

Using a Personalized, Adaptive and Cooperative MultiAgent System to Predict Protein Secondary Structure

Alessandro Orro¹, Massimiliano Saba¹,
Eloisa Vargiu¹, and Gianmaria Mancosu²

¹ University of Cagliari, Piazza d'Armi, I-09123, Cagliari, Italy
{orro,saba,vargiu}@diee.unica.it

<http://iasc.diee.unica.it/>

² Shardna Life Sciences, Piazza Deffenu 4, I-09121 Cagliari, Italy
{mancosu}@shardna.it

Abstract. In this paper, a generic architecture designed to support the implementation of applications agent-based, aimed at retrieving information among different Internet sources is presented. Information is filtered and organized according to personal interests explicitly stated by the user. User profiles are improved and refined throughout time by suitable adaptation techniques. The overall architecture has been called PACMAS, being a support for implementing Personalized, Adaptive, and Cooperative MultiAgent systems. PACMAS agents are autonomous and flexible, and could be personalized, adaptive and cooperative depending on the given application. The peculiarities of the architecture are highlighted by illustrating a relevant case study aimed at predicting protein secondary structures.

1 Introduction

Massive amounts of raw data are currently being generated by biologists while sequencing organisms. In fact, a common problem among biologist, physicians and computer scientists is how to share massive amounts of data, which are often unstructured, dynamic, heterogeneous and distributed over the Web. Most of these data must be analyzed gradually by resorting to various computer programs, and the problems that arise while retrieving them from various public web databases is typically not negligible. From a computer scientist perspective, the large number of heterogeneous and dynamically changing databases deemed relevant for a biologist have drawn the attention on suitable technologies able to tackle the complexity of the related problems. In our view, multiagent information gathering systems may significantly contribute in this research field.

Assuming that information sources are a primary operational context for software agents, the following categories can be identified focusing on their specific role: (i) “information agents”, able to access to information sources and to collect and manipulate such information [31], [26], (ii) “filter agents”, able to transform

information according to user preferences [25], [16], (iii) “task agents”, able to help users in various problem-solving activities and in exchanging information with other agents [5], [17], (iv) “interface agents”, in charge of interacting with the user such that she/he interacts with other agents throughout them [29], [24], and (v) “middle agents”, devised to establish communication among requesters and providers [13], [34]. Although this taxonomy is focused on a quite general perspective, alternative taxonomies could be defined focusing on different features. In particular, one may concentrate on capabilities rather than roles. For instance, a software agent can embed any subset of the following capabilities [30]: (a) autonomy, (b) reactivity, (c) proactiveness, (d) social ability, (e) flexibility, (f) personalization, (g) adaptation, (h) cooperation, (i) deliberative capability, and (j) mobility. Table 1 briefly depicts such capabilities and the corresponding focus.

Table 1. Capabilities of software agents

| Capability | Focus on ... |
|-------------------------|--|
| Autonomy | The ability of operating without the intervention of users. |
| Reactivity | The ability of reacting to a stimulus of the underlying environment according to a stimulus/response behaviour. The decision about which action should be undertaken may also depend on the current state of the software agent. |
| Proactiveness | The ability of exhibiting goal-directed behavior in order to satisfy a design objective. |
| Social ability | The ability of interacting with other agents according to the syntax and semantics of some selected communication language. |
| Flexibility | The ability of exhibiting reactivity, proactiveness, and social ability simultaneously [36]. |
| Personalization | The ability of personalizing the behavior to fulfill user’s interests and preferences. |
| Adaptation | The ability of adapting to the underlying environment by learning how to react and/or interact with it. |
| Cooperation | The ability of interacting with other agents in order to achieve a common goal. |
| Deliberative capability | The ability of reasoning about the world model and of engaging planning and negotiation, possibly in coordination with other agents. |
| Mobility | The ability of migrating from node to node in a local- or wide-area network. |

In this paper, we present a generic multiagent architecture designed to support the implementation of applications aimed at: (i) retrieving heterogeneous data spread among different Internet sources, (ii) filtering and organizing them according to personal interests explicitly stated by each user, and (iii) providing

adaptation techniques to improve and refine throughout time the profile of each selected user.

Each agent is autonomous and flexible, and may implement (one or more of) the following capabilities: personalization, adaptation, and cooperation. The overall architecture has been called PACMAS, being a support for implementing Personalized, Adaptive, and Cooperative MultiAgent Systems. The PACMAS architecture can easily give rise to specific systems (1) by identifying the characteristics of the dataflow that occurs from information sources to users (and vice versa), and (2) by customizing each involved agent according to its actual role and capabilities. In particular, in this paper, we illustrate how the PACMAS architecture can be used to solve the problem of predicting protein secondary structures.

In Section 2 relevant related work is briefly discussed. In Section 3 the PACMAS architecture is presented. In Section 4 all customization devised for explicitly dealing with secondary structure prediction are discussed. Section 5 draws conclusions and future work.

2 Related Work

In this section, some relevant related work is recalled, according to both an applicative and a technological perspective. The former is mainly focused on the task of secondary structure prediction, whereas the latter concerns the field of software agents, which the proposed system stems from.

2.1 Secondary Protein Structure Prediction

The problem of protein structure prediction is very complex, as the underlying process involves biological, chemical, and physical interactions. In this section we introduce principal aspects of the problem of the proteins prediction, mainly focusing on secondary structures.

Secondary structures are usually classified into alpha-helices, beta-sheets and random coils. Despite this simplification, the information about secondary structure often provides useful information for predicting protein functional sites. The accuracy of a prediction can be measured in different ways: The traditional per-residue prediction accuracy index Q_3 measures the percent of residues correctly predicted versus the overall number of residues, whereas parameters, Q_α , Q_β and Q_γ represent the percent of residues (related to a specific conformational status, i.e., alpha-helix, beta-sheet, and random coil, respectively) that have been correctly predicted. A variety of secondary structure methods have been proposed in the literature. Early prediction methods were based on a combination of statistical theory and empirical rules, applied to each amino acid of the protein to be predicted [11]. Artificial neural networks (ANNs) have been widely applied to this task [22] and represent the core of many successful secondary structure prediction methods, thanks to their ability of finding patterns without the need for predetermined models or known mechanisms.

The most significant innovation introduced in prediction systems was the exploitation of long-range and evolutionary information contained in multiple alignments. It is well known, in fact, that even a single variation in a sequence may dramatically compromise its functionality. To figure out which substitutions can possibly affect functionality, sequences that belong to the same family can be aligned, with the goal of highlighting regions that preserve a given functionality. PHD [33] is one of the first ANN-based methods that make use of evolutionary information to perform secondary structure prediction. In particular, PHD generates a profile using a BLASTP [1] search; then, the result is filtered using ClustalW [21].

Further improvement are obtained with both more accurate multiple alignment strategies and more powerful neural network structures. For instance, PSI-PRED [2] exploits the position-specific scoring matrix built during a preprocessing performed by PSI-BLAST [23]. This approach outperforms PHD thanks to the PSI-BLAST ability of detecting distant homologies. In a more recent work [6], [7], Recurrent ANNs (RANNs) are exploited to capture long-range interactions. The actual system that embodies such capabilities is SPRO [32].

2.2 Agents in Bioinformatics

A large amount over the past ten years has gone into developing algorithms (pattern matching, statistical, and/or heuristic/knowledge-based) able to deal with bioinformatics issues. Many of these are available to biologists in various implementations, and many are available over the web. Meta-sites combine many published algorithms and supply information about particular topics such as protein motifs. Multiagent systems have a lot to contribute to these efforts. Several features make a multiagent approach to this problem particularly attractive: information is available from many distinct locations; information content is heterogeneous; information content is constantly changing; biologists wish to both make their findings widely available, yet retain control over the data; new types of analysis and sources of data are appearing constantly.

In this section, we briefly introduce two multiagent systems that have been proposed in the literature.

BIOMAS [15] is a multiagent system for automated annotation and database storage of sequencing data for herpesviruses. It is based on DECAF [19], a multiagent system toolkit based on RETSINA [14] and TAEMS [12]. The resulting system eliminates tedious and hand analyses, makes the data and annotations available for other researchers (or agent systems), and provides a level of query processing beyond even some high-profile web sites. BIOMAS uses the distributed, and open nature of its multiagent solution to expand the system in several ways that will make it useful for biologists studying more organisms, and in different ways.

Hermes [27] is a layered middleware system to design and execute activity-based applications in distributed environments. In particular, Hermes provides an integrated environment where application domain experts can focus on designing activity workflow and ignore the topological structure of the distributed

environment. Hermes is structured as an agent-oriented system with a 3-layer architecture: run-time, system, and user. The layered middleware has been adopted in a bioinformatics domain, where mobile agents are used to support data collection and service discovery, and to simulate biological system through autonomous components interactions.

3 The PACMAS Architecture

PACMAS is a generic multiagent architecture, aimed at retrieving, filtering and reorganizing information according to users' interests. PACMAS agents can be personalized, adaptive, and cooperative, depending on their specific role.

The overall architecture (depicted in Figure 1) encompasses four main levels (i.e., information, filter, task, and interface), each being associated to a specific role. The communication between adjacent levels is achieved through suitable middle agents, which form a corresponding mid-span level.

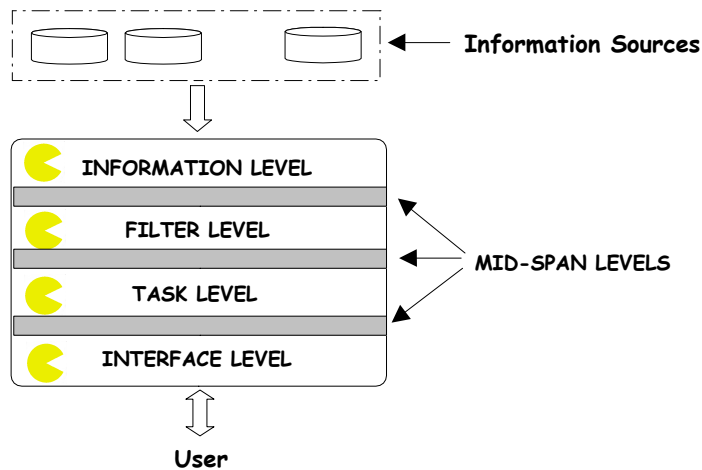


Fig. 1. The PACMAS Architecture.

Each level is populated by a society of agents, so that communication may occur both horizontally and vertically. The former kind of communication supports cooperation among agents belonging to a specific level, whereas the latter supports the flow of information and/or control between adjacent levels through suitable middle-agents.

At the information level, agents are entrusted with extracting data from the information sources. Each information agent is associated to one information source, playing the role of wrapper. Upon extraction, the information is then made available to the underlying filter level.

At the filter level, agents are aimed at selecting information deemed relevant to the users, and cooperate to prevent information from being overloaded and redundant. Two filtering strategies can be adopted: generic and personal. The former applies the same rules to all users; whereas the latter is customised for a specific user. Each strategy can be implemented through a pipeline of filters, since data undergo an incremental refinement process. The information filtered so far is then made available to the task level.

At the task level, agents arrange data according to users' personal needs and preferences. In a sense, they can be considered as the core of the architecture. In fact, they are devoted to achieve users' goals by cooperating together and adapting themselves to the changes of the underlying environment. In general, they can be combined together according to different connection modes, depending on the specific application.

At the interface level, a suitable interface agent is associated with each different user interface. In fact, a user can generally interact with an application through several interfaces and devices (e.g., pc, pda, mobile phones, etc.). Interface agents usually act individually without cooperation. On the other hand, they can be personalized to display only the information deemed relevant to a specific user. Moreover, in complex applications, they can adapt themselves to progressively improve their ability in supplying information to the user.

At the mid-span level, agents are aimed at establishing communication among requesters and providers. In the literature, several solutions have been proposed: e.g., blackboard agents, matchmaker or yellow page agents, and broker agents (see [13] for further details). In the PACMAS architecture, agents at the mid-span level can be implemented as matchmakers or brokers, depending on the specific application.

Each level of the architecture is composed by a population of agents that can be combined together in accordance with the following modes: pipeline, parallel, and composition (see Figure 2). In particular, (i) pipelines can be used to distribute information at different levels of abstraction, so that data can be increasingly refined and adapted to the user's needs, (ii) parallel connections can be used to model a cooperation among the involved components aimed at processing interlaced information, whereas (iii) compositions can be used for integrating different capabilities, so that the resulting behavior actually depends on the combination activity.

Keeping in mind that agents may be classified along several ideal and primary capabilities that they should embed, let us first recall the agent taxonomy proposed in [30]. In such taxonomy, three primary capabilities have been identified: autonomy, learning, and cooperation (see Figure 3-a). In our view agents are always autonomous and flexible, hence we deem that autonomy should not be explicitly listed in a diagram. On the contrary, we claim that personalization should be taken into account as a primary feature while depicting the characteristics of software agents. The resulting taxonomy is depicted in Figure 3-b, where personalization, adaptation, cooperation.

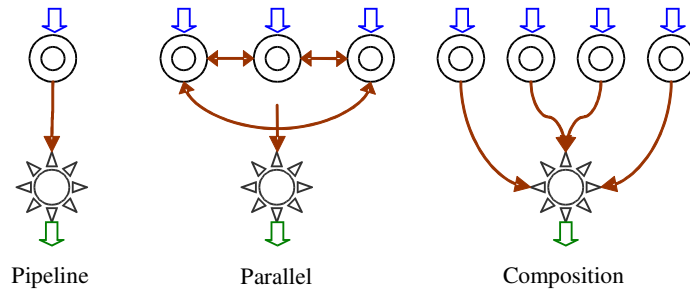


Fig. 2. Agents Connections.

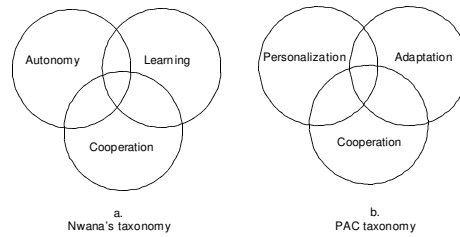


Fig. 3. Agents taxonomies.

As for personalization, an initial user profile is provided in form of a list of keywords, representing users' interests. The information about the user profile is stored by the agents belonging to the interface level. It is worth noting that, to exhibit personalization, filter and task agents may need information about the user profile. This flows up from the interface level to the other levels through the middle-span levels. In particular, agents belonging to mid-span levels (i.e., middle agents) take care of handling synchronization and avoiding potential inconsistencies. Moreover, the user behavior is tracked during the execution of the application to support explicit feedback, in order to improve her/his profile.

As for adaptation, a model centered on the concept of "mixtures of experts" has been employed. Each expert is implemented by an agent able to select relevant information according to an embedded string of feature-value pairs, features being selectable from an overall set of relevant features defined for the given application (the decision of adopting a subset of the available features has been taken for efficiency reasons, being conceptually equivalent to the one usually adopted in a typical GA-based environment [18], which handles also dont-care symbols).

As for cooperation, agents at the same level exchange messages and/or data to achieve common goals, according to the requests made by the user. Cooperation is implemented depending on the guidelines imposed by the composition modes defined above (i.e., pipeline, parallel, and composition). The most important form of cooperation concerns the "horizontal" control flow that occurs between peer agents. For instance, filter agents can interact in order to reduce

the information overload and redundancy, whereas task agents can work together to solve problems that require social interactions to be solved.

4 BIOPACMAS: PACMAS for Protein Secondary Structure Prediction

This section describes a case study to give the reader a flavor about how the proposed architecture can be easily customized for dealing with a specific application. The case study is concerned with the problem of predicting protein secondary structures. The resulting application has been called BIOPACMAS to put into evidence that the PACMAS architecture has been used in a typical bioinformatics problem.

A prototype of the system has been implemented using Jade [8] as underlying environment.

4.1 Agents for Predicting Protein Secondary Structures

Keeping in mind that the PACMAS architecture encompasses several levels, each one hosting a set of agents, in the following, we illustrate how each level supports the implementation of the proposed case study.

At the information level, agents play the role of wrappers, which –in our view– can be considered a particular kind of filters, devised to process information sources. Each wrapper is associated to one information source: in particular, in this specific application, (i) an agent wraps the selected training set (the TRAIN database), (ii) an agent wraps the test set (the R126 database), and (iii) an agent wraps a database containing information about the domain knowledge (the AAindex database). Datasets are briefly recalled in the Table 2.

In the current implementation, information agents are not personalized, not adaptive, and not cooperative (shortly \overline{PAC}). Personalization is not supported at this level since information agents are only devoted to wrap the datasets containing proteins. Adaptation is also not supported, since information sources are invariant for the system and are not user-dependent. Cooperation is also not supported by the information agents, since each agent retrieves information from different sources and each information source has a specific role in the chosen application.

At the filter level, agents embody encoding methods. Let us briefly recall that encoding methods play an important role in the prediction of protein secondary structures. In fact, they describe the chemical-physics properties of aminoacid deemed more interesting for the prediction. Several populations of filter agents have been implemented, each of them performing a different encoding technique: one-shot, substitution matrices, multiple alignment algorithms, and a technique that combines the specificity of multiple alignment with the generality of substitution matrices. Personalization is not supported by filter agents, as they always embody the same encoding methods for all users. Adaptation is also not supported, as encoding methods do not change during the application. Cooperation

Table 2. Information sources

| Dataset | Description |
|---------|---|
| TRAIN | It has been derived from a PDB selection obtained by removing short proteins (less than 30 aminoacids), and with a resolution of at least 2.5 Å. This dataset underwent an homology reduction, aimed at excluding sequences with more than 50% of similarity. The resulting training set consists of 1180 sequences, corresponding to 282,303 amino acids. |
| R126 | It has been derived from the historical Rost and Sander’s protein dataset (RS126) [33], and corresponds to a total of 23,363 amino acids (the overall number has slightly varied over the years, due to changes and corrections in the PDB). |
| AAindex | It contains information about hydrophobicity, dimension, charge and other features required for evaluating the given metrics. In the current application eight domain-specific metrics have been devised and implemented. A sample metrics is: Check whether hydrophobic amino acids occur in a window of predefined length according to a clear periodicity, whose underlying rationale is that sometimes hydrophobic amino acids are regularly distributed along alpha-helices. |

is supported by filter agents, as some implemented encoding methods bring together several algorithms (e.g., the encoding method that combines multiple alignment with substitution matrices).

At the task level, a population of task agents, which are the core of this case study, perform the protein secondary structure prediction. The “internals” of each task agent is based on the micro-architecture proposed for NXCS-Experts [4]. Being I and O an input and an output space, let us assume that an oracle A exists that, for each $x \in I$, maps x to a value $A(x) \in O$. A globally-scoped expert H is a function that approximates A on the whole input space, whereas a locally-scoped expert h is a partial function that approximates A on a subset of the input space. In its basic form, each NXCS expert E can be represented by a triple $\langle g, h, w \rangle$, where: (i) g is a function that selects inputs according to the value of some relevant features, (ii) h whose activation depends on $g(x)$, and (iii) w is a weighting function used to perform output combination. Hence, the output of E coincides with $h(x)$ for any input x that matches $g(x)$, otherwise it is not defined. In the case E contributes to the final prediction (together with other experts), its output is modulated by the value $w(x)$ of its weighting function, which represents the expert strength in the voting mechanism. This value may depend on several features, including the input x , the overall fitness of the corresponding expert, and the reliability of the prediction made by the embedded expert. Typically, the guard g of a generic NXCS classifier is implemented by an XCS-like classifier, able to match inputs according to a set of selected features deemed relevant for the given application, whereas the embedded classifier h consists of a feed forward ANN, trained and activated on the inputs acknowledged

by the corresponding guard. Task agents are not personalized, adaptive, and cooperative (shortly \overline{PAC}). Personalization is not required, since task agents exhibit the same behaviors for all the users. Adaptation is required, since each expert is suitably trained through a typical evolutionary behavior (as will be illustrate in the next Section). Cooperation is required, since they usually need other task agents to successfully achieve their goals.

At the interface level, agents are aimed at interacting with the user. In the current implementation, this kind of agents has not been developed. Nevertheless, we are investigating how to implement a flexible behavior at the user side. In particular, a suitable web interface is under study. We envision an interface personalized for each user, in which she/he can input a protein to be predicted, also being given the possibility of selecting the encoding technique applied. The resulting information agents will be personalized, adaptive, and not cooperative (shortly PAC). Personalization will be required in order to allow each user to customize the user interface. Adaptation will be required, as agents could adapt themselves to the changes that occur in the user preferences. Cooperation will not be required by the agents belonging to this architectural level.

As for the mid-span levels, the corresponding middle agents exhibit a different behavior depending on the mid-span level that they belong to. In particular, let us recall that, in the proposed architecture, there are three mid-span levels, one between information and filter levels (in the following, IF level), one between filter and task levels (in the following, FT level), and one between task and interface levels (in the following, TI level). Personalization and adaptation are not supported by middle agents, since they are only devoted to connect together agents belonging to adjacent levels. Cooperation is supported by agents belonging to the IF and the FT levels, since in the training phase they are used to verify the prediction.

Table 3. Agents Roles and Capabilities

| Agents | The ability of ... | Capabilities |
|--------------------|---|------------------|
| information agents | wrapping databases containing the test set, the training set, and the domain knowledge | \overline{PAC} |
| IF middle agents | verifying the prediction during the training phase in cooperation with the FT middle agents | \overline{PAC} |
| filter agents | encoding proteins | \overline{PAC} |
| FT middle agents | verifying the prediction during the training phase together with the IF middle agents | \overline{PAC} |
| task agents | predicting protein secondary structure | \overline{PAC} |
| TI middle agents | controlling the interactions among task and interface agents | \overline{PAC} |
| interface agents | interacting with the user | \overline{PAC} |

Table 3 summarizes the involved agents, together with their corresponding roles and capabilities.

4.2 Training Task Agents

The phase aimed at learning the underlying model is characterized by a typical, GA-like, evolutionary behavior.

In particular, the underlying model is learnt through an iterative process that takes care of creating, training, and deleting task agents. Once selected a set of suitable task agents, according to the constraints imposed by the embedded guards, each task agent receives from the filter agents, through the mid-span level, the protein to be predicted. After the prediction, through the cooperation of middle-agents, the secondary structure prediction has been compared with the one belonging to the information source, giving rise to a rewarding mechanism used for updating the fitness of the corresponding task agent. Moreover, through a cooperative mechanism, task agents exhibit an evolutionary behavior. In particular, task agents can be deleted according to a policy that allow to perform agent’s deletion in the case their fitness goes under a given threshold. Furthermore, agents could generate two offspring using the standard recombination and mutation operators.

4.3 Preliminary Experimental Results

To assess the performance of the system, and to facilitate comparison with other systems, preliminary experiments has been carried out. Since, the interface level is currently under study, to perform experiments an ad-hoc interface has been devised. As information sources we have adopted the R126 dataset for testing, widely used as a secondary structure prediction benchmark. R126 consists of 126 non-redundant proteins, having no pairs of proteins in the set with more than 25% of similarity over a length of more than 80 residues.

In particular, BIOPACMAS has been compared with NNSSP, PHD, DSC, PREDATOR, CONSENSUS, and SSPRO. The corresponding experimental results are summarized in Table 4. Also considering that the proposed application

Table 4. Experimental results, obtained from the R126 dataset

| System | Q3 |
|------------------|-------------|
| PREDATOR | 70.3 |
| DSC | 71.1 |
| NNSSP | 72.7 |
| PHD | 73.5 |
| CONSENSUS | 74.8 |
| BIOPACMAS | 74.9 |
| SSPRO | 76.6 |

is in a preliminary phase, results are encouraging, and we are investigating how to improve the results to perform better.

5 Conclusions and Future Work

In this paper a generic architecture designed to support the implementation of applications tailored for information retrieval tasks has been presented. The overall architecture has been called PACMAS, being a support for implementing Personalized, Adaptive, and Cooperative MultiAgent Systems. PACMAS agents are autonomous and flexible, and can be personalized, adaptive, and cooperative depending on the implemented application.

The peculiarities of PACMAS have been highlighted by depicting a relevant case study concerned with the prediction of secondary structures. Preliminary experimental results, performed on sequences taken from well-known protein databases are encouraging, and point to the validity of the approach.

As for the future work, we are investigating how to improve the behavior of information agents, to facilitate the task of implementing several encoding and multiple alignment algorithms (including PSI-BLAST).

References

1. Altschul, S.F., Gish, W., Miller, W., Myers, E.W., Lipman, D.J.: Basic local alignment search tool. *J. Mol. Biol.* (1990) 215:403–10
2. Altschul, S.F., Madden, T.L., Schaeffer, A.A., Zhang, J., Zhang, Z., Miller, W., Lipman, D.J.: Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Res.* (1997) 25:3389–3402
3. Armano, G., Murru, A. and Roli, F., "Stock Market Prediction by a Mixture of Genetic-Neural Experts", *Int. Journal of Pattern Recognition and Artificial Intelligence*, 16(5), (2002) 501–526.
4. Aroyo, L., De Bra, P.: Agent-oriented Architecture for Task-based Information Search System. *Proceedings of the Interdisciplinaire Conferentie Informatiewetenschap*, (1999) 94–98.
5. Baldi, P., Brunak, S., Frasconi, P., Soda, G., Pollastri, G.: Exploiting the Past and the Future in Protein Secondary Structure Prediction. *Bioinformatics.* (1999) 15:937–946.
6. Baldi, P., Brunak, S., Frasconi, P., Pollastri, G., Soda, G.: Bidirectional Dynamics for Protein Secondary Structure Prediction. In Sun, R., Giles, C.L. (eds.): *Sequence Learning: Paradigms, Algorithms, and Applications*. Springer-Verlag, (2000) 80–104.
7. Bellifemine, F., Poggi, A., and Rimassa, G.: Developing multi-agent systems with JADE Eventh International Workshop on Agent Theories, Architectures, and Languages (ATAL-2000), (2000).
8. Blundell, T.L., Johnson, M.S.: Catching a common fold. *Prot. Sci.* (1993) 2(6):877–883
9. Chotia, C.: One thousand families for the molecular biologist. *Nature* (1992) 357:543–544
10. Chou, P.Y., Fasman, U.D.: Prediction of protein conformation. *Biochem.* (1974) 13:211–215
11. Decker, K.S., and Lesser, V.R.: Quantitative modeling of complex computational task environments. *Proceedings of the Eleventh National Conference on Artificial Intelligence*, Washington, (1993) 217–224.

12. Decker, K., Sycara, K., and Williamson, M.: Middle-agents for the Internet. Proceedings of the 15th International Joint Conference on Artificial Intelligence (IJCAI'97), 578–583.
13. Decker, K.S., and Sycara, K.: Intelligent adaptive information agents. *Journal of Intelligent Information Systems*, 9(3) (1997) 239-260.
14. Decker, K., Khan, S., Schmidt, C., Situ, G., Makkena, R., and Michaud, D.: BioMAS: A Multi-Agent System for Genomic Annotation. *International Journal Cooperative Information Systems*, 11(3) (2002) 265–292.
15. Falk, A., and Josson, I.M.: PAWS: An agent for WWW-retrieval and filtering. Proceedings of Practical Application of Intelligent Agents and Multi-agents Technology (PAAM-96), (1996) 169–179.
16. Giampapa, J.A., Sycara, K., Fath, A., Steinfeld, A., and Siewiorek, D.: A Multi-Agent System for Automatically Resolving Network Interoperability Problems. Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems, (2004) 1462 – 1463.
17. Goldberg, D.E.: *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley (1989)
18. Graham, J., and Decker, K.S.: Towards a distributed, environment-centered agent framework. *Intelligent Agents VI*, N.R. Jennings and Y. Lesperance (Eds.), LNAI-1757, Springer Verlag (2000) 290-304.
19. Henikoff, S., Henikoff, J. G.: Amino acid substitution matrices from protein blocks. *Proc. Nat. Acad. Sci.* (1989), 10915–10919.
20. Higgins, D., Thompson, J., Gibson T., Thompson, J.D., Higgins, D.G., Gibson, T.J.: CLUSTAL W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice. *Nucleic Acids Res.* (1994) 22:4673–4680
21. Hirst, J.D., Sternberg, M.J.E.: Prediction of structural and functional features of protein and nucleic acid sequences by artificial neural networks. *Biochemistry*, 31 (1992) 7211-7218.
22. Jones, D.T.: Protein secondary structure prediction based on position-specific scoring matrices. *J. Mol. Biol.* (1999) 292:195–202.
23. Lieberman, H.: *Autonomous Interface Agents*. Proceedings of the ACM Conference on Computers and Human Interface (CHI-97), (1997) 67–74.
24. Lutz, E., Kleist-Retzow, H.V., Hoernig, K.: MAFIAan active mail-filter-agent for an intelligent document processing support. *ACM SIGOIS Bulletin*, 11(4) (1990) 16 – 32.
25. Maes, P.: Agents that Reduce Work and Information Overload. *Communications of the ACM*, 37(7) (1994) 31–40.
26. Corradini F., and Merelli, F.: Hermes: agent-based middleware for mobile computing. *Tutorial Book of 5th International School on Formal Methods for the Design of Computer, Communication and Software Systems: Mobile Computing*, LNCS Vol. 3465 (2005).
27. Merikoski, J.: On the submultiplicativity of norms of Hlder and Minkowski type. *Linear and Multilinear Algebra* 6 (1978) 51-54.
28. Metral, M.E.: *Design of a Generic Learning Interface Agent*. BSc Thesis, Department of Electrical Engineering and Computer Science, MIT (1993).
29. Nwana, H.: Software Agents: An Overview. *Knowledge Engineering Review*, 11(3) (1996) 205–244.
30. Papazoglou, M.P., Laufman, S.C., and Sellis, T.K.: An Organizational Framework for Cooperating Intelligent Information Systems. *Journal of Intelligent and Cooperative Information Systems*, 1(1) (1992) 169–202.

31. Pollastri, G., Przybylski, D., Rost, B., Baldi, P.: Improving the Prediction of Protein Secondary Structure in Three and Eight Classes Using Neural Networks and Profiles. *Proteins* (2002) 47:228–235
32. Rost, B., Sander, C.: Prediction of protein secondary structure at better than 70% accuracy. *J Mol Biol* (1993) 232:584-599
33. Sycara, K.: Multi-agent infrastructure, agent discovery , middle agents for Web services and interoperation. *Mutli-agents systems and applications*, Carbonell, J.C., and Siekmann, J. (Eds.) (2001) 17–49.
34. Altschul, S.F., Madden, T.L., Schffer1, A.A., Zhang,J., Zhang, Z., Miller, W., and Lipman, D.J.: Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nuclic acid Research*,1997 Vol.25 No.17 33893402
35. Wooldridge, M., and Jennings, N.R.: *Agent Theories, Architectures, and Languages: a Survey*. Wooldridge and Jennings (Eds.), *Intelligent Agents*, Berlin: Springer-Verlag, (1995) 1-22.