Automatic Planning for Autonomous Spacecrafts Constellations

Eric Bornschlegl
European Space Agency
ESTEC, P.O Box 299
2200 AG Noordwijk ZH-NL
ebornsch@estec.esa.nl

Christophe Guettier
Axlog Ingénierie
19-21, rue du 8 mai 1945
94110 Arcueil, F
christophe.guettier@axlog.fr

Jean-Clair Poncet
Axlog Ingénierie
19-21, rue du 8 mai 1945
94110 Arcueil, F
jean-clair.poncet@axlog.fr

Abstract

Autonomous agents are a challenging concept for future unmanned operations. Previous space and aeronautic missions highlight the lack of on-board intelligence to increase the decision making capability and to efficiently react to unexpected situation or events. Many agent-based system approaches exhibit good properties, as demonstrated in flight during the Deep Space1 mission. This paper presents a multi-agent system framework with on-board planning, based on constraint solving models, applicable to a constellation of satellites

Introduction

A constellation of satellites (formation flying) will require more cost effective operation and significant increase in performances, pointing accuracy, reactivity and orbital control. It requires a global management of operations and interconnections, as well as an optimization of the utilization of communication links either between the ground and space segments or between satellites. This requires to deal at the same time with several related system functions layered in different level of granularity. For example, a global configuration and operation modes management of the constellation has to be compliant with the diverse local modes switching or state transitions, involving cooperative or reflexive policies.

The use of distributed functions over a spacecraft constellation (such as planetary observations) makes mission planning more complex. Today's technology enables us to safely process on-board each satellite, a certain number of functions traditionally performed on the ground (such as attitude & orbit, state transition and configurations, planning & scheduling). By joining distributed systems theory and emerging artificial intelligence results, recent researches carried out in space (Ber98; MP98) and aeronautics (Yav94) domains emphasized the benefit of using Multi Agent Systems (MAS) (HJ96). In practical examples, MAS appears as a constructive approach that can tackle underlying difficulties like uncertainty, completeness (HJ96) and many applicative requirements (Yav94) such as:

- *Affordability*: Modern systems can make use of components off-the-shelf which requires a better modularity of software and hardware.
- *Performances*: Autonomous systems have to execute missions under real-time constraints requiring powerful observations, rapidly analyzing input data and selecting the right alternative.
- *Flexibility*: Systems are expected to adapt their behavior as required by changes in environment. This may require the system to dynamically reconfigure. It must then have the ability to build a good representation of its environment.
- *Availability*: Systems must be able to handle and complete requests while reacting under real-time constraints to environment modifications.
- *Survivability*: Systems that evolve in an hostile environment must deal with possible failures. Thus they must use fault tolerant mechanisms and possess dynamic reconfiguration capabilities.
- *Safety*: Agent deployment must guarantee a safe behavior in respect to both other agents and human beings.

MAS allows the design of global intelligent behaviors modeled through symbolic and logical representations (HJ96; WJ94). For instance, it is possible to formally specify how several agents can collaborate to perform a global "goal oriented" mission or to perform dedicated specific actions using a limited set of both discrete and continuous resources (payloads, energies, processing elements, ...). The use of different forms of logic, symbolic reasoning and arithmetic inference provide various ways to model different problems. However benefit of these results is limited due to highly combinatorial explosion and complex collaborative behaviors. Thus decision making and operational capability is restricted if the implementation is only based on simple heuristics.

Those combinatorial problems have been widely investigated in the Constraint Programming (CP) community. Stemming from logic programming, integer and mathematical programming, Constraint Logic Programming (CLP) languages are recognized as powerful tools to

cope with difficult and large combinatorial problems (DHS90; CL96; GH99). Underlying models of the approaches presented below could be expressed using a CLP language based on the simplex algorithm (Col87) and other results from integer and linear programming (such as branch and cut (PR91)) but the solving method would be restricted to a linear existential part of the constraint language. In addition, memory space explosion prohibits the use of $0-1$ modeling techniques. In contrary, CLP offers a higher compositionality to express and solve complex NP-Hard problems by separating declarative and operational semantics defined on finite parts of R and N with all their classical operators. On the declarative side, those constraint languages enable us to model several related complex problems with a slight loss of completeness. On the operational side, dedicated search operators can be specified in order to design global search strategies (CL96). Furthermore, addressing task scheduling and multiple resource allocation problems, recent results emphasized the efficiency of those approaches (DHS90; VSD95; Jou95; VSD95; CL96; GH99). In this paper, we consider a spacecraft constellation as a multi-agent system. In order to maximize the final system efficiency, we investigate how planning under operational constraints can be modeled and related to the MAS behavior through a constraint programming approach. This necessitates considering at the same time several NP-hard sub-problems such as motion planning, collaboration constraints, observation and orbit allocation. The CLP properties offer a more compositional, generic and flexible way to separate model's expression from search strategy.

Background

Formation Flying Mission, (for example, a deep space or earth observation mission), can be characterized by a set of global goals to achieve with the delivery of mission data products. A goal is for example interferometry or stereovision from in-flight combined observations with multiple sensing resources (such as cameras distributed on several satellite platforms , ...).

Space mission

We will consider a spacecraft constellation which for an operator submits mission goals with timing constraints. The constellation is composed of a fixed number of spacecraft (four for instance). Spacecraft fly in formation, very close from each other (a few tens of meters). Each spacecraft is represented by a platform with Inter-Satellite Link (ISL) capability, and by a payload (synthetic aperture radar, spectrometer, ...).
In order to insure flexibility, availability and performances requirements, the constellation can be divided into sub formations that will simultaneously achieve distinct observations from different orbits. Spare satellites might also be needed. Thus separating and/or combining observation maneuvers within the formation are part of the global mission plan. Consequently, flexibility of the platforms depends on the MAS global & local planning capabilities that control their roles (operation and configuration in hard real-time, from predicted or unpredicted discrete events).

A Multi-agent approach

Global operation planning over the whole formation must always be safe for the mission, while local satellite control insures survivability. In our approach, when the local control task fails or an unexpected event occurs, an emergency procedure is triggered to secure the formation and a new plan is generated. It appears to be a suitable technique to dynamically combine and allocate various discrete event control locations levels and criticalities within a global planning.

Spacecraft constellation as a category of MAS We assume that the constellation can constitute a single formation flying or be split into several ones. The constellation is represented as a MAS, composed of a set of collaborative cognitive agents, each one associated with a spacecraft. At any time, only the leader spacecraft of a formation can generate a global mission plan while the others are considered as reactive agents. Thus one formation can be seen as a reactive centralized architecture. However, a set of formations has to satisfy collaboration constraints defined below, so that when the constellation is divided into several formations, the associated MAS behaves as an hybrid deliberative architecture (HJ96).

Action planning We assume that mission autonomy is mainly provided by the on-board system. We are considering in this section only actions and underlying constraints affected by the mission goals. Actions under consideration are discrete actuations for orbit manoeuvres, observation, separation and/or reconstitution of formations.

Collaborative Policy At the low level, to achieve an increased survivability, the collaborative behavior of the MAS framework includes emergency actions that place the constellation in a secured state. If this topic is not under the scope of this paper, it necessitates the computation of a new plan as the previous plan may be no longer feasible after an emergency has occured. At a higher level, a long term collaborative behavior must also be considered by the plan. It specifies how plans from separated formations are compliant. At a minimum, the collaborative policy states that every spacecraft of a given formation obey the leader, and that formations have to be assembled in a given orbit, in a specified time window.

Coordination constraints Chaining orbit changes and observations requires scheduling on-board activities in a compliant way with the formation plan. Thus, we have a local and a global level of coordination. In our model, we consider reconfiguration tasks and action execution supported by operational modes. Coordination constraints are solved at the planning level.

Model-based planning

In our approach, we consider the planning function as solving a set of constraint-based combinatorial problems expressed as MAS or operational models like collaborative policy, altitude or observations.

CP and model-based computing A Constraint Logic Programming language can be viewed as an extension of Logic Programming where unification is replaced with constraint satisfaction. Logical predicates can be constraints interpreted in a mathematical algebra (JL87; VSD95) which is over the finite domains in our context : $\{\mathcal{P}(R), +, -, *, >, =\}$. Such a language enables predicates composition through logical operators and quantifiers. This leads to a more understandable, compositional and modular problem representation.

Model based computing is a practical way to take advantage of those CLP properties (Jou95; AG98). The modeling method (Jou95; GH99) extracts and adapts invariant of each problem by a recursive decomposition until a tractable expression can be formalized. Those formal expressions involve mathematical variables and constrained predicates that represent respectively the problem combinatory and its invariants. So doing, the resulting global problem is represented with different related and heterogeneous models. Relations between models are conjunctions of constraints that maintain the global consistency by propagating local solutions between models. The distinction between the problem formulation and the solving facilities allows to find solutions for various goals automatically.

Model based computing & constellation planning In the following we decompose the planning problem in different sub-problems. Thus, the spacecraft position and velocity are decomposed through *path planning*, *altitude* and the *timing* models. Therefore by combining solutions with classical orbitography, it is possible to extract attitude, positioning and speed along the plan. In fact combination of numerical computation and symbolic reasoning is essential in autonomous systems (WN96). For instance, we still have to prepocess orbitography functions before solving the planning problem. Moreover, the planning also satisfies *collaborative* and *coordination* models.

Path planning using graphs

Each orbit is interpolated by a set of navigation points. Transfers are allowed between the different orbits in specific conditions defined on transition points. The very simplified orbit interpolation described above allows us to project orbits in an oriented 1-graph $G = [X, U]$, where the set of vertices X represents navigation points and the set of edges U represents orbit's arcs.

Event based navigation model

We define a time horizon $h \in N$ that bounds the maximum length of a plan in event time. We duplicate the graph according to h and event time to model trajectory evolution through time defining a set of graphs $\{G_0, G_1, ..., G_h\}$, each one connected to its immediate immediate neighbor in the event time by adding to each vertex an edge to its corresponding vertex at the next event time occurrence. The costs of additional edges are fixed and equal zero.

At each vertex x of the graph G, we associate the set $\omega(x)$ of connected edges and a function d: $\forall x \in X, d_x() : \{0, ..., h\} \rightarrow Q$ where $d_x(j)$ represents the operational time on vertex x in the event time j. If $d_x(j) = 0$ the formation is not on the vertex x at the event time j, otherwise the formation is on the vertex x at operational time $d_x(j)$ in event time j.
Identicaly each edge u of the graph G is associated to a function e_u: $\forall u \in U, e_u() : \{0, ..., h\} \rightarrow \{0, 1\}$ where each $e_u(j)$ represents the use of the edge u in the event time j. If $e_u(j) = 1$ the path uses the edge u in the event time j, otherwise the edge u is not used in the event time j. This model is solved by instantiating $d_x(j)$ and $e_u(j)$ for each time event j. The graph G^h is extended using additional edges and vertex in order to define a circuit between two points. This allows us to express paths constraint along the graph, as shown in the following models.

Path consistency model

On each vertex of the graph, the first Kirshoff law must be verified, stating that the sum of the incoming flow must equal the sum of the outgoing flow (constraint 1). A flow of capacity less or equal to 1 insures a path consistency along the graph edges. However, this well-known model (GM95) in Operation Research has to be extended to fit the event time representation. For each vertex $x \in X$ we respectively define sets $\omega^+(x)$, $\omega^-(x)$ of incoming and outgoing edges.

$$\forall x \in X, \sum_{u \in \omega^+(x)} \Big[\sum_{j=0}^{h-1} e_u(j)\Big] = \sum_{u \in \omega^-(x)} \Big[\sum_{j=0}^{h-1} e_u(j)\Big] \quad (1)$$

This constraint defines the global flow over the extended graph G^h and also states the connectivity of the path over the graph.

Transition Constraints

Now, the operational times have to be consistent with the flows by defining a supplementary constraint (2) for transition over edges in U:

$$\forall\, u = \overrightarrow{xx'} \in U,\ \forall j \in \{0, ..., h-1\},$$
$$e_u(j) = (d_x(j) \neq 0) \wedge (d_{x'}(j+1) \neq 0) \quad (2)$$

This constraint states that an edge $\overrightarrow{xx'}$ can belong to the path in the event time j if and only if the path goes through the vertex x at event time j and through the

vertex x' at event time $j+1$. Finally, constraint (3) specifies that obviously, a route cannot use more than one edge for each time event:

$$\forall j \in \{0,...,h-1\},\ \sum_{u \in U} e_u(j) \leq 1 \qquad (3)$$

Altitude model

For each time event j and each spacecraft of the constellation we associate a discrete altitude slot as a global resource using a vector $h(j) \in \{0,1\}^n$ of $0-1$ variables. The constant n is the number of possible slots. At the time event j, a spacecraft will be located at the altitude slot k if the k^{th} component is set to $h_k(j) = 1$. Thus, we impose $\forall j,\ \sum_{k=1}^{n} h_k(j) = 1$ to insure the spacecraft belongs to a unique altitude slot at any time event. Each slot represents an interval which corresponds to a minimal perigee and a maximal apogee such that two orbits belonging to two distinct slots cannot overlap . The model also gives the travelling duration $\rho_u(j) \in Q$ required to cross an edge u according to the initial and final altitudes at the time event j. Solving the altitude model necessitates finding the appropriated slot for each time event while satisfying the constraints defined hereafter. First, when the spacecraft does not change its altitude, the constraint (4) expresses the travel duration across an edge thanks to a linear function. Second, when the spacecraft altitude changes, the constraint (5) approximates the travel duration across an edge using a linear combination of both starting and destination durations.

$$\begin{aligned} &\forall j \in \{0,...,h-1\} \ / \ h(j) = h(j+1) \Rightarrow \\ &\exists \overrightarrow{xx'} \in U / d_x(j) > 0 \ \wedge \ d_{x'}(j+1) > 0 \\ &\wedge \ \rho_{\overrightarrow{xx'}}(j) = h(j) \times \theta_{\overrightarrow{xx'}} \end{aligned} \qquad (4)$$

$$\begin{aligned} &\forall j \in \{0,...,h-1\} \ / \ h(j) \neq h(j+1) \Rightarrow \\ &\exists \overrightarrow{xx'} \in U \ / \ d_x(j) > 0 \ \wedge \ d_{x'}(j+1) > 0 \\ &\wedge \ \rho_{\overrightarrow{xx'}}(j) = \lambda_{\overrightarrow{xx'}} h(j+1) \times \theta_{\overrightarrow{xx'}} + \lambda'_{\overrightarrow{xx'}} h(j) \times \theta_{\overrightarrow{xx'}} \end{aligned} \qquad (5)$$

In both constraints, the constant vector $\theta \in Q^n$ represents the time duration required to cross the edge for each possible altitude. The resulting time duration is given by the variable $\rho(j)$. The two constant scalars $\lambda, \lambda' \in Q \times Q$ allow to approximate the time duration for changing orbits. Those constants can be preprocessed using traditional orbitography. This model illustrates the principles of modeling a set of elements (edges between different orbital locations) by taking advantage from problem's regularities and symetries instead of dealing with explicit set of elements. Nevertheless additional edges or more complex constraints can be defined to model non-regular orbital behaviors.

Timing and Observability constraints

For a given event time j, the value of $d_x(j)$ depends on the operational time $d_{x'}(j-1)$ on the preceding vertex x' in the path and the necessary time to go from x' to x. Then, if $u = \overrightarrow{x'x} \in \omega^+(x)$ is the edge that go from x' to x in G, we can state that if this edge belongs to the path, then the operational time $d_x(j)$ is defined as $d_x(j) = \rho_{\overrightarrow{x'x}} + d_{x'}(t-1)$, otherwise $d_x(j) = 0$. By extending the preceding relation to the whole edges in $\omega^+(x)$, we define $d_x(j)$ using a maximum on $(\rho_u + d_{x'}(j-1)) \times e_u(j-1)$ in the preeceding event time for all $u \in \omega^+(x)$.

For an observation zone, we have to specify that the path must include the travel of this zone in a specific operational time window. Moreover, this zone may be observed from several different orbits (i.e. several vertices of the graph G), and possibly with discrete sequences of operational time windows. In the following, a vertex x of G corresponds to a possible observation of a location in the operational time interval $[t_{min}, t_{max}]$ if it belongs to a triplet $\delta = (x, t_{min}, t_{max})$. A discrete observation window of a specific location from various observation points can then be represented as a set Δ of δ involving the same vertices x: $\Delta = \{\delta = (x, t_{min}, t_{max}) \mid x \in X,\ (t_{min}, t_{max}) \in Q^2\}$

According to constraint (6), a necessary observation of a zone Δ requires that the path contains at least one travel onto vertex x / (x, t_{min}, t_{max}):

$$\begin{aligned} &\forall \Delta,\ \exists \delta = (x, t_{min}, t_{max}) \in \Delta, \\ &\exists j \in \{0,...,h\} \ / \ (d_x(j) \geq t_{min}) \wedge (d_x(j) \leq t_{max}) \end{aligned} \qquad (6)$$

Collaboration model

To maximize the use of the flying formation, according to the availability and performance requirements, the model allows us to divide the formation into several sub-formations . We assume that a plan is composed with a set of several separation steps between the constellation and formations and one assembling step at the end of the mission. The set of spacecrafts is denoted $\mathcal{S}$, while a formation is denoted as $\mathcal{F}$. A leader spacecraft of the constellation is denoted $S \in \mathcal{S}$. A spacecraft $s \in \mathcal{S}$ is separated (resp. assembled) at the event time j^s_{sep} (resp. j_{ass}). In the constraints exposed hereafter we extend our notation to introduce quantification on spacecrafts.

Separating and assembling spacecrafts

We consider a single separation date for each spacecraft along a plan horizon. Before separation, the spacecraft is controlled by a leader and thus follows the same route at the same altitude according to constraint(7):

$$\begin{aligned} &\forall s \in S, \forall\ j \in [0,\ldots,h] \ / \ j \leq j^s_{sep}\ \exists x \in X \ / \ d^S_x(j) > 0 \\ &\Rightarrow \ d^s_x(j) > 0 \ \wedge \ h^s(j) = h^S(j) \end{aligned} \qquad (7)$$

In order to maximize safety between spacecrafts differents routes always corresponds to different altitude

slots. Thus we relate collaborative model to altitude to fulfill anti-colliding requirements:

$$\forall s \in \mathcal{S}, \forall\, j \in\ [0, \ldots, h],\ j > j^s_{sep} \Rightarrow h^s(j) \neq h^s(j) \quad (8)$$

Solving the collaboration problem requires us to instantiate the variable j^s_{sep} for each spacecraft. At least, with constraint (9), all the formations have to join in orbital rendez-vous defined by the variable j_{ass}:

$$\forall s \in \mathcal{F}, \forall s' \in \mathcal{F}',\ h^s(j_{ass}) = h^{s'}(j_{ass})$$

$$\wedge\ \forall\ x \in X,\ d^s_x(j_{ass}) = d^{s'}_x(j_{ass}) \quad (9)$$

Consistent behavior inside a formation

The collaborative behavior requires us to solve constraints inside the same formation. After a separation, a spacecraft of a formation remains at the same altitude slot, this leading to constraint (10):

$$\forall s \in S, \forall\, j \in\ [0, \ldots, h],\ j > j^s_{sep} \Rightarrow h^s(j) = h^s(j+1) \quad (10)$$

Moreover, if two spacecrafts belong to the same formation, they have the same separation date as well as the same route and altitude after a separation.

Coordination model

The planning function involves a coordination model for a given agent or for a whole formation. For each vertex of the graph and for each time event, a spacecraft is in a given operational mode. If the modes are different between two vertices of the same edge, then a reconfiguration task is required. The operational mode of a spacecraft $s \in \mathcal{S}$, at the time event $j \in [0, \ldots, h]$ for a given vertex $x \in X$ is denoted $M^s_x(j)$.

Local coordination

Along the plan, some of the modes are imposed by the vertex type, others have to be solved by the planner. In any case, the time taken to travel across the edge must be greater than the time taken to locally reconfigure a given spacecraft (constraint 11):

$$\forall \overrightarrow{xx'} \in U\ /\ e_{\overrightarrow{xx'}}(j) = 1\ \wedge\ M_{x'}(j+1) \neq M_x(j)$$

$$\Rightarrow\ d_{x'}(j+1) - d_x(j) \geq C(M_x(j), M_{x'}(j+1)) \quad (11)$$

The predicate $C(M, M')$ corresponds to the time taken to reconfigure the spacecraft between mode M and mode M'. It is statically computed and remains in the agent knowledge.

Global coordination

As separating and assembling actions are executed commonly we state that all the operational modes have to be equal to a specific mode during the separation. An analog constraint fit for the assembling action. Using this approach it is possible to represent complex configuration and compatibility problems. Even those are not within the scope of this paper.

Solving the planning problem

On the modeling baseline, many kinds of goals can be investigated. It is possible to search for a plan that satisfies all the model constraints, to complete an existing partial plan or to optimize an objective function corresponding to an applicable trade-off in several ways (for example insuring availability by maximizing the number of spare spacecraft or minimizing the overall completion time).

Automatic Solving Using CLP Language

To solve the goal, a global control strategy has to be designed. We use Concurrent Constraint operators over Finite Domains (CC(FD)) (VSD95) as well as domain heuristics. To insure the consistency between model solutions, and to reinforce the concurrency between the solving processes, CC(FD) offers the powerful control operators *Ask* & *Tell*. The satisfaction operator *Tell* states a constraint to the solver and the entailment operator *Ask* checks if a constraint is already satisfied. At the search process level, the *Tell* operator is used to exchange partial solutions between models through relations. When two partial solutions are not consistent, the system generates a backtrack event. The *Ask* operator is used for the synchronization of the global search. Finally using those operators, global solving strategies including domain heuristics are designed to control the composite global search over models. An experiment has been developed using the *Claire* language (CL96) and the Finite Domain *Eclair* library (GS99) based on CC(FD) operators. Activated from a simulation framework (running on a Ultra Sparc Workstation), this preliminary implementation can handle up to 6 spacecrafts.

Conclusion

The pertinence of using Constraint Model Based Programming for specifying and solving the complex problem of a Multi-Agent plan has been shown. We give a better alternative to heuristic-based behavior of traditional multi-agent planners and open a new way to tackle complex cooperative behaviors. Moreover, this work highlights the feasibility of the approach for solving spacecrafts formation plans, by taking advantage of a more complete specification and by allowing the concurrent use of different models. However, models have to be validated according to their level of granularity and their quality of representation. Continuation of this work should focus on search strategies, satisfying properties like anytime and real-time.

References

[AG98]L.Alenius, V.Gupta *Modeling an AERcam: a case study in modeling with concurrent constraint languages*, Proceedings of the CP'97 Workshop, Modeling and computation in the concurent constraint languages, Pisa 1998.

[Ber98]D.E.Bernard & al., *Design of the Remote Agent Experiment for Spacecraft Autonomy*, Proceedings of the 1998 IEEE Aerospace conference, Aspen CO., 1998.

[Col87]A.Colmerauer, *Opening the Prolog III universe*, , Bytes, August 1987.

[CL96]Y.Caseau, F.Laburthe, *Cumulative Scheduling with Task Intervals*, In Proc. of the Joint Int. Conference and Symposium on Logic Programming, M. Maher ed., the MIT Press 1996.

[DHS90]M.Dincbas, P.Van Hentenryck, H. Simonis, *Solving Large Combinatorial Problems in Logic Programming*, Journal of Logic Programming, Vol. 8, p.75-93, 1990.

[GH99]C.Guettier, J.F.Hermant, *A Constraint Based Model for High Performance Real Time Computing*, In Proc. of the Int. Conf. on Parallel and Distributed Computing Systems, Florida, 1999.

[GM95]M.Gondran, M.Minoux, *Graphes et algorithmes*, ed. Eyrolles, 1995.

[GS99]S.de Givry, P.Saveant, J.Jourdan, J.Mattioli *Eclair Reference Manual*, Tech. report TSI-99-905, Thomson-CSF Corporate Research Laboratory, Corbeville, France, 1999.

[HJ96]J.M.P.O'Hare, N.R.Jennings, *Foundations of Distributed Artificial Intelligence*, Sixth-Generation Computer Technology series, 1996.

[JL87]J.Jaffar, J-L Lassez, *Constraint Logic Programming*, In Proc. of the 14^{th} ACM Symposium on Principles of Programming Languages, Munich, January 1987.

[Jou95]J.Jourdan, *Concurrence et coopération de modèles multiples dans les langages de contraintes CLP et CC* PhD thesis, Université Denis Diderot, Paris VII, 1995.

[MP98]N.Muscettola, P.Pandurang Nayak, B.Pell, B.C.Williams *Remote Agent: To boldly go where no AI system has gone before*, NASA Ames Research Center, 1998.

[PR91]M.Padberg and G.Rinaldi, *A branch-and-cut algorithm for the resolution of large-scale symmetric traveling salesman problems*, SIAM Review, Vol. 33, No. 1, pp 90-100, March 1991.

[VSD95]P.Van Hentenryck, V.Saraswat, Y.Deville, *Design, Implementation and Evaluation of the Constraint Language CC(FD)*, In Constraint Programming: Basics and Trends, A. Podelski Ed., pp. 68-90, 1995.

[WN96]Brian C.Williams, P.Pandurang Nayak, Immobile Robots: AI in the New Millenium, *AI Magazine, 1996.*

[WJ94]M.Woodridge, N.R.Jennings, *Intelligent Agents: Theory and Practice*, Knowledge Engeneering Review, October 1994.

[Yav94]A.Yavnai. *Distributed Decentralized Architecture for Autonomous Cooperative Operation of Multiple Agent System*, in Proc. of the IEEE 1994 Symp. on Autonomous Underwater Vehicle Technology.