Modeling and Simulation of a Team Game with Coloured Petri Nets

POPA MARIN
DRĂGAN MIHĂIȚĂ
POPA MARIANA
Technology Department
University of Bucharest
Bd. M. Kogalniceanu N^0.36-46, Zip Code 050107, Bucharest
ROMANIA
marpopa2002@yahoo.com , *dragan_mihaita.@yahoo.com*

Abstract: - This paper work describes an example of how systems (economic, social, political, sporting, etc.) can be modeled by colored Petri nets. Here we present the modeling and the simulation of a team game. The network, which modeled the system, is described in detail. CPN-Tools are used for drawing a network colored Petri,in order to obtain the coverage graph that appears following evolution of a network, starting from a certain state of the system and to obtain a report that provides information about : boundedness properties, liveness properties and fairness properties of the modeling network .

Key-Words: - coloured nets, location, transition, multiset, liveness, fairness.

1 Introduction

The system that will be modeled is taken from the a sports club activity, where are practiced more team games, more specifically the activity of a team football that play in the National League. How nowadays the latest generation technologies are present in almost all activity areas, the sport also is a dynamic domain and can not stay away from the use of new technologies that can bring major benefits to improve performance.

IT products - software and hardware - occupy a central place in the accelerated development of modern technologies. Using these discoveries of high priority, the sport coaches can substantially improve the performance of their teams.

To achieve these goals through modeling and simulation of such a system, the sport coaches can get a lot of statistics, such as:

- The number of properly ball passings, that were made of its own team;
- The number of wrong ball passings, made by the pleyers of its own team;
- The number of fouls suported by theit's own team;
- The number of fouls committed by players of it's own team;
- The number of shoots at gate;
- The number of shoots on the gate sent of players of it's own team.

Analyzing the data such obtained, the coach can intervene in time to remedy any deficiencies arising in the his game team.

2 Description of the Model

For modeling the system we use the following locations and transitions [1, 2]:

A. Locations that provide statistics needed to evaluate the game by the coach.

NFCR – here is added the fouls number of the red card made by the opposing team.

NSPP – here is added the number of shoots on the gate, made by his own players.

NSSP – here is added the number of shoots at gate made by his team.

MLJ – specifies at which player is the ball.

JCM – specifies the next player that will pass the ball.

NPgr – here is added the number of wrong passes, made by his team.

NPbu – here is added the number of properly passes made by his team.

NFS – here is added the number of fouls supported by the players of his team.

NFDJ – here is added the number of fouls of game made by the opposing team.

NFCG – count the fouls for which the opposing team received yellow cards.

NFC – here is added the number of fouls committed by his team.

NCR – here is added the red cards number, accumulated in the game by his own team.

NFU – here is added the number of easy game fouls comitted by his own team.

NCG – here is added the number yellow cards accumulated by his team.

Locations [1, 2, 3] that helps carrying the game in optimal conditions.

fp – specifies when a player of his team made a foul.
cr – specifies whether a player of own team received a red card.
cg – specifies whether a player of own team received a yellow card.
f easy - specifies whether a player of his team committed easy foul game.
fad – specifies when a player from the opposing team has committed a foul.
pgr – specifies when a player of own team passes the ball wrong.
pbu – specifies when a player of own team made a properly passing of the ball to a teammate.
fdj – specifies whether a player from the opposing team has made an easy foul game.
fcg – specifies whether a player from the opposing team has comitted a foul of yellow card.
fcr – specifies whether a player from the opposing team has comitted a foul of red card.
spp – specifies when one of his players shoots on the gate.
ssp – specifies whenone of his players shoots at the gate.

B. Transitions [1, 2, 3] are the following :
dribbling - transition that indicates the player found in possession of the ball forward with the ball towards the opposing gate.
passes - transition indicating that the player found with the ball passes the ball to a teammate.
recuper – transition by which a player of his own team recovers the ball after a pass done wrong, by the opponent.
sutlp – transition by which a player of own team shoots to the opposing gate.
degajare – transition by which a player of own team recovers the ball after the oppenent shoots to the gate.
degaj – transition by which a player of own team recovers the ball after this was released by the opponent, after a shooting on the gate.
fault – transition which indicates the moment when the game stops, after a foul of a player which has the ball in its possession or has committed a foul on this player.
gravit – transition in which is deciding severity of a foul comitted of an opponent.
executa – transition which indicates execution of the free kick, resulting from a game foul comitted by the opponent.
lovlib – transition which indicates execution of the free kick, resulting from a yellow card foul of the opponent.
lovdir – transition which indicates the execution of a free kick, that results from a red card foul of the opponent.
lovitlib – transition which indicates the execution of a free kick, that results from an easy foul of the own team.
preia – transition which indicates the execution of the free kick that results from a foul of yellow card of the own team.
lovitdir – transition which indicates the execution of the free kick, that results from a foul of red card made by the own team.
tipfault – transition in which is deciding the level of danger of a foul made by own team.

The family sets of colors *F* of the network that models the football game has the following composition : *F = {A, B, F, G, N, S, T }* in which :
- *A = {ion, vas, mar, ghe, flo, zac, con, mut, dud, lun, mor }*. Sets of colors that paints the location "MLJ", indicating the own team players.
- *B = {pbu, pgr }*. Sets of colors that paint the locations NPbu and NPgr, which specifies the ball passings quality, made by the own player found in possession of the ball, colors that are understood such so: pbu = passes to a teammate, pgr = passes to the opponent.
- *F = { fpr, fad }*. Sets of colors which paint the locations NFS and NFC, colors that can be interpreted such as: fpr = fouls made by the own team, fad = fouls made by the opposing team.
- *G = { fj, fcg, fcr }*. Sets of colors which paint the locations NFDJ, FCG and NFCR, that specifies the gravity of the fouls committed by the opposing team, colors that can be read such as: fj = easy foul, fcg = yellow card foul, fcr = red card foul.
- *N = { n }*. Sets of colors which paint all the helpful locations, that allow a logical links of the other locations and transitions according to the principles of the footbal game in real time.
- *S = { sp, so }*. Sets of colors which paint the locations NSSP and NSPP, by which are specified the quality of the shoot by the own players, colors that can be interpreted suchas: *sp* = shoot on the gate, *so* = shoot shipped outside the frame of the gate.
- *T = {fu, fga, fro }*. Sets of colors that paint the locations *NFUS*, *NCG* and *NCR*, by which are specified the gravity of fouls committed by own team, colors that can be read such as: *fu* = easy foul game , *FGA* =foul of yellow card, fro = foul of red card.

The expressions on the arcs are steady multisets such as *1 `n*, variables such as x which takes values in the multisets of colors representing the players or the logical conditions, by which the arcs are weighted with multisets over the color crowds presented above, or with the multiset "empty".

We also use the variables *(x, y, z, t, w, s)* that take values in the sets of colors A, B, F, T, G, N and respectively S, variables that decide the passes type, the foul severity ,the efficiency of shoots at the gate, etc.

3 Description of Football Game Evolution

The football game has the following issues:

At some point the ball is in player's possession from MLJ location, who leads the ball and reaches at the JCM location.

In this moment of the game, the player found in possession of the ball can go on like one of these situations:

a) make a wrong pass (*pgr* - send the ball to an opponent). In this case, the first player who can recover the ball, is the player that appears in the location *MLJ*.

b) make a good pass (pbu - sends the ball to a teammate). If the pass is good, then the teammate that receives the ball shoot at gate :

b1) eighter shoots wrong (*so*),the ball recover can be made by the player that appears in the location *MLJ*.

b2) or shoots on the gate (*SPP*) - the ball recovery after it is released by opponent goalkeeper can be made by the player in the location *MLJ*.

It is likely that the player that has the ball to not be able to pass before being foul by the opponent (*fad*), or possibly commited a foul himself (*fpr*).

If he's foul, then is executes a free kick, no mather what kind of foul it was, and the ball will reach the teammate that appears in the location *MLJ*.

If the player that has the ball commits a foul, then the opposing team will have a free kick, no mather the kind of foul it was, and the first player who can recover the ball at this moment is the one that appears in the location *MLJ*.

Depending on the real situation on the field, during the game, the locations that provides statistical data (those written in block letters in the circles of the network, less *MLJ* and *JCM*), starting from the initial value zero, increments with the value *1* each time when an action occurs from transition that is connected to by an weighted arc, with the instruction that decide this and it also marks with a black token (*1 `n*) the pair location that allows sequel :

(ie $NFC \leftarrow NFC+1$ *and* $fpr \leftarrow fpr+1$, *when* $z = fpr$).

4 Description of Coloured Network with which is modelled the System Evolution

We consider the locations in this order, given by the vector *: (fpr, cr, cg, fusor, fad, pgr, pbu, fdj, fcg, NFCR, fcr, NSPP, NSSP, ssp, MLJ, JCM, NPgr, NPbu, NFS, NFDJ, NFCG, NFC, NCR, NFUS, NCG).*

We start from the initial state of the game given by the initial marking:

$\mu_0 = (0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1`mor,0,0,0,0,0,0,0,0,0,0)^t$ the marking expressing that the ball is in the player`s possession "mor". Zero indicates the empty multiset.

The unique transition that is possible at this time is "dribbling". By producing this transition the resulted marking is:

$\mu_1 = (000000000000000,0,1`mor,000000000)^t$

the marking which express that the player "mor" is preparing to pass the ball to a teammate.

On this marking are allowed transitions: "passes" and "foul".

If the transition "passes" occurs, then the player having the ball forwards the ball to:

- a teammate (pbu). In this case we obtain the marking:

  $\mu_2 = (000000,1`n,00000000000,1`pbu,0000000)^t$

- or send wrong the ball (pgr). In this case we obtain the marking :

  $\mu_3 = (00000,1`n,00000000000,1`pgr,00000000)^t$

At the marking μ_2 the unique transition which is allowed is "recuper", that ,by occuring, shows the next own player who recovers the ball after a wrong pass of a teammate.

At the marking μ_3 the unique transition which is allowed is "sutlp", which indicates the possibility of the shoot to the opposing gate of a teammate, who receivesd a good pass.

We continue analogous by producing the transitions corresponding to those actions that indicates the real state of the field. By the occurance of these transitions, each time is obtained the marking that specifies the status on the field.

Network coloured[1, 2] (Fig. 1) which modeles the football game was drawn through the CPN-Tools.

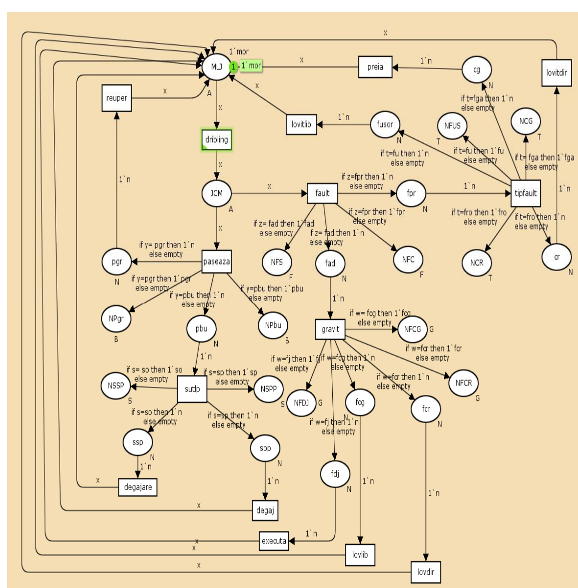

Fig. 1

Also with CPN-Tools has been generated the covering graph, which contains inside the nodes, the corresponding markings of all states through which the game has been until a certain moment. First part of this graph is shown in Fig. 2.

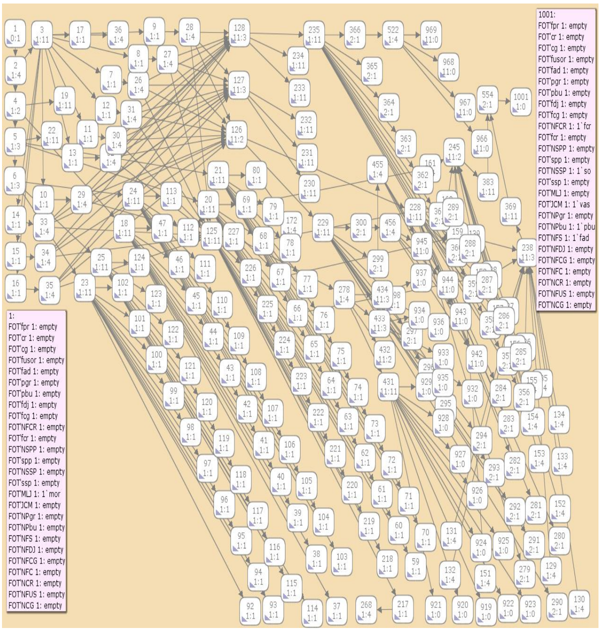

Fig. 2

Analyzing the marking from a top of the graph we can obtain useful information about red cards received by own team, and also about those received by the opposing team, about the quality of the passes made by own team, etc. at the corresponding time of top of the graph.

Using the CPN-Tools[4] we can obtain the covering graph, following a fixed number established before of transitions (arcstop), after the occurance of a fixed number established before of game states (nodestop) or after a certain time expressed in seconds expires (seccstop).

All these parameters can be set using CPN-Tools, if it is followed the path :

ToolBox ↣ State Space ↣ calcSS, where we can set the following variables: nodestop, arcstop or seccstop at the desired values.

In our example we set these variables with corresponding values :

- nodestop : 1000
- seccstop : 20.

CPN-Tools[4], has build covering graph to find its 1000 nodes or until it expired the established time (20 seconds).

Also with CPN-Tools[4] we obtained a report that provides detailed information about :

- State of the coverage graph (fully or partially) and the number of nodes and the number of own arcs.
- Number of nodes and arcs of reduced digraph, associated with it.
- Minimum values and maximum values in the locations made at the network evolution.
- Markings the trap[3] (the states which blocks the football game).
- Dead transitions[1, 2] (actions of the game which never occur).
- Information about fairness [1, 2] of the game actions (fair transitions), etc.

Report containing all the information, is show in Fig. 3.

```
Statistics
  State Space                                    Scc Graph
     Nodes:  1001                                   Nodes:  1001
     Arcs:   2116                                   Arcs:   2116
     Status: Partial                                Secs:   0
Boundedness Properties
  Best Integer Bounds
                    Upper      Lower
FOT'JCM  1            1          0             FOT'NSSP 1        2          0
FOT'MLJ  1            1          0             FOT'cg   1        1          0
FOT'NCG  1            2          0             FOT'cr   1        1          0
FOT'NCR  1            2          0             FOT'fad  1        1          0
FOT'NFC  1            2          0             FOT'fcg  1        1          0
FOT'NFCG 1            2          0             FOT'fcr  1        1          0
FOT'NFCR 1            2          0             FOT'fdj  1        1          0
FOT'NFDJ 1            2          0             FOT'fpr  1        1          0
FOT'NFS  1            2          0             FOT'fusor1        1          0
FOT'NFUS 1            2          0             FOT'pbu  1        1          0
FOT'NPbu 1            2          0             FOT'pgr  1        1          0
FOT'NPgr 1            3          0             FOT'spp  1        1          0
FOT'NSPP 1            2          0             FOT'ssp  1        1          0
  Best Upper Multi-set Bounds
FOT'JCM1=1`ion++1`vas++1`mar++1`ghe++1`flo++1`zac++1`con++1`mut++1`dud++
1`lun++1`mor
FOT'MLJ1=1`ion++1`vas++1`mar++1`ghe++1`flo++1`zac++1`con++1`mut++1`dud++
1`lun++1`mor |
FOT'NCG  1          2`fga                      FOT'cg   1        1`n
FOT'NCR  1          2`fro                      FOT'cr   1        1`n
FOT'NFC  1          2`fpr                      FOT'fad  1        1`n
FOT'NFCG 1          2`fcg                      FOT'fcg  1        1`n
FOT'NFCR 1          2`fcr                      FOT'fcr  1        1`n
FOT'NFDJ 1          2`fj                       FOT'fdj  1        1`n
FOT'NFS  1          2`fad                      FOT'fpr  1        1`n
FOT'NFUS 1          2`fu                       FOT'fusor1        1`n
FOT'NPbu 1          2`pbu                      FOT'pbu  1        1`n
FOT'NPgr 1          3`pgr                      FOT'pgr  1        1`n
FOT'NSPP 1          2`sp                       FOT'spp  1        1`n
FOT'NSSP 1          2`so                       FOT'ssp  1        1`n
Home Properties
  Home Markings                 None
Liveness Properties
  Dead Markinngs                447 [999,998,997,996,995,...]
  Dead Transition Instances     None
  Live Transition Instances     None
Fairness Properties             No infinite occurrence sequences.
```

Fig. 3

5 Presentation of modeling results

Analyzing the information provided by the report presented in Fig. 3, is seen that the covering graph has 1001 nodes, 2116 arcs and it is incomplete.

Maximum values in the locations are presented in the following vector :

(1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 1, 2, 1, 2, 1, 1, 1, 3, 2, 2, 2, 2, 2, 2, 2, 2).

The significance of these values is described in terms of multisets by the vector:

(1`n, 1`n, 1`n, 1`n, 1`n, 1`n, 1`n, 1`n, 2`fcr, 1`n, 2`sp, 1`n, 2`so, 1`n, 1`team, 1`team, 3`pgr, 2`pbu, 2`fad, 2`fj, 2`fcg, 2`fpr, 2`fro, 2`fu, 2`fga), where :

1`team=1`ion++1`vas++1`mar++1`ghe++1`flo++1`zac ++1`con++1`mut++1`dud++1`lun++1`mor.

All this tell us that during the game occured 2116 game actions and the game went through 1001 states. At this time all players have participated to the game, the team has sent only 2 good passes, 3 passes were made wrong, were shot to the gate 2 times and 2 were shot outside the gate. Were made 2 fouls of the game, same did the opposing team, received 2 yellow cards and two red cards, as well as the opposing team.

Network doesn't have marking the trap, has 447 dead markings, and the last are: 999, 998, 997, 996, 995, etc., doesn’t have dead transitions, doesn’t have liveness transitions and has no infinite sequences of transitions occurences, and therefore cannot give information about the accuracy of the football game actions.

Thus, to determine the state of game at any desired time, the corresponding marking for that moment is analised. For example, the marking (1001) corresponding to the following state of the game:

$$\mu_{1001} = \begin{pmatrix} 0,0,0,0,0,0,0,0,0,1`fcr,0,0,0,1`so,0,0, \\ 1`vas,0,1`pbu,1`fad,0,1`fcg,0,0,0,0 \end{pmatrix}^t$$

This is the state when the player "vas" has tha ball, the team had a shoot outside the gate, was made a good pass, and the opponents committed a foul by red card.

6 Conclusion

The model presented in this article allows to a football coach (the model can be easily transformed and adapted for any other team sports) to know at any time, the efficiencyand also the errors that occur in his team game- number of good passes, efficiency of the shots at the gate, fouls taken, fouls given, etc.. In addition, if it`s assumed that the team consists of only 2 players, it can track their behavior more closely in a small football game.

References:

[1] K. Jensen, “*Coloured Petri Nets”*, Vol 1,2,3 Spinger-Verlag Berlin Heidelberg, 1992.
[2] K. Jensen, L. M. Kristensen,*“ Coloured Petri Nets-Modelling and Validation of Concurrent Systems”*, Spinger-Verlag Berlin Heidelberg, 2009.
[3] G.W.Brams, “*Reseaux de Petri : Theorie et pratique*”, Vol 1,2, Masson, Paris, 1983.
[4] http://www.cs.au.dk/CPnets/CPN-Tools, 2009.