Efficient Semantic Search using Finite Automata

P. Panagiotopoulos, M. Falelakis and A. Delopoulos

Department of Electrical and Computer Engineering, Aristotle University of Thessaloniki, Thessaloniki, Greece

Abstract

An efficient scheme for identifying Semantic Entities within data sets such as multimedia documents, scenes, signals etc. is proposed in this work. Expression of Semantic Entities in terms of Syntactic Properties is proved to be isomorphic to appropriately defined finite automata, which also model the identification procedure. Based on the structure and properties of these automata, formal definitions of attained Validity and Certainty and also required Complexity are defined as metrics of identification efficiency. The main contribution of the paper relies on organizing the identification and search procedure in a way that maximizes its validity for bounded Complexity budgets and reversely minimizes computational Complexity for a given required Validity threshold.

1. Introduction

The procedure of semantic search/indexing is essentially equivalent to the computation of the *degree* that a semantic entity (e.g. an event, an object, a concept etc) is identified within a particular environment (e.g. a multimedia document in the framework of MPEG-7, a scene in computer vision applications, a set of multi-sensor measurements in the case of surveillance systems etc). This type of computations and the resulting *identification degrees* correspond to fuzzy operations and membership values in our setup respectively. The entire computation procedure relies on a simple type of knowledge base, which in our framework contains formal definitions of all searchable semantic entities. In fact, a *hierarchical scheme* is adopted where each semantic entity is defined by decomposing it into either "simpler" semantic entities or elementary properties that can be quantified and are reserving the name *syntactic entities*. The aforementioned decomposition is assumed to obey the "modus ponens" approach, eg. a semantic entity A is decomposed to the (simpler) semantic and/or syntactic entities X, Y, Z in the sense that identification of any of X, Y, Z implies identification of A to a certain degree (relation value) $F_{AX}, F_{AY}, F_{AZ} \in [0,1]$ respectively. The whole collection of (i) Semantic entities definitions, (ii) Algorithms employed to quantify syntactic entities, (iii) Relation values, constitute what we call "Semantic Encyclopedia" (see also [3] and references therein for similar definitions of semantic encyclopedias).

Present work focuses on two main contributions regarding the use of such encyclopedias for semantic search and indexing. The first is the modelling of use of the aforementioned "semantic" hierarchical schemes by means of finite automata. The second is the design of efficient methods for the computation of identification degrees taking into account the tradeoff between limitations of computational cost (ie. algorithmic complexity) versus obtained validity of the identification. Our approach could be particularly important in real time and/or bulky search/indexing procedures where hard limits on computational budget are inevitable.

The next section is devoted to the formal definition of semantic and syntactic entities along with their attributes. Modelling of semantic search by means of finite automata is introduced in Section 3. Design of semantic search strategies is described in Section 5 on the basis of validity versus complexity measures defined in Section 4. Experimental results that clarify our approach and provide evidence for its advantages can be found in Section 6. Finally comments on the obtained results and a listing of open issues has been included in our last section.

2. Semantic Encyclopedia - Definitions

2.1. Syntactic Entities

As *syntactic feature t* we define any measurable quantity (eg. brightness, frequency, straightness etc) that can be obtained by applying a *corresponding algorithm* on a given data set (eg. a scene, an image, a signal etc). For simplicity we assume real valued syntactic features of either 1-dimension (eg. brightness on R) or multi-dimension (eg. color on R^3).

A syntactic entity or property $y_i(t)$ is a fuzzy set on a syntactic feature t. For instance the property "very bright" is defined on the feature "brightness" as illustrated in Figure 1. We assign the label Y_i to a particular Syntactic Entity $y_i(t)$ and assume a finite set $Y = \{Y_i\}$ of such labels corresponding to the entire collection of Syntactic Entities of interest.

It is essential to point out that aforementioned computa-

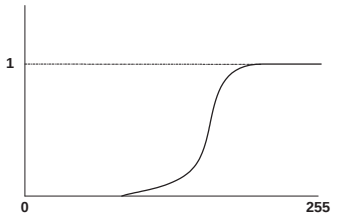


Figure 1: *The syntactic property "very bright".*

tional cost/budget refers to the algorithms τ employed for measuring the data set under examination in order to assess the degree $\mu \equiv y(\tau)$ up to which the particular data set assumes property $y(t)$.

2.2. Semantic Entities

Objects, events, categories or other concepts that may be handled by human perception/logic are collectively assigned the term *Semantic Entity*. In our discussion we assume a set E of Semantic Entities of interest with a further assumption that each entity with label $E_i \in E$ can be "described" on the basis of other Semantic Entities within E and/or Syntactic Entities within Y in a manner explained below, in Section 2.3. Note that E and Y form the building blocks of the Semantic Encyclopedia.

Provided that description are not cyclic, every Semantic Entity can be fully described by gradually decomposing it into lower level Entities (either Semantic or Syntactic).

We also assume that more than one different descriptions for each entity may be available.

Figure 2 illustrates three descriptions of the form $A \rightarrow \{a, C\}$ or $A \rightarrow \{a, d\}$ and $C \rightarrow \{b, c\}$, where $\rightarrow$ denotes that the LHS is described on the basis of the RHS entities and using the convention that lower (upper) case labels correspond to syntactic (semantic) entities.

Substitution C with by its own description, yields two alternative descriptions for A, namely: $A_1 \rightarrow \{a, b, c\}$ and $A_2 \rightarrow \{a, d\}$

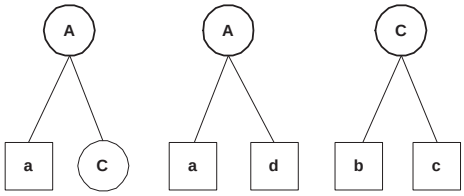


Figure 2: *Descriptions.*

2.3. Definitions

The presented qualitative description of a Semantic Entity on the basis of simpler entities can be enriched by more quantitative information regarding the degree of relation between a Semantic Entity and its successors.

Since alternative descriptions of an entity E_k provide different amount of information about it, we define as *validity* m_{kJ} of a description J of E_k a real number in [0,1] measuring the amount and the quality of the information provided. Equivalently m_{kJ} is the degree up to which the particular description characterizes E_k.

In addition, Entities (either Syntactic or Semantic) that are included in a description have different importance quantified by a set of corresponding weights. These weights can be considered as elements of a fuzzy relation on $S \times S$, where $S \equiv Y \bigcup E$ (see [3] for a similar discussion). For a particular Semantic Entity $E_k \in E$ we define $F_{kJ} : S - \{E_k\} \rightarrow [0,1]$ for those $S_i \in S$, participating in a certain description J (one of possible alternatives).

We define as a *primary definition* of E_k in terms of J the discrete fuzzy set

$$E_{kJ} = F_{kJ1}/S_1 + F_{kJ2}/S_2 + ... + F_{kJn}/S_n \qquad (1)$$

As mentioned above, by replacing the semantic entities by their respective descriptions and by repeating this procedure recursively, one can base the primary definition of an entity on syntactic characteristics only. The substitution of a Semantic Entity S_j in Equation 1 by its own description involves application of fuzzy operations between the corresponding weights and validity coefficients. In order to avoid complicated expressions necessary to describe substitution rules of the general case, we quote here a simple example corresponding to the definition of Figure 3. According

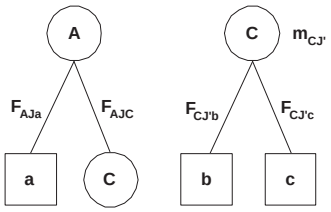


Figure 3: *A definition example*

to that, $A = F_{AJa}/a + F_{AJC}/C$ and $C = F_{CJ'b}/b + F_{CJ'c}/c$. Transforming this to a primary definition we obtain

$$\begin{aligned} A \;=\; & F_{AJa}/a + \mathcal{I}(\mathcal{I}(m_{CJ'}, F_{CJ'b}), F_{AJC})/b \\ & + \mathcal{I}(\mathcal{I}(m_{CJ'}, F_{CJ'c}), F_{AJC})/c \end{aligned} \qquad (2)$$

Operations $\mathcal{U}$ and $\mathcal{I}$ denote fuzzy union (t-conorm) and intersection (t-norm) respectively. The substitution procedure yields a number of alternative definitions of E_k on the basis of syntactic only entities of the form of Equation 3

$$E_{kJ} = F_{k1}/Y_1 + F_{k2}/Y_2 + ... + F_{km}/Y_m\,. \qquad (3)$$

In view of Equation 3, our initial problem of identifying E_k on the basis of available data reduces to running algorithms that evaluate the degree up to which Syntactic Entities $Y_i, i = 1 \cdots m$ appear within this data. These degrees are next combined with Equation 3 to provide the identification degree of Entity E_k due to each particular definition E_{kJ}. An overall identification degree is next obtained by combining the identification degrees of all available alternative definitions.

3. Modelling via Automata

3.1. Elementary Automaton

The simplest possible definition of the form of Equation 3 is the one containing a single alternative based on a single Syntactic Entity: $A = F_{Aa}/a$. In order to identify A, we need to run only algorithm $a \in \Omega$, where Ω is the set of all available algorithms and we choose to label the algorithm borrowing the names of the corresponding Syntactic Entity. Consider a scenario where multiple algorithms of Ω are invoked in the course of obtaining information from the available data in a sequential manner. Identification of A begins only when a is invoked and actually completes when a has finished. This procedure can be represented by the *elementary automaton* depicted in Figure 4.

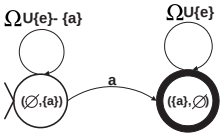


Figure 4: *An elementary automaton.*

3.2. Augmented Automaton

It turns out that definition of the general form of Expression 3 can be modelled by finite automata resulting as intersections ([1]) of elementary automata corresponding to each single term, F_{ki}/Y_i, of the expression. It can be also proved that if more than one alternative descriptions E_{kJ} are available for a Semantic Entity E_k, the identification procedure can be modelled by the union of the finite automata corresponding to each alternative definition. The initial state of the resulting *augmented automaton* is followed by a number of independent branches, '1-1' corresponding to all available alternative definitions. Each branch has its own single final state which is reached when all algorithms corresponding to a particular definition have been invoked. Each state of the automaton corresponds to the evaluation of a subset of syntactic properties involved in the definition of an entity, or, equivalently, to the execution of a subset A of the corresponding algorithms Ω. Since more than one alternative definitions may rely on the same syntactic properties, when evaluating a subset of them, the automaton is nondeterministically going over to more than one states simultaneously. For instance, should the following definitions for A be given: $A_1 = \{a,b,c\}$ and $A_2 = \{a,d\}$, the search for A is shown in Figure 5. The empty symbol e in Figure 5 denotes non-deterministic "in vacuo" transition (ie. change of state without invocation of any algorithm). Each state of the automaton is labelled by an ordered pair (A,B), where A denotes the set of algorithms already run to reach this state of identification procedure and B the set of algorithms pending in order to complete evaluation of a certain alternative definition.

4. Metrics of Search and Identification Procedures

Three types of metrics are introduced to characterize the identification procedure, namely validity, complexity and certainty. The first two depend on the content of the semantic encyclopedia while the third one relies on the data under examination.

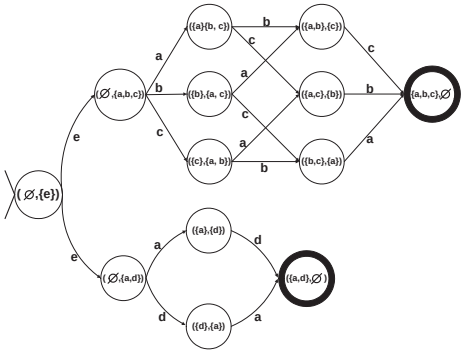


Figure 5: *An augmented automaton representing a semantic search.*

4.1. Validity

Each state $q = (A,B)$ of the augmented automaton represents a "partial description" of a Semantic Entity, attained by using the syntactic characteristics included in A. We define the *validity* of this state, as the amount of information already gathered by running the corresponding algorithms. If m_{kJ} is the "reliability" of the primary description J of E_k, the validity of q is defined as:

$$\underset{J}{v}(E_k/q) \equiv \underset{J}{v}(E_k/(A,B)) = \mathcal{I}(\, m_{kJ},\, \underset{t\in A}{\mathcal{U}}\,(\,F_{kt}\,)\,) \qquad (4)$$

where t-conorm $\mathcal{U}$ is deliberately used as a multiple argument operator due to its associativity property [2].

On Equation 4 we can comment the following:

- a partial description cannot be more valid than the primary one
- validity cannot be reduced as we traverse the automaton. Thus, further consideration of syntactic characteristics can only increase the validity of the description.

Taking into account that running a set of algorithms may simultaneously lead to more than one states (ref. Section 3.2), the obtained *total validity* is expressed as the fuzzy union of the validity of these states, i.e.,

$$V(E_k/Q) = \underset{q\in Q}{\mathcal{U}}\,[\underset{J}{v}(E_k/q)] \qquad (5)$$

where $Q = \{q_1, q_2, ..., q_n\}$

For instance when traversing the automaton of Figure 5 and having used algorithms a and b, we reach states $q_1 = (\{a,b\},\{c\})$ and $q_2 = (\{a\},\{d\})$, thus $Q = \{q_1, q_2\}$ which yields

$$V(E_k/Q) = \mathcal{U}[(\mathcal{I}(m_{kJ_1}, \mathcal{U}(F_{kJ_1a}, F_{kJ_1b}))), (\mathcal{I}(m_{kJ_2}, (F_{kJ_2a})))]. \qquad (6)$$

4.2. Certainty

Certainty quantifies the degree of our belief that a Semantic Entity, E_k, has been identified within a data set. The value

of this metric certainly depends on the results of those algorithms employed to evaluate syntactic properties appearing in the definitions of E_k. Within our modelling approach a particular certainty value is attained moving from state to state of the augmented automaton. In a formal manner, certainty of a state $q = (A,B)$ is defined as

$$\underset{J}{\mu}(E_k/q) \equiv \underset{J}{\mu}(E_k/(A,B)) = \mathcal{I}(\, m_{kJ},\, \underset{t\in A}{\mathcal{U}}\,(\mathcal{I}(F_{kt}, \mu(t)))) \quad (7)$$

where $\mu(t)$ denotes the degree up to which the data set assumes a particular syntactic property evaluated by running the algorithm t. We observe that for every state q, the resulting certainty is no greater than its validity. Thus

$$\underset{J}{\mu}(E_k/q) \leq \underset{J}{\nu}(E_k/q) \quad (8)$$

In correspondence with total validity, while searching for the entity E_k we define as *total certainty* of a set of simultaneous states $Q = \{q_1, q_2, ..., q_n\}$ the fuzzy union of the certainty of the participating states:

$$\mu(E_k/Q) = \underset{i=1,...,n}{\mathcal{U}} [\underset{J}{\mu}(E_k/q_i)] \quad (9)$$

It's evident, because of Equations 5, 8 and 9, that the total certainty of a state is less than or equal to its corresponding validity:

$$\mu(E_k/Q) \leq V(E_k/Q) \quad (10)$$

4.3. Validity vs Certainty

We must point out the fact that validity is computed a priori for every state and depends only on the reliability of the given definitions. On the other hand, certainty is dynamically computed, while traversing the automaton and it is in accordance with the results of the already ran algorithms. Furthermore certainty cannot be greater than validity, which means that we cannot be sure about the existence of an entity, without having taken into consideration enough information, which in turn would make the answer more valid.

4.4. Complexity

The computational cost of reaching a state $q = (A,B)$ equals to the overall complexity of those algorithms contained in A. Hence we define as *complexity* of a state $q = (A,B)$, where $A = \{t_1, t_2, ..., t_n\}$ the algorithms required to reach this state, the sum of the complexity of each algorithm:

$$\mathcal{C}((A,B)) = \sum_{t\in A} \mathcal{C}(t) \quad (11)$$

Similarly we define as *total complexity* of a set of states $Q = \{(A_1,B_1),(A_2,B_2),\cdots\}$, the quantity

$$\mathcal{C}_{total}(Q) = \sum_{t\in \bigcup_J A_J} \mathcal{C}(t) \quad (12)$$

5. Design Methodologies

Towards the goal of obtaining a search result which gives maximum validity with minimum computational complexity we propose here two design methodologies for the search procedure. Those are used a priori (before any algorithm is used) to determine the sequence of algorithms to be run.

However, the fact that, while traversing the augmented automaton, a set of algorithms can lead us to more than one states simultaneously, implies extensive computation of the characteristics of complex states and thus increased computational complexity.

In order to overcome this we next introduce another structure for the representation of the search process.

5.1. Equivalent Augmented Automaton

Considering an augmented automaton M_1 which depicts the alternative descriptions $\Omega_1, \Omega_2, ..., \Omega_n$ of an entity E_k then we form an *equivalent augmented automaton* M_1 which includes only one description, namely $\Omega = \Omega_1 \cup \Omega_2 \cup ... \cup \Omega_n$.

Each state (A,B) of M_2 is a mapping of a set of states of M_1, $(A,B)_i$, which we reach by running the algorithms belonging to A.

$$(A,B)_i = (\Omega_i \cap A\,,\, \Omega_i \cap B) \quad (13)$$

In correspondence with Equations 4, 5 and 13, we define as validity of (A,B) of M_2, the total validity of $(A,B)_i$ of M_1:

$$\begin{aligned} \nu(E_k/(A,B)) &= \underset{i}{\mathcal{U}}(\underset{J}{\nu}(E_k/(A,B)_i)) \\ &= \underset{i}{\mathcal{U}}(\underset{J}{\nu}(E_k/((\Omega_i \cap A\,,\, \Omega_i \cap B)) \end{aligned} \quad (14)$$

Certainty arises in similar manner , according to Equations 7, 9 and 13:

$$\mu(E_k/(A,B)) = \underset{i}{\mathcal{U}}[\mu_i(E_k/((\Omega_i \cap A\,,\, \Omega_i \cap B)))] \quad (15)$$

Finally, observing that $(A,B) = \underset{i}{\mathcal{U}}(A,B)_i$, complexity is defined as:

$$\mathcal{C}((A,B)) = \underset{i}{\mathcal{C}_{total}}((A,B)_i) = \underset{i}{\mathcal{C}_{total}}((\Omega_i \cap A\,,\, \Omega_i \cap B)) \quad (16)$$

Thus the equivalence between M_1 and M_2 stands on the fact that by running the same set of algorithms, we achieve a description of the same validity, certainty and complexity.

Considering the automaton of Figure 5, its equivalent is depicted in Figure 6

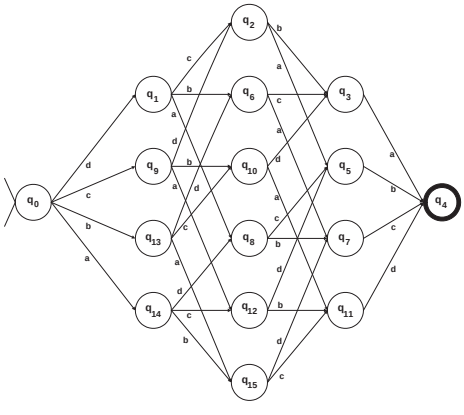


Figure 6: *An Equivalent Augmented Automaton.*

5.2. Design in terms of Validity

Supposing that we want to obtain a "valid" result while searching for an entity, we define a validity threshold, M, under which no answer is accepted.

We first find all the states of the constructed equivalent augmented automaton that satisfy this criterion:

$$Q_{kM} = \{q/v(E_k/q) \geq M\} \tag{17}$$

We next choose the state which requires the less complexity. Thus:

$$q_0 = \underset{q \in Q_{kM}}{minimizer}\, \mathcal{C}(q) \tag{18}$$

It is evident that the following restriction must be applied

$$M \leq \underset{i}{\mathcal{U}}[\underset{J}{\mathcal{I}}(E_k/f_i)] \tag{19}$$

so as to avoid the set Q_{kM} being empty.

Thus, it suffices to run the set of algorithms that leads to the state q_0, regardless of the order of execution.

5.3. Design in terms of Complexity

An alternative way of design results when the search is constrained by a particular time/complexity threshold $C > 0$. In this case, a set of states Q_{kC} which suffice this constraint is found:

$$Q_{kC} = \{q/\mathcal{C}(q) \leq C\} \tag{20}$$

From the set Q_{kC}, q_0 is chosen which provides maximum validity:

$$q_0 = \underset{q \in Q_{kC}}{maximizer}\, v(E_k/q) \tag{21}$$

Once again, the order of execution of the algorithms plays no role.

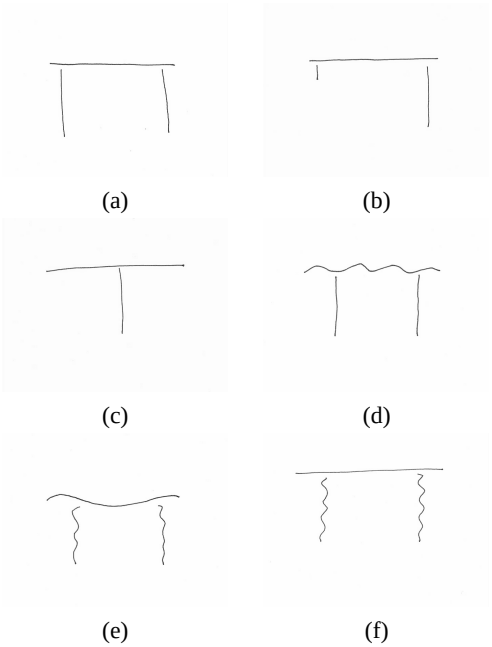
5.4. An incremental Scheme

A modification of the methodology arises if we choose to begin the search using a low threshold of complexity and to continue the search only should we receive satisfactory certainty results. This approach would be proved to be particularly useful in case there is a multiplicity of entities $E_k, (k = 1, ..., N)$ to be identified at real time, with a limited complexity budget. In that case, the algorithms that belong to the initial set chosen to be run, can be considered as *triggers*. Initiation of further inspection of the document (by employing more syntactic features and the corresponding algorithms) is based upon the triggers' results.

6. Experimental Results

Two different types of tests were carried out in order to evaluate the performance of the proposed methodology.

First Experiment During the first experiment, the system was given the images shown in Figure 7 and it searched for the entity "table". The definition of "table" was given, as shown in Figure 8, $E_{01} = 0.9/Y_{01} + 0.7/E_{02}$ and $E_{02} = 0.6/Y_{02} + 0.9/Y_{03} + 0.8/Y_{04}$.



(a) (b) (c) (d) (e) (f)

Figure 7: *Drawings of "tables"*

As a fuzzy intersection operator,the product was chosen:

$$\mathcal{I}(a,b) = ab \tag{22}$$

And its complementary, the algebraic sum, for union:

$$\mathcal{U}(a,b) = a + b - ab \tag{23}$$

Composing the two descriptions as described above, the following primary definition of "table" is obtained:

$$E_{01} = 0.9/Y_{01} + 0.378/Y_{02} + 0.567/Y_{03} + 0.567/Y_{04} \tag{24}$$

For each syntactic property an estimate of its complexity was experimentally obtained as shown in the following list. Complexity units in this list correspond to 10^3 FLOPS.

Algorithm	Complexity
Horizontal surface	$\mathcal{C}(1) = 3.6$
Two straight lines	$\mathcal{C}(2) = 4.8$
Two vertical lines	$\mathcal{C}(3) = 4.5$
Two lines of same length	$\mathcal{C}(4) = 3.3$

Results of design in terms of validity are illustrated in the first four columns of Figure 9, while the corresponding attained certainty values for each drawing (a)-(f) have been included in the next six columns. Rows of the table correspond to design setting validity threshold to $M = 0.2, 0.46, 0.73, 0.785$. Two comments are worth to be made: (1)Modifying M results in selection of different algorithms (see e.g. rows one and two). (2) Relatively high validity and certainty is obtained at reasonably low computational cost, but pushing the validity threshold to its high levels causes abrupt increase of the required complexity. Similarly, design results in terms of complexity have been included in the table of Figure 10 for complexity bounds $C = 3.7, 8, 13, 7$. Commenting on these results, decent validity levels are attained even under low

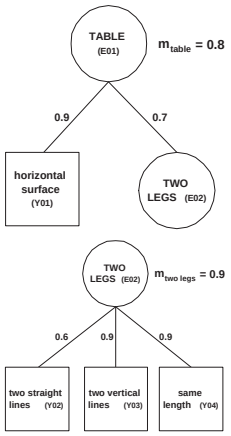


Figure 8: *Description of the entity "table".*

/

M	$Validity$	$Complexity$	$Algorithms$	(a)	(b)	(c)	(d)	(e)	(f)
0.2	0.45	3.3	4	0.4	0.09	0	0.438	0.432	0.403
0.46	0.72	3.6	1	0.705	0.7	0.675	0.396	0.188	0.685
0.73	0.77	6.9	1, 4	0.753	0.703	0.675	0.617	0.519	0.743
0.785	0.79	16.2	1, 2, 3, 4	0.782	0.763	0.675	0.73	0.64	0.765

Figure 9: *Design in terms of validity.*

complexity constraints. Allowing higher complexity budgets enhances both validity and certainty but the gained increase is not proportional to the additional computational cost. The results from both design strategies indi-

/

C	$Validity$	$Complexity$	$Algorithms$	(a)	(b)	(c)	(d)	(e)	(f)
3.7	0.72	3.6	1	0.705	0.7	0.675	0.396	0.188	0.685
8	0.7654	6.9	1, 4	0.753	0.703	0.675	0.617	0.519	0.743
13	0.7827	11.4	1, 3, 4	0.771	0.746	0.675	0.692	0.572	0.752
17	0.79	16.2	1, 2, 3, 4	0.782	0.763	0.675	0.73	0.64	0.765

Figure 10: *Design in terms of complexity.*

cate that efficient policies can be adopted for optimal use of resources by balancing between complexity and validity.

Second Experiment An Equivalent Augmented Automaton was created, as the intersection of 9 elementary automata where the syntactic properties had random, uniformly distributed, weights. We also considered a uniformly distributed in [1,6] complexity of the corresponding algorithms. The total complexity was 32.33 (for exhaustive search of all syntactic properties) which corresponds to validity 1. In Figure 11 one can see the variation of the essential complexity, for different values of the validity threshold, when designing in terms of validity. In Figure 12 the corresponding results are shown for the design in terms of complexity.

In both cases, the obtained results confirm the observations of the first experiment.

7. Conclusions and Future Work

In this paper we described a method for performing efficient semantic search in setups of limited complexity budgets. The theoretic analysis yielded tools for balancing between search accuracy (validity) and corresponding computational

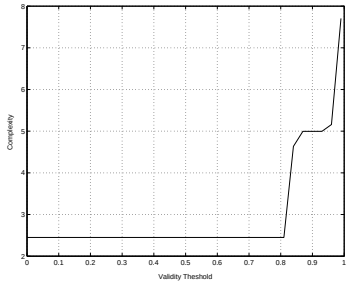


Figure 11: *Design in terms of validity.*

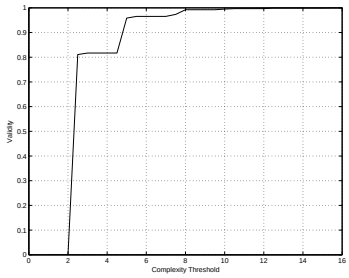


Figure 12: *Design in terms of complexity.*

cost (complexity). A couple of experiments confirmed the value of the proposed methodology.

Our analysis heavily relies on the definition of the so-called Equivalent Augmented Automaton (EAA) used to model the procedure of semantic search. Optimization as described by Equations 17 and 20 is equivalent to appropriate parsing EAA and finding a cut corresponding to Equations 18 and 21 respectively. The size of EAA expands exponentially with respect to the number of syntactic properties. One of our next goals is certainly to develop efficient graph traversal algorithms for solving this problem. It seems that they should take into account the fact that both validity and complexity are non-decreasing functions of the number of the involved algorithms. The need of such fast optimization methods becomes more apparent if we choose to comply with requirements regarding the attained certainty levels, dynamically, during the search.

References

[1] Harry R. Lewis and Christos H. Papadimitriou. *Elements of the Theory of Computation.* Prentice Hall, 1998

[2] George J. Klir and Bo Yuan. *Fuzzy Sets and Fuzzy Logic; Theory and Applications.* Prentice Hall, 1995

[3] G. Akrivas, G. B. Stamou and S. Kollias. Semantic Association of Multimedia Document Descriptions through Fuzzy Relational Algebra and Fuzzy Reasoning *IEEE Transactions On Systems, Man and Cybernetics, part A.*, **34** 2004