Multi-contextual Trust Model for Multi-Agent Systems

Jan Samek*
Department of Intelligent Systems
Faculty of Information Technology
Brno University of Technology
Božetěchova 2, 612 66 Brno, Czech Republic
samejan@fit.vutbr.cz

Abstract

This paper deals with trust modelling for distributed systems especially to multi-context trust modelling for multi-agent distributed systems. There exists many trust and reputation models but most of them do not dealt with the multi-context property of trust or reputation. Therefore, the main focus of this thesis is on analysis of multi-context trust based models and provides main assumptions for new fully multi-contextual trust model on the bases of them. The main part of this thesis is in providing new formal multi-context trust model which are able to build, update and maintain trust value for different aspects (contexts) of the single entity in the multi-agent system. In our proposal, trust value can be built on the bases of direct interactions or on the bases on recommendations and reputation. Moreover we assume that some context of one agent is not fully independent and on the bases of trust about one of them we are able to infer trust to another's. Main contribution of this new model is increasing the efficiency in agent decision making in terms of optimal partner selection for interactions. Proposed model was verified by implementing prototype of multi-agent system when trust was used for agents' decision making and acting.

Categories and Subject Descriptors

C.2.4 [**Computer-Communication Networks**]: Distributed Systems; I.2.11 [**Distributed Artificial Intelligence**]: Multiagent systems

Keywords

trust, multi-context trust, reputation, distributed multi-agent systems, HMTC

*Recommended by thesis supervisor: Assoc. Prof. Petr Hanáček. Defended at Faculty of Informatics Technology, Brno University of Technology on January 4, 2012.

Samek, J. Multi-contextual Trust Model for Multi-Agent Systems. Information Sciences and Technologies Bulletin of the ACM Slovakia, Vol. 4, No. 1 (2012) 44-54

1. Introduction

Nowadays, we have many trust and reputation models [8, 17, 10], but only few of them deals with multi-context nature of trust and provide some computational solution.

In a single-context environment, entities trust in system is evaluated into one dimensional value, when trust is a number from some interval (for example $[0, 1]$), when the lower limit represents that entity is *fully untrustworthy* and the upper limit represents that entity is *fully trustworthy*. This single-context environment concept is sufficient for many models [14, 13, 17] and applications [15, 21, 22, 9] and it is well studied.

In our research, we concentrate to *multiple context* (or multi-context) trust environment [13], which is more complex in trust evaluating and it is closer to real word trust concept. In a multi-context environment, entities trust is always associated with some *context*. Toward this, for multi-context environment we can't simply say that entity is good or bad, we have to say that entity in such aspect or in providing some service is good/bad (trustworthy/untrustworthy), in another service is good/bad and so on.

Complexity of models with multi-context solution is typically greater than it is in the models with a single-context. Moreover, *context* have not a stable meaning overall of the models which propose any solution. For example, context in [11] is seen as a situation of transaction where several aspects are considered in. In the other example [13] a context is a set of attributes and their instantiated values about an environment and reputation is defined as relation between two agents in given context.

The main contribution of this paper is in providing new multi-context trust model for multi-agent systems which is able in some cases provide trust inferring between contexts. We have developed *Hierarchical Model of Trust in Contexts* (HMTC) [20] which correspond to agent's belief base from different aspects of their possible counterparts, these aspects (contexts) are connected into multi-level graph and connections between different aspects represents their relationship. We also provide computation model [20, 19] for inferring trust into contexts with incomplete knowledge. We also provide robust trust evaluation mechanism [18] which is based on statistical methods for estimation parameter of Normal distribution. Our results are validated by implementation of experimental multi-agent framework, when agents decision making and interacting is based on the trust.

The structure of this paper is organised as follow: In Section 2 we describe related work and state of art in the area of multi-contextual trust and reputation models. Our main contribution, Hierarchical Model of Trust in Contexts, is formalised in Section 3. Section 4 described trust evaluation principle from direct interactions between agents based on statistical methods. Next Section 5 provide experimental verification of our proposal for different cases. The last Section 6 contains a conclusion and outlines possible directions of future research.

2. Related Work

At this section we make shortly brief to the meaning of trust and next describes most representative multi-contextual trust models in the area of computer sciences.

2.1 Trust

Trust as an explicit concept is not the one that has a mutually accepted definition. We have identified the existence of trust and reputation in many disciplines of human behaviour, for example: economists, sociologists and computer science [13, 1]. In different areas we have different definitions as well as several different definitions in one discipline.

In many scientific works in related area, we can found an *Diego Gambetta* definition of trust, which have origin in work *Can We Trust Trust?* [5] and states as follows:

> „*... trust (or, symmetrically, distrust) is a particular level of the subjective probability with which an agent will perform a particular action, both before [we] can monitor such action (or independently of [our] capacity of ever to be able to monitor it) and in a context in which it affects [our] own action.*“

The most adopted trust definition in the area of computer sciences and formal trust models is a formulation by Lik Mui *et al.* [14]:

> „*Trust: a subjective expectation an agent has about another's future behavior based on the history of their encounters.*“

As we say before, in different literature uses the term trust with a variety of meanings. This meaning and definition overview can be found at [11, 7, 6].

2.2 Multi-Contextual Trust Models

One of the first well formalized models is described in dissertation of *S. Marsh* [11]. In this work, the multi-context property of trust is expressed in the term *situation trust* which is a representation for the amount of trust an agent has in another one in a given situation. There are also two other kinds of trust: *basic* and *general*, but the situation trust is of most importance when trust is considered in cooperative situations. To estimate the situation trust *Marsh* uses general trust (trust between two agent irrespective to situation), *utility* and *importance* factors for specified context.

Cognitive trust model proposed by authors *Castelfranchi & Falcone* in [3] considers the relation between trust and delegation. Trust between agents in a task is evaluated considering some essential groups of beliefs (competence, dependence, disposition, fulfilment). Delegation action in this proposal corresponds to the decision making based on trust in a specific *situational context*. This situational context is defined as a set of propositions describing the ”state of the world”.

In [2] *Rahman & Hailes* propose their trust model, which is based on trust fuzzy degree representation. Direct trust in this model is defined as agent's belief in another agent's trustworthiness within a certain *context* to a certain degree (fuzzy set). The term *context* is open in this work and agents are able to define their own contexts when using this trust model.

The model called REGRET proposed in [16] by *Sabater & Sierra* is one of the most cited reputation models in multi-agent systems that respects multi-context aspect of trust and reputation. The context in REGRET is encapsulated in the term *subject* which is a part/variable of the *outcome* of a dialogue between two agents. In this model, three different dimensions are recognized: the *individual* (direct interactions), the *social* (experiences of group members, information from third party agents) and the *ontological* (combination of multiple aspects in order to build a reputation of complex concepts).

3. Hierarchical Model of Trust in Contexts

Structure, behaviour and properties of the proposed *Hierarchical Model of Trust in Contexts* (HMTC) will be introduced during the following sections. First of all we state some premises. This model is intended to be used for agent's reasoning which is based on some beliefs about some qualities of other elements in the system. Our concept supposes that each element in the system has different qualities which are not completely independent and these qualities are correlated together with using specification and generalisation – some qualities or attributes in lower abstraction level can create one or more common qualities in higher abstraction level.

Entities in the system can be rated in these different qualities and agent's reasoning is done by combination of different beliefs in different qualities with using connections in such qualities. In this paper, we do not describe how a hierarchical model of qualities for different entities is made and how qualities are connected. We suppose that model is created empirically from knowledge of the real system of interest.

3.1 Trust Representation

In our proposal of HMTC trust model, we use a unique trust representation by the interval. The *trust interval* is an interval which is bounded by limits $x, y \in [0, 1]$, where x denotes the lower limit of trust and y denotes the upper limit of trust. This representation of trust allows us to express *uncertainty* directly in trust value. The lower limit is the worst possible trust rating, the upper limit is the best possible trust rating. Thus, we denote τ as a set of all possible trust intervals $[x, y]$ where $x, y \in [0, 1]$ and $x \leq y$. In case we do not have any knowledge on how to built trust, the uncertainty is maximum, we have to set the implicit trust to interval $[0, 1]$.

The meaning of this trust interval is that an agent subjectively perceives another agent's capability and reliability with certain amount of uncertainty in specified context.

The value 0 means absolutely un-trusted opinion (distrust), on the other hand, value 1 means absolutely trusted opinion. This interval reflects how trustworthy an agent is in specific context and also uncertainty of this trustworthiness. For example, trust interval $[0.7, 0.9]$ means that it is expected that in the worst case the outcome of the service provided by this agent is 0.7 and in the best case it is 0.9.

Moreover, for the case of the belief *conflict* we need to define special trust value. For this, we use an empty set value $\emptyset$ and finally trust interval τ is defined as follow:

$$\tau \subset [0, 1] \cup \emptyset \tag{1}$$

Meaning of the $\emptyset$ value and conflict principle will be described in following sections.

3.2 HMTC structure

HMTC is a multilevel graph defined as usually as a set of nodes and a set of edges. Nodes represent trust contexts and edges represent relations between the contexts. The multilevel property creates hierarchical structure of contexts, where higher levels of contexts represent more general properties and the lower levels represent more specific properties of the entity. Thus, each context (graph node) has exactly defined its level. The top level of the graph – root node, level 1 – represents the overall trust of the entity. Nodes on the bottom level are *leaf-nodes*.

Definition 1. Hierarchical Model of Trust in Contexts is an n-tuple $HMTC = (N, E, T, \rho, w)$, where:

1. N is finite set of nodes. Nodes can be split into disjunction non-empty sets $N_1, N_2 \dots N_n$ so that $N = N_1 \cup N_2 \cdots \cup N_n$, which define n levels of graph.
2. E is finite set of edges. As well as for the node set there exist division of the edges set to the sets $E_1, E_2 \dots E_{n-1}$, then $E = E_1 \cup E_2 \cdots \cup E_{n-1}$. Edge connects pair of nodes from the node set and we denote edge between nodes $u, v \in N$ as $(u, v) = e_{u,v}$. Edges are allowed only between two nodes in neighbour levels.
3. T is ordered set of time moments. Our model is a discrete system which behaviour is defined in some time moments which constitute a time set T. We denote particular time moments as $T = \{t_1, t_2 \dots t_i\}$ for which condition $t_1 < t_2 < \cdots < t_i$ must be valid.
4. ρ is *trust function*. Trust function maps every pair: *node* and a *time moment* to the *trust interval*:
$$\rho : N \times T \to \tau \, . \tag{2}$$
5. w is *weight function*. Weight function is used to express strength of relation between different nodes. Weight function maps each *edge* to the *weight interval*, which is interval $[0, 1]$ from the set of $\mathbb{R}$.
$$w : E \to [0, 1] \tag{3}$$
For every non-terminal (terminal nodes are nodes, which have not any sub-nodes in lower level connected by the edge) nodes, there is a restriction that weights sum of node's incoming edges (direction from a lower level to a higher level) is always equal to 1:
$$\forall u^l \in N - N_T : \sum_{\forall v^{l+1}} w((u^l, v^{l+1})) = 1 \tag{4}$$
where N_T is set of terminal nodes.

   Definition of this function also includes restriction that weight of edges in forward direction is equal to weight of vertex in opposite direction:
$$\forall e^l_{u,v} : w(e^l_{u,v}) = w(e^{-l}_{u,v}). \tag{5}$$
   We suppose that after the model is made, the weight function is defined and can't change in time.

3.3 Trust Inferring and Computation

In HMTC we recognize two types of trust inferring which depends on their direction in respect of evaluated nodes. Before describing mentioned inferring procedure, we need to define basic functions over trust interval which are used in trust inferring procedure.

3.3.1 Functions over trust interval

For describing trust computation in HTMC we need to define basic operations on extended trust interval τ. For any variable $\vartheta, \gamma \in \tau$ and for any constant $k \in \mathbb{R}$ we define basic operations as follows:

Multiplication by constant

$$\begin{aligned} \vartheta \cdot k &= [\vartheta_{min}, \vartheta_{max}] \cdot k \\ &= norm\,[\vartheta_{min} \cdot k, \vartheta_{max} \cdot k] \end{aligned} \tag{6}$$

Division by constant

$$\frac{\vartheta}{k} = norm\left[\frac{\vartheta_{min}}{k}, \frac{\vartheta_{max}}{k}\right] \tag{7}$$

Addition

$$\vartheta + \gamma = norm\,[\vartheta_{min} + \gamma_{min}, \vartheta_{max} + \gamma_{max}] \tag{8}$$

Subtraction

$$\vartheta - \gamma = norm\,[\vartheta_{min} - \gamma_{max}, \vartheta_{max} - \gamma_{min}] \tag{9}$$

Intersection

$$\vartheta \cap \gamma = \{x : x \in \vartheta \wedge x \in \gamma\} \tag{10}$$

As you can see, addition and subtraction comes from classical interval arithmetic [23] and intersection operation is defined as excepted from the set theory.

Function *norm* ensures, that the result of above mentioned operations is always normalized to the trust interval, thus *min* a *max* components will always be mapped to the basic trust interval $[0, 1]$.

$$norm\,[x, y] = [min(max(0, x), 1), min(max(y, 0), 1)] \quad \text{pro } x \leq y \text{ a } x, y \in \mathbb{R} \, . \tag{11}$$

3.3.2 Type of computations

In a trust inferring procedure we recognize two types of computations. First kind of computation is called *up-direction* computation and is used for the parent node trust computation on the bases on knowledge about its child nodes trusts. This computation can be described with a function *up*:

$$up(a^l, t) = \sum_{\forall e^l_{a,b}, \forall b^{l+1}} w(e^l_{a,b}) \cdot \rho(b^{l+1}, t) \qquad (12)$$

where a^l denotes node a in level l; $e^l_{a,b}$ denotes an edge between nodes a and b. The example of trust inferring with using *up-direction* computation for node a is in Fig. 1.

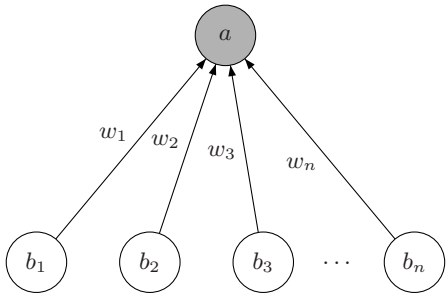

Figure 1: Example of trust inferring with using *up-direction* computation.

Second kind of computation is called *down-direction* computation and is used in a case when a child node trust is computed and the computation is based on knowledge about parent(s) and neighbour(s) node(s) trusts.

$$down(a^l, t) = \bigcap_{\forall e^{l-1}_{b,a}, \forall b^{l-1}} \frac{\rho(b^{l-1}, t) - \sum_{\forall c^l, \forall e^{l-1}_{b,c}} w(e^{l-1}_{b,c}) \cdot \rho(c^l, t)}{w(e^{l-1}_{b,a})} \qquad (13)$$

Simple example of down-direction computation for trust inferring of node b_1 is shown in Fig. 2.

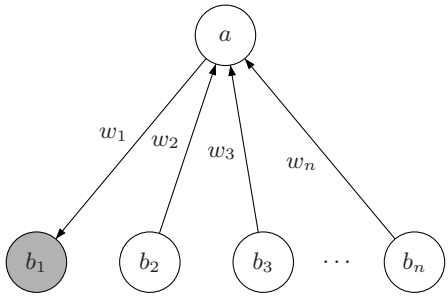

Figure 2: Example of trust inferring with using *down-direction* computation.

As we can see, up-direction computation can be used directly to the root node and on the other hand down-direction computation can be directly used for terminal nodes. For the non-root and non-terminal nodes, combination of both computations has to be used with using *intersection* to combine both reverse directions. Thus, we can finally define *eval* function for any node a in any level l of HMTC with maximal level L:

$$eval(a^l, t) = \begin{cases} up(a^l, t) & \text{for root node} \\ down(a^l, t) & \text{for terminal nodes} \\ up(a^l, t) \cap down(n^l_a, t) & \text{otherwise} \end{cases} \qquad (14)$$

3.4 Events and Behaviour of the Model

Behaviour of the basic HMTC model is driven by two kinds of events: *external event* and *internal event*. Each node can be updated by an external event, which comes from outside of the model (new knowledge about entity trustworthiness from interaction, recommendation, etc.). Result of such event is that the targeted node's trust interval is updated as the intersection of the event interval with the original node's interval. Consequently to the primary event, after the target node trust interval is evaluated, all the parent and child nodes may be updated with some internal events.

3.4.1 Event

An event in HMTC is always associated with a node from N and trust interval from τ which represent *obtained* trust interval for the node. New node interval is an intersection of its previous trust interval with this obtained trust interval.

In a case when we talk about an external event, amount of the obtained trust interval corresponds to the function *external*. Consider that *external* function is an abstract function which maps node from N in a time moment from T to the trust interval:

$$external : N \times T \to \tau \qquad (15)$$

In a second case, when we talk about internal events, amount of the obtained trust interval corresponds now to the *eval* function which was presented in equation (14). Finally, we can define *event* as a tuple:

$$u = (n, \delta) \qquad (16)$$

Where $u \in U$, $U$is an events set, $n \in N$ and $\delta \in \tau$. Obtained trust interval δ is defined as:

$$\delta = \begin{cases} external(n, t) & \text{for external event} \\ eval(n, t) & \text{for internal event} \end{cases} \qquad (17)$$

Trust interval of the node n in time t_i can be changed only by event $u_i = (n, \delta_i)$ by the following formula with respecting previous node trust (in time t_{i-1}):

$$\rho(n, t_i) = \rho(n, t_{i-1}) \cap \delta_i \tag{18}$$

3.4.2 Algorithm of Bubbling Events

As we mentioned above, HMTC is driven by events where each node can be updated. In following text we recognize two types of events: *external* and *internal* event. External event is update which is done from environment and directly update trust interval of any node of HMTC. This event triggers process called *algorithm of bubbling events* (ABE) which may consequently cause internal events.

```
input  : external event: u_e = (δ_e, t_i) for node a
output: set of internal events generated by u_e: U_i
begin
    ρ(a, t_i) ← ρ(a, t_{i-1}) ∩ δ_e
    if ρ(a, t_i) ≠ ρ(a, t_{i-1}) then
        Open ← ∅
        pushAll (Open, getParents (a))
        pushAll (Open, getChilds (a))
        while Open ≠ ∅ do
            node ← pop (Open)
            δ_i ← eval(node, t_i)
            u_i ← (node, δ_i)
            ρ(node, t_i) ← ρ(node, t_{i-1}) ∩ δ_i
            if ρ(node, t_i) ≠ ρ(node, t_{i-1}) then
                U_i ← U_i ∪ {u_i}
                Parents ← getParents (node)
                while Parents ≠ ∅ do
                    p ← pop (Parents)
                    if p ≠ a and p ∉ Open then
                        push (Open, p)
                Children ← ∅
                Children ← getChilds (c)
                while Children ≠ ∅ do
                    ch ← pop (Children)
                    if ch ≠ a and ch ∉ Open then
                        push (Open, ch)
```

Algorithm 1: Algorithm of Bubbling Events – ABE

Algorithm ABE presented in 1 is demonstrated on an example, when a node a in a time moment t_i is updated by event $u_e = (\delta_e, t_i)$. Algorithm starts with adding every parent and child nodes of a into the *Open* queue. To use a queue ensures that the parent nodes are re-computed first in the next iteration and the children nodes are re-computed later. The algorithm works in iterations while the *Open* queue is not empty. For each `pop`ed node new trust interval is re-computed with using function `eval` (see equation (14)).

Naturally one node may be re-computed and/or updated more than once during the algorithm run. This happens when it is added again to the *Open* queue after a previous re-computation of the some adjacent node.

3.5 Conflict in computation

External event may cause situation, where a node trust is assumed to an *empty set*. The source of this problem comes from intersection operator, which can produce empty set from two different intervals. However it is a valid assumption because empty set is a valid element of τ. This trust assumption to an empty set indicated that current knowledge about a context is *in conflict* with new incoming event – knowledge. Let's assume that trust of node a in time t_{i-1} is set out by $\rho(a, t_{i-1}) = [0.2, 0.5]$ and in time t_i comes external event $u_e = (a, \delta_e)$ where $\delta_e = [0.6, 0.8]$. With using application of event equation (18) we get:

$$\begin{aligned} \rho(n, t_i) &= \rho(n, t_{i-1}) \cap \delta_e \\ &= [0.2, 0.5] \cap [0.6, 0.8] \\ &= \emptyset \end{aligned} \tag{19}$$

Conflicts are caused when two different trust intervals are intersected and the result is an empty set. As we said before, conflicts indicate that two different information about one context cannot be combined together. In this section we propose some solutions how to solve conflicts with using *events* and *paths* which will define an *extended HMTC*.

3.5.1 Path of events

The ABE algorithm describes how an external event causes some internal events and how these events are generated. In fact, the ABE algorithm generates *sequences of events*, where the first event of this sequence is an external event and rests of a sequence are internal events. Our conflict avoiding solution is based on removing events which are responsible that conflict arise. Toward this we to define the sequence of events caused by some external event.

We define *path* p as a finite sequence of events $u_0 u_1 u_2 \ldots u_{k-1} u_k$, where each term of sequence is an event from the set of all events U. We denote the path from u_0 to u_k as (u_0, u_k). Event u_0 denotes *beginning of the path* and is always an external event. Event u_k then denotes *end of the path*. All events from the path except the start event are internal events – these events are also denoted as *rest of the path*. Set of all paths in the actual configuration of the model denoted as P.

Toward to the path definition, we are able to make explicit correlation between *events* and *time moments* (set denoted as T in definition 1). Our model is discrete system and T is a discrete time moments ordered set. Thus, for any time moment from T, we are able to determine *current configuration* of the model. The *configuration* of the model can be changed only by event, respectively by the set of events or even better, by the sequence of events. All causal events (*rest of the path*) must be atomically evaluated in the case, when an external event (*begin of the path*) occur. This atomic evaluation is labeled with new time moment from T. We notice that each path from P is associated with exactly one time from T. The atomicity evaluation of the path requires that all other external events (if some occurs concurrently) must wait in the events queue to be processed.

Finally, *extended HMTC* (eHMTC) is a basic HMTC extended by two following components: U as a set of events and P as a set of paths:

Definition 2. Extended Hierarchical Model of Trust in Contexts is an n-tuple $eHMTC = (N, E, T, U, P, \rho, w)$, where:

- U as a finite set of all applied events,
- P as a finite set of all paths,
- rest of the components – N, E, T, ρ and w – are defined as we introduced in definition 1.

3.5.2 Chaining of events

Trust of the node can be changed only by an event with using intersection in correspond to equation (18). Toward this equation and commutative property of intersection operation we can say that the order of application of events for a single node does not matter. Thus for instance, when we have a node n and events u_1, u_2 and u_3 which are mapped to this node n we can simply change the order of their applications.

For instance, if we have four events $u_0 = [n, \delta_0]$, $u_1 = [n, \delta_1]$, $u_2 = [n, \delta_2]$, $u_3 = [n, \delta_3]$ for node n which occurred in time moments t_0, t_1, t_2, t_3 where $t_0 < t_1 < t_2 < t_3$ (event u_0 precedes to event u_1, and so on) - the trust for node n compute in t_3 is computed as:

$$\begin{aligned} \rho(n,t_0) &= \rho(n,t_0-1) \cap \delta_0 \\ \rho(n,t_1) &= \rho(n,t_0) \cap \delta_1 \\ \rho(n,t_2) &= \rho(n,t_1) \cap \delta_2 \\ \rho(n,t_3) &= \rho(n,t_2) \cap \delta_3 \end{aligned} \tag{20}$$

thus

$$\begin{aligned} \rho(n,t_3) &= \rho(n,t_0-1) \cap \delta_0 \cap \delta_1 \cap \delta_2 \cap \delta_3 \\ &= \rho(n,t_0-1) \cap \delta_3 \cap \delta_2 \cap \delta_1 \cap \delta_0 \end{aligned} \tag{21}$$

With respect to the above, the path should be also evaluated irrespective to time, when the events in a path arise. On the bases of this fact, we propose conflicts avoiding solutions in next section.

3.5.3 Conflict avoiding solution

Conflicts are caused when two different trust intervals are intersected and the result is an empty set. It may happen at the beginning of the *path* (by intersection of external events). Toward this, we propose solutions how to solve conflicts with using *events* and *paths* in eHMTC.

There are always at least two events which are responsible for the conflict because intersection is a binary operator. First of them is always the event on which the conflict is detected. Next possible conflicted events should be determined by using intersection operation together with first conflicted event. We recognize two types of conflicts: *single conflict* when only two events are in conflict and *multi conflict* when more the two events are in conflict. In this section we present solution for solving *single conflict* and *multi conflict* is handled with similar manner.

Let's consider that u_a is a first conflict event and u_b is a second conflict event. If we would like to withdraw one of these events we have to also withdraw all relevant events – respectively all *relevant paths* for the event.

Relevant paths to the event u_a are all paths, for which beginning of the path is same as beginning of the path where u_a was occurred. Let p_a be a path, where event u_a occurred, thus we have an event u_{a_0} (possibly $u_a = u_{a_0}$) which is begin of the path p_a. The set of paths $P_{a_0} \subset P$ is a set, where each path begins with event u_{a_0}.

$$\begin{aligned} p_a &= u_{a_0} u_{a_1} \dots u_a \dots u_{a_{i-1}} u_{a_i} \\ \forall p \in P : begin(p) = u_{a_0} &\Rightarrow p \in P_{a_0} \end{aligned} \tag{22}$$

Removal of all events, respectively all paths from P_{a_0} can solve the conflict and we call it as *events rollback*. As we can see, the events rollback is possible because time when events are occurred is irrelevant (described in section 3.5.2) and we are able to remove any event irrespective to their application order to the node.

Single event removing should be demonstrated on example which is presented in section 3.5.2. We have four events (u_0, ..., u_3) mapped to the node n. Trust of the node n when all events are applied should be computed as:

$$\rho(n,t_3) = \rho(n,t_0-1) \cap \delta_0 \cap \delta_1 \cap \delta_2 \cap \delta_3 \tag{23}$$

If we would like to remove event u_2, we have just to recompute node n trust by modified equation above like this:

$$\rho(n,t_3) = \rho(n,t_0-1) \cap \delta_0 \cap \delta_1 \cap \delta_3 \tag{24}$$

Algorithm 2 shows a simple example how to remove a set of relevant paths P_{a_0}. Function `explodePath` transform a path (sequence of events) to a set of events:

$$\begin{aligned} p_a &= u_{a_0} u_{a_1} \dots u_{a_{i-1}} u_{a_i} \\ explodePath(p_a) &= \{u_{a_0}, u_{a_1}, \dots, u_{a_{i-1}}, u_{a_i}\} \end{aligned} \tag{25}$$

```
input: relevant paths: P_a0
begin
    U_open ← ∅
    U_closed ← ∅
    foreach p ∈ P_a0 do
        U_open ← U_open ∪ explodePath (p)
        foreach u ∈ U_open do
            if U_closed ∩ {u} = ∅ then
                removeEvent (u)
                U_closed ← U_closed ∪ {u}
```

Algorithm 2: Removing relevant paths

3.5.4 Event weight

Decision about removing conflicted event is based on the *weight of event*. For this purposes we extended an event formal definition by new component, which defines a weight of the event. Thus, event is now defined as a tuple:

$$u = (n, \delta, \omega) \tag{26}$$

where ω is weight of event u and $\omega \in [0, 1]$. Event weight is evaluated on the bases on the source of external event. When event is based on direct interaction, weight takes value 1. In case, when source of external event is based on recommendation, weight computed by using function *trans* defined as follow:

$$trans(\vartheta, r) = x + (y - x) * r \tag{27}$$

where $\vartheta = [x, y]$ is trust interval and $r \in [0, 1]$ is subjective *optimism* parameter of evaluated agent. Weight of event based on recommendation from agent B to agent A takes value $trans(\mathcal{T}_{A,B}, r_A)$, where $\mathcal{T}_{A,B}$ is a trust of agent B from A point of view and r_A is subjective optimism of A agent.

4. Trust evaluation for HMTC

For our trust evaluation proposal, we assume that the trust interval is a qualitative estimation of agent reliability in such context. The limits of the trust interval are used to determine probability density function (PDF) of normal distribution. Normal distribution, which is typically denoted as $\mathcal{N}(\mu, \sigma^2)$ or $\mathcal{N}(\mu, \sigma)$ (where μ denote *mean*, σ^2 denote *variance* and σ denote *standard deviation*), is also used as model of agent behaviour. Relation between trust interval, normal distribution and agent behaviour model will be described in next subsection.

4.1 Model of Agent Behaviour

First of all we define model of agent behaviour. Model of agent's behaviour express agent quality and reliability in all contexts and is expressed by normal (or Gaussian) distribution. It means, that agent ability to behave in such context is expressed by quality which is determined by *mean* and *variance*. The mean can be understood as agent *typical behavior* and variance is used to express oscillations around this behaviour. For describing agent behaviour we use notation $\mathcal{N}(\mu, \sigma)$ and equivalent notation with using *behaviour interval* $[\mu - 3\sigma, \mu + 3\sigma]$, where following three conditions must be valid:

$$\begin{array}{ll} 1. & 0 \leq \mu \leq 1 \ , \\ 2. & 0 \leq (\mu - 3\sigma) \ , \\ 3. & (\mu + 3\sigma) \leq 1 \ . \end{array} \tag{28}$$

The interval representation of behaviour comes from empirical *three-sigma rule* [4] which states that for a normal distribution, nearly all (99.73%) of the values lie within 3 standard deviations of the mean. By the conditions in (28) we ensure, that behaviour interval have same limits as trust interval.

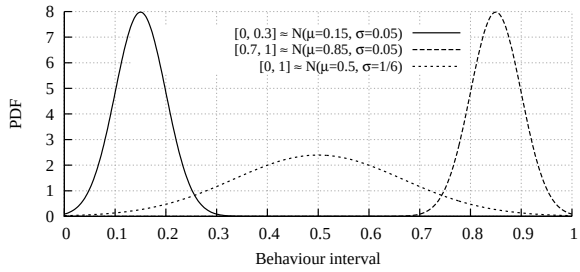

Figure 3: Examples of model behaviour.

Three examples of agent behaviour model for such context is illustrated on Figure 3. As we can see, models are denoted with both representation – notation for normal distribution and for behaviour interval. In our framework we use this behaviour model to generate an random number with given PDF at the time, when an interaction from agent is required.

Trust in such context, respectively trust interval, is an another agent estimation about agent behaviour model, based on direct experiences. In the respect this, trust estimation can be generalized to normal distribution parameters estimation (μ and σ) from a random sample (outcomes from interactions). In general, we recognize two types of estimations: *point estimation* and *interval estimation*.

4.2 Estimation Trust Interval with Using Confidence Interval

An *estimator* or *point estimate* is a statistic that is used to infer the single value of an unknown population parameter. However, it is important to understand how good is the estimate obtained. The point estimate says nothing about how close $\hat{\mu}$ is to μ (when μ denote real mean of the statistical model of normal distribution and $\hat{\mu}$ denote its estimation). Toward this, we use an *interval estimation* which uses point estimate to calculate an interval of possible values of an unknown statistical population (or statistical sample) parameter. An interval estimate for a population parameter is called a *confidence interval* [12].

4.2.1 Point Estimate for Mean and Variance

As we noted above, point estimate for mean μ is denoted as $\hat{\mu}$ and analogically, point estimate for variance σ^2 is denoted as $\hat{\sigma^2}$. Estimator $\hat{\sigma^2}$ is also known as *biased sample variance* and for variance estimate we often use *unbiased sample variance* denoted as s^2 rather then biased one (σ^2). Estimate for mean from population in normal distribution is just simple arithmetic mean:

$$\hat{\mu} = \overline{x} = \frac{1}{n} \sum_{i=1}^{n} x_i \ , \tag{29}$$

where n is population size. Biased estimate for sample variance in normal distribution is calculated as:

$$\hat{\sigma}^2 = \frac{1}{n} \sum_{i=1}^{n} (x_i - \overline{x})^2 \tag{30}$$

and unbiased sample variance:

$$s^2 = \frac{1}{n-1} \sum_{i=1}^{n} (x_i - \overline{x})^2 \ . \tag{31}$$

4.2.2 Confidence Interval

A confidence interval estimate for μ is an interval of the form $\mu_{min} \leq \mu \leq \mu_{max}$ where the endpoints μ_{min} and μ_{max} are computed from the sample data. For these interval limits, following probability statement is true: $P(\mu_{min} \leq \mu \leq \mu_{max}) = 1 - \alpha$ where $0 \leq \alpha \leq 1$.

The end-points or bounds μ_{min} and μ_{max} are called the **lower-** and **upper-confidence** limits, respectively, and $1 - \alpha$ is called the *confidence coefficient* [12]. These limits are calculated, for a normal distribution with unknown variance, as is described in formulae (32). Also, confidence interval for sample mean is typically denoted as:

$$\underbrace{\overline{x} - t_{\alpha/2}(n-1)\frac{s}{\sqrt{n}}}_{\mu_{min}} \leq \mu \leq \underbrace{\overline{x} + t_{\alpha/2}(n-1)\frac{s}{\sqrt{n}}}_{\mu_{max}} \tag{32}$$

where n is the sample size, $t_{\alpha/2}(n-1)$ is $100(1 - \alpha/2)$ quantile of *Student distribution* (t-distribution) with $n-1$ degrees of freedom. Confidence coefficient $1 - \alpha$ is typically selected near the 1 and is denoted with using percentage points: 90% (for $\alpha = 0.1$), 95% (for $\alpha = 0.05$) and 99% (for $\alpha = 0.01$).

A confidence interval estimate for unknown variance σ^2 is evaluated in similar manner with using sample variance point estimator s^2 and where lower (σ^2_{min}) and upper (σ^2_{max}) confidence limits are calculated with using *Chi-square distribution* by following formulae:

$$\frac{(n-1)s^2}{\chi^2_{\alpha/2}(n-1)} \leq \sigma^2 \leq \frac{(n-1)s^2}{\chi^2_{1-\alpha/2}(n-1)} \tag{33}$$

where $\chi^2_{1-\alpha/2}(n-1)$ and $\chi^2_{\alpha/2}(n-1)$ are the upper and lower $100\alpha/2$ percentage points of the Chi-square distribution with $n-1$ degrees of freedom, respectively [12].

4.2.3 Trust Interval

As we say before, trust in such context, respectively trust interval, is an agent estimation about another agent behaviour model. Thus, we are able to estimate trust, with specified probability, from a sample of direct interaction between agents by using confidence interval. The probability that our estimated trust equals an agent model behaviour is specified by confidence coefficient.

Estimated trust interval $[x, y]$ by the agent A in context c about and agent B after their mutual n interactions, when B model behaviour in context c equals $\mathcal{N}(\mu, \sigma)$ can be stated as:

$$\begin{aligned} x &= \mu_{min} - 3\sigma_{max} \\ y &= \mu_{max} + 3\sigma_{max} \ , \end{aligned} \tag{34}$$

where μ_{min}, μ_{max} are limits of confidence interval in (32) and σ_{max} is square root of the corresponding limit of confidence interval in (33).

5. Experimental results

The presented experiments were primarily focused on the comparison of single- and multi-contextual approach. In our scenario, trust is used primary to decision making about cooperation partner. Trust is estimated from the direct cooperation between agents with using confidence intervals described above. Agents in this scenario provide some services which correspond to different contexts. Behaviour model for each agent and context was specified. Primary observed parameter is an agent number of usage which is very closely related to its trustworthy from other agent's point of view. Big amount number of usage implies that agent is trustworthy in such context, because many agents in system use her/him.

5.1 Experimental scenario

5.1.1 Basic Scenarios

- **First „single“ scenario**. In the first scenario, we use only *single-context update* trust model, where all interaction results (irrespective to the interaction context) was aggregated into one trust value (trust interval) for each agent.

- **Second „multi“ scenario**. In the second scenario, *multi-context update* trust model was used. For each interaction result the interaction context was determined and only trust in this context was updated.

5.1.2 Common Parameters

For all executed experiments, we use multi-agent environment with ten agents. These agents was named *Agent1,...*, *Agent10*. Each agent provide to all other agents two different services (corresponding to two different *contexts*) denoted as „context 1“ and „context 2“. The agents behaviour models was same for all experiments and scenarios and is described in Table 5. The HMTC model structure was set same for all agents and is illustrated in Figure 4.

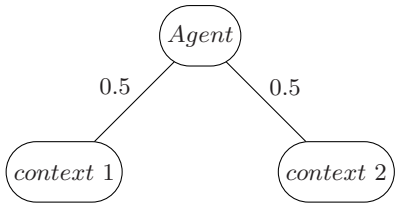

Figure 4: HMTC graph structure for multi-context scenario.

As you can see in Table 5, agents can be split into three different groups by their behaviour model in such contexts:

1. *Good agents* (Agent1, Anget2): have same „good“ behaviour model in both contexts defined as $\mathcal{N}(\mu = 0.85, \sigma = 0.05)$ which equals to behaviour interval $[0.7, 1.0]$.
2. *Half good/bad agents* (Agent3, Agent4): have one context in „good“ behaviour – $\mathcal{N}(0.85, 0.05)$ and second in „bad“ behaviour – $\mathcal{N}(0.15, 0.05)$ (equal to behaviour interval $[0, 0.3]$).
3. *Bad agents* (Agent5,...,Agent10): have same „bad“ behaviour model in both contexts – $\mathcal{N}(0.15, 0.05)$.

5.1.3 Interaction cycle

A one hundred *interaction cycles* was done in each experiment. In each interaction cycle all agents in system randomly generates required service (context) and with using decision making mechanism form DMRP selects, on the bases on trust, best interaction partner for such context. Toward this, in each *interaction cycle*, 10 interaction are performed. Summary after 100 interaction (one experiment) cycles a total of 1000 interactions are performed. In each interaction cycle we observe these data: cycle, source, context, target, quality. *Cycle* is a interaction cycle number, *source* is the name of the agent which initiated the interaction, *context* is a interaction context, *target* is agent selected for interaction and the quality is outcome of interaction in such context.

Each experiment for each scenario was repeated for 10 times for makes an experiment more efficient and helps

	Model behaviour	
Agent	context 1	context 2
Agent1	N(0.85, 0.05)	N(0.85, 0.05)
Agent2	N(0.85, 0.05)	N(0.85, 0.05)
Agent3	N(0.15, 0.05)	N(0.85, 0.05)
Agent4	N(0.85, 0.05)	N(0.15, 0.05)
Agent5	N(0.15, 0.05)	N(0.15, 0.05)
⋮	⋮	⋮
Agent10	N(0.15, 0.05)	N(0.15, 0.05)

Figure 5: Model of agents behaviour.

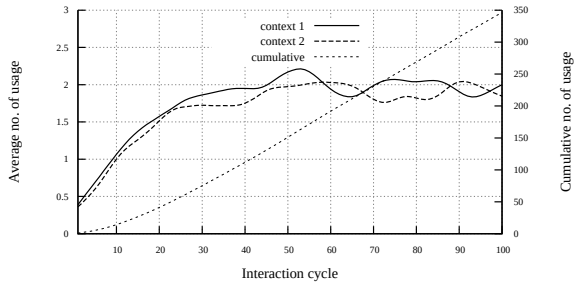

Figure 6: Single-context environment, usage of Agent 1.

keep the variability low. The presented graphs show approximated (with using *cubic spline*) arithmetic mean values from all experiments in each scenario.

5.2 Results and Discussion

The first result, presented by graph in Figure 6 and Figure 7, shows the situation, where number of usage of *Agent*1 in each interaction cycles is observed. *Agent*1 have „good“ behaviour model in both contexts. In first single-context case (Figure 6), only one trust value is used for *Agent*1 and its both contexts. After 30 interaction cycles, *Agent*1 is used approximately two times for each context and each cycle, the overall sum of usage after 100 cycles is nearly 350 (shown on the axis Y2).

Opposite this, in Figure 7 shows situation where trust value is updated separately for each context. Number of usage increase more slowly then in single-context case and overall sum of usage after 100 cycles is 264.

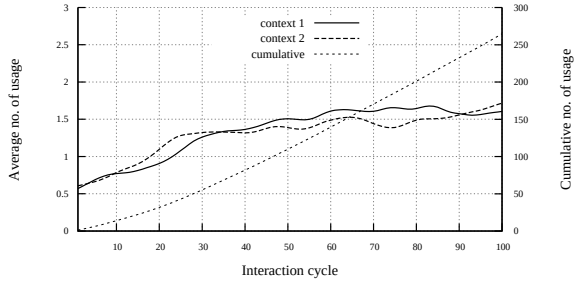

Figure 7: Multi-context environment, usage of Agent 1.

When we compare these two different approaches, it's seems that the better results give us single-context case (usage decrease by -25% points). The quality of the agent is estimated faster and have more amount of usage. This is primarily due to the fact that the interval estimate are more accurate, when the larger number of interaction in such context is performed.

More interesting result bring comparison of „half good/bad“ agents. We observer an usage of *Agent*3, which have „good“ behaviour in „context 2“ a „bad“ behaviour in „context 1“. We can see in Figure 8 that single-context update is not able to recognize which context is „good“ and which is „bad“ and the total number of usage gain only 92.

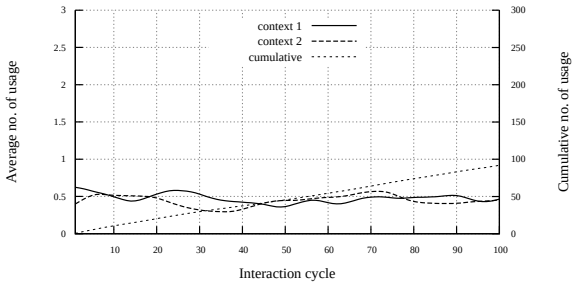

Figure 8: Single-context environment, usage of Agent 3

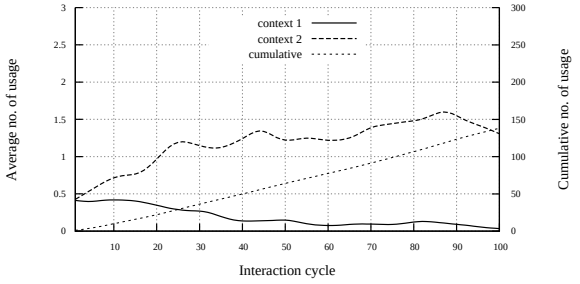

Figure 9: Multi-context environment, usage of Agent 3

In the mutli-context case (Figure 9) the situation is quite different and trustworthy of agent in different contexts is very quickly recognized and is used mainly in „good“ context. Overall number of usage is 138, which is quite better (usage increase by $+50\%$ points) than in single-context case.

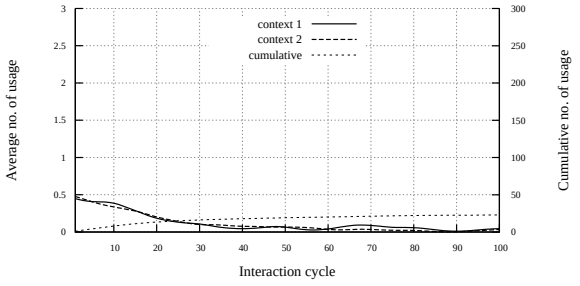

Figure 10: Single-context environment, usage of Agent 5

In next case, we compare difference between single- and multi-context scenario of „bad“ behaviour agent, especially Agent5. Graphs (10 and 11) shows, that number of usage in comparison single-context and multi-context update is very similar and their un-trustworthy is very quickly estimated by other agents in booth cases.

6. Conclusion

The area of trust and reputation modelling is a multidisciplinary field of research and this phenomena took place in many areas of our everyday life. This paper is focused on modelling trust with multi-contextual approach and target area of our research is an artificial intelligence, espe-

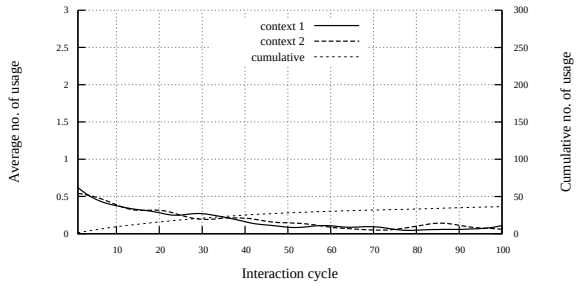

Figure 11: Multi-context environment, usage of Agent 5

cially distributed agent and multi-agent systems. Multi-contextual trust modelling means, that we are able to provide trust level for the various aspects of one entity which is subjectively judged from different point of views

We presented a new distributed trust model, called *Hierarchical Model of Trust in Contexts* (HMTC), which is based on multi-contextual trust where different aspects of single entity are always part of the whole and cannot be considered separately. Toward this assumptions, we propose hierarchical structure of contexts with dependency and principle of trust inferring from one context to another.

A unique representation of trust with using interval has been proposed. This interval is able to represent not only trustworthy, but also an uncertainty which is always associated with trust assessment. The trust estimation method from direct interactions between agents is based on statistical methods, especially on the confidence interval and this estimated trust value is aggregated for specified contexts with using HMTC.

Experimental results shown, that in the simple cases where agent's behaviour in all its interaction contexts is same, than the single-context trust evaluation has slightly better results than in multi-context approach. This is due the fact, that interval estimation have low precision in case of a small number of interactions. But in the more complex, sophisticated and realistic cases, where agents have different quality and capability in different interaction contexts, multi-context trust approach is clearly better.

6.1 Future work

In our future work we will focus on two marginal targets. Firstly we need to integrate recommendation and reputation management into current proposal and verify the assumption, that we are able to increase effectiveness in agent decision making with using recommendations. Next, we need to provide methodology for definition and validating established HMTC structures. This methodology should reflect possible dynamic topology when different connection between context could be updated, removed or newly added.

Acknowledgements. This work was partially supported by the grants GACR 102/09/H042,BUT FIT-S-11-1 and the research plan MSM0021630528.

References

[1] A. Abdul-Rahman and S. Hailes. Using recommendations for managing trust in distributed systems. In *IEEE Malaysia International Conference on Communication '97 (MICC'97)*, 1997.

[2] A. Abdul-Rahman and S. Hailes. Supporting trust in virtual communities. *Hawaii International Conference on System Sciences*, 6:6007, 2000.

[3] C. Castelfranchi and R. Falcone. Principles of trust for MAS: Cognitive anatomy, social importance, and quantification. In *Proceedings of the 3rd International Conference on Multi Agent Systems*, ICMAS '98, pages 72–79. IEEE Computer Society, 1998.

[4] S.-H. Dai and M. Wang. *Reliability Analysis in Engineering Applications*. Van Nostrand Reinhold, 1992.

[5] D. Gambetta. Can we trust trust? In D. Gambetta, editor, *Trust: Making and Breaking Cooperative Relatioins*, chapter 13, pages 213–237. Published Online, 2000.

[6] E. L. Gray. *A Trust-Based Reputation Management System*. PhD thesis, Trinity College, University of Dublin, April 2006.

[7] A. Jøsang, C. Keser, and T. Dimitrakos. Can We Manage Trust? In *Proceedings of the Third International Conference on Trust Management (iTrust), Versailes*, pages 93–107. Springer-Verlag, 2005.

[8] M. Kinateder and K. Rothermel. Architecture and algorithms for a distributed reputation system. In *Proceedings of the 1st international conference on Trust management (iTrust 2003)*, pages 1–16. Springer-Verlag, 2003.

[9] F. G. Mármol and G. M. Pérez. Providing trust in wireless sensor networks using a bio-inspired technic. In *Networking and Electronic Commerce Research Conference (NAEC 08)*, 2008.

[10] F. G. Mármol and G. M. Pérez. Towards pre-standardization of trust and reputation models for distributed and heterogeneous systems. *Computer Standards & Interfaces*, 32(4):185–196, 2010.

[11] S. P. Marsh. *Formalising Trust as a Computational Concept*. PhD thesis, University of Stirling, April 1994.

[12] D. C. Montgomery and G. C. Runger. *Applied Statistics and Probability for Engineers*. John Wiley & Sons, 3 edition, 2003.

[13] L. Mui. *Computational Models of Trust and Reputation: Agents, Evolutionary Games, and Social Networks*. PhD thesis, Massachusetts Institute of Technology, 2003.

[14] L. Mui, M. Mohtashemi, and A. Halberstadt. A computation model of trust and reputation. In *Proceedings of the 35th Annual Hawaii International Conference on System Sciences (HICSS'02)*, volume 7, page 188, 2002.

[15] P. Resnick and R. Zeckhauser. Trust among strangers in internet transactions: Empirical analysis of ebays reputation system. In *NBER Workshop on Emperical Studies of Electronic Commerce*, 2000.

[16] J. Sabater and C. Sierra. REGRET: reputation in gregarious societies. In *Proceedings of the fifth international conference on Autonomous agents*, pages 194–195, 2001.

[17] J. Sabater and C. Sierra. Review on computational trust and reputation models. *Artificial Intelligence Review*, 24(1):33–60, 2005.

[18] J. Samek, O. Malačka, F. Zbořil, and P. Hanáček. Multi-agent experimental framework with hierarchical model of trust in contexts for decision making. In *Proceeding of the 2nd International Conference on Computer Modelling and Simulation*, pages 128–136. Department of Intelligent Systems FIT BUT, 2011.

[19] J. Samek and F. Zbořil. Algorithmic evaluation of trust in multilevel model. In *EUROSIM 2010 - 7th EUROSIM Congeres on Modelling and Simulation*. Czech Technical University in Prague, 2010.

[20] J. Samek and F. Zbořil. Hierarchical model of trust in contexts. In *Networked Digital Technologies*, volume 88 of *Communications in Computer and Information Science (CCIS)*, pages 356–365. Springer Verlag, 2010.

[21] S. Sen and N. Sajja. Robustness of reputation-based trust: boolean case. In *AAMAS '02: Proceedings of the first international joint conference on Autonomous agents and multiagent systems*, pages 288–293, 2002.

[22] C. Weifang, L. Xiangke, S. Changxiang, L. Shanshan, and P. Shaoliang. *A Trust-Based Routing Framework in Energy-Constrained Wireless Sensor Networks*, volume 4138/2006. Springer Berlin / Heidelberg, 2006.

[23] Wikipedia. Interval arithmetic – Wikipedia, The Free Encyclopedia, 2010. [Online; cit. 5.12.2010].

Selected Papers by the Author

J. Samek, O. Malačka, F. Zbořil, P. Hanáček. Multi-Agent Experimental Framework with Hierarchical Model of Trust in Contexts for Decision Making. In *Proceeding of the 2nd International Conference on Computer Modelling and Simulation*, pages 128–136, Brno, Czech Republic, 2011. Department of Intelligent Systems FIT BUT.

O. Malačka and J. Samek and F. Zbořil and F. V. Zbořil. Decision Making and Recommendation Protocol Based on Trust for Multi-Agent Systems. In *Proceedings of Trust, Reputation and User Modeling Workshop (TRUM 2011)*, pages 33–40, Girona, Spain, 2011.

J. Samek and O. Malačka and F. Zbořil. Event Driven Multi-Context Trust Model. In *10th International Conference on Intelligent Systems Design and Applications (ISDA 2010)*, pages 911–917, Cairo, EG, 2010. IEEE Computer Society.

J. Samek and F. Zbořil. Hierarchical Model of Trust in Contexts. In *Networked Digital Technologies*, Communications in Computer and Information Science, volume 88, pages 356–365. Springer, 2010.

J. Samek and F. Zbořil. Algorithmic Evaluation of Trust in Multilevel Model. *EUROSIM 2010 - 7th EUROSIM Congeres on Modelling and Simulation*, Prague, Czech Republic, 2010. Czech Technical University in Prague.

O. Malačka and J. Samek and F. Zbořil. Increasing Profit in Agent Business Model with Trust. *EUROSIM 2010 - 7th EUROSIM Congeres on Modelling and Simulation*, Prague, Czech Republic, 2010. Czech Technical University in Prague.

J. Samek and F. Zbořil. ContextGraph: Simulation Tool for Hierarchical Model of Trust in Context. In *Proceedings of CSE 2010 International Scientific Conference on Computer Science and Engineering*, pages 265–270, Košice, SK, 2010. The University of Technology Košice.

J. Samek and F. Zbořil. Agent Reasoning Based On Trust And Reputation. In *Proceedings MATHMOD 09 Vienna - Full Papers CD Volume*, pages 538–544, Vienna, AT, 2009. ARGE Simulation News.

J. Samek and F. Zbořil. Multiple Context Model for Trust and Reputation Evaluating in Multi-Agent Systems. In *Proceedings of CSE 2008 International Scientific Conference on Computer Science and Engineering*, pages 336–343, Košice, SK, 2008. The University of Technology Košice.

J. Samek. A Trust-Based Model for Multi-agent Systems. In *Proceedings of the 6th EUROSIM Congress on Modelling and Simulation*, 6 pages, Viena, SI, ARGESIM, 2007.