Inter-Organizational Case Assignment Based on Agent Attributes and Functions

Dominik Reichelt, Alexander Lawall and Thomas Schaller
University of Applied Sciences Hof
Institute for Information Systems
95028 Hof, Germany
Email:{firstname.lastname}@iisys.de

***Abstract*—This contribution describes an approach that assigns arriving documents in an organization to agents. The assignment is based on the attributes and functions, i.e. positions and roles, of the agents. These attributes and functions are contained in an organizational model that includes all potential recipients. Using algorithms based on this model, the assignment of the documents works without machine learning methods. It can be regarded as non-probabilistic classification of the documents, with the agents serving as labels. These labels are not just denominators, but are also interconnected via the organizational model.**

I. Introduction

Organizations are faced with a high number of documents that are received. Often, such documents are not addressed to specific agents within the organization, but to the organization as a whole. High amounts of work are required to find out *if* a specific document is relevant for the organization, and if so, *which agents* are the most appropriate recipients.

Documents arriving in the organization can take different forms:

- Structured electronic forms, e.g. from web portals or via defined data interchange standards, as described in [1]
- Case-related documents, e.g. correspondence tagged with a case identifier
- Unstructured correspondence, e.g. product requests

The first two categories of documents can be handled with relative ease. The communication partners have already agreed upon standards for communication (in the case of structured documents), or already have the responsible agents assigned (the stakeholders of a specific case).

Unstructured correspondence, however, does not have any pre-defined responsible agents or processes that they belong to. Consequently, agents that are not addressed specifically by the external sender may miss important information, while other agents have to handle irrelevant information.

This problem is increased as external organizations do not necessarily have enough information on the internal organizational structure to address the correct agents. This can be alleviated by using a propagation approach as described in [2]. This may not always be feasible, however, as it relies on the exchange of organizational data between the two organizations beforehand.

In principle, this problem can be described as classification problem. According to [3, p.6], one aim of classification is

> *"(. . .) to establish a rule whereby we can classify a new observation into one of the existing classes"*.

As opposed to the approaches discussed in [3], the approach described in this contribution is not based on statistical methods (i.e. "Supervised Learning") solely based on observations, but is based on a concrete reference model. This reference model represents the organizational structure. The observations (i.e. documents) are mapped to this reference model.

In the terminology of machine learning, the *classes* (also called labels) are elements of the reference model. More precisely, the classes are concrete agents that are part of the organization. The actual assignment (i.e. classification) is based on attributes and relations within this reference model.

This use of an additional reference model (which is not generated by "learning") makes the approach somewhat similar to recommender systems. As opposed to recommender systems, however, the approach is not based on "profile(s) of interests" (cf. [4]) that are trained from recommendations provided by people, but on a reliable organizational model.

The organizational model is limited to organizational and domain-specific attributes. Technical attributes and groups, as can be found in widely employed directory servers, e.g. LDAP (cf. [5], [6]) servers, are no criteria for the assignment.

Outline

This paper is structured as follows. First, a representative scenario is introduced. It consists of an organizational reference model and an example document. Section III then formally introduces the core ideas and algorithms used by the approach. The algorithms are also illustrated based on the example introduced in section II. The section also discusses

In: W. Schmidt, A. Fleischmann, L. Heuser, A. Oberweis, F. Schönthaler, C. Stary, and G. Vossen (Eds.): Proceedings of the Workshop on Cross-organizational and Cross-company BPM (XOC-BPM) co-located with the 17th IEEE Conference on Business Informatics (CBI 2015), Lisbon, Portugal, July 13, 2015.

the impact of explicitly expressing knowledge within the organizational model. Section IV concludes with further areas of research.

II. Representative Scenario

In order to exemplify the problem, we introduce a representative scenario for inter-organizational case assignment. The scenario is comprised of a model that represents a concrete organization. All agents that can be assigned a new case are part of this organization, and consequently included in the model. In the context of this contribution, we consider the incoming documents as the starting point of a *case*. According to [7]:

> *"A case is a contextualized piece of knowledge representing an experience that teaches a lesson fundamental to achieving the goals of the reasoner."*

We regard the incoming document as the initial context of the case that needs to be assigned. The definition strongly focuses on *how* the case is executed, and what can be learned to improve the execution of future similar cases. We, however, focus on *who* is able to construct and execute a concrete case based on the document. That is why we consider the *assignment* of the incoming document our main goal.

A. Organizational Circumstances

Figure 1 depicts the reference organization that will be used as an example to illustrate the approach. It is a manufacturing company.

As such, it has an *Engineering* department that is responsible for keeping the production processes and the required machinery running. For the overall process planning and improvement, a *Process Engineer* is designated. The actual maintenance and refitting of the machines is the goal of the subsidiary *Maintenance* department. It contains a designated *Engineer* and a number of *Technician*s.

The company also has a number of buildings as construction facilities and for administrative offices. The *Facility Management* department is in charge of these facilities. It also contains a subsidiary *Maintenance* department that employs *Technician*s for the upkeep of the buildings.

In addition to these departments, the company contains a *Human Resources* department that is responsible for personnel tasks, such as enlistment and dismissal. It has a *Head* and several *Employee*s.

B. Initial Event and Intentions

This company is contacted by an educational company that provides training services with offers for training services. There exists, however, neither a specified training process, nor an outline agreement of any kind with the external company. This offer takes the shape of a training brochure[1].

Table I shows the 10 most frequent words in the brochure. The preprocessing steps taken were:

[1]The brochure discussed in this contribution is a real-world example: http://www.toolingu.com/images/pdf/2014/2014_CorporateBrochure.pdf, online, accessed 2015-01-12

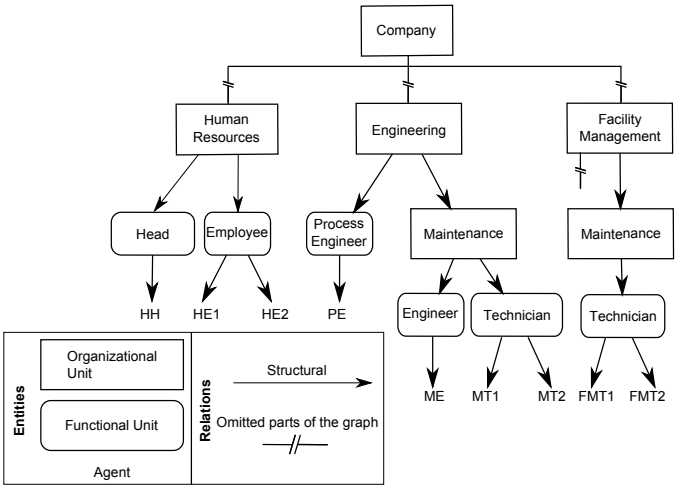

Fig. 1. Model of the Organizational Structure

Word	Frequency
training	56
manufacturing	19
knowledge	17
program	14
results	12
programs	10
skills	10
learning	9
right	9
toolingucom	9

TABLE I. Word Frequencies in the Brochure

1) Removing punctuation
2) Removing whitespaces
3) Conversion to lower case
4) Removing numbers
5) Removing common English stop words[2]

For a more comprehensible discussion of the algorithms, we will focus on an extracted section of the full text:

```
   MACHINING MAINTENANCE
STAMPING/FORMING/FABRICATING
WELDING ASSEMBLY / FINAL STAGE
PROCESSES PLASTICS PROCESSING
COMPOSITES PROCESSING ENGINEERING
FOUNDATIONAL
```

Section III-C uses this excerpt to describe the approach by example. Even though the conversion to lower case is part of the preprocessing, upper case will be used in the course of the discussion to indicate words from the text.

III. Approach

In order to illustrate the approach we will describe its different components. It is based on the organizational metamodel discussed in detail in [9]. We recapitulate relevant parts of it for clarity. The attribute-based search relies on additional definitions that are shown together with the core algorithms.

[2]The stop word list used was provided by [8].

A. Organizational Meta-Model

The meta-model consists in general of the set of entity-types $\mathcal{V}$ and the relation-types $\mathcal{R}$.

The excerpt of the meta-model consists of the following entity-types $\mathcal{V} = \mathcal{O} \cup \mathcal{F} \cup \mathcal{A}$:

- Organizational units $\mathcal{O}$
- Functional units $\mathcal{F}$
- Agents[3] $\mathcal{A}$

The set of relation-types $\mathcal{R} = \mathcal{R}_s \cup \mathcal{R}_o$ consists of:

- The structural relation-type $r \in \mathcal{R}_s$ with

$$r \subset \mathcal{O} \times (\mathcal{O} \cup \mathcal{F}) \tag{1}$$

$$r \subset \mathcal{F} \times (\mathcal{F} \cup \mathcal{A}) \tag{2}$$

- The set of organization-specific relation-types $\mathcal{R}_o$ (deputy, supervision and reporting)[4], cf. fig. 1, with $\forall r \in \mathcal{R}_o$:

$$r \subseteq \mathcal{A} \times (\mathcal{A} \cup \mathcal{F}) \tag{3}$$

$$r \subseteq \mathcal{F} \times (\mathcal{F} \cup \mathcal{A}) \tag{4}$$

$$r \subseteq \mathcal{O} \times (\mathcal{O} \cup \mathcal{F} \cup \mathcal{A}) \tag{5}$$

Additionally, the function $f_{val} : \mathcal{ATT} \mapsto \mathbb{W}$ assigns values $v \in \mathbb{W}$ to attributes $att \in \mathcal{ATT}$.[5] Attributes are assigned to both meta-model elements – $\mathcal{V}$ as well as $\mathcal{R}$. The set of attribute-types $\mathcal{ATT}$ consists of:

- obligatory attribute-types id and $name$
- user-defined attribute-types that can be defined as needed (i.a. certificate, expertise or hiring year of agents)

B. Attribute-Based Search

This section introduces the algorithms that describe the attribute-based search to find appropriate agents for documents. The following definitions are used in these algorithms:

Definition 1: Word w_i in text $\mathcal{T}$ with $i = 1, .., |\mathcal{T}|$

Definition 2: Text $\mathcal{T}$ is the vector $\overrightarrow{w}$ of words w_i: $\overrightarrow{w} = \begin{pmatrix} w_1 \\ w_2 \\ ... \\ w_{|\mathcal{T}|} \end{pmatrix}$. Equal words are more than once included in $\overrightarrow{w}$.

Definition 3: Let R be the set of relations of the organizational model that is mapped to relation-types $\mathcal{R}$ of the organizational meta-model.

Definition 4: Let V be the set of entities of the organizational model that is mapped to entity-types $\mathcal{V}$ of the organizational meta-model.

Definition 5: Let $v_j \in V$ be entities in the organizational model with $j = 1, .., |V|$

Definition 6: Let O be the set of organizational units of the organizational model that is mapped to the set of organizational unit entity-types $\mathcal{O}$ of the organizational meta-model. This includes a specific organizational unit $o \in O$.

Definition 7: Let F be the set of functional units of the organizational model that is mapped to the set of functional unit entity-types $\mathcal{F}$. This includes a specific functional unit $f \in F$.

Definition 8: Let A be the set of agents of the organizational model that is mapped to the set of agent entity-types $\mathcal{A}$ of the organizational meta-model. This includes a specific agent $a \in A$.

Definition 9: $type(e)$ returns the entity-type of the organizational model entity e ($e \in O \cup F \cup A$).

Definition 10: Let $I_t(w_i)$ with $t = O \vee F \vee A$ be the index lookup function that returns all entities (of type t) of the organizational model concerning the word w_i.

Definition 11: $\Omega_{\mathcal{T}}$ denotes the tuples of words and entities $(w_i, v_j) \in \Omega_{\mathcal{T}}$ resulting from the inverted index lookup on the organizational model: $\Omega_{\mathcal{T}} \subseteq \mathcal{T} \times (V \cup \{\emptyset\})$[6].

Definition 12: The words w_i of the Text $\mathcal{T}$ related to agents $a \in A$ are in the incidence matrix $\mathcal{I} = (x_{w_i,a})$. If agent a is not included for w_i in $\Omega_{\mathcal{T}}$ then $x_{w_i,a} = 0$ otherwise $x_{w_i,a} = 1$.

Definition 13: The notation $v \rightarrow^*_{r \in R_s} A_X$ with $v \in V \setminus A \wedge A_X \subseteq A$ indicates the transitive closure starting in v and resulting in set A_X. The operator denotes the traversal of relations r with $r \in R_s$.

Definition 14: Weight tuples $W \subseteq A \times \mathbb{N}_0$ of agents $a \in A$ and integers $n \in \mathbb{N}_0$ describe the number of occurrences of agent a in the candidates list C.

A bundle of words is extracted from an arbitrary text. These words are used to search entities based on their attributes within the organizational model, cf. algorithm 1. The result of the search is a list of organizational units O, functional units F and agents A[7]. The list contains exclusively entities that are returned at least once by the index search.

Afterwards, the list is expanded according to the entity-types ($\mathcal{O}, \mathcal{F}$ and $\mathcal{A}$), cf. algorithm 2. If needed, the organizational model is traversed to find agents A_X concerning the organizational unit O or functional unit F.[8] The result of the expansion is a list of agents C that are candidates for the assignment of the document. During this algorithm, the incidence matrix $\mathcal{I}$ is filled. It can be used to mathematically ensure the significance.

The assignment is done by calculating a ranking of the candidate list (list of agents C). The first element of this ranking is the "most fitting" candidate, the second agent is the next one, and so on. This calculation is described in algorithm 3.

[3]Agents are human or mechanical (i.a. persons, application systems and machines).

[4]The relations can be constrained, cf. [10].

[5]The values are arbitrary (like numbers, strings, etc.).

[6]It is possible that $w_k = w_l$ with $k \neq l$ (e.g. "Processing") but it is unique in sense of identifying the element w_i in $\Omega_{\mathcal{T}}$.

[7]The list containing different entities of different entity-types can be empty.

[8]Traversal for agents A is not necessary because they are still in the list.

Algorithm 1 Full-text search in the organizational model $search()$

Require: organizational model $ORG = (V, R)$, vector of words $\overrightarrow{w}$, sets of entities E_O, E_F, E_A, resulting tuples $\Omega_{\mathcal{T}} = \{\}$.
  for all $w_i \in \overrightarrow{w}$ **do**
    $E_O = I_O(w_i)$ {Index lookup for organizational units}
    if $E_O \neq \emptyset$ **then**
      for $o \in E_O$ **do**
        $\Omega_{\mathcal{T}} = \Omega_{\mathcal{T}} + (w_i, o)$
      end for
    end if
    $E_F = I_F(w_i)$ {Index lookup for functional units}
    if $E_F \neq \emptyset$ **then**
      for $f \in E_F$ **do**
        $\Omega_{\mathcal{T}} = \Omega_{\mathcal{T}} + (w_i, f)$
      end for
    end if
    $E_A = I_A(w_i)$ {Index lookup for agents}
    if $E_A \neq \emptyset$ **then**
      for $a \in E_A$ **do**
        $\Omega_{\mathcal{T}} = \Omega_{\mathcal{T}} + (w_i, a)$
      end for
    end if
  end for
  $expand(\Omega_{\mathcal{T}})$ {generate candidate list}

Algorithm 2 Entity expansion $expand()$

Require: tuples $\Omega_{\mathcal{T}}$, list of candidates $C = \{\}$, Incidence matrix $\mathcal{I}$: $\forall w, v : x_{w,v} = 0$.
  for all $(w, v) \in \Omega_{\mathcal{T}}$ **do**
    if $type(v) = \mathcal{O} \vee \mathcal{F}$ with $v \in (w, v)$ **then**
      $v \rightarrow^*_{r \in \mathcal{R}_s} A_X$ {$A_X \subseteq A$, transitive closure for v}
    else if $type(v) = \mathcal{A}$ with $v \in (w, v)$ **then**
      $A_X = \{v\}$ {add agent v}
    end if
    $C = C + A_X$ {add set of agents A_X to the list C}
    for all $a \in A_X$ **do**
      $x_{w,a} = 1$ {fill incident matrix $\mathcal{I}$}
    end for
  end for
  $rank(C)$ {calculate the ranking of the agents}

Algorithm 3 Calculation of the candidate ranking $rank()$

Require: list of candidates/agents C, weight tuples $W = (a, n)$: $\forall a \in C : (a, n) = (a, 0)$.
  for all $a \in C$ **do**
    $n = n + 1$ with $n \in (a, n)$ {increment occurrences of agent}
  end for
  return $sorted(W)$ {return the descending list of agents}

Assignment of Documents to Agents: The assignment of documents to agents is dependent on a variable top_A. The value of this variable can be assigned by humans or by assistant systems[9]. The value is an integer determining the number of the topmost agents of the descending list (cf. result of algorithm 3). A value of $top_A = 10$, for example, denotes the 10 most fitting agents. These agents get the document. An example for the assignment is given in section III-C.

This assignment can be evaluated by using a rank-sum test as described in [11] and compared with the result of different ranking methods, i.e. based on machine learning techniques (cf. [3]), or with manual assignments.

Search for Secondary Candidates: An additional variable top_{Deputy} for the deputy lookup is introduced to get the most appropriate "secondary" candidates. The value of the variable is, like top_A, an integer with $top_{Deputy} \leq top_A$ (e.g. $top_{Deputy} = 3$). It indicates the number of agents for which deputies are to be searched. If an agent of the "top top_{Deputy}" is unavailable, a mechanism is needed to get their best fitting deputies.

Such an approach is given by [12]. It describes the search for deputies on different levels of knowledge. This means that most fitting agent deputies are found first. If these deputy candidates are also unavailable, the next fitting deputies for the "starting" agent are searched. The further procedure of the deputy search works analogously.

C. Theory Explained by Example

In the following, the algorithms described previously are explained on basis of the example from section II. The example text is mapped to entities of the organizational reference model shown in figure 1.

1) Index Lookup: The index lookup uses a normalized representation of the words to find entities in the organizational model represented in figure 1. This includes stemming, as described algorithmically in [13] or based on a framework (cf. [14]). This leads to different words yielding the same search result. The stemmed representation of the example is:

```
MACHIN MAINTEN STAMP FORM FABRIC WELD
ASSEMBL FINAL STAGE PROCESS PLASTIC
PROCESS COMPOSIT PROCESS ENGIN FOUNDAT
```

The following elements of the organizational model are returned by the index lookups I_t for the example text[10], cf. algorithm 1:

[9] An assistant system calculates a probabilistic value (e.g. derived from historical data) to assign it to variable top_A.

[10] For empty results, the generic I_t is used as index symbol, otherwise the index variable denotes the type of the elements returned. The index assigned to elements in the result set indicates superordinate organizational units where needed for uniqueness.

$$
\begin{aligned}
I_t(MACHIN) &= \emptyset \\
I_O(MAINTEN) &= \{Maintenance_E, Maintenance_{FM}\} \\
I_t(STAMP) &= \emptyset \\
I_t(FORM) &= \emptyset \\
I_t(FABRIC) &= \emptyset \\
I_t(WELD) &= \emptyset \\
I_t(ASSEMBL) &= \emptyset \\
I_t(FINAL) &= \emptyset \\
I_t(STAGE) &= \emptyset \\
I_F(PROCESS) &= \{ProcessEngineer\} \\
I_t(PLASTIC) &= \emptyset \\
I_F(PROCESS) &= \{ProcessEngineer\} \\
I_t(COMPOSIT) &= \emptyset \\
I_F(PROCESS) &= \{ProcessEngineer\} \\
I_O(ENGIN) &= \{Engineering\}, \\
I_F(ENGIN) &= \{ProcessEngineer, Engineer\} \\
I_t(FOUNDAT) &= \emptyset
\end{aligned}
$$

As can be seen, the index lookup returns only entities that contain values with a high similarity to the words in the document. In practice, all attributes, such as description texts, are relevant to the lookup. In this example, the entities only contain a name attribute with corresponding value. This results in the final list of tuples:

$$
\begin{aligned}
\Omega_{\mathcal{T}} = \{&(MAINTEN, Maintenance_E), \\
&(MAINTEN, Maintenance_{FM}), \\
&(PROCESS, ProcessEngineer), \\
&(PROCESS, ProcessEngineer), \\
&(PROCESS, ProcessEngineer), \\
&(ENGIN, Engineering), \\
&(ENGIN, ProcessEngineer), \\
&(ENGIN, Engineer)\}
\end{aligned}
$$

The consequence of the index lookup is that any words that can not be mapped to organizational elements have no further effect on the assignment process. This causes the complexity of all subsequent steps to scale with the degree of coverage, not with the actual length of the document.

2) Generation of the List of Candidates: Algorithm 2 $expand()$ expands the elements contained in the list of tuples $\Omega_{\mathcal{T}}$ to a list of candidates C. As $\Omega_{\mathcal{T}}$ can contain non-agent members (functional or organizational units), it resolves the corresponding agents by forming the transitive closure:

1) $Maintenance_E \rightarrow^*_{r \in R_s} \{ME, MT1, MT2\}$
2) $Maintenance_{FM} \rightarrow^*_{r \in R_s} \{FMT1, FMT2\}$
3) $ProcessEngineer \rightarrow^*_{r \in R_s} \{PE\}$
4) $ProcessEngineer \rightarrow^*_{r \in R_s} \{PE\}$
5) $ProcessEngineer \rightarrow^*_{r \in R_s} \{PE\}$
6) $Engineering \rightarrow^*_{r \in R_s} \{PE, ME, MT1, MT2\}$
7) $ProcessEngineer \rightarrow^*_{r \in R_s} \{PE\}$
8) $Engineer \rightarrow^*_{r \in R_s} \{ME\}$

At this point, it can make sense to introduce a caching mechanism to store intermediate results of this operation for frequent start entities, e.g. $ProcessEngineer$. [15] gives an overview of strategies that could be employed.

Applying the expansion algorithm to the example results in the following list of candidates:

$$
\begin{aligned}
C = \{&ME, MT1, MT2, \\
&FMT1, FMT2, \\
&PE, PE, PE, \\
&PE, ME, MT1, MT2, \\
&PE, \\
&ME\}
\end{aligned}
$$

3) Ranking: At this point, algorithm 3 $rank()$ transforms the list of candidates C into a set W of tuples that contain the candidate and their respective weight for the assignment. The weight of the agents is the sum of their occurrences in C. The example yields:

$$
\begin{aligned}
W = \{&(ME, 3), (MT1, 2), (MT2, 2), \\
&(FMT1, 1), (FMT2, 1), (PE, 5)\}
\end{aligned}
$$

This corresponds to the sorted list:

$$
\begin{aligned}
&(PE, 5) \\
&(ME, 3) \\
&(MT1, 2), (MT2, 2) \\
&(FMT1, 1), (FMT2, 1)
\end{aligned}
$$

The assignment of documents to agents is parameterized by the variable top_A. The value for this variable is e.g. $top_A = 3$. Thus, the document is assigned to $PE, ME, MT1$ and $MT2$. The agent PE is the "most fitting" ($weight = 5$), ME is the second ranked agent ($weight = 3$) and the third rank is $MT1$ and $MT2$ ($weight = 2$). Both, $MT1$ and $MT2$, are assigned to the document as both have the same ranking position. The value of the variable is not a hard constraint[11] $top_A = 3$. It represents the specification of all agents that have the same position in the sorted list.

In case that "fitting" agents – $top_A = 3$ – are unavailable, the mechanism for searching deputies is also parameterized with the variable $top_{Deputy} = 1$. This determines that deputies are only searched for the top-ranked agent, PE. The first top_{Deputy} agents are declared so important, that appropriate deputies should be returned for them if they are unavailable. [12] provides a way to resolve deputies that are as closely related to the original agents as possible.

D. Impact of Explicit Organizational Models

The previous considerations were based on just an extract of the original document, as the chosen extract describes concrete competencies to be affected by the offered training services. The most frequent words in the overall document (cf. table I) however indicate a strong connection to training and skill management.

[11] Hard constraint means that the value is obligatory.

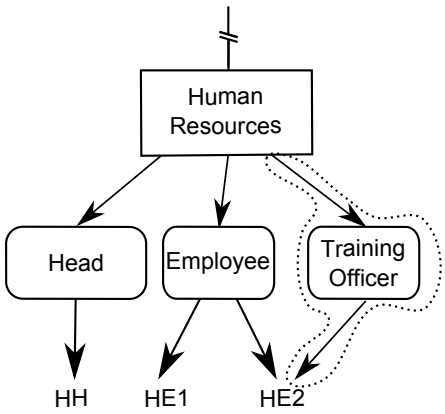

Fig. 2. Altered Organizational Model for Human Resources

Training and development is a core aspect of human resource management, as can be seen in [16]. In the original organizational model (fig. 1), the *Human Resources* department is not represented very diversely. It only consists of a *Head* and several *Employee*s. The actual function of the employees is not made explicit. As a consequence, the most frequent word from the document, `training`, does not yield any results in the organization.

Figure 2 now introduces a new functional unit into the model: *HE2* is appointed *Training Officer*. This functional unit matches an index search for the word `training`, which occurs 56 times in the document. This causes algorithm 1 to put the tuple $(TRAIN, TrainingOfficer)$ 56 times into $\Omega_{\mathcal{T}}$. Similar to the tuple $(PROCESS, ProcessEngineer)$ from the excerpt discussed in detail, this causes the respective agent *HE2* to be ranked extremely high as recipient for the document.

This shows that the explicit representation of organizational knowledge has a deep impact on the quality of the document assignment. A similar result can be achieved by making the responsibility for training an attribute value of *HE2*.

IV. Conclusion and Outlook

The example discussed in this contribution demonstrates the importance of making functions in organizational models explicit. Such explicit models can serve as a reliable basis for further process automation, such as the determination of incident and case stakeholders. As shown by the introduction of the additional functional unit into the model, the quality of the assignment can be vastly increased by making organizational knowledge explicit.

The approach is highly suitable for parallelization. It consists exclusively of read operations. This eliminates resource locking issues. It also has several steps that can be performed in parallel. The lookup in different indices can be split by entity type. The whole chain of algorithms could also be executed in parallel on a per-word basis (except for the final summation). This is very suitable for the MapReduce family of algorithms, cf. [17].

Because the meta-model that forms the basis for the organizational model supports the assignment of attributes to relations, the search algorithm 1 can also return relations. This is not covered in this contribution, but the algorithms can be extended to resolve additional agents based on this information. An example for such a case would be relations in the organizational model that are only valid in a specific context. The document received by the company could be mapped to such a context, increasing the list of candidates and improving the ranking.

Currently, the approach for searching the appropriate agents relies on a close match on a word / attribute value basis. It can be assumed that a richer semantic model of the organization and its processes can lead to a higher precision. Currently, any (normalized) word not found in the organizational model by information retrieval methods is essentially considered a stop word and has no further consequences for the assignment process. At this point, a more semantic Bag-of-Concepts approach [18] needs to be evaluated.

References

[1] T. Janner, A. Schmidt, C. Schroth, and G. Stuhec, "From EDI to UN/CEFACT: An evolutionary path towards a next generation e-business framework," in *Proceedings of the 5th International Conference on e-Business*, vol. 2006, 2006.

[2] A. Lawall, D. Reichelt, and T. Schaller, "Propagation of Agents to Trusted Organizations," *2014 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT)*, vol. 3, pp. 433–439, 2014.

[3] D. Michie, D. J. Spiegelhalter, C. C. Taylor, and J. Campbell, Eds., *Machine Learning, Neural and Statistical Classification*. Upper Saddle River, NJ, USA: Ellis Horwood, 1994.

[4] P. Resnick and H. R. Varian, "Recommender Systems," *Commun. ACM*, vol. 40, no. 3, pp. 56–58, Mar. 1997.

[5] K. Zeilenga, "Lightweight directory access protocol (ldap): Technical specification road map," 2006. [Online]. Available: https://tools.ietf.org/html/rfc4510

[6] J. Sermersheim, "RFC 4511: Lightweight Directory Access Protocol (LDAP): The Protocol," *Internet Engineering Task Force, Tech. Rep*, 2006. [Online]. Available: https://tools.ietf.org/html/rfc4511

[7] J. Kolodner, *Case-based reasoning*. Morgan Kaufmann, 2014.

[8] D. D. Lewis, Y. Yang, T. G. Rose, and F. Li, "RCV1: A New Benchmark Collection for Text Categorization Research," *J. Mach. Learn. Res.*, vol. 5, pp. 361–397, Dec. 2004.

[9] A. Lawall, T. Schaller, and D. Reichelt, "Enterprise Architecture: A Formalism for Modeling Organizational Structures in Information Systems," in *Enterprise and Organizational Modeling and Simulation*, ser. Lecture Notes in Business Information Processing, J. Barjis and R. Pergl, Eds. Springer Berlin Heidelberg, 2014, vol. 191, pp. 77–95.

[10] ——, "Who Does What – Comparison of Approaches for the Definition of Agents in Workflows," in *Web Intelligence (WI) and Intelligent Agent Technologies (IAT), 2013 IEEE/WIC/ACM International Joint Conferences on*, vol. 3, Nov 2013, pp. 74–77.

[11] F. Wilcoxon, "Individual Comparisons by Ranking Methods," *Biometrics Bulletin*, vol. 1, no. 6, pp. 80–83, Dec. 1945.

[12] A. Lawall, T. Schaller, and D. Reichelt, "Local-Global Agent Failover Based on Organizational Models," in *Web Intelligence (WI) and Intelligent Agent Technologies (IAT), 2014 IEEE/WIC/ACM International Joint Conferences on*, vol. 3, Aug 2014, pp. 420–427.

[13] M. Porter, "An algorithm for suffix stripping," *Program*, vol. 14, no. 3, pp. 130–137, 1980.

[14] M. F. Porter, "Snowball: A language for stemming algorithms," 2001. [Online]. Available: http://snowball.tartarus.org/texts/introduction.html

[15] R. Karedla, J. Love, and B. Wherry, "Caching strategies to improve disk system performance," *Computer*, vol. 27, no. 3, pp. 38–46, March 1994.

[16] R. S. Schuler and S. E. Jackson, "Linking competitive strategies with human resource management practices," *The Academy of Management Executive (1987-1989)*, pp. 207–219, 1987.

[17] J. Dean and S. Ghemawat, "MapReduce: Simplified Data Processing on Large Clusters," *Commun. ACM*, vol. 51, no. 1, pp. 107–113, Jan. 2008.

[18] M. Sahlgren and R. Cöster, "Using Bag-of-concepts to Improve the Performance of Support Vector Machines in Text Categorization," in *Proceedings of the 20th International Conference on Computational Linguistics*, ser. COLING '04. Stroudsburg, PA, USA: Association for Computational Linguistics, 2004.