Towards Distributed Reasoning for Behavioral Optimization

Michael Cebulla

Technische Universität Berlin, Fakultät für Elektrotechnik und Informatik,
Institut für Softwaretechnik und Theoretische Informatik,
Franklinstr. 28/29, 10587 Berlin
mce@cs.tu-berlin.de

Abstract. We propose an architecture which supports the behavioral self-optimization of complex systems. In this architecture we bring together specification-based reasoning and the framework of ant colony optimization (ACO). By this we provide a foundation for distributed reasoning about different properties of the solution space represented by different viewpoint specifications. As a side-effect of reasoning we propagate the information about promising areas in the solution space to the current state. Consequently the system's decisions can be improved by considering the long term values of certain behavioral trajectories (given a certain situational horizon). We consider this feature to be a contribution to *autonomic computing.*

1 Introduction

The main target of our research consists in the definition of an architecture which supports the integration of reasoning and optimization thus enabling autonomic systems behavior. We take our starting point in the introduction of concepts for the knowledge-based fuzzy specification of various systemic aspects (extending our results from [1]). We show how it is possible to inexpensively check the conformance of these properties by traversing the solution space. These traversals of the solution space are performed by ant colonies. Areas of the solution space which are promising w.r.t. to a certain specification are marked with numerical information (frequently called *trail*). This information is propagated to the current state where it can be exploited for the optimization of behavior. Since there are multiple aspects of systems behavior which are examined by ant colonies different sorts of trail have to be evaluated in the process of decision making. As we will see this task is performed by an entity referred to as the *queen.*

In this paper we propose a hybrid architecture which integrates knowledge-based modeling [2], automata-based techniques of reasoning [3] with ant colony algorithms [4] in order to enable intelligent behavior of complex systems. In order to give specific support for robustness of reasoning and behavior we rely on fuzzy concepts for knowledge representation and reasoning.

After firstly giving a brief introduction to our usage of fuzzy description logics (in Section 2) we discuss a simplified application scenario (Section 3). The semantic and algorithmic aspects of reasoning are described in Section 4 and 5. The architectural integration is discussed in Section 6.

2 Fuzzy Description Logics

For the fuzzification of description logics fuzzy sets [5] are introduced into the semantics instead of the crisp sets used in the traditional semantics (cf. [2]). For more detailed discussion of these issues cf. e.g. [6, 7].

If C is a concept then $C^{\mathcal{I}}$ will be interpreted as the *membership degree function* of the fuzzy concept C w.r.t. $\mathcal{I}$. Thus if $d \in \Delta^{\mathcal{I}}$ is an object of the domain $\Delta^{\mathcal{I}}$ then $C^{\mathcal{I}}(d)$ gives us the degree of being the object d an element of the fuzzy concept C under the interpretation $\mathcal{I}$ [6]. For some selected constructors which were considered for description logics the interpretation function $\cdot^{\mathcal{I}}$ has to satisfy the following equations:

$$
\begin{aligned}
\top^{\mathcal{I}}(d) &= 1 \\
\bot^{\mathcal{I}}(d) &= 0 \\
(C \sqcap D)^{\mathcal{I}}(d) &= min(C^{\mathcal{I}}(d), D^{\mathcal{I}}(d)) \\
(C \sqcup D)^{\mathcal{I}}(d) &= max(C^{\mathcal{I}}(d), D^{\mathcal{I}}(d)) \\
(\neg C)^{\mathcal{I}}(d) &= 1 - C^{\mathcal{I}}(d) \\
(\exists R.C)^{\mathcal{I}}(d) &= sup_{d' \in \Delta^{\mathcal{I}}}\{min(R^{\mathcal{I}}(d,d'), C^{\mathcal{I}}(d')\} \\
(\exists T.C)^{\mathcal{I}}(d) &= sup_{d' \in \Delta^{\mathcal{I}}}\{min(T^{\mathcal{I}}(d,o), C^{\mathcal{I}}(o)\} \\
(qR.C)^{\mathcal{I}}(d) &= \{d | d \in \Delta^{\mathcal{I}}, |\{d' | R(d,d') > 0\}| \geq q\} \\
(mod_q R.C)^{\mathcal{I}}(d) &= \{d | d \in \Delta^{\mathcal{I}}, mod(|\{d' | R(d,d') > 0\}|) \geq q\} \\
(\langle q_1, \ldots, q_n \rangle R.C^{\mathcal{I}}(d) &= \{d | d \in \Delta^{\mathcal{I}}, \forall i \in \{1, \ldots, n\}, \#_i(R(d,d')) \geq q_i\}
\end{aligned}
$$

Remarks. In addition to roles we also support functional roles T which are needed for the integration of fuzzy concrete domains Δ_D (with $o \in \Delta_D$). We support a very simple style of quantifications allowing the use of positive rational numbers q or fuzzy modifiers *mod* (defined by piecewise linear membership functions). In addition we support a construct of tuple-valued cardinality which will be used to represent quantification concerning different aspects. As we will see such tuples contain global numerical information from different sources (commonly referred to as *trail*).

From Subsumption to Conformance. Deviating from common approaches based on description logics we do not focus on model-based reasoning about equivalence or subsumption. This is the reason why we do not rely on tableaux-based reasoning and thus do not have to face the resulting computational complexity (cf. [2]). In contrast we propose a syntax-directed approach for reasoning

about the conformance of specifications. Specifications concerning different aspects of systems behavior are compared with the specification of the solution space. As we will see this is done by colonies of artificial ants. The requirements related to conformance are represented by sets of constraints describing *morphisms*. This approach is heavily influenced by [8].

3 Fuzzy Specifications

In this section we describe a simple example which we use to illustrate some characteristics of our approach. We apply fuzzy terminologies to the description of *locations* in sensor networks (cf. [9]). We use this concept as a high-level abstraction since in many cases it does not make sense to address the individual components of a sensor network explicitly. Alternatively such systems provide the transparent access to data from interesting places (without having to mention individual components). Similar arguments and more details about sensor networks can be found in [10].

A terminological description of a location is shown in the following (simplified example):

$$\begin{aligned}
\mathsf{loc\text{-}wl\text{-}xyz} &\doteq 1.0\ \mathsf{contains.Sensor1} \sqcap 0.7\ \mathsf{contains.Sensor2} \sqcap \\
&\quad\ 0.6\ \mathsf{contains.Sensor3} \\
\mathsf{Sensor1} &\doteq \exists \mathsf{energy}.{}_{=.7} \sqcap \exists \mathsf{water\text{-}level}.{}_{=8.5} \\
\mathsf{Sensor2} &\doteq \exists \mathsf{energy}.{}_{=.8} \sqcap \exists \mathsf{water\text{-}level}.{}_{=9.1} \\
\mathsf{Sensor3} &\doteq \exists \mathsf{energy}.{}_{=.6} \sqcap \exists \mathsf{water\text{-}level}.{}_{=8.9}
\end{aligned}$$

As an example we introduce a terminological description of a simple location which contains three individual sensor components. The containment relation *contains* is quantified by a relevance value of the specific sensor component in the actual location. Note that this value is taken from the interval $[0, 1]_{\mathbb{Q}}$. For the sake of our example sensor components contain data describing the current energy level and data concerning the water level. Note that the relations *energy* and *water-level* have to be treated as functional roles containing objects which represent concrete domain values (cf. Section 2).

Scenario. For the sake of this presentation we assume that the system has to decide which location it prefers in order to retrieve information in a given situation. Thus the solution space in this scenario contains various behavioral alternatives which each corresponds to the choice of an individual location. Obviously the quality of a solution is determined by parameters like *relevance* and *energy-level.* In our approach ant colonies have the task to retrieve this information (related to these two different aspects) and to mark the trajectories which conform best to their viewpoint specifications.

Viewpoint Specifications. In order to gain interesting information about the current situation different viewpoint specifications are checked throughout the

solution space. These specifications may concern different aspects like availability of energy, like relevance or like flooding.

$$\begin{aligned} \mathsf{high\text{-}energy} &\doteq \mathsf{Most\ contains.energy(High)} \\ \mathsf{high\text{-}relevance} &\doteq \mathsf{Most\ contains.Sensor} \\ \mathsf{flooding} &\doteq \mathsf{Most\ contains.water\text{-}level(High)} \end{aligned}$$

Each of these three viewpoint specifications describe a criteria whose value is important for the systems decision making. Note that in such specifications we can use fuzzy terms like *most* and *high.* While *most* is defined in terms of fuzzy role quantification *high* is defined on concrete domain values of energy. Note that we are able to formulate less restrictive (and more robust) constraints using fuzzy concepts. In our framework such specifications are associated with colonies of ants which traverse the solution space (within a certain horizon) and check accessible paths for their conformance w.r.t. a certain specification.

4 Semantics

In the following we will extensively use the fact that we can consider specifications as tree-like term structures. For example both the systems specification as well as the viewpoint specifications can be represented in such a way (cf. Figure 1). We propose a syntax-directed approach for reasoning about *fuzzy conformance* which is based on the concurrent traversal of trees. Note that the promising paths in the solution space are marked with their value of conformance as a side-effect of reasoning.

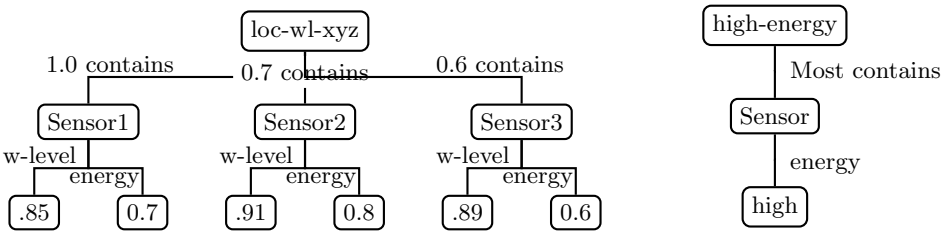

Fig. 1. Tree-like Representations of Specifications

Observations and Experiments. Intuitively, we say that two specifications are conform (to a certain degree) if they support similar observations and experiments. Observations and experiments on fuzzy specifications are described by fuzzy transition systems (FTS).

signature FTS
obs: $X \times T \to [0,1]_{\mathbb{Q}}$
next: $X \times T \times X \to [0,1]_{\mathbb{Q}}$

In the signature of the transition system we can see that the observation and transitions functions are multi-valued. In fact we are interested in observations (whose content is represented by a fuzzy terminological concept) which hold to a certain degree. On the other hand we want to know which costs are coupled with a certain state transition (triggered by a terminological role). Thus the application of the transition function *next* yields a result which corresponds to the (fuzzy quantification) of the role term which is used as parameter (frequently referred to as *costs*).

Fuzzy Morphisms. In order to get a foundation for the notion of fuzzy conformance we rely on the notion of fuzzy morphism (cf. [11]).

Definition 1 (Fuzzy Morphism on FTS). *A fuzzy morphism f between two fuzzy transitions systems A_1 and A_2 is given by: $obs_2(T) \leq obs_1(f(T))$, and $next_2(X_2, T, X_2') \leq next_1(f(X_2), f(T), f(X_2'))$,*

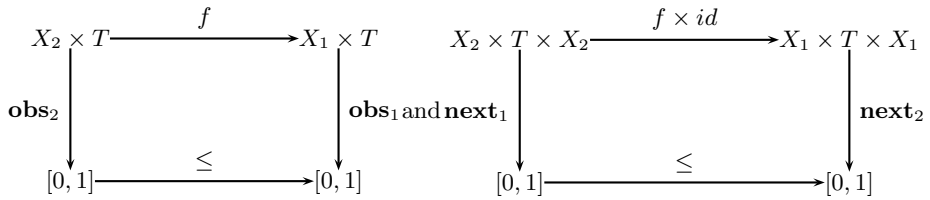

Fig. 2. Fuzzy Homomorphism on Signatures

Note that we sometimes assume (in Figure 2 and elsewhere) that both specifications use the same terminology T. We interpret positive differences between observations as conformance values (e.g. $obs_2(T) - obs_1(T)$). As a side-effect of reasoning the solution space is annotated with these values. In the following we heavily rely on an operational interpretation of these morphisms. For more details concerning this approach cf. [12].

5 Reasoning

In this section we heavily use the argument that morphisms can be mapped on bisimulations between automata (cf. [13]). We construct an automaton whose purpose is to recognize such bisimulations between two fuzzy tree automata. Such an automaton is called tree tuple automaton [3]. For the description of transition rules we use concepts from membrane computing (cf. [14]).

Membrane-Based Tuple Tree Automaton. Intuitively the automaton checks whether the systems description D supports the same experiments and observations as a certain viewpoint specification S. If there are multiple experiments necessary copies of the automaton are created for every experiment.

We exploit the characteristics of membrane computing and its computational properties for the creation and handling of multiple copies of tree automata. In our proposal the automaton for the recognition of bisimulation is implemented as a rewriting P-system (on structured objects) (cf. [14]). In this formalism we can use embedded membranes in order to articulate the tree-like structure of a term. In addition membranes contain multisets of terminal symbols or other information (e.g. trail). Consequently we can define transition rules for the processing of expressions using the paradigm of multiset rewriting. While (atomic) concept expressions are represented by molecules role expressions (and their cardinalites) are encoded in membrane labels. In order to deal with fuzzy expressions we support rational cardinalities of multisets. For more details on our usage of P-systems cf. [12].

Definition 2 (P-system for Bisimulation). *A P-system for bisimulation is defined as a tuple* $P_{BS} = \langle T, \mu, w1, \ldots, w_m, R_1, \ldots R_m \rangle$, *where* T *is a terminology and* $\mu = [_0[_D]_D[_S]_S]_0$ *is the initial configuration (of the membrane structure).*

While the membrane $[_D]_D$ represents the systems description $[_S]_S$ contains a viewpoint specification. Intuitively in each step of the simulation the necessary observations and experiments are drawn from the viewpoint specification (by obs_S and $next_S$) and then applied to the systems description (by obs_D and $next_D$). For simplicity we assume that both automata (and both specifications) use the same terminology T.

The behavior of the automaton is defined by the following rewriting rules:

1. Atomic Concepts $A \in T$:

   $$[[_D X]_D[_S A[_Q \alpha]_Q]_S] \rightarrow [[_D X, \mathbf{t_i^d}]_D[_S[_Q \alpha]_Q[_S], \text{when } \mathbf{obs}_D(A) \geq \mathbf{obs}_S(A)$$

   Note that a molecule representing the conformance value d is introduced into the systems specification per side-effect. Intuitively the molecule t (for *trail*) is related to the ith viewpoint specification and represents a rational conformance value d.
2. Basic Roles $Q \in T$:

   $$[[_D X]_D[_S[_Q \alpha]_Q]_S] \rightarrow [[_D \mathbf{ann\text{-}lbl}(\mathbf{t_i^d})]_D[_S \alpha]_S], \quad \text{when } \mathbf{next}_D(Q) \geq \mathbf{next}_S(Q)$$

   These rules describe a test concerning the costs of transitions. Thus by a call to $next_S$ we can retrieve the transitions which have to be supported by the system's description D in order to stay conform to the specification. Again trail is left when conformance is detected. We use the operation *ann-lbl* to introduce the molecule representing the trail information into the tuple-valued quantification of the membrane-label (not explicitly shown in the rules).

3. For the treatment of the constructors from fuzzy description logics the following rules can be used (selected examples). Note that the operators have to be treated according to fuzzy semantics (cf. Section 2).

$$\begin{aligned}[[_D X]_D[_S A \sqcap B]_S] &\rightarrow [[\sqcap[[_D X]_D[_S A]_S][[_D X]_D[_S B]_S]]\sqcap] \\ [[_D X]_D[_S \exists Q.C]_S] &\rightarrow [\sqcup_{i\in\{1,\ldots,k\}}[_i[\sqcap[_D \mathbf{next}^i]_D[_S Q]_S \\ &\quad [_D \mathbf{next}^i]_D[_S C]_S]\sqcap]_i],\end{aligned}$$

   In the rules for the processing of quantified role expressions (involving atomic roles) the terms labeled with i have to be created for each $i \in [1, k_T]$ where k_T is the branching factor of the tree. Intuitively this corresponds to the creation of a copy of the automaton for each role filler of Q. $next^i$ are procedures for the explicit navigation in trees (retrieving the ith subtree of the current node). As we will see the fuzzy semantics of the operators controls the propagation of trail.
4. The treatment of numeric quantification is similar to the treatment of quantification described above. For each role filler a copy of the automaton is created.

$$\begin{aligned}[[_D X]_D[_S q Q.C]_S] &\rightarrow [\sqcup_{i\in\{1,\ldots,k\}}[_i[\sqcap[_D \mathbf{next}^i]_D[_S Q]_S \\ &\quad [_D \mathbf{next}^i]_D[_S C]_S]\sqcap]_i] \text{ when } \#_D Q \geq q \\ [[_D X]_D[_S mod_q Q.C]_S] &\rightarrow [\sqcup_{i\in\{1,\ldots,k\}}[_i[\sqcap[_D \mathbf{next}^i]_D[_S Q]_S \\ &\quad [_D \mathbf{next}^i]_D[_S C]_S]\sqcap]_i] \text{ when } mod(\#_D Q) \geq q\end{aligned}$$

Example. While comparing the specifications from our example the situation in Figure 3 is constituted at some point. On the left hand side of the transformation we see that experiments concerning *most contains* are prescribed by the specification. Since there are three such experiments possible on the system's description three copies of the automaton are created. Then the experiments concerning *contains* are initiated concurrently (as shown at the right hand side of the rule). For this mechanism we rely on the fact that *membrane division* is a common and inexpensive operation in membrane computing (cf. [14]).

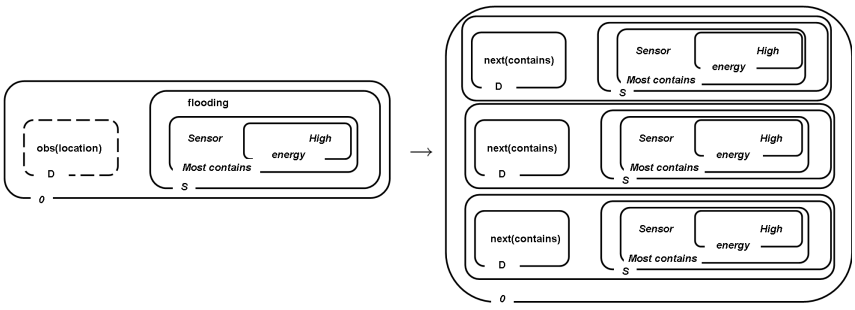

Fig. 3. Example: Behavior of Tuple Tree Automaton

Trail Propagation. In the final phase of reasoning about the conformance of a viewpoint specification the information represented by the trail is propagated. For this sake the numerical information is treated like a *synthesized*-attribute in attributed grammars (cf. [15]). Intuitively the information about promising locations in the solution space is *propagated* to the the current state thus increasing the quality of the system's decisions. Again we exploit some characteristics of membrane computing for the reactive propagation of this information.

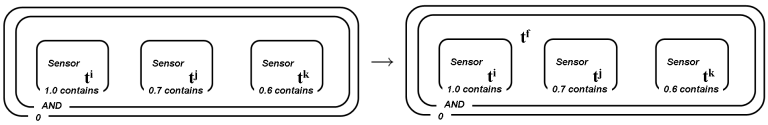

Fig. 4. Example: Trail Propagation

In the example transition in Figure 4 we can see how trail information is propagated bottom-up through the membrane hierarchy. The rules for the propagation are determined by the semantics of the operators (in this case $\sqcap$). According to Zadeh's logic the value of f is defined as the minimum of i, j, k.

6 Distributed Reasoning

We propose an architecture for the integration of high-level modeling, automated reasoning and ant-colony optimization. Our goal is to propagate information about valuable places in the solution space in order to support decision making in the current state. Generally we propose that systems rely on solutions found by ant colonies during normal situations while they have to depend on default values for their decision in highly dynamic situations (represented by so-called myopic heuristic information, cf. [4]).

Integration. In order to collect additional information about the long-term values of behavioral alternatives ant colonies iteratively traverse the accessible trajectories. A given ant colony is embodied by a fuzzy tuple tree automaton as discussed in the previous sections. Note that tree automata are copied into multiple instances while examining different branches of a tree thus keeping the correspondence to the metaphor of ant colonies. Since we use fuzzy morphisms in our framework we assume that ants leave large amounts of *trail* on traces in the systems state space where the conformance to their specification is strong. The global choice between these solutions is performed by the queen. Note that the marked traces in the systems behavior can be considered as trajectories in an n-dimensionary hypercube. Unfortunately we cannot discuss the issue of *trail evaporation* [4] due to space limitations.

Example. In a simplified example we consider the situation that the system (represented by the queen) has to choose between two locations. Obviously at first sight it is not possible to decide which group of sensors to use (since heuristic information about the costs – as represented by attribute *def* – are equal concerning both alternatives).

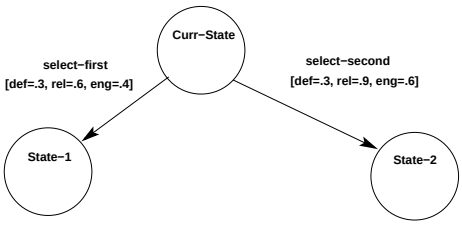

Fig. 5. Example Situation

We assume that the ants examinations (concerning relevance and energy) resulted in the values shown in Figure 5. Consequently it is the queen which has to infer the best alternative from the available information. For this sake we define the queen as a tree tuple automaton which performs a simple version of multi-criteria optimization: the queen always selects the behavioral alternative which is marked with the greatest global value (depending on environmental parameters). Note that for simplicity we assume a non-fuzzy decision behavior of the queen always resulting in the choice of exactly one behavioral alternative.

$$\left[_{\langle\alpha,\beta\rangle}\left[_{\langle s_0,s_1,s_2\rangle A_1}\right]_{\langle s_0,s_1,s_2\rangle A_1}\left[_{\langle t_0,t_1,t_2\rangle A_2}\right]_{\langle t_0,t_1,t_2\rangle A_2}\right]_{\langle\alpha,\beta\rangle} \rightarrow \left[_{A_1}\ldots\right]_{A_1}$$
$$\text{when } max_{1,2}(s_i) > max_{1,2}(t_i) \text{ and } \alpha > \beta$$
$$\left[_{\langle\alpha,\beta\rangle}\left[_{\langle s_0,s_1,s_2\rangle A_1}\right]_{\langle s_0,s_1,s_2\rangle A_1}\left[_{\langle t_0,t_1,t_2\rangle A_2}\right]_{\langle t_0,t_1,t_2\rangle A_2}\right]_{\langle\alpha,\beta\rangle} \rightarrow \left[_{A_1}\ldots\right]_{A_1}$$
$$\text{when } s_0 > t_0 \text{ and } \beta > \alpha$$

For the sake of example we give two rules which describe the queen's decision making in our scenario. We encode the information about trail into membrane labels. The first rule describes how behavioral alternative A_1 is selected on the basis of trail information (represented by t_1, t_2, s_1, s_2). The global parameter α denotes the weight of trail information (during normal environmental conditions) while β contains a high value when the situation is highly dynamic. This case is described by the second rule where also A_1 is selected but this time on the basis of heuristic information (contained in s_0, t_0). Remember that trail information tends to be useless in the presence of environmental changes.

7 Conclusion

Our motivation in this paper is directed towards an architectural integration of high-level knowledge-based modeling, of automated reasoning and of techniques for local optimization in order to support context-aware behavior in

autonomic systems. For this reason we propose to use colonies of artificial ants for the exploration of the solution space of complex systems. In this framework we support distributed and robust reasoning about high-level specifications. Knowledge about the systems properties is diffusing through the solution space thus supporting decentral and distributed forms of decision-making and control.

References

1. Peter Pepper, Michael Cebulla, Klaus Didrich, and Wolfgang Grieskamp. From program languages to software languages. *The Journal of Systems and Software*, 60, 2002.
2. Franz Baader and Werner Nutt. Basic description logics. In Franz Baader, Diego Calvanese, Deborah McGuinness, Daniele Nardi, and Peter Patel-Schneider, editors, *The Description Logic Handbook*. Cambridge University Press, Cambridge, U.K., 2003.
3. H. Comon, M. Dauchet, R. Gilleron, F. Jacquemard, D. Lugiez, S. Tison, and M. Tommasi. Tree automata techniques and applications. Available on: http://www.grappa.univ-lille3.fr/tata, 1997. release October, 1rst 2002.
4. Marco Dorigo and Thomas Stützle. The ant colony optimization metaheuristic: Algorithms, applications, and advances. In Fred Glover and Gary A. Kochenberger, editors, *Handbook of Metaheuristics*. Kluwer Acad. Publ.: Boston et.al, 2003.
5. Lofti A. Zadeh. Fuzzy sets. *Information and Control*, 8(3):338–353, 1965.
6. Umberto Straccia. Reasoning with fuzzy description logic. *Journal of Artificial Intelligence Research*, 14:137–166, 2001.
7. Steffen Hölldobler, Hans-Peter Störr, and Tran Dinh Khang. The fuzzy description logic ALC_{FH} with hedge algebras as concept modifiers. *International Journal of Advanced Computational Intelligence and Intelligent Informatics*, 7(3):294–305, 2003.
8. Dusko Pavlovic, Peter Pepper, and Douglas Smith. Specification engineering. *to appear*, 2006.
9. David Culler, Deborah Estrin, and Mani Srivastava. Overview of sensor networks. *IEEE Computer*, 37:41–49, August 2000.
10. S. Madden, R. Szewczyk, M. Franklin, and D. Culler. Supporting aggregate queries over ad-hoc wireless sensor networks. In *IEEE Workshop on Mobile Computing Systems and Applications*, 2002.
11. George J. Klir and Bo Yuan. *Fuzzy Sets and Fuzzy Logic. Theory and Applications.* Prentice Hall, Upper Saddle River, N.J., 1995.
12. Michael Cebulla. Reasoning about knowledge and context awareness. In *Proc. of FLAIRS'06*, 2006.
13. Jan Rutten. Automata and coinduction (an exercise in coalgebra). Technical report, Centrum voor Wiskunde en Informatica, Amsterdam, 1998.
14. Gheorghe Păun. *Membrane Computing. An Introduction.* Springer, Berlin, 2002.
15. Donald E. Knuth. Semantics of context-free languages. *Theory of Computing Systems*, 2(2):127–145, June 1968.