(Re)Configuration using Web Data: A case study on the reviewer assignment problem

Technical report

Anna Ryabokon[1], Axel Polleres[2], Gerhard Friedrich[1],

Andreas Falkner[2], Alois Haselböck[2], Herwig Schreiner[2]

[1] Universität Klagenfurt, Universitätsstraße 65-67, A-9020 Klagenfurt

[2] Siemens AG Österreich, Siemensstraße 90, 1210 Wien, Austria

(Re)Configuration using Web Data: a case study on the reviewer assignment problem*

Anna Ryabokon[1], Axel Polleres[2], Gerhard Friedrich[1], Andreas A. Falkner[2], Alois Haselböck[2], and Herwig Schreiner[2]

[1] Alpen-Adria Universität, Klagenfurt, Austria
`firstname.lastname@ifit.uni-klu.ac.at`
[2] Siemens AG Österreich, Vienna, Austria
`firstname.{middleinitial.}lastname@siemens.com`

Abstract. Constraint-based configuration is – on the one hand – one of the classical problem domains in AI and also in industrial practice. Many traditional approaches view configuration as a one-shot process of picking a suitable configuration from a fixed set of previously-known components. Additional problems arise, when configuration objects come from an open environment such as the Web, or in case of reconfiguration, that is, an adaptation of a previously consistent configuration. On the other hand, (re)configuration is a reasoning task very much ignored in the current (Semantic) Web reasoning literature, despite (i) the increased availability of structured data on the Web, particularly due to movements such as the Semantic Web and Linked Data, (ii) numerous practically relevant tasks in terms of using Web data involve (re)configuration. To bridge these gaps, we discuss the challenges and possible approaches for reconfiguration in an open Web environment, based on a practical use case leveraging Linked Data as a "component catalog" for configuration: we have picked the Reviewer Assignment Problem as a classical configuration example, due to the fact that a lot of relevant data for reviewer selection is readily available as Linked Data, and commonly used review management systems apparently do neither make use of these data, nor support reconfiguration properly. In this paper, we present challenges and techniques to enhance existing review management systems with reconfiguration facilities and provide a practical evaluation.

1 Introduction

Constraint-based configuration, i.e. picking and linking a suitable set of components from a component catalog s.t. some predefined constraints are satisfied is a classical problem in AI and also in industrial practice. As users of the Web, we often solve such configuration tasks where in theory the "component catalog" is the Web, for instance as private persons configuring an itinerary (flight, accommodation, hotel, etc.), as human resource manager when looking for suitable candidates on the Web to compose a team or fill a position [1], or as academics, in the task of assigning expert reviewers to papers.

* This work has been developed within the scope of the project RECONCILE (reconciling legacy instances with changed ontologies) and was funded by FFG FIT-IT (grant number 825071).

With the emerging availability of Linked Data on the Web [18], the data for building such component catalogs is increasingly becoming available in a structured format, readily providing a structured “component catalog”, and thus potentially allowing us to apply known configuration techniques to such problems that have been solved manually by Web search. However, configuration is a reasoning task so far largely ignored by the Linked Data and adjacent Semantic Web communities, which mainly focus on taxonomic reasoning and ontologies (RDF Schema, OWL), to better structure Web data or infer implicit Web data. Recently, also non-standard reasoning in the form of probabilistic or statistical approaches is advocated [19, 27], but complex reasoning tasks such as configuration on top of Web data have been largely ignored so far, to the best of our knowledge.

When configuration does not start from scratch, but rather a previously consistent configuration has to be adapted, we speak about reconfiguration. For example, in the reviewer assignment problem, when a PC member drops out unforeseen, or declares a late conflict of interest, the existing configuration (=assignment) needs to be adapted, where typically the degree of change on the original assignment implies some cost. Reconfiguration is an important task in the after-sale life-cycle of configurable products and services, because requirements are changing and there is a need to keep a product or a service up-to-date [7]. A re-engineering organization has to decide which modifications should be introduced to an existing configuration such that the new requirements are satisfied but change costs are minimized. As it has been shown in previous work [9], reconfiguration tasks can be efficiently handled by Answer Set Programming (ASP) [14, 22, 13]. The ASP paradigm has gained much attention over the past decade because it allows modeling and solving of both decision and optimization problems in a declarative way. ASP extends logic programming and includes a predefined modeling language and corresponding solving tools [3].

For this paper, we have chosen the reviewer assignment problem as a showcase for illustrating the feasibility of (re)configuration based on Open Web Data in a practical scenario: The decision if a paper is accepted on the conference depends on reviews made by the program committee. Therefore, it is required to assign every paper to a number of reviewers such that on one hand these reviewers are interested in reading the paper and on the other hand have enough expertise.

A number of conference management systems are available: EasyChair[3], ConfMaster[4], Linklings[5] and others which support various features on automatically assigning papers based on bids, etc. However, none of these so far assists the chairs in finding expert reviewers, despite the availability of rich data on the Web that could help: citation indexes such as DBLP[6] as well as conferences such as ISWC and ESWC offer their metadata (including publications, authors, data about past program committees), etc. as Linked Data since several years.[7] This data could be readily used to support the configuration task of PC/reviewer assignment automatically matching well-suited reviewers

[3] `http://www.easychair.org/`
[4] `http://www.confmaster.net/`
[5] `http://www.linklings.com/`
[6] `http://dblp.l3s.de/d2r/`
[7] `http://data.semanticweb.org/`

to papers, e.g. by structured queries using the SPARQL query language [28] that narrow down a set of relevant candidate reviewers for a particular paper.

Likewise, current conference management systems – to the best of our knowledge – do not offer support for changes in the assignment over time. For instance, if a reviewer drops out, her papers should be redistributed among the remaining reviewers. This process can be viewed as a reconfiguration of the paper assignment. In the current paper, we will demonstrate how reviewer assignment and re-assignment tasks can be supported by leveraging Open Data and deploying methods of reconfiguration. We will use SPARQL and ASP as concrete formalisms to support these tasks.

The paper is structured as follows: in Section 2 we describe the reviewer assignment problem. Then, we elaborate on how to obtain relevant data for finding suitable reviewers from the Web using Linked Data and SPARQL in Section 3. In Section 4 we present a definition of the reconfiguration problem and exemplify different reconfiguration scenarios. Section 5 provides a description of test instances and evaluation results. And finally, we discuss the related work in Section 6 and conclude in Section 7.

2 The reviewer assignment problem

The reviewer assignment problem can be viewed as a configuration task where papers must be linked to reviewers such that a set of problem specific constraints are fulfilled. Furthermore, preferences among solutions are expressed based on an optimization function which ranks the set of valid reviewer/paper assignments (i.e. configurations).

For the encoding of the problem instances we have chosen a logic based formalism. The input of the Reviewer Assignment Problem can be represented using the following predicates: The set of reviewers and papers are given by `reviewer(revId)` and `paper(papId)` facts where `revId` and `papId` represent reviewer and paper identifiers.

In addition, reviewers typically specify their preferences in a process of bidding on the one hand, and on the other hand papers should be reviewed by the most competent reviewers among the program committee (PC). Whereas bidding preference are usually collected by a conference management system, the "expertise match" between reviewers and papers is normally not given explicitly and has to be estimated by program or area chairs while assigning the papers in existing systems, if it is taken into account at all.[8] We will elaborate more on how Semantic Web data can be leveraged to assist this task in Section 3. In particular, the facts `paperExp(papId,revId,exp)` express the expertise `exp` of reviewers with respect to papers. We represent the expertise of a reviewer using the categories:

0 *conflict* if a reviewer is an author of the paper or biased by some other circumstances
1 *low* expertise
2 *moderate* expertise
3 *high* expertise

[8] Some systems like ConfMaster allow reviewers to choose a number of keywords they feel interested/expert in, but these are mainly used for filtering papers by keywords during bidding, rather than being an objective assessment of the reviewer's expertise.

The predicate `reviewerBid(revId,papId,bid)` expresses the preferences of reviewers for papers provided by the bidding process. The bids of the reviewers are encoded as:

0 denotes an explicit *conflict* of interest declared by a reviewer
1 indicates *indifference*, i.e. no bid is provided
2 corresponds to the *weak* willingness to review, e.g. "I can review" in EasyChair
3 expresses *strong* willingness to review the paper, e.g. EasyChair's bid "I want to review"

The output of the reviewer assignment problem is an assignment of papers to reviewers expressed by facts `assign(papId, revId)`. To summarize, reviewers express preferences for papers and papers express preferences for reviewers, where the goal is to find a match between these two parties such that different preferences are reconciled. Goldsmith and Sloan [15] propose to view this problem as a variant of the well known stable marriage problem.

Definition 1. *Given n men and n women as well as a ranking list for each person with unique numbers between 1 and n in the order of preference about persons they want to marry. The goal is to find an assignment (match) where each man is married to exactly one woman such that there are no people of opposite sex who would prefer rather to marry other than their actual partners. Otherwise, the marriage is unstable. [10]*

In other words, a paper/reviewer assignment (marriage) is stable if there does not exist an alternative assignment `assign(papId, revId)` in which `papId` *and* `revId` are individually better off than in their current assignment. Consequently, a reviewer cannot spot a paper which she prefers more and for which she has more competence compared to the current assignments.

There are several variants of the stable matching which differ from the classic stable marriage problem:

Polygamy Men and women can marry more than one person.
Incomplete Lists Some men or women refuse a marriage to particular partners.
Indifference The preferences express a preset number of preference equivalence classes.

Each variation of the Stable Marriage Problem mentioned above on its own can be solved in polynomial time [15]. The problem becomes more complicated and is known to be NP-hard if both incomplete lists and indifference occur [23] even for the one-to-one assignment (monogamy).

In our case papers are assigned to several reviewers and vice versa. Furthermore, some assignments are forbidden due to conflicts of interest. Finally, reviewers do not provide a strict order on all papers but associate papers to predefined preference classes. Consequently, the paper assignment variant of the stable marriage problem corresponds to polygamy with indifference and incomplete preference list. Therefore, a problem solving method which is able to deal with NP-hard problems is required and justifies the usage of ASP as a problem representation and solving framework.

Note, that requiring stability as a hard constraint may increase the unbalance in the workload of reviewers. Therefore, we consider stability as a soft constraint and minimize the number of assignments which do not fulfill the stability property.

Moreover, the further hard constraints are: (1) each paper must be assigned to a fixed number of reviewers and (2) fairness of the workload should be achieved. In order to distribute the papers among the reviewers as uniformly as possible, we add a *balancing criterion* as a hard constraint, which limits the minimum and maximum number of papers assigned to each reviewer. We use a specific preprocessing step to identify upper and lower bounds by iteratively running a solver. First we set upper and lower bounds equal to $[avg, avg]$, where avg is the average number of papers per reviewer rounded to the next higher integer. We relax subsequently the lower and then the upper bound until a solution exists. Consequently, we have found boundaries where the variance of the individual workload is as small as possible.

In addition to the stability we can optimize the solution according to the "satisfaction" of papers and reviewers. In particular, the more competent reviewers are assigned to papers the more satisfied are the papers. Conversely, the more reviewer bids are satisfied the more satisfied are the reviewers.

The goal is to assign papers to reviewers with respect to the following optimization criteria which are ordered according to their priority starting with the most important one. The following priorities favor the happiness of papers.

P1 Minimize assignments of papers to reviewers with "low" expertise.
P2 Minimize assignments of papers to reviewers who have only "moderate" expertise on a paper.
P3 Minimize assignments of papers to reviewers who did not provide any preferences for these papers, i.e. "indifference" bid.
P4 Minimize assignments of papers to reviewers who indicated only weak willingness to review them, i.e. "can review" bid.
P5 Minimize the number of assignments which are not stable.

These priorities and any other priority order can be easily encoded within the ASP framework (see Section 4 below).

Consider a small example illustrating the approach described above. Assume there are 6 papers submitted to a conference, which program committee includes 6 members, who provided the bids presented in Table 1. Each paper should be reviewed three times, therefore 18 reviews are required with an average number of papers per reviewer $avg = 3$. Iterative relaxation of lower and upper bounds allows us to obtain the solution presented in Figure 1 with $[avg - 1, avg + 1]$ balancing bounds.

	Reviewers					
Papers	pc1	pc2	pc3	pc4	pc5	pc6
p1	0/0	1/3	3/2	1/1	1/1	2/1
p2	2/1	2/3	1/0	3/2	1/1	1/1
p3	1/3	0/0	0/0	0/2	2/2	3/1
p4	2/3	2/3	1/2	0/0	0/0	1/1
p5	2/3	1/3	0/0	0/2	1/2	1/1
p6	2/3	0/1	1/0	1/2	1/3	0/0

Table 1. Table of paper expertise/reviewer preferences.

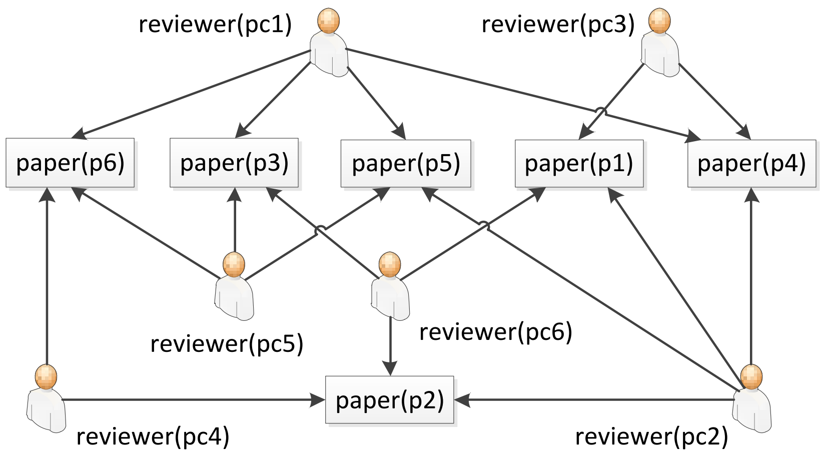

Fig. 1. Configuration solution

3 Leveraging Linked Data for finding candidate reviewers, possible conflicts, and assessing expertise

In a reviewing system, on the one hand, we can assume that papers and abstracts (typically at least the abstracts are available at bidding time) along with their authors, as well as the set of candidate reviewers (program committee) are available. On the other hand, citation indexes such as DBLP[9] as well as conference-metadata are available at `data.semanticweb.org` and provide insights about past publications at conferences in the form of Linked Data.

Linked Data [18], roughly, is a collection of RDF [24] statements, often called "triples" published in various sites and possibly linked through the common use of URIs in these triples, that can be dereferenced, pointing to other datasets. Structured queries on datasets published using the standard RDF query language SPARQL can be used to extract relevant data. For lack of space, we refer to the respective W3C specifications on details about RDF [24] and SPARQL [28].

For instance, the Linked Data extract of DBLP contains RDF statements[10] indicating that Jim Hendler recently co-authored a paper with Tim Berners-Lee:

```
<dblp.uni-trier.de/rec/bibtex/journals/ai/HendlerB10>
   foaf:maker
      dblpperson:Jim_Hendler ,
      dblpperson:Tim_Berners-Lee .
```

This data is complemented by `data.semanticweb.org` containing information about abstracts of papers from conferences pertinent to the Semantic Web field,

[9] `http://dblp.l3s.de/d2r/`

[10] in Turtle [2] RDF syntax, where the URI prefixes `dswperson:` and `dblpperson:` stand for `http://data.semanticweb.org/person/` and `http://dblp.l3s.de/d2r/resource/authors/`, resp.; for other prefixes used, cf. `http://prefix.cc`

e.g. about Jim Hendler, his papers in these conferences, his affiliation and roles like PC member:

```
<data.semanticweb.org/conference/www/2010/paper/main/213>
    dc:creator dswperson:james-hendler .
    swrc:abstract "The Semantic Web community, until now,
                  has used traditional database systems
                  for the storage and querying of RDF
                  data. [...]" .
...
<www.rpi.edu>
    foaf:member dswperson:james-hendler ,
               dswperson:deborah-mcguinness .
...
```

Additionally, there is significant research ongoing on cross-linking datasets in linked data by triples such as:

```
dswperson:james-hendler owl:sameAs dblpperson:Jim_Hendler .
```

These "linking" triples either have to be provided by publishers, or the linkage-task can be partially automated, for instance – particularly mentioning the example of linking DBLP and `data.semanticweb.org` – cf. [16].

From this data we can extract valuable information about connections between authors, such as recent co-authorship or joint affiliation, which would recognize conflicts of interests automatically, or create expertise profiles from abstracts of past papers, i.e. compare abstracts or keywords of published papers to submissions, in order to determine the level of expertise.

For a proof-of-concept implementation we have selected a fictitious set of reviewers composed of persons mentioned at `data.semanticweb.org`, as well as a subset of papers mentioned there as fictitious set of submissions. We also retrieve information about recent co-authorship from `http://dblp.l3s.de/d2r/`. For linking between DBLP and data.semanticweb.org, at this point we only link authors with unambiguous unique names present in both datasets.

We use the set of abstracts of a reviewer's published papers, extracted from SPARQL queries to `data.semanticweb.org` as an indicator of her expertise. Conversely, the abstracts of a submitted paper is used to indicate the required expertise to evaluate the paper. We make the reasonable assumption that the more similar the paper abstract and the abstracts of a reviewer are, the more competent the reviewer is to evaluate the paper. In order to compute these similarities we apply established methods from information retrieval and recommender systems [21]. First the abstracts of papers and reviewers are analyzed to derive a list of relevant keywords. This is achieved by considering only those terms which are provided by the PC chair of a conference in form of keywords. Next, we clean the keywords by employing a lemmatizer such as `http://morphadorner.northwestern.edu/`. The result of this process is a term vector for each reviewer and each paper, which we use in a standard *term frequency – inverse document frequency* (TF/IDF) weighting of the paper's abstract as well as of the union of abstracts for each reviewer. The similarities of vectors describing the papers and vectors describing the reviewers are computed by the *cosine similarity measure* [21]. For a paper the top 5% most similar reviewers are considered to have high expertise. The next 5% of reviewers are associated with moderate expertise. All other reviewers have low expertise. In addition, if a *conflict of interest* is detected, the expertise value is set to 0; as for such conflicts of interest, we check whether (i) the reviewer is herself an author of a paper, (ii) whether we detect recent co-authorship, i.e. within the last 5 years, or, finally, (iii) whether author and PC member share the same affiliation, all from data within `data.semanticweb.org`. The retrieved data describing co-authorship using the following query:

```
SELECT DISTINCT ?Person ?Person2
WHERE { SERVICE  <http://data.semanticweb.org/sparql>
    {
      SELECT DISTINCT ?Person ?Person2 ?Year
      {
        ?R a swc:ProgrammeCommitteeMember ; swc:isRoleAt
             <http://data.semanticweb.org/conference/iswc/2011> .
        ?Person swc:holdsRole ?R .
        [ dc:creator ?Person, ?Person2; swrc:year ?Year ]
        FILTER (?Person != ?Person2 && 2012-xsd:integer(?Year) < 5)
      }
      ORDER BY ?Person ?Person2
    }
}
```

We note that co-authorship could also be determined from other datasets, such as DBLP, possibly covering a wider range of publication venues, but since their are no explicit links between DBLP and `data.semanticweb.org` in the currently published data, for the moment we do not make use of these additional conflicts; as mentioned above, link discovery methods, such as for instance those in [16] could be used to bridge this gap. As for the assessment of expertise, we only use the paper abstracts at the moment, more sophisticated methods based on full-text could be deployed, but are out of scope for our current work.

4 (Re)configuration

Based on (i) data gathered from `data.semanticweb.org` via SPARQL queries and (ii) the expertise scores computed from this data following the approach described the previous Section, we can encode the reviewer configuration problem as an ASP program, following the methodology from [9].[11] ASP is a formalism flexible enough to encode multiple optimization criteria **P1**-**P5** mentioned in Section 2. An example of successful application of ASP with multi-criteria optimization for configuration tasks can be found in [12].

The output of the ASP program is a configuration of reviewers and papers. However, requirements might change over time demanding changes of a legacy configuration. For instance, in the reviewer assignment problem, reviewers may drop out, papers could be withdrawn, or additional conflicts of interests may be discovered. Reconfiguration is

[11] We omit details on the queries and encoding here for space restrictions, the respective SPARQL queries and ASP encoding can be found at `https://sites.google.com/site/reviewersevaluation/`. ASP encodings for (re)configuration and optimization require roughly (25) 15 clauses.

a transformation of a legacy configuration into a target one such that all current requirements are satisfied. In our application case the legacy configuration is described by the set of papers, reviewers and paper/reviewer assignments. The transformation of the legacy configuration possibly requires that some of its parts are deleted. Therefore, a new configuration problem instance is generated including the current requirements and transformation knowledge regarding reuse or deletion of parts of a legacy configuration. We employ the modeling patterns described in [9] to formulate a reconfiguration problem instance. The principle idea is, that for every element of the legacy configuration a decision has to be made whether or not to delete or reuse this element. The reused elements are complemented on demand by addition of new elements in order to fulfill all requirements. For instance, in case a reviewer drops out her assignments to papers must be deleted, eventually new assignments are added to other reviewers. If changes happen in the middle of the reviewing process and some papers are already reviewed by PC members, we can declare that these paper/reviewer assignments must be reused for the reconfiguration.

Since we are interested in an optimized reconfiguration solution, costs can be associated to the transformation depending on whether elements are deleted, reused or created. These costs are exploited in an objective function for optimization. For our reconfiguration problem the new sets of papers and reviewers are predefined, so no decision regarding deletion or reuse of these elements has to be made, whereas paper/reviewer assignments may be reused or deleted and possibly new assignments must be created. Consequently, three types of reconfiguration costs can be distinguished:

Creation costs for reviewer to paper assignments absent in the legacy configuration;
Reuse costs for the assignments present in both reconfiguration and legacy configuration;
Deletion costs for the assignments of legacy configuration absent in the reconfiguration.

These domain-specific costs may be defined as needed in order to reflect the preferences of the PC chair. All reconfiguration costs are minimized to obtain the preferred solution. This optimization criterion **PR** extends the set of optimization criteria mentioned in Section 2 and possesses the highest priority level. In our application case we assume that both the *reuse* of a paper/reviewer assignment as well as its *deletion* have zero costs; i.e., we assume that reviewers are satisfied if their workload is not changed or even reduced. However, for creating new assignments some costs are associated.

We take the configuration solution from the previous section as a legacy configuration and illustrate some reconfiguration scenarios which may occur in practice (for resp. ASP encodings cf. footnote 11).

1. *Late conflict of interest*. Assume the conference submission stage is over, all papers are assigned to reviewers fulfilling all requirements described in Section 2 and a number of reviewers declare late conflicts of interest with papers assigned to them. For example, the program committee member pc5 declares a conflict with the paper p5 assigned to her in a legacy configuration. The reconfiguration process eliminates the existing inconsistent match and assigns p5 to another reviewer. An optimal reconfiguration solution is presented in Figure 2.

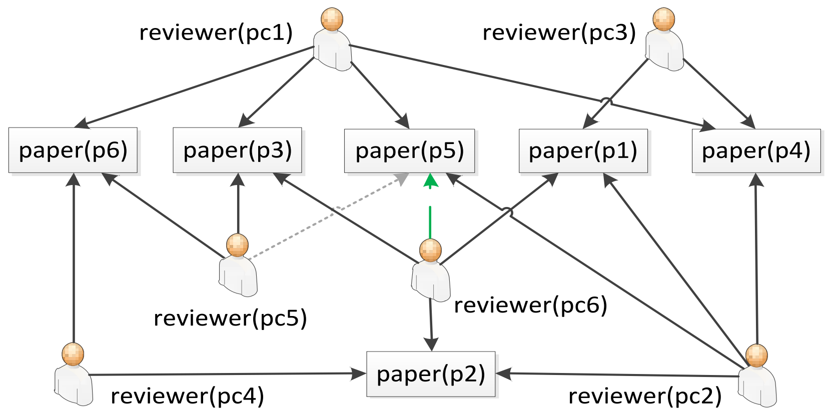

Fig. 2. Reconfiguration solution (scenario 1)

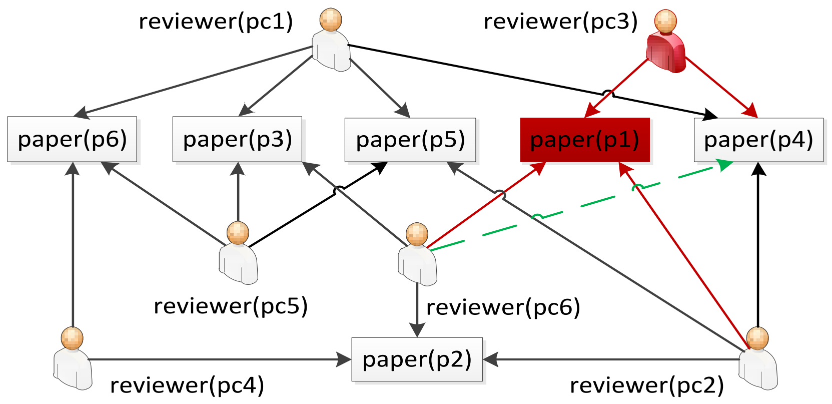

Fig. 3. Reconfiguration solution (scenario 2)

2. *A reviewer drops out and authors withdraw a paper*. In our example we require reviewer pc3 and paper p1 are excluded from the target configuration. Moreover, all assignments including these individuals must be removed as well. All resulting unassigned papers are reassigned to the remaining reviewers in order to fulfill the requirements of the problem. In Figure 3 the sample reconfiguration for the mentioned scenario in provided.
3. *Late conflict of interest, a reviewer drops out and authors withdraw a paper*. The last scenario combines the first and the second cases. Reviewer pc5 declares a late conflict of interest with paper p5, reviewer pc3 drops out and authors withdraw their paper p2. An optimal reconfiguration solution is presented in Figure 4.

Given their simplicity the reconfiguration solutions of the provided examples are straight forward. However, in the real-world cases presented in the evaluation we observed that complex reassignments are required to fulfill all the requirements.

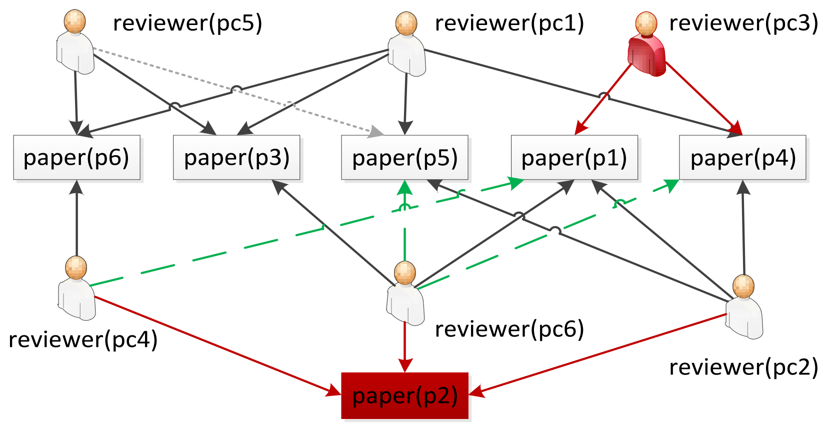

Fig. 4. Reconfiguration solution (scenario 3)

5 Evaluation

To simulate a conference bidding process we generated a set of reviewer bids in addition to the paper expertise relation presented in Section 3. For each reviewer r_i we ordered the set of papers by their similarity to reviewer r_i. As for "emulating" bidding, from the top 20% of the papers we randomly selected 3 to 7 papers and assigned to reviewer r_i a *strong willingness* (see Section 2) to review theses papers. 4 to 8 bids of the type *weak willingness* were chosen from the next 30% of the papers and 0 to 5 conflicts were generated from the last 50%. Finally, we assign a bid of type *indifference* to all remaining papers. Authorship conflicts are taken into account during the conflict generation.

The data retrieved from the Web was used to generate configuration instances of different size in terms of numbers of papers and reviewers. Each configuration instance was solved using the approach presented in Section 2. Obtained solutions were then used as legacy facts in reconfiguration instances. Moreover, each instance was extended with additional facts describing reconfiguration changes such as the deletion of a paper and a reviewer as well as the declaration of a late conflict of interest. In our evaluation we differentiated between 4 possible cases: cases 1-3 are described in the previous section and the case 4 corresponds to the situation when a number of late conflicts of interest were declared and some of the reviewers already provided the reviews. The latter means that some of the made assignments cannot be changed, i.e. they have to be reused in a reconfiguration solution. Thus, for the first case we added up to 3% of assignments as late conflicts of interests. In the second case from 3 to 10% of all reviewers dropped out and papers were withdrawn. The same settings were used in the third case, which is a combination of the previous two. In the fourth case we declared that 5-10% of assignments have to be reused and provided 1-2% of late conflicts of interest. The instances were generated in such a way that the optimal reconfiguration costs for each instance was known prior to the experiment.

Instance	Configuration								Reconfiguration									
	avg	bounds		Optimization criteria					Reconf.	avg	bounds		Optimization criteria					
	papers	min	max	P1	P2	P3	P4	P5	case	papers	min	max	PR	P1	P2	P3	P4	P5
0p70r45	5	4	5	115	66	162	13	96	1	5	4	5	4	114	66	161	13	95
1p70r45	5	4	5	128	59	169	10	80	3	5	4	5	23	132	50	160	10	70
0p90r65	5	4	5	136	72	172	10	109	3	5	4	5	27	146	62	178	7	100
1p90r65	5	4	5	143	62	178	11	101	1	5	4	5	5	144	62	178	11	102
0p110r57	6	5	6	209	68	267	15	123	1	6	5	6	6	210	67	266	16	123
1p110r57	6	5	6	186	87	268	22	108	2	6	5	6	21	175	77	246	19	104
0p150r81	6	5	6	242	80	311	18	93	2	6	5	6	28	235	70	295	14	97
1p150r81	6	5	6	247	83	307	11	89	3	6	5	6	44	251	64	306	10	86
0p210r114	6	5	6	323	136	408	30	137	3	6	5	6	67	334	110	411	20	142
1p210r114	6	5	6	333	98	417	35	85	1	6	5	6	12	334	99	420	36	88
0p225r130	6	5	6	360	128	414	28	75	1	6	5	6	13	362	127	414	28	76
1p225r130	6	5	6	356	119	409	30	70	2	6	5	6	33	352	114	408	28	72
0p270r147	6	5	6	414	136	553	20	108	1	6	5	6	12	417	134	554	20	110
1p270r147	6	5	6	402	148	518	37	69	3	6	5	6	**82**	420	113	495	30	97
0p300r163	6	5	6	463	174	608	42	118	2	6	5	6	**90**	467	155	575	39	156
1p300r163	6	5	6	465	172	610	35	102	4	6	5	6	11	467	171	610	36	105

Table 2. Evaluation results for configuration and reconfiguration scenarios of the reviewer assignment problem. In the instance name *TpPPPrRRR* the number *T* distinguishes between two test cases, *PPP* is the number of papers and *RRR* is the number of reviewers.

The evaluation results[12] presented in Table 2 show that the reasoner was able to find a solution for all test instances. In both configuration and reconfiguration cases the algorithm started the evaluation with bounds for the number of paper assignments per reviewer as described in Section 2. The resulting balancing bounds indicated in Table 2 were obtained with a timeout set to 180 seconds for checking the balancing criterion. If for given bounds the solver does not provide a solution, the bounds are relaxed. In all cases the upper bound corresponds to the average number of papers per reviewer. For the depicted balancing bounds we obtained the best configuration solutions that can be computed within a timeout period of 900 seconds; proving optimality for such (re)configuration instances seems to be infeasible in practice. The number of violations of each optimization criteria **P1**-**P5** mentioned in Section 2 for each best solution is indicated in Table 2. Note that, the performed experiments have realistic number for PC members and submissions comparable with e.g. the last ISWC conferences from which we took the data. For the reconfiguration problem instances the solver was able to find solutions with optimal reconfiguration costs **PR** in all but the two biggest cases `1p270r147`, and `0p300r163`. A solution with optimal reconfiguration costs was usually identified by the solver in the first 10 seconds of the solving process excluding the grounding time. For the two cases mentioned above (showed in bold), the solver found solutions which reconfiguration costs are 20% and 8% higher than the optimum. The obtained results show that the proposed method is feasible for realistic reviewer assignment problems.

[12] The evaluation experiments were performed using Potassco ASP collection (gringo-3.0.3 and clasp-2.0.4) `http://potassco.sf.net` on a system with Intel i7-3930K CPU (3.20GHz), 32Gb of RAM and running Ubuntu 11.10.

6 Related work

The problem of assigning paper submissions to reviewers is known widely and studied in detail by several researchers mainly from two perspectives. One branch deals with the automatic generation of the relations between reviewers and papers. The second branch investigates methods for computing the assignment between reviewers.

For the automatic generation of the expertise relation, content-based recommendation techniques were applied. Such methods are using different classification algorithms such as latent semantic indexing [6] or vector space models [30]. We leverage this approach by exploiting Linked Data and Semantic Web technology to extract data which we use to compute the expertise relation based on a vector space model. In addition, the reviewers bids might be extended by recommender system techniques [4] which is out of the scope of this paper.

The proposals for the automatic computation of the assignment relation employ different strategies depending on the constraints. The most popular problem solving algorithms are based on network flow models [17, 15] or (mixed) integer programming, e.g. [11]. However, these approaches assume just one weighted relation between papers and reviewers (either expertise or bids) with the exception of [15]. This work considers both relations and proposes the application of the stable marriage property. Flach et al. [8] suggest an approach to RAP that is similar to ours, except bid initialization, i.e. our system works as an add-on of a conference management system that does not changes its behavior. However, such initialization could improve our results as well, since the opinion of a reviewer is influenced by a recommendation and becomes biased to the scores stored in the system. The score calibration method described in [8] helps PC chairs to make a final decision on a paper and is out of scope our work.

Note that, in none of the works mentioned above the reconfiguration of reviewer/paper assignments is considered. In the ASP framework we could model these problems succinctly. Basically, one could view reconfiguration as a form of belief revision or updates of a knowledge-base (e.g. [5, 29]), since facts about the legacy configuration must be revised. The central idea of belief revision is to apply operators to a knowledge-base and to define the semantics of these operators either based on syntactical characterizations, such as defining preferred changes of the axioms of the knowledge-base, or by criteria based on logical models. In reconfiguration we assume that the new knowledge-base is consistent and therefore adding a legacy configuration never leads to an inconsistent knowledge-base because in the worst case the complete legacy configuration can be deleted. Consequently, a central point of belief revision, i.e. dealing with inconsistent updates, is not present in our domain. Furthermore, in belief revision the goal is to design general change operators based on some first principles (e.g. the AGM postulates) whereas in reconfiguration the knowledge engineer specifies by logical descriptions and a cost function which changes to the legacy configuration are allowed and which changes are preferred.

7 Conclusion

We have argued for and illustrated the feasibility of complex (re)configuration tasks based on Semantic Web Data for a practical use case. We strongly believe that, in order to leverage Linked Data/Semantic Web Data focusing only on taxonomic/classification reasoning (DLs/OWL) alone is not sufficient and especially (re)configuration is a good example for a practical reasoning task occurring in many potential applications of Web Data usage. In particular, in the area of the reviewer assignment problem the tasks of configuration and reconfiguration for various optimization criteria must be modeled and solved. As we have shown, non-standard reasoning techniques such as ASP can be deployed to solve such tasks with reasonable effort and performance in realistic examples. Particularly, ongoing work on providing interfaces for Semantic Web languages and technologies such as RDF, and SPARQL for ASP engines [26, 20, 25][13] promises readily available tool-sets to address such use cases.

References

1. Aleman-Meza, B., Bojārs, U., Boley, H., Breslin, J., Mochol, M., Nixon, L., Polleres, A., Zhdanova, A.: Combining RDF vocabularies for expert finding. In: Proceedings of the 4th European Semantic Web Conference (ESWC2007). LNCS, vol. 4519, pp. 235–250 (2007), http://www.polleres.net/publications/alem-etal-2007.pdf
2. Beckett, D., Berners-Lee, T.: Turtle - Terse RDF Triple Language (2008), http://www.w3.org/TeamSubmission/turtle/
3. Brewka, G., Eiter, T., Truszczynski, M.: Answer set programming at a glance. Communications of the ACM 54(12), 92–103 (2011)
4. Conry, D., Koren, Y., Ramakrishnan, N.: Recommender systems for the conference paper assignment problem. In: Proceedings of the third ACM conference on Recommender systems. pp. 357–360 (2009)
5. Delgrande, J.P., Schaub, T., Tompits, H., Woltran, S.: Belief revision of logic programs under answer set semantics. In: Proceedings of the 11th International Conference on Principles of Knowledge Representation and Reasoning, KR 2008. pp. 411–421 (2008)
6. Dumais, S., Nielsen, J.: Automating the assignment of submitted manuscripts to reviewers. In: Proceedings of the 15th Conference on Research and Development in Information Retrieval. pp. 233–244 (1992)
7. Falkner, A., Haselböck, A.: Challenges of Knowledge Evolution in Practice. In: Workshop on Intelligent Engineering Techniques for Knowledge Bases (IKBET 2010). pp. 1–5 (2010)
8. Flach, P.A., Spiegler, S., Golénia, B., Price, S., Guiver, J., Herbrich, R., Graepel, T., Zaki, M.J.: Novel tools to streamline the conference review process: experiences from sigkdd'09. SIGKDD Explorations 11(2), 63–67 (2009)
9. Friedrich, G., Ryabokon, A., Falkner, A., Haselböck, A., Schenner, G., Schreiner, H.: (Re)configuration using Answer Set Programming. In: Proceedings of the IJCAI 2011 Workshop on Configuration. pp. 17–25 (2011)
10. Gale, D., Shapley, L.: College admissions and the stability of marriage. The American Mathematical Monthly 69(1), 9–15 (1962)
11. Garg, N., Kavitha, T., Kumar, A., Mehlhorn, K., Mestre, J.: Assigning Papers to Referees. Algorithmica 58(1), 119–136 (2010)
12. Gebser, M., Kaminski, R., Kaufmann, B., Schaub, T.: Multi-criteria optimization in answer set programming. In: ICLP (Technical Communications). pp. 1–10 (2011)

[13] See also the dlvhex ASP engine's Semantic Web plug-ins project at http://sourceforge.net/projects/dlvhex-semweb/

13. Gebser, M., Kaufmann, B., Schaub, T.: Conflict-driven answer set solving: From theory to practice. Artif. Intell. 187, 52–89 (2012)
14. Gelfond, M., Lifschitz, V.: The stable model semantics for logic programming. In: 5th International Conference and Symposium on Logic Programming. pp. 1070–1080 (1988)
15. Goldsmith, J., Sloan, R.: The AI conference paper assignment problem. In: Proceedings of AAAI Workshop on Preference Handling for Artificial Intelligence. pp. 53–57. Vancouver, BC, Canada (2007)
16. Groza, T., Dragan, L., Handschuh, S., Decker, S.: Bridging the gap between linked data and the semantic desktop. In: 8th International Semantic Web Conference (ISWC2009) (2009)
17. Hartvigsen, D., Wei, J., Czuchlewski, R.: The conference paper-reviewer assignment problem*. Decision Sciences 30(3), 865–876 (1999)
18. Heath, T., Bizer, C.: Linked Data: Evolving the Web into a Global Data Space, Synthesis Lectures on the Semantic Web: Theory and Technology, vol. 1. Morgan & Claypool (2011)
19. Hitzler, P., van Harmelen, F.: A reasonable semantic web. Semantic Web Journal 1(1), 39–44 (2010)
20. Ianni, G., Krennwallner, T., Martello, A., Polleres, A.: Dynamic querying of mass-storage rdf data with rule-based entailment regimes. In: Proceedings of the 8th International Semantic Web Conference (ISWC 2009). Lecture Notes in Computer Science (LNCS), vol. 5823, pp. 310–327. Washington DC, USA (2009)
21. Jannach, D., Zanker, M., Felfernig, A., Friedrich, G.: Recommender Systems: An Introduction. Cambridge University Press (2010)
22. Leone, N., Pfeifer, G., Faber, W., Eiter, T., Gottlob, G.and Perri, S., Scarcello, F.: The DLV system for knowledge representation and reasoning. ACM Transactions on Computational Logic (TOCL) 7(3), 499–562 (2006)
23. Manlove, D., Irving, R., Iwama, K., Miyazaki, S., Morita, Y.: Hard variants of stable marriage. Theoretical Computer Science 276(1-2), 261–279 (2002)
24. Manola, F., Miller, E.: RDF Primer. W3C Recommendation, W3C (2004), `http://www.w3.org/TR/2004/REC-rdf-primer-20040210/`
25. Obermeier, P., Marano, M., Polleres, A.: Processing RIF and OWL2RL within DLVHEX. In: Web Reasoning and Rule Systems – Fourth International Conference, RR 2010. LNCS, vol. 6333, pp. 244–250 (2010)
26. Polleres, A.: From SPARQL to rules (and back). In: Proceedings of the 16th World Wide Web Conference (WWW2007). pp. 787–796 (2007), `http://www2007.org/paper435.php`, extended technical report version available at `http://www.polleres.net/TRs/GIA-TR-2006-11-28.pdf`, slides available at `http://www.polleres.net/publications/poll-2007www-slides.pdf`
27. Polleres, A., Hogan, A., Harth, A., Decker, S.: Can we ever catch up with the web? Semantic Web – Interoperability, Usability, Applicability 1(1-2), 45–52 (2010), `http://www.semantic-web-journal.net/sites/default/files/swj36_0.pdf`
28. Prud'hommeaux, E., Seaborne, A.: SPARQL Query Language for RDF. W3C Recommendation, World Wide Web Consortium (2008), `http://www.w3.org/TR/rdf-sparql-query/`
29. Slota, M., Leite, J.: On semantic update operators for answer-set programs. In: Proceedings of 19th European Conference on Artificial Intelligence, ECAI 2010. pp. 957–962 (2010)
30. Yarowsky, D., Florian, R.: Taking the load off the conference chairs: Towards a digital paper-routing assistant. In: Proceedings of the Joint SIGDAT Conference on Empirical Methods in NLP and Very-Large Corpora. pp. 90–99 (1999)