Technical Report

Department of Computer Science
and Engineering
University of Minnesota
4-192 EECS Building
200 Union Street SE
Minneapolis, MN 55455-0159 USA

TR 00-005

Combining Hierarchical Filtering, Fuzzy Logic, and Simulation with Software Agents for IP (Intellectual Property) Selection in Electronic Design

Jian Liu, Eugene Shragowitz, and Wei-tek Tsai

February 04, 2000

Combining Hierarchical Filtering, Fuzzy Logic, and Simulation with Software Agents for IP (Intellectual Property) Selection in Electronic Design

Jian Liu, Eugene Shragowitz, Wei-Tek Tsai
Department of CSE, University of Minnesota
Email: {jliu,shragowi,tsai}@cs.umn.edu

Abstract

One of the central issues in electronic design is reuse of existing design solutions (IPs). This paper describes an Internet-based distributed system of software agents, performing dialogue with the users, search of IPs over Internet, filtering and evaluation of IPs based on fuzzy logic and vendor site simulation environment. Application of this system allows the potential user to speed up the search process and improve the quality of selected solutions.

Keywords: Intellectual Property, hardware design, hardware/software codesign, software agent, filtering, information retrieval, Internet, fuzzy logic, framework.

1 Introduction

1.1 The emerging problem of IP selection

The growing size of the System-on-Chip (SoC) and pressure to reduce time to market led to emergence of the market of IPs (Intellectual Properites), where potential customers can select existing design solutions that satisfy their needs.

The trend is for more and more designers to use IPs from the third parties [2] [6] [7]. An important characteristic of SoC contemporary design is parallel and concurrent development of hardware and software [1]. In the phase of hardware/software partitioning, one task is to select appropriate IPs to realize hardware modules.

Information on IPs on the web is available from variety of heterogeneous sources ranging from small vendors marketing their own designs to companies running databases on a large number of IPs available on the market. Vendors of their own IPs typically present general information in the natural language format and descriptions of designs in the form of hardware description language models. The large centralized databases of IPs are similar to catalogs. They may or may not contain structured on unstructured general information on IPs in their catalogs. So far there is no accepted standard on types of general information provided by the centralized databases. Most of the centralized databases provide links to the sites of IP vendors.

For many specific functionalities, there are multiple corresponding IPs. Each of IPs is defined by multiple parameters introduced at different hierarchical levels, e.g. behavioral, RTL, logic and physical models. Parameters may also be associated with the specific development environment defined by CAD tools. Therefore, it is very likely that no existing IP can exactly meet all the requirements of a user in terms of interfaces and performance characteristics. It is also very likely that somewhere in the process of investigation user may encounter such characteristics of IP that make it unacceptable for this given application.

Absence of standards for IP documentation introduces additional complications. Information

that presents interest for the user may be intentionally or unintentionally omitted by a vendor. In this environment it is difficult for users to make an intelligent choice.

The problem is exacerbated by the fact that the documentation on IPs in public domain is very limited. The tools which perform selection based on some general classifications do not measure the divergence between desirable and available characteristics of a design.

On one hand, a designer who intends to reuse IP should collect a lot of information about existing designs to pick up the best IP candidate. On the other hand, due to copyright aspects of the situation, IP users may not get an access to the source code without prior written agreements/licenses. The situation described here is difficult for both potential users and vendors.

In this environment a process of IP selection for reuse consists of the following major steps.

1. The user specifies the system and determines the parts targeted for implementation by IPs. The specification may be a mixture of behavioral and structural models in hardware description languages.

2. The user searches various sites ranging from catalogs to individual vendors to find IPs matching his specifications. In the process the user may need to study descriptions of multiple IPs from variety of sources. He may or may not have an access to IP models.

3. After identifying potential candidates IPs, the user starts negotiations with IP vendors, which cover a wide range of technical, legal and business issues.

This process of search is not straightforward.

1.2 Related works

In recent years, the Internet gradually has become a part of the EDA environment, some prototypes of Web-based tools for EDA applications have been developed [20] [21] [22].

Many industry organizations are involved in standardizing of IP-based designs with the goal

to promote IP reuse. VSIA (Virtual Socket Interface Association) [16] is particularly chartered to promote standard specifications of data formats, test methodology and interfaces. Rapid (Reusable Application-Specific Intellectual Property Developer) [17] focuses on some important commercial issues of IP reuse, such as increasing IP quality, standardizing IP delivery. VCX (Virtual Component Exchange) [18] is now tackling the business and legal issue associated with SoC challenges.

IP vendors are also making efforts to promote the IP reuse. Synopsys [15] introduced MORE assessment program, which rates soft IPs with respect to adherence to rules and guidelines of the design, verification and delivery described in the Reuse Methodology Manual (RMM) [1] for soft IPs. Design And Reuse Initiative [19] offers three types of yellow pages services: Virtual Components, Embedded Software, and Service. It maintains a catalog for about 2000 IPs. Users can search this database by either keywords or using hierarchical IP taxonomy which serves as a selection tree for IP products.

Recently some case-based reasoning tools, such as READEE [5], were developed to facilitate the IP selection. These tools evaluate the overall similarity, assigning weights to individual characteristics. A technology was developed to accelerate the IP evaluation process in programmable logic [23]. This technology utilizes encryption to generate protected models for evaluation purpose.

Vendors generally like to post information about their products on the Internet. Most information published on the web is presented in a loosely structured natural language text form. Various general-purpose software agents are developed to assist users to retrieve useful information in such environment. One example is the Shopping Agent [4]. This tool assumes that every product is described in easily recognized formats. This assumption makes it easy to parse the HTML files to retrieve useful information by searching for keywords. Other agents are also developed to explore and categorize documentation from the web [8] automatically.

2 Research Approach: SAFIPS (Software Agents for IP Selection)

Previous works mentioned above do not take into consideration a distributed nature of the information for IPs. For example, the large centralized database contain only limited generic information on IPs, not sufficient for technically justifiable IP selection. In fact, a choice made based only on parameters given in centralized database, as in [17], happens to become a wrong one, when more information is available. From this follows that the function of the search through the centralized databases should be changed from the positive selection as in [5] to filtering when some IPs are excluded from the further investigation because of unsatisfactory value of the components in the vector measuring a fit of the IP for the design in consideration.

In our system, filtering is made possible during a process of queries for the parameter values and categorical characteristics of the design. The selection process starts at the centralized databases by one software agent automatically migrates to other software agents on the vendor sites. At the vendor sites a search evaluation process either terminates prematurely because of gross mismatch between expected and available IP characteristics, or successfully goes through all the steps of search followed by simulation-evaluation steps.

In this work, the process of evaluation of the match between expectations of the user and real IP characteristics is based on application of fuzzy logic in the form of memberships in fuzzy sets defined by a user and natural-language-like rules, capable to capture relative importance of different factors in evaluation of IPs.

The fuzzy logic inference engine provides evaluation of the design dynamically. It means that a measure of conformity of IP parameters to those expected by the user is evaluated after each new parameters is being brought into consideration.

To alleviate a major concern of IP vendors, related to release of models of their IP solutions for testing and evaluations, this system allows to conduct simulation on the vendor site under the supervision of the local software agent. For the purpose of matching high-level models from the

user and their low level implementation by IP the high level model and the IP model are jointly simulated on the vendor site, and only results of such simulation and their comparison are sent back to the user via the Internet.

Current tools and techniques for IP selection are not adapted to changing technologies, and are not capable of handling heterogeneous data structures and computation environments.

Our goal is to make a selection process faster, easier, and drastically improve a match between expectations of a user and characteristics of the IP designed by a vendor. This goal is achieved by establishing a web-based framework, which integrates IP databases, vendor's web-sites, web-based standard tools, web-based custom tools, and CAD tools into unified interactive collaborative environment supporting a design process.

In this part, we present our approach to solve the problem of IP selection. Section 2.1 gives the overview of the framework. Section 2.2 presents the structure of software agent system. Section 2.3 introduces the fuzzy logic inference engine utilized in this work. Section 2.4 describes the distributed simulation environments.

2.1 Overview

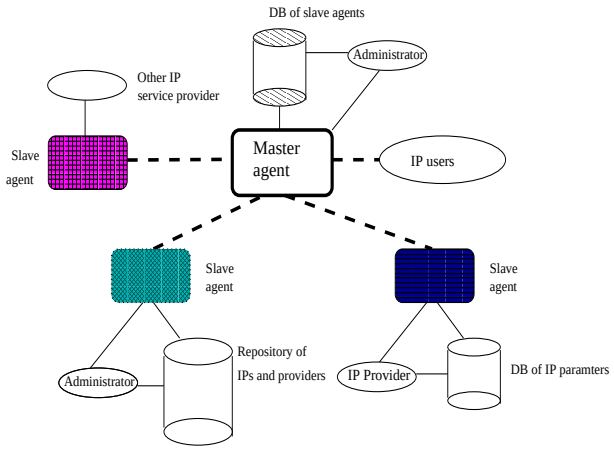


Figure 1: Master agent and slave agents for IP selection

In this paper we describe main features of the system. One is a structure of the software agents, positioned at different locations on the web. Conceptually we can identify three major participants in the IP selection process: IP users, IP vendors and the environment that supports communication between sides which we call SAFIPS. Software agents are using filtering and fuzzy logic in the process of decision-making. We will describe fuzzy logic aspect of the system later. Fig. 1 conceptually illustrates the structure and the interaction between SAFIPS, vendors and users.

2.2 System of agents

In this system, there are two types of agents: master agents and slave agents. The master agent provides a user interface to specify the requirements for IP cores. It consists of the following components.

1. Management tools for a database with the generic information about available IPs and contact information for a group of slave agents.

2. User-SAFIPS dialogue facility that creates fuzzy logic decision tree for IP selection in the process of a dialogue with the customer.

3. Fuzzy logic rule inference engine. This part of the system contains user-specific IP information in a form of membership functions and fuzzy logic rules.

4. Databases and file management systems for users. They maintain user's information such as accounts, inquiries, etc.

Slave agents represent individual IP vendors, IP repository sites and other IP services. Each slave agent has its own local database, which maintains information about products and services. These databases are inherently heterogenous and distributed on the Internet. The slave agents take responsibilities to keep the information current. The slave agents also know the address of the master agent. A slave agent mainly consists of the following components.

1. Management tools for a database which records information about products and services provided by the specific vendor or organization.

2. User-SAFIPS dialogue facility which allows users to inquire information interactively.

3. Simulation environment. It makes possible for users to simulate the IP remotely before starting negotiations about acquisition.

It is not practical to support a centralized DB which contains complete information on all the IPs. There are several reasons for it. The first reason is that there is no standard for a format and contents of the information necessary and sufficient for a potential user. The second reason is that IPs may have unique features in architecture, implementations, etc., which cannot be covered by any uniform format. The third reason is that some information is very volatile and updated frequently due to changes in proposed solutions, prices, etc. Therefore, in the proposed structure, the master agent considers centralized DBs of IPs repositories of "general" information and expects to get "detailed" information from the slave agents that resides on the vendor sites.

2.3 Fuzzy logic inference engine

In this work, we apply fuzzy logic inference engine to evaluate IP cores. Fuzzy logic [10] has been widely used in many fields of science and engineering. One of its merits is in providing convenient vehicle to balance or optimize multiple objectives. [11]

Reasoning is based on fuzzy logic rules, which are specified in response to prompts by users and are introduced in a simple *IF/THEN* form. An example of the rule is *IF (hardness is satisfactory_hardness) AND (target_technology is satisfactory_technology) AND (architecture is satisfactory_architecture) THEN general_information is satisfactory_information.* In this rule, every clause provides a level of membership in the respective fuzzy set of satisfactory solutions for each important category and connectives are interpreted according to rules of fuzzy logic.

Two basic operations in fuzzy logic are *intersection* and *union*, which are interpreted as fuzzy

AND and *OR* respectively. Some classes of operators for fuzzy intersection and union are given in [12]. In this work, the following operators are used for in the example given in section 4.3.

- Algebraic product: $Intersection(a, b) = ab$.
- Algebraic sum: $Union(a, b) = a + b - ab$.

These two operations are compensatory, i.e. the result of the operation depends on membership in both fuzzy sets.

The system also allows users to give preferences to some parameters over the others. In such cases, exponents are used to reflect the preferences. See Appendix for more details.

2.4 Online simulation

Users may need to check whether or not the IP functionality is consistent with the desired behavioral model. In general, simulation and formal verification are the two major methods to establish equivalence between the behavioral model, presented by the user, and the RTL level model, which represents the highest level of IP model.

The formal verification for such a problem definition is now impractical in the majority of cases. Only simulation is left as an acceptable solution.

Currently, some IP providers release their protected models online for demonstration or free download. Users are allowed to customize models for simulation and evaluation [3]. This implies that the users have the simulation environment and necessary EDA tools for the product design at this phase. A distributed simulation environment has been recently proposed in [20]. In this case, the simulation process could be very slow and even abort in case when the network traffic is heavy and the circuits are complex. That may lead to a huge number of interactions across the whole Internet.

The simulation environment in our project is established at the site designated by an IP vendor. Users can access the website of the slave agents to upload the test bench, which will be forwarded

to the simulation environment. After the files pass compilation, the simulation is run at the back end and then the result will be sent back to the users. The whole process is automated.

Presently, our simulation environment is based on V-System/PLUS Workstation simulation system of the Model Technology Incorporated. It supports both the IEEE 1076-1987, 1076-1993 VHDL standards, and IEEE 1364 Verilog Hardware Description Languages.

3 Selection and filtering

The IP selection is performed in three basic steps. On the first step, users make a preliminary selection to narrow the list of the candidates by acquiring websitei address of providers. On the second step, users communicate with the slave agents to get more detailed information. The third step is an optional one, on which users submit test benches to test the functionalities of the cores of interest.

3.1 Step 1: Preliminary selection

On this phase, fuzzy logic rules are used to evaluate candidate IP cores. Users login the website of the master agent. At this website, users are directed to fill up a series of forms to create an inquiry for a desired IP core. For instance, a user may specify some requirements like IP_TYPE = *(microcontroller), target_technology = (FPGA).*

In this work, IP parameters are divided into six groups: *general information, form factor, technical data, development environment, verification and testing, and business related issues.* For each group, users define the desired values of parameters or rate possible values for the parameters with satisfaction levels. Users supply evaluation rules for each of the six groups. Finally, users specify the overall evaluation rules for the IP cores. For example, a rule may look like: *If (gen_info is satisfactory_gen) and (tech_data is satisfactory_tdata) then IP is satisfactory_IP.* These steps establish an evaluation tree illustrated in Fig. 2.

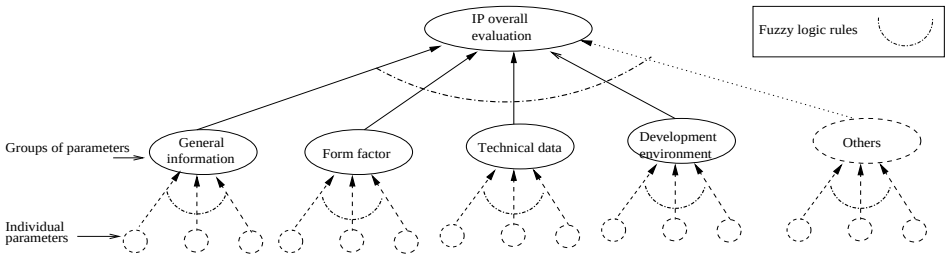


Figure 2: IP evaluation tree for "preliminary selection" by SAFIPS

As the second part of the preliminary selection, the system analyzes the set of the solutions, which are retrieved by the communications between the master agent and the slave agents. Fuzzy logic inference engine is used to rank these solutions with respect to the user requirements.

The system will produce a scalar number that presents a measure of the vector defined by the levels of satisfaction of the user by values of individual attributes in the design.

3.2 Step 2: Interaction with slave agents for more detailed information

After the first step, the list of the potential IP cores is narrowed. Users can decide to either communicate or not with the slave agents directly. At this step, users can ask more specific questions, which were not necessarily covered by the first step. Slave agents reply to the inquiries by retrieving the pertinent information from local databases. For some questions, there may be no ready answers. In this case, the software agent will ask the IP providers to supply needed information. If such new information is provided, the local DBs will be augmented to include the newly added information.

3.3 Step 3: Simulation

After the above two steps, users can basically find out which IPs are potential solutions. They can elect to run the simulation at agent sides of the IPs. User's model and testbench are uploaded to the agents. Next the simulation are launched by agents at back end. The simulation results for

different models are compared and sent back to users. The Fig. 2 shows the process.

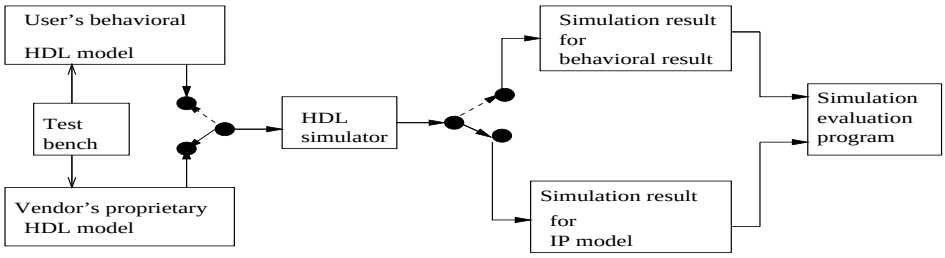


Figure 3: Simulation at agent side

4 Implementation

The prototype of SAFIPS was developed using Java, Javascript, Perl and C.

4.1 Design of software agents

We deployed a master agent and slave agents emulating different IP providers. One of the slave agents represents Silicore product SLC1655, which is VHDL 8-bit RISC Microcontroller Core [13].

There are basically three layers in communications between agents: transportation layer, syntax layer and semantics layer. The transportation layer is realized by a socket interface [24]. The socket interface is supported by many popular programming languages, and it could be utilized in most platforms. (Knowledge Query Manipulation Language) [9] is adopted as the Agent Communication Language for the syntax layer. KQML is a high-level, message oriented language, it is independent of the content language, transportation mechanism. A set of keywords and rules are defined for the semantics layer for interpretations. This layer unwraps and parses the content parts in the KQML message, and performs correspondent actions.

This architecture establishes an open environment. A KQML-speaking software agent can easily include itself into the framework by following the communication protocols.

The master and slave agents work in a client/server model. A slave agent opens a socket in a

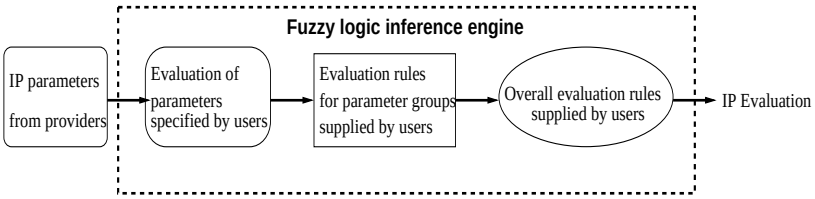


Figure 4: Process of IP Evaluation

host, waiting for connections from hosts located on the Internet. Upon user's request, the master agent creates a message written in KQML format, and then sends it via the socket to the target slave agent. The slave agent will unwrap and parse the information, and then generate a feedback.

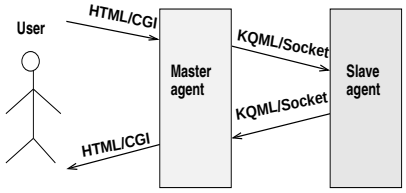
4.2 Preliminary evaluation of IPs using software agent and fuzzy logic

As described earlier, the *AND/OR* in fuzzy logic rules are interpreted as *intersection/union* respectively, and we use the bounded difference and bounded sum to implement the *AND* and *OR* operators.

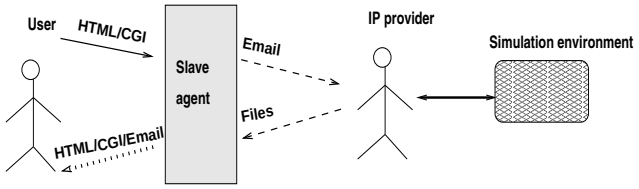
As shown in Fig. 2, evaluation is performed in a bottom-up fashion. For every individual parameter, groups of parameters and the overall evaluation, there is a related linguistic variable *satisfactory_term* with its own membership function. (Here, the *term* is the name of a parameter, a group, or IP.) Membership functions are constructed in the process of a dialog between the master agent and the user. On higher hierarchical levels for the groups of parameters and overall evaluation, membership in the fuzzy set is defined by fuzzy logic operation over memberships in fuzzy sets on the lower levels.

IP parameters are evaluated at the lowest level of the evaluation tree. Then groups of parameters are evaluated based on evaluation of individual parameters. And finally, the whole IP is evaluated based on evaluation of groups. This process is illustrated in Fig. 3.

These parameters can be classified into two types. 1) The first type covers numerical values (for example, clock rate). Users have to specify the parameter values for 100%, 80%, 60% and



(a) The interaction between user, master agent and slave agent



(b) The interaction between user, slave agent and IP provider

Figure 5: Major parts of communications in SAFIPS

40% levels of satisfaction. The pairs of values and satisfaction levels are used to construct piece-linear membership functions. 2) The second type defines categorical values. For parameters of this type, users also apply levels of satisfaction to describe the evaluation. For example, the source code format has two choices: VHDL and Verilog. A user can specify that the VHDL format is 100% satisfactory, and the Verilog format is 80% satisfactory. A special case of this type has two options: "yes" and "no". By selecting either of them, users indicate whether or not such features are mandatory. For instance, "on-site support" is a parameter of this subtype.

4.3 Example

The prototype was tested on selection of a microcontroller soft core, which is software compatible with the industry standard PIC16C5X [14].

Preliminary selection The part (a) of Fig. 5 illustrates communications between the user, the master agent and the slave agent.

1) The user applies an Internet browser to access the website of a master agent and logs onto the system.

2) The master agent provides a catalogue of IP types such as DSP, Microcontroller, etc. The user selects Microcontroller as a type and soft cores compatible with the industry standard PIC16C5X [14] as a subtype.

3) The user is directed to fill up a set of forms, each is related to one of the six parameter groups described earlier. Every form consists of two parts. (1) The first part is filled with the expected values for parameters or with evaluation of options for categorical parameters. (See 5.2 for details.) By default, the missing data are assumed to be on the level of 80% satisfaction. Users are allowed to change the setting. (2) The second part of the form contains evaluation rules for the group of parameters. The user also is expected to fill up a special form to specify rules for the overall evaluation of an IP.

Some details for step 3 are given below.

3.1) In the *General Information* form, parameters for microcontrollers are IP hardness, architecture, instruction set. Defaults for these parameters are *soft IP, Harvard architecture, RISC* respectively. In this experiment, the user requires *FPGA* as the target technology. The user also specifies a rule to evaluate the parameters of this group. For example data, see Table 1 and Table 2.

3.2) In the *Form Factor* form, the user specifies values that correspond to the 100%, 80%, 60%, 40% levels of satisfaction for the parameters of "I/O pin number" and "estimated gate number" respectively. (See Table 1.) The user provides a rule to evaluate the parameters of this group by filling in variables and choosing connectives from the group window. For example data, see Table 1 and Table 2.

3.3) In the *Technical Data* form, the user specifies the values that corresponds to the 100%, 80%, 60%, 40% levels of satisfaction for the parameters "clock rate" and "ROM size" respectively. (See Table 1.) The user also specifies a rule to evaluate the parameters of this

Table 1: Expected values for some parameters

Numerical Parameters					Other Parameters	
levels of satisfaction	1.0	0.8	0.6	0.4	Hardness	soft
I/O pin #	64	48	32	24		
estimated gate #	2000	3000	4000	5000	Target technology	FPGA
ROM size #	2K	1K	512	256		
clock rate (Mhz) #	80	60	40	20	Architecture	Harvard

group. For the example data, see Table 1 and Table 2. Fig. 6 shows the curve for the four membership defined above.

3.4) In the *Development Environment* form, the user rates HDLs. The user also specifies a rule to evaluate the parameters of this group. For the example data, see Table 1 and Table 2.

3.5) Assume that the user has no specific requirements for the parameters in *Verification and Testing* and *Business-related Issues* group, then he can directly fill up the *Overall Evaluation* form. In this form, the user supplies the rules to find a core with satisfactory *General Information, Form Factor, Technical data, and Development environment.* (See Table 2)

4) The user finally submits the requirements to the master agent and exits the system. The expected parameters and evaluation rules are separately saved to the disk files in the user's directory.

5) The master agent retrieves a list of IP providers which supply microcontroller cores compatible with PIC16C5X from its local database.

6) The master agent opens a socket to each of slave agents and sends inquiries to the slave agents for parameters of interest, such as architecture, target technology, maximal clock rate, etc. The inquiries are in KQML syntax.

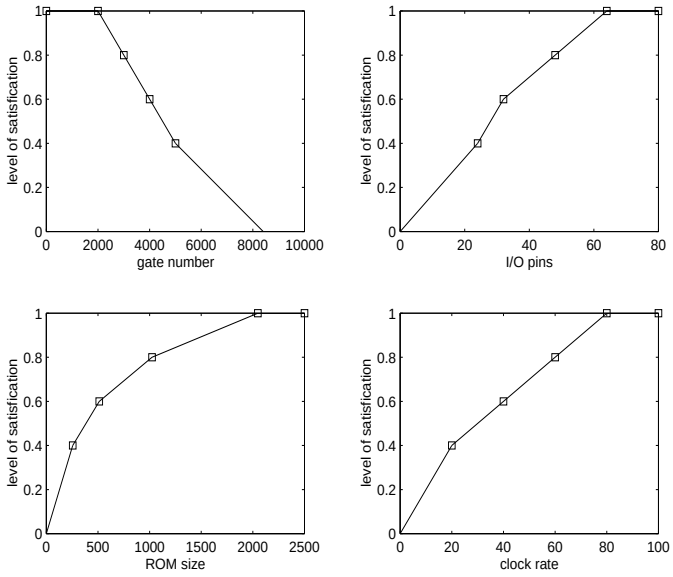


Figure 6: Membership functions in the example

Table 2: Rules used in the experiment

Groups of Parameters	Rules
General Information	IF (hardness is satisfactory_hardness) AND (target_tech is satisfactory_tgt) AND (architecture is satisfactory) THEN gen_info is satisfactory_gen
Form Factor	IF (I/Opin# is satisfactory_IO) AND (gate# is satisfactory_gnum) THEN form_fct is satisfactory_ffct.
Technical Data	IF (clk_rate is satisfactory_crate) AND (ROM_size is satisfactory_rsz) THEN tech_data is satisfactory_tdata.
Development Environment	IF (src_fmt is VHDL) OR (src_fmt is Verilog) THEN dev_env is satisfactory_denv.
Overall Evaluation	IF (gen_info is satisfactory_gen) AND (form_fct is satisfactory_ffct) AND (tech_data is satisfactory_tdata) AND (dev_env is satisfactory_denv) THEN it is satisfactory_IP.

For example, the master sends inquiries to all slave agents for information about the source code format of IPs. The message in KQML syntax is below.

```
(ask-all
      :sender      master
      :receiver    silicore
      :ontology    (source code)
      :content     (format)
      :reply-with  (query 5)
      :language    (plain text)
)
```

7) The slave agents residing on the Internet retrieve relevant information from their local databases upon the request, and send back the reply in KMQL syntax. For example, Silicore[tm] [13] SLC1665 is written in VHDL. The slave agent for Silicore replies to the inquiry for the source code format to the master agent by the message in KQML syntax.

```
(tell
      :sender      silicore
      :receiver    master
      :reply-to    (query 5)
      :ontology    (source code)
      :content     (VHDL)
      :language    (plan text)
)
```

If some parameters are not available, the slave agents will inform the providers via email and add the information to the databases.

Table 3: Parameters of PIC16C5X-compatible soft IPs

IP core	A	B	C	D
source code format	VHDL	VHDL	VHDL	Verilog
estimate gate #	3000	-	2700	1500
ROM size in words	512	2K	-	2K
RAM size in words	24	72	-	32
max clock rate(Mhz)	100	25	50	78
I/O pin #	72	24	20	24

Table 4: Parameter evaluation according to user defined memberships

soft IP cores #	A	B	C	D
rate for gate #	0.8	0.8	0.86	1.0
rate for ROM size #	0.4	1.0	0.8	1.0
rate for clock rate #	1.0	0.4	0.7	0.78
rate for I/O pin #	1.0	0.4	0.33	0.4

8) The master agent caches all the received information to local data files. In our project, we emulate four slave agents for IP cores. Some parameters are listed in Table 3. Among them, A is the Silicore[tm] SLC1665. The information about three other cores is collected from the public websites on the Internet [25], [26]. The examples listed here were available as of Oct. 1999. Due to the rapid changes in the Internet they may no longer be available.

9) The master agent feeds the data to the fuzzy logic inference engine. The evaluation is based on the parameter evaluation and fuzzy logic rules supplied by the user (see Fig. 2).

9.1) Every parameter is evaluated individually using the rules supplied by the user at steps 3.1-3.4. Evaluations of individual parameters for the experimental data are given in the Table 4.

Table 5: Final evalutions of IP cores by fuzzy logic rules

Soft core	A	B	C	D
Evaluation	0.32	0.13	0.16	0.31

9.2) Each group of parameters is evaluated using the fuzzy logic rules constructed at the steps 3.1-3.4.

9.3) The overall evaluation of each alternatives is performed by fuzzy logic inference engine using the rules constructed at step 3.5. It allows to rank solutions A, D, C, B from the highest to the lowest. (See Table 5)

10) The master agent sends back a result of ranking to the user as well as the website addresses of the slave agents for these cores via email.

11) The user revisits the master agent, requiring whether there is a standard license available for A and D. A responses "yes" while D responses "No". Therefore, D is removed from further consideration.

Final selection and simulation The part (b) of Fig. 5 illustrates communications between the user, slave agents, and the IP provider.

1) The user applies an Internet browser to access the website of slave agent for core A.

2) The slave agent for core A provides a dialogue facility for keyword-based inquiries. The interface of the facility has two parts. The first part allows users to enter one keyword at a time to retrieve related information. The second part displays the result of the previous inquiry.

The user continues the inquiry by using keywords such as "price" and then submits it to the dialogue facility.

3) The dialogue facility parses the keyword, then sends the keyword to the internal database search engine via a socket. The inquiry is done in plain text. For example: "subject: price"

will be sent to the DB search engine after the previous step.

4) The DB search engine retrieves the related information from its local databases. The reply for such information consists of two parts: the first part is a brief text relevant to the inquiry, the second part is optional. It contains a URL link connected to a file or webpage. The DB search engine sends back the reply to the dialogue facility via the socket. In case there is no information available in the database, it will also send an email to the providers saying "a customer is interested in the subject price, which can not be found in current databases!".

5) The dialogue facility parses the reply and displays the reply to the user. For the inquiry of "price", the slave agent displays something like "Developer's Kit $9,995. Evaluation Kit $1,500. Follow the link below for more information". A URL link *http://silicore.net/coreprce.htm* is also displayed.

6) The steps 2-5 can be repeated. The subjects of inquiries are recorded into a separate file "query.history". This file will be processed later to improve the quality of online queries.

7) If the user determines that the IP is a good solution, he requests a simulation by uploading his test bench written in VHDL for this soft IP. The test bench is transferred to the agent side. Then the user leaves the contact information and logs out of the system.

8) The slave agent notifies the provider for such a request via email saying "a customer liux0202@tc.umn.edu submitted a testbench for this soft core."

9) The provider retrieves the test benches and runs the simulation at his development environment, the result will be saved into VCD (Value Change Dump) format. Fig. 7 shows the waveform of the simulation. In the future, results of the comparison of simulation run for the testbench and soft IP will be evaluated by the fuzzy logic based procedures.

10) The date files of the simulation result are returned to the slave agent by the IP provider. The slave agent informs the user for the simulation result via an email with the data files attached.

11) After he receives the email, the user examines the simulation results for consistency with his models. If the results are satisfactory to the user, he can officially begin to negotiate acquisition of IP with the vendor.

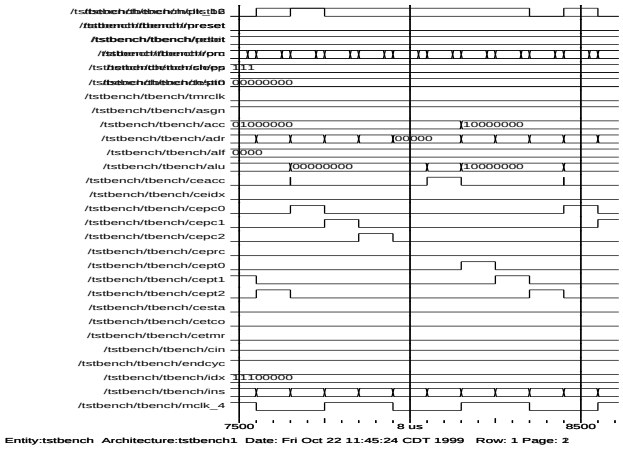


Figure 7: Simulation waveform window

5 Conclusion

The prototype of SAFIPS demonstrates the capability of software agents in facilitating IP selection. The system has the several advantages. It does not require establishing and maintaining huge centralized databases with the detailed information about IPs. Instead of this, the SAFIPS works with the decentralized information distributed over the web. The hierarchical structure of software agents allows to limit and simplify functions of each particular slave agent. Providers of IPs can easily customize their resident slave agents for their IP solutions without any modifications of the master agent and other slave agents. The SAFIPS interactively creates for each user his own decision-making structure in a form of fuzzy logic rules and membership functions and applies this information for selection of IP solutions. Distributed simulation capacity under supervision of SAFIPS is one more important advantage.

Overall, SAFIPS saves IP users' time spent on surfing websites for IPs and drastically improves

quality of solutions presented to the users for final consideration. Simultaneously SAFIPS helps providers of IPs by saving their time spent on answering many questions, which could be answered by the slave agent. SAFIPS helps providers to improve their service according to information accumulated by interactions between software agents and users.

6 Acknowledgements

This work was supported in part by the NSF Award No. MIP9628022.

7 Appendix: Basics of fuzzy logic

7.1 Fuzzy set

Unlike as in classical set theory in fuzzy logic an element may partially belong to a fuzzy set. An element may partially belong to a fuzzy set. A formal definition of a fuzzy set is given below:

Definition: If X is a collection of objects denoted generically by x, then a fuzzy set A over X is a set of ordered pairs: $\tilde{A} = \{(x, \mu_{\tilde{A}}(x)) | x \in X\}$, $\mu_{\tilde{A}}(x)$ is called the membership function which binds a number in the interval $[0, 1]$, representing the degree of membership of $x \in A$, to each element x of X.

The figure below illustrates a fuzzy set for "large memory". The universe of discourse X is defined as the interval [0, 32K].

7.2 Linguistic variables

Definition: A linguistic variable is represented by a quintuple $(x, T(x), U, G, \tilde{M})$ where x is the name of linguistic variable; $T(x)$ is the value set of x; U is the universe of discourse; G is a syntactic rule for generating the terms; and $\tilde{M}$ is a semantic rule, associating with each linguistic value its meaning $\tilde{M}(X)$, which is a fuzzy subset of U.

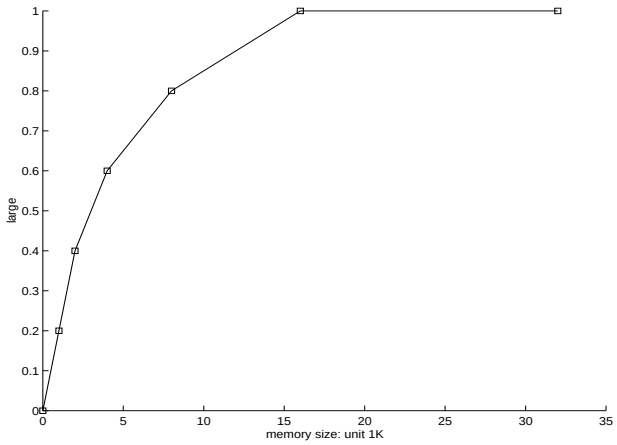


Figure 8: A fuzzy set for "large memory"

The Fig. 9 provides an example for linguistic variable named "memory size". For this instance, $T(x)$ includes following values: *large, median, small*. U is the interval $[0, 32k]$. The three curves define $\tilde{M}$ for each linguistic value respectively.

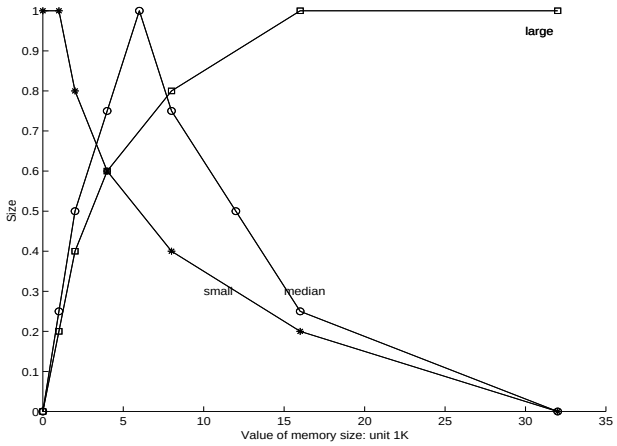


Figure 9: A linguistic variable for "memory size"

7.3 Operations over fuzzy sets

Set operators such as *union, intersection* can be extended into fuzzy sets. For example, in many fuzzy logic applications, *min* and *max* are used as the operators of *intersection* and *union* respec-

tively. Due to the nature of membership functions, results of fuzzy logic operations are also fuzzy sets.

The operators in fuzzy logic can be broken into two categories: compensatory and non-compensatory operators. If non-compensatory operators are used, the weaker elements are overshaded by the stronger elements. For instance, the *max* and *min* are operators of this type. On the other hand, the compensatory operators take into account all the informations on their operands.

In this work, compensatory operators AND and OR is defined as follows:

- Algebraic product: $Intersection(a, b) = ab$.
- Algebraic sum: $Union(a, b) = a + b - ab$.

7.4 Fuzzy logic rules

Fuzzy decisions are made using plain language rules, which describe relation between linguistic variables with their membership values. The rules are presented in a simple $IF/THEN$ form. The input values in IF part can be combined using AND and/or OR operations. These connectives are mapped into *intersection* and *union* of fuzzy sets. The predicate part assigns a rate of membership, computed by the left part of the rule to the candidate node in the fuzzy set defined by the right part of the rule. This rate of membership could vary for different candidate nodes. Also $THEN$ part of a fuzzy rules may characterize more than one linguistic variable.

7.5 Preferences in fuzzy logic rules

In a fuzzy logic rule, the degree of importance associated with each criteria may be different. Users are allowed to assign preferences to individual criteria. The below statement is such an example.

C1 has (weak, moderate, strong) preference over C2, where $C1$ and $C2$ are two criteria in the IF part of a fuzzy logic rule.

There are many methods to specify preferences. A common method is to give weighting coefficients to each criterion. Another method is to assign different weights as exponents to each criterion. Different ways can be adopted to set the exponent numbers for preferences. In this work, the Table 6 defines the numbers for preference $C1$ over $C2$.

Table 6: Exponents for preferences of C1 over C2

degree of preference	AND		OR	
	w1	w2	w1	w2
no preference of C1 over C2	1	1	1	1
weak preference of C1 over C2	3/2	2/3	2/3	3/2
moderate preference of C1 over C2	2	1/2	1/2	2
strong preference of C1 over C2	5/2	2/5	2/5	5/2

Taking preferences into consideration, the *intersection* and *union* are now tuned respectively as below:

$Intersection(a, b) = Intersection(a^{w1}, b^{w2})$

$Union(a, b) = Union(a^{w1}, b^{w2})$

7.6 Soft fuzzy reasoning approach

When a complex solution is evaluated by multiple criteria, a hierarchical structure is usually adopted. Such a structure consists of multiple levels. On the low levels of hierarchy, linguistic variables for various criteria are evaluated by fuzzy logic rules, and on the higher hierarchical level, the low level criteria are presented by the values of their membership functions in different fuzzy sets. These values of membership functions are used to generate a value of the membership for the function on the higher hierarchical level. Such a process continues until the rules on the top level of the hierarchy are executed. The Fig. 10 conceptually shows such a structure.

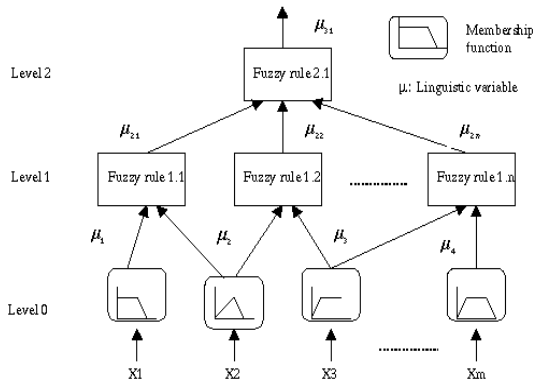


Figure 10: The hierarchy of fuzzy logic reasoning

References

[1] Michael Keating and Pierre Bricaud, "Reuse Methodology Manual", Kluwer Academic Publishers, 1998.

[2] D. D. Gajski, "IP-Based Design Methodology" proc. of the Design Automation Conference, page 43, June 1999.

[3] Xilinx CORE Generator System. http://www.xilinx.com/products/logicore/coregen/ 1999.

[4] R.B. Doorenbos, O. Etzioni, and D.S. Weld, "A scalable Comparison-Shopping Agent for the World-Wide Web", Technical Report, Department of Computer Science, University of Washington, 1996.

[5] P. Oehler, I. Vollrath, P. Conradi, R Bergmann, T. Wahlmann, "READEE-Decision Support for IP Selection Using a Knowledge-based Approach", Proc. of IP'98, Frankfurt, pages 229-241, 1998.

[6] S. Olcoz, J. L. Avellano, L Ayuda, and J. M Insenser, "A Hierarchical Database Approach for Soft-Cores Design Reuse", Proc. of IP'98, Frankfurt, pages 245-251, 1998.

[7] W. Eisenmann, S. Scharfenberg, D Seidler, J. Geishauser and H. Ranise, "Hard IP Reuse Methodology for Embedded Cores", Proc. of IP'98, Frankfurt, pages 311-315, 1998.

[8] S. Han, D. Boley, M. Gini, R. Gross, K. Hastings, G. Karypis, V. Kumar, B. Mobasher, and J. Moore. "WebACE: A Web Agent for Document Categorization and Exploration", Proc. of Autonomous Agents'98 Conference. Minneapolis, May 1998.

[9] Y. Labrou and T. Finin. A Proposal for a New KQML Specification. Te ch. Report TR-CS-97-3, Computer Science and Eletrical Engineering Dept., Univ. of Maryland, Baltimore County, Baltimore Md, 1997.

[10] H. J. Zimmermann. "Fuzzy set theory and its applications". 3rd Edition. Kluwer Academic Publishers, 1996.

[11] J.Y. Lee. "Multi-objective Technology Mapping For Field-Programmable Gate Arrays". Ph.D. dissertation, Dept of CSE, Univ of Minnesota May 1996.

[12] G.J. Klir and B. Yuan. "Fuzzy sets and fuzzy logic: theory and applications". Prentice Hall, 1995.

[13] Homepage of Silicore. http://www.silicore.net 1999

[14] Myke Predko, "Handbook of microcontrollers". GcGraw-Hill, 1999.

[15] http://www.synopsys.com/partners/ip/ip_more.html 1999.

[16] Homepage of Virtual Socket Interface Association. http://www.vsia.com 1999.

[17] Homepage of Reusable Application-Specific Intellectual Property Developer. http://www.rapid.org 1999.

[18] Homepage of Virtual Component Exchange. http://www.vcx.org 1999

[19] Homepage of Design & Reuse. http://www.design-reuse.com 1999

[20] M. Dalpasso, A. Bogliolo, and L. Benini. Virtual Simulation of Distributed IP-based Designs. Proc. of the Design Automation Conference, pages 50-55. June 1999.

[21] G. Konduri and A. Chandrakasan. A Framework for Collaborative and Distributed Web-Based Design. Proc. of the Design Automation Conference, pages 414-419. June 1999.

[22] F. Chan, M. Spiller and R. Newton. WELD - An environment for Web-based electronic design. Proc. of the Design Automation Conference, pages 146-151. 1998.

[23] P. Molson. Accelerating the IP Evaluation Process in Programmable Logic. pages 255-261. Proc. of IP'98 Frankfurt.

[24] A. S. Tanebaum. Computer Networks. 3rd Edition, Prentice Hall, March, 1996.

[25] http://www.mindspring.com/ tcoonan/synthpic.html. 1999.

[26] http://www.sican.com/. 1999.