Applying Saaty's Multicriterial Decision Making Approach in Grid Resource Management

Katarzyna Wasielewska[1], Maria Ganzha[1], Marcin Paprzycki[1] Paweł Szmeja[1], Michał Drozdowicz[1], Ivan Lirkov[2], and Costin Bădică[3]

[1] Systems Research Institute Polish Academy of Sciences, Warsaw, Poland
{pawel.szmeja, katarzyna.wasielewska, drozdowicz}@gmail.com,
{maria.ganzha, marcin.paprzycki}@ibspan.waw.pl
[2] Institute of Information and Communication Technologies Bulgarian Academy of Sciences, Sofia. Bulgaria
[3] University of Craiova, Craiova, Romania

Abstract. In this paper we consider combining ontologically represented information with Saaty's Analytic Hierarchy Process (*AHP*) to facilitate decision support for Grid users. The context for the proposal is provided by the *Agents in Grid* project (*AiG*), which aims at development of an agent-based infrastructure for efficient resource management in the Grid. In the *AiG* project, agents form teams, managers of which negotiate with clients and workers terms of potential collaboration. Here, we focus on the scenario, in which the user is searching for resources to execute a task, while the resources and the expert domain knowledge are organized in an ontology. Taking into account the complex nature of resource description and domain knowledge, multicriterial assessment of how accurate is the user description of her needs, and how it can be extended/refined, plays a crucial role. For instance, it should help the user to choose optimal algorithm and/or resource to solve her problem. Furthermore, ontologically described contract proposals, that are the results of autonomous negotiations, require multicriterial assessment. The *AHP* method is based on pairwise comparisons of criteria, and relies on the judgment of a panel of experts. In the context of the *AiG* project, we show on the example of the *AHP* method, how multicriteria group decision making can be used to support user in resource selection and assessment of contract proposals.

1 Introduction

Nowadays, Grids are recognized as a valuable approach to create highly available computing infrastructure, facilitating access to the "computational power" and/or "big data" (however, here, we focus our attention on the computational Grids). For such infrastructure, consisting of heterogeneous, geographically distributed computer resources, it was initially assumed that it could become a source of income to its owners [14] (e.g. in a way similar to the Sun Grid by Oracle [1]). However, integrating of "business aspects" into the Grid requires

assurances of (i) resource availability, and (ii) timeliness of job execution. Therefore, it is necessary to formalize the contract between the user and the service provider in the form of a *Service Level Agreement* (*SLA*). Observe that, this is one of important differences between "commercial" Grids and volunteer computing initiatives (e.g. SETI@HOME).

Separately, a key issue of user support should be addressed. Specifically, in the *AiG* project it is assumed that the user describes the needed resources to execute her job. However, it is also well known that users quite often could use help in interacting with a Grid-like system. This concerns both the existing users, who could be helped in selecting "better" resources, and "novices" who need help to start using a Grid system, thus reducing their learning curve. This is a well recognized problem, and one of possible ways of providing the needed support could be the development of "decision support systems" for users of computational resources. Interestingly, while development of such systems has been tried in the early 1990's (see [22], [7]), it did not gain much traction. Therefore, work presented here represents another attempt at achieving similar goals, while applying currently available tools.

To develop the decision support system within the *AiG* project the *Analytical Hierarchy Process* (*AHP*; [23–25]) was selected as a starting point for a multicriteria decision making method. Note that, *AHP* may be replaced with other group multicriteria decision making methods e.g. PROMETHEE-GAIA, MAUT, ontological matchmaking [**?**, **?**]. However, since knowledge is organized in an ontology that can be universally used, it is crucial for the selected method to efficiently construct decision problem and deal with multiple decision-makers. *AHP* is a multicriteria decision making method known since the 1970s that helps to solve complex problems with both objective and subjective criteria. The idea behind the *AHP* method (combination of pairwise comparisons and scaling, to compare individual criteria) stems from psychological observations concerning way in which humans reach optimal decisions. There have been multiple applications of the method in its original version as well as further developments in areas such as project management [17], strategy selection [21], supplier selection [9, 10], and many more [3]. Moreover, the method has been extended and evaluated in multiple application areas (for more details see [4, 12]). The reasons why the *AHP* was selected as a multicriterial decision method in the case of the *AiG* were: (i) support for group decision making, (ii) structuring the criteria as a hierarchy, which gives the user a better view on the problem, and more precisely determines the problem's scope as well as nicely corresponds to the semantic knowledge representation, (iii) mapping between the verbal judgments and the numerical scale (user-friendliness), (iv) methods for verification of consistency criteria intensity assessment done by the user.

Before we proceed, let us make the following remark. The *AiG* project has been initialized before the global "change of interest" from the Grid computing to the Cloud computing. However, taking into account the research results of the, recently concluded, mOSAIC project [18], ideas presented here could be naturally adjusted also to this application area. At the same time, progressing

development of grid middlewares (e.g. Globus [11] and Condor [13], [8]), as well as the progress of the EMI initiative [2], [5], show that there is still place for the "old fashioned" Grid initiatives.

The aim of this paper is to continue considerations initiated in [16] and show in more detail how the multicriteria decision making method, specifically *AHP*, can be applied to the user support in the *AiG* system. The remaining parts of the paper are organized as follows. First, the scope of the *AiG* project is briefly described and the Saaty's *AHP* method is introduced. Next, we outline the structure of the domain ontology applied to support Grid users. Finally, an example is presented, illustrating how to utilize the *AHP* to support user in defining requirements regarding Grid resources, as well as in the proposal evaluation (during contract negotiations).

2 Agents in Grid

Let us start, by briefly introducing the *Agents in Grid* (*AiG*) project, which aims at the development of an infrastructure for resource management in the Grid. In the *AiG* project it is assumed that teams of software agents act as resource brokers and managers. Specifically, agents representing users can either (1) join a team and earn money, or (2) find a team to execute a job. Furthermore, agents form teams that consist of managers and workers. Managers of teams negotiate with clients and workers terms of potential collaboration. The initial overview of the approach can be found in [19], while the two main scenarios (team joining and finding team to commission job execution) were discussed in [26, 27]. Since the scope of Grid resources description and contracts (terms of collaboration) is likely to evolve over time, one of the main assumptions of the *AiG* system is to use (everywhere) ontologies and semantic data processing, as a flexible and easily extendable knowledge representation. Therefore, we have modified the CoreGrid ontology (see [28]) to develop the *AiG Grid ontology*, extended by ontologies needed for contract negotiations (for details see [20]). In [19] we have assumed that the proposed agent-semantic system will be based on the following tenets (for more details see also the discussion in [15]):

- agents work in teams (groups of agents),
- each team has a single leader—*LMaster agent*,
- each *LMaster* has a mirror *LMirror agent* that can take over its job,
- incoming workers (*worker agents*) join teams based on user-criteria,
- teams (represented by *LMaster*s) accept workers based on team-criteria,
- each *worker agent* can (if needed) play role of the *LMirror* or the *LMaster*,
- incoming users (*user agents*) search teams to commission job execution based on user-criteria,
- yellow-page matchmaking between incoming user/worker requirements and resources available within registered teams is facilitated by the *CIC* component.

Observe that, in both scenarios (team joining and job execution), the goal is to autonomously (through agent negotiations) reach "agreement on rules of collaboration." Such an agreement is then formalized in a *Service Level Agreement* (*SLA*) "document." In the *AiG* system, ontologically demarcated *SLA*'s are to be a result of multiround negotiations, within which offers are analyzed with respect to multiple criteria. Such criteria can include, among others: price, completion time, availability of a specific hardware or software, etc. Obviously, different criteria may be of different importance to the user (and the team leader), e.g. completion time may be significantly more important than price (having a cheaper contract immediately vs. having a more lucrative contract some time in the future). Therefore, multicriterial assessment of proposals is going to play a crucial role in the negotiations. This being the case, as a starting point we have selected the *AHP* method for the multicriterial analysis within the *AiG* system. In this paper we will illustrate that it can be used not only during contract negotiations, but also to provide user decision support. Let us, therefore, describe briefly the key aspects of the *AHP* method.

3 Saaty's Analytic Hierarchy Process

The *AHP* method was proposed by Saaty in [23–25]. It is based on pairwise comparisons of criteria, which enable to derive the priority scales, and evaluate alternatives based on the judgment of experts (stakeholders).

In the *AiG* project, the *AHP* method can be used at least in two areas. First, for "assessment" of user resource specification, e.g. to check if the proposed computing resource(s) is(are) appropriate for the problem to be solved. Second, for multicriterial analysis of offers submitted during the negotiation process, e.g. to establish, which combination of resources, cost and schedule is the most advantageous, taking into account user preferences. Observe that, some evaluation criteria can be assessed in a straightforward way, e.g. in the contract proposal assessment, it is often the case that the lower the price for the job execution the better for the user. However, to assess, which payment mechanism, or which CPU, is better for a given type of problem, an expert knowledge is required. Here, the expert knowledge may be "single-expert" or "multi-expert" based. The first case to be considered is when knowledge of a single expert, or knowledge of multiple experts "combined together," can be instantiated (through an application of the consensus method); and applied within the *AHP* framework. In the second case, knowledge of multiple experts will be stored separately and applied in the *AHP* method (aggregated on judgments and priorities). Observe that the latter case is the most natural scenario for the application of the *AHP* approach.

In the general case, the *AHP* method proceeds in three main phases: (i) selection of evaluation criteria, (ii) selection of experts (for more discussion concerning expert selection see [6]), and (iii) quantitative evaluation. The first two phases are completed during the design stage of the system, when the *AHP* method is being adjusted to the specific application. First, the information about the

decision that is to be made, and factors that may influence it, is gathered and structured (represented in an ordered way). Second, possible experts that will be able to evaluate alternative offers, are collected. In the *AHP*, multiple experts may (and should if only possible) be included in the evaluation of offers during the decision-making process. The goal of the quantitative phase of the *AHP* method, is to evaluate each offer numerically and to rank all of them according to how well they meet the user preferences. The following steps can be distinguished when performing the quantitative evaluation. Let us start by introducing the needed notation:

- $C_1, \ldots, C_M$—all criteria for alternatives evaluation,
- $A = \{A_i\}_{i=1,\ldots,N}$—the set of alternatives,
- $E = \{e_i\}_{i=1,\ldots,K}$—the set of experts.

1. Notice that preferences and factors that influence a decision can be depicted as a hierarchy (tree), such that the root is the specific goal (e.g. the contract or the resource). The level below the root contains the main objectives, and a group of factors influencing the decision, followed, in subsequent levels, by the criteria, on which they depend (see Fig. 1). Specifically, there are n major factors (selected from the set of M factors/criteria) influencing the decisions, and m subcriteria that influence $FACTOR_n$. Note that $n, m <= M$ since they correspond to the number of criteria on a given level, below the goal. Next, this structure is utilized to construct special matrices used to define the pairwise comparison of criteria, and finally to calculate the global priorities of all criteria that should be considered in the final evaluation of alternatives. The information stored in the comparison matrices is the most crucial to the application of the *AHP* method.

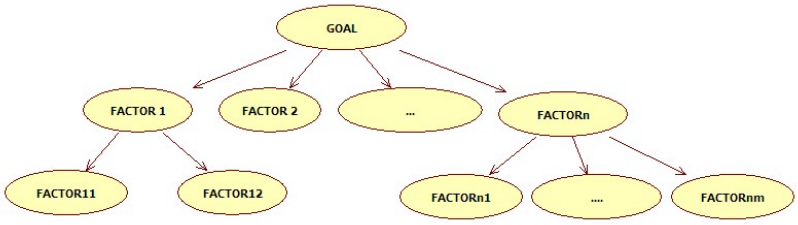

Fig. 1. Example of decision hierarchy

2. Next, a set of pairwise comparison matrices (see Matrix 1) are constructed for the criteria on a given hierarchy level. All elements on a given level are compared with respect to the element immediately above them (in the tree structure described above). From the comparison matrices, priorities of criteria on each hierarchy level can be calculated, i.e. priorities of sub-criteria with respect to a given criteria.

Each comparison c_{ij}, $i, j = 1, ..., m$ (assume that we construct matrix for subcriteria of $FACTOR_n$ from Fig. 1) is made using a specific "scaling values," proposed by Saaty that indicate to what extent one element is more important than another (see Table 1).

Intensity of importance	Definition
1	**criteria/expert i and j are equally important**
2	criteria/expert i is weakly more important than j
3	**criteria/expert i is moderately more important than j**
4	criteria/expert i is moderately plus more important than j
5	**criteria/expert i is strongly more important than j**
6	criteria/expert i is strongly plus more important than j
7	**criteria/expert i is very strongly more important than j**
8	criteria/expert i is very very strongly more important than j
9	**criteria/expert i is extremely more important than j**

Table 1. Scale used for criteria/experts comparisons ([25], [4])

Note that values in bold are treated as a fundamental scale, while these not in bold are the intermediate values.

$$\begin{bmatrix} 1 & c_{12} & ... & c_{1m} \\ c_{21} & 1 & ... & c_{2m} \\ ... & ... & ... & ... \\ c_{m1} & c_{m2} & ... & 1 \end{bmatrix} \tag{1}$$

Note that: $c_{ij} = \frac{1}{c_{ji}}$ and, therefore, $c_{ii} = 1$ and c_{ij} cannot be 0.
As it was proved in [23], matrix (1) has a maximal eigenvalue $\lambda_{max} = m + \Delta\lambda$ (where $\Delta\lambda$ is small) and a corresponding eigenvector c with positive coordinates $c_i > 0, i = 1, ..., m$ corresponding to the λ_{max}. In the *AHP* method, the priorities of the criteria are the coordinates of the normalized eigenvector $\bar{c}$:

$$\bar{c}_i = \frac{c_i}{\sum_{i=1}^{m} c_i}$$

$$\sum_{i=1}^{m} \bar{c}_i = 1.$$

3. Check the consistency ratio (CR) for the constructed comparison matrices, using a consistency index (CI) proposed by Saaty. If the CI is less than 0.1 the matrix can be considered as having acceptable consistency.
4. To obtain global priorities for all criteria c_i, $i = 1, ..., M$ the priorities evaluated from the comparisons of the criteria on each level are used to weight the priorities on the level immediately below.

5. If there is a group of K experts involved in the alternative evaluation of offers, the comparison matrix (with dimension K) for experts should be constructed and priorities of individual experts should be estimated in the same way as it was done for the individual criteria. If there is only one expert that assesses the alternatives, then her weight is 1. If there are K experts ($K > 1$), whose estimates have the same importance, then their priorities are $\frac{1}{K}$. Otherwise, the procedure allows to specify how much more important is the estimate of one expert over another. Priorities of experts (e_j) are the coordinates of the normalized eigenvector w of the comparison matrix corresponding to its maximal eigenvalue:
$w = (w_j)_{j=1,...,K}$ and $\sum_{j=1}^{K} w_j = 1$
6. The last step in the quantitative evaluation phase is to evaluate alternatives with respect to the experts. For every alternative offer we create evaluation matrix. In this matrix, every entry is the estimation of badness of the n-th alternative for the j-th expert with respect to the i-th criteria $x_{ij}^n, i = 1, ..., M, j = 1, ..., K$ (see Table 3). Here, we use the scale from Table 2 to estimate the alternatives.

Intensity of badness	Definition
-9	extremely bad
-7	very strongly bad
-5	strongly bad
-3	moderately bad
-1	bad
0	does not affect
1	good
3	moderately good
5	strongly good
7	very strongly good
9	extremely good

Table 2. Scale used for alternative estimation

	e_1	e_2	...	e_K
C_1	x_{11}^n	x_{12}^n	...	x_{1K}^n
C_2	x_{21}^n	x_{22}^n	...	x_{2K}^n
...	...	...	...	...
C_M	x_{M1}^n	x_{M2}^n	...	x_{MK}^n

Table 3. Evaluation matrix for the n-th alternative

The estimate M_n of the n-th alternative is calculated as:
$M_n = \sum_{i=1}^{M}(c_i \sum_{j=1}^{K} x_{ij}^n w_j)$
The best alternative is the one with the maximal estimate.

4 Use Cases in the *AiG* Project

In the *AiG* system, the *AHP* method can be applied primarily in two situations: (i) user is specifying requirements for resources needed to execute a job; here, the system, based on the expert domain knowledge stored in a dedicated domain ontology (*AiGExpertOntology*, see Section 5), can provide support to extend / make more accurate the requirements and in this way help to choose, as good as possible, algorithm and/or computer system for the problem, and (ii) during the multicriterial evaluation of contract proposals, as part of the negotiation between agent representing the user and managers of teams with resources meeting the criteria.

In the first scenario, let's consider a situation when user is looking for a team to commission job execution. Here, the user could specify only the specific resources needed to execute the job. However, she could also indicate the "job profile," i.e. the information about a problem and, depending on her knowledge, about the algorithm and/or input data. In order to store expert domain knowledge and to bind job profiles with expert knowledge, the *AiGExpertOntology* has been developed. Specifically, Section 6.1 presents an example where user wants to execute a job in the Grid. The job is to find a solution to system of linear equations, where the input data are in the form of an almost block diagonal matrix. User does not specify / know, which algorithm should be used for such a combination of problem and input data, however, she specifies that the preferable operating system is Linux and available storage space should be at least 1024 MB. What a user may expect from the system, is for it to suggest the best algorithm to solve the problem and eventually extend/correct her resource requirement specification so that they are optimal for solving a given problem with the suggested algorithm for a given input data. Finally, after applying the domain expert support, the personal agent representing the user sends a request to the *CIC* agent and as a response obtains list of team managers that have resources satisfying the (possibly modified) requirements.

In the second scenario, after finding teams that satisfy user requirements concerning resources, the negotiations phase starts. First, a *Call for Proposal* message is sent to managers of selected teams. Managers respond with ontologically demarcated job execution contract offers that have to be ranked, and eventually one of the offers may be accepted. Section 6.2 presents continuation of the example from Section 6.1. User sends *Call for Proposal* with resource requirements as well as restrictions on deadline penalty, job execution timeline and payment conditions with properties: (i) delay penalty, and (ii) fixed utilization price. Furthermore, the fixed utilization price has properties: (i) peak time price, (ii) off peak time price, and (iii) discount when lightly loaded. After receiving responses with contract offers from two team managers, the *AHP* method is applied to select the best alternative.

5 Domain ontology

The expert knowledge that we referred to in Section 4, is stored in the *AiGExpertOntology*. This ontology is extending the existing *AiG* ontologies [20] and, as a starting point, is focused on computational linear algebra. The main goal of the ontology is to provide concepts necessary to capture three key aspects of the selected domain: (i) problems to be solved, (ii) algorithms to solve them, (iii) objects that these algorithms operate on. Moreover, two additional concepts were introduced: domain experts (the *DomainExpert* class) and expert opinions (the *ExpertOpinion* class). These concepts represent experts recommendations concerning matching problems and algorithms. The *ExpertOpinion* class has property *hasRecommendedResource*, which points to a resource that, according to the expert, is the most suitable for solving a specific problem. Obviously, descriptions of available resources originate from the *AiG* ontology. Fig. 2 presents the preliminary hierarchy of problems in computational linear algebra. Here, we distinguish five types of problems represented with OWL classes: eigenproblem that can be further categorized into eigenvalue or eigenvector problem, least squares problem, solution of a system of linear equations, and calculation of a matrix norm. Note that, in what follows, we present only fragments of the developed ontology. These fragments are needed to illustrate the application of the *AHP* method in the two use case scenarios. Presentation and discussion of the complete ontology of computational linear algebra is out of scope of this contribution.

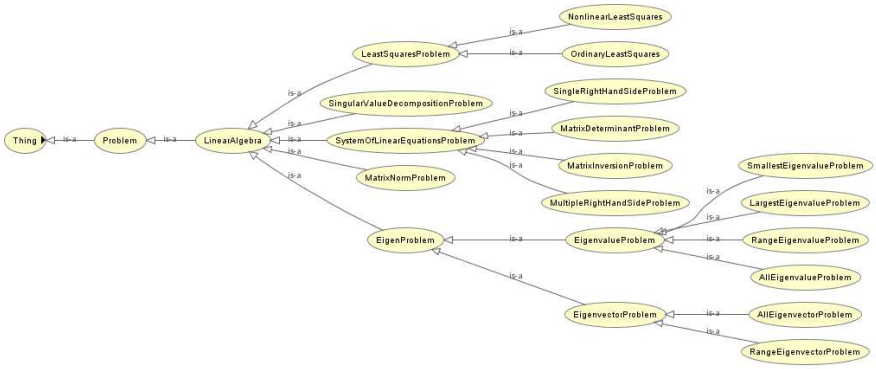

Fig. 2. Hierarchy of problems in *AiGExpertOntology*

Another important concept in the ontology is the *Algorithm* superclass, used for classes representing specific algorithms that can be used to solve problems from Fig. 2, for a given input data (represented by the *Matrix* class). Notice that there are dedicated classes for different types of solvers e.g. *Structural Matrix Solver* and classes for corresponding factorizations e.g. *GEABD* indicated with *hasFactorization* property. Part of the hierarchy is presented in Fig. 3.

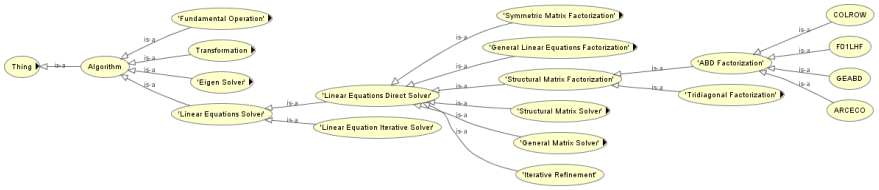

Fig. 3. Part of hierarchy of algorithms in *AiGExpertOntology*

The final parts of the ontology are the *Matrix* and the *MatrixProperty* classes (Fig. 4), and the property *hasMatrixProperty* that defines their relationship. The *MatrixProperty* class is a superclass for a hierarchy of properties that describe the matrix (e.g. symmetricity, density, structure, etc.). In the context of the introduced example, input data shall be defined as an individual of the class *Matrix* with the value of *hasProperty* being *Almost Block Diagonal Matrix*. Notice that the *Almost Block Diagonal Matrix* is a subclass of the class representing the *Block Structured Matrix*. Obviously, Fig. 4 presents only a (needed here) fragment of the already developed ontology.

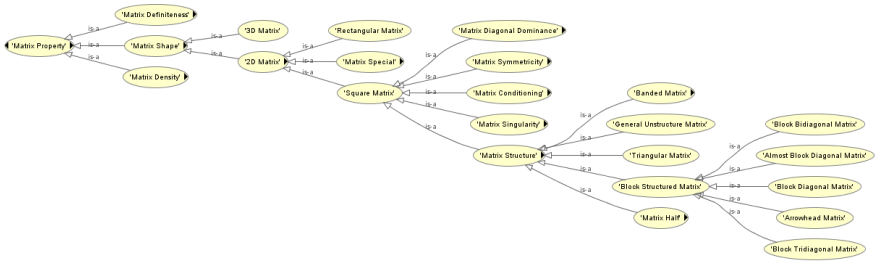

Fig. 4. Part of hierarchy of matrix properties in *AiGExpertOntology*

6 Application example

6.1 User support in resource selection

According to the description in Section 4, let's assume that the user is looking for a resource that has Linux operating system and storage space available size at least 1024 MB to solve a system of linear equations with an almost block diagonal matrix. User specifies these parameters in the web application that generates an ontologically demarcated description of resource restrictions. The generated ontology is sent to the user's personal agent for further processing. The following code snippet presents the user restrictions on the specified resource.

```
<Class rdf:about="TeamConditions#TeamCondition"><equivalentClass>
<Class>
<intersectionOf rdf:parseType="Collection">
<rdf:Description rdf:about="unnamed.owl#ComputingElement"/>
<Restriction>
  <onProperty rdf:resource="AiGGridOntology#hasStorageSpace"/>
  <someValuesFrom><Class>
  <intersectionOf rdf:parseType="Collection">
  <rdf:Description rdf:about="unnamed.owl#StorageSpace"/>
  <Restriction>
    <onProperty rdf:resource="AiGGridOntology#hasAvailableSize"/>
    <someValuesFrom><rdfs:Datatype>
    <onDatatype rdf:resource="http://www.w3.org/2001/XMLSchema#integer"/
        >
    <withRestrictions rdf:parseType="Collection"><rdf:Description>
    <xsd:minExclusive rdf:datatype="http://www.w3.org/2001/XMLSchema#
        integer">1024</xsd:minExclusive>
    </rdf:Description></withRestrictions>
    </rdfs:Datatype></someValuesFrom>
  </Restriction>
  </intersectionOf>
  </Class></someValuesFrom>
</Restriction>
<Restriction>
  <onProperty rdf:resource="AiGExpertOntology#hasJobProfile"/>
  <hasValue rdf:resource="AiGExpertOntology#LSEJobProfile"/>
</Restriction>
<Restriction>
  <onProperty rdf:resource="AiGGridOntology#isRunningOS"/>
  <hasValue rdf:resource="AiGGridOntology#debian_5.0"/>
</Restriction>
</intersectionOf>
</Class></equivalentClass>
</Class>
```

The personal agent, representing the user in the system, has the following tasks: (i) interpret the ontology and transform it into a structure that can be used in the *AHP* method, (ii) evaluate user requirements with the help of the expert knowledge, (iii) if necessary / possible, extend the user requirements with the expert suggestions, regarding algorithm / resource to be used and adequately modify structures needed for the *AHP* method.

First, the ontology snippet should be transformed into the hierarchical structure (see Section 3, step 1). Notice that, in the ontology a set of classes and properties describe the resource restrictions structure. Since an ontology can be represented as an acyclic directed graph, one can determine the structure of the decision hierarchy with the top node being the "main goal" (see Fig. 5).

Table 4 presents a comparison matrix initially accepted by the user to evaluate the resource descriptions (before introduction of expert suggestions), in which the storage space is moderately more important than the operating system. Since the available size is the only sub-criteria (with the priority set to 1) considered for the storage space, there is only one comparison matrix that needs to be constructed for such set of requirements.

In the *AHP* algorithm, priorities of the criteria are coordinates of the normalized eigenvector corresponding to the maximal eigenvalue. Therefore, priorities calculated from the user's initial comparison matrices are: 0.25 for the operating system, and 0.75 for the storage space (available size).

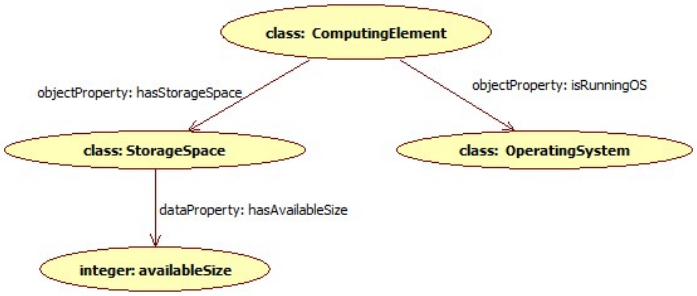

Fig. 5. Hierarchical structure of resource description

	OperatingSystem	StorageSpace
OperatingSystem	1	$\frac{1}{3}$
StorageSpace	3	1

Table 4. Comparison matrix for *ComputingElement* sub-criteria

Additionally, the user indicated the job profile (property *hasJobProfile* value) for the resource, e.g. solution of the system of linear equation with an almost block diagonal matrix. User's personal agent searches the domain ontology to find expert opinions for the pair of the system of linear equations and the input being an almost block diagonal matrix. Assume that in the ontology, there are three expert opinions (as in Section 5). Each expert recommends different algorithm and a slightly different resource.

Expert	Algorithm	Recommended resource
1	ARCECO factorization	2 processor cores and 512 MB RAM memory
2	GEABD factorization	more than 1 processor cores
3	GEABD factorization	4 processor core

Table 5. Expert opinions

Moreover, for experts from a given domain, a comparison matrix is constructed in order to estimate priorities of individual experts, since some of them may be more experienced and their opinion of greater value to the user, e.g. estimates done by a panel of Grid / HPC experts may be treated as more significant than estimates of a user who wants to commission job execution. Each comparison is made using specific "scaling values" proposed by Saaty that indicate to what extent one element is more important than another (see Table 1). In our example, Expert 1 opinion is strongly more important than Expert 2 and very strongly less important than Expert 3. Next, Expert 3 is moderately more important than Expert 1 and very strongly more important than Expert 2. Thus

priorities calculated from Table 6 are: 0.28 for Expert 1, 0.07 for Expert 2 and 0.65 for Expert 3.

	Expert 1	Expert 2	Expert 3
Expert 1	1	5	$\frac{1}{3}$
Expert 2	$\frac{1}{5}$	1	$\frac{1}{7}$
Expert 3	3	7	1

Table 6. Comparison matrix for experts

The suggested algorithms, and recommended resource configuration, for a given problem and a given input are evaluated based on the expert priorities. The algorithm and recommendation that obtains the best score, is the one suggested to the user as the optimal. In this example, following Saaty, alternatives that need to be evaluated correspond to the three available expert opinions. For the respective evaluations see Tables 7 to 9, where 0 indicates that criteria score is not affected by the alternative, 3 indicates that alternative is moderately good and 5 that alternative is strongly good with respect to a given criteria. Notice, that if user's job profile included only the information about the profile and not the input, then other criteria that could be evaluated would constitute the input.

Criteria	Expert 1	Expert 2	Expert 3
Algorithm	5	0	0
CPU (cores)	5	3	0
Memory (RAM)	5	0	0

Table 7. Badness of alternative 1 for experts

Criteria	Expert 1	Expert 2	Expert 3
Algorithm	0	5	5
CPU (cores)	3	5	3

Table 8. Badness of alternative 2 for experts

The 1-st offer evaluated in Table 7 has high score for the algorithm for the expert that suggested it, and non-affecting score for the other algorithms (the scale for the assessment is described in [16]). The criteria concerning the recommended resource are assessed respectively. The estimate M_1 of the 1-st alternative (where all criteria are treated as equally important) is:
$M_1 = \frac{1}{3} * (0.28 * 5 + 0.07 * 0 + 0.65 * 0) + \frac{1}{3} * (0.28 * 3 + 0.07 * 5 + 0.65 * 0) + \frac{1}{3} * (0.28 * 5 + 0.07 * 0 + 0.65 * 0) = 1.48$

Criteria	Expert 1	Expert 2	Expert 3
Algorithm	0	5	5
CPU (cores)	0	3	5

Table 9. Badness of alternative 3 for experts

The estimate M_2 of the 2-nd alternative is:
$M_2 = \frac{1}{2} * (0.28 * 0 + 0.07 * 5 + 0.65 * 5) + \frac{1}{2} * (0.28 * 3 + 0.07 * 5 + 0.65 * 3) = 3.37$
The estimate M_3 of the 3-rd alternative is:
$M_3 = \frac{1}{2} * (0.28 * 0 + 0.07 * 5 + 0.65 * 5) + \frac{1}{2} * (0.28 * 0 + 0.07 * 3 + 0.65 * 5) = 3$
As we can see, the 2-nd alternative has the best estimate and should be selected. Consider the fact that two out of three experts indicated the GEABD factorization (one of them is the expert with highest weight), and furthermore, recommendation for the CPU cores from the 2-nd alternative overlap with recommendations of other experts, thus validating it.

Next, the requirements specified by the user are compared with the specification of the selected resource recommendation. If the requirement on that parameter is missing in the requirements, then it is added. If the requirement on that parameter is present in the requirements, but the expert resource specification does not match it, the user should be notified.

Since the GEABD factorization and the corresponding resource specification prepared by Expert 2 was chosen as the optimal expert recommendation, it should be considered within the requirements from the user. Furthermore, experts suggest to use a machine with more than 1 processor core. Therefore, the user constraints should be extended with this condition. The following code snippet shows the extended user constraints, and Fig. 6 presents the modified hierarchical structure of the user requirements.

```
<Class rdf:about="TeamConditions#TeamCondition"><equivalentClass>
<Class>
<intersectionOf rdf:parseType="Collection">
<rdf:Description rdf:about="unnamed.owl#ComputingElement"/>
<Restriction>
  <onProperty rdf:resource="AiGGridOntology#hasStorageSpace"/>
  <someValuesFrom><Class>
  <intersectionOf rdf:parseType="Collection">
  <rdf:Description rdf:about="unnamed.owl#StorageSpace"/>
  <Restriction>
    <onProperty rdf:resource="AiGGridOntology#hasAvailableSize"/>
    <someValuesFrom><rdfs:Datatype>
    <onDatatype rdf:resource="http://www.w3.org/2001/XMLSchema#integer"/
        >
    <withRestrictions rdf:parseType="Collection"><rdf:Description>
    <xsd:minExclusive rdf:datatype="http://www.w3.org/2001/XMLSchema#
        integer">1024</xsd:minExclusive>
    </rdf:Description></withRestrictions>
    </rdfs:Datatype></someValuesFrom>
  </Restriction>
  </intersectionOf>
  </Class></someValuesFrom>
</Restriction>
<Restriction>
  <onProperty rdf:resource="unnamed.owl#hasCPU"/>
  <someValuesFrom>Class>
  <intersectionOf rdf:parseType="Collection">
```

```
  <rdf:Description rdf:about="unnamed.owl#CPU"/>
  <Restriction>
    <onProperty rdf:resource="AiGGridOntology#hasCores"/>
    <someValuesFrom><rdfs:Datatype>
    <onDatatype rdf:resource="http://www.w3.org/2001/XMLSchema#integer"/
        >
    <withRestrictions rdf:parseType="Collection"><rdf:Description>
    <xsd:minExclusive rdf:datatype="http://www.w3.org/2001/XMLSchema#
        integer">1</xsd:minExclusive>
    </rdf:Description></withRestrictions>
    </rdfs:Datatype></someValuesFrom>
    <hasValue rdf:datatype="http://www.w3.org/2001/XMLSchema#integer">4<
        /hasValue>
  </Restriction>
  </intersectionOf>
  </Class></someValuesFrom>
</Restriction>
<Restriction>
  <onProperty rdf:resource="AiGExpertOntology#hasJobProfile"/>
  <hasValue rdf:resource="AiGExpertOntology#LSEJobProfile"/>
</Restriction>
<Restriction>
  <onProperty rdf:resource="AiGGridOntology#isRunningOS"/>
  <someValuesFrom><Class>
  <intersectionOf rdf:parseType="Collection">
  <rdf:Description rdf:about="unnamed.owl#Linux"/>
  </intersectionOf>
  </Class></someValuesFrom>
</Restriction>
</intersectionOf>
</Class></equivalentClass>
</Class>
```

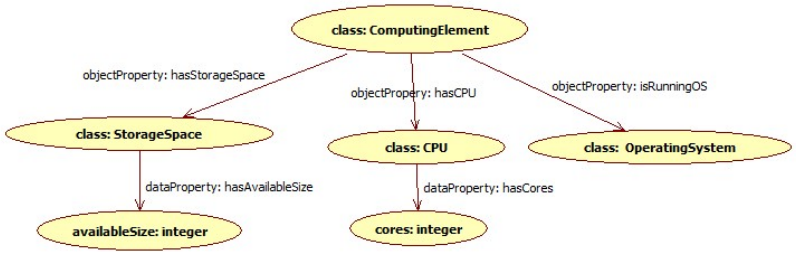

Fig. 6. Modified hierarchical structure of the resource description ontology

Moreover, the comparison matrices for the available resource description parameters should be prepared by the experts (constructed as combined expert comparison matrices as part of the group decision making [4]). Again, they should be combined with the user's comparison matrices and extended with the expert suggestions about parameters that were added to the requirements, e.g. the restriction on the *CPU*. Let's assume that from the experts comparison matrices it can be deduced that the storage space is strongly more important than the CPU, and that operating system is very strongly more important than the CPU. For the modified comparison matrix see Table 10.

	OperatingSystem	StorageSpace	CPU
OperatingSystem	1	$\frac{1}{3}$	3
StorageSpace	3	1	5
CPU	$\frac{1}{3}$	$\frac{1}{5}$	1

Table 10. Modified comparison matrix for *ComputingElement* sub-criteria

Thus the priorities calculated from the modified comparison matrix are: operating system 0.2, storage space 0.68 and CPU 0.12. These priorities shall be used to further evaluate the resource proposals.

Finally, the personal agent sends the request with the adjusted requirements to the *CIC*, to obtain the list of team managers that have available resources matching the criteria.

6.2 Evaluation of contract proposals

The second scenario (see Section 4) for applying the *AHP* method concerns the evaluation of contract offers, during the negotiation phase. First, user specifies her requirements regarding the contract, extending the resource requirements with the business terms e.g. concerning the time and the payment. Next, a *Call for Proposal* message is sent to team managers that were found at the end of the scenario outlined in Section 6.1. As a response, the user's personal agent receives contract offer(s). In the case when there is more than one offer, alternatives need to be evaluated and ranked. To be able to evaluate offers with the Saaty's method, first, a set of criteria has to be defined and the global priority has to be computed for each criterion. Fig. 7 presents the hierarchical structure generated from the ontologically demarcated contract requirements (see the following code snippet).

```
<Class rdf:about="JobExecutionConditions">
<equivalentClass><Class>
<intersectionOf rdf:parseType="Collection">
<rdf:Description rdf:about="AiGConditionsOntology#JobExecutionConditions
    "/>
<Restriction>
  <onProperty rdf:resource="AiGConditionsOntology#contractedResource"/>
  <someValuesFrom><Class><intersectionOf rdf:parseType="Collection">
  <rdf:Description rdf:about="unnamed.owl#ComputingElement"/>
  ...
  </intersectionOf></Class></someValuesFrom>
</Restriction>
<Restriction>
  <onProperty rdf:resource="AiGConditionsOntology#jobExecutionTimeline"/
      >
  <someValuesFrom><Class><intersectionOf rdf:parseType="Collection">
  <rdf:Description rdf:about="http://www.w3.org/2006/time#Instant"/>
  <Restriction>
    <onProperty rdf:resource="http://www.w3.org/2006/time#hasEnd"/>
    <hasValue rdf:resource="AiGConditionsOntology#Dec24"/>
  </Restriction>
  </intersectionOf></Class></someValuesFrom>
</Restriction>
<Restriction>
```

```
  <onProperty rdf:resource="AiGConditionsOntology#paymentConditions"/>
  <someValuesFrom><Class><intersectionOf rdf:parseType="Collection">
  <rdf:Description rdf:about="AiGConditionsOntology#PaymentConditions"/>
  <Restriction>
    <onProperty rdf:resource="AiGConditionsOntology#
        fixedUtilizationPrice"/>
    <someValuesFrom><Class><intersectionOf rdf:parseType="Collection">
    <rdf:Description rdf:about="AiGConditionsOntology#Pricing"/>
    <Restriction>
      <onProperty rdf:resource="AiGConditionsOntology#
          discountWhenLightlyLoaded"/>
      <someValuesFrom><rdfs:Datatype>
      <onDatatype rdf:resource="http://www.w3.org/2001/XMLSchema#float"/
          >
      <withRestrictions rdf:parseType="Collection"><rdf:Description>
      <xsd:minExclusive rdf:datatype="http://www.w3.org/2001/XMLSchema#
          float">5</xsd:minExclusive>
      </rdf:Description></withRestrictions>
      </rdfs:Datatype></someValuesFrom>
    </Restriction>
    <Restriction>
      <onProperty rdf:resource="AiGConditionsOntology#offPeakTimePrice"/
          >
      <someValuesFrom><rdfs:Datatype>
      <onDatatype rdf:resource="http://www.w3.org/2001/XMLSchema#float"/
          >
      <withRestrictions rdf:parseType="Collection"><rdf:Description>
      <xsd:maxExclusive rdf:datatype="http://www.w3.org/2001/XMLSchema#
          float">100</xsd:maxExclusive>
      </rdf:Description></withRestrictions>
      </rdfs:Datatype></someValuesFrom>
    </Restriction>
    <Restriction>
      <onProperty rdf:resource="AiGConditionsOntology#peakTimePrice"/>
      <someValuesFrom><rdfs:Datatype>
      <onDatatype rdf:resource="http://www.w3.org/2001/XMLSchema#float"/
          >
      <withRestrictions rdf:parseType="Collection"><rdf:Description>
      <xsd:maxExclusive rdf:datatype="http://www.w3.org/2001/XMLSchema#
          float">100</xsd:maxExclusive>
      </rdf:Description></withRestrictions>
      </rdfs:Datatype></someValuesFrom>
    </Restriction>
  </intersectionOf></Class></someValuesFrom>
  </Restriction>
  <Restriction>
    <onProperty rdf:resource="AiGConditionsOntology#delayPenalty"/>
    <someValuesFrom><rdfs:Datatype>
    <onDatatype rdf:resource="http://www.w3.org/2001/XMLSchema#float"/>
    <withRestrictions rdf:parseType="Collection"><rdf:Description>
    <xsd:minExclusive rdf:datatype="http://www.w3.org/2001/XMLSchema#
        float">20</xsd:minExclusive>
    </rdf:Description></withRestrictions>
    </rdfs:Datatype></someValuesFrom>
  </Restriction>
</intersectionOf></Class></someValuesFrom>
</Restriction>
<Restriction>
  <onProperty rdf:resource="AiGConditionsOntology#deadlinePenalty"/>
  <someValuesFrom><rdfs:Datatype>
  <onDatatype rdf:resource="http://www.w3.org/2001/XMLSchema#float"/>
  <withRestrictions rdf:parseType="Collection"><rdf:Description>
  <xsd:minExclusive rdf:datatype="http://www.w3.org/2001/XMLSchema#float
      ">10</xsd:minExclusive>
  </rdf:Description></withRestrictions>
  </rdfs:Datatype></someValuesFrom>
</Restriction></intersectionOf>
</Class></equivalentClass>
```

</Class>

The requirements specified by the user are: deadline penalty more than 10 monetary units, delay penalty more than 20, peak time price less than 100, off-peak time price less than 100, discount when lightly loaded greater than 5, job execution timeline end on December 24th, available storage space greater than 1024 MB, CPU with 4 cores, Linux operating system.

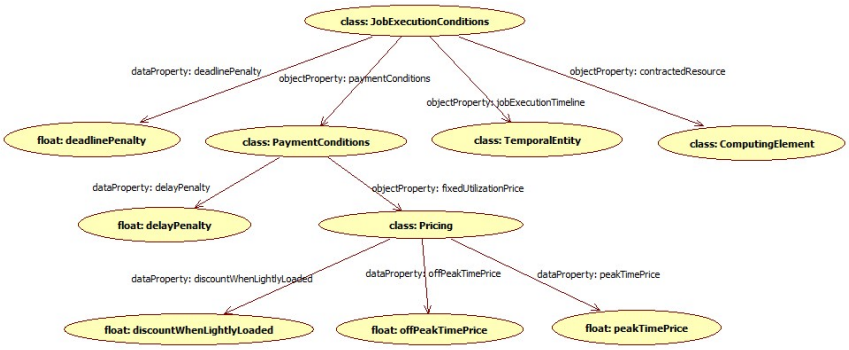

Fig. 7. Hierarchical structure of contract description

Based on the properties defined in the ontology, we can construct matrices with pairwise comparisons of criteria corresponding to these properties. Tables 11, 12, 13 present comparison matrices for each level in the hierarchy from Fig. 7, where the elements in the lower level are compared with respect to the element immediately above them. Notice that, under node *class: ComputingElement*, the structure from Fig. 5 should be placed.

	deadlinePenalty	paymentCondition	jobExecTimeline	contractedResource
deadlinePenalty	1	$\frac{1}{5}$	$\frac{1}{5}$	$\frac{1}{9}$
PaymentCondition	5	1	2	$\frac{1}{7}$
jobExecTimeline	5	$\frac{1}{2}$	1	$\frac{1}{7}$
contractedResource	9	7	7	1

Table 11. Comparison matrix for *JobExecutionCondition* sub-criteria

Priorities calculated from Table 11 are: deadline penalty 0.04, payment conditions 0.16, job execution timeline 0.11 and contracted resource 0.69.

Priorities calculated from Table 12 are: delay penalty 0.12, fixed utilization price 0.88.

Priorities calculated from Table 13 are: discount when lightly loaded 0.08, off-peak time price 0.49, peak time price 0.43.

	delayPenalty	fixedUtilPrice
delayPenalty	1	$\frac{1}{7}$
fixedUtilPrice	7	1

Table 12. Comparison matrix for *PaymentCondition* sub-criteria

	discountWhenLightlyLoaded	offPeakTimePrice	peakTimePrice
discountWhenLightlyLoaded	1	$\frac{1}{7}$	$\frac{1}{5}$
offPeakTimePrice	7	1	1
peakTimePrice	5	1	1

Table 13. Comparison matrix for *Pricing* sub-criteria

The next step in the *AHP* method is to calculate the global properties for all restricted criteria. The priorities evaluated from comparisons of the criteria on each level are used to weight the priorities on the level immediately below and to obtain global priorities (see Table 14).

Criteria	Global priority
deadlinePenalty	0.04
delayPenalty	$0.16 * 0.12 = 0.02$
discountWhenLightlyLoaded	$0.16 * 0.88 * 0.08 = 0.01$
offPeakTimePrice	$0.16 * 0.88 * 0.49 = 0.07$
peakTimePrice	$0.16 * 0.88 * 0.43 = 0.06$
jobExecutionTimeline	0.11
StorageSpace (available size)	$0.69 * 0.68 = 0.47$
CPU (cores)	$0.69 * 0.12 = 0.08$
OperatingSystem	$0.69 * 0.2 = 0.14$

Table 14. Global priorities for *JobExecutionContract* sub-criteria

Let's assume that two contract offers were received, with parameter values from Table 15.

For every alternative, an evaluation matrix is created, estimating badness of that alternative for a given criteria, according to the given expert (see Table 16). Let's assume that alternatives are evaluated by one expert, i.e. the system is applying the scale presented in Table 2. In the ontologically demarcated contract offers, exact values for parameters are received. Next, these exact values are mapped onto the values from a proposed absolute scale of judgment. This mapping utilizes rules that are defined in the system, as well as experts knowledge e.g. price should be minimized.

The 1-st offer evaluation (by one expert - the system) is:
$M_1 = 5 * 0.04 + 1 * 0.02 - 1 * 0.01 + 1 * 0.07 + 1 * 0.06 + 1 * 0.11 + 1 * 0.47 + 1 * 0.08 + 1 * 0.14 = 1.14$
The 2-nd offer evaluation becomes:

Criteria	Alternative 1	Alternative 2
deadlinePenalty	19	15
delayPenalty	21	35
discountWhenLightlyLoaded	3	10
offPeakTimePrice	90	90
peakTimePrice	90	80
jobExecutionTimeline	Dec24	Dec24
StorageSpace (available size)	1024	2056
CPU (cores)	2	4
OperatingSystem	Linux	Linux

Table 15. Parameter values received in offers

Criteria	Alternative 1	Alternative 2
deadlinePenalty	5	3
delayPenalty	1	5
discountWhenLightlyLoaded	-1	1
offPeakTimePrice	1	1
peakTimePrice	1	3
jobExecutionTimeline	1	1
StorageSpace (available size)	1	3
CPU (cores)	1	3
OperatingSystem	1	1

Table 16. Evaluation matrix for alternatives

$M_2 = 3 * 0.04 + 5 * 0.02 + 1 * 0.01 + 1 * 0.07 + 3 * 0.06 + 1 * 0.11 + 3 * 0.47 + 3 * 0.08 + 1 * 0.14 = 2.38$

As can be seen, the 2-nd alternative has a better estimate and should be selected. Specifically, considering the criteria with the highest priorities, the 2-nd offer has better available storage space and peak time price; while the operating system, the job execution timeline and the CPU in both cases meet the criteria and have equal values.

Concluding remarks

In this paper we have demonstrated, how to apply the *AHP* method to support the user in resource selection, to formulate requirements for job execution, and to compare ontologically described offers in generic contract negotiations. User support presented in the first scenario was used to: (i) advise user on algorithm selection, (ii) extend both resource requirements and comparison matrices that are the crucial components in the *AHP* method. The outcome of the first scenario was further included in the alternative evaluation that takes place during the negotiation phase. Notice that a full graph of the ontological resource description as well as job execution contract, would result in far more complex Saaty‘s hierarchy graph. However, the structure of the ontology makes the mapping between the ontological description and a decision hierarchy from

the *AHP* quite intuitive and, therefore, enables to utilize the *AHP* method by the software agents representing user/workers to perform multicriterial analysis. Moreover, the method should be able to deal with arbitrarily large ontologies and corresponding problem structures. Additionally, the domain knowledge acquired from the experts and ontologically represented shall be further included in the *AiGExpertOntology*, and incorporated into the user support mechanism.

References

1. Sun Utility Computing. `http://www.sun.com/service/sungrid/`, 2012.
2. EMI, MNA3.2 Open Source Software Initiative, jan 2013. EMI Collaboration.
3. A.Ishizaka and A.Labib. Analytical Hierarchy Process and Expert Choice: Benefits and Limitations. *ORInsight*, 22(4):201–220, 2009.
4. A.Ishizaka and A.Labib. Review of the main developments in the Analytic Hierarchy Process. *Expert Systems and Applications*, 38(11):14336–14345, 2011.
5. C.Aiftimiei, A.Aimar, A.Ceccanti, M.Cecchi, A.Di Meglio, F.Estrella, P.Fuhrmann, E.Giorgio, B.Konya, L.Field, J.K. Nilsen, M.Riedel, and J.White. Towards next generations of software for distributed infrastructures : the European Middleware Initiative.
6. C.Badica, S.Ilie, M.Kamermans, G.Pavlin, A.Penders, and M.Scafes. Multi-Agent Systems, Ontologies and Negotiation for Dynamic Service Composition in Multi-Organizational Environmental Management. In *Software Agents, Agent Systems and Their Applications*, NATO Science for Peace and Security Series, pages 286–306. IOS Press, 2012.
7. J.S. Dodgson, M. Spackman, A. Pearman, and L.D. Phillips. Multi-criteria analysis: a manual. Economic history working papers, London School of Economics and Political Science, Department of Economic History, 2009.
8. D.Petcu and V.Negru. Interactive system for stiff computations and distributed computing. In *Proceedings of IMACS'98: International Conference on Scientific Computing and Mathematical Modelling*, pages 126–129. IMACS, June 1998.
9. D.Thain, T.Tannenbaum, and M.Livny. Condor and the Grid. In T.Hey F.Berman, G.Fox, editor, *Grid Computing: Making the Global Infrastructure a Reality*. John Wiley & Sons Inc., December 2002.
10. Janos Fulop. Introduction to decision making methods.
11. G.Kabir and R.S.Sumi. An Ontology-Based Intelligent System with AHP to Support Supplier Selection. *Suranaree Journal of Science and Technology*, 17(3):249–257, 2010.
12. I.Chamodrakas, D.Batis, and D.Martakos. Supplier selection in electronic marketplaces using satisficing and fuzzy AHP. *Expert Systems with Applications*, 37:490–498, 2010.
13. I.T.Foster. Globus Toolkit Version 4: Software for Service-Oriented Systems. In *NPC*, volume 3779 of *Lecture Notes in Computer Science*, pages 2–13. Springer, 2005.
14. J.A.Alonso and M.T.Lamata. Consistency in the Analytic Hierarchy Process: A New Approach. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 14(4):445–459, 2006.
15. J.Basney and M.Livny. Deploying a High Throughput Computing Cluster. In R.Buyya, editor, *High Performance Cluster Computing: Architectures and Systems, Volume 1*. Prentice Hall PTR, 1999.

16. J.Foster and C.Kesselman, editors. *The Grid 2, Second Edition: Blueprint for a New Computing Infrastructure*. The Elsevier Series in Grid Computing. Elsevier, 2004.
17. K.Wasielewska, M.Drozdowicz, M.Ganzha, M.Paprzycki, N.Attaui, D.Petcu, C.Badica, R.Olejnik, and I.Lirkov. Negotiations in an Agent-based Grid Resource Brokering Systems. In P. Ivanyi and B.H.V. Topping, editors, *Trends in Parallel, Distributed, Grid and Cloud Computing for Engineering*. Saxe-Coburg Publications, Stirlingshire, UK, 2011.
18. K.Wasielewska and M.Ganzha. Using analytic hierarchy process approach in ontological multicriterial decision making - Preliminary considerations. In *AIP Conference Proceedings*, volume 1487/1, pages 95–103. American Institute of Physics, 2012.
19. Kamal M. Al-Subhi Al-Harbi. Application of the AHP in project management. *International Journal of Project Management*, 19(1):19–27, 2001.
20. B.Di Martino, D.Petcu, R.Cossu, P.Goncalves, T.Mahr, and M.Loichate. Building a Mosaic of Clouds. In *Evaluating open-source cloud computing solutions, MIPRO, 2011*, Lecture Notes in Computer Science, pages 529–536. Springer, 2011.
21. M.Dominiak, W.Kuranowski, M.Gawinecki, M.Ganzha, and M.Paprzycki. Utilizing agent teams in grid resource management—preliminary considerations. In *Proc. of the IEEE J. V. Atanasoff Conference*, pages 46–51, Los Alamitos, CA, 2006. IEEE CS Press.
22. M.Drozdowicz, K.Wasielewska, M.Ganzha, M.Paprzycki, N.Attaui, I.Lirkov, R.Olejnik, D.Petcu, and C.Badica. Ontology for Contract Negotiations in Agent-based Grid Resource Management System. In P.Ivanyi and B.H.V.Topping, editors, *Trends in Parallel, Distributed, Grid and Cloud Computing for Engineering*. Saxe-Coburg Publications, Stirlingshire, UK, 2011.
23. M.K.Chen and S.-C.Wang. The critical factors of success for information service industry in developing international market: using analytic hierarchy process (AHP) approach. *Expert Systems with Applications*, 37:694–704, 2010.
24. M.Lucks. *A Knowledge-Based Framework for the Selection of Mathematical Software*. PhD thesis, Southern Methodist University, 1990.
25. T.L. Saaty. *The Analytic Hierarchy Process*. RWS Publications, Pittsburg, 1990.
26. T.L.Saaty. How to make a decision: The Analytic Hierarchy Process. *European Journal of Operational Research*, 48:9–26, 1990.
27. T.L.Saaty. Decision making with the analytic hierarchy process. *International Journal of Services Sciences*, 1(1):83–98, 2008.
28. W.Kuranowski, M.Ganzha, M.Gawinecki, M.Paprzycki, I.Lirkov, and S.Margenov. Forming and managing agent teams acting as resource brokers in the Grid—preliminary considerations. *International Journal of Computational Intelligence Research*, 4(1):9–16, 2008.
29. W.Kuranowski, M.Ganzha, M.Paprzycki, and I.Lirkov. Supervising Agent Team an Agent-based Grid Resource Brokering System – Initial Solution. In F.Xhafa and L.Barolli, editors, *Proceedings of the Conference on Complex, Intelligent and Software Intensive Systems*, pages 321–326, Los Alamitos, CA, 2008. IEEE CS Press.
30. W.Xing, M.D.Dikaiakos, R.Sakellariou, S.Orlando, and D.Laforenza. Design and Development of a Core Grid Ontology. In *Proc. of the CoreGRID Workshop: Integrated research in Grid Computing*, pages 21–31, 2005.