Recent Extensions to BIG: A Resource-Bounded Information Gathering System *

Victor Lesser Bryan Horling Frank Klassner Anita Raja
Thomas Wagner Shelley XQ. Zhang

UMass Computer Science Technical Report 1999-13

March 12, 1999

Abstract

BIG (resource-Bounded Information Gathering) is a next generation information gathering agent which integrates several areas of Artificial Intelligence research under a single umbrella. To date, reported work has presented the rationale, architecture, and implementation of the system. This has included planning, reasoning about resource trade-offs of different possible gathering and extraction approaches, information extraction from both structured as well as unstructured documents, and opportunistic refinement of the search process using the extracted information. In this paper, we present recent improvements made to BIG, which make it a more versatile and robust system. These include documentation classification to handle distraction, sophisticated information fusion techniques, and finally the logistics behind search precision versus coverage tradeoffs. We also present empirical evaluations which show the performance improvement due to these extensions.

* This material is based upon work supported by the Department of Commerce, the Library of Congress, and the National Science Foundation under Grant No. EEC-9209623, and by the National Science Foundation under Grant No. IRI-9523419 and Grant No. IRI-9634938 and the Department of the Navy and Office of the Chief of Naval Research, under Grant No. N00014-95-1-1198. The content of the information does not necessarily reflect the position or the policy of the Government or the National Science Foundation and no official endorsement should be inferred.

1 Introduction

The information explosion that has occurred on the World Wide Web (WWW) has made it an invaluable information resource. However there are different levels of accessibility, reliability, completeness [1], and associated costs to the available information. The complexity of the information gathering problem lies in the fact that manual navigation and browsing of even a subset of the relevant information obtainable through advancing information retrieval (IR) and information extraction (IE) technologies [2, 8, 5, 9] is ineffective without high-level filtering. The time/quality/cost tradeoffs offered by the collection of information sources and the dynamic nature of the environment lead us to conclude that the user cannot (and should not) serve as the detailed controller of the information gathering (IG) process.

In [10, 11], we present BIG (resource-Bounded Information Gathering), a single information gathering agent that takes the role of the human information gatherer and incorporates different Artificial Intelligence (AI) technologies, namely scheduling, planning, text processing, information extraction, and interpretation problem solving to achieve this task.

In response to a query, BIG locates, retrieves, and processes information to support a decision process. Implementationally, we have concentrated on the software domain; BIG's specific area of expertise is in helping clients select software packages to purchase. For example, a client may instruct BIG to recommend a database package for Windows 98, and specify constraints on the amount of money to pay for such a product and the amount of time and money to spend locating information about database products. The client may also specify a preference for information precision versus coverage, a higher coverage preference will result in more products being discovered, but with less information about each product. A preference for greater precision will result in BIG spending more resources to construct very accurate models of products. BIG will then plan, locate, and process relevant information, returning a recommendation to the client along with the supporting data.

In this paper, we emphasize the recent improvements made to BIG, which make it a more versatile and robust system. These include documentation classification to handle distraction, sophisticated information fusion techniques, and finally the logistics behind search precision versus coverage tradeoffs.

2 BIG Overview

The overall BIG agent architecture is shown in Figure 1. The agent is comprised of several sophisticated components that are complex problem-solvers and research subjects in their own rights. All of these components are implemented and integrated in BIG. The construction, adaptation, and integration of these components was a non-trivial process. The fruition of these efforts in BIG not only produced an interesting research tool, but, the integration has also influenced and refined the research directions pertaining to the individual components as well. The following is a brief overview of the major architectural components. Functional details of each of the components are presented in [10, 11]

The complexity of our objective requires support for reasoning about tradeoffs among resource constraints (e.g. the search cost must be less than $5), the quality of the selected item and the quality of the decision process (the width and depth of the search, effectiveness of IE methods usable with specified time and cost limits). Such support mandates a high level of sophistication in the design of our information gathering agent's components. A domain problem solver, the RESUN [3, 4] planner, translates a client's information need into a set of goals and generates plans to achieve those goals. The strength of the RESUN planner is that it identifies, tracks, and plans to resolve sources-of-uncertainty (SOUs) associated with blackboard objects, which in this case correspond to gathered information and hypotheses about the information. To support reasoning about time/quality/cost trade-offs, and thus a range of different resource/solution paths, the planner enumerates multiple different ways to go about achieving the goals and describes them statistically in three dimensions, duration, quality, and cost, via discrete probability distributions. Another sophisticated problem solv-

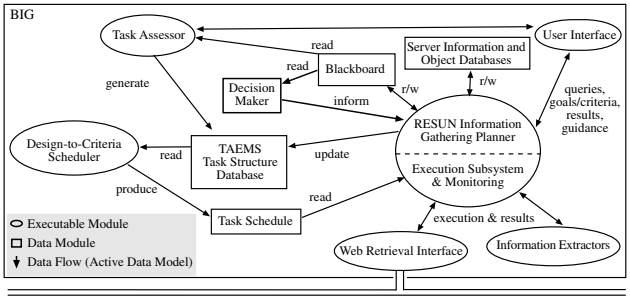

Figure 1: The BIG Agent Architecture

ing component, is the Design-to-Criteria [12] scheduler, which examines the possible solutions paths and determines a set of actions to carry out and schedules the actions – coping with an exponential scheduling problem in real-time through the use of approximation and goal directed focusing. The resulting schedule is a single-agent schedule that contains parallelism and overlapping executions when the primitive actions entail non-local processing, e.g., issuing requests over the network. The non-local activities can be embedded within primitive actions or explicitly modeled as primitive actions with two components, one for initiation and one for polling to gather results, separated by propagation delays. This enables the agent to exploit parallelism where possible and where the performance of the parallel activities will not adversely affect the duration estimates associated with its activities.

As BIG retrieves documents, yet another sophisticated problem solver, an IE system [7] in conjunction with a set of semantic, syntactic, and site specific tools, analyzes the unstructured text documents in order to construct information objects that can be used by the planner for decision making and refinement of other information gathering goals. It is important to note that the advent of widespread structure markup languages like XML, or tools to wrap web sites so that data can be retrieved in a structured form, will only improve BIG's ability to gather and process information. In essence, the IE system used in BIG fills the role of these other tools and approaches. In the event that structured information is available, the extraction step can be bypassed and the information incorporated directly into BIG's reasoning structures.

Other complex components in BIG include a framework for modeling domain tasks, a Web site information database, an idle-time Web site probe for refining the database, and a task assessor to assist in translating the problem solver's domain plans into a domain independent representation appropriate for use by the Design-to-Criteria scheduler and other high-level components.

The main distinguishing characteristics of this research are:

Active Search and Discovery of Information BIG does not rely entirely upon a pre-specified set of sites from which to gather information. BIG also utilizes general URL search engines and sites / information sources discovered during previous problem solving sessions.

Resource-boundedness BIG problem solves to meet real-time deadlines, cost constraints, and quality preferences. BIG reasons about which

actions to take to produce the desired result and plans accordingly. This is accomplished through the use of the Design-to-Criteria scheduler and by employing an end-to-end, rather than reactive, control process.

Opportunistic and Top-down Control BIG blends opportunistic, reactive, problem solving behaviors with the end-to-end view required to meet real-time deadlines and other performance constraints. This enables BIG to respond dynamically to uncertainties in the product models as well as newly learned information.

Information Extraction and Fusion The ability to reason with gathered information, rather than simply displaying it for the user, is critical in the next generation of information gathering systems. BIG uses research-level extraction technology to convert free format text into structured data; the data is then incorporated and integrated into product models that are examined by BIG's decision process, resulting in a product recommendation.

Incorporation of Extracted Information In addition to building product models, extracted information is incorporated in BIG's search as it unfolds. For example, competitor products discovered during the search are included in BIG's information structures, possibly resulting in new goals to pursue additional information on these products.

In the remainder of this paper, we present interesting research issues addressed by BIG in Section 3 and provide details from actual BIG runs. In Section 4 BIG is discussed from a holistic perspective, including empirical experiment results and an execution trace. Conclusions and future directions are presented in Section 5.

3 New Extensions in BIG

In this section we present and discuss recent extensions to BIG from an isolated perspective. A more holistic view of BIG is given in Section 4.1 where a detailed walk through and aggregate empirical results are presented.

3.1 The Importance of Document Classification

Until recently, BIG has been plagued by an interesting extraction problem when dealing with products that are complimentary to the class of prod-

	# Rejected	Top Candidates	Selected Product
No Filter or Classifier	0 / 13	Portuguese Dictionary Module Norwegian (Nynorsk) Dictionary Module Norwegian (Bokmal) Dictionary Module The Nisus Dictionary Collection US Definition Dictionary	Portuguese Dictionary Module
Simple Filter	31 / 44	EndLink 2.0 Spelling Coach Pro 4.1 Retrieve It! 2.5 Nisus Writer 5.1 CD ROM with Manual Microsoft Word 6.01	EndLink 2.0
Filter & Classifier	53 / 74	ClarisWorks Office 5.0 Corel WordPerfect 3.5 ACADEMIC Nisus Writer 5.1 CD ROM with Manual Corel WordPerfect 3.5	ClarisWorks Office 5.0

Figure 2: Advantages of Document Classification

ucts in which a client is interested. For example, when searching for word processors BIG is likely to come across supplementary dictionaries, word processor tutorials, and even document exchange programs like Adobe Acrobat. These products are misleading because their product descriptions and reviews often contain terminology that is very similar to the terminology used to describe members of the target class. When BIG processes one of these misleading documents, it gets *distracted* and future processing is wasted in an attempt to find more information about a product that is not even a member of the target class. Experiments indicate that this type of distraction can be reduced through the use of a document classifier before text extraction is performed on candidate documents. Documents that do not seem to be members of the target class are rejected and text extraction is not performed on them – thus no new distracting information objects are added to BIG's blackboard.

Figure 2 provides a sample of our initial results. BIG was run in three different modes: 1) BIG alone, 2) BIG with the use of a simple grep-like pattern-matching filter to classify documents, 3) BIG with the use of Naive Bayes document classifier [6] and the simple grep filter. The grep-like filter examines the document for instances of terms that describe the software genre in question, e.g., "word processor." These terms are hand produced for each query genre – in essence, hardwired into the system. In contrast, the document classifier is trained using positive and negative examples – it learns term-based similarity and difference measures.

In the first run, shown in the figure, no filter and no classifier are used. All documents retrieved are processed by the information extractors. None of the top five objects in this test case are members of the target product class, i.e., they are all related to word processors but none of them is actually a word processing product. Clearly, BIG does very poorly when relying on outside sources like vendor's search engines to classify products. In the

second run, the simple grep-like filter is used to check documents before processing; 31 documents are rejected by the filter and the overall results are a little better. There are word processing products among the candidates, but the selected product is not a word processor. In the last run, both classifier and filter are used to check documents; 53 documents are rejected. Almost all top-ranked candidates are word processing products and the top product, "ClarisWorks Office 5.0" is an integrated office suit that includes a word processing package.

Clearly, document pre-classification at the agent side is necessary. Vendor search engines are typically keyword based and generally return numerous products that are not members of the target class but are instead related or supplementary products. Improving the classification of documents and widening the training corpus for the classifier are areas of future development.

3.2 Scheduling for Hard-Deadlines

Design-to-Criteria (DTC) scheduling is the soft real-time process of evaluating the quality, cost, duration, and certainty trade-offs of alternative ways to achieve a task, and producing a custom schedule for task achievement that meets the requirements, e.g., real-time deadlines, cost constraints, quality preferences, etc., of the client. DTC is an approach to coping with exponential combinatorics through satisficing, approximation, and goal-directed schedule generation, all of which is documented in [12].

During the course of the BIG project, we encountered an interesting problem with the satisficing focusing methodology used in Design-to-Criteria when it is combined with hard deadlines and certain classes of very large task structures. Without delving into exhaustive detail, the problem is that in order to cope with the high-order combinatorics in these particular situations, the scheduling algorithm must prune schedule approximations, called *alternatives*, and develop only a subset of these. Herein lies the problem.

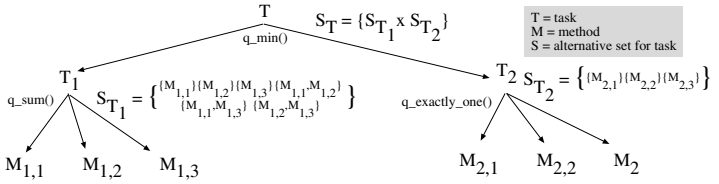

Figure 3: Alternative Sets Lead to Cumbersome Combinatorics

Alternatives are constructed bottom-up from the leaves of the task hierarchy to the top-level task node, i.e., the alternatives of a task are combinations of the alternatives for its sub-tasks. Figure 3 shows the alternative set generation process for a small task structure. Alternatives are generated for the interior tasks T_1 and T_2, and these alternatives are combined to produce the alternative set for the root task, T. The complexity of the alternative generation process is pronounced. A task structure with n methods leads to $O(2^n)$ possible alternatives at the root level. We control this combinatorial complexity by focusing alternative generation and propagation on alternatives that are most likely to result in schedules that "best" satisfice to meet the client's goal criteria; alternatives that are less good at addressing the criteria are pruned from intermediate level alternative sets. For example, a criteria set denoting that certainty about quality is an important issue will result in the pruning of alternatives that have a relatively low degree of quality certainty. After the alternative set for the high-level task is constructed, a subset of the alternatives are selected for scheduling.

For situations in which there is no overall hard deadline, or in which shorter durations are also preferred, the focusing mechanism works as advertised. However, in the BIG project, we are also interested in meeting real-time deadlines and other hard resource constraints (in contrast to those that are relaxable), and often these preferences are not accompanied by a general preference for low duration or low cost. In these cases, the problem lies in making a *local* decision about which alternatives to propagate (at an interior node) when the decision has implications to the local decisions made at other nodes – the local decision processes are interdependent and they interact over a shared resource, e.g., time or money. Casting the discussion in terms of Figure 3: assume T has an overall deadline of 5 minutes and T_1's alternatives require anywhere from 2 minutes to 20 minutes to complete, and T_2's alternatives are similarly characterized. Assume that quality is highly correlated with duration, thus the more time spent problem solving, the better the result. If the criteria specifies maximum quality within the deadline, the alternatives propagated from T_1 to T will be those that achieve maximum quality (and also have high duration). Likewise with the alternatives propagated from T_2. The resulting set of alternatives, S_T at node T will contain members characterized by high quality, but also high duration, and the scheduler will be unable to construct a schedule that meets the hard deadline. The optimal solution to this problem is computationally infeasible ($\omega(2^n)$ and $o(n^n)$) as it amounts to the general scheduling problem because of task interactions and other constraints.

Two approximate solutions are possible. One approach is to preprocess the task structure, producing small alternative sets at each node that characterize the larger alternative population for that node. Then examining the ranges of alternatives at each node and heuristically deciding on an allocation or apportionment of the overall deadline or cost limitation to each of the interior nodes. This local-view of the overall constraint could then be used to focus alternative production on those that will lead to a root-level set that meets the overall constraint. The other approach, which we have employed, is to detect when the local-view of the decision process is problematic and in those cases sample from the population of alternatives, producing a subset that exhibits similar statistical properties, and propagating these alternatives. This leads to a less-focused set of root level alternatives than the prior approach, but it saves on the added polynomial level expense of the first approach; this solution has served us well in the BIG project and enabled the scheduler to maintain its soft real-time level of performance and still produce good schedules.

3.3 Information Fusion

We use the term *information fusion* to denote the process of integrating information from different sources into a single product object; the information may be complimentary, but also contradictory or incomplete. There are several aspects to the fusion issue. The most straightforward type of fusion is information addition – where a document provides the value to a slot that is not yet filled. A more interesting type of fusion is dealing with contradictory single value information, e.g., two documents reporting different prices for a product, or two documents identifying a different maker for the product. When BIG encounters this fusion issue, the item with the highest associated degree of belief is used. The degree of belief is a weighted measure computed from the information source quality as well as the quality of the information extraction technology. Another issue is how to integrate different opinions about the product. The latter is done in BIG by associating two metrics with every review document, one representing information or site quality, and one representing the quality of the product as expressed in the review. This dual then conceptually represents a value / density pair – the information quality metric determines the weight given to the product quality metric when comparing different metrics for different reviews. To illustrate BIG's fusion process, consider the following partial trace.

In this example, BIG is searching for word processor products for the

Macintosh. In response to a general query to the Server Database about word processing products, the *MacMall* retail site returns a list of URLs. URL_A, from Figure 4, is selected by BIG for retrieval and processed. BIG extracts "Dramatica Pro 2.0" from the document as the title of the software package; it also extracts that "Screenplay" (Inc.) as the maker and that the package sells for a price of $289.99. [1]

```
URL_A  http://www.cc-inc.com/sales/detail.asp?dpno=79857&catalog_id=2
URL_B  http://www.freedombuilders.com//dramatica.htm
URL_C  http://st2.yahoo.com/screenplay/dpro30mac.html
URL_D  http://www.heartcorps.com/dramatica/questions_and_answers/..
URL_E  http://www.zdnet.com/macuser/mu_0796/reviews/review12.html
URL_F  http://www.macaddict.com/issues/0797/rev.dramaticapro.html
```

Figure 4: URLs for Documents Retrieved During Processing

Since the document provides very little additional information about Dramatica, BIG associates an *uncertain-support* SOU with the object. Because the product object is a promising area of exploration, relative to everything else on the blackboard, BIG decides to attempt to resolve the SOU. Toward that end, it queries Infoseek about Dramatica, resulting in a long list of URLs that are combined with their descriptive text to create candidate document description objects which are added to the blackboard. BIG selects and retrieves a subset of these, starting with URL_B, which is a detailed description of the product. Processing the description results in the addition of platform specifications to the product object, namely that it runs on Windows95 and Apple Macintosh systems. The description also contains sufficient verbiage that it is analyzed using a keyword-based review processing heuristic that looks for positive and negative phrases and rates products accordingly weighing the product features by the user preference for such features. Though the verbiage praises the product, it is given a rating of -.57 because the review does not praise the product for the features in which the client is interested. In other words, even though the review is positive, it does not make specific reference to the product features in which the client is interested and thus it is given a negative value to denote that the product is below average quality-wise. However, since the document in question is not widely referenced by other documents, it is given a low information quality (squality) rating and the negative review (pquality) rating

[1] Dramatica is actually a product contained in our corpus of word processor class documents used to train the document classifier. Thus, the pursuit of Dramatica as a word processing package is valid from BIG's perspective, though the classification of Dramatica as a word processor is perhaps debatable.

will thus have little weight when compared to other sources.

In response to the continued existence of the *uncertain-support* SOU, BIG decides to gather more information. It selects and retrieves URL_C, URL_D, URL_E, and URL_F, in that sequence. Space precludes presenting an exhaustive sequence of product object transformations as information is integrated into the object. In general, it is observed that elevation of the product's overall quality rating and the increase in the various rating criteria like ease-of-use and stability. For free format reviews such as URL_D (in contrast to sites that employ consistent numerical rating systems), these metrics are determined by a set of heuristics that examine the text for certain positive or negative expressions.

The remaining documents are retrieved, processed, and integrated in a similar fashion. The final product object is subsequently compared to other product objects during the decision process. While this example results in the construction of a fairly complete product object, the objects used in the final decision process are not all at the same level of completeness. Some objects may contain less information (but not much) and some may contain more product details or more review summarization statistics. The decision process takes into account the quantity and quality of the information pertaining to the objects.

3.4 Precision versus Coverage

Precision versus *coverage* is an issue often discussed in literature relating to information gathering or information retrieval. In the BIG context, once a satisfactory amount of information has been processed to support a high quality decision process, the issue becomes how best to spend the remaining time, cost, or other most constrained resource. One alternative is to spend the time gathering more information about other products, i.e., discovering new products and building models of them. Another alternative is to spend the time discovering new information about the existing products in order to increase the precision of the product models. Both alternatives can lead to higher quality decision processes since both expand the range of information on which the decision is based.

BIG supports both of these behaviors, and a range of behaviors in between the binary extremes of 100% emphasis on precision and 100% emphasis on coverage. BIG clients specify a precision/coverage preference via a percentage value that defines the amount of "unused" (if there is any) time that should be spent improving product precision. The remainder is

spent trying to discover and construct new products. For example, if a client specifies .3, this expresses the idea that 30% of any additional time should be spent improving precision and 70% should be spent discovering new products.

BIG achieves this trade-off behavior in two ways: by planning and scheduling for it *a priori*, and by responding opportunistically to the problem solving context within the constraints of the schedule. Scheduling for the precision / coverage trade-off is accomplished by relating the precision and coverage specification to quality for the Design-to-Criteria scheduler and giving the scheduler a set of options, from which to choose a course of action.

Table 1 shows BIG's ability to trade-off precision and coverage. The table contains data for three sets of runs, for the same query and with the same criteria settings (only the precision setting is varied). In each run, three trials are performed, each with a different precision preference setting, namely 10%, 50%, and 90% respectively. Since network performance varies during execution, and there is some element of stochastic behavior in BIG's selection of equally ranked documents, no two trials are identical even if they have the same preference settings. Note the general trends in the different runs. As more weight is given to increasing precision, the number of products (T.P.) decreases, as does the number of products used in the decision process (#P). The difference between these two values is that some product objects lack sufficient information to be included in the decision process and some of the product objects turn out to relate to products that do not meet the client's specification (e.g., wrong hardware platform, wrong product genre, price too high, etc.), an extreme example of this is in run number two in the third trial where only one product is produced. As the number of products decrease as more weight is given to precision, the average information coverage per object (A.C.) *increases*, as does the information extraction / processing accuracy (P.A.). The decision confidence also generally increases, particularly in runs two and three, though this item takes into account the total coverage represented by the products as well as the precision of the product models so its increase is not proportional to the other increases.

From an end user perspective, the precision/coverage specification enables clients to express preferences for one solution class over another. For a client who needs a speedy result, and has an accordingly short deadline, the preference specification may result in a slight difference at best. However, for a client with more generous time resources, the difference can be

pronounced.

#	Pref	Sched.	Exec.	T.P.	#P.	A.C.	P.A.	D.C.
1	0.1	629	587	33	7	1.86	1.38	0.85
	0.5	622	720	14	6	3.83	1.47	0.89
	0.9	651	685	8	3	7.0	2.12	0.89
2	0.1	629	656	33	8	1.75	1.32	0.85
	0.5	622	686	14	4	3.0	1.5	1
	0.9	652	522	7	1	7.0	2.12	1
3	0.1	629	702	29	7	1.71	1.47	0.85
	0.5	622	606	15	6	2.33	1.52	1
	0.9	651	572	7	2	4.5	1.7	0.99

Key: # is the run number, Pref = preference for precision, Sched. = total execution time as predicted by model and anticipated by scheduler, Exec. = actual execution time, T.P. = total product objects constructed, #P = total products passed to decision process, A.C. = average coverage per object, P.A. = extraction processing accuracy per object, D.C. = overall decision process confidence.

Table 1: Trading-Off Precision and Coverage

4 BIG in Action

In this section we present and discuss BIG from a holistic perspective.

4.1 Empirical Results

Table 2 illustrates how the system operations under different time constraints. The experiments are for searches to find word processing products and the search and product criteria is the same for all runs. Only the time alloted for the search varies.

The first four columns of data provide information about the duration of each search. *User Time* denotes the users target search time; the value in parenthesis represents the upper bound on how far over the target search time the scheduler was permitted to go in order to achieve a good quality/cost/duration tradeoff. *Sched.* denotes the expected total duration of

the schedule produced by the Design-to-Criteria scheduler and *Exec.* denotes the actual duration of the discovery and decision process. The difference in these values stems from the high variance of web-related activities and reflects issues like changes in network bandwidth during the search, slow downs at remote sites, and so forth. The statistical characterizations of these activities are also often imperfect, though they are improved over time. Given the variances involved, we are satisfied with the relationship between expectations and reality.

The next four columns denote number of considered products (#p), total number of products found (T.P.), aggregate information coverage (I.C.), and average information coverage per product object (A.C.). These values reflect the number and qualities of the information sources used to generate the final decision. Given additional time, BIG will adjust its searching behavior in an attempt to find both more sources of information, and more supporting information for previously discovered products. The results of this behavior can be seen in the correlation between longer running time and larger information coverage values; these values represent the total number of documents found and the average number of supporting documents a product has, respectively. As one would expect, the larger number of information sources also serves to increase both the number of known products and the size of the subset selected for consideration, which in turn affects the confidence BIG has in its final decision.

The last column describes how confident the system is in the decision making processes. Decision confidence, generated by the decision maker, reflects the likelihood that the selected product is the optimal choice given the set of products considered. This value is based on the quality distributions of each product, and represents the chance that the expected quality is correct. It should be noted that decision confidence is not dependent on execution time or processes effort.

Our query for the test runs is that of a client looking for a word processing package for the Macintosh costing no more than $200, and would like the search process to take 300/600/900 seconds and the search cost to be less than five dollars. The client specifies the relative importance of price to quality to be 60/40 and the relative importance of coverage to confidence to be 50/50.

The decision confidence value is affected by both the number of products considered and their respective attributes and qualities. BIG first selects a product, based on its attributes and the user's preferences. It then calculates the decision confidence by determining the probability that the selected

U.T.	#	Sched.	Exec.	#p	#T.P.	I.C.	A.C.	D.C.
300	1	311	367	4	10	12	1.5	1
(330)	2	308	359	3	10	16	1.3	1
	3	305	279	3	10	11	1.3	1
	4	311	275	3	11	13	1.67	1
	5	321	286	4	10	12	1.5	1
	6	321	272	3	10	12	1.3	0.84
	7	262	327	3	11	12	1.67	1
	8	262	337	3	10	11	1.3	1
	9	262	301	2	11	10	1.0	1
	10	259	292	2	11	11	1.5	1
ave.		302	310	3	10.4	12	1.4	0.98
600	1	658	760	6	17	45	4.0	0.99
(660)	2	658	608	4	17	44	6.75	1.0
	3	645	732	5	20	46	5.4	1.0
	4	649	809	10	28	49	3.1	0.96
	5	649	730	7	17	42	4.3	0.84
	6	653	774	4	23	55	6.5	0.99
	7	653	671	4	18	35	5.3	0.99
	8	653	759	6	18	41	4.8	0.84
	9	653	760	5	28	50	5.4	0.94
	10	653	852	5	18	42	4.6	0.85
ave.		652	746	5.6	20	45	5.0	0.95
900	1	951	975	5	37	61	5.8	0.99
(990)	2	968	956	8	30	55	4.1	1
	3	914	919	8	23	64	4.0	1
	4	960	796	6	34	64	5.3	0.96
	5	960	1026	9	24	32	4.1	0.99
	6	987	968	8	27	60	4.4	0.94
	7	987	1102	8	27	63	5.5	0.94
	8	987	896	5	32	69	5.4	0.84
	9	987	918	7	32	66	5.1	0.84
	10	978	1289	14	39	79	3.9	1
ave.		968	985	7.8	31	61	4.8	0.95

Key: U.T. = users preferred search time (linearly decreasing utility post-deadline in this case), Sched. = total execution time as predicted by model and anticipated by scheduler, Exec. = actual execution time, I.C. = information coverage, T.P. = total product objects constructed, #P = total products passed to decision process, A.C. = average coverage per object, D.C. = overall decision process confidence.

Table 2: Different Time Allotments Produce Different Results

product is the optimal choice, given the available subset of products. In the 300 second runs, the total number of considered products is fairly low, which increases the chance that the pool of products is heterogeneous. In such a population, it is more likely that a single candidate will stand out from the others, which goes to explain the large percentage of perfect scores in the shortest run. When BIG is given more time to find more products, the chance that individual candidates will sharply contrast is reduced. Greater average coverage affects this contrast by increasing the likelihood that product candidates will be fully specified. This will typically make the candidate set have a higher quality rating which makes the population more homogeneous. It is this blurring across attribute dimensions which reduces BIG's confidence in the final decision.

Two interesting cases in this last column are worth explaining in more detail. In the sixth 300 second run, one can see that the decision quality was calculated to be 0.84, much lower than other runs in the same set. This was due to the fact that two of the three products considered were actually the same product, but one was an academic version. These two products had relatively similar quality ratings, which were significantly higher than the remaining product, which caused BIG to have a lower confidence in its decision. The second anomaly occurs in the tenth run in the 900 second scenario. In this case, 14 products were considered for selection. Of the group, 11 had a price higher than $400, two were above $200 and the remaining product was roughly $70 with good coverage of the user's desired characteristics. This large price discrepancy led the selected product to have a much higher quality rating than the competition, which led to the high decision confidence.

5 Strengths, Limitations, and Future Directions

The combination of the different AI components in BIG has equipped it with some powerful capabilities. In contrast to most other work done in this area, BIG performs *information fusion* not just document retrieval. That is, BIG retrieves documents, extracts attributes from the documents, converting unstructured text to structured data, and integrates the extracted information from different sources to build a more complete model of the product in question. The use of the RESUN interpretation-style planner enables BIG to reason about the sources-of-uncertainty associated with particular aspects of product objects and to plan to resolve these uncertainties

by gathering and extracting more information that serves as either corroborating or negating evidence.

Several features of BIG are not discussed in this paper. One such feature is BIG's ability to meet deadlines and reason about quality, cost, duration, and certainty trade-offs of different possible courses of action. This ability comes from the integration of the Design-to-Criteria agent scheduler [12]. Another such feature is that BIG can learn information about information resources and products over time – it benefits from prior problem solving instances [10].

In terms of limitations and extensibility, many of the components used in the system, such as the web retrieval interface and some of the information extractors, are generic and domain independent. However, certain aspects of the system require domain specific knowledge and adapting BIG to operate in another domain, perhaps the auto-purchase domain, would require the addition of specific knowledge about the particular domain. Another potential limitation is the ambitious use of text extraction technology to drive further processing; XML and other web content structuring languages may remove or decrease the importance of this issue.

Future work includes improving the integration of the top-down view of the Design-to-Criteria scheduler and the opportunistic bottom-up view of the RESUN planner. Another future direction involves moving BIG into a multi-agent system.

References

[1] C. Mic Bowman, Peter B. Danzig, Udi Manber, and Michael F. Schwartz. Scalable Internet Resource Discovery: Research Problems and Approaches. *Communications of the ACM*, 37(8):98–114, 1994.

[2] J. P. Callan, W. Bruce Croft, and S. M. Harding. The INQUERY retrieval system. In *Proceedings of the 3rd International Conference on Database and Expert Systems Applications*, pages 78–83, 1992.

[3] Norman Carver and Victor Lesser. A new framework for sensor interpretation: Planning to resolve sources of uncertainty. In *Proceedings of the Ninth National Conference on Artificial Intelligence*, pages 724–731, August 1991.

[4] Norman Carver and Victor Lesser. The DRESUN testbed for research in FA/C distributed situation assessment: Extensions to the model of

external evidence. In *Proceedings of the First International Conference on Multiagent Systems*, June, 1995.

[5] J. Cowie and W.G. Lehnert. Information extraction. *Communications of the ACM*, 39(1):80–91, 1996.

[6] Naive bayes document classifier. Supplement to Machine Learning by Tom Mitchell, 1997, McGraw Hill. http://www.cs.cmu.edu/afs/cs/project/theo-11/www/naive-bayes.html.

[7] D. Fisher, S. Soderland, J. McCarthy, F. Feng, and W. Lehnert. Description of the UMass Systems as Used for MUC-6. In *Proceedings of the 6th Message Understanding Conference*, Columbia, MD, 1996.

[8] Leah Larkey and W. Bruce Croft. Combining classifiers in text categorization. In *Proceedings of the 19th International Conference on Research and Development in Information Retrieval (SIGIR '96)*, pages 289–297, Zurich, Switzerland, 1996.

[9] W.G. Lehnert and B. Sundheim. A performance evaluation of text analysis technologies. *AI Magazine*, 12(3):81–94, 1991.

[10] Victor Lesser, Bryan Horling, Frank Klassner, Anita Raja, Thomas Wagner, and Shelley XQ. Zhang. BIG: A resource-bounded information gathering agent. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence (AAAI-98)*, July 1998. See also UMass CS Technical Reports 98-03 and 97-34.

[11] Victor Lesser, Bryan Horling, Frank Klassner, Anita Raja, Thomas Wagner, and Shelley XQ. Zhang. A next generation information gathering agent. In *Proceedings of the Fourth International Conference on Information Systems, Analysis, and Synthesis*, pages 41–50, July 1998. In conjunction with the World Multiconference on Systemics, Cybernetics, and Informatics (SCI'98), Volume 2. Also available as UMASS CS TR 1998-72.

[12] Thomas Wagner, Alan Garvey, and Victor Lesser. Criteria-Directed Heuristic Task Scheduling. *International Journal of Approximate Reasoning, Special Issue on Scheduling*, 19(1-2):91–118, 1998. A version also available as UMASS CS TR-97-59.