Intelligent ALMM System - implementation assumptions for its Knowledge Base

Ewa Dudek-Dyduch[1], *Zbigniew* Gomolka[2*], *Boguslaw* Twarog[2] and *Ewa* Zeslawska[3]

[1]AGH University of Science and Technology Cracow, Department of Biomedical Engineering and Automation, Krakow, Poland
[2]University of Rzeszow, Faculty of Mathematics and Natural Sciences Department of Computer Engineering, ul. Pigonia 1, 35-959 Rzeszow, Poland
[3]University of Information Technology and Management in Rzeszow, Faculty of Applied Informatics, Department of Applied Information, ul. Sucharskiego 2, 35-225 Rzeszow, Poland

Abstract. The paper introduces the concept of implementation assumptions about the Knowledge Base (KB) system cooperating with intelligent information system for the discrete optimization of problem solving, named Intelligent ALMM System. This system utilizes a modeling paradigm named Algebraic Logical Meta Model of Multistage Decision Processes (ALMM) and its theory, both developed by Dudek-Dyduch E. The system solves combinatorial and discrete optimization problems including NP-hard problems with possible user assistance. The models of problems are stored in a Problem Model Library. In this paper the idea of KB for the storage of the properties of problems is presented. The concept of the KB on problems presented in previous works has been extended by introducing an additional module pertaining to the properties of a problems library. A discussion was presented in the context of the selection of tools that enable the construction of such a library as well as its architecture. In the adopted strategy of storing the properties of problems, the interface for exchanging information is compatible with the library of problems using polymorphic and component properties of object-oriented programming. Considerations are explained by means of a sample UML diagram and interface prototypes.

1 Introduction

The article concerns the Intelligent System ALMM designed for solving combinatorial problems and problems of discreet optimization. This system is based on the paradigm of the Algebraic Logic Meta Model (ALMM) presented in chapter 2. The concept of the Intelligent ALMM system was proposed by Dudek-Dyduch E. in [1]. The main purpose of the ALMM System is not only to solve the problems of discrete optimization, but also to help the user in choosing the right method and algorithm for solving them. This intelligent approach is based on the use of specific properties of solved problems and through the links they have with appropriate methods or algorithms. This is accomplished via the knowledge

* Corresponding author: zgomolka@ur.edu.pl

base consisting of two parts: Library of Problem Models and Module of Problem Properties. The investigations on the system takes place simultaneously in two areas: theoretical and conceptual concerning the way the system works, its structure and the assumptions of the implementation project. Assumptions and fundamentals concerning the Library of Problem Models were presented in [2 - 3]. This article concerns the basic assumptions about the implementation of the project Module of Problem Properties and presents the development of links between the Library of Problem Model and the Module of Problem Properties.

2 ALMM modelling paradigm

At this point the paradigm of creating formal models of problems of discrete optimization based on the Algebraic Logical Meta-Model of multi-stage decision-making processes (ALMM) will be presented. [3] The idea of ALMM paradigm was proposed and developed by Dudek-Dyduch E. [4, 5], [6 - 7] and mentioned in many papers among others [8]. It has a wide variety of applications. ALMM includes formal models of two types of multistage decision processes: dynamic and common. The model of multistage dynamic decision process (MDDP) takes into account the fact that all restrictions may be time-dependent, while for common process (cMDP) these constraints are static. Let us recall the main concept of the Algebraic Logical Meta Model of the multistage dynamic decision process. The concept of the so-called 'generalized state' is used here, defined as a pair: a state and a time instant.

Definition. "Multistage dynamic decision process is a process that is specified by the sextuple $MDDP=\left(U,S,s_0,f,S_N,S_G\right)$ where U is a set of decisions, $S=X\times T$ is a set named a set of generalized states, X is a set of proper states, $T\subset\Re^+\cup\{0\}$ is a subset of non-negative real numbers representing the time instants, $f:U\times S\to S$ is a partial function called a transition function, (it does not have to be determined for all elements of the set $U\times S$), $s_0=\left(x_0,t_0\right)$, $S_N\subset S$, $S_G\subset S$, are respectively: an initial generalized state, a set of not admissible generalized states, and a set of goal generalized states, i.e. the states in which we want the process to take place at the end. Subsets S_G and S_N are disjoint i.e. $S_G\cap S_N=\varnothing$.

The transition function is defined by means of two functions, $f=\left(f_x,f_t\right)$ where $f_x=U\times X\times T\to X$ determines the next state, $f_t:U\times X\times T\to T$ determines the next time instant. It is assumed that the difference $\Delta t=f_t\left(u,x,t\right)-t$ has a value that is both finite and positive.

The transition function f is defined as partial. This allows various restrictions on "reasonable" decisions in different states too be taken into account by means of the so-called sets of possible decisions $U_p\left(s\right)$ defined as: $U_p\left(s\right)=\left\{u\in U:\left(u,s\right)\in Domf\right\}$. An important feature of this modeling paradigm is that decisions can be generally complex decisions. Therefore in the most general case, sets U and X may be presented as a Cartesian product $U=U^1\times U^2\times\ldots\times U^m$, $X=X^1\times X^2\times\ldots\times X^n$, i.e. $u=(u^1,u^2,...,u^m)$, $x=(x^1,x^2,...,x^n)$. In particular, u^i, $i=1,2,\ldots,m$ represent separate decisions that must or may be taken at the same time or at the same stage (for cMDP) and relate to particular objects. There are no limitations imposed on types of the sets; in particular they do not have

to be numerical. Thus values of particular co-ordinates of a state or a decision may be names of elements (symbols) as well as some objects (e.g. finite set, sequence etc.).

We obtain the definition of the common multistage decision process (cMDP) from the definition of MDDP by reducing the set S to the set X and the transfer function f to the function f_x Because all the elements specifying MDDP, i.e. $U, S, s_0, f, S_N, S_G, U_p(s)$ can be defined using both algebraic and logic relations, hence the name "algebraic logic". Based on the ALMM, the formal models of problems, the so-called AL models, can be created for an extensive class of optimization problems (and their instances), for which the solution can be presented as a sequence (or set) of decisions. All limitations of a given problem regarding the admissible solution are represented by elements of the appropriate multistage decision process marked as **P**. The optimization problem is defined as a pair **(P, Q)** where **Q** is a criterion. An optimal (or at least admissible only) sequence of decisions and the corresponding trajectory is sought. In the articles [4 -6, 9 - 12] that lay the groundwork for the basic ALMM theory, Dudek-Dyduch E. has presented AL models for discrete manufacturing process control, for logistics problems, knapsack problem, and a model for a complex real-life problem encompassing both manufacturing and logistics. Other models, including the disturbances in production processes, are presented in [13]. In addition, the ALMM paradigm allows for the definition of a number of properties of problems relevant to their final solution. Based on the ALMM paradigm, new methods for solving discrete optimization problems have been created too and described in [3, 14 - 17]. An overview of the new methods is given in [6].

ALMM provides a structured way of recording knowledge of the goal and all relevant constraints that exist within the problems modeled. In line with this idea, all information is split into pre-defined basic components (elements) with appropriate links, defined both through algebraic and logical formulae.

3 Concept of ALMM Intelligent System

The concept of an intelligent system based on ALMM technology comes from Dudek-Dyduch E. and was firstly presented in [1]. The basic goals of this system are as follows:

- determining admissible or optimal (possibly good enough) solutions for combinatorial and discrete optimization problems,
- support the user with an appropriate choose of the method or algorithm which gives a solution.

Basic knowledge about the problem is passed on with the help of the AL problem model. Solving problem instance relies on the appropriate generation and/or modification of one trajectory or appropriate subsets of the trajectories. In work [18] the concept of ALMM Solver being a software tool aimed at solving discrete optimization problems using constructive algorithms has been presented. As part of the further development of ALMM Solver, Dudek-Dyduch E. and Korzonek S. presented in [2 - 3] a way of creating software representations of AL models and the component construction of a new crucial module, namely the Library of Problem Models significantly expanding the original concept of the ALMM Solver. The next step was a completely revolutionized concept of the Intelligent ALMM System [1]. Fig. 1. shows a schematic structure of its main modules.

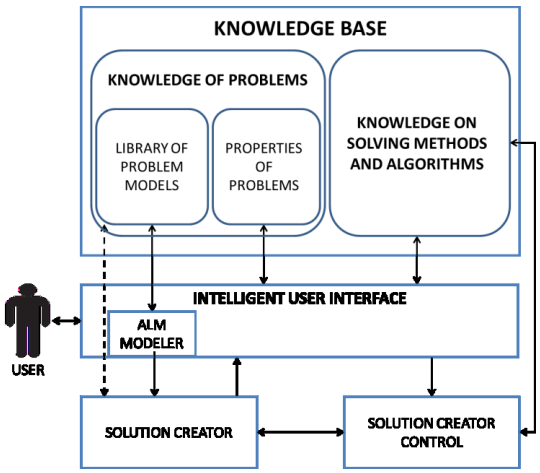

Fig. 1. The structure of Intelligent ALMM System [1]

The basis of the system is the information about the problem being solved in the form of an AL problem model and its instance as well as its software representation model. Both can be entered directly by the user, downloaded from the Library of Problem Models (PML) or created from the components of this library. Thus knowledge about the problem and its instance is forwarded from PML or ALM Modeler to the main module, named the Solution Creator, that carries out the computations.

The Knowledge Base consists of two main mutually joined components: "Knowledge on Solving Methods and Algorithms" and "Knowledge of Problems". The first part contains basic methods and algorithms written for the system using ALMM technology. The method (an algorithm) chosen by the user controls the calculation of the module Solution Creator via the Solution Creator Control.

The second part of the Knowledge Base consists of two further parts: Library of Problem Models and Properties of Problems. The idea of the component construction of the Library of Problems results from the structure of AL, models of problems and their mutual dependencies, which have been already described in [1], [3].

The intelligence presented in the [1] ALMM System consists in the implementation of two functions: helping a user create new AL models from components found in PML as well as support a user in determining the appropriate method and algorithm for problem solving. For both these functions, it is necessary to define a number of problem properties and link them with methods and algorithms. In works [8, 10, 19] a series of definitions of the properties of discrete optimization problems expressed using ALMM terminology have been given. In particular, they refer to the elements defining the process P and the most frequently used criteria. A number of trajectory properties were also given. In [1] it was shown how relevant properties are exploited to select methods such as branch & bound, (or pruning) dynamic programming, A* algorithms, and learning methods based on ALMM technology [20 - 25].

4 Assumptions for implementing the Problem Properties Knowledge Base

Knowledge Base is responsible for collecting data about solved problems and their properties. Due to the programming environment assumed at the initial stage of the project and programming language to be used for implementation of the IT model, three different

models of databases were considered that could be used to construct the Knowledge Base (KB) module:

- databases working in the object oriented technology,
- databases operating in the mixed relational-object technology,
- knowledge bases using the properties of individual objects and semantic relationships between them.

4.1 Accessible tools and programming strategies

In accordance with the adopted Microsoft C# programming environment for the System under design, the solution to the task of organizing the Problem Library and their properties will be possible using a MS SQL Server and the Language INtergrated Query (LINQ) module. LINQ is part of the .NET Framework technology that allows us to manage object collections in a similar way to the SQL query language (see Fig. 2). Thanks to this, full control of data types and their conversions in individual mechanisms mediating the downloading of data is maintained. Data can come from the database (LINQ to SQL) and ordinary objects (LINQ to Objects). The knowledge representation is the basic concept for the various types of decision-making based processes and becomes the fundamental problem that has not been fully resolved. In knowledge engineering, the representation of knowledge is treated as a way of presenting in formal language the whole range of knowledge which is usually required for intelligent system behavior. The main aspects of knowledge representation includes the syntax, as the structure of representation (language), semantics, or by the meaning of represented knowledge (interpretation) and inference, i.e. the process by which knowledge is used to derive conclusions [26 - 28].

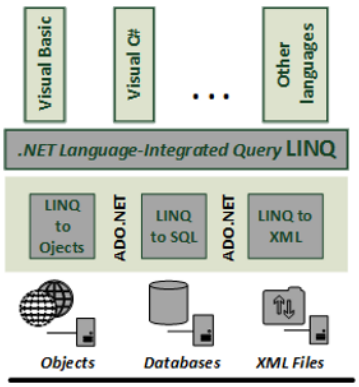

Fig. 2. Architecture of Language INtegrated Query proposed by authors

4.2 Knowledge base about problems in the ALMM concept

The intelligent system interface is currently prepared for well-known users who know the names of most problems and have the knowledge of ALMM terminology. There are three classes of users:

- oriented - who know the names of the classic problems of discrete optimization well and the algorithms to solve them,
- semi-oriented possessing partial knowledge,
- un-oriented who do not have knowledge about problems, but they want to use an intelligent system to solve their own problems, e.g. searching for the shortest path, the existence of mutual connections in the set of objects.

The knowledge base is constructed in such a way that all three groups can use the Intelligent ALMM System. The Knowledge of Problems consists of the Library of Problem Models and Properties of Problems. Due to the fact that the problem library will be implemented using the C# language, it was decided that the MS SQL Server data-base and LINQ database as well as the Object Relational Database will be used.

Knowledge Base about problems will store knowledge about:

- different properties of individual problems with the separation of this knowledge into process properties and criteria ones,
- rules combining properties of problems with the possibilities of using selected methods and algorithms to solve problems.

A characteristic feature of the created knowledge base is the fact that, once saved, properties will not be modified. However, it is expected that new properties might be added, which will be modified with relative infrequency. The Intelligent ALMM System created will be a multi-access system and work in a client-server standard. The organization of the knowledge base will depend significantly on the type of queries generated by system users and the recording range of additional properties. The organization of properties storing from the point of view of problems property implementation is divided into the following properties (Fig. 3):

- universal (suitable for many problems, whether or not encountered by them),
- private (closely related to the problem under consideration).

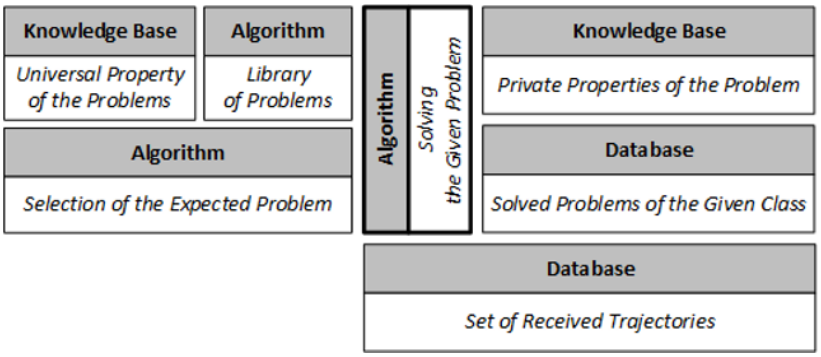

Fig. 3. The organization of Properties of Problems in Knowledge Base

Therefore, its universal properties will be stored in the knowledge base about properties: Knowledge Properties of Problems.

4.3 Software representation of properties knowledge base AL Model's

The implemented Problem Properties Library is a set of functions and data structures with a set of rules for combining elements of a given set into a single functional module. The C# object platform used allows for the organization of data and methods that correlate their processing in a communicative way in management. The proposed library is characterized by high flexibility and many possibilities of adapting to a diverse class of problems. It will be possible to expand the properties library of problems using the step-by-step method, including simple rules for adding new com-ponents. The proposed library provides a set of component properties for various problems solved in the Intelligent ALMM System. Any application or module cooperating with the Library will cooperate in the client server technology.

In the article [2, 3] the IT project was proposed for AL problem model designated as an IT model problem. Therefore, the methodology of the module's implementation must easily

link properties to individual problems and to the relevant elements of their AL model. The problem property model is implemented as a class named cProblemProperties.

In order to present the structure of the problems in the reality which are described by data, processes, data structures and programs that construct the system structure, a logical diagram of classes in UML was proposed. Fig. 4 presents a diagram of the classes and dependencies between the property interfaces of individual AL model elements that will be responsible for communication between the modules of the Library of Problem Models and Properties of Problems: iProperStateSet, iControlDecisionSet, iNonAdmissibleStateSet, iGoalStateStatement, iPossibleDecisionSet, iInitialState, iTransferFunctionALConditions.

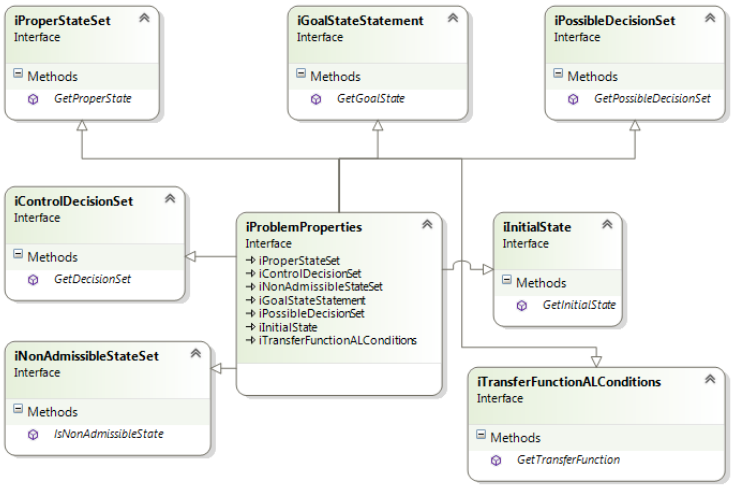

Fig. 4. Class diagram of Problem Properties

The Algebraic Logical Meta Model defines the individual elements of AL problem models and allows to define their detailed properties. The following property types should be defined in the *cProblemProperties* class: properties related to decision structure, state structure, initial state, transition function, set of decisions possible in given states, set of not admissible states, set of goal states and properties of problem data structures. The class interface modeling the problem property should provide methods related to the above elements:

- *cStateSet GetProperState(cProblem ProcessProperties)* – representation of the properties of a set of proper states;
- *cStateSet GetDecisionSet(cProblem ProcessProperties*) – representation of the properties of a set of control decisions or control signals;
- *cStateSet IsNonAdmissibleState(cProblem ProcessProperties)* – representation of the property of a set of generalized nonadmissible states;
- *cGoalState GetGoalState(cProblem ProcessProperties)* – representation of the properties of the set of generalized target states;
- *cStateSet GetPossibleDecisionSet(cProblem ProcessProperties)* – representation of the properties of a set of proper states;
- *cStateSet GetInitialState(cProblem ProcessProperties)* – representation of the properties of a generalized initial state;
- *cFunctionAL GetTransferFunction(cProblem ProcessProperties)* – representation of properties of the transition function;

5 Conslusion

The paper refers to a specialised IT System applied to solve discreet optimization problems; specifically an intelligent ALMM System. The system uses the special paradigm of problem modelling, namely algebraic-logical general model of multistage decision processes, both common and dynamic. The intelligent ALARM system not only has to solve discrete optimization problems, but also advises the user to apply the appropriate method/algorithm, based on the problem's properties. The system has a knowledge base containing the properties of the Library of Problem Models and Properties of Problem. The model of each problem is defined by means of elements: state structure, decision structure, initial state, transition function, sets of possible decisions and a set of admissible and non-admissible states. The main contribution of this article is to present the concept of assumptions regarding the implementation of the knowledge base of problems, in particular the relationship model problems with their properties. Three basic types of knowledge base construction were considered, and appropriate programming tools were proposed in the context of the data architecture modelling. The initial implementation of the project modules for managing the flow of information about the properties of the problem is presented using the UML scheme and class declarations in C #.

References

1. E. Dudek-Dyduch, Intelligent ALMM System for Discrete Optimization Problems – the Idea of Knowledge Base Application, ISAT, vol. 657, (2017)
2. E. Dudek-Dyduch, S. Korzonek: ALMM Solver for combinatorial and discrete optimization problems - Idea of Problem Model Library, ACIIDS Part I. LNAI, vol. 9621, (2016)
3. S. Korzonek, E. Dudek-Dyduch, Component Library of Problem Models for ALMM Solver, Journal of Information and Telecommunication, 1:3, pp. 224-240, (2017)
4. E. Dudek-Dyduch,: Modeling Manufacturing Processes with Disturbances - a New Method Based on Algebraic-Logical Meta-Model. ICAISC, Part II. LNCS, vol. 9120, (2015)
5. E. Dudek-Dyduch, E. Kucharska, L. Dutkiewicz, K. Rączka: ALMM Solver-A Tool for Optimization Problems. ICAISC 2014, pp. 328-338 (2014)
6. E. Dudek-Dyduch, E. Kucharska: Learning method for co-operation. ICCCI 2011 Part II. LNCS, vol. 6923, pp. 290-300. Springer, Heidelberg, (2011)
7. E. Dudek-Dyduch, L. Dutkiewicz: Substitution tasks method for discrete optimization. ICAISC 2013, Part II. LNCS, vol. 7895, pp. 419-430, (2013)
8. E. Dudek-Dyduch, E. Kucharska: Optimization Learning Method for Discrete Process Control. In: ICINCO 2011, vol. 1, pp. 24-33, (2011)
9. E. Dudek-Dyduch, E. Kucharska, L. Dutkiewicz, K. Rączka: ALMM Solver-A Tool for Optimization Problems. ICAISC 2014, pp. 328-338, (2014)
10. L. Dutkiewicz, E. Dudek-Dyduch: Substitution Tasks Method for Co-operation. In: Recent Developments in Computational Collective Intelligence, pp. 103-113, (2014)
11. E. Dudek-Dyduch: Learning based algorithm in scheduling. Journal of Intelligent Manufacturing (JIM), vol. 11, no 2, pp. 135-143., (2000)
12. E. Dudek-Dyduch,: Problems of knowledge representation in expert system aided control of DPP (in Polish), part I, pp. 147-154, Wrocław, (1993)

13. E. Dudek-Dyduch,: Algebraic Logical Meta-Model of Decision Processes - New Metaheuristics. ICAISC, Part 1. LNCS, vol. 9119, pp. 541-554, (2015)
14. E. Dudek-Dyduch,: Modeling Manufacturing Processes with Disturbances – Two-Stage AL Model Transformation Method, MMAR, pp. 782-787 (2015)
15. E. Dudek-Dyduch: Discrete determinable processes - compact knowledge-based model, Notas de Matematica No 137, Universidad de Los Andes Venezuela, (1993)
16. E. Dudek-Dyduch: Formalization and analysis of problems of discrete manufacturing processes. Scientific bulletin of AGH University, Automatics vol.54, (in Polish), (1990)
17. K. Rączka, E. Dudek-Dyduch, E. Kucharska, L. Dutkiewicz: ALMM Solver: the Idea and the Architecture. In: Rutkowski at al. (eds.) ICAISC 2015, Part II. LNCS, vol. 9120, pp. 504-514, Springer International Publishing, (2015)
18. P. Jędrzejowicz, E. Ratajczak-Ropel: Reinforcement Learning Strategy for Solving the MRCPSP by a Team of Agents, Intelligent Decision Technologies, Smart Innovation, Systems and Technologies, vol. 39, Springer, pp. 537-548, (2015)
19. E. Dudek-Dyduch: Information systems for production management (in Polish) Wyd. Poldex, Kraków ISBN 83-88979-12-4,(2002)
20. L Anselma,. L. Piovesan, A. Sattar, B. Stantic, A. Paolo Terenzian, Comprehensive Approach to 'Now' in Temporal Relational Databases: Semantics and Representation, IEEE Transactions On Knowledge And Data Engineering, Vol.: 28, Issue: 10, pp: 2538-2551
21. E. Dudek-Dyduch, T. Dyduch: Formal approach to optimization of discrete manufacturing processes. in: Hamza, M.H. Proc. of the Twelfth IASTED, Acta Press, Zurich, (1993)
22. F. Rossi, P. Van Beek, T. Walsh,: Handbook of Constraint Programming, Elsevier, (2006)
23. P. Terenziani, Nearly Periodic Facts in Temporal Relational Databases, IEEE Transactions on Knowledge and Data Engineering, Volume: 28, Issue: 10 , Pages: 2822-2826, (2016)
24. E. Kucharska, E. Dudek-Dyduch: Extended Learning Method for Designation of Cooperation. In: Transactions on Computational Collective Intelligence XIV, pp. 136-157, (2014)
25. J. M. Medina, C. D. Barranco, O. Pons, Evaluation of Indexing Strategies for Possibilistic Queries Based on Indexing Techniques Available in Traditional RDBMS, (2016)
26. F. Abdelhedi, A.A. Brahim, F. Atigui, G. Zurfluh, Big Data and Knowledge Management: How to Implement Conceptual Models in NoSQL Systems?, Knowledge Engineering And Knowledge Management, vol. 3 (KMIS), pp: 235-240, (2016)
27. J Błażewicz., K. Ecker, E. Pesch, G. Schmidt, J. Węglarz: Handbook on Scheduling. Springer Berlin Heidelberg New York, ISBN 978-3-540-28046-0, (2007)
28. R.E. Smith, N. Taylor: A Framework for Evolutionary Computation in Agent-Based Systems, Proc. of Int. Conf. on Intelligent Systems, pp. 221-224. 1SCA Press, (1998)