Article

A new game theory algorithm simulates soccer matches: Reducing complexity to its irreducible essence

Alessandro Ferrarini
Department of Evolutionary and Functional Biology, University of Parma, Via G. Saragat 4, I-43100 Parma, Italy
E-mail: sgtpm@libero.it, alessandro.ferrarini@unipr.it

Received 9 Nov 2014; *Accepted 12 Nov 2014*; *Published online 1 December 2014*

Abstract
How much complex is a human event like a soccer match? How much difficult is to predict its result? Can we disentangle the complexity behind such event? In this work, I state that the use of multiagent systems to simulate soccer events is improper: too many possible space-time configurations are possible, and the resulting complexity is unimaginable. A proper way to simulate such complex event is to turn its complexity into its irreducible essence. When such irreducible essence is tamed, stochasticity and iteration can then be added. I describe here in outline a math algorithm, named *Soccer-Decoder* and implemented through the software *Soccer-Lab*, that is based on game theory and differential calculus and that exactly does this: 1) it turns the complexity of a soccer match into its irreducible and structural essence, 2) it simulates soccer matches by adding stochasticity and iteration to such structural essence. An illustrative example is given. The philosophy on the underside of *Soccer-Decoder* is that even very complex real world events, when transformed into their irreducible essence, can be understood and predicted.

Keywords complexity; game theory; iteration, simulation; soccer; sport event; stochasticity.

Selforganizology
URL: http://www.iaees.org/publications/journals/selforganizology/online-version.asp
RSS: http://www.iaees.org/publications/journals/ selforganizology /rss.xml
E-mail: selforganizology@iaees.org
Editor-in-Chief: WenJun Zhang
Publisher: International Academy of Ecology and Environmental Sciences

1 Introduction

In a soccer match, there are 2 teams $<T_1, T_2>$, 22 players $<P_1...P_{22}>$, a rectangular field F where each point has its coordinates F_{xy}, and a clock vector C assuming continuous values from 0 to 5400 seconds (i.e. 90 minutes). Probably the most obvious scientific way to simulate such soccer event could seem the use of multiagent systems (Ferber 1999). By the way, I invite to think about some aspects dealing with the space and time extents of this complex event. Let's suppose to divide the soccer field F into discrete squares of 1 sq. meter. Thus, the whole (100 m * 60 m) soccer field F would result divided into 6000 discrete squares. At each moment C_i $(0 \leq i \leq 5400)$, each square can assume 23 different configurations. In fact, it can be occupied by any of the 22 players, but it can also be empty. It follows that at each moment, we have 23^{6000} possible spatial configurations of the soccer game, where each player can of course occupy only 1 square at any moment. If we also consider that, at any moment C_i, in any square a player can play at least 10 actions (e.g., run, pass, shot

etc.) it follows that $10^{\wedge}(23^{6000})$ events are possible at each moment. It results that, during the whole soccer match, if we consider together the spatial and temporal dimensions we have $5400*(10^{\wedge}(23^{6000}))$ possible space-time configurations of soccer events. The amount of such grandness is unimaginable.

This is the reason why I state that the use of multiagent systems to simulate soccer events is improper: too many possible configurations are possible. The only way to simulate such event is to turn its complexity into its irreducible essence. When such irreducible essence is tamed, stochasticity can then be added.

In this paper, I describe in outline a math algorithm, named *Soccer-Decoder* (Ferrarini, 2012a) and implemented through the software *Soccer-Lab* (Ferrarini, 2012b), based on game theory (Brandenburger, 2014; Maynard Smith, 1982), which exactly does this: 1) it turns soccer complexity into its irreducible and structural essence, and 2) simulates soccer matches by adding stochasticity to such structural essence.

2 A Simulation Framework Based on Game Theory

The math algorithm *Soccer-Decoder* turns a soccer match into the following variables and parameters:

- defensive skill (*DS*)
- midfield skill (*MS*)
- offensive skill (*OS*)
- goalkeeper skill (GS)
- field factor (*FF*)
- trainer skill (*TS*)
- players experience (*PE*)
- athletic decay (*AD*)
- game style (*GS*)

Midfield skill is given by

$$MS = \sum_k M_k \qquad (1)$$

where M_k is the skill of each midfielder. The number of midfielders is set-up by the user.

Defensive skill is given by

$$DS = \sum_i D_i + \frac{1}{2}\sum_k M_k \qquad (2)$$

where D_i is the skill of each defender, while M_k is the skill of each midfielder. The rationale is that the defensive phase is made by defenders above all, but also midfielders give a (lesser) contribution.

The number of defenders is set-up by the user.

Offensive skill is given by

$$OS = \sum_j S_j + \frac{1}{2}\sum_k M_k \qquad (3)$$

where S_j is the skill of each striker, while M_k is the skill of each midfielder. The rationale is that the offensive phase is made by strikers above all, but also midfielders give a (lesser) contribution. The number of strikers is set-up by the user.

The field factor (*FF*), the trainer skill (*TS*) and the players experience (*PE*) add scores to *DS*, *MS* and *OS*. The athletic decay *AD* during the match acts as follows:

$$\begin{cases} \dfrac{dDS}{dt} = -AD * DS \\ \dfrac{dMS}{dt} = -AD * MS \\ \dfrac{dOS}{dt} = -AD * OS \end{cases} \quad (4)$$

To date, two game styles (*GS)* are possible in *Soccer-Decoder*: ball possession (*BP*) and counter-attack (*CA*). For example, a *BP* action of team 1 happens using the following algorithm:

MS of team 1 VS *MS* of team 2
if *MS* of team 2 wins the battle, then the action of team 1 is over
else
OS of team 1 VS *DS* of team 2
if *DS* of team 2 wins the battle, then the action of team 1 is over (5)
else
OS of team 1 VS *GS* of team 2
if *GS* of team 2 wins the battle, then the action of team 1 is over
else GOAL

How to decide the winner of each single battle (e.g. MS_1 *vs.* MS_2 or OS_1 vs. DS_2)? To do this, *Soccer-Decoder* makes use of the following algorithm. Let's suppose that we want to simulate, for a single action, the battle between *MS* of team 1 and *MS* of team 2. *Soccer-Decoder* produces a random number R_1 between 0 and MS_1. R_1 is sampled from a statistical Erlang's distribution (Fig. 1)

$$\begin{cases} f(x) = \dfrac{1}{\beta(m-1)!}\left(\dfrac{x}{\beta}\right)^{\alpha-1} e^{-x/\beta} \\ \text{with:} \\ m > 0 \\ \beta > 0 \end{cases} \quad (6)$$

with peak exactly equal to *MS* (Fig. 1). In other words, the random number R_1 has higher chance to be close to *MS* but it can also, with lower probability, bear values < *MS*.

The rationale behind this algorithm is clear. During each battle, a team's unit (defence, midfield, attack) can't do better than its best. Hence, stochasticity must generate a number that is equal or lower than the unit's overall skill (i.e. DS, MS, OS). Of course, such number can't be completely random. Erlang's distribution (Fig. 1) is effective in order to produce realistic random numbers which, at most, are equal to the unit's overall skill. I have also tried numerous other statistical distributions, for instance Chi and Chi-squared distributions, extreme value (Gumbel) distribution, gamma and log-normal ones. By the way, Erlang's distribution provided the best results when I compared the outcomes of Soccer-Decoder to real life soccer matches.

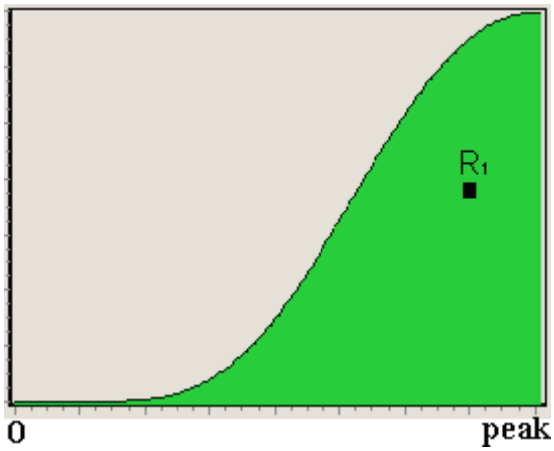

Fig. 1 Erlang's statistical distribution used by the math algorithm *Soccer-Decoder* to play battles.

Then, *Soccer-Decoder* does the same for team 2 and a battle happens where the higher score wins:

$$\begin{aligned} &\text{if } R_1 \in [0, MS_1] > R_2 \in [0, MS_2] \text{ then the action continues} \\ &\qquad \text{else} \\ &\text{the action of team 1 is over} \end{aligned} \tag{7}$$

It should be noted that both MS_1 and MS_2 (but also DS and OS) change continuously over time based on eq. (4). This assures a realistic dynamical evolution of the soccer match, where players have an athletic decay that, step-by-step, lowers their performances. In practice, *Soccer-Decoder* (Ferrarini, 2012a) is a math algorithm that iteratively produces game theory battles with dynamical and stochastic parameters. In order to do this, *Soccer-Decoder* merges together game theory with differential equations and stochastic simulations. Calculations are performed through the software *Soccer-Lab* (Ferrarini, 2012b) programmed in Visual Basic (Balena, 2001; Pattinson, 1998).

Soccer-Lab (Fig. 2) can simulate one match, but also N matches (e.g., N = 1,000,000). The simulation of N matches obeys the following pseudo-code:

```
FOR MATCHES = 1 TO N
    FOR ACTIONS=1 TO 100
        FOR TEAMS=1 TO 2
            APPLY the Soccer-Decoder algorithm          (8)
        NEXT TEAMS
    NEXT ACTIONS
NEXT MATCHES
```

For each team, 100 actions are simulated during each single match, since 100 is a common number of actions of recent soccer matches. By the way, the user can define a different number of actions.

The definition of "action" is given in (5). Both in ball possession (BP) and counter-attack (CA) game styles, one action consists of one to several battles depending on the winner of such battles.

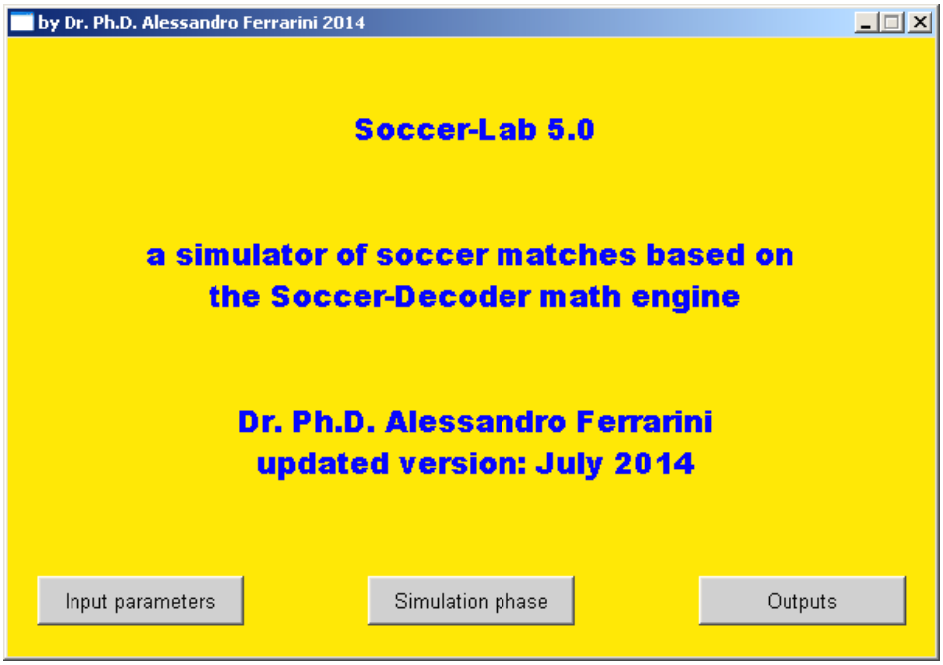

Fig. 2 The splash screen of *Soccer-Lab* (Ferrarini 2012b). *Soccer-Lab* makes use of a math algorithm based on game theory.

Further improvements to *Soccer-Lab* are underway. The most interesting one is the joining of *Soccer-Decoder* with Evolutionary Network Modelling (EVN; Ferrarini, 2011; Ferrarini, 2013a; Ferrarini, 2013b; Ferrarini, 2013c; Ferrarini, 2013d; Ferrarini, 2013e; Ferrarini, 2014). Such improvement is thought for two purposes:

1) given a real soccer match, EVN can estimate a set of optimized parameters for *Soccer-Decoder* in order to fit the simulated match to the real one;

2) EVN can find the optimized parameters for a soccer team in order to change a probable simulated defeat into a probable simulated victory (e.g. changing the number of defenders or midfielders or strikers, passing from ball possession to counter-attack and vice versa etc…).

Further advances are almost complete. Among these:

1) each team can change its game style (BP or CA) at any moment during the match

2) each team can change its strategy (e.g. 3-5-2, 3-4-3, 4-4-2, 4-3-3 etc.) at any moment of the match

3) player substitutions are possible at any point of the match.

3 An Illustrative Example

Let's consider the two imaginary soccer teams of Fig. 3.

There are two teams with differently skilled players and different playbooks. The Red Team plays a 3-5-2 strategy, while the Blue Team plays a 4-4-2 scheme. It follows that the Red Team mainly bets on the strength of its midfielders to overcome the Blue Team. Instead, the Blue Team adopts a more prudent game strategy with 4 defenders.

From a strategic viewpoint, it is logical that the Red Team adopts a ball possession (BP) type of game where actions are mainly driven by its midfielders. As opposite, it is logical that the Blue Team opts for a game strategy through which it can jump the Red Team's midfield using long launches from the defensive players toward the strikers. In other words, the most likely game strategy for the Blue Team is counter-attack (CA).

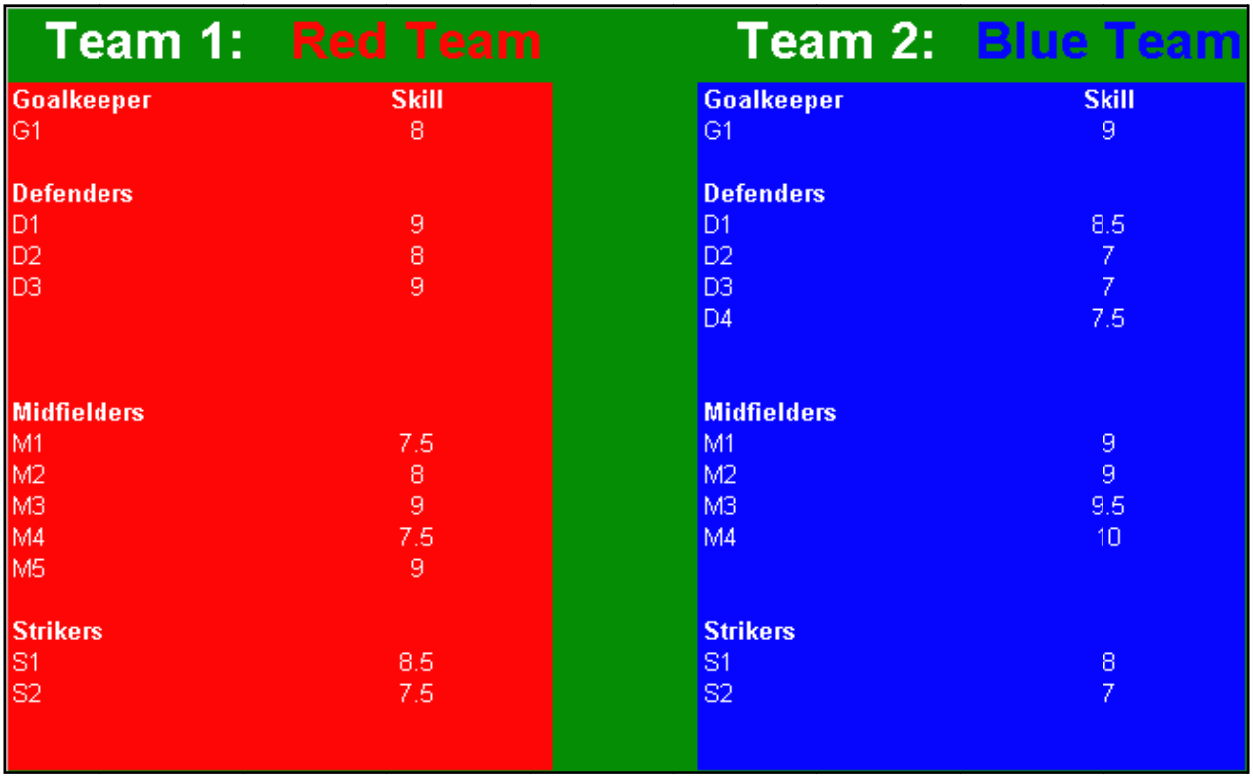

Fig. 3 Two imaginary soccer teams with differently skilled players and different playbooks.

I'll make use of the game parameters of Table 1 for the 2 teams.

Table 1 Game parameters.

Game parameters	Red Team	Blue Team
Field factor (*FF*)	**0**	**0**
Trainer Skill (*TS*)	**2**	**1**
Players experience (*PE*)	**3**	**2**
Athletic decay (*AD*)	**0.35%**	**0.20%**
Game style (*GS*; 1= ball possession, 2= counter-attack)	**1**	**2**

The two teams play on a neutral field (*FF*=0 for both teams). The Red team plays ball possession, while the Blue Team makes use of counter-attack. The Red Team is superior for the ability of its trainer and the experience of its players. The Blue Team has a better athletic condition, hence its players' performances will decrease less as the match proceeds.

Soccer-Decoder first calculates the overall parameters for each team (Table 2).

Table 2 Overall game parameters.

Game parameters	Red Team	Blue Team
defensive skill (DS)	**51.5**	**51.75**
midfield skill (MS)	**46**	**40.5**
offensive skill (OS)	**41.5**	**36.75**

Now I'll simulate just 1 soccer match (Table 3). The Red Team is predicted to win 2-1 (2-0 after the first half of the match). It is interesting to note that, since the Red Team plays ball possession, 50 out of 100 of its actions have been stopped by the opponent midfield. Instead Team 2, which plays a counter-attack game, has been prevalently stopped by opponent defence (73 times out of 100). The Read Team has shot 10 times on goal, the Blue Team just 2 times (Table 3). The assignment of goals to defenders, midfielders and strikers follows a complex algorithm not described here.

Table 3 Game synthesis. The Red Team is predicted to win 2-1.

Match synthesis	Red Team	Blue Team
goals	2	1
goals by defenders	0	0
goals by midfielders	1	1
goals by strikers	1	0
actions blocked by opponent midfield	50	24
actions blocked by opponent defence	38	73
actions blocked by opponent goalkeeper	10	2

Now I'll simulate 1000 soccer matches between the two teams. Depending on several parameters, each match is the result of about one thousand game theory battles. This means that the simulation of 1000 matches requires about 1 million battles to be calculated. Results are showed in Table 4.

Table 4 Results of the simulation of 1000 soccer matches between the Red Team and the Blue one.

Simulation of 1000 matches	Red Team	Blue Team
won matches	633	101
drawn matches	266	266
lost matches	101	633
scored goals	1274	379
opponent goals	379	1274
most likely result	1	0

After 1000 simulated matches, we can conclude that the Red Team has a probability equal to 63.3% to win the match (26.6% of getting a draw, and 10.1% of losing the match), and that the most likely match result is 1-0 for the Red Team (277 times out of 1000) with goal by a midfielder in the first half of the match. The second most probable result is 2-1 for the Red Team (161 times out of 1000; 2-0 in the first half of the match).The third most probable result is 1-1 (137 times out of 1000) with goals by a striker (Red Team; first half of the match) and a midfielder (Blue Team; second half).

4 Conclusions

Game theory, differential calculus and stochastic simulations are combined by the math algorithm *Soccer-Decoder* in order to simulate the complexity of a human event like a soccer match. The philosophy on

the underside of *Soccer-Decoder* is that even very complex real world events, when transformed into their irreducible essence, can be understood and predicted.

Improvements to *Soccer-Lab* are underway. The most interesting one is the joining of *Soccer-Decoder* with Evolutionary Network Modelling for eliminating any subjectivity in the attribution of simulation parameters, and for estimating the optimized set of parameters of a soccer team in order to change a probable defeat into a probable victory.

References

Balena F. 2001. Programming Microsoft Visual Basic 6.0. Microsoft Press, Redmond, WA, USA

Brandenburger A. 2014. The Language of Game Theory: Putting Epistemics into the Mathematics of Games. World Scientific Series in Economic Theory: Vol. 5. World Scientific, Singapore

Ferber J. 1999. Multi-Agent Systems: An Introduction to Artificial Intelligence. Addison-Wesley Press, USA

Ferrarini A. 2011. Some thoughts on the controllability of network systems. Network Biology, 1(3-4): 186-188

Ferrarini A. 2012a. *Soccer-Decoder*: A Math Engine for Simulating Soccer Matches. Manual, 82 pages (in Italian)

Ferrarini A. 2012b. *Soccer-Lab*: A Math Simulator of Soccer Matches. Manual, 84 pages (in Italian)

Ferrarini A. 2013a. Exogenous control of biological and ecological systems through evolutionary modelling. Proceedings of the International Academy of Ecology and Environmental Sciences, 3(3): 257-265

Ferrarini A. 2013b. Controlling ecological and biological networks via evolutionary modelling. Network Biology, 3(3): 97-105

Ferrarini A. 2013c. Computing the uncertainty associated with the control of ecological and biological systems. Computational Ecology and Software, 3(3): 74-80

Ferrarini A. 2013d. Networks control: introducing the degree of success and feasibility. Network Biology, 3(4): 115-120

Ferrarini A. 2013e. Control-Lab 5.0: A software for ruling quantitative ecological networks using Evolutionary Network Control. Manual, 137 pages

Ferrarini A. 2014. Local and global control of ecological and biological networks. Network Biology, 4(1): 21-30

Maynard Smith J. 1982. Evolution and the Theory of Games. Cambridge University Press, USA

Pattison T. 1998. Programming Distributed Applications with COM and Microsoft Visual Basic 6.0. Microsoft Press, Redmond, WA, USA