdCheap* in Table 5). From the plugin's perspective, the major goal is reducing ontology engineering costs, as crowdsourcing costs will depend on the constraints of the ontology engineer and are hard to generalise.

Data quality. Lower completion times and costs should not have a negative effect on the quality of the crowdsourced data. Since we do not possess a baseline for either of the two tasks, we will perform a comparative evaluation and contrast inter-rater agreement levels between ontology engineers with those of crowdworkers. We have measured inter-rater agreement with Fleiss' Kappa which is used to assess reliability of agreement with a fixed number of raters and categorical ratings assigned to a number of items.

Table 6 presents inter-rater agreement per task and per setting, with the number of raters per task given in parentheses. According to the interpretation of Landis and Koch [15] the inter-rater agreement among manual expert evaluators (*S_Manual*) is moderate. Agreement among the four groups of CrowdFlower workers is substantial (*S_Crowd*). The combined agreement (manual expert and crowdworkers) is always higher than for manual evaluators alone. A detailed inspection of results reveals that judgement is difficult on some questions, for example relevance of given concepts for the climate change domain often depends on the point of view and granularity of the domain model. But in general, crowdworkers have a higher inter-rater agreement, which often corresponds with the majority opinion of manual experts, thereby raising Fleiss' kappa (*S_ManualCrowd*). Also, the agreement between the crowd and experts is higher than among experts, possibly because crowdsourcing data is the majority view derived from 5 judgements as compared to a single expert judgement.

Evaluation of the InstanceOf verification task For evaluating the task *Verification of Relation Correctness - InstanceOf (T2)* we used the Wine Ontology, specifically the classes *WineGrape* and *Region*, with 8 individuals for WineGrape and 36 for Region. To test the plugin, we added 26 non-relevant individuals to the class of WineGrape, and 36 non-relevant individuals to the class of Region. The individuals added to WineGrape stem from a list of important multi-national company names while in the case of Region we used famous persons. In sum, this leads to 116 individuals for verification, 44 for WineGrape and 72 for Region. For all individuals, CrowdFlower workers were asked to judge whether the instanceOf relation between the class and the individual is actually correct.

Results and Analysis As the task's difficulty was low, the crowd and the domain experts answered 100% of units correctly, so the interesting point is the comparison of time and cost between crowd workers and domain experts. Domain experts needed on average 45.6 minutes to complete this task, while the crowdsourcing tasks took on average about 120 minutes. This still translates in important cost savings with the

[7]In practice, considering benefits, overhead and vacation, the actual costs for a productive hour are likely to be higher than $26. Nevertheless, we decided to keep $26 in order to be able to compare our findings to similar studies.

	Climate Change Ontology						Finance Ontology					
	T_DomRel			T_SubsCorr			T_DomRel			T_SubsCorr		
	T_{oe}	T_{cs}	T_{tt}	T_{oe}	T_{cs}	T_{tt}	T_{oe}	T_{cs}	T_{tt}	T_{oe}	T_{cs}	T_{tt}
S_Manual (Avg)	27.4	0	27.4	23.0	0	23.0	21.3	0	21.3	15.0	0	15.0
S_Manual (StdDev)	5	0	5	6.2	0	6.2	7.1	0	7.1	5.6	0	5.6
S_Crowd (Avg)	2	240	242	2	280	282	2	140	142	2	200	202
S_Manual/S_Crowd	13.7	-	0.11	12.5	-	0.08	10.65	-	0.15	7.5	-	0.07

Table 4

Task duration in minutes per ontology, evaluation task and setting.

	Climate Change Ontology						Finance Ontology					
	T_DomRel			T_SubsCorr			T_DomRel			T_SubsCorr		
	C_{oe}	C_{cs}	C_{tt}	C_{oe}	C_{cs}	C_{tt}	C_{oe}	C_{cs}	C_{tt}	C_{oe}	C_{cs}	C_{tt}
S_Manual (Avg)	11.9	0	11.9	9.9	0	9.9	9.2	0	9.2	6.5	0	6.5
S_Crowd (Avg)	0.9	8.48	9.38	0.9	3.58	4.48	0.9	6.49	7.39	0.9	1.67	2.57
Cost Savings (%)	92.4	-	21.2	90.1	-	54.7	90.2	-	19.7	86.15	-	60.5
S_CrowdCheap (Avg)	0.9	1.02	2.1	0.9	0.43	1.33	0.9	0.78	1.68	0.9	0.2	1.1
Cost Savings (%)	92.4	-	82.3	90.1	-	86.5	90.2	-	81.7	86.15	-	83

Table 5

Average costs (in \$) for the ontology engineer (C_{oe}), crowd-workers (C_{cs}) and the entire task (C_{tt}) across ontologies and settings.

	Climate Change Ontology		Finance Ontology	
	T_DomRel	T_SubsCorr	T_DomRel	T_SubsCorr
S_Manual	0.338 (8)	0.502 (8)	0.496 (8)	0.419 (8)
S_Crowd	0.633 (4)	0.841 (4)	0.520 (4)	0.826 (4)
S_ManualCrowd	0.392 (12)	0.582 (12)	0.505 (12)	0.508 (12)

Table 6

Fleiss' Kappa values of inter-rater agreement per setting and when combining the data of the two settings.

	No. of instances	Accuracy	Avg. time (Min)	Avg. Cost (\$)
Manual	116	1.0	45.6	19.76
CrowdFlower	116	1.0	120	6.31

Table 7

Results of the InstanceOf Verification (T2) task.

crowdsourced experiments costing only \$6.31 as opposed to \$19.76 corresponding to experts' pay.

Evaluation of the Relation Specification task. To evaluate the plugin regarding the task *Specification of Relation Type (T3)* we used two ontologies which were generated automatically in the domains of climate change and tennis (the sport). These ontologies each contain 24 unnamed relations between subject and object concepts, i.e., 48 unlabeled relations in total. Domain experts and crowd workers were asked to assign one of the relation types "has-effect-on", "is-affected-by", "is-a", "is-opposite-of", "has-part", "is-part-of", "used-by" or"uses" to each unlabeled relation, or "other" is no type fits for the relation.

Results and Analysis. Similarly to the feasibility study on concept relevance and subsumption relations, we applied Fleiss' Kappa for a comparative evaluation of inter-rater agreement regarding relation name suggestion. Apart from retrieving judgements from crowd workers, five domain experts assess relation labels for the 24 relations each in the domains of climate change and tennis. Table 8 provides Fleiss' Kappa re-

	Climate Change	Tennis
S_Manual	0.536 (5)	0.366 (5)
S_ManualCrowd	0.531 (6)	0.368 (6)

Table 8

Fleiss' Kappa values for Specification of Relation Type (T3).

sults of inter-rater agreement for the two domains, with the number of raters per task given in parentheses. *S_Manual* refers to the ratings of the five domain experts, *S_ManualCrowd* adds the result from the crowd to these five judgements.

The agreement in the domain of climate change is moderate, in the domain of tennis it is fair, according to Landis and Koch [15]. The lower agreement values compared to those reported in Table 6 can be explained by the higher complexity of this task as it requires users to select a relation label from nine candidates. To complicate matters further, in some cases multiple relations labels are suitable candidates. The addition of crowd worker judgements to the judgements of domain experts influences the Fleiss' Kappa values only slightly, which suggests that the quality of crowd worker results is comparable to domain experts' quality.

Plugin usability was assessed by means of the System Usability Scale (SUS), the most used questionnaire for measuring perceptions of usability [3]. Based on data collected from the entire test population (i.e., all 8 ontology engineers), we obtained a SUS score of 85, which corresponds to the 90th percentile rank and positions the plugin in the class of "A" type system, that is systems with maximal usability. All evaluators agreed that (a) they would prefer using the plugin instead of performing the tasks manually and that (b) the use of the plugin saved a lot of their time. They considered the recursive task verification particularly useful when focusing on large (parts of the) ontologies. One suggested making the plugin data and results more visually appealing, and showing the anticipated cost before crowdsourcing – both of which have been implemented in the meantime. Given the small scale of the usability evaluation, we consider it only as indicative that the Plugin has a good usability. A larger usability study is planned as future work.

6. Scalability evaluation results

While the experiments above confirm the cost savings made possible by the plugin as well as the plugin's usability, it is important to also investigate the scalability of the proposed approach and its applicability when working with large and domain-specific ontologies. Table 9 lists the results of the large scale experiments for both tasks of Domain and Subsumption Verification on the *Human* ontology.

The domain verification task was completed by crowd-workers in 19 hours. To estimate the time it would have taken to perform this task manually we refer to the experiment duration measurements performed during the feasibility evaluations (see Table 4). For the climate change ontology, ontology engineers needed 27.4 minutes on average to judge the domain relevance for 101 concepts thus leading to an average concept verification speed of 3.68 concepts/min. Similarly, in the financial domain 21.3 minutes were needed on average to verify 77 concepts, resulting in an average speed of 3.61 concepts per minute. As the speeds of concept verification are very close across domains, for our estimation we consider an average speed of 3.65 concept verifications per minute. With this speed, ontology engineers would need 1179 minutes (19.65 hours) to verify the 4304 concepts.

Although the manual and crowd performed verification process would take the same amount of time in hours, for larger projects it is important to translate these hours into working days: indeed, while the crowd is available continuously, the availability of domain experts is determined by working hour schedules. Therefore, when translating working hours into working days, we define a crowd working day as having 24 hours, while an ontology engineer working day has 8 hours. It can be easily seen, that in practice, an ontology engineer would spend more than two days on this task (this is a best case assumption that does not take into account fatigue and breaks) while crowdsourcing could return results within one day.

In term of costs, for the domain verification task we spent a total of $130, where $104 are actual worker costs and $26 are CF transaction fees (TF). Using the $26 hourly wage as for our feasibility experiments, the estimated cost for manually performing this task is $511. Therefore, crowdsourcing costs are a quarter of the ontology engineering costs (25%).

The quality of the results was very high. CrowdFlower statistics show an inter-worker agreement of 98%. Indeed, crowd workers rated 4260 of 4304 concepts correctly, which corresponds to a remarkably high accuracy of 99%. There were only 7 false positives, i.e. non-relevant terms from the domains of climate change and sports which were rated as relevant to human anatomy. For example the term "Forehand" (a type of shot in tennis) was wrongly judged as part of the human body. The number of false negatives was higher (37) and included some ambiguous terms in human anatomy such as “Pyramid” or “Curved Tube”.

The subsumption verification task took significantly longer, needing 136 hours (5.6 days) to complete. Following a similar procedure as above, we compute an average subsumption verification speed of 1.6 relations/minute. As expected, this speed is lower than for the domain relevance verification since the task is more complex. We estimate that ontology engineers would need, on average 2350.6 minutes (39.20 hours) to verify 3761 subsumption relations.

In terms of costs we paid a total of $194 for the crowdsourcing task (out of which $39 were transaction fees). An ontology engineer employed for 39,20 hours would have costed $1019. Therefore, the total cost of crowdsourcing accounts to only 19% of the amount to be paid to the ontology engineer.

As some classes in the *Human* ontology have multiple super-classes, after swapping 1600 classes, the resulting ontology contains 2008 correct, and 1753 incorrect, subClass relations. CF workers judged 1812 relations as correct, 1949 as incorrect, so there was a substantial number of false negative ratings. Overall the accuracy of CF workers is 0.895 because 3367 of 3751 worker judgements were correct according to the ground truth provided by the *Human* ontology.

A detailed analysis reveals that the results contain 1713 true positives and 99 false positives, whereas the number of true negatives is 1654, leaving 295 false negatives. Regarding the false positives, many of the judgements intuitively make sense, e.g., that “Penis Erectile Tissue” is a subclass of “Reproductive System”, or that “Upper Lobe Of The Right Lung” is a subclass of “Organ”, none of which is stated by the *Human* ontology. Therefore, crowd judgements could help to identify questionable modelling assumptions made by domain experts which might need to be revised. Most wrong judgements concern false negatives, many of which are hard to assess from the concept labels. Examples include “Egg” subclass of “Germ Cell”, “Anatomic Sites” subclass of “Other Anatomic Concept”, and many similar types of relations.

7. Conclusions and Future Work

In this paper we investigated the idea of closely embedding crowdsourcing techniques into ontology engineering. Through an analysis of previous work using crowdsourcing for ontology engineering, we concluded that a set of basic crowdsourcing tasks are repeatedly employed to achieve a range of ontology engineering processes across various stages of the ontology life-cycle. We then presented a novel tool, a Protégé plugin, that allows ontology engineers to use these basic crowdsourcing tasks from within their ontology engineering working context in Protégé.

Our evaluation focused on assessing the concept of the crowdsourcing plugin. Although we plan to make use of the uComp platform, this particular evaluation forwarded all tasks directly to CrowdFlower and therefore is not influenced by the particularities of the uComp framework. As a first evaluation of the plugin, we focused on small-scale ontologies. An evaluation of the plugin in an ontology engineering scenario where automatically learned ontologies in two different domains are assessed for domain relevance and subsumption correctness, revealed that the use of the plugin reduced overall project costs, lowered the time spent by the ontology engineer (without extending the time of the overall tasks to over 4 hours) and returned good quality data that was in high agreement with ontology engineers. Finally, our evaluators provided positive feedback about the usability of the plugin.

A set of scalability evaluations aimed (1) to asses how well the proposed concept would scale to large ontologies and (2) to investigate whether crowdworkers can replace domain experts in specialized domains. The cost reductions were significant, with the crowdsourcing payments accounting to only a quarter or less of the estimated ontology engineering costs. Timings are comparable among the two approaches, however, we believe that a self-managed batching of the tasks would have lead to faster completion of tasks as opposed to using the built-in CrowdFlower batch based processing. The obtained quality depends on task difficulty, and ranges from very high (99% accuracy) for a simpler task to an acceptable 89% accuracy for the more difficult task of judging subsumption correctness. As such, our results are in-line with earlier studies that obtained similar quality results from crowds in specialized domains such as philosophy [9] and bio-medicine [19].

7.1. Limitations

The results that can be obtained with the uComp Protégé plugin might vary greatly between different tasks (depending on their type and the difficulty of

	Domain Verification		**Subsumption Verification**	
	Crowd	**Est. Manual**	**Crowd**	**Est. Manual**
Time (H/Day)	19/0.8	19.65/2.5	136/5.6	39.20/4.9
Cost ($)	104+26TF	511	155+39TF	1019
Quality (Accuracy)	0.99	–	0.895	–

Table 9

Results of the large scale evaluation.

the domain) but also depending on the timing when the tasks are crowdsourced and the response of the available work-force. Therefore, *integrating such techniques in projects where strict deadlines must be met is a challenging task*.

Additionally, our tool will benefit those ontology development projects which mostly involve the tasks we support and need to perform them extensively. For example, an ontology development project focusing on an ontology with a rich hierarchy and many inheritance levels, will benefit more from the tasks of subsumption verification than projects which develop large, but flat ontologies. Since ontology development processes greatly differ among themselves, there are currently no systematic studies about the relative importance of each task in ontology development projects, on average. As a result, *we cannot estimate the usefulness of the plugin for ontology development projects in general* either.

Since the plugin focuses on crowdsourcing, we must consider the following two *issues of legal and ethical nature*, which have so far not received sufficient attention. Firstly, no clear guidelines exist for how to properly acknowledge crowd contributions especially if their work would lead to some scientific results. Some volunteer projects (e.g., FoldIt, Phylo) already include contributors in the authors list [7,13]. The second issue is contributor privacy and well-being. Paid-for marketplaces (e.g., MTurk) go some way towards addressing worker privacy, although these are far from sufficient and certainly fall short with respect to protecting workers from exploitation, e.g. having basic payment protection [10]. The use of mechanised labour (MTurk in particular) raises a number of workers' rights issues: low wages (below $2 per hour), lack of protection, and legal implications of using MTurk for longer term projects. We recommend at the least conducting a pilot task to see how long jobs take to complete, and ensuring that average pay exceeds the local minimum wage.

7.2. Future Work

We consider our work as a first step towards the wide adoption of crowdsourcing by the ontology engineering community, and therefore, we see ample opportunities for future work.

Methodology and best practices. Since the use of crowdsourcing has matured enough, it is a good time to move on from isolated approaches towards a methodology of where and how crowdsourcing can efficiently support ontology engineers. Such methodological guidelines should inform tools such as our own plugin, while our plugin could offer a means to build and test these guidelines. Future work will also reveal the best use cases for the plugin – identifying those cases when it can be used to collect generic knowledge as opposed to application areas where it should be used to support the work of a distributed group of domain experts.

Towards expert sourcing. Although our first evaluation of an anatomy-specific ontology lead to good results, some highly specialized domains will benefit much more from the possibility of engaging a community of domain experts. The question therefore is whether the current plugin concept could be adapted for expert-sourcing. We already received a request for using the tool in this way by an ontology engineer who needed to collect domain knowledge from domain experts but was hampered in her task by the lack of convenient interfaces for knowledge acquisition from experts. The presented plugin can already be used as an interface between ontology engineers and domain experts, by simply activating the Internal Channel mode of CrowdFlower: with this setting, the created CrowdFlower tasks can be shared through a URL with the chosen expert group as opposed to being presented to the crowd (which corresponds to the External Channel). Therefore, the use of the tool for expert-sourcing is already possible with its current connection to CrowdFlower, but extensions that would connect Protégé to other domain specific tools are not included and will constitute the subject of our future work.

Plugin Development. In terms of the plugin development, we plan further extending its functionality (1) to support additional crowdsourcing tasks; (2) to allow greater control over job settings as well as (3) to permit monitoring of the results as they become available from within Protégé. Based on feedback from our research colleagues, we will prioritize supporting ontology localization with the plugin by crowdsourcing the translation of ontology labels into desired languages.

Further evaluations. We plan to evaluate the plugin in other ontology engineering scenarios as well (e.g., ontology matching) and to conduct larger scale usability studies.

Increasing task complexity. Recent advances in crowdsourcing include moving away from solving very simple tasks towards enabling experts drawn from the crowd to solve more complex tasks by creating micro-organizations and managing these organizations themselves [21]. While encouraging results have been reported on solving creative tasks such as animation creation or course curricula design, the applicability of this new approach for knowledge creation tasks is still unclear. In fact, Chilton explored different crowdsourcing approaches for the task of creating a taxonomy to describe a collection of text snippets (i.e., questions from Quora), and concluded the necessity of breaking down this task into manageable units of work [6]. An interesting future work question therefore refers to reconciling these two contradicting conclusions and exploring what is the maximum complexity of knowledge acquisition tasks that can be crowdsourced. Such fundamental investigations could result in redesigning our plugin to move on towards crowdsourcing more complex knowledge acquisition tasks than currently.

Acknowledgments.

The work presented in this paper was developed within the uComp project, which receives the funding support of EPSRC EP/K017896/1, FWF 1097-N23, and ANR-12-CHRI-0003-03, in the framework of the CHIST-ERA ERA-NET. Dr. Sabou's work was additionally supported by the Christian Doppler Forschungsgesellschaft, the Federal Ministry of Economy, Family and Youth, and the National Foundation for Research, Technology and Development - Austria.

References

[1] K. Bontcheva, H. Cunningham, I. Roberts, A. Roberts, V. Tablan, N. Aswani, and G.. Gorrell. GATE Teamware: A Web-based, Collaborative Text Annotation Framework. *Lang. Resour. Eval.*, 47(4):1007–1029, December 2013.

[2] K. Bontcheva, I. Roberts, L. Derczynski, and D. P. Rout. The GATE crowdsourcing plugin: Crowdsourcing annotated corpora made easy. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2014, April 26-30, 2014, Gothenburg, Sweden*, pages 97–100. The Association for Computer Linguistics, 2014.

[3] J. Brooke. SUS-A quick and dirty usability scale. *Usability Evaluation in Industry*, 189:194, 1996.

[4] I. Celino, S. Contessa, M. Corubolo, D. Dell'Aglio, E. Della Valle, S. Fumeo, and T. Kruger. Linking Smart Cities Datasets with Human Computation - The Case of UrbanMatch. In *In Proceedings of the International Semantic Web Conference*, pages 34–49, 2012.

[5] M. Cheatham and P. Hitzler. Conference v2.0: An Uncertain Version of the OAEI Conference Benchmark. In P. Mika, T. Tudorache, A. Bernstein, C. Welty, C. Knoblock, D. Vrandecic, P. Groth, N. Noy, K. Janowicz, and C. Goble, editors, *The Semantic Web - ISWC 2014*, volume 8797 of *LNCS*, pages 33–48. Springer International Publishing, 2014.

[6] L. B. Chilton, G. Little, D. Edge, D. S. Weld, and J. A. Landay. Cascade: Crowdsourcing Taxonomy Creation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '13, pages 1999–2008. ACM, 2013.

[7] S. Cooper, F. Khatib, A. Treuille, J. Barbero, J. Lee, M. Beenen, A. Leaver-Fay, D. Baker, Z. Popovic, and Foldit players. Predicting protein structures with a multiplayer online game. *Nature*, 466(7307):756–760, 2010.

[8] G. Demartini, D. E. Difallah, and P. Cudré-Mauroux. ZenCrowd: Leveraging Probabilistic Reasoning and Crowdsourcing Techniques for Large-scale Entity Linking. In *Proc. of the 21st Int. Conf. on World Wide Web*, pages 469–478. ACM, 2012.

[9] K. Eckert, M. Niepert, C. Niemann, C. Buckner, C. Allen, and H. Stuckenschmidt. Crowdsourcing the Assembly of Concept Hierarchies. In *Proc. of the 10th Annual Joint Conf. on Digital Libraries*, JCDL '10, pages 139–148. ACM, 2010.

[10] K. Fort, G. Adda, and K.B. Cohen. Amazon Mechanical Turk: Gold Mine or Coal Mine? *Computational Linguistics*, 37(2):413 –420, 2011.

[11] Y. Gil. Interactive knowledge capture in the new millennium: how the Semantic Web changed everything. *The Knowledge Engineering Review*, 26(1):45 – 51, 2011.

[12] J. Howe. Crowdsourcing: Why the Power of the Crowd is Driving the Future of Business, 2009. http://crowdsourcing.typepad.com/.

[13] A. Kawrykow, G. Roumanis, A. Kam, D. Kwak, C. Leung, C. Wu, E. Zarour, and Phylo players. Phylo: A Citizen Science

Approach for Improving Multiple Sequence Alignment. *PLoS ONE*, 7(3):e31362, 2012.

[14] A. Kittur, E. H. Chi, and B. Suh. Crowdsourcing User Studies with Mechanical Turk. In *Proc. of Human Factors in Computing Systems*, pages 453–456, Florence, Italy, 2008.

[15] J.R. Landis and G.G. Koch. The measurement of observer agreement for categorical data. *Biometrics*, 33(1):159–174, 1977.

[16] F. Laws, C. Scheible, and H. Schütze. Active Learning with Amazon Mechanical Turk. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 1546–1556. ACL, 2011.

[17] H. Lieberman, D. Smith, and A. Teeters. Common Consensus: A Web-based Game for Collecting Commonsense Goals. In *In Proceedings of the Workshop on Common Sense and Intelligent User Interfaces held in conjunction with the 2007 International Conference on Intelligent User Interfaces (IUI 2007)*, Hawaii, USA, 2007.

[18] T. Markotschi and J. Völker. Guess What?! Human Intelligence for Mining Linked Data. In *Proc. of the WS. on Knowledge Injection into and Extraction from Linked Data at EKAW*, pages 28–39, Lisbon, Portugal, October 15 2010. CEUR.

[19] N. F. Noy, J. Mortensen, M. A. Musen, and P. R. Alexander. Mechanical Turk As an Ontology Engineer?: Using Microtasks As a Component of an Ontology-engineering Workflow. In *Proc. of the 5th Annual ACM Web Science Conf.*, WebSci '13, pages 262–271, 2013.

[20] M. Poesio, U. Kruschwitz, J. Chamberlain, L. Robaldo, and L. Ducceschi. Phrase Detectives: Utilizing Collective Intelligence for Internet-Scale Language Resource Creation. *Transactions on Interactive Intelligent Systems*, 3(1):1–44, April 2013.

[21] D. Retelny, S. Robaszkiewicz, A. To, W. S. Lasecki, J. Patel, N. Rahmati, T. Doshi, M. Valentine, and M. S. Bernstein. Expert Crowdsourcing with Flash Teams. In *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology*, UIST '14, pages 75–85. ACM, 2014.

[22] M. Sabou, K. Bontcheva, and A. Scharl. Crowdsourcing Research Opportunities: Lessons from Natural Language Processing. In *Proceedings of the 12th International Conference on Knowledge Management and Knowledge Technologies*, i-KNOW '12, pages 17:1–17:8. ACM, 2012.

[23] M. Sabou, K. Bontcheva, A. Scharl, and M. Föls. Games with a Purpose or Mechanised Labour?: A Comparative Study. In *Proc. of the 13th Int. Conf. on Knowledge Management and Knowledge Technologies (iKNOW)*, pages 1–8, 2013.

[24] M. Sabou, A. Scharl, and M. Föls. Crowdsourced Knowledge Acquisition: Towards Hybrid-genre Workflows. *Int. J. of Semantic Web and Information Systems*, 9(3):14–41, 2013.

[25] C. Sarasua, E. Simperl, and N. F. Noy. CrowdMap: Crowdsourcing Ontology Alignment with Microtasks. In *Proc. of the 11th Int. Conf. on The Semantic Web*, ISWC'12, pages 525–541, 2012.

[26] A. Scharl, M. Sabou, and M. Föls. Climate Quiz: a Web Application for Eliciting and Validating Knowledge from Social Networks. In *Proc. of the 18th Brazilian symposium on Multimedia and the web*, WebMedia '12, pages 189–192. ACM, 2012.

[27] K. Siorpaes and M. Hepp. Games with a Purpose for the Semantic Web. *Intelligent Systems, IEEE*, 23(3):50 –60, 2008.

[28] S. Thaler, E. Simperl, and K. Siorpaes. SpotTheLink: Playful Alignment of Ontologies. In *Proc. of the ACM Symp. on Applied Computing*, pages 1711–1712, 2011.

[29] S. Thaler, E. Simperl, and S. Wölger. An Experiment in Comparing Human-Computation Techniques. *IEEE Internet Computing*, 16(5):52–58, 2012.

[30] T. Tudorache, C. Nyulas, N. F. Noy, and M. A. Musen. WebProtege: A Collaborative Ontology Editor and Knowledge Acquisition Tool for the Web. *Semant. Web J.*, 4(1):89–99, January 2013.

[31] D. Vickrey, A. Bronzan, W. Choi, A. Kumar, J. Turner-Maier, A. Wang, and D. Koller. Online Word Games for Semantic Data Collection. In *Proc. of the Conference on Empirical Methods in Natural Language Processing*, pages 533–542, 2008.

[32] L. von Ahn and L. Dabbish. Designing games with a purpose. *Commun. ACM*, 51(8):58–67, 2008.

[33] L. von Ahn, M. Kedia, and M. Blum. Verbosity: A game for collecting common-sense facts. In *Proc. of the SIGCHI Conference on Human Factors in Computing Systems, CHI*, pages 75–78, New York, NY, USA, 2006. ACM.

[34] J. Waitelonis, N. Ludwig, M. Knuth, and H. Sack. WhoKnows? Evaluating Linked Data Heuristics with a Quiz that Cleans Up DBpedia. *Interact. Techn. Smart Edu.*, 8(4):236–248, 2011.

[35] G. Wohlgenannt, A. Weichselbraun, A. Scharl, and M. Sabou. Dynamic Integration of Multiple Evidence Sources for Ontology Learning. *Journal of Information and Data Management*, 3(3):243–254, 2012.

[36] L. Wolf, M. Knuth, J. Osterhoff, and H. Sack. RISQ! Renowned Individuals Semantic Quiz - a Jeopardy like Quiz Game for Ranking Facts. In *Proc. of the 7th Int. Conf. on Semantic Systems*, I-Semantics '11, pages 71–78, 2011.