Open access • Book Chapter • DOI:10.1007/978-3-7908-1837-6_7

A generic architecture for hybrid intelligent systems — **Source link**

Hans-Arno Jacobsen

Institutions: Lawrence Berkeley National Laboratory

Topics: Intelligent decision support system, Systems design, System of systems, Hybrid system and Knowledge-based systems

Related papers:

- A generic architecture for hybrid intelligent systems

- Multiple Approaches to Intelligent Systems

- From model-based strategies to intelligent control systems

- Integrating agents and soft computing in Intelligent Manufacturing System models

- A new methodology for intelligent and distributed process control based on neural networks

A generic architecture for hybrid intelligent systems

Hans–Arno Jacobsen

Institute of Information Systems

Humboldt University, Berlin

Abstract— The integration of different learning and adaptation techniques in one architecture, to overcome individual limitations and achieve synergetic effects through hybridization or fusion of these techniques, has in recent years contributed to a large number of new intelligent system designs. Most of these approaches, however, follow an ad hoc design methodology, further justified by success in certain application domains. Due to the lack of a common framework it remains often difficult to compare the various systems conceptually and evaluate their performance comparatively.

In this paper we first aim at classifying state–of–the–art intelligent systems, which have evolved over the past decade in the soft computing community. We identify four categories, based on the systems' overall architecture: (1) single component systems, (2) fusion–based systems, (3) hierarchical systems, and (4) hybrid systems.

We then introduce a unifying paradigm, derived from concepts well known in the AI and agent community, as conceptual framework to better understand, modularize, compare and evaluate the individual approaches. We think it is crucial for the design of intelligent systems to focus on the integration and interaction of different learning techniques in one model rather then merging them to create ever new techniques.

Two original instantiations of this framework are presented and discussed. Their performance is evaluated for prefetching of bulk data over wireless media.

Keywords— Intelligent systems, neuro–fuzzy approaches, agent paradigm, intelligent prefetching over wireless media.

I. Introduction

Complex adaptive systems, also referred to as *intelligent systems*, have in recent years been developed for modeling expertise, for decision support, and for process control, among others. Many of these approaches go beyond simply applying one problem solving technique, but rather, combine different knowledge representation schemes, decision making models, and learning strategies in one system. This integration aims at overcoming limitations of individual techniques through hybridization or fusion of various techniques.

These ideas have lead to the emergence of many different kind of intelligent system architectures in the soft computing literature in the past decade. We have identified four categories based on the systems' overall architectural design. We distinguish between single and multi–component approaches and between hybridization and fusion based approaches.

Most systems are designed in an ad hoc manner, further justified by demonstrations of successful applications. Such approaches make it often hard to adapt the design ideas to domains governed by different conditions. Moreover, this

makes it difficult to compare the individual approaches and evaluate their relative performance, since no common basis for such comparison is available. It remains therefore difficult to precisely pinpoint merits and demerits of the different approaches. Especially, when new techniques, based on the integration of know algorithms, are proposed this is a major drawback.

In an attempt to alleviate these problems we introduce a paradigm, well known in the AI and agent community, as conceptual framework to better understand, modularize, evaluate, and compare the individual approaches. This framework defines an intelligent system in a modular manner which allows one to focus on the interaction of different system components and their overall utility for the problem solving task.

We think it is crucial for the design of intelligent systems to primarily focus on the integration and interaction of different techniques rather than merge different methods to create ever new techniques. Techniques, already well understood, should be applied to solve specific domain problems within the system. Their weaknesses must be addressed by combining them with complementary methods. The focus must therefore lie on the component–wise integration of different methods and be geared towards studying their mutual dependencies, synergetic effects and precise interactions.

The rest of the paper is organized as follows. Section II surveys state–of–the–art intelligent systems and identifies four distinct categories. Section III reviews the *agent paradigm* and illustrates how it is applied in the design of intelligent systems. We instantiate it with two original systems. Section IV validates these systems experimentally.

II. Intelligent systems based on neural and fuzzy techniques

Neural networks are well suited for learning and adaptation tasks. In general, however, a neural network constitutes a black box. This means it is not possible to understand how a neural system works. Furthermore, it is very hard to incorporate human a priori knowledge into a neural network. This is mainly due to the fact that the connectionist paradigm gains most of its strength from a distributed knowledge representation.

Fuzzy knowledge based systems, on the other hand, exhibit complementary characteristics. The incorporation and interpretation of knowledge is straight forward, whereas learning and adaptation constitute major problems. Table I gives a more clear cut juxtaposition of these characteristics.
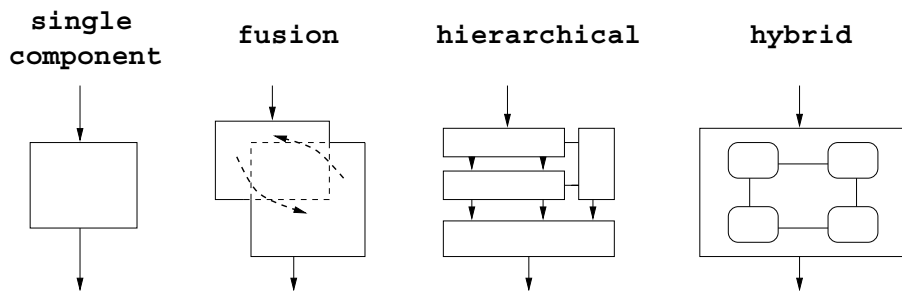
Fig. 1. Different categories of intelligent system designs based on neural and fuzzy techniques.

TABLE I

JUXTAPOSITION OF COMPLEMENTARY CHARACTERISTICS OF PURE
NEURAL AND PURE FUZZY SYSTEMS.

neural concepts	fuzzy concepts
learnable, adaptive devices	static devices. a priori non–adaptive
black–boxes, not interpretable	rule–based, interpretable
learns from scratch	domain knowledge expressed in rules

Due to this complementarity it is not surprising that many approaches have evolved which combine neural network and fuzzy techniques in one system. A complete survey of all these approaches is out of the scope of this paper. Rather, we try to focus on the different kind of architectures developed over the past few years. Figure 1 depicts the different approaches metaphorically, characterized below. The four categories we have identified are: *single component system*, *fusion based systems*, *hierarchical systems*, and *hybrid systems*. The boundaries between the different categories are not strict. For many cases one could argue that a given system could belong to more then one class.

The **single component system** class contains systems based solely on one technique. It contains the 'puristic' approaches, such as plain fuzzy control, TSK–control, or multi–layered perceptron based approaches. Many successful applications of such approaches have been demonstrated in the literature.

The **fusion based system** class includes systems which combine different techniques into one single computational model. Instances of this class are, for example, AN-FIS [Jan91], NEFCON [NK92], FUN [STGV93], Eppler's approach [Epp93], FINEST [TOA96], and FLINS [Oea95], among many others.

Common to these approaches is their network–like architecture which is often based, in one way or another, on the five staged fuzzy rule base evaluation scheme (fuzzification, premise evaluation, truth value propagation, conclusio aggregation, and defuzzification). Like the approaches in the previous class, these systems realize a mapping from an input space to an output space. The system does not contain other components which perform strategic planing or self–assessment.

The **hierarchical system** class comprises more architecturally complex systems. Its instances are build in a hierarchical fashion, associating a different functionality with each layer (e.g., preprocessing of sensor data, planing, and action selection). The correct functioning of the system, thus, depends on the correct operation of all layers; a possible error in a lower layer is propagated up through the hierarchy directly effecting the system output. Examples are [ATFS95], [Tan97], and [TNS+97], among others.

Finally, the **bybrid system** class contains approaches that put different techniques on a side by side basis and focus on their interaction in the problem solving task. It is this interaction which we deem important, since it allows to integrate alternative techniques and exploit their mutuality. Furthermore, the conceptual view of the agent allows one to abstract from the individual techniques and focus on the global system behavior, as well as study the individual contribution of each component. Examples are ARIC [Ber92], GARIC [BK92], SHADE [JIG94], [JI94] and our work on the fuzzy relation adaptation architecture [Jac95], [JW96].

We are proponents of this latter class of systems. As we believe, they exhibit greater potential for solving difficult tasks (learning, classification, and control), due to the inherent self–assessment capabilities of the approaches and their potential to gracefully degenerate with the loss or unavailability of one of their component functions.

III. A MODULAR ARCHITECTURE FOR HYBRID INTELLIGENT SYSTEMS FOLLOWING THE AGENT PARADIGM

We now review an abstract framework of a learning agent architecture to more easily capture the complexity of intelligent systems, to better understand and modularize such systems, and to obtain a terminological framework within which future intelligent system designs can be evaluated and compared. We illustrate how this framework can be used to instantiate intelligent systems by discussing two architectures derived from it. The framework is based on the rational agent approach introduce by Russell and Norvig [RN95] who describe a *rational agent* as *'something that perceives and acts in a rational way'*. The general agent architecture is given in Figure 2 (adapted from [RN95]). This framework should be seen as a tool to characterize and analyze complex intelligent systems. Its basic prin-
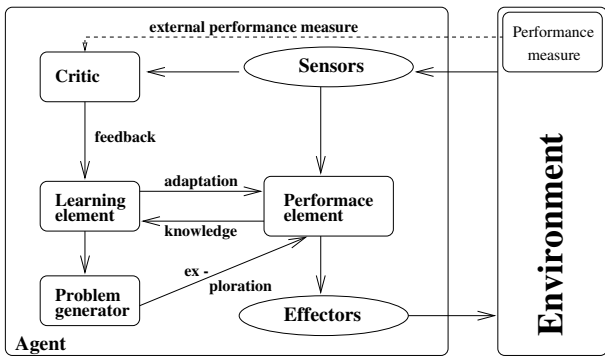
Fig. 2. Conceptual learning agent architecture according to Russell & Norvig (adapted from [RN95]).

ciples may be found with varying degrees in all systems presented earlier. The architecture consists of four components [RN95] and the *environment* upon which the agent acts:

The **environment** constitutes the problem task, e.g., the process to be controlled, the decision space to be analyzed, or the learning problem to be solved. Abstractly speaking, it is described by a state vector perceived by the agent through its **sensors** and influenced by it through its **effectors**.

The **performance element** (PE) is the actual 'controller' mapping environment states to actions.

The **learning element** (LE) updates the knowledge represented in the PE in order to optimize the agent's performance w.r.t. to an outside performance measure. It has access to the environment state, the agent's past actions, and an immediate reinforcement signal indicating the appropriateness of the action that last influenced the environment state. Given this information it updates the PE so that in future situations more pertinent actions are chosen over less pertinent ones.

The **critic** faces the problem of transforming an external reinforcement signal into an internal one. The crux is that the external reinforcement signal may be very poor, an indication of failure, for example, and it may even be delayed, indicating failure after an entire sequence of actions has influenced the environment. The internal reinforcement signal, on the contrary, must be more informative and immediate. It indicates for each action taken whether it was beneficial or not.[1]

The **problem generator**'s role in the agent architecture is to contribute to the exploration of the problem space. Abstractly speaking, it proposes different actions

---

[1] This problem is discussed in the AI literature as *credit assignment problem* (CAP): *depending on the overall situational outcome credit or blame have to be distributed among actions and decision steps involved in the agent's reaction behavior.* Sutton [Sut84] differentiates between *temporal credit assignment* and *structural credit assignment*. Temporal credit assignment is the distribution of credit for outcomes to actions. The question is *when* the action occurred that caused the outcome. Structural credit assignment is the distribution of credit for actions to internal decisions that caused the action. It effects the internal structure of a system. With this differentiation it is clear that the critic faces the temporal CAP and the LE the structural CAP.

which might lead to the discovery of new and better solutions. In most existing systems it is realized by adding a small amount of random noise to the output action. The amount added depends on the system performance. If the system performs well the need for new and better solution is not as urgent as if it performs poorly.

Clearly, this framework does not require the use of a specific technique for realizing the individual components. These techniques may be chosen entirely according to their strength and according to the problem task at hand.

Russell and Norvig [RN95] present a rich set of instantiations of this framework with diverse machine learning techniques implementing different component functions. In the soft computing community, however, little attention is paid to conceptual learning agent frameworks. We are aiming at closing this gap.

We now introduce two instantiations of this framework partially explored in previous work [Jac92], [JI94], [Jac95], [JW96], but never explored as entire architectures. In Section IV we will demonstrate their effectiveness on different applications. Figure 3 and 4 show how their architectures derive from the framework presented above. **Reinforcement–driven fuzzy–relation–adaptation** The PE is instantiated by a rule–based fuzzy
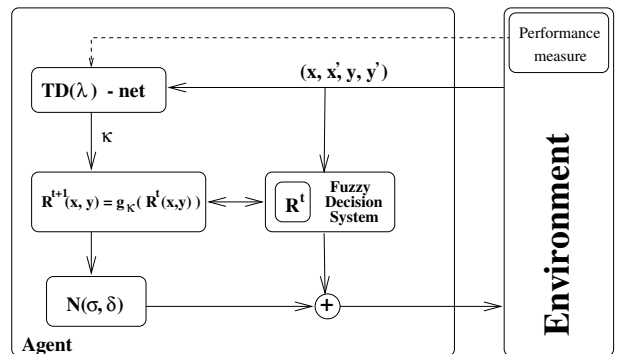

Fig. 3. Reinforcement–driven fuzzy–relation–adaptation architecture.

decision support system, i.e., a mapping from observed input states to an output decision (action). The mapping is defined by a fuzzy rule base.

The following discussion is based on previous work presented in [Wei95] and [JW96], [Jac95]. In the following, without loss of generality, we will consider only a *one input/one output* system. All results are easily extended to systems with many input and many output variables. We will therefore, consider here just two variables, the input variable x and the output variable y, with their respective universes of discourse U_x and U_y. Additionally, we denote the generic elements of U_x and U_y by u and v, respectively. Furthermore, let $\tilde{A}_i$ and $\tilde{B}_i$ represent fuzzy sets on the universes of discourse U_x and U_y, respectively ($i \in \{1, \ldots, n\}$). We will denote the fuzzy rule "IF x is $\tilde{A}_i$ THEN y is $\tilde{B}_i$" by $[\tilde{A}_i \Rightarrow \tilde{B}_i]$.

The general framework for handling a fuzzy rule base $[\tilde{A}_i \Rightarrow \tilde{B}_i]$ is to transform each rule into a fuzzy relation

$\tilde{R}_i = transform(\tilde{A}_i, \tilde{B}_i)$ on $U_x \times U_y$, to aggregate these implication relations to $\tilde{R} = aggregate(\tilde{R}_i)$, and to apply the resulting so called meta rule $\tilde{R}$ by using max–min composition. That is, given the actual input $\tilde{A}'$ on U_x, the result $\tilde{B}'$ on U_y of applying the fuzzy rule base $[\tilde{A}_i \Rightarrow \tilde{B}_i]$ is determined by computing:

$$\tilde{B}' = \tilde{A}' \circ \tilde{R}, \tag{1}$$

$$\mu_{B'}(v) = \max_{u \in U_x} \min\{\mu_{A'}(u), \mu_R(u,v)\}. \tag{2}$$

The meta rule $\tilde{R}$ is given as follows:

$$\tilde{R}_i = transform(\tilde{A}_i, \tilde{B}_i) = \tilde{A}_i \cap \tilde{B}_i, \quad \text{and} \tag{3}$$

$$\tilde{R} = aggregate(\tilde{R}_i) = \bigcup \tilde{R}_i = \bigcup_i (\tilde{A}_i \cap \tilde{B}_i). \tag{4}$$

with the appropriate t/t–co–norms (*min* and *max* in our case. The meta relation can thus be computed, given the fuzzy production rules governing the problem task. However, during the adaptation step there is no need to consider the way the meta rule is constructed. We may just take the fuzzy relation $\tilde{R}$ for granted and adapt it according to the critic's reinforcements. This is due to the fact, that *any* meta rule is processed using max–min composition according to eq. (1).

The LE is instantiated with the reinforcement–driven fuzzy relation adaptation algorithm developed for the aggregated relation representation of the fuzzy rule base.

Given the input x and the output y we know exactly *how* and *why* the selected decision was chosen from the set of possible decisions. Observing the effect of the output on the process it becomes thus possible to reinforce the selection of the same control action or to suppress its selection in future situations. This is achieved by directly modifying the underlying knowledge relation. Clearly, the objective is to reinforce good actions and to suppress bad actions. Several different *reinforcement schemes* for updating the relation have been developed: *point–wise updates*, *neighborhood incorporating updates* and *fuzzy set oriented updates*. We now denote $\mu_R(u,v)$ by $\tilde{R}(u,v)$ to better emphasize the relational nature of the approach. Note, u, v denote the generic input/output variables, whereas x, y denote the specific system input and observed output.
Point-wise update:

$$R(x,y) = \min\{1, \max\{0, R(x,y)\} + \alpha\,\kappa\}\},$$

with $0 \le \alpha \le 1$ a learning rate and κ the reinforcement signal ($\kappa > 0$ for rewards and $\kappa < 0$ for punishments). The $\min, \max$ operations serve to enforce the boundary conditions.
Neighborhood incorporating update:
$\forall u_i \in U_x$ and $\forall v_j \in U_y$

$$R^t(u_i, v_j) = \min\{1, \max\{0, R^t(u_i, v_j)$$

$$+ \alpha\,\kappa\,R^t(u_i, v_j)\,e^{(-(d^{x,y}_{u_i,v_j})^2)/\sigma_t^2}\}\}$$

with α and κ as above, σ_t an adaptive variance[2] and d a distance measure. The adaptation is here additionally a function of time. With increasing time (number of iterations) the updated neighborhood decreases, finally converging to the center point.

Fuzzy set oriented update:
$\forall u_i \in U_x$ and $\forall v_j \in U_y$

$$R(u_i, v_j) = \max\{\gamma\,R(u_i,v_j), \min\{\mu_{I_{w_2}}(u_i), \mu_{O_{w_1}}(v_j)\}\},$$

with $0 < \gamma \le 1$ a discount factor and w_1, w_2 parameters specifying the fuzzy set $\tilde{I}$ on the input domain and the fuzzy set $\tilde{O}$ on the output domain centered around the crisp state-action pair (x, y). The discount factor γ was introduced to discount the relation in situations where the process response patterns change.

A more detailed description of this algorithm, evaluation of its behavior (performance and robustness) and its application to control problems may be found in [JW96], [Jac95]. Its instantiation in the above framework and combined use with a critic is original to this work.

The PG is a module that adds a small amount of random noise to the output, depending on the performance of the system.

The critic is implemented by a feed–forward neural network trained with the TD(λ) rule [Sut88].

This approach resembles the adaptive critic developed by Barto *et al.* [BSA83], with the obvious differences in terms of component techniques applied here.

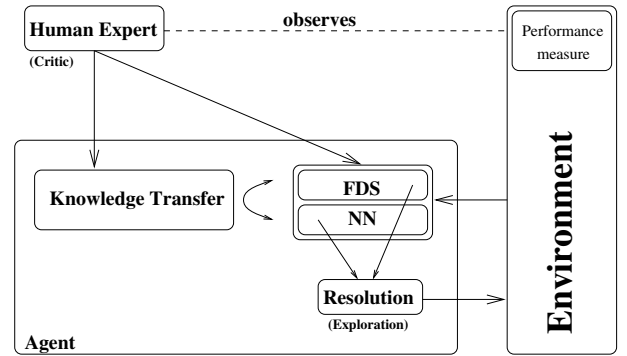**Expert–guided hybrid neuro–fuzzy system** The PE is

Fig. 4. Expert–guided hybrid neuro–fuzzy system

instantiated by two entirely independent modules, a feed–forward neural network (NN) and a rule–based fuzzy decision support (FDS) system. Both map the system input state to an output decision (action). The neural network component is trained on a sample set (if available) by back-propagation. The FDS is derived in concert with a human domain expert and an additional fine tuning stage. In our system we rely on a manual tuning stage, but any number of algorithms explored for this design stage could equally well be employed. The final PE output is determined by a

[2]The adaptive variance: $\sigma^t = \sigma_{initial}\left(\frac{\sigma_{final}}{\sigma_{initial}}\right)^{t/t_{max}}$.

conflict resolution scheme and depends on the operational mode of the system. We chose mechanisms as developed in our previous work [JI94], [JIG94] (emphasize on NN or on FDS depending on the domain knowledge available; action combination through a resolution operation.) To obtain exploratory behavior these resolution schemes must be adapted to account for a slight degree of randomness according to the system performance.

The LE is a *knowledge transfer* component which allows knowledge to be explicitly transferred from either one of the modules, inherent to the PE, to the other. Several modes of operation are possible, see Table II.

TABLE II

Knowledge transfer modes.

From NN to FDS
Rule extraction algorithm explored in [JI94] that describes the neural network in terms of fuzzy production rules.
From FDS to NN
Bootsrapping of a neural network with the inverse of the extraction algorithm described in [JI94].
Generation of samples by applying randomly generated inputs to the FDS, inferring an output and training the NN with the random sample.

The system requires the assistance of a human critic which evaluates the decisions inferred and possibly intervenes to tune the system's parameters until it operates reliably. This approach is less appropriate for online learning due to the human interactions required. We plan to experiment with an automated critic in the future.

The overall architecture derives from the SHADE system fully explored in [Gia92] incorporating a symbolic, on classical logic based, expert system. In [Jac92], [JIG94], [JI94] we extended the system to incorporate a fuzzy inference mechanism, generalizing the expert system component of the system, and extend the knowledge transfer operations available.

In the system instantiations presented here we have mainly focused on neural network and fuzzy system based techniques for realizing the individual components. Clearly, other techniques might equally well be used instead, e.g., believe networks, dynamic believe networks, decision trees, or symbolic processing techniques. Moreover, the individual components may be arbitrarily complex constituting any one instance of the architectures presented in Section II. We leave such exploration open for future work.

IV. Intelligent prefetching for mobile aware applications in wide area wireless networks

Wireless networking is becoming an increasingly important communication means, yet high wide–area wireless data connectivity is difficult to achieve due to technological and physical limitations. To alleviate these problems an alternative has been proposed in [YJK97]. that places many high bandwidth local "islands" of *info–stations* dispersed throughout the low bandwidth wide–area wireless network. These info–stations are deployed in a transparent manner, often not geographically visible to the user. Applications must therefore be designed *mobile–aware* and be able to account for changing network characteristics by optimally utilizing the available network resources.

Ye *et al.*, [YJK97], experiment with an incremental map downloading application for road travelers employing different prefetching strategies which prove to hide latency from the user better than a mobile–unaware prefetching algorithm.

Ye *et al.* incorporate vehicle location and speed information into the prefetching algorithm. These hints are used to predict the user's future reference needs. This more exact information eases request prediction and consequently improves performance (i.e., decreases user perceived latency). However, their results indicate that, under certain conditions, little of the information prefetched is actually touched by the application, thus wasting valuable network resources.

We show in our experiments that the performance of this prefetching algorithm may be further enhanced, by incorporating an adaptive user model. Performance is measured as percentage of data touched from the amount prefetched.

The prefetching algorithm in [YJK97] implements the function:

$$f_{\texttt{Route}}((x,y),v,t) \to \{ \texttt{ Requestable pages } \}$$

where, $\texttt{Route}$, denotes start and end point of the journey, (x,y) denote the vehicles location relative to an origin chosen, v denotes the vehicle speed, and t denotes an attribute identifying the point in time in the journey.

We use both of the above developed architectures to experiment with this application and validate their design. Prior to retrieval, a request, generated by the regular prefetching algorithm, is additionally passed through the intelligent system, which assesses the utility of retrieving the page.

The FDS is initialized with a set of fuzzy rules which describe prefetching heuristics of the form: *If the speed is high then the utility of prefetching is low, if the speed is low then the utility of prefetching is high.*

The TD–net is trained online such that it predicts the 'time to page access'. This signal is used to reinforce the FDS's knowledge representation. The TD–net receives an external failure signal for every prefetched page which is overwritten without being touched. Results are given in Table III.

The second architecture is used with an uninitialized neural network, thus demonstrating the plain FDS performance.

The results shown compare the location and speed driven prefetching, the greedy prefetching [YJK97], and the *intelligent prefetching* algorithms. The results are preliminary. It can, however, be seen that the addition of a critic to the system, has considerable effects on the overall performance.

TABLE III

Utilization (%) of data retrieved.

Total (MB)	0.8	1.65	2.6	3.35	3.75	4.3	4.7	5.2
greedy	6.1	4.1	5.7	4,5	4.2	4.9	3,8	3.3
loc/speed	12.9	10.5	10.0	8.0	8.0	9.3	8.3	8.5
arch. (1).	43.2	44.1	41.6	38.9	36.7	47.1	41.3	46.2
arch. (2).	31.5	36.3	33.6	32.4	29.8	29.3	32.6	36.7

V. Conclusion

We argue that, to obtain a less ad hoc design methodology for designing intelligent systems, attribute should be paid to the learning agent architecture which has long been discussed in the AI community. We have presented two original instantiations of this architecture and have experimentally validated their designs.

Our experiments present a preliminary study of the interaction of the different components in the agent. We intend to further study their mutual effects and experiment with alternative instantiations.

Acknowledgements

References

[ATFS95] M. Arao, Y. Tsutsumi, T. Fukuda, and K Shimojima. Flexible intelligent system based on fuzzy neural networks and reinforcement learning. In *Proceedings of 1995 IEEE International Conference on Fuzzy Systems.*, number 5 in 1, pages 69–70. IEEE, March 1995.

[Ber92] H. R. Berenji. A reinforcement learning-based architecture for fuzzy logic control. *Int. J. Approximate Reasoning*, 6:267 –292, February 1992.

[BK92] H. R. Berenji and P. Khedkar. Learning and tuning fuzzy logic controllers through reinforcements. *IEEE Trans. Neural Networks*, 3:724–740, September 1992.

[BSA83] A. Barto, R. Sutton, and C. Anderson. Neurolike adaptive elements that can solve difficult learning control problems. *IEEE Trans. Systems, Man & Cybernetics*, 1983.

[Epp93] W. Eppler. *Pre-structuring of Neural Networks with Fuzzy Logic (in German)*. PhD thesis, University o Karlsruhe (TH), 1993.

[Gia92] A. Giacometti. *Hybrid models for expertise*. PhD thesis, Ecole National Supérieur des Télécommunications, Paris, Nov 1992. (in French).

[Jac92] H.-A. Jacobsen. Fuzzy inference in a hybrid expert system. Unpublished Report University of Karlsruhe (TH) (in French, 'Studienarbeit' carried out at LIFIA, Grenoble, France), Feb 1992.

[Jac95] H.-A. Jacobsen. Adaptive fuzzy systems for control. Master's thesis, University of Karlsruhe (TH), Karlsruhe, Germany, Aug 1995. (in English, also available as ICSI Tech Report).

[JI94] H.-A. Jacobsen and I. Iordanova. Approach to extraction of fuzzy production rules from a connectionist component of a hybrid expert system. In *6th International Conference on Artificial Intelligence: Methodology, Systems and Applications, September 1994, Sofia, Bulgaria*, Sofia, Bulgaria, Sep 1994.

[JIG94] H.-A. Jacobsen, I. Iordanova, and A. Giacometti. Extraction des regles floues dans une systeme hybride. In *5th International Conference on Processing and Management of Uncertainty*, Paris, France, July 1994. (in French).

[JW96] H.-A. Jacobsen and J. Weisbrod. Reinforcement–driven adaptation of control relations. In *North American Conference on Fuzzy Information Processing*, Berkeley, CA, July 1996. NAFIPS.

[Jan91] R. Jang. ANFIS: Adaptive-network-based fuzzy inference systems. *IEEE Trans. Systems, Man & Cybernetics*, 1991. submitted.

[NK92] D. Nauck and R. Kruse. A neural fuzzy controller learning by fuzzy error propagation. In *NAFIPS92*, pages 388–397, Puerto Vallarta, December 1992.

[Oea95] W. Okamoto and S. Tano et al. Flins-fuzzy natural language communication system. In *Proceedings of 1995 IEEE International Conference on Fuzzy Systems . The International Joint Conference of the Fourth IEEE International Conference on Fuzzy Systems and The Second International Fuzzy Engineering Symposium*, 1995.

[RN95] S. Russell and P. Norwig. *A modern approach to artificial intelligence*. Prentice Hall, Englewood Cliffs, NJ, 1995.

[STGV93] S. M. Sulzberger, N. N. Tschichold-Gürman, and S. J. Vestli. Fun: Optimization of fuzzy rule based systems using neural networks. In *Proc. IEEE Int. Conf. on Neural Networks 1993*, pages 312–316, San Francisco, March 1993.

[Sut84] R. Sutton. *Temporal Crdit Assignment in Reinforcement Learning*. PhD thesis, University of Massachusetts, 1984.

[Sut88] R. Sutton. Learning to predict by the method of temporal differences. *Machine Learning*, 1988.

[Tan97] S. Tano. Potential of fuzzy symbols and computational inference for multi modal user interface. In *IFSA World Congress*, Prague, 1997.

[TNS+97] S. Tano, Y. Namba, H. Sakao, T. Tomita, and H. Aoshima. Design concept based on real–virtual–intelligen user interface and its software architecture. In *HCI'97*, 1997.

[TOA96] S. Tano, T. Oyama, and T. Arnould. Deep combination of fuzzy inference and neural network in fuzzy inference. *Fuzzy Sets and Systems*, 82(2):151–60, 1996.

[Wei95] J. Weisbrod. Fuzzy control revisited — why is it working? In P. P. Wang, editor, *Advances in Fuzzy Theory and Technology, Vol. III*, pages 219–244. Bookwrights, Durham (NC), 1995.

[YJK97] T. Ye, H.-A. Jacobsen, and R. Katz. Mobile awareness in a wide area wireless network of info-stations. Technical report, University of California, Berkeley, May 1997.