TESTING ACCESS TO EXTERNAL INFORMATION SOURCES IN A MEDIATOR ENVIRONMENT*

Zoltán Ádám MANN
Budapest University of Technology and Economics
Department of Control Engineering and Information Technology
zoltan.mann@cs.bme.hu

Jacques CALMET and Peter KULLMANN
University of Karlsruhe
Institute for Algorithms and Cognitive Systems
{calmet,kullmann}@ira.uka.de

Abstract This paper discusses the testing of communication in the increasingly important class of distributed information systems that are based on a mediator architecture.

A mediator integrates existing information sources into a new application. In order to answer complex queries, the mediator splits them up into subqueries which it sends to the information sources, and it combines the replies to answer the original query. Since the information sources are usually remote, autonomous systems, the access to them can be erroneous, most notably when the information source is subject to modifications. Such errors may result in incorrect behaviour of the whole system. This paper addresses the problem of deciding whether an information source as part of a mediatory system was successfully queried or not.

The primary contribution is a formal framework for the general information access testing problem. Besides proposing several solutions, it is investigated what the most important quality measures of such testing methods are. Moreover, the practical usability of the presented approaches is demonstrated on a real-world application using Web-based

*This paper was published in the Proceedings of the 14th IFIP International Conference on Testing Communicating Systems, 19-22 March 2002, Berlin (Germany). Published by Kluwer Academic Publishers

information sources. Several empirical experiments are conducted to compare the presented methods with previous work in the field.

Keywords: mediator, information access testing, wrapper verification, information integration, autonomous information sources

1. Introduction and previous work

In the past decades a tremendous amount of data has been stored in electronic form. In recent years, as a consequence of the unbelievable evolution of the World Wide Web, practically any information one can imagine can be found in one format or the other in the WWW.

However, this is not enough for the requirements of the future information society. The problem is not the amount of available information, which is already more than sufficient, but its usability. Information sources (ISs) are designed for their particular purposes, but need to be reused in completely different applications, in conjunction with other pieces of information. This not only holds for the Web but also for a variety of other ISs, such as relational and object-oriented databases, electronic documents, computer algebra systems, specialized software libraries, *etc.*

Mediators. To address the problem of integrating heterogeneous, autonomous ISs, Wiederhold suggested the *mediator* pattern in [16]. The mediator implements the common tasks of splitting complex queries into simpler ones that can be sent to the underlying ISs and combining the replies to answer the original query. The latter also includes the detection and handling of potential conflicts that may arise if two ISs return contradictory results.

The ISs are bound into the mediator architecture via *wrapper*s: components that translate queries from the language of the mediator into that of the ISs and the answers from the language or the data model of the IS into that of the mediator. The resulting architecture is depicted in figure 1. More information on mediators can be found *e.g.* in [6, 15].

The context of the work presented in this paper was provided by KOMET (Karlsruhe Open MEdiator Technology [1]), a logic-based mediator shell developed at the University of Karlsruhe. KOMET uses the declarative language KAMEL (KArlsruhe MEdiator Language), which is based on annotated logic. It provides a framework for the easy construction of mediators, and enables the reuse of existing code.

A particular mediator system, called MetaSearch [2], which is implemented using KOMET, was of special interest. MetaSearch is a meta Web search program that takes queries and delegates them to various

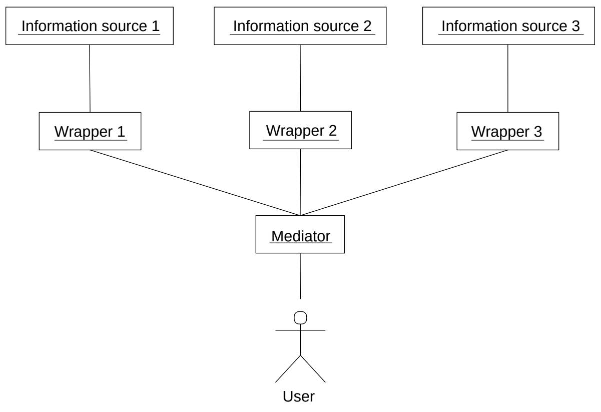

Figure 1 The mediator architecture

Internet search engines such as AltaVista and Google. It then combines the answers of the search engines into one answer page.

MetaSearch is very important for two reasons. First, with the spread of the World Wide Web, the integration of Web-based ISs becomes a major challenge and the most important application for mediator systems, and MetaSearch can be regarded as a prototype of such applications. Second, although MetaSearch is a relatively simple mediator application, it demonstrates very well the features of KOMET and also the problems that arise as a consequence of integrating remote, autonomous ISs.

Since the actual access to the external ISs is performed in the wrappers, they are of special importance from this paper's point of view. In the case of MetaSearch, the conversion from the language of the mediator into that of the IS is simple: it boils down to encoding the query into a URL. The conversion of the answer from the language of the IS into that of the mediator is somewhat more challenging, since the actual answer – the URL, title and excerpt of about 10 relevant pages – is embedded into an HTML page, along with plenty of other data, such as banners and statistics. Therefore the task of the wrapper is to extract the actual answer from the resulting page. In MetaSearch, regular expressions are used for that purpose.

Information access testing. Although regular expressions are not the only possibility for the extraction of information from HTML, this method illustrates the problem well: the information extraction mechanism has to rely on some regularity of the output of the IS. Since the IS is an autonomous system, it may be altered, which may in turn prevent the wrapper from extracting correctly. Thus, the aim of this paper is to investigate the problem of testing communication (*i.e.* access to the external ISs) in the special but increasingly important class of distributed information systems using a mediator-like middleware platform.

There has already been some research on this problem, though not much. The most relevant work is that of Kushmerick [8, 9]. His algorithm Rapture uses statistical features, such as length, number of words, number of special characters *etc.* to characterize the extracted text segments. It learns the parameters of normal distributions describing the feature distributions of the extracted pieces of texts. These normal distributions are used to estimate the probability that a new wrapper output is correct.

Much of the remaining scientific work focuses on the automatic creation of wrappers (see *e.g.* [5, 13] and references therein), but there are also some results that can be used for the wrapper testing problem as well. Cohen [3] uses a notion of textual similarity to find "structure" in Web pages: this is useful mainly in wrapper induction but may also be used in wrapper testing and maintenance, because it can detect changes in the structure of the HTML page. Lerman *et. al.* developed DataPro [10], an algorithm that uses tokens (words or generalizations of words) to represent text, and learns significant sequences of tokens – sequences that are encountered significantly more often than would be expected by chance. Thus the tuples to be extracted can be characterized by a set of significant token sequences, and a change in them can help discover changes in the HTML layout.

It can be seen that although there are already some promising results, the problem of IA (information access) testing in a mediator environment needs more thorough research because the existing results apply only to a very restricted problem class. The most important restrictions are: (i) only Web-based applications have been investigated; (ii) only errors produced by layout changes have been covered; (iii) only syntactic testing methods for the wrapper output have been proposed. Accordingly, this paper addresses the more general question: how to check whether or not the access to an external IS was correct?

Main results and paper organization. One of the main contributions of the paper is a formal notational framework that enables the definition of the information access testing problem (IATP) in its most general form (section 2). Existing results can be placed naturally in this framework. Moreover, it enables proving the hardness of the IATP (Theorem 1).

Another major problem with the existing results is that although the proposed methods can detect almost every change in the HTML layout, they often give false positives, *i.e.* they relatively often claim correct wrapper outputs to be erroneous. Section 3 investigates how the quality of IA testing can be measured, and demonstrates why the high rate of false positives is intrinsic (Theorem 2).

In section 4, several methods are presented that can be used generally for IA testing in a mediator environment. In particular, the applicability in a Web-based setting is also covered. Section 5 is a case study: it illustrates how the presented methods can be applied in practice, namely in the case of MetaSearch. It is demonstrated with several measurements, how – with appropriate techniques – better testing quality can be achieved. Section 6 concludes the paper.

2. Problem definition

This section provides a general notational framework for the investigation of the information access testing problem (IATP). Note that the framework will have to cope with the following difficulties: (i) it will have to allow and tolerate some degree of ambiguousness in the replies of ISs; (ii) the testing theory of ISs as systems under test is different from the better understood case of finite state machines (FSM) because of the unavailability of state and specification information.

In order to make the definitions clear, figure 2 shows the whole communication model.

Definition 2.1 (Messages on communication channels) *Let the alphabet of the communication system be the finite set Σ. Thus, messages are elements of Σ^*. ε, the empty string, is also an element of Σ^*. Let ν be a symbol with $\nu \notin \Sigma^*$. ν means 'no message'.*

Note that $\nu \neq \varepsilon$, *i.e.* we differentiate between an empty message and 'no message'.

The following definition describes the interface of ISs. Note that – since ISs are assumed to be completely autonomous – we have no knowledge about their internal state. The only thing we know is that an IS accepts queries, to which it (usually) gives a reply.

Definition 2.2 (Information source) *An information source is a family of functions $I_t : \Sigma^* \to \Sigma^* \cup \nu$ $(t \in \mathbb{R})$. $I_t(q) = r, r \in \Sigma^*$ means that, at time t, the reply of the IS on query q is r. On the other hand, if $r = \nu$, this means that the IS gave no reply.*

Definition 2.3 (Wrapper) *A wrapper W is a pair of Turing machines, $W = (T, E)$. The translator T implements a function $F_T : \Sigma^* \to \Sigma^*$, translating queries. The extractor E implements a function $F_E : \Sigma^* \cup \nu \to \Sigma^*$, extracting the replies of the IS.*

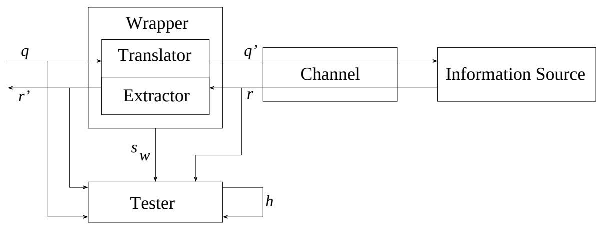

Figure 2 The underlying model

Note that, in contrast to ISs, wrappers have a known internal structure (Turing machine). Moreover, they do not change over time. It is also important to note that even if the IS gives no reply, the wrapper has to return something, *i.e.* $F_E(\nu) \in \Sigma^*$. (Although not stated explicitly in the definition, it is assumed that the wrapper can also aggregate and reuse information from its usage history, and thus perform learning.)

Now the task is to implement the IA tester (IAT, or simply: tester), *i.e.* the function that can tell whether the access was correct or not:

Definition 2.4 (Information access tester) *An IAT is a function $V : \Sigma^* \times (\Sigma^* \cup \nu) \times \Sigma^* \times S_W \times H \to \{correct, incorrect\} \times H$, where $V(q, r, r', s_W, h)$ is the judgement of the tester on the wrapper output r', extracted from the reply r of the IS, given on query q; s_W is any state information of wrapper W, and h is any additional 'historical' information that the tester has collected over time, which is also updated as part of the IAT output.*

It can be seen from this definition that the tester may make use of many observations and different kinds of knowledge; however, it does not have to. As noted in the Introduction, previous work has focused on IA testing using only the wrapper output r'.

The definition assumes that wrappers are not autonomous (as opposed to ISs) so that the IAT has access to their state information. So wrappers are 'observable' Turing machines.

The above definition allows many different testers; the goal is to create the best one in some sense:

Definition 2.5 (Information access testing problem) *Let U be the set of possible testers, $f : U \to \mathbb{R}$ the objective function. The general IATP consists of constructing the tester V^* which maximizes f on U.*

It would be logical to use the objective function

$$f_0(V) = \begin{cases} 1 & \text{if } V'\text{s judgement is always right} \\ 0 & \text{otherwise} \end{cases}$$

which would mean finding the *perfect* tester. However, this is infeasible:

Theorem 1 *The perfect tester has to be able to tell if the IS or the wrapper has fallen into an infinite loop. Therefore, the perfect tester would solve the Halting Problem of Turing machines, which is not algorithmically solvable.* □

Consequently, we will have to settle for a less ambitious objective. That is, we are looking for methods that work well in most practical cases. To that end, we suggest more realistic objective functions for the evaluation of testers in section 3. Moreover, we have surveyed the typical errors that have to be coped with during the access to external information sources in [12].

Note that the proof of Theorem 1 cannot be made fully precise until the notion of the 'perfect tester' is formalized. This is an important future research issue. Moreover, it might be argued that the IAT already assumes the wrapper output in its input, so there is no need to detect infinite loops. However, this is not true since one of the tasks of the IAT is exactly input handling, *i.e.* deciding if it should wait longer for its input or take it as ν. (Note that it is required that the IAT produce its output after a finite amount of time.)

Finally, one more definition is given. A restricted set of ISs, namely search engines, is of special importance in the paper:

Definition 2.6 (search engine) *A search engine is an IS with* $I_t(q) = (T, U, X)^c$*, where* c *is the number of tuples returned, and each tuple consists of the title* T*, the URL* U*, and the excerpt* X *of a Web page.*

3. Quality measures of IA testing

Since it is infeasible to strive for a perfect tester, the aim should be to construct one that is as 'good' as possible. This section tries to formalize the word 'good'. The benefit of this is threefold: (i) the presented quality measures can guide the construction of high-quality testers; (ii) a well-defined comparison of IA testing mechanisms (to be presented in section 4) is made possible; (iii) the introduced formalism sheds some light on common problems with the methods presented in the literature, and it can be proven why this is intrinsic.

If it is infeasible to expect that the judgement of the tester always be correct, it is a natural requirement that its correctness should be as high as possible, in a statistical sense. In order to formalize this, assume that the tester performs N tests, *i.e.* it tests N accesses of a wrapper W to an external IS I_t. The number of flawless accesses is k, the number of accesses during which an error occurred is $N-k$. In each test, the tester must give a judgement whether or not the access was erroneous. The number of tests in which the judgement of the tester is right is denoted by m, thus the number of tests in which the judgement of the tester is false, is $N - m$.

Definition 3.1 (statistical correctness) *The statistical correctness of the tester is $SC = m/N$.*

What does this number express? For instance, is a SC of 0.9 bad or good? Is a SC of 0.999 much better than that of 0.99? The answer to these questions depends on the value of k/N. Generally, it can be assumed that the wrapper works properly most of the time, and errors rarely occur. This implies $k \approx N$. A 'dull' tester that simply judges every access to be correct, has $m = k$ and thus its statistical correctness is $SC_{dull} = k/N \approx 1$!

As can be seen, even a very high value of SC can be bad if it is not significantly higher than k/N. What follows is that SC alone is not expressive enough. The cause of this phenomenon is that one of the outcomes of the test has a much higher probability than the other.

A similar problem arises in the medical sciences, in the context of routine tests for diseases. Since only a small minority of the population is infected, the probability that the patient suffers from a given disease is very low. The problem of evaluating the quality of a particular test methodology is analogous to the problem of evaluating IA testers. To solve this problem, different quality measures are used that are more expressive than SC [4]:

Definition 3.2 (specifity and sensitivity)

$$\begin{aligned} \mathit{specifity} &= Pr(\mathit{test\ result\ is\ negative} | \mathit{patient\ is\ healthy}) \\ \mathit{sensitivity} &= Pr(\mathit{test\ result\ is\ positive} | \mathit{patient\ is\ ill}) \end{aligned}$$

Adapted to the case of IA testing:

$$\begin{aligned} \mathit{specifity} &= Pr(\mathit{judgement:}\ '\mathit{correct}' | \mathit{access\ was\ correct}) \\ \mathit{sensitivity} &= Pr(\mathit{judgement:}\ '\mathit{erroneous}' | \mathit{access\ was\ erroneous}) \end{aligned}$$

The definition of specifity and sensitivity is symmetric, but the large difference between the probabilities of the possible judgements makes them differently expressive. In order to clarify this, we introduce the reverse conditional probabilities:

Definition 3.3 (negative and positive predictive values)

$$\begin{aligned} NPV &= Pr(\mathit{access\ correct} | \mathit{judgement\ is:}\ '\mathit{correct}') \\ PPV &= Pr(\mathit{access\ erroneous} | \mathit{judgement\ is:}\ '\mathit{erroneous}') \end{aligned}$$

These are actually the most important quality measures because they show how 'trustworthy' the tester is. That is: if the tester judges the access to be correct/erroneous, what is the probability that the access was really correct/erroneous, respectively?

There is an interesting connection between these values, as captured by the theorem below. First some abbreviations: x = specifity, y = sensitivity, $p = Pr$(the access was erroneous), $q = Pr$(judgement is: 'erroneous').

Theorem 2 *(i) If $x \to 1$ and $y \to 1$ and $p \to 0$ and $q \to 0$, then $NPV \to 1$, but PPV does not necessarily converge to 1.*
(ii) If $x \to 1$ and $y \to 1$, then

$$\frac{\partial PPV}{\partial x} \to \frac{1-p}{p} \quad \textit{and} \quad \frac{\partial PPV}{\partial y} \to 0.$$

As already noted, usually the wrapper functions correctly. It can also be assumed that the judgement of the tester is correct in most cases. This is why the theorem deals only with the case when x and y are high (near 1) and p and q are low (near 0). In this ideal case, one would expect that both predictive values converge to 1. The first claim of theorem 2 shows that this is true for NPV, but not for PPV. If PPV is not high enough, this means that the tester gives many false positives. This is exactly the reason why the testing mechanisms presented in the literature suffer from a relatively large number of false positives. This is an intrinsic property of the IATP. So, an otherwise almost perfect tester guarantees high NPV but not necessarily a high PPV. This also implies that only together can NPV and PPV be used as an expressive quality measure.
The second claim of theorem 2 explains the reason of this phenomenon: in the region where both specifity and sensitivity are high, PPV does not depend on the sensitivity anymore, but it depends heavily on the specifity. (Note that $\frac{1-p}{p}$ is a huge number.) Consequently, if the specifity is a little below 1, the positive predictive value suffers from it severely.

Proof. Using Bayes' theorem,

$$\begin{aligned} NPV &= \frac{Pr(\text{access correct, judgement : 'correct'})}{Pr(\text{judgement : 'correct'})} = \\ &= \frac{\text{specifity} \cdot Pr(\text{access correct})}{Pr(\text{judgement : 'correct'})} = \frac{x(1-p)}{1-q} \to 1 \end{aligned}$$

$$PPV = \frac{Pr(\text{access erroneous, judgement : 'erroneous'})}{Pr(\text{judgement : 'erroneous'})} =$$

$$= \frac{\text{sensitivity} \cdot Pr(\text{access erroneous})}{Pr(\text{judgement}: \text{'erroneous'})} = \frac{yp}{q}$$

The latter does not converge, since the limit would depend on whether p or q converges more rapidly to zero. This proves the first claim. (Note that the problem of a low PPV arises if and only if $p \ll q$, which condition also implies that the tester gives many false positives.)

To prove the second claim, the last expression is transformed so that it contains also x but not q:

$$PPV = \frac{yp}{yp + (1-x)(1-p)} = \frac{1}{1 + \frac{(1-x)(1-p)}{yp}}$$

Obviously, p depends neither on x nor on y. So the partial derivatives are:

$$\begin{aligned} \frac{\partial PPV}{\partial x} &= \frac{1-p}{yp}\left(1 + \frac{(1-x)(1-p)}{yp}\right)^{-2} \to \frac{1-p}{p} \\ \frac{\partial PPV}{\partial y} &= \frac{(1-x)(1-p)}{y^2 p}\left(1 + \frac{(1-x)(1-p)}{yp}\right)^{-2} \to 0 \end{aligned}$$

which proves the theorem. □

Beside the requirement that the tester should give correct judgements, there are other, non-functional, requirements that must also be met. To summarize: the 'price' that must be paid for the automatic testing should be kept as low as possible. The most important non-functional quality measures are: efficiency, programming effort (also including maintenance), generality (wrapper-independence), reusability, adaptability, total cost of testing. (A more detailed description can be found in [12].)

4. Testing methods

This section presents various IA testing mechanisms, classified according to the principal concept of their mode of operation. At the end of the section a comparison of the presented methods is given. The practical applicability of the suggested testers will be covered in section 5.

Testing while the wrapper is extracting. The first group of testing schemes test the operation of the wrapper, and not only its output. They can be regarded as white-box test methods. The most important advantage of these methods is that at this point all information is available, whereas if only the output of the wrapper is investigated, the state information of the wrapper (S_W in definition 2.4) is lost. For instance, if the wrapper output is ε, this might be the result of an error but it can also be a valid reply of the IS; if the way the wrapper has worked is investigated, it might be possible to differentiate between these two cases.

The most important disadvantage of such methods is the high wrapper-dependence. Thus, if the wrapper is modified for some reason, the tester also has to be updated. This can increase maintenance costs and decrease reusability significantly.

Testing the output of the wrapper. The methods described in this paragraph make no use of information internal to the wrapper, only its output. This can be thought of as black-box testing. The advantages and disadvantages are exactly the opposite as before: the complete independence of the internal state of the wrapper is advantageous; however, the lack of this information may cause problems. As explained in the Introduction, most existing results use this paradigm for wrapper testing.

The principal idea behind these methods is that the wrapper output takes its values normally from a real subset of Σ^* (denoted by L), *i.e.* certain strings from Σ^* are not plausible. So the task of the tester can be defined as follows: given a wrapper output r, is $r \in L$? The theory of formal languages has invented several methods to represent large, possibly infinite sets finitely and compactly, supposed that the given set is sufficiently structured. For example, a regular grammar, or equivalently, a finite automaton can be used in some cases to encode the set L.

However, the classical results of the theory of formal languages are in many cases insufficient. Most existing work in the literature of wrapper verification focuses on more powerful methods for the testing of the wrapper output (see section 1 and [12] for more details).

Testing using additional ISs. The detection of semantic errors requires additional, more or less domain-specific knowledge. In a mediator system the most natural way to integrate additional knowledge is the integration of new ISs. These ISs can then be used by the tester to determine if the output of the accessed IS makes sense in the given context. Hence, redundancy is achieved, which can generally be used to improve system reliability [14].

Test queries. In testing conventional software systems, regression tests play a vital role [8]. They involve comparison of the output of the system with the known correct output for various inputs – as in black-box testing in general. Unfortunately, this scheme cannot be used directly in the case of the IATP, because it is by no means guaranteed that the IS answers the same query always with the same reply. Hence, this method can only be applied to the very restricted class of static ISs. However, similar methods can be used more generally. For instance, in order to decide whether an empty wrapper output is correct or a result of an error, a test query can be used for which the IS will surely return a non-empty reply.

It can be generally stated that methods in this category can only be used efficiently in conjunction with other testing methods. However, they are very useful in some cases.

Testing the mediator system. All methods described above can also be used at a higher level, namely to test the whole mediator system instead of just one wrapper. This is in many cases more practical. If q ISs are integrated into the mediator system, this reduces the number of necessary testers from q to 1. Thus, testing is cheaper, more efficient and also more robust because the tester does not have to be modified after every change in the wrappers or the ISs. In many cases, it may not even be feasible to test the replies of each IS because the necessary additional knowledge is not available.

The main disadvantage of such methods is that not necessarily all errors are clear from the output of the mediator system. Thus, some errors can remain undetected, leading to lower sensitivity.

Comparison of the suggested methods. In most applications where communication is textual (*e.g.* in the WWW) the best choice is probably a syntactic test of the wrapper output, because it is relatively simple, it does not depend on the implementation details of the wrappers, and – as a result of the redundancy introduced by textual communication – many errors can be detected with such methods. Unfortunately, these methods can be quite inaccurate in some cases.

If syntactic testing schemes are not enough, more intelligent methods must be used. In this case, the integration of additional ISs should be considered, because this would make it possible to detect semantic errors. However, this will also increase testing costs and overhead significantly.

The other methods (test queries and testing during the operation of the wrapper) are rather special, but there are many cases in which they can be used successfully. Test queries should only be used together with other methods because they can only detect a fraction of all possible errors. Testing during the operation of the wrapper may be very effective; however, it makes maintenance quite hard, so it should only be used if absolutely necessary. If possible, an interface should be defined for the wrapper to communicate its internal state.

Several testing mechanisms have been proposed in this section. However, it is yet to be clarified how this functionality can be best embedded into the mediator architecture. Figure 1 reveals several places suitable for the integration of the tester. Also note that the choice of the tester and the way it should be integrated into the mediator architecture are not completely orthogonal. For a review of the most important possibilities for integrating the testing functionality into the mediator architecture, see [12].

5. Case study

We implemented several testing methods and conducted several experiments in order to: (i) demonstrate the applicability of the suggested methods; (ii) evaluate and compare the suggested methods; (iii) compare the results with others in the literature.

Implementation. The implementation took place in three steps: (i) an interactive, simplified version of MetaSearch (called ISMS) was created; (ii) several testing mechanisms were implemented; (iii) a batch program was written for the test of the implemented testing methods on a large set of data.

Implementation details of ISMS as well as the experiences made with it are described in [11]. Here we only report on the implemented testing methods and the batch test.

The following testing methods were implemented:

- A syntactic tester for the URLs, using the following regular expression:
  ```
  ^((http\://)|(ftp\://))?[-\+\w]+(\.[-\+\w]+)+(\:\d+)?\
  (/\~[-\+\.%\w]+)?(/[-\+\.\,\:%\w]+)*/?(\?([-\+\.%\w]+\
  =[-\+\.%\w/]+&)*[-\+\.%\w]+=[-\+\.%\w/]+)?$
  ```

- A syntactic tester for the titles. For this, a very simple but practically well performing method was used: `length(title)<100`.

- A similarly simple, syntactic tester for the excerpt:
  `0<length(excerpt)<300`.

- A more time-consuming semantic tester for the wrapper output. For this, the Internet itself was used as additional IS. Namely, the tester checks if the extracted URLs point to existing pages.

The third step of the implementation consisted of creating a batch program that evaluated the realized testing methods on a sufficiently large set of real-world test examples. Since wrapper errors are relatively rare, it is by no means easy to collect a set of test cases that contains enough erroneous accesses. Fortunately, Dr. Kushmerick of Department of Computer Science, University College Dublin has made the data he had collected in the period May – October 1998 [8, 9] available. This set of data contains the reply pages of 27 actual Internet sites on a couple of specific queries, collected for 6 months.

For the evaluation of the implemented testing methods, the replies of AltaVista, Lycos and MetaCrawler on the query 'happy' have been selected. In the given period of time there were two changes in the case of AltaVista, one in the case of Lycos and two in the case of MetaCrawler. Accordingly, three wrappers have been constructed for AltaVista, two for Lycos, and three for MetaCrawler. Each page was processed with the corresponding correct wrapper as well as with the other wrapper(s) of the particular search engine. This amounted to about 440 test cases.

Ultimately, we have conducted another set of experiments with a smaller set of current data. The goal was twofold: (i) to check if the results based on the three-year-old data set can be transferred to the current situation; (ii) to identify trends that may have an impact on future IA testing possibilities. In particular, we knew that lots of the URLs returned by the search engines three years ago would be invalid by now, thus corrupting our test results. So we checked how many of the URLs currently returned by the three search engines for the query 'happy' are invalid.

Test results. The results of the batch test are described in full detail in [11]. Five figures were stored for each test case: the number of extracted tuples, and the number of likely errors found by the four implemented testing methods.

The most important conclusion to be drawn from the results is that the implemented, intentionally very simple testing methods can be combined to a virtually perfect tester. Namely, there is a big difference in the behaviour of the utilized methods in the case of correct and incorrect wrapper outputs. In the erroneous case, the wrapper output was either empty, or at least one of the applied methods (but often all of them) judged all tuples to be erroneous. In the case of a correct wrapper output, the number of tuples judged to be syntactically incorrect was at most 40%, and of those judged to be semantically incorrect at most 70%. Moreover, it turned out from our experiments with current data that this last figure is normally much lower, in the range of 0-10%.

Thus, there is indeed a wide gap in the behaviour of the testing methods in the correct and incorrect cases.

It is also interesting to investigate how much the individual testing methods contribute to the overall very good performance. Actually, two of the four methods – the syntactic testing methods for the URL and for the excerpt – would have been sufficient to distinguish between correct and incorrect wrapper outputs. But of course other errors are also possible for which the other tests are necessary. Hence, it is important to have at least one testing method for each element of the tuples so that errors concerning only that particular element can be detected.

It has to be noted as well that the principles behind MetaSearch and MetaCrawler are very similar, so that testing the output of MetaCrawler can also be regarded as the prototype of testing a whole mediator system. Hence, it is clear from the tests that the suggested methods are also appropriate for the testing of the whole mediator system.

Another important observation is that the set of changes of the output of search engines in 1998 and in 2001 was very similar. It seems that these modifications are caused by human behaviour, rather than by some particular technology because the latter changes rapidly. Consequently, similar changes are likely to occur in the future as well. This makes it possible to construct testers that can be used robustly for many years.

6. Conclusion

In this paper, we have presented the general information access testing problem (IATP) that arises in mediator systems, when making use of the integrated external, autonomous information sources. A formal framework was presented that allows the investigation of the problem in a setting as general as possible. This generalizes past results from the literature [3, 7, 9, 10]. We have proven that the general IATP is not solvable algorithmically in its most natural form, so we presented quality measures that make it possible to quantify imperfect, but in most cases properly working testers. We have investigated the connection between the most important quality measures, namely specifity, sensitivity and positive and negative predictive values.

In the other, more practically-focused part of the paper, possible solutions were suggested. Moreover, to illustrate the practical applicability of the presented algorithms, some testing schemes have been evaluated on MetaSearch, a simple, but real-world example mediator system.

The most important contributions of the paper are: (i) a general framework for the IATP; (ii) existing results can be transferred to and explained in the new framework; (iii) with careful redundancy techniques better results can be achieved; (iv) the framework allows the proof of intrinsic properties of the IATP with mathematic rigor; (v) the mathematical methods can be combined with an engineering approach to obtain the most important rules for the construction of efficient, economic testers; (vi) the framework shows many future research directions. In particular, the work presented in this paper could be applied to similar environments, where statistical testing is the only solution, raising the problems of significance of the test, quality and coverage of the test, and correctness of the test.

References

[1] J. Calmet, S. Jekutsch, P. Kullmann, and J. Schü. KOMET – A system for the integration of heterogeneous information sources. In *10th International Symposium on Methodologies for Intelligent Systems (ISMIS)*, 1997.

[2] J. Calmet and P. Kullmann. Meta web search with KOMET. In *International Joint Conference on Artificial Intelligence, Stockholm*, 1999.

[3] W. W. Cohen. Recognizing structure in web pages using similarity queries. In *AAAI-99 (Orlando)*, 1999.

[4] Cornell University, College of Veterinary Medicine. Diagnostic accuracy. `http://web.vet.cornell.edu/CVM/Imaging/notes_diagnosis.htm`.

[5] T. Goan, N. Benson, and O. Etzioni. A grammar inference algorithm for the world wide web. In *Proc. of the 1996 AAAI Spring Symposium on Machine Learning in Information Access (MLIA), Stanford, CA.* AAAI Press, 1996.

[6] Intelligent Information Integration, Inofficial home page. `http://www.tzi.org/grp/i3`.

[7] C. A. Knoblock, K. Lerman, S. Minton, and I. Muslea. Accurately and reliably extracting data from the web: a machine learning approach. *Data Engineering Bulletin.*

[8] N. Kushmerick. Regression testing for wrapper maintenance. In *Proc. of AAAI-99 (Orlando)*, pages 74–79, 1999.

[9] N. Kushmerick. Wrapper verification. *World Wide Web Journal*, 3(2):79–94, 2000. (Special issue on Web Data Management).

[10] K. Lerman and S. Minton. Learning the common structure of data. In *AAAI-2000 (Austin)*, 2000.

[11] Z. Á. Mann. Dynamische Validierung von Zugriffen auf externe Informationsquellen in einer Mediatorumgebung. Master's thesis, University of Karlsruhe, Institute for Algorithms and Cognitive Systems, 2001.

[12] Z. Á. Mann. Validating access to external information sources in a mediator environment. Technical report, `http://www.cs.bme.hu/~manusz/work/mediator-2001/techrep.pdf`, 2001.

[13] I. Muslea, S. Minton, and C. Knoblock. Hierarchical wrapper induction for semistructured information sources. *Journal of Autonomous Agents and Multi-Agent Systems*, 2001.

[14] B. Randell, J.-C. Laprie, H. Kopetz, and B. Littlewood, editors. *Predictably dependable computing systems*. Springer-Verlag, Berlin, 1995.

[15] V. S. Subrahmanian, S. Adalı, A. Brink, R. Emery, J. J. Lu, A. Rajput, T. J. Rogers, R. Ross, and C. Ward. HERMES: A heterogeneous reasoning and mediator system. Technical report, University of Maryland, 1995.

[16] G. Wiederhold. Mediators in the architecture of future information systems. *IEEE Computer*, 25(3):38–49, March 1992.