Context-dependent Lexical and Syntactic Disambiguation in Ontology Population *

Natalia Garanina and Elena Sidorova

A.P. Ershov Institute of Informatics Systems,
Lavrent'ev av., 6, Novosibirsk 630090, Russia
{garanina,lsidorova}@iis.nsk.su

Abstract. We suggest an approach to resolution of context-dependent lexical and syntactic ambiguity in a framework of ontology population from natural language texts. We show that a set of maximally determined ontology instances can be represented as a Scott information system with an entailment relation as a collection of information connections. Moreover, consistent primary lexical instances form FCA-concepts. These representations are used to justify correctness of lexical disambiguation and to define syntactic ambiguity and its resolution. This information system generates a multi-agent system in which agents resolve the ambiguity of both types.

1 Introduction

Ontological databases are currently widely used for storing information obtained from a great number of sources. To complete such ontologies, formalisms and methods that allow one to automate the process are developed. Features of automatic information retrieval cause ontology population ambiguities. In linguistics several kinds of ambiguities are considered: lexical, syntactic, semantic, and pragmatic [2]. In a process of ontology population from natural language texts we use our algorithms [5] in which the following ambiguity types appear: (1) several ontology instances or data attributes correspond to the same text fragment, (2) some value is incorrectly assigned to some attribute of some instance, (3) some value is incorrectly assigned to attributes of several instances, (4) some value is incorrectly assigned to several attributes of some instance, (5) several values are assigned to one-valued attribute of some instance. The first type corresponds to lexical ambiguity, and other types are syntactic ambiguity. An algorithm for lexical disambiguation was represented in [6]. In this work we suggest the modified algorithm for resolving lexical ambiguity, a new algorithm for syntactic disambiguation, and we justify the correctness of both of them.

In [6] we demonstrated that the process of retrieval of information in a form of a set of ontology instances can be presented as a Scott information system [13].

* The research has been supported by Russian Foundation for Basic Research (grant 15-07-04144) and Siberian Branch of Russian Academy of Science (Integration Grant n.15/10 "Mathematical and Methodological Aspects of Intellectual Information Systems").

This process produces maximally determined instances for ontology population. In this paper we prove that consistent sets of instances and lexical objects which values assign attributes of these instances form FCA concepts [3]. This fact grantees that information states of ambiguous conflicting agents do not intersect. This implies correctness of lexical disambiguation.

Besides, now we use a representation of ontologies which does not consider ontology relations as special structures. Only classes are allowed in these ontologies, and relations are represented as special attributes of classes. Well-known ontology representation language OWL uses the notation of this kind. This representation is a good solution for specification of polyadic relations. Our algorithms for ontology population are simpler with this representation because class and relation instances are packed in the same item.

Automatic techniques of disambiguation usually do not use an input data context in full. This can lead to incomplete and incorrect ambiguity resolution [1, 9, 8, 7]. Our approach tries to ease these drawbacks. For disambiguation we use a distributed approach. Every retrieved instance is related to agent. These agents detect and resolve ambiguities with help of a special master agent. This approach takes polynomial time for disambiguation.

The rest of the paper is organized as follows. In Section 2, an approach to ontology population in the framework of information systems is discussed. Section 3 describes lexical and syntactic disambiguation in terms of the system defined in the previous section. The next Section 4, gives definitions for a multi-agent system of context-dependent ambiguity resolution. Section 5 informally describes agents of our systems, their action protocols, and the main conflict resolution algorithm. In the concluding Section 6, directions of future researches are discussed.

2 Scott Information Systems in Ontology Population

Let we be given an ontology of a subject domain, the ontology population rules, semantic and syntactic model for a sublanguage of the subject domain and a data format, and input data as a finite natural language text with information for population of the ontology. We consider *ontology O of a subject domain* which includes (1) finite nonempty set C_O of classes for concepts of the subject domain, (2) a finite set of attributes with names in $Dat_O \cup Rel_O$, each of which has values in some data domain (data attributes in Dat_O) or is some instance of the ontology (relation attributes in Rel_O, which model relations), and (3) finite set D_O of data types. Every class $c \in C_O$ is defined by a tuple of typed attributes: $c = (Dat_c, Rel_c)$, where every data attribute $\alpha \in Dat_c \subseteq Dat_O$ has type $d_\alpha \in D_O$ with values in V_{d_α} and every relation attribute $\rho \in Rel_c \subseteq Rel_O$ is of class $c_\rho \in C_O$. Let a set of all values of all attribute be $V_O = \cup_{d_\alpha \in D_O} V_{d_\alpha}$. *Information content* IC_O of ontology O is a set of class instances, where every instance $a \in IC_O$ is of form (c_a, Dat_a, Rel_a), where c_a is a class of the instance, every data attribute in Dat_a has name $\alpha \in Dat_{c_a}$ with value(s) in V_{d_α} and every relation attribute in Rel_a has name $\rho \in Rel_{c_a}$ with a value as an instance

of class c_ρ. *Ontology population problem* is to compute an information content for a given ontology from given input data. Input data for the ontology population process are natural language texts. These data are finite and our algorithms of ontology-oriented text analysis can generate a finite set of ontology instances [5]. Finiteness of the set is guaranteed by prohibition for the rules from generating infinite information items by one position. We suggest to consider this process of forming ontology instances as work with Scott information systems. A *Scott information system* T is a triple $(T, Con, \vdash)$, where

- T is a set of tokens and $Fin(T)$ is a set of finite subsets;
- Con is a consistency predicate such that $Con \subseteq Fin(T)$, and
  - **1.** $Y \in Con$ and $X \subseteq Y \Rightarrow X \in Con$,
  - **2.** $a \in T \Rightarrow \{a\} \in Con$;
- $\vdash$ is an entailment relation such that $\vdash \subseteq Con \setminus \{\emptyset\} \times T$ and
  - **3.** $X \vdash a \Rightarrow X \cup \{a\} \in Con$,
  - **4.** $X \in Con$ and $a \in X \Rightarrow X \vdash a$,
  - **5.** $\forall b \in Y : X \vdash b$ and $Y \vdash c \Rightarrow X \vdash c$.

The *information retrieval system* based on an ontology, finite input data, and rules of the ontology population and the data processing is defined as a triple $R = (A, Con, \vdash)$. Set of tokens A consists of a set of all (underdetermined) ontology *p-instances* formed by the rules in the determination process of initial p-instances which are retrieved from an input text by the special preprocess. Every p-instances $a \in A$ has form (c_a, Dat_a, Rel_a, P_a), where

- class $c_a \in C_O$, and
- every data p-attribute $\alpha_a \in Dat_a$ is of form (α, IV_α), where
  - name $\alpha \in Dat_{c_a}$, where
  - its information values $\bar{v} \in IV_\alpha$ has form $(v_{\bar{v}}, g_{\bar{v}}, s_{\bar{v}})$ with
    - data value $v_{\bar{v}} \in d_\alpha$, a set of all values of α is $Val_{\alpha_a} = \{v_{\bar{v}} \mid \bar{v} \in IV_\alpha\}$,
    - $g_{\bar{v}}$ is grammar information (morphological and syntactic features), and
    - $s_{\bar{v}}$ is structural information (position in input data);
- every relation p-attribute $\rho_a \in Rel_a$ is of form (ρ, O_{ρ_a}), where
  - a name $\rho \in Rel_{c_a}$, and
  - every $\bar{o} \in O_{\rho_a}$ has form (o, p_o), where
    - o is an instance of class c_{ρ_a}, and
    - $p_o \in P_o$ is its position,
  - a set of all relation objects of a is $O(Rel_a) = \{a\} \cup_{\rho_a \in Rel_a} \{o | \bar{o} \in O_{\rho_a}\}$;
- P_a is structural information (a set of positions in input data).

We consider a special set of tokens: a set of lexical objects LO corresponding to values of data attributes retrieved from input data. Every lexical object is a p-instance which has only a single data attribute with a single information value. P-instances correspond to ontology instances in a natural way. Let $a = (c_a, Dat_a, Rel_a, p_a)$ be p-instance, then its corresponding ontology instance is $a' = (c_a, Dat_{a'}, Rel_{a'})$, where every $\alpha \in Dat_{a'}$ has value(s) in Val_{α_a} and every $\rho \in Rel_{a'}$ has value o with $(o, p_o) \in O_{\rho_a}$. Further we omit prefix "p-" if there is no ambiguity. An *information order* relation $\prec$ is defined on ontology instances. Let $a, a' \in A$: $a \prec a'$, if $a = a'$ everywhere except for at least one attribute, with

the number of values of this attribute in a being strictly less than that in a'. For $x, x' \in A$: if $x \prec x'$, then x' is *information extension of* x.

Rules of ontology population and data processing $Rules = \{rule_1, \ldots, rule_n\}$ map finite sets of instances of ontology classes to an instance which is an informational extension of some instance of the domain set or a new instance. This sets must be linguistically and ontologically compatible: specified sets of their attributes and some instances have to satisfy conditions on values, grammatical and structural information [10]:
$rule_i : Dom_i \mapsto A$, $Dom_i \subseteq 2^A$, such that
$\forall X \in Dom_i \ : \ LingCons_i(\cup_{x' \in X} Dat_{x'} \cup Rel_{x'}) = true \wedge \forall x' \in X : c_{x'} \in Class_i$, where predicate $LingCons_i$ and set of classes $Class_i \subseteq C_O$ detect linguistic and ontological compatibility of the instance set, correspondingly. Let for $X \in Dom_i, x \in A$:
$rule_i(X) = x$ iff $((\exists y \in X : y \prec x \wedge c_x = c_y) \vee (\forall y \in X : y \not\prec x \wedge c_x = gen_i(X))) \wedge$
$(Dat_x = \emptyset \vee Dat_x = \cup_\alpha(\alpha, \cup\{(f_i(\bar{V_\alpha}), g_i(\bar{V_\alpha}), s_i(\bar{V_\alpha})) \mid$
$\exists Y_\alpha \subseteq X : dat_\alpha \subseteq \cap_{y \in Y_\alpha} Dat_y \wedge \bar{V_\alpha} = \cup_{\beta \in dat_\alpha}\{\bar{v} | \bar{v} \in IV_\beta\}\})) \wedge$
$(Rel_x = \emptyset \vee \forall o \in O(Rel_x) : o \in X \cup_{y \in X} O(Rel_y))$,
where $gen_i(X)$ generates a new class for a new instance, $f_i(\bar{V})$ produces a value based on values in $(\bar{V})$ for an attribute of instance x, and $g_i(\bar{V})$ and $s_i(\bar{V})$ inherit grammatical and structural information from set of information values $\bar{V}$.

Consistency predicate Con and entailment relation $\vdash$ correspond to the rules of ontology population and data processing. Let $x, x' \in A$ and $X \subseteq A$. The entailment relation connects informationally associated tokens:

– $X \vdash x$, iff $x \in X$, or $x \notin X \wedge$
$((\exists X' \subseteq X, X'' \subseteq A, rule_i \in Rules : rule_i(X' \cup X'') = x) \vee$
$(\exists x' \in A, X'' \subseteq A, rule_i \in Rules : X \vdash x' \wedge rule_i(\{x'\} \cup X'') = x))$,
i.e. instance x is entailed from X, if it is in this set, or information from tokens of this set is used for evaluating attributes of x.

The consistency predicate defines informationally consistent sets of tokens:

– $X \in Con$, iff for some $rule_i \in Rules$ holds $\forall X' \subseteq X(\cup_{x' \in X'} c_x \subseteq Class_i) \Rightarrow$ $(\exists x \in A, X'' \subseteq A : rule_i(X' \cup X'') = x)$, i.e. if there exists some rule which can find in a set of tokens some instances satisfying its class compatibility then these instances should be consistent with some other set of tokens with respect to the rule. Class compatible, but linguistically incompatible sets cannot be processed by rules, hence we do not consider them consistent.

Let us prove the following theorem for the system R:

Theorem 1. *Triple* $R = (A, Con, \vdash)$ *is a Scott information system.*

Proof. Let us show that the consistency predicate Con and the entailment relation $\vdash$ satisfy properties 1–5 of information systems.

1. $Y \in Con$ and $X \subseteq Y \Rightarrow X \in Con$. This fact follows from the definition of the consistency predicate directly because the condition of definition should hold for every subset of a consistent set.

2. $a \in A \Rightarrow \{a\} \in Con$. By the definition for every $rule_i \in Rules$ the class of a is not included in $Class_i$ or single $\{a\}$ can be complemented by some set of tokens in such a way that the $rule_i$ produces a new token.

3. $X \in Con$ and $X \vdash a \Rightarrow X \cup \{a\} \in Con$. $X \cup \{a\}$ is consistent because $a \in X$ or lexical information of a is inherited from X by definitions of the entailment relation and process rules.

4. $X \in Con$ and $a \in X \Rightarrow X \vdash a$. By the def. of the entailment relation.

5. $\forall b \in Y : X \vdash b$ and $Y \vdash c \Rightarrow X \vdash c$. Let $Y_b = Y \setminus \{b\}$. $X \vdash c$ iff $(\exists b \in Y, Y_b \subseteq A, rule \in Rules : X \vdash b \wedge rule(\{b\} \cup Y_b) = c))$ by the def. of the entailment relation.■

The proposition below directly follows from monotonicity of the entailment relation and finiteness of input data.

Proposition 1. *Information retrieval process of ontology population terminates.*

For token $x \in A$: $x^{\uparrow} = \{x\} \cup \{x' \mid x \prec x'\}$ and $x^{\downarrow} = \{x\} \cup \{x' \mid x' \prec x\}$ are *upper and down cones of* x. Let a set of *maximally determined* instances (maximal instances or tokens), which is the result of the analysis of input data, be $A^{\uparrow} = \{x \in A \mid x^{\uparrow} = \{x\}\}$. These instances may populate an ontology. Obviously,

Proposition 2. *Triple* $I = (A^{\uparrow}, Con, \vdash)$ *is a Scott information system.*

An information descendants of token $a \in A^{\uparrow}$ are all maximal tokens (all information) that can be obtained from this token by the entailment relation: $Ds(a) = \{x \in A^{\uparrow} | \{a\} \vdash x\}$. *An information ancestors* of token $a \in A^{\uparrow}$ are all maximal tokens from which a can be obtained: $An(a) = \{x \in A^{\uparrow} | \{x\} \vdash a\}$. In our framework for lexical objects the following equality holds: $An(a) = \{a\}$ because ontology instances are based on retrieved lexical objects. Information descendants are a particular case of Scott information states [14]. Like in the cited paper, we show that tokens from LO and their information descendants form a concept lattice.

Proposition 3. *Consistent sets of lexical objects form FCA concepts. Every consistent set of instances is a base for FCA concepts.*

Proof. Let every set x of information descendants of LO be an object, and every $l \in LO$ be an attribute. Lexical object l is an attribute of x iff $l \in x$. The extension of a set of attributes $L \subseteq LO$ is the set $L' = \{x | L \subseteq x\}$ and the intension of L' is the set $\{l | \forall x \in L', l \in x\}$. L is a concept iff the condition on the intension of the extension of L holds: $L = \{l | \forall x \in L', l \in x\}$ iff L is an information state of information system I iff L is a consistent set. The intension of a set of infostates X is the set $X' = \{l | \forall x \in X, l \in x\}$ and the extension of X' is the set $\{x | X' \subseteq x\}$. X is a concept iff the condition on the extension of the intension of X holds: $X = \{x | X' \subseteq x\}$ iff a set of all instances in set of infostates X forms an infostate too: $X^i = \{a \mid a \in x \in X\}$, hence X^i is a consistent set. Hence every consistent set of instances is a base for FCA concepts.■

3 Ambiguity and Resolution

(1) Lexical ambiguity.
Let $l, l' \in LO$ be in a conflict $l \leftrightsquigarrow l'$ iff $s(l) \cap s(l') \neq \emptyset$. Let set *AmbLO*

be a set of conflict lexical objects and Lex be a set of their descendants. We consider that rules in $Rules$ cannot generate instances which include inconsistent information. I.e. for every $rule_i \in Rules$ holds $\forall X \in Dom_i, a, a' \in X, l, l' \in An(a) \cap An(a') \cap LO : \neg(l \leftrightsquigarrow l')$. Hence for lexical objects l and l' in conflict: $Ds(l) \cap Ds(l') = \emptyset$.

For the lexical disambiguation of two conflicting lexical objects we prefer a lexical object which is more incorporated in an input text than its competitor. For $l, l' \in LO$ if $|Des(l)| > |Des(l')|$ we take l for evaluating attributes of ontology instances and ignore l'.

(2) Syntactic ambiguity.
Detection of syntactic ambiguity frequently requires analysis of homogeneous groups. Syntactic ambiguity is defined for ontology instances, not for lexical objects. Our types of syntactic ambiguity can depend on an ontology specification, hence here we consider syntactic-semantic ambiguity really. We omit "-semantic" for the brevity. Syntactic ambiguity usually can be expressed by corresponding single lexical object to several ontology items in various ways. For disambiguation it is necessary to find in an input text an evidence of correctness of the correspondence. This could be performed using the following inequalities.

For every instance a and its data attribute $\alpha \in Dat_a$ *a set of information values equal to* $v \in d_\alpha$ is $EQ(a, \alpha, v) = \{\bar{v} \in IV_\alpha \mid v_{\bar{v}} = v\}$. For every instance a and its relation attribute $\rho \in Rel_a$ *a set of relation objects with an instance equal to* $e \in c_{\rho_a}$ is $EQ(a, \rho, e) = \{(o, p_o) \in O_\rho \mid o = e\}$. A power of these sets is *an evidence power*. A triple $(a, \alpha, \bar{v})$ denotes information value $\bar{v} \in IV_\alpha$ of data attribute $\alpha \in Dat_a$ of instance a. A couple (a, ρ) denotes a value of relation attribute $\rho \in Rel_a$ of instance a. A set of *information values which effects on* $(a, \alpha, \bar{v})$ is $V(a, \alpha, \bar{v}) = \{(c, \gamma, \bar{w}) \mid \exists rule_i \in Rules, X \subseteq A^\uparrow : rule_i(X) = a \wedge c \in X \wedge \gamma \in dat_\alpha \cap Dat_c \wedge \bar{w} \in IV_\gamma \wedge \bar{v} = (f(\bar{V_\alpha}), g(\bar{V_\alpha}), s(\bar{V_\alpha}))\}$. A set of *instances which effects on* (a, ρ) is $I(a, \rho) = \{e \in A^\uparrow \mid \exists rule \in Rules, X \subseteq A^\uparrow : rule(X) = a \wedge e \in X \wedge O_\rho \cap O(Rel_e) \neq \emptyset\}$. Now we define a method of syntactic disambiguation.

(1) *Some value is incorrectly assigned to some attribute of an instance (Synt11).* An example: "The old men and women sat on the bench." The women may or may not be old. Hence, attribute "age" of instance "women" may not has value "old". Let a set of instances with ambiguity of this type be denoted as $Synt11$. Let in instance a information value $(c, \gamma, \bar{w})$ effect on $(a, \alpha, \bar{v})$: $(c, \gamma, \bar{w}) \in V(a, \alpha, \bar{v})$. Then in a case of the ambiguity, $(c, \gamma, \bar{w})$ is declared as effecting on $(a, \alpha, \bar{v})$ iff $|EQ(a, \alpha, v_{\bar{v}})| > 1$. Let in instance a instance e effect on (a, ρ): $e \in I(a, \rho)$. Then in a case of the ambiguity instance e is declared as effecting on (a, ρ) iff $|EQ(a, \rho, e)| > 1$.

(2) *Some value is incorrectly assigned to attributes of several instances (Synt12).* An example: "Someone shot the maid of the actress who was on the balcony." Either the actress or the maid was on the balcony. Hence, either attribute "place" of instance "actress" or attribute "place" of instance "maid" may has value "balcony". Let a set of instances with ambiguity of this type be denoted as $Synt12$. Let in instances a and b information value $(c, \gamma, \bar{w})$ effect on $(a, \alpha, \bar{v})$

and $(b, \beta, \bar{u})$: $(c, \gamma, \bar{w}) \in V(a, \alpha, \bar{v}) \cap V(b, \beta, \bar{u})$. Then in a case of the ambiguity $(c, \gamma, \bar{w})$ is declared as effecting on $(a, \alpha, \bar{v})$ and not on $(b, \beta, \bar{u})$ iff $|EQ(a, \alpha, v_{\bar{v}})| > |EQ(b, \beta, \bar{u})|$. Let in instances a and b instance e effect on (a, ρ) and (b, o): $e \in I(a, \rho) \cap I(b, o)$. Then in a case of the ambiguity instance e is declared as effecting on (a, ρ) and not on (b, o) iff $|EQ(a, \rho, e)| > |EQ(b, o, e)|$.
(3) *A value is incorrectly assigned to several attributes of an instance (Synt112).* An example: "Cuban jazz band." A group of Cuban musicians performing jazz music or a group of musicians performing Cuban jazz. Hence, attribute "country" or attribute "style" of instance "band" may has value "Cuban". Let a set of instances with ambiguity of this type be denoted as $Synt112$. Let in instance a information value $(c, \gamma, \bar{w})$ effect on $(a, \alpha, \bar{v})$ and $(a, \beta, \bar{u})$: $(c, \gamma, \bar{w}) \in V(a, \alpha, \bar{v}) \cap V(a, \beta, \bar{u})$. Then in a case of the ambiguity $(c, \gamma, \bar{w})$ is declared as effecting on $(a, \alpha, \bar{v})$ and not on $(a, \beta, \bar{u})$ iff $|EQ(a, \alpha, v_{\bar{v}})| > |EQ(a, \beta, v_{\bar{u}})|$. Let in instance a instance e effect on (a, ρ) and (a, o): $e \in I(a, \rho) \cap I(a, o)$. Then in a case of the ambiguity instance e is declared as effecting on (a, ρ) and not on (a, o) iff $|EQ(a, \rho, e)| > |EQ(a, o, e)|$.
(4) *Several values are assigned to one-valued attribute of an instance (Synt211).* An example: "Shakespeare is an author of the piece." A gender of Shakespeare may be either male or female. Hence, one-valued attribute "gender" of instance "person" may has value either "male" or "female". Let a set of instances with ambiguity of this type be denoted as $Synt211$. Let in instance a information values $(b, \beta, \bar{u})$ and $(c, \gamma, \bar{w})$ effect on $(a, \alpha, \bar{v})$ and $(a, \alpha, \bar{v'})$, respectively: $(b, \beta, \bar{u}) \in V(a, \alpha, \bar{v})$ and $(c, \gamma, \bar{w}) \in V(a, \alpha, \bar{v'})$. Then in a case of the ambiguity $(b, \beta, \bar{u})$ is declared as effecting on $(a, \alpha, \bar{v})$, and $(c, \gamma, \bar{w})$ is declared as not effecting on α iff $|EQ(a, \alpha, v_{\bar{v}})| > |EQ(a, \alpha, v_{\bar{v'}})|$. Let in instance a instance e and e' effect on (a, ρ): $e, e' \in I(a, \rho)$. Then in a case of the ambiguity instance e is declared as effecting on (a, ρ), and e' is declared as not effecting on (a, ρ) iff $|EQ(a, \rho, e)| > |EQ(a, \rho, e')|$.

In a case of equalities of evidence powers the conflict is not resolved. We consider systems in which all these ambiguities are independent, i.e. pairwise intersections of sets Lex, $Synt11$, $Synt12$, $Synt112$ and $Synt211$ are empty. Informal description of action protocols for instance agents presents resolution of independent lexical and syntactic ambiguities. These protocols work correctly if resolution of references and detection of syntactic ambiguities are correct.

4 Multi-agent Ambiguity Resolution

Let a set of lexical objects which effect on some information value of data attribute α of instance a be $L(a, \alpha) = \{l \in LO \mid \exists rule_i \in Rules, X \subseteq A^{\uparrow} : rule_i(X) = a \wedge (\exists x \in X : x \in Ds(l) \wedge (\exists \beta \in dat_\alpha \cap Dat_x : l \in L(x, \beta)))\}$. For every $x \notin A^{\uparrow}$, the corresponding maximally determined instance is $\tilde{x}$ such that $\tilde{x} \in A^{\uparrow} \wedge x \prec \tilde{x}$. Entailment relation $\vdash$ generates *information connections* between maximally determined instances. Let $X \vdash x$ and $y \in X \wedge y \notin x^{\downarrow}$. Then

– *information connections* between $\tilde{y}$ and $\tilde{x}$ are
  – $\tilde{y} \xrightarrow{\tilde{\omega}} \tilde{x}$ iff $(\exists \alpha \in Dat_x, \beta \in Dat_y : \omega \in L(x, \alpha) \cap L(y, \beta)) \vee (\omega \in$

$O(Rel_x) \cap O(Rel_y)$

– of *updating type* $\tilde{y} \xrightarrow{\tilde{\omega}^u} \tilde{x}$ iff $\exists x' \in X \ : \ x' \prec x$,

– of *generating type* $\tilde{y} \xrightarrow{\tilde{\omega}^g} \tilde{x}$ iff $\nexists x' \in X \ : \ x' \prec x$.

An information system of information retrieval R generates a multi-agent system with typed connections. Agents of the system resolve the ambiguities by computing and comparing the context cardinalities and evidence powers. Information system $(A, Con, \vdash)$ generates *Multi-agent System of Ambiguity Resolution* (MASAR) as a tuple $S = (A, C, I, T)$, where

– $A = \{a_x \mid x \in A^{\uparrow}\}$ is a finite set of agents corresponded to maximally determined instances;

– $C = \{\tilde{\omega} \mid \exists x, y \in A : \tilde{x} \xrightarrow{\tilde{\omega}} \tilde{y}\}$ is a finite set of connections;

– mapping $I : C \longrightarrow 2^{A \times A}$ is an interpretation function of ordered connections between agents: $I(c) = (a_x, a_y)$ iff $\tilde{x} \xrightarrow{c} \tilde{y}$;

– mapping $T : C \times A \times A \longrightarrow \{gen, upd\}$ is types of connections: $T(c, a_x, a_y) = gen$ iff $I(c) = (a_x, a_y) \rightarrow (\tilde{x} \xrightarrow{c^g} \tilde{y})$, and $T(c, a_x, a_y) = upd$ iff $I(c) = (a_x, a_y) \rightarrow (\tilde{x} \xrightarrow{c^u} \tilde{y})$. Let *(conflict) lexical agents* correspond to (conflict) lexical objects.

Not every instance from $A^{\uparrow}$ is used for ontology population. There is a set of utility instances Utl. They do not resolve ambiguities or populate an ontology. They just transfer information to its descendants. Hence $A^{\uparrow} = LO \cup Ont \cup Utl$, where only instances from Ont may populate an ontology.

For every agent $a \in A$ we define the following sets of agents and connections. We omit symmetric definitions of ancestors $Anc*$ (for $Des*$) and utility predecessors $UtP*$ (for $UtS*$) for the brevity:

– $C_a = \{c \in C | \exists a' \in A : (a, a') \in I_C(c) \bigvee (a', a) \in I_C(c)\}$ is connections of a;

– $Scg_a^c = \{a' \in A \mid (a, a') \in I_C(c) \wedge T(c, a, a') = gen\}$ is a set of generated successors by c connection;

– $Scu_a^c = \{a' \in A \mid (a, a') \in I_C(c) \wedge T(c, a, a') = upd\}$ is a set of updated successors by c connection;

– $Sc_a^c = Scg_a^c \cup Scu_a^c$ is a set of all successors by c connection;

– $Pr_a^c = \{a' \in A \mid (a', a) \in I_C(c)\}$ is a set of predecessors by c connection;

– $UtS_a^c = \{a' \in Utl \mid (a, a') \in I_C(c)\}$ is a set of utility successors by c;

– $Des_a^c = Sc_a^c \cup \bigcup_{a' \in Sc_a^c} Des_{a'}^c$ is descendants by c connection;

MASAR is a multiagent system of information dependencies. In these systems agents can use information from predecessors and can pass the (processed) information to successors. Hence $Des_a^c \cap Anc_a^c = \emptyset$, i.e. every connection has no cycle because of information transfer.

A weight of an agent corresponds to the number and the quality (in a case of generation) of its non-utility ancestors and descendants. For every $a \in A$

– $wt_{Pr}^a(c) = 1 + \sum_{a' \in Pr_a^c} wt_{Pr}^{a'}(c)$ is *the weight of connection ancestors*,

– $wt_{Sc}^a(c) = 1 + \sum_{a' \in Scg_a^c} wt(a') + \sum_{a' \in Scu_a^c} wt_{Sc}^{a'}(c)$ is *the weight of connection descendants*,

– $wt_{Ut(P/S)}^a(c) = 1 + \sum_{a' \in Ut(P/S)_a^c} wt_{Ut(P/S)}^{a'}(c)$ is *the weight of connection utility ancestors/descendants*,

– $wt(a) = 1 + \sum_{c \in C_a} (wt_{Pr}^a(c) + wt_{Sc}^a(c) - (wt_{UtP}^a(c) + wt_{UtS}^a(c)))$ is *the weight*

of information agents.
Weight of system S is $wt(S) = \sum_{a \in Ont} wt(a)$.

Problem of conflict resolution in MASAR is to get a conflict-free MASAR of the maximal weight. A multiagent algorithm below produces such system.

5 Conflict Resolution in MASAR

In this paper we consider independent ambiguities only. In this case an order of their resolution is irrelevant. But it is naturally to resolve lexical ambiguity first, because this disambiguation effects on existence of ontology instances. Syntactic disambiguation refines distribution of information among instances.

Action protocols for conflict resolution used by MASAR agents form a multiagent system of conflict resolution MACR. The system MACR includes a set of MASAR agents and an agent-master. Note, that a fully distributed version of our algorithm could be developed but it should be very ineffective. The result of agents' interactions by protocols described below is the conflict-free MASAR. All agents execute their protocols in parallel until the master detects termination. The system is dynamic because MASAR agents can be deleted from the system. The agents are connected by synchronous duplex channels. The master agent is connected with all agents, MASAR agents are connected with their successors and predecessors, and conflict lexical agents are connected too. Messages are transmitted via a reliable medium and stored in channels until being read.

For correct lexical disambiguation it is necessary to find groups of lexical agents which effect on weights of each other in a case of removing. Let us denote these *groups of relatives* as *Relatives*. Agents of groups from *Relatives* have common descendants: $\forall Rlt \in Relatives(\forall a \in Rlt(\exists b \in Rlt : Ds(a) \cap Ds(b) \neq \emptyset \wedge \forall c \in AmbLO \setminus Rlt : Ds(a) \cap Ds(c) = \emptyset))$. Due to the mutual effect of relatives on their weights it is necessary to resolve conflicts between groups of relatives. Let $GR_1 \subseteq Rlt_1$ and $GR_2 \subseteq Rlt_2$, where $Rlt_1, Rlt_2 \in Relatives$. *Relative groups* GR_1 *and* GR_2 *are in a conflict* $GR_1 \leftrightsquigarrow GR_2$ iff $(\forall a \in GR_1 \exists b \in GR_2 : a \leftrightsquigarrow b) \wedge (\forall b \in GR_2 \exists a \in GR_1 : b \leftrightsquigarrow a)$. Let sets $G_1 = \cup_{i=1}^{n} GR_1^i = \cup_{i=1}^{m} Rlt_1^i$ and $G_2 = \cup_{i=1}^{n} GR_2^i = \cup_{i=1}^{m} Rlt_2^i$, where $Rlt_1^i, Rlt_2^i \in Relatives$ for every $i \in [1..m]$ be *groups of friends*. These *groups of friends are in a conflict* iff $(\forall i \in [1..n] : GR_1^i \leftrightsquigarrow GR_2^i)$. Note that due to proposition 3 set *AmbLO* can be disjoined to nonintersecting subsets of relatives. *A conflict is resolved* for a benefit of the group with the greater weight, i.e. if $\sum_{a \in G_1} wt(a) > \sum_{b \in G_2} wt(b)$, then agents of group G_2 are removed from the system, and their descendants delete their inherited values of attributes or the descendant is removed itself if the lexical value from a lexical agent in G_2 is generating for this descendant.

Hence, for resolving all conflicts in the system it is necessary to perform the following steps: (1) to compute weights of agents, (2) to detect relative groups, (3) to compute independent conflict groups of friends, (4) to resolve lexical conflicts between the groups, (5) to make the corresponding change in the system, and (6) to resolve all kinds of syntactic ambiguity. An agent-master coordinates MASAR agents. It computes conflict groups and detects agents to be

removed. All other activities are performed by MASAR agents asynchronously. Due to parallel execution all computations take polynomial time.

(1) An interface protocol for system agents

This protocol specifies agent's reactions for incoming messages. These messages include information which actions should be performed by the agent: (1) `Start`: to start; (2) `CompWeight`: to compute its weight; (3) `FindRlt`: to find relatives; (4) `Remove`: to remove connections or itself; (5) `ResSynt*`: to resolve some syntactic ambiguity. Until an input message causes an agent to react the agent stays in a wait mode. Messages for an agent are stored in its input channel.

(2) The main algorithm for conflict resolution

Let us give an informal description of protocol `Master`. First, the agent-master computes set of lexical agents LO, then it finds set of conflict lexical agents $AmbLO$. After that it sends `Start` to all agents and launches parallel computing agents' weights and finding relatives for conflict lexical agents. After all agents finish their job, the master computes conflict groups of relatives, then detects conflict groups of friends. By comparing weights of conflict groups of friends, it forms a list of agents to be removed. After finishing of this resolution of group conflicts, the master launches the corresponding system changes. After termination of the changing, it initiates all kinds of syntactic disambiguation for instance agents in parallel.

Below we give informal descriptions of several protocols of the system agents.

(3) Computing agents' weight

Following the definitions of the weights agent a computes in parallel weights of (utility) descendants and (utility) ancestors by every connection $c \in C_a$, launching the corresponding subprocesses for each $c \in C_a$. These non-utility subprocesses send the weights of their descendants (ancestors) increased by 1 to predecessors (successors) respectively. Utility subprocesses do not increase the weights. If connection c is of type *gen* then the corresponding descendants' subprocesses send the weight of a to the predecessors. When these parallel computations are finished, the agent computes its own weight. The protocol of weights computing belongs to the class of wave echo algorithms [12].

(4) Computing agents' relatives

Let agents from $AbmLO$ be numerated. Computing relatives consists of two stages. Agents act asynchronously. (1) Pairwise search. Elder agent a using id of every younger agent b sends couple of ids $(a.id, b.id)$ to its descendants via its successors. If some descendant of a finds both numbers among its connections then it returns to a the id of b. After receiving agent a adds b to set of its relatives. Termination of this computation can be detected by AB-algorithm from [4]. (2) Merging. Elder agent a sends a request to every younger agent b for a set of its relatives $b.Rlt$. If $a.Rlt \cap b.Rlt \neq \emptyset$, then agent a merges both sets and agent b removes its set of relatives and stops its computation. After termination of the computation there are several agents with nonempty sets of relative groups.

(5) Removing LO-agents from the system

If agent a has to be removed from the system, then (1) all its predecessors remove all connections with it and delete a from sets of successors; (2) its descendants

remove a) all connections with it, b) the corresponding predecessors c) the corresponding attribute value; and d) if the removing connection is of generating type then the descendant has to be removed from the system.

Resolution of syntactic ambiguities Synt11 and Synt12 consists of two steps.

(6) Synt11 resolution.

(1) Ambiguity detection. Every agent, using sets of its successors, checks if some attribute value effects on values of several instances. If yes and these instances form a homogeneous group, and satisfy a predefined grammar condition, then it sends a message with the type of the conflict and the conflict value to every agent in the group excluding the first agent in the group. (2) Agents in the group resolve the ambiguities following the resolving formulas for Synt11. For this they compute an evidence power of the ambiguous value.

(7) Synt12 resolution.

(1) Ambiguity detection. Every agent, using sets of its successors, checks if some attribute value effects on values of several instances. If yes and these instances do not form a homogeneous group, and satisfy a predefined grammar condition, then it sends a message with the type of the conflict, the conflict value, and ids of the competitors to every agent in the group. (2) Agents in the group send their evidence power to the competitors. Then they resolve the ambiguities following the resolving formulas for Synt12.

(8) Synt112 resolution.

If an agent finds attributes ω_1 and ω_2 with value c then it compares evidence powers $EQ(a, \omega_1, c)$ and $EQ(a, \omega_2, c)$. The attribute value is removed from values of an attribute with the less power.

(9) Synt211 resolution.

If an agent finds attribute ω with values c_1 and c_2 then it compares evidence powers $EQ(a, \omega, c_1)$ and $EQ(a, \omega_1, c_2)$. The attribute value with the less power is removed from values of the attribute.

6 Conclusion

In this paper, we show that maximal instances of the ontology classes that take part in the process of population form, together with the rules of data processing and ontology population, a Scott information system. This result justifies resolution of context-dependent lexical ambiguity by calculating context cardinalities. The Scott information system is also a basis for our approach to syntactic context-dependent ambiguity resolution. This system generates a multi-agent system in which agents resolve the ambiguities by computing the cardinality of their contexts and evidence powers. The suggested algorithm of lexical ambiguity resolution chooses the most powerful group of agents and removes their competitors. The choice is based on agents' weights and their effect on the system.

We considered independent lexical and syntactic ambiguities only. In the near future we plan to study disambiguation of combination of various types syntactic and lexical ambiguities. In this work it is useful to introduce a membership

probability of attribute ambiguity values and a degree of their effect on other instances. We would like to try the developed technique for resolving references also.

References

1. **Alfawareh H.M., Jusoh S.** *Resolving Ambiguous Entity through Context Knowledge and Fuzzy Approach* // International Journal on Computer Science and Engineering (IJCSE). ISSN: 0975-3397, Vol. 3, No. 1, 2011. pp. 410–422
2. **Berry, D.M., Kamsties, E., Krieger, M.M.** *From contract drafting to software specification: Linguistic sources of ambiguity (2003)*, http://se.uwaterloo.ca/dberry/handbook/ambiguityHandbook.pdf (31.01.2016)
3. **Ganter B., Wille R.** *Formal Concept Analysis.* Mathematical Foundations. Springer Verlag, 1996.
4. **N. O. Garanina, E. V. Bodin.** *Distributed Termination Detection by Counting Agent* // Proc. of the 23nd International Workshop on Concurrency, Specification and Programming (CS&P 2014), Chemnitz, Germany, 29. September - 01. Oktober 2014. Humboldt-Universitat zu Berlin, 2014, pp. 69–79.
5. **Garanina N., Sidorova E., Bodin E.** *A Multi-agent Approach to Unstructured Data Analysis Based on Domain-specific Onthology* // Proc. of the 22nd International Workshop on Concurrency, Specification and Programming, Warsaw, Poland, Sept. 25-27, 2013. CEUR Workshop Proceedings, Vol. 1032, pp. 122–132
6. **Garanina N., Sidorova E.** *An Approach to Ambiguity Resolution for Ontology Population* // Proc. of the 24th International Workshop on CS&P. Rzeszow, Poland, Sep. 28-30, 2015. – University of Rzeszow, 2015, Vol. 1, pp. 134–145.
7. **Gleich B., Creighton O., Kof L.** *Ambiguity Detection: Towards a Tool Explaining Ambiguity Sources* // Proc. of 16th International Working Conference Requirements Engineering: Foundation for Software Quality, Essen, Germany, June 30–July 2, 2010, LNCS Vol. 6182, pp. 218-232.
8. **Kim D.S., Barker K., Porter B.W.** *Improving the Quality of Text Understanding by Delaying Ambiguity Resolution* // Proc. of the 23rd International Conference on Computational Linguistics, Beijing, 2010. pp. 581–589
9. **Navigli R.** *Word sense disambiguation: a survey.* ACM Computing Surveys, 41(2):1–69, 2009.
10. **Sidorova E., Kononenko I., Zagorulko Yu.** *Knowledge-based approach to document analysis* // International Journal "Information Technologies and Knowledge". Vol.2, No. 1, 2008. pp. 17–22.
11. **Spasic I., Zhao B., Jones C., Button K.** *KneeTex: an ontology-driven system for information extraction from MRI reports.*// J. Biomedical Semantics, 6, 2015, p. 34.
12. **Tel G.** *Introduction to Distributed Algorithms.* Cambridge University Press, 2000.
13. **Winskel G.** *The Formal Semantics of Programming Languages: An Introduction.* MIT Press, 1993.
14. **Zhang G.-Q.** *Chu Spaces, Concept Lattices, and Domains* // Electronic Notes in Theoretical Computer Science (ENTCS). Vol. 83, Jan 2013, pp. 287–302