OXSY 2021 Team Description

Sebastian Marian, Dorin Luca, Bogdan Sarac, Ovidiu Cotarlea

OXygen-SYstems laboratory, Str. Constantin Noica,
Bl.5, Sc.C, Ap.36, C.P. 550169, Sibiu, ROMANIA
Email: (sebastian.marian@oxsy.ro) Web: (http://oxsy.ro)

Abstract. Oxsy team was founded in July 2002 for a graduation project of one student, Sebastian Marian, in the field of Multi-Agent Systems [1], at the Department of Computer Science of Lucian Blaga University (Sibiu - Romania). After graduation he continued the work on this project and so was born Oxsy team. As we started from scratch, our ideas, concepts and beliefs, was implemented year by year and today, we are happy to see that we are on the right way, as our team was growing in these years, more than we expected from the beginning. If we will qualify to the competition, this year we'll reach at the 17^{h} participation, in RoboCup [4] Soccer Simulation League.

Keywords: RoboCup, 2D Soccer Simulation, Coach, Neural Network, Offside, RSA accumulator.

1 Introduction

In July 2003 at RoboCup [4] competition, which was held in Padua - Italy, we won the first round and for us it was a good surprise for first year of participation. Then, in the next year, we participated in Lisbon - Portugal for the second time, and again we obtained a good result (the 11th place). In 2005 in Osaka – Japan, we participated for the third time and finally we entered in the first 8 teams of soccer simulation league, as we won (the 8th place). In 2006 the competition was held in Bremen – Germany and we won (the 7th place). In 2007 we went to Atlanta – Georgia (U.S.A), where we obtained (the 5th place), the same result which we achieved in 2008 in Suzhou – China. Finally, in 2009 in Graz, we entered in the first 3 teams in the soccer simulation league, as we won (the 3rd place), the same result which we achieved in 2010 in Singapore. In 2011 we came back from Istanbul - Turkey with (4th place). In 2012 we were in Mexico City, where we had a bad experience as we made some major changes in our defensive system, and also many others overall our team strategy, changes which was not very well balanced at that time, with all others characteristics of our team, as we were not qualified for finals, from the second round groups. In 2013 we came back in top, as we won (the 6th place), in Eindhoven – Netherlands. In 2014 the competition was held in Joao Pessoa – Brazil, and we entered on the stage for the third time in our participation history, as we won again (the 3rd place). In 2015 we won the 4th place as we played the semifinals in Hefei – China. In 2016 the competition was held in Leipzig – Germany, we missed the semifinals and we came back with (5th place). In 2017 we came from Nagoya with (3rd place) for the fourth time in

our participation history. In 2018 the competition was held in Montreal – Canada and we won the 4th place as we played in the semifinals. In 2019, unfortunately, our team missed the competition which was held in Sydney – Australia. In 2020 the competition was not held due the pandemic situation but this year was decided that the competition will be held exclusively online. As we already have a very good experience in 2D Soccer Simulation league, we hope that our new ideas and improvements will be reflected in the competition where we will also test other tactical elements developed.

2 Using RSA accumulator to recognize and handle some past situations

2010 was the first year when we have involved the coach in our team strategy. Beside of his classical attributions, of changing player types or recognizing opponent player's type, which already were implemented, we felt that we can use it more efficiently, in order to give some tactical advises during the game. As the coach has the privilege to receive full visual information, without any kind of noise, we can use it to make an opponent modeling. In fact, we believe that it is more important to adapt the strategy during the game, instead of before it starts. We also think that importance of the coach is not speculated very well right now, and maybe it will be a good point for research, not only for our team, but also for all the teams involved in soccer simulation. So, on one hand based on some typical neural networks [3], [5] that we developed to be used by the coach in some specific way and on the other hand based on the power of the coach, who has a full view of the whole field without any kind of noise, here is a review of most important things that our coach performs today:

- Adaptive offside trap [2], in defensive phase [6], based on opponent's attackers behavior modeling.
- Adaptive selection of the next action, in offensive phase, based on the starting cluster and on its route scoring.
- Creating more spaces behind the opponent defense line, by recognizing the pattern of opponent's defending style and also by finding theirs weak points, in this way our offense clearly advised by the coach, should creates more chances to score goals.
- Obtaining a good world model, for the opponent's players positioning, in both phases of the game, depending of the game phase and the position of the ball, which will be useful in offensive phase for the pass decision, when the world model of the opponent's players, generated by sensor information is incomplete, and in defensive phase for the defending strategy, in the same situation of incomplete world model information of our opponent.
- Choosing the best position of our offenders, in spaces created between or behind the opponent's defenders, when the ball is controlled by a player from our team, depending of the position of the ball.
- Deciding if we should use offside trap with current opponent.
- Changing our team shape in the field, to achieve some tasks which can not be per formed, with opponent that we are facing, using actual formation positioning.

This year we decided to implement and use the RSA accumulator [7] in our strategy. As it is a very novel domain we introduced it in something that could be very easily observed and attested. So, for the first time we introduced RSA accumulator to help

us in changing our team shape in the field during the game, depending of the situation and the opponent we are facing.

2.1 The RSA accumulator

The RSA accumulator is based on modular exponentiation with an RSA modulus. In its simplest form, it works as follows. The accumulator key is an RSA modulus N = pq, where p and q are strong primes [8], and a base x ∈ ZN. The modulus should be at least k bits, where k is the number of bits in the largest element that will be accumulated. The accumulation function computes the accumulation value for a set P = {p1, . . . , pn} of prime numbers as:

$$acc(P) = x^{(p1 \cdots pn)} \bmod N \quad (1)$$

The witness-generation function computes the witness Wpi,P for element pi in P by accumulating all elements of P except pi :

$$Wpi,P = x^{(p1 \cdots pi-1 pi+1 \cdots pn)} \bmod N \quad (2)$$

Finally, the authentication function authenticates an element pi and a witness Wpi,P with an accumulation acc(P) by testing:

$$(Wpi,P)^{pi} \equiv acc(P) \bmod N \quad (3)$$

Prime Representatives. It is important to note that the inputs to this accumulator must be restricted to prime numbers in order for it to be collision-free. Since most practical uses of accumulators need to be able to accumulate arbitrary integer values, it is necessary to compute a prime representative of each desired input to use as the actual input for the RSA accumulator.

One method of computing prime representatives, proposed by Sander, Ta-Shma, and Yung in [9] and described by Goodrich, Tamassia, and Hasi´c in [10], is based on two-universal hash functions (introduced by Carter and Wegman in [11]). It involves defining a two-universal function h(x) = Fx, where F is a k × 3k binary matrix, and searching for a prime 3k-bit preimage of a k-bit element e by sampling O(k^2) times from the set of inverses h^−1(e).

The second method of computing prime representatives was described by Bari´c and Pfitzmann, also in [12]. They refer to it as the “RSA Accumulator with Random Oracle,” but it is essentially the same as the standard RSA accumulator with a random oracle prime representative generator.

2.2 Changing our team shape in the field using RSA accumulator

As we are facing during one game, with many different situations in both phases, defensive and offensive, we must handle them correctly. This can be done if we will dynamically change our team strategy during the game. Changing our team shape is an important element of the team strategy. Depending of the situation, defensive or offensive, changing the shape it helps us to obtain better results than those which were obtained before changing it.

For example if you are facing one team with a great ability of defending, or maybe one that is defending with many players, you probably should come with many players in theirs own third in case you want to be sure, that you will be able to score against it. In the same way, if you are facing with a good attacking team, maybe you must reconsider your defense, and probably you have to reshape your team if you want to face up on theirs many attacking situations. The answer of this question:

Which is the best shape in one given phase of the game?, should come from the coach, who has to analyze the game and come up with the best solution in any situation and moment of the game.
Here we introduced this year the RSA accumulator to hold a "memory" of a good handled situations. Actually we have two RSA accumulators, one for defensive and one for offensive phase. What the coach actually does is that he is tracking the game, and when a situation of defending / attacking is handled with success he will create a map, where the key is the opponent action pattern clustered in some special structures which are holding position transitions of each opponent player, and the value is the current formation used, loaded from a predefined set of formations. This key is then hashed in a byte array and added into the dedicated accumulator. One of the most important propriety of the RSA accumulator is that the inclusion or non-inclusion proof of one set in the current accumulator can be done faster using Proof of Exponentiation, based on Wesolowski's proof. Simply explained, more details can be seen in the Appendix A., the coach could easily determine if the current situation which need to be handled is included or not in the accumulator. If it is included, he then can take the value of this key, which is the formation that should be used in this situation. Using this approach we observed a good improvement of our team during some games, when in some certain situations the coach changed the formation and adapted it dynamically to can handle better some almost identical past situations, included in the RSA accumulator.

3 Future work

For the next future, we will involve our coach in many other issues, where the team really needs his help. Even if the free form messages, are limited by count and periods of sending, the power of the coach remains very important, as he can receive free-noise information. In this way, he can analyze many important aspects of the games and if he will deliberate based on these information, he can give valuable advice to his own team. We must accept that right now, many teams involved in this competition, adapt their strategy before the start of the game instead of while it is running. A team will be more powerful, if it can adapt correctly its strategy depending on the opponent's behavior and not by the opponent's name, and also if it can do this during the game and not only before its start. In this way, we tried to adapt our team to some unexpected situations, which are generated by differently playing style of our opponents. In the real soccer, the role of the coach during the game is very important, and his importance is motivated not only because of the players that he is changing, but because of many good advice that he gives to his team during the game. In the same way, we must really think to the power of the coach and how we can involve him, more and more, in our simulator.

References

1. Peter Stone Layered Learning in Multi-agent System [D]. Pittsburgh: school of computer science, Carnegie Mellon University, 1998.
2. Soccer Tactics. An analysis o attack and defense, by Lucchesi Massimo, originally printed in Italy – 1999 by Edizioni Nuova Prhomos Via O.Bettaccini.

3. M. Riedmiller and Artur Merke, " Using machine learning techniques in complex multiagent domains," In I. Stamatescu, W. Menzel, M. Richter and U. Ratsch, editors, Perspectiveson Adaptivity and Learning, LNCS, Springer, 2002.
4. Itsuki Noda et al, Soccer Server Manual, RoboCupFederation. http://www.robocup.org.
5. Y. Jinyi, C. Jiang, and S. Zengqi. An application in RoboCup combining Q-learning with Adversarial Planning. In the 4th World Congress on Intelligent Control and Automation (WCICA'02), Shanghai, China, June 2002.
6. Pressing, by Lucchesi Massimo, editing Bryan R. Beaver, printed by DATA REPRODUCTION Auburn, Michigan.
7. Real-World Performance of Cryptographic Accumulators, Edward Tremel
8. Alfred J Menezes, Paul C Van Oorschot, and Scott A Vanstone. Handbook of applied cryptography. CRC press, New York, 1997.
9. Tomas Sander, Amnon Ta-Shma, and Moti Yung. Blind, auditable membership proofs. In Yair Frankel, editor, Financial Cryptography, number 1962 in Lecture Notes in Computer Science, pages 53–71. Springer Berlin Heidelberg, January 2001
10. Michael T. Goodrich, Roberto Tamassia, and Jasminka Hasi´c. An efficient dynamic and distributed cryptographic accumulator. In Agnes Hui Chan and Virgil Gligor, editors, Information Security, number 2433 in Lecture Notes in Computer Science, pages 372–388. Springer Berlin Heidelberg, January 2002.
11. J Lawrence Carter and Mark N Wegman. Universal classes of hash functions. Journal of computer and system sciences, 18(2):143–154, 1979.
12. Niko Bari´c and Birgit Pfitzmann. Collision-free accumulators and fail-stop signature schemes without trees. In Advances in Cryptology – EUROCRYPT'97, pages 480–494. Springer, 1997.

Appendix A. RSA accumulator construction

Summary:

- one-way RSA function a->g^a mod N
- set $\{a_1,a_2,...a_n\}$ is compactly represented by the accumulator $A = g^{(a_1 * a_2 * ... * a_n)}$
- witness w for element a_i is built like A but without a_i element
- check witness by checking $w^{a_i} = A$

Setup:

- Choose RSA modulus N = p*q where p, q are secret large primes
- H: Hash function that maps into primes
- $A_0 = g \in Z_N$ (initial state)

Add(A_i,x):

- $A_{i+1} = A_i^{H(x)}$

Del(A_i,x):

- $A_{i+1} = A_i^{1/H(x)}$

Convention: x = H(x)

Add set of values $S=\{s_1, s_2, ..., s_n\}$:

- $u = s_1 * s_2 * ... * s_n$
- $A_t = g^u$

InclusionProof(A,x):

- $\pi = A^{1/x} \in G$
- Computed using trapdoor(p,q) or you have to know the whole set

Verify(A,x,π):

- $\pi^x = A$

ExclusionProof(A,x):

- $A = g^u$
- a * x + b * u = gcd(x,u) = 1 (Bézout coefficients)

Aggregate inclusion proof:

- $\pi_1^x = A$, $\pi_2^y = A$
- Shamir's trick: a * x + b * y = 1 -> $\pi_{1,2} = \pi_1^b * \pi_2^a$
- $\pi_{1,2}^{x * y} = A$

Delete with trapdoor(A_t,x):

- $A_{t+1} = A_t^{1/x}$ (using knowledge of p,q)

Delete with inclusion proof(A_t,x,π)

- $A_{t+1} = \pi$ ($\pi = g^{u/x}$)

Batch delete(A_t,x,y,π_1,π_2)

- Compute $\pi_{1,2}$ s.t. $\pi_{1,2}^{x * y} = A_t$
- $A_{t+1} = \pi_{1,2}$
- (no state, no trapdoor, asynchronous)

Faster verification using Proof of Exponentiation based on Wesolowski's proof