City Research Online

City, University of London Institutional Repository

Citation: Alonso, E. and Fairbank, M. (2013). Emergent and Adaptive Systems of Systems. Paper presented at the 2013 IEEE International Conference on Systems, Man, and Cybernetics (SMC), 13-10-2013 - 16-10-2013, Manchester, UK.

This is the accepted version of the paper.

This version of the publication may differ from the final published version.

Permanent repository link: http://openaccess.city.ac.uk/5197/

Link to published version:

Copyright and reuse: City Research Online aims to make research outputs of City, University of London available to a wider audience. Copyright and Moral Rights remain with the author(s) and/or copyright holders. URLs from City Research Online may be freely distributed and linked to.

City Research Online: http://openaccess.city.ac.uk/ publications@city.ac.uk

Emergent and Adaptive Systems of Systems

Eduardo Alonso
Department of Computer Scicnce
City University London
London EC1V, United Kingdom
E.Alonso@city.ac.uk

Abstract– **The paper provides a first attempt to formalize Systems of Systems based on game theory and Multi-Agent Systems. We propose to use control agents to enforce joint strategies, which solve problems arising from the unconstrained interaction of autonomous agents and where Nash equilibria are suboptimal. In essence, in the resulting systems new solutions emerge, forming Systems of Systems that can form hierarchies. In addition, we present a learning algorithm that allows the resulting Systems of Systems to adapt to varying conditions and uncertainty –in the strategies the constituents follow and/or in their payoffs. The paper presents mechanisms to formalize Systems of Systems and their two main characteristics, namely, emergency and adaptability.**

Keywords-multi-agent systems; system of systems; autonomous agents; adaptive systems; emergence

I. SYSTEMS OF SYSTEMS

Systems of Systems (SoS) are understood as systems that describe the integration of large numbers of independent systems to optimize global functions and multi-system goals. SoS are characterized by their geographic distribution, their operational and managerial autonomy, and by the fact that they adapt over time as the constituent systems are changed, added or removed [1][2]. Notwithstanding their ubiquity and strategic importance, there is little agreement on how to conceptualize and develop SoS. At this point, we are still lacking theories and methods to specify, design, implement and validate SoS. This paper constitutes an attempt to solve this situation by cataloguing SoS according their two main characteristics, namely, *autonomy* and *emergence*, and providing a formal model for SoS and an algorithm for *adaptive* SoS.

We first distinguish between single entities, the primitives, and entities that consist of a number of entities. Primitives are classified according to their autonomy. An autonomous primitive, an agent, has its own utility function that tries to maximize independently. Objects, on the other hand, are non-autonomous primitives. A number of primitives can form an un-structured collection, or a system, where the primitives show structured behavior. Unlike collections, systems cannot be reduced to the mere aggregation of their constituent primitives and new properties emerge. Formally, collections are modeled as sets, and systems as groups [3]. We thus have Collections of Objects (CoO, or Collections of Systems, CoS, following [4][5] broader ontology) and Collections of Agents (CoA, aka Multi-Agent Systems (MAS), a well-established area of research whose nomenclature we are using henceforth to avoid misunderstandings). On the other hand, there will exist Systems of Objects (SoO) and Systems of Agents, which we describe as SoS (again, to avoid confusion with existing practice). MAS (that is CoA) result from the unconstrained interaction amongst autonomous agents, where as SoS are systems of autonomous agents whose co-ordination can be intervened to allow for the emergence of new solutions that satisfy the agents themselves as well as the system as a whole. On the other hand, CoO are gatherings of objects without any particular order or function; such order and function appear in CoS, when a centralized controller organizes them with a particular goal in mind.

As an example: a screw is an object, where as a human being is an agent. A CoO can be a collection of the different objects that would make up an engine. Without a structure, however, they are just a bunch of objects. Under the control of an engineer they are assembled to form a SoO. The constituents are not autonomous, yet, as a system, they are now organized, and show emergent properties: the engine converts energy. If we take agents, a MAS is a collection of agents, each trying to maximize their own utility, like, say, a collection of drivers in a Mad Max movie. A system of agents, a SoS, reflects a structure where autonomous agents get co-ordinated in an effective way, for example by following the advice as given in warning notices in the motorway. The constituents or primitives are still autonomous but, without an explicit centralized control, they form a traffic network in which they interact in new ways that guarantee they achieve their objectives more efficiently (they will get at their destinations safely and quickly); and, as a result, the SoS as whole also satisfies higher-order goals (avoid congestions, reduction of accidents, drop in pollution).

In this paper we explore which type of MAS is needed to represent SoS, paying particular attention to emergence and adaptability. In so doing, we provide the SoS community with a valuable framework for the development of SoS and, at the same time, propose solutions that can be applied to the effective control of MAS. The rest of the paper is structured as follows: we next describe briefly the main characteristics of software agents and of MAS in terms of game theory; and then introduce the idea of C-agents, which enrich the type of behavior needed to formalize SoS. In the second part of the paper we present a learning algorithm that, if used by C-agents, will guarantee the adaption of a SoS controller under changing conditions, that is, its convergence to optimality as the SoS evolves.

II. AGENTS AND EMERGENCE IN SYSTEMS OF SYSTEMS

An agent is assumed to exist in an environment described by a set of possible states S, in which it executes actions from a given set A. Each time t, the agent "perceives" the state of the environment, $s \in S$, and performs an action $a \in A$. As a result, the environment enters into a new state according to a *transition function*, f(s, a) = s'. At the same time, the agent receives an immediate real-valued reward following a *reward function*, $r(s, a) = r \in \Re$. The corresponding system can be formalized as Markov Decision Process MDP = {S, A, f, r}. Notice that the definition of "agent" is kept intentionally broad: an agent can be anything that gets input from the environment in which it is situated through sensors, and acts through actuators. Hence, an agent can be a human being, a physical systems or a software system. We don't need to make assumptions about the internal state of the agent. In particular, the transition function and the reward function, which together constitute the model of the world, may or may not be known by the agent. Likewise, the environment can be deterministic or stochastic, episodic or sequential. Typically, MDPs apply to fully observable environments, in which the agent receives all relevant information through its sensors, but our formalism can be extended to partially observable MDPs. The goal of the agent is to choose a sequence of actions, that is, a policy $\pi : S \rightarrow A$, which maximizes the cumulative reward (the optimal policy). This approach follows Rational Choice Theory (RCT) in that agents make decisions that maximize a utility function that represents their preferences.

A. Multi-Agent Systems and Games

A Multi-Agent System (MAS) is defined as a collection of autonomous agents, where each agent will be individually motivated to achieve its own goal and to maximize its own utility. As a result, no assumptions can be made about agents working together cooperatively. This contrast sharply with CoS (Distributed Problem Solvers in Artificial Intelligence parlance) where control is centralized and individual problem solvers contribute towards the achievement of a common goal (the maximization of global utility)[6][7]. The way autonomous agents interact in MAS can be understood as a *game*. In a game, agents are *players*, which execute *moves*. At the end of the game each agent receives a *payoff* or return. A set of choices or moves, one per agent, is called a *joint strategy*. There are as many joint strategies as combinations of individual choices. For instance, in Fig. 1 $Agent_1$ has two choices, A and B, where as $Agent_2$ has three, L, C, and R. Hence, there are six joint strategies, namely: {A,L},{A,C},{A,R},{B,L},{B,C}, and {B,R}. Each joint strategy gives each agent an outcome, as represented in the cells of the payoff matrix.

		$Agent_2$		
		L	C	R
$Agent_1$				
	A	3, 4	2, 5	1, 3
	B	4, 8	3, 2	2, 5

Figure 1. A two-player game, with six joint strategies, represented as a payoff matrix in normal form. Cells represent the payoff for each agent using the corresponding joint strategy.

The strategies the agents follow are modulated by their attitude towards risk, that is, whether they are risk-averse or they tolerate risk. The strategies also vary according to the nature of the interaction (one-shot or continuous, simultaneous or sequential) and to the type of game they are playing, for instance, if the agents engage in a zero-sum game (where what one gains the other loses) or in a cooperative game (where there may exist win-win solutions). In any case, joint strategies are demanded to be in *Nash-equilibrium*: No agent should have an incentive to individually deviate from agreed-upon strategies. Once a strategy is adopted, under the assumption that one agent uses it, the other agent(s) cannot do better by using a different strategy.

Nash equilibrium is a fundamental concept in game theory since, for RCT agents, that is, for agents that, at each time point, try to maximize their utility function, it gives a dominance criterion. It is well-know, however, that such solution presents serious drawbacks: Firstly, Nash equilibria can be socially irrational as the *Prisoners' Dilemma* illustrates (Fig. 2). In this game, two criminals, P_1 and P_2, are arrested and are given two options: each can either confess (C) or not confess (NC). If both confess, they get 2 years in prison each. If neither confesses, they stay only 1 year in prison. However, if one confess and the other doesn't, the defector is rewarded for his co-operation and set free, where as the one who doesn't confess gets 3 years in prison. In this game, the Nash equilibrium is {confess, confess}: P_1 would reason that if he confesses then P_2 will have an incentive to not confess, and the same applies to the second prisoner. So neither will choose NC. Notice that even if the prisoners are allowed to communicate and reach an agreement, they cannot trust that the other part is going to keep their word. Thus, the criminals will choose the Nash equilibrium, which is clearly an irrational strategy. Both agents will gain if they don't confess. In addition, there are situations in which there is no Nash equilibrium (*Matching Pennies*), or where there are more than one Nash equilibrium (*Battle of the Sexes*).

		P_2	
		Not confess	Confess
P_1	Not confess	1 year each	P2 goes free 3 years P1
	Confess	P1 goes free 3 years P2	2 years each

Figure 2. In the Prisoners' Dilemma, the agents settle for a sub-optimal Nash equilibrium, {C,C}, ignoring the Pareto optimal solution {NC,NC}.

B. C-agents and Systems of Systems

The problem we are facing in the *Prisoners' Dilemma* is not that the concept of Nash equilibrium is "wrong", but rather that autonomous agents are free-riders who try to maximize their own utility and cannot trust one another. As a consequence, agents under-achieve, and settle for sub-optimal outcomes, or reach a stalemate where no decisions are made (in *Matching Pennies* and the *Battle of Sexes*). Hence, relying on individual autonomous agents to get co-ordinated on their own, or, in other words, assuming that, in a MAS, the aggregation of agents will result in harmonious collective behavior is a chimera. We argue that the SoS notion gives us the right balance between blind cooperation in CoS and unrestricted autonomy in MAS, and, following [8], we propose to shift from the traditional *homo economicus* approach to multi-agent co-ordination to a *homo sociologicus* approach. In order to do this, we introduce the notion of *C-agent*.

A C-agent is a special type of agent whose transition function f operates on states defined as *joint strategies* (rather than individual strategies), and whose actions are *normative actions*. Such actions create a deontic structure (in terms of permissions and obligations) and enforce joint strategies –such as {NC,NC} in the *Prisoners' Dilemma*. C-agent's actions change a sub-optimal joint strategy into an optimal one. C-agent's reward function, can take many forms and metrics, from the addition of individual utilities (for instance, all vehicles get to their destination quickly and safely) to the maximization of global utilities (traffic volume, congestion, pollution, casualties). The important thing is that the system's properties *emerge* from the provision of new solutions, through a C-agent, to individuals' problems, not all the way around. This is the essence of a SoS.

To clarify this concept, let's recapitulate on the three types of systems we are considering using economic systems as an example:

- *Planned Economy:* a central controller dictates an optimal joint plan to maximize global utility. This would be a CoS, a collection of systems where the parts or systems it consists of are organized according to a common goal. Co-ordination takes the form of commands from higher nodes, the controllers, to lower nodes, forming strict hierarchies. Although problems as the ones described above are avoided "by decree", agents are not seen as autonomous entities.
- *Free Market:* a collection of autonomous agents work under the assumption that an "invisible hand" self-regulates the behavior of the system. This will correspond to MAS in their purest form, where free-riders try to maximize their own utility. The resulting structures go from anarchy to heterarchies of various topologies.
- *Interventionism:* a C-agent intervenes so that new solutions "emerge". Notice that there is no central controller setting the game, the players, their actions or their outcomes. And there is no global utility function as such that is being maximized. The controller only intervenes to "solve problems" and to guarantee that the individuals maximize their utility functions. We argue that this corresponds to SoS. Importantly, hierarchies are formed as in Fig. 3.

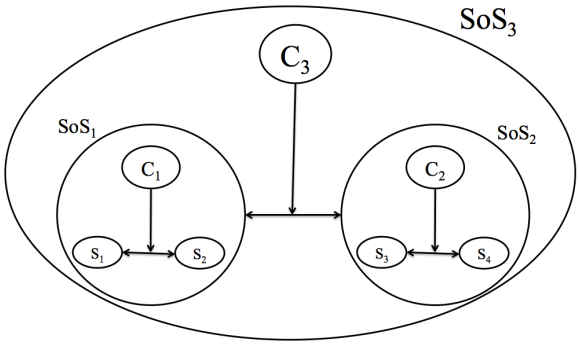

Figure 3. A hierarchy of systems. C-agents (C_i) intervene in the co-ordination of systems (s_i), forming SoS_i. In turn, such SoS_i are systems of higher-order SoS.

This behavior can be seen in all types of SoS. If smart grids or the Internet were to function in a deregulated way, the greed of the components will prevent them from reaching mutually beneficial solutions, solutions that also benefit the system as a whole, avoiding faults, blockages, etc. For instance, in traffic networks drivers have their own goals and make their own choices (when to drive and where to go, which route to take, etc.). However, for them to reach their objectives (reach the destination in the shortest time and safely) they cannot act in an unconstrained way. They must abide by some rules, the traffic code. In so doing, they maximize their utility functions and that of the SoS: the network works smoothly, traffic flows without jams. It is precisely this "order within chaos" where new behaviors (solutions to the traffic game) emerge.

Apart from their "constrained" autonomy and emergence, SoS are characterized by their adaptability. Complex systems are highly dynamic and inherently uncertain. Hence, they need to learn from and adapt to changing conditions.

III. ADAPTIVE SYSTEMS OF SYSTEMS

Reinforcement Learning (RL) is the most influential learning paradigm in implementing adaptive agents. According to RL, an agent interacts with the environment and, by trial and error, learns an optimal policy. Formally, the idea is to learn a separate value for each action leading out of a state V(s, a), defined as the sum of the immediate reward the agent receives and predictions that decay with time according to a factor γ (illustrating that predictions on future values are uncertain and thus carry less weight). With this information, the agent calculates the "error" between successive predictions, using a delta rule like Q-learning's [9],

$$\Delta V(s, a) = r + \gamma \max_{a'} V(s', a') - V(s, a) \quad (1)$$

The output is used to update the V value according to

$$NewV(s, a) = OldV(s, a) + \alpha \Delta V(s, a) \quad (2)$$

where α is a learning rate. When accompanied with a trace, λ, representing the eligibility of a given (s, a) pair, these two equations define the so-called Temporal Difference algorithm (TD). TD complies with Bellman's Principle of Optimality [10]: If the target value is met for all states in the state space S, then the policy is globally optimal.

A. Multi-Agent Reinforcement Learning

There exist a plethora of proposals in the literature on how to extend RL to MAS, all of which follow in one way or another the *Free Market* model described in Section II (see [11] for a survey). The simplest way to extend the single-agent Q-learning algorithm to multi-agent games is to add a subscript to the original formulae, that is, to have the learning agent pretend that the environment is passive. However simple this technique may be, the definition of the V-values assumes incorrectly that they are independent of the actions selected by other agents. Second, it is no longer sensible to use the maximum of the V-values to update individual policies. In order to solve this problem, a minimax Q-learning algorithm can be used. However, minimax only applies to strictly competitive games. Alternatively, an agent can update its V-values based on its expectations on the likelihood of the other agent's policies. This approach, called Joint Action Learners, is useful in the context of common-payoff games (aka team games or pure co-ordination games) in which agents that receive the same pay-off at each outcome co-operate, but, like minimax, it does not apply to general-sum games. As a general solution, it has been proposed that agents update the V-values based on some Nash equilibrium, using Nash-Q learning or one of its many variants. Notwithstanding its merits, the conditions for convergence in Nash Q-learning are very restrictive, and it only converges in special cases. In addition, as in Section II, the solution, a Nash-equilibrium, has no prescriptive force.

This inability to solve the RL problem in MAS has led to the abandonment of the equilibrium agenda in favor of optimal agent designs that comply with social criteria. Our proposal, to apply RL to C-agents as a mechanism to make MAS, rather SoS, adaptive, can be framed within this new approach.

B. C-agent Learning

To develop adaptive SoS we don't need to consider the integration of a collection of independent learners à la Multi-Agent RL. Rather, it is enough to consider how the C-agent of the SoS learns. In order to do so, we can use single-agent TD. In the case of C-agents, the states and the actions in the V-values, V(s.a), are associated to joint strategies and normative actions. However intuitive this approach is, there are two problems which prevent us from applying directly the Q-learning algorithm to C-agents and SoS, namely:

1. Q-learning, as any TD algorithm, is model-free. That is, the agents do not know the transition function or the reward function. In SoS on the other hand, the C-agent does have a model of the joint strategies and of the resulting payoffs. That is, learning in SoS is model-based.
2. Since Q-learning agents are model-free, they must explore the whole state space to abide by Bellman's Principle of Optimality. As a result, Q-learning converges to optimality only in tabular cases [12]. Contrarily, C-agents do not require exploration. This is convenient since SoS work with large state spaces.

Recently, we have presented a model-based variation of TD, which we call Value Gradient Learning (VGL) [13]. VGL works in continuous state spaces, and it does so without exploration. It seems thus an ideal candidate for adaptive C-agents. The main difference between TD and VGL lies on *what* is learned: VGL learns gradients of values as opposed to TD algorithms that only learn values. We define the value gradient as

$$G(s, a) = \frac{\partial V(s, a)}{\partial s} \quad (3)$$

and the *approximate* value gradient as

$$\tilde{G}(s, \vec{w}) = \frac{\partial \tilde{V}(s, a)}{\partial s} \quad (4)$$

The VGL algorithm is defined at each t by a weight update of the form

$$\Delta \vec{w} = \alpha \sum_t \left(\frac{\partial \tilde{G}}{\partial \vec{w}} \right)_t \left(G'_t - \tilde{G}_t \right) \quad (5)$$

where G' is the *target* value gradient defined recursively by

$$G'_t = \left(\frac{Dr}{Ds} \right)_t + \gamma \left(\frac{Df}{Ds} \right)_t \left(\lambda G'_{t+1} + (1 - \lambda) \tilde{G}_{t+1} \right) \quad (6)$$

We have proved that any greedy trajectory for which the target gradient is met, is locally extremal, and often locally optimal. With VGL, the trajectories follow a greedy gradient, that is, they *bend* towards optimality at each point. This local optimality condition needs satisfying only over a single trajectory, whereas for TD the corresponding optimal condition (Bellman's) needs satisfying over the whole state space. Crucially, a trajectory can reach a fixed point using TD and still be far away from optimality. It should be noticed that VGL is not a differentiated form of TD, that is, VGL does not abide by Bellman's equation in continuous spaces (HJB equation). In fact, VGL is equivalent to Pontryagin's Mimimum Principle (PMP) commonly used in control systems theory [14].

IV. CONCLUSION

In this paper we present a new approach to SoS based on a classification of different types of systems and the way the interaction and get co-ordinated. In particular, we identify C-agents as a notion that allows us to move from MAS to SoS. The agents playing such role can intervene and enforce new solutions to a MAS problem, and are thus instrumental in providing space for emergent properties. We believe that this analysis is a starting point for the development of a methodology that may lead to the systematic design of SoS. Examining the rules of composition of the subsystems and their coordination as agents in a larger system defines a challenging new area for research and requires links across many disciplines. Multi-Agent theory and adaptive games may provide an avenue for describing the new notion of the *system play* that now provides the substitute to the interconnection topology of the standard systems theory. The potential for applications is well beyond the traditional engineering field.

In this new framework for SoS, subsystems appear as autonomous systems which follow their own interests and thus act as agents participating in games. This introduces the idea that the system can be seen from a controller perspective where actions are defined as the outcomes of adaptive games. We would like to stress that such view is hierarchical and that it gives as a formal tool to construct increasingly complex SoS. We have also introduced VGL as a model-based algorithm that allows the controller to learn on-line how to adapt the SoS so as to convergence to optimality. Providing a representation of the *system play* in more general cases requires some further work and the introduction of a formal set up. This is an area under investigation.

REFERENCES

[1] N. Karcanias and A.G. Hessami, "Complexity and the notion of Systems of Systems: Part (I) General Systems and Complexity," Proc. of the 2010 World Automation Congress International Symposium on Intelligent Automation and Control (ISIAC) 19-23 September 2010, Kobe Japan.

[2] N. Karcanias and A.G. Hessami, "Complexity and the notion of Systems of Systems: Part (II) Defining the notion of Systems of Systems," Proc. of the 2010 World Automation Congress International Symposium on Intelligent Automation and Control (ISIAC) 19-23 September 2010, Kobe Japan.

[3] E. Alonso, N. Karcanias and A. G. Hessami, "Symmetries, groups and groupoids for Systems of Systems," Proceedings of the 7th IEEE International Systems Conference (SysCon 2013), Piscataway, NJ: IEEE Press. Orlando, FL, 15-18 April, 2013.

[4] N. Karcanias and A.G. Hessami, "System of Systems Emergence: Part (I) Principles and Framework," Proc. of the ICETET 2011, 4th International Conference on Emerging Trends in Engineering and Technology, SV129, Port Louis, Mauritius, November 18-20, 2011.

[5] N. Karcanias and A.G. Hessami, "System of Systems Emergence: Part (II) Synergetics Effects and Emergence," Proc. of the ICETET 2011, 4th Internationa Cnference on Emerging Trends in Engineering and Technology, SV129, Port Louis, Mauritius, November 18-20, 2011.

[6] E. Alonso, "AI and Agents: State of the Art, AI Magazine," 23 (3), pp. 529–551, 2002.

[7] E. Alonso, "Actions and Agents," K. Frankish and W. Ramsey (eds.), The Cambridge Handbook of Artificial Intelligence, Chapter 5. Cambridge, England: Cambridge University Press. 2013.

[8] A. Alonso, "Rights and Argumentation in Open Multi-Agent Systems," Artificial Intelligence Review, 21(1), pp 3-24, 2004.

[9] C.J.C.H. Watkins, "Learning from Delayed Rewards," Ph.D. Thesis, Cambridge University, 1989.

[10] R. Bellman, "Dynamic Programming," Princeton, NJ: Princeton University Press, 1957.

[11] E. Alonso, M. d'Inverno, D. Kudenko, M. Luck and J. Noble, "Learning in Multi-Agent Systems," Knowledge Engineering Review 16 (3), pp. 277-284, 2001.

[12] C.J.C.H. Watkins and P. Dayan, "Q-learning", Machine Learnig, 8, pp. 279-292, 1992.

[13] M. Fairbank, D. Prokhorov and E. Alonso, "Approximating Optimal Control with V alue Gradient Learning," F. L. Lewis and D. Liu (Eds.), Reinforcement Learning and Approximate Dynamic Programming for Feedback Control, Hoboken, NJ: Wiley-IEEE Press, pp. 142-161, 2013.

[14] L.S. Pontryagin, V.G. Boltyanskii, R.V. Gamkrelidze and E.F. Mishchenko, "The Mathematical Theory of Optimal Processes," New York, NJ: Gordon and Breach Science Publishers, 1962.