Dance One Scenario for Choreographic Coordination of Intelligent Agents in Entertainment Applications

Rodrigo B. V. Lima Patrícia A. Restelli Tedesco Geber L. Ramalho

Federal University of Pernambuco, Informatics Center, Brazil

Abstract

Most researches in agent coordination for computer games focus on tactical/spatial coordination of NPCs (Non Playable Characters) seeking effective methods for accomplishing a collective task in the context of war games, FPS (First Person Shooters) and RTS (Real Time Strategy) (e.g., [Almeida et al. 2004; Madeira et al. 2005; Marthi et al. 2005]). Some research efforts have been employed to make the movements of group of agents more realistic and pleasant [Messick 1999; Reynolds 1999]. However, the aesthetic aspect plays a secondary role in the complexity of the task the group of agents must accomplish. This work represents a step forward towards a deeper concern on the aesthetic/choreographic dimension that may arise from the NPCs interactions. Based on a framework for supporting multi-agent choreographic coordination [Lima et al. 2005], we present an original 2D entertainment application, named DiscoTech, where agents reason, communicate and coordinate with each other in order to form groups and perform choreographies.

Keywords: Group behavior, choreography coordination, dance and music, believable characters

Authors' contact:
rbvl@cesar.org.br
{pcart, glr}@cin.ufpe.br

1. Introduction

It is noticeable that, during the last few years, some new and interesting concepts have emerged out of the game industry as a means to diversify the market, and also as an attempt to encourage more and more people to join the game world. Some of these new ideas are specially focused on coupling games and musical elements in some quite unique ways such as "musical shooters", musical composition games, choreography based games, etc.

One good example of such kind of game is *Rez*[1], developed by United Artists/ SEGA. "The objective of *Rez* is to appreciate the aesthetic experience, of which destruction is only a part, albeit an important one" [Smith 2002]. In fact, the experience of playing this game goes far beyond a standard shooter, since the player dynamically "composes" a sound track while shooting objects.

Another game that falls into this category is *Lumines*[2], from Q? Entertainment. *Lumines* is a puzzle game that resembles the classic Tetris in many aspects. However, it adds the music element as an important part of the experience: the blocks created by the player will only be cleared out when a musical staff bar that is synchronized with the music sweeps across the screen.

The previous two games are just a small example of what is out there. There are many other games that also follow this idea of art games, such as *Guitar Hero*[3], *Electroplankton*[4], *Dance Dance Revolution*[5], *Donkey Konga*[6], etc, proving that the industry is craving for new ideas and forms of entertainment, and that musical related applications seem to be a good spot to place our bets on.

These kinds of games are certainly helping to define new and different forms of entertainment, making their way to establish a genre we call "Art Games": games where the player's main objective is not only related to overcoming a given obstacle or enemy (following the play-and-defeat-boss standard), he or she must also to try to create somewhat of a beautiful, pleasant and enjoyable environment, be it by means of graphical elements, sound effects or a combination of both. The key idea behind this concept is that particular art related elements, such as visual and musical experiences, play a role as important as, for instance, the game's main objective. Although different in essence, all applications mentioned before share one property: the desire to explore the musical world in computer games.

Our main goal with the present work is to attempt to widen the set of possible outcomes that might be generated by merging computer games with musical elements. We will ground our work on the idea of intelligent agents capable of reasoning about the execution of dance movements by a group of individuals; something we have named *Choreographic Coordination*.

In the remaining of this paper, we are going to discuss a little further the *Choreographic Coordination* concept, and apply the result obtained to build an application called *DiscoTech*. This application was

[1] http://ps2.ign.com/articles/166/166546p1.html

[2] http://www.ubi.com/US/Games/Info.aspx?pId=1937
[3] http://www.guitarherogame.com/
[4] http://electroplankton.nintendods.com/flash.html
[5] http://www.konami.com/gs/gameinfo.php?id=24
[6] http://www.donkeykonga.com/dk1.html

developed as a means to experiment our ideas and the underlying concepts. However, it is important to note that *DiscoTech* must not be seen as a final and complete application, but as a representation of a whole new class of applications that might emerge in the near future.

In the next section we will define in more details what *Choreographic Coordination* is all about, its main characteristics, as well as a set of requirements an agent must fulfill in order to be able to properly deal with this concept. Afterwards, we are going to present a brief overview of a framework called DANCE (Dancer AgeNts Computational Environment) [Lima et al. 2005], which was developed as a foundation to implement the *DiscoTech* application, followed by the description of the application itself. In the last section, we will discuss our final conclusions and a brief summary of some interesting future works.

2. Choreographic Coordination and Dancer Agents

Nowadays, most computer games are populated with a great number of NPCs. These computer controlled characters, often modeled as intelligent agents, interact with one another and also with the player, in order to accomplish tasks, creating the illusion of an immersive world. The more realistic these interactions are, the more believable the environment will be.

However, one very important question remains unanswered: how can NPCs interact in a way that they can produce something visually beautiful, such as a dance choreography? By properly answering this question it would be possible to deal with an interesting ensemble of applications: from new game genres, passing by choreographic composition applications, up to social interaction software.

Some research works, on a fine-grained level, have already been conducted in order to make unit movements more realistic and, in a certain sense, more beautiful [Pottiger 1999]. This is typically the case of steering behaviors, such as flocking [Reynolds 1999], which can mimic the collective trajectories of animals, like birds and fishes. However, this technique, yet useful, does not provide enough control to implement dance choreographies. In fact, differently from collective trajectories, choreographies can be seen as harmonious collections of different (in quality and orientation) movements. Thus, to create an interesting results, the dancing units (modeled as agents) must be able to explicitly reason about the group formation process (has this group the ability to execute movements that will fit well with mine?), to communicate their abilities and interests and negotiate with each other (which movements will be executed and when), etc. Such tasks cannot be properly executed by using steering behavior techniques alone.

The tasks mentioned before closely maps what happens in the real world. When inside a group, dancers must take into account what nearby dancers are doing. His/her movements are defined as a function of the movements executed by the other dancers s/he must interact with. Therefore, an extra set of concerns must be considered by the dancer, such as not interfering with other dancers movements and synchronizing his/her own movements with movements from different people, etc.

The main objective of our research is to work with the aesthetical dimension of coordinated movement. The idea is to explore the possibilities of creating autonomous agents capable of reasoning about the choreography/aesthetic coordination aspect of agent interaction, particularly in the dance context. We have focused our efforts on defining and creating dancer agents, which are capable of executing dance choreographies, both alone and in a group.

Our first step was to identify a set of requirements related to the task of executing group choreographies. Once defined, these would allow us to design agents capable of handling these constraints.

Below we present a brief description of the most important requirements a dancer agent must fulfill.

- *Reasoning:* dancer agents should be able to decide what is the best dance step to be executed; if they are better on their own, or if they need to be part of group etc.
- *Coordination:* assuming that agents will be dancing together, they must coordinate their actions so that one agent does not interfere negatively on another agent's actions.
- *Communication:* coordination can only be achieved if there is a mechanism to support it. One of these mechanisms is communication. By communicating, different agents can make their intentions public, coordinate actions, negotiate etc.
- *Synchronization:* music and dance are strongly related mainly because of their temporal nature. It is expected that dance movements be synchronized with a song.
- *Movement mechanisms:* if an agent is supposed to execute dance steps, it must be able to move around its environment. This mechanism shall be as efficient and elegant as possible, providing a natural look to the movement itself.
- *Different set of abilities:* as happens in the real world, each dancer may have its own characteristics and personality traces. These differences make one dancer more or less suitable for one kind of choreography than others.

Based on this set of requirements, a C++ framework for creating applications based on dancer agents was developed. The next section gives only a brief overview of the framework, since the scope of this

paper is mainly on the *DiscoTech* application and on the artificial intelligent aspects that allowed us to implements agents capable of following the idea of choreographic coordination. More specific details about the DANCE framework can be found in [Lima et al. 2005].

3. The DANCE Framework

Given the myriad of possible ways to address our problem, our first efforts were focused on the construction of a general framework to aid developers on the process of creating applications that combine multi-agent systems and dance choreography. The DANCE (Dancer AgeNts Computational Environment) framework is formed by a group of reusable tools, developed in C++/DirectX, from which it is possible to create multi-agent applications that take advantage of the characteristics described before.

The DANCE framework consists of three main modules: a multi-agent communication system, a musical synchronization tool and a movement mechanism module. In this section, we will present a brief description of each one of these modules.

3.1 Multi-Agent Communication Module

The multi-agent communication module consists of a set of classes, which may be specialized and/or used by a developer to define new kinds of agents and create multi-agent applications, as well as an execution environment responsible for controlling the agents life cycle.

Each agent may have one or more associated behaviors. A behavior defines the necessary steps to be taken in order to perform a particular task. There can be, for instance, one behavior to control its movements, another to send messages to other agents and yet another to process incoming messages.

The existence of an execution environment makes it easier for a developer to create MAS applications, since s/he does not need to worry about low level tasks, such as message delivery, agent execution etc, focusing only on the problem itself.

3.2 Musical Synchronization Module

This module of the framework is responsible for analyzing sound information acquired from a given MIDI (Musical Instrument Digital Interface) file [Messick 1997] and propagating the results of this analysis to the other two modules of the framework. Although MIDI does not have the same sound quality found on digital audio, such as wav, mp3, etc, we have opted to work with this format mainly because it is simpler to manipulate, since it does not transmit sound information, only commands.

Our algorithm is based on the tempo as well as other resolution characteristics of the MIDI files, working as a detector of temporal events. Whenever a beat event is detected, the tool informs whoever is registered to it that an event has just occurred, delegating the analysis of such information to the registered entities. For instance, one entity that represents a virtual dancer could update the current image being displayed every time it receives a notification, resulting on an animation that is synchronized to a song being played.

3.3 Movement Module

This module provides a series of mechanisms that allow agents to execute fluid movements. By fluid we mean movements that do not have a much accentuated "robotic" look. To achieve this, we have decided to ground our movement system on steering behaviors techniques [Reynolds 1999]. Such techniques try to give autonomous agents the ability to navigate through their environment in a realistic and improvised way.

This approach is especially important to our work because it makes it possible for a dancer agent to perform dance movements in a natural and fluid way. It also simplifies movement composition operations, since more complex movements are computed as operations over vectors.

4. The DiscoTech Application

We have developed the *DiscoTech* application in a way that allowed us to explore the multi-agent nature of our problem. The idea behind this application is to construct an environment that simulates a "virtual night club" populated by dancers of two kinds: *leaders* and *followers*.

At the beginning of the simulation, common dancer agents (the followers) would wander around the environment dancing only by themselves. However, specific dancer agents (the leaders) have the extra responsibility of trying to form a group to execute a given group choreography. These agents must "invite" and "convince" other agents to be part of the proposed choreography. Once the group is formed, the choreography can be executed.

There are three important elements of the application:

- *Dance Environment*: place where dancer agents will exist. Agents should be able to interact with the environment itself as well as with other agents;
- *Follower Agents*: agents that execute choreographies, individually or in group;
- *Leader Agents*: agents that, besides executing choreographies, also try to convince other agents to be part of group choreographies.

As we have said before, an agent may have one or more behaviors associated to itself at a given moment of its lifecycle. Hence, we have created some choreography coordination oriented behaviors, which are associated to our dancer agents. These behaviors make it possible for agents to execute dance movements based on information coming from a song being executed in a given moment.

In this first version of DiscoTech, dancers are represented as geometric figures (spheres) that change their colors in order to represent different dance steps. We were mainly focused on the architecture and internal processing for how choreographed sequences can be created, that is, in the group formation process itself, as well as the underlying AI concepts involved. We do know that it is important to also look on the aspects of how choreographed sequences should be expressed visually as well. Any AI technique for such a domain should be coupled with a consideration for not only the cognitive processing inside the agents but expressing it visually to the audience watching the performance as well.

Although it might sound a little simplistic at first, the choice of using spheres that change their colors as a response to music stimulus has proved to be an extremely pleasant experience, very well evaluated by everyone who has seen it. However, we have developed the *DiscoTech* application independent enough so that the dancer representation used could be substituted, with not too much effort, by other ones, such as, for instance, dancing humanoid figures. By choosing this approach, we were able to focus a little further on specific multi-agent concepts, ultimately, the main objective of this work.

Figure 1 shows a snapshot of the *DiscoTech* application. As stated before, dancer agents are represented as colored spheres. In this particular snapshot there are dancers in four different colors: green (isolated spheres), red (group 1), yellow (group 2) and blue (group 3). Green agents represent agents that are currently dancing alone in the "night club". There are different tonalities of green; each one reflects a particular dance step. When dancing alone, agents keep changing their tonality based on the current song being played, so that they can "dance" synchronized to it. Once again, the images used to represent these dance steps could be substituted by any other, without any major changes in the application logic. For instance, instead of having spherical dancer agents that change color, we could have a set of sprites representing humanoid agents clapping hands or stepping their feet.

When part of a group, an agent will drop the green color in favor of another one, specific for the group that s/he has just joined. In figure 1 there are three different groups of agents (red, yellow and blue). While part of a group, an agent will synchronize and coordinate its movements not only with the song being played, but also with the movements of the other agents that are part of the group. The group formation process is initiated by *leader agents*, according to the procedures and mechanisms that are going to be explained further on.

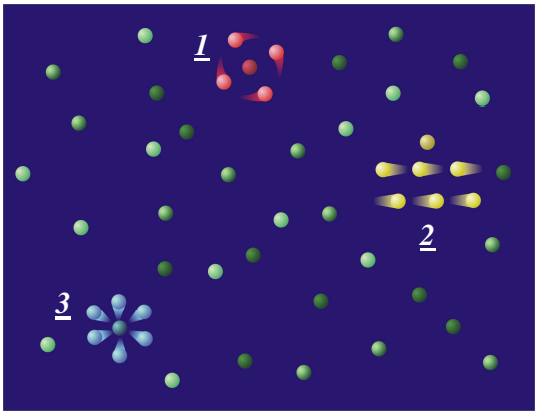

Figure. 1. Snapshot of the DiscoTech application

In the following sections we will describe in more details the strategy used to achieve our objective of developing intelligent agents capable of dancing synchronized to any MIDI file. In the end, the strategy used would be evaluated as effective if we were able to construct an application where: dancer agents could dance synchronized to the song being played (both individually and in group); form groups to execute choreographies; and if they could coordinate movements with other dancer agents.

4.1 Description of Dancer Agents

The application consists of a set of n dancer agents, n = $\{A_1, A_2, \ldots, A_n\}$, which can be either leaders or followers. All agents have communication and negotiation capabilities and can execute dance movements. The basic difference between these two kinds of agents is that leader agents are responsible for initiating the negotiation process in an attempt to form a group. This process is governed by a negotiation protocol that will be described throughout the next section.

The main role of a leader agent is to inform other agents about the presence of a choreography. It must gather a certain number of follower agents (this number is choreography dependent) that are willing to be part of a group, and also, that have enough capabilities to execute the proposed choreography.

A group choreography can be seen as a task to be executed, but for which it is necessary to form a group, since one agent alone does not have enough capabilities to execute it. Hence, it is necessary to form a coalition between agents [Vijsel and Anderson 2004], [Klunsch and Gerber 2002], [Gilpin and Sandholm 2006]. Once this coalition is formed, the group choreography can be executed.

Based on the ideas presented in [Shehory and Kraus 1998], each choreography (or task) is described by a capabilities vector (as shown in equation 1). For our particular case, the capabilities that describe a group choreography are:

- Number of necessary agents (*num_agents*);
- Level of difficulty (*chor_diff*);
- Duration in cycles (*chor_dur*).

All these capabilities are represented by a positive and constant real number and they are used to decide whether or not a given agent can be part of a coalition.

$$C_c = \{number\ of\ agents,\ difficulty,\ duration\}. \quad (1)$$

Similarly, follower agents are also described by a capabilities vector (as shown in equation 2). Each capability is represented by a non-negative real number and can be either or non-consumable [Shehory and Kraus 1998]. The capabilities are:

- Agent's ability (*ab*), which is non-consumable;
- Agent's current level of enthusiasm (*enth*), which is non-consumable;
- Stamina (*st*), which is consumable.

As its name says, the ability of an agent indicates how talented it is. The higher its value, the more skilled the agent will be, therefore, being able to be part of more difficult choreographies.

$$C_a = \{ability,\ enthusiasm,\ stamina\}. \quad (2)$$

The enthusiasm indicates the wiliness an agent has to be part of a group. The higher its value, the easier an agent will accept the invitation to join a group. These two capabilities (ability and enthusiasm) are non-consumable but their values do change during the simulation. If an agent joins a group, their values will change as a function of the values of these attributes for the agents inside the group. Hence, they can either increase or decrease depending on the situation. It is important to notice that an agent does not know what will be the outcome of these values beforehand. Therefore, it cannot use this information to influence on its decision to join a group.

The last capability is the agent's stamina. Unlike the other two, this ability is consumable: one stamina point will be consumed for every cycle the agent participates of a group. Hence, a follower agent can join a group only if it has a stamina value greater than or equal to the number of cycles required by the choreography. To recover stamina, the follower agent must not be part of group. In this case, one stamina point will be recovered for every simulation cycle, up to its maximum value.

These three capabilities reflect the "different set of abilities" requirement described in the beginning of this paper (section 2).

4.2 Choreographies as Group Tasks

The mechanism used to implement the task sharing and group formation process is based on the Contract-Net (CNET) protocol [Smith 1977]. The group formation process begins whenever a leader agent is inserted into the system. From time to time the leader agent determines if there are agents nearby (based on a predefined vision radius). If there is at least one, it will send a proposal to this agent and will wait for an answer.

One proposal is a FIPA-ACL[7] message using the performative PROPOSE and containing the reward being offered. This reward can be seen as a "monetary incentive" that is offered to follower agents in case they decide to join the group. Its value is a function of the choreography difficulty and a group reward (*R(G)*) parameter (defined by the user), as follows:

$$reward = R(G) * choreo\_diff. \quad (3)$$

When a follower agent receives a proposal, the first verification it must perform is if it has enough stamina to join the group. If it does not fulfill this requirement, it will send a message back to the leader agent with the REJECT_PROPOSAL performative. On the other hand, if it does have enough stamina, it will then analyze the proposal.

Follower agents follow the principle of utility maximization [Russel and Norvig 2003]. Once a proposal is received, they will analyze whether it is more advantageous (more profitable) to be part of a group or to stay alone. Being rational, one agent will always choose the option that will make it "happier", or in other words, that will maximizes its utility. The utility of being alone (*U(A)*) is defined as:

$$U(A) = (R(A)*chor\_dur)*alone\_factor. \quad (4)$$

Where *R(A)* is the reward for staying alone (defined by the user, similar to *R(G)*); *chor_dur* is the duration of the choreography in cycles; and the *alone_factor* is a multiplier defined as a relation between the agent enthusiasm and its ability, as follows:

$$alone\_factor = ab / enth. \quad (5)$$

[7] FIPA ACL Message Structure Specification: http://www.fipa.org/specs/fipa00061/SC00061G.pdf

Similarly, the utility of being part of a group (*U(G)*) is computed as follows:

$$U(G)= (reward*group\_factor). \quad (6)$$

Where reward is the total reward offered by the leader agent; and *group_factor* is the inverse proportion of *alone_factor*.

After analyzing the proposal, the follower agent must inform its final decision to the leader who has sent the message. It can either accept it or refuse it, based on the values of *U(A)* and *U(G)*. If both have the same value it will randomly choose one. Finally, the agent will send back a message using one of the following performatives: ACCEPT_PROPOSAL or the REJECT_PROPOSAL.

This whole process will go on until the leader agent forms a group. Once a group is formed, all agents will remain in the group until the choreography is over. When it is done, the group will be dissolved and follower agents will update the values for the enthusiasm and ability capabilities.

The simulation will continue, and these agents may eventually be part of other choreographies, or even the same one again.

5. Conclusions

Presentation is crucial for entertainment applications. Thus, providing different aesthetic experience, by adding choreographic elements, or simply bonding strongly graphics, music and sound effects, is an interesting direction to explore in order to offer to users unique experiences. It is easy to find players thrilled by what they are indirectly creating, seeing and listening.

This project presents an original application, *DiscoTech*, which illustrates a category of possible applications more concerned with dance aesthetics. This project also resulted in a reusable framework for creating applications that fall into this category or even in a broader one concerned with visual aesthetics and multi-agent systems in general.

Perhaps the greatest contribution of our work is that we have addressed, employing explicitly stat-of-the-art multi-agent systems techniques, the group formation task, as well as the group coordination problem, from a novel perspective. It is fair to say that most researches are focused on autonomous agents that do not care at all about the aesthetical aspects of their movements. Our research opens a whole set of new and interesting possibilities to be explored by future works, such as new game genres and/or artistic software.

Beyond the "art game" genre, other kinds of games and applications can benefit from choreographic coordination. This is the case, for example, of a simulation game similar to *SimCity*, but related to a night club management where agents could express their level of satisfaction about the environment by executing different kinds of choreographies. Some situations where a group of characters (human or computer controlled ones) participate in a party, watch to show, etc. could also benefit from choreographic coordination mechanisms.

A possible interactive application of choreographic coordination could involve people controlling, by means of handheld device (e.g. cellphone, PDA), a virtual character capable of executing dance movements, and thus interacting with other avatars, on a very large screen installed in a public place, such as shopping centers. The whole environment would work as a kind of hypermedia chat and dance application where people could express their emotions and interests through dance as well.

Yet another possibility would be the creation of collaborative software, allowing human dancers to interact, in real-time, with virtual characters projecting to the images of a mixed reality dance stage. Finally, choreographers could use the software to experiment new choreographies, or simply, to explain to the dancers, through an animation, a given choreography they must execute. These are just a few examples; the list can certainly grow much longer.

Acknowledgements

The authors would like to thank the dancer and choreographer Maria Alcserald, who has greatly contributed with this project.

References

ALMEIDA, A., RAMALHO, G., SANTANA, H., TEDESCO, P., MENEZES, T., CORRUBLE, V. AND CHEVALEYRE, Y. 2004. "Recent Advances on Multiagent Patrolling" in Proceedings of the 17th Brazilian Symposium on Artificial Intelligence, Lecture Notes on Artificial Intelligence, LNAI 3171, Springer Verlag, pp. 474-483.

BELLIFEMINE F., CAIRE G., POGGI A. AND RIMASSA G. 2003. "JADE: A White Paper". Telecom Itlay Lab. Available at: http://jade.cselt.it/papers/WhitePaperJADEEXP.pdf

GILPIN A. AND SANDHOLM T. 2006. "A competitive Texas Hold'em poker player via automated abstraction and real-time equilibrium computation". Available at: http://www.cs.cmu.edu/~sandholm/texas.aaai06.pdf

KLUSCH M. AND GERBER A. 2002. "Dynamic coalition formation among rational agents". IEEE Intelligent Systems. (May/June), Vol. 17, No. 3. pg. 42-27

LIMA R., TEDESCO P AND RAMALHO G. 2005. "Dance: A Framework for Multi-Agent Choreographic Coordination in Games". Proceedings of the CGAMES'2005: 7th International Conference on Computer Games: AI, Animation, Mobile, Educational and Serious Games., (November 2005) Angouleme, France.

MADEIRA, C., CORRUBLE, V. AND RAMALHO, G. 2005. Generating Adequate Representations for Learning from Interaction in Complex Multiagent Simulations. In IEEE/WIC/ACM International Conference on Intelligent Agent Technology (IAT 2005). Compiègne: ACM/IEEE

MARTHI B., RUSSELL S., LATHAM D. AND GUESTRIN C. 2005 Concurrent Hierarchical Reinforcement Learning. In Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence, Edinburgh.

MESSICK P. 1997. Maximum MIDI: Music Apllication in C++. Manning.

POTTINGER D. 1999. "Coordinated Unit Movement" [online]. Game Developer Magazine (Jan): available at: http://www.gamasutra.com/features/19990122/movement_01.htm. [Accessed 01 August 2006]

REYNOLDS, C. W. "Steering Behaviors for Autonomous Characters". Proceedings of Game Developers Conference. San Jose, California: 1999. p. 763-782.

RUSSELL, S. AND NORVIG, P. 2003. Artificial Intelligence: A Modern Approach. 2. ed. Prentice Hall , New Jersey.

SHEHORY O. H AND KRAUS S. 1998. "Methods for task allocation via agent coalition formation". Artificial Intellicence. Volume 101, issue 1-2, (May). pg. 165-200

SMITH D. 2002. "Rez Review" [online]. IGN Playstation 2" (Jan): available at: http://ps2.ign.com/articles/166/166546p1.html. [Accessed 01 August 2006]

SMITH R. G. 1977. "The CONTRACT NET: a formalism for the control of distributed problem solving". Proceeding of the 5th International Joint Conference on Artificial Intelligence (IJCAI-77), Cambridge, MA

VIJSEL M. AND ANDERSON J. 2004. "Coalition formation in multi-agent systems under real-world conditions". Proceedings of the AAAI-04 Workshop on Forming and Maintaining Coalitions and Teams in Adaptive Multiagent Systems, (July 2004) San Jose, CA.