CSIT Conference 2019, Yerevan, Armenia, September 23-27

Knowledge-Based Solvers for RGT Combinatorial Problems

Sedrak Grigoryan
Division of Computational and Cognitive networks, IPIA, NAS RA
Yerevan, Armenia
e-mail: addressforsd@gmail.com

Nairi Hakobyan
Department of System Programming, RAU
Yerevan, Armenia
e-mail: hakobyannairi@gmail.com

Tadevos Baghdasaryan
Division of Computational and Cognitive networks, IPIA, NAS RA
Yerevan, Armenia
e-mail: tadevos.baghdasaryan@gmail.com

ABSTRACT
In this work, we present means for complete at the time usage of RGT Solvers, that include knowledge acquisition, matching and decision making algorithms for RGT (Reproducible Game Trees) problems, particularly from the given specification of the problem it acquires knowledge and provides for adequate to expert decisions.
The RGT Solvers are being regularly improved and the current implementation tries to cover the drawbacks of previous versions, improved presentation of classifiers and mental doings, as well as enhancements in interface.
We also discuss ways of knowledge acquisition for marketing, battlefield problems by RGT Solvers.

Keywords
Competition, strategy, planning, chess, expert systems.

1. INTRODUCTION
1.1. Description of RGT Solvers, achievements.
We develop tools for solving RGT [1-3] (Reproducible Game Trees) problems in adequate to expert ways. RGT is a class of problems, which includes the following requirements:
- there are (a) interacting actors (players, competitors, etc.) performing (b) identified types of actions in the (c) specified types of situations;
- there are identified utilities, goals for each actor;
- actions for each actor are defined.

1.1.1. Many urgent problems of combinatorial nature, including marketing and management, network protection from various types of intrusions by hackers [4], chess and chess like problems [1], certain problems of decision making in battle fields [5], marketing (competing in oligopoly marketing environment) and management (supply chain management case, as an example) problems [12], etc. RGT problems are reducible to each other, particularly, to some standard kernel RGT problem K, e.g., chess [2, 5]. Thus, we also follow the common approach in experimenting knowledge-based systems for chess [6].
1.1.2. RGT Solvers aim to solve such problems from specification and expert knowledge bases to search for optimal strategies.
Generally talking, RGT Solvers consist of the following main modules: 1) Interface to interact with human, insert knowledge, provide situations and get the output for executed algorithms, 2) Knowledge base, which acquires the knowledge, adequate constructions and modules for effective usage of them, particularly in matching to situations, 3) Decision making modules, which provide adequate to human decision making algorithms in such problems.
1.2. Background of knowledge base.
1.2.1 Human deals with realities, some of which are not classified while human mainly deals with classified ones.
Classified realities can be divided into regularized and not regularized.
1.2.2. A mighty way of enhancement of effectiveness of mental systems is the regularization of classifiers induced by mdoers and mental systems.
Namely, classifiers ‘Cl’ of members ‘x’ of communities ‘C’ are regularized in ‘C’ if accompanied by ontological in ‘C’ methods, instructions allowing ‘x’ regularly provide positive samples of inputs of ‘Cl’ as well as let the members of ‘C’ do the same by communicating with ‘x’ [7].
1.2.3. In the mentioned approach it was stated that: i. Algorithms are modeling and constructively regularize deterministic methods, ii. OO Languages are constructively regularized and strongly expand algorithms, iii. Mentals are constructively regularized and strongly expand OOL, iv. Mentals can consist of functional and connectivity mental models.
Natural languages contain a large number of constructively classified mentals, e.g., English has about 300000 classifiers.
Algorithms are type of systems constructively modeling computational mental doings over numeric input IDs of realities and OO languages expand them by adding attributing/have, parenting/be and do relations.
1.3. Planning algorithms adequate to human planning are used to construct strategies. Plans are certain general descriptions of strategies. Each plan represents a hierarchy of goals. Those goals are attempted to be achieved in given situations, while playing by the plan. The essences of the plans are to select the goals, which get the maximal profit from the current plan aims.
Realizing the current plan the shell can determine the goal in the agenda, which in its turn determines basic attributes to be considered followed by indication of the arguments of those attributes.
1.4. Based on the provided background we demonstrate the adequate abilities of knowledge acquisition, matching and decision making algorithms, particularly strategy searching based on provided expert knowledge.
We also provide evidence of adequacy of developed models to expert knowledge specifications for problems other than experimental chess problem, including marketing and battlefield problems.
Particularly in contrast with previous versions of Solvers, we aim to provide completely systemic classification with integrated TZT in decision making algorithms.

2. KNOWLEDGE PRESENTATION IN RGT SOLVERS
RGT problem acquisition from experts and decision providing based on a certain problem specification requires several steps. In the following section we provide detailed description of how problems, strategy related knowledge and situations are integrated in RGT Solver and how situations are matched.

2.1. Acquisition of RGT problems
Developed RGT Solvers are able to acquire the knowledge from experts by the provided interface.
Based on this we concentrate on developing adequate constructive mentals that extend OOP class and aim to

provide natural language classifiers abilities, thus currently provide certain types of classifiers with have/be/do appropriate for now relations, which can be extended per required new classifier relations by experts.

Each classifier defined by the expert can contain its relations with other classifiers, have virtualization similar to OOP virtual/abstract classes and provide regularities and dependencies required to classify an instance of that.

As showed in [8] RGT Solvers define several types of classifiers to make it adequate to expert classifiers presentation, where definition of each problem starts from the simplest ones, or in nuclears in Solvers. Nuclear classifiers provide simple classification based on regularity defined in it or based. Compositions of nuclears can describe basic classifiers, i.e., the simplest ones that appear in situations (e.g., figures in chess, battling units in battles).

2.1.1. To achieve a regularity of expert knowledge acquisition for RGT Games an interface was developed. The interface was designed to acquire an expert knowledge in a form of patterns (classifiers) and transform situations from natural to symbolic presentations. Classifiers are used to define classes as well as operations, thereby providing a considerable uniformity of the structure of the language. Classifiers are composed from attributes, which can be filled with objects that instantiate other abstracts.

2.1.1.1. New classifiers can be built based on already existing ones by both composition and inheritance. The derived classifiers inherit all the attributes of the parent class including incorporations, as well as enable activation of actions defined for the parent. It is also allowed to add new attributes and modify the relations between attributes of the child.

2.2. Experimenting

Let's discuss some examples for more clarity.

2.2.1. The nuclear classifiers for chess are coordinates (X and Y), figure type and figure color, combination of which provides description of the chess field and the situation is provided by 64 fields descriptions, thus, 64 tuples of dimension 4.

Nuclear classifiers include one rule.

We demonstrate acquisition of classifiers relations by the example of chess classifier 'field under attack of knight' (figure). It is a composite classifier, which is a derivative of 'field under attack' and includes attributes (relation have) a) 'field' and b) 'move knight' where the 'field' inside 'move knight' ('moveKnight.field') is the same as the 'field' inside this classifier. That condition is being checked in the rule of nuclear attributes of 'fieldUnderAttackofKnight.field'.

2.2.2. At this stage, we put some limitations on battlefield problem, like discretizing actions and limitations on newly appearing units.

Here we consider the following nuclear classifiers: unit type, unit location (coordinates in the given situation) and whom the unit belongs to. In this term, this is similar to chess, and the basic knowledge acquisition and situation input is being done similarly to it.

We also add some more nuclears, such as power of attack, minimal and maximal distances of attack and radius of damage, "health" of the unit (also discrete).

Military units are considered as minimal classifiers. Actions for which are "attack" and "move" for each type of military unit and the postcondition for attacks can be noting that the target is destroyed or damaged.

2.2.3. Marketing and management problems can also be brought to the RGT class by splitting their specifics into corresponding terms of RGT meaning and defining them to meet the needed requirements of the class [12]. The goals in these problems are to get aimed values of profit, market share, achieving company stable state, enough conditions for survival in the market, needed rating in the market, etc. As a management goal, it might be an acceptable level of feedback, low wastes in spending, time or/and other resources, etc.

2.2.3.1. Classifiers for marketing problem are: a) companies (competing against each other), which are actors with certain nuclears depending on their type, b) actions, which are changes of the product price and quality, c) situations, which are determined by the states and actions of competition. On each turn, states are determined by the set of parameters of current competition and scenarios of it, i.e., the competition template formed from the conceptual basis of management theory.

2.2.3.2. The management problem is considered on the example of customer-company-supplier model. There are two separate parts in this model – in the first one, the company deals with customers and in the second it deals with suppliers.

In this model several companies (agents) compete in a market of certain product that can be assembled in predefined number of configurations. Configurations depend on types of limited main supply components used. Every main component is being produced by the number of supplier brands and available in couple of qualitive options (highspeed/lowspeed, big/small, cheap/expensive, etc.). Besides, there are limitations in compatibility between certain main components.

The classifiers here are a) customers, which compose the demand of different types of product, b) companies (agents), with their daily assembly abilities, bank accounts and warehouse, c) suppliers, which produce the supply components with their daily production capacity number of quality options for that components. The actions defined for different classifiers above are like "offer a product with X configuration and Y price", "request the X component from Y supplier for Z price", "accept offered X component from Y supplier for P price", "produce X number of product from listed components during N days", etc.

2.3. Matching classifiers to situations

As described above, situations present groups of nuclear classifiers of specified situations, however, natural presentation of situations is different, thus, tools for regular transformation of input situations to solver required symbolic presentation are developed.

2.3.1. Situation insertion interface

RGT situations are presented in a symbolic way including groups of nuclear classifiers. We have developed a system for transforming natural presentation of situations, which we consider as images to their symbolic presentation. The system and algorithm, on which it is based, are provided in [10].

The system uses Neural Network MobileNet SSD [11] for detecting actors, particularly, figures in chess and military units for battlefield problem. Afterwards, by comparing its coordinates it creates the symbolic presentation of situation, which is considered as a situation input in RGT Solver.

We have developed a situation transformation system for

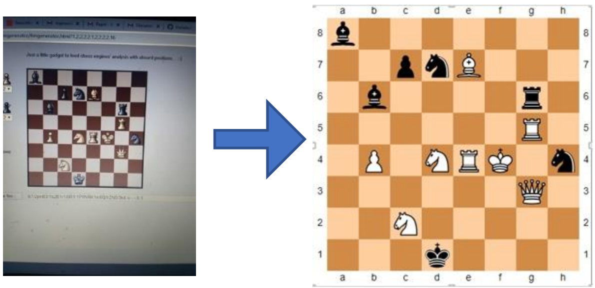

Fig. 1 Result of chess situation transformation

chess and battlefield: a) input for chess is an image with the existing chess board and b) input for battlefield problem is a screenshot from strategic game “Generals: Command and Conquer”.

2.3.1.1. The transformation system includes detection of objects (figures, military units) from the given image. Afterwards, the system calculates the approximate shape of the field and by comparison, it constructs the initial field (board, military area) based on the detected objects' coordinates.

For chess, the NN was trained on ~1000 images including the chess board of different chess figure and board shapes, which ensures the universalization of the model. Accuracy of the model is high, approximately 95 % of figures match to their exact place on final board.

2.3.1.2. For Battlefield problem, the NN was trained on ~500 images of screenshots from the game ‘Generals: Command and Conquer”, including 17 different classes of vehicles and military units: tanks, rocket launchers, soldiers, medical vehicles, etc. The model is not as accurate as in chess due to some similarities of units (for example, some types of tanks), some similarity between not marked and marked objects etc. However, the results of NN are acceptable at this stage: detection precision is 76.3%, detection recall is 87.4% and classification accuracy (for correctly detected figures) is 96.9%. Currently, the model is being enhanced by more data collection and annotation that is more detailed.

2.3.2. Matching Algorithm

For the already symbolic presentation of situations, Solvers are able to match developed classifiers in two main directions: 1) matching network of classifiers and 2) matching certain required classifiers [9].

2.3.2.1. Matching of nuclears is as simple as checking the rule in it.

Minimal classifiers and composites are matched by both children and attributes full set matching. Children also activate their virtual parents.

Actions are matched by matching of precondition and the actor. Once activated they can be applied on the situation to change it.

Dynamic classifiers are matched if precondition is matched and postcondition for all of the leaf situations is matched.

Goals precondition shall be matched in order to consider the goal doer, after which a game tree is generated and final situations are evaluated if an evaluation function is defined.

2.3.2.2. Certain given classifier matching is matched only by the subnetwork of classifiers, which relates directly to the given classifier, and matching is performed in the reverse direction (starting from the required classifier to the nuclears, which are needed to match it).

3. DECISION MAKING AND ACTION SELECTION FOR GIVEN SITUATIONS

For the strategy construction Personalized Planning and Integrated Testing (PPIT) algorithms are used, which create strategies using plans and are adequate to expert approaches. Plans are certain general descriptions of strategies. Each plan represents chains of goals sorted by their priorities. Those are the goals, which are attempted to be achieved in given situations while playing by the plan. The PPIT program was designed in [1] and integrated in Solvers were provided in [3]. We enhanced Solvers by trajectory-zones-based technique (TZT) [4] algorithms, which provides adequate solutions for achieving a goal originally suggested to estimate utilities of captures only of the opponent pieces. For example, to choose capture with max utility TZT chains the moves to each piece of the opponent (trajectories) without accounting possible handicaps for real capturing then using all available knowledge “plays the zones” of the game tree induced by the trajectories followed by estimation of their values to choose the best.

Realizing the current plan the shell can determine the goal in the agenda, which, in turn, determines basic attributes to be considered followed by indication of the arguments of those attributes. In the following, we provide integration of TZT into Solvers.

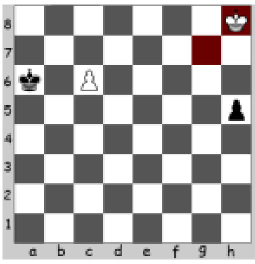

Fig. 2 Retie etude situation, white to move

3.1. Execution of PPIT

The algorithm of the realized PPIT algorithm is detailed in [3]. So for each Plan that is appropriate for the given situation by some classifiers (a precondition for the plan), it starts searching actions that bring to the goal with the highest priority in the chain. If the goal is achieved the processing of plan is finished, otherwise further goals are being processed.

Processing of goals in previous versions of PPIT [4] was implemented by generating trees without deep analysis of classifiers in it. The current goal achievement algorithms will be described in the next section.

The adequacy of PPIT algorithms are experimented in chess, particularly in Retie etude [1, 3], for which the algorithm using the given plan can find a solution by processing ~15 times less positions than chess engines, such as Stockfish chess engine. The goal is to achieve a draw in the given Figure 2 situation.

The considered number of positions is brought in Figure 3.

	RGT Solver18
Move 1	22
Move 2	27
Move 3	27
Move 4	10
Move 5	9
Move 6	9
Total	104

Depth	Considered Nodes
1	36
2	136
3	570
4	1088
5	1414
6	2183
7	3309
8	4005
9	6717
10	9256
11	16987

Fig. 3 Considered positions in Retie etude solution left: in RGT Solver, right: Stockfish engine

3.2. Integration of TZT

For going deeper into the game tree, various types of tree cutting solutions are applied, including alpha-beta pruning. For chess engines based on some defined knowledge they might drop many branches and achieve big depths. We follow the line suggested by Botvinnik [6] and developed within the intrusion protection problem algorithms [4], which rely on trajectories of attacks and zones of counteractions named TZT. Idea of TZT is defined in [6] and enhanced in [4].

3.2.1. We implement the TZT solution into RGT Solvers PPIT algorithms to achieve goals in more effective ways.

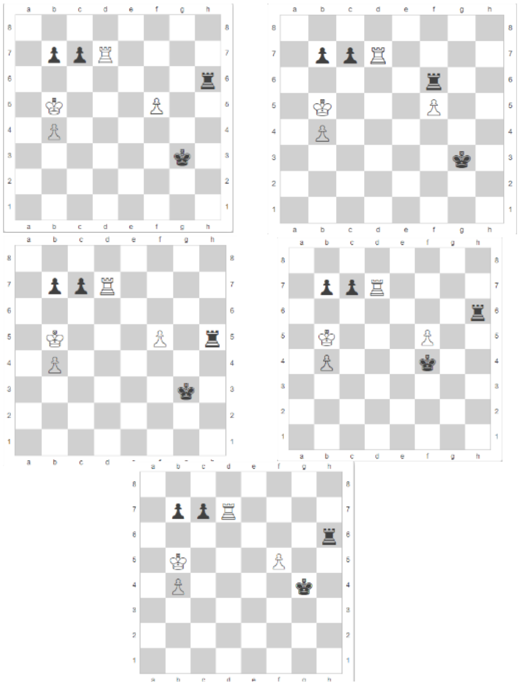

Fig. 4 Example of processing TZT

So goal achievement checking is performed by the following steps: a. Generation of a tree of situations with the defined depth that leads to goal achievement, b. Extension of situation chains by all possible counteractions by the opponent, c. Checking if goal can be achieved and evaluation if defined.

Let us consider the following example (Figure 4) where black goal is to take white pawn in vertical ‘f’. Let us consider generation of 6-depth tree; evaluation of situation is done by material values. Following sequence of actions by achieve the goal: 1) Rf6-> Rxf5, 2) Rh5->Rxf5, 3) Kf4->Kxf5, 4) Kg4-> Kxf5, etc.

At step a. algorithm generates all the possible sequences of situations with the maximum (depth of tree) / 2 distance that can achieve the goal. The restriction of (depth of tree) / 2 is because it can perform only that number of actions in the tree, the rest will be opponent’s actions.

At step b. algorithm generates all the possible counteractions by opponent for each sequence of actions and actions opposing counteractions of opponent.

1)Rf6->Rd5->(c6+->Kb6->cxd5+->Kxb7)->Rxf5, 2) Rh5->Rd5->(c6+->Kb6->cxd5+->Kxb7)->Rxf5, Rh5->Rf7->Kf4->Kc5->Rxf5 3)Kf4->Rd4->(c6+->Kb6->cxd5+->Kxb7)->Kxf5, Kf4->Rf7->Rh5->Kc5->Rxf5 4) Kg4->Rd4->(c6+->Kb6->cxd5+->Kxb7)->Kxf5, Kg4->Rf7->Rh5->Kc5->Rxf5

3.2.2. It is obvious that there are many actions not considered when generating the tree for the goal (we can consider 4 different situations at first level, however there are 25 situations). If for the given situation we assume average number of moves by a side is ~20, the overall 6 depth tree would have 20^6 situations by brute force processing, however if we consider ~4 different situations on each level it would have 4^6 situations, which is a radically smaller number of situations.

At step c. all the final situations are evaluated to consider the best ones and select the branches as the best, accordingly suggesting the actions that bring to the expected best branches. This list is passed to the next goal in the plan chain as it is described in [1, 3] and similarly being processed with steps a, b, c for the goal.

4. CONCLUSION

a. RGT Solver is able to acquire problems and provide adequate to expert matching to classifiers based on systemic classification, while situations provided by natural presentation, say an image can successfully be transformed to Solver required presentation by the developed tools in Solver interface.

b. Integrated planning algorithms for decision making are described including PPIT and TZT algorithms by effective adequate to experts approaches.

c. The ability of acquisition of various RGT problems, including chess, consumer dialogue, battlefield, marketing and management problems are demonstrated.

5. ACKNOWLEDGEMENT

Authors would like to thank to Professor Edward Pogossian for supervising this work.

REFERENCES

[1] E. Pogossian, V. Vahradyan, A. Grigoryan, “On competing agents consistent with expert knowledge”, *Lecture Notes in Computer Science, AIS-ADM-07: The International Workshop on Autonomous Intelligent Systems - Agents and Data Mining*, pp. 229-241, 2007.

[2] E. Pogossian, “Specifying Personalized Expertise”, *International Conference Cognition and Exploratory Learning in Digital Age* (CELDA), pp 151-159, 2006.

[3] S. Grigoryan, “Structuring of Goals and Plans for Personalized Planning and Integrated Testing of Plans”, *Mathematical Problems of Computer Science, vol. 43*, pp. 62-75, 2015.

[4] E. Pogossian, A. Javadyan, E. Ivanyan "Effective Discovery of Intrusion Protection Strategies". *Workshop on Agents and Data Mining*, pp. 263-274, 2005.

[5] E. Pogossian, D. Dionne, A. Grigoryan, J. Couture, E. Shahbazian, "Developing Goals Directed Search Models Empowering Strategies Against Single Ownership Air Threats," *International Conference in Computer Sciences and Information Technologies*, pp. 155-163, 2009.

[6] M. Botvinnik, “Computers in chess: solving inexact search problems”, *Springer Series in Symbolic Computation, with Appendixes, Springer-Verlag*, 1984.

[7] E. Pogossian, “Constructing Adequate Mental Models”, *Mathematical Problems of Computer Science, vol. 50*, pp. 35-51, 2018.

[8] S. Grigoryan, N. Hakobyan, H. Vrtanesyan, “Object-Oriented Modeling of Matching to Systemic Classifiers”, *Mathematical Problems of Computer Science, vol. 49*, pp. 115-121, 2018.

[9] S. Grigoryan, N. Hakobyan, “Experimenting with Acquisition of and Matching to Systemic Classifiers”, *Mathematical Problems of Computer Science, vol. 50*, pp. 96-103, 2018.

[10] N. Hakobyan, “A System for Transforming Images to Symbolic Presentation for Combinatorial Defense and Competition Problems”, *NAS RA and NPUA Series of technical sciences, vol. LXXII, N2*, pp. 199-209, 2019.

[11] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, Ch. Fu, A. Berg, “SSD: Single Shot MultiBox Detector”, *ECCV*, 2016.

[12] T. Baghdasaryan, “Adapting RGT Solver Interface to Management Strategy Search Problems”, *Mathematical Problems of Computer Science vol. 39*, pp. 135-145, 2013.