EasyChair Preprint

№ 4812

A Domain on Set of Possible Inputs to a Game Transition Functions.

Frank Appiah

December 28, 2020

A DOMAIN ON SET OF POSSIBLE INPUTS TO A GAME TRANSITION FUNCTIONS.

FRANK APPIAH.

KING' COLLEGE LONDON, SCHOOL OF ENGINEERING, ENGLAND, UNITED KINGDOM.

frank.appiah@kcl.ac.uk

appiahnsiahfrank@gmail.com.

Extended Abstract[+]. This research is on determining the possible inputs to a game domain from its transition functions. This is based on [1], that describes a transition function of a game developed in a programming language, Java.

Keywords. transition, game, function, domian, lifecycle, remote, command, range, inputs.

Year of Study: 2020 **Year of Publication: 2020**

1 *AFFILIATE. UNIVERSITY OF LONDON, KING'S COLLEGE LONDON, DEPARTMENT OF ENGINEERING, LONDON, UK.

1 INTRODUCTION

A state transition diagram[2] of the Game Automaton(G*A*)[2,3,4] is called GM_1. It has five states labeled *GUI, CP, PTA , PDE and PPA* and five conditions labeled *PV, VE, PU, TR and EA*. The start state of WFA is GUI and it is normally indicated by a pointing arrow to the state GUI[5]. The accept state is the *GUI* state and it is normally

indicated by double circle/round-shape around the state. The arrows moving from one state to another is called transitions. When the automaton receives an input string {*GUI, CP, PTA , PDE, PPA*}, it processes that string and produces an output.

The output is either accept or reject. The processing begins in $\mathbf{GM_1}$ start state. The automaton receives the symbols from the input string one by one from left to right. After playing the symbols, GM_1 moves from one state to another along the transition that has symbol as its label. When it plays the last symbol now it is in the accept state. The processing of GM_1 as follows:

1. start in state GUI;
2. play PU, follow transition from GUI to CP;
3. play TR, follow transition from GUI to PTA;
4. play TR, follow transition from CP to GUI;
5. accept because GM_1 is in accept state GUI;
6. play TR, follow transition from PTA to PPA;
7. play VE, follow transition from PTA to GUI;
8. accept because GM_1 is in accept state GUI;
9. play EA, follow transition from PTA to PDE;
10. play VE, follow transition from PDE to PTA;
11. play EA, follow transition from PPA to PTA.

The machine processing(MP) of GM will now be represented by domain rules.
A domain[7] Rule(DR) has form:

$$play : A\ X\ B \rightarrow C \quad ,$$

where A, B and C are actions and effects in a game.

1. $play : GUI\ X\ \mathrm{CP} \rightarrow \mathrm{PU}$,after MP 2.
2. $play : GUI\ X\ PTA \rightarrow \mathrm{TR}$,after MP 3.
3. $play : CP\ X\ \mathrm{GUI} \rightarrow \mathrm{TR}$,after MP 4.
4. $play : PTA\ X\ \mathrm{PPA} \rightarrow \mathrm{TR}$,after MP 6
5. $play : PTA\ X\ GUI \rightarrow VE$,after MP 7.
6. $play : PTA\ X\ PDE \rightarrow EA$,after MP 9.

7. $play: PDE\ X\ PTA \rightarrow VE$,after MP 10
8. $play: PPA\ X\ \mathrm{PTA} \rightarrow EA$, after MP 11.

2 TRANSITION FUNCTION AND DOMAIN

The transition function[2, 3] is used to define the rules of moving. The notation of the transition function is δ(state, input)=state. The transition functions for remote commanding are as follows:

1. δ(GUI, PU) = CP
2. δ(GUI, TR) = PTA
3. δ(CP, TR) = GUI
4. δ(GUI, PU) = CP
5. δ(PTA, TR) = PPA
6. δ(PTA, TR) = CP
7. δ(PTA, VE) = GUI
8. δ(PTA, EA) = PDE
9. δ(PDE, VE) = PTA
10. δ(PPA, EA) = PDA.

The transitive domain will be represented by:

$State: Input \rightarrow State$

1. $CP: GUI \rightarrow PU$
2. $PTA: \mathrm{GUI} \rightarrow \mathrm{TR}$
3. $GUI: \mathrm{CP} \rightarrow \mathrm{TR}$
4. $CP: \mathrm{GUI} \rightarrow \mathrm{PU}$

5. $PPA:\ \mathrm{PTA} \rightarrow \mathrm{TR}$
6. $CP:\ \mathrm{PTA} \rightarrow \mathrm{TR}$
7. $GUI:\ \mathrm{PTA} \rightarrow \mathrm{VE}$
8. $PDE:\ \mathrm{PTA} \rightarrow \mathrm{EA}$
9. $PTA:\ \mathrm{PDE} \rightarrow \mathrm{VE}$
10. $PDA:\ \mathrm{PPA} \rightarrow \mathrm{EA}$.

3 CONCLUSION

This research is about transitive domain of game automaton developed from earlier research in [1]. Here, there is an enumeration of domain functions rule form in two different of models. The first, $play: A\ X\ B \rightarrow C$ is based on the machine processing statements and second, $State: Input \rightarrow State$ is based on state-input transition functions that gives the rules of movement in an automaton[3]. There is about 18 domain possibilities that rules only in the game domain in this short paper.

Compliance with Ethical Standards

(In case of funding) Funding: This is research is funded by King's Alumni Group, Association of Engineering with ISAreference grant number: 204424 20821845.

Conflict of Interest:

Author, Dr. Frank Appiah declares that he has no conflict of interest.

REFERENCES

1. Appiah F. (2020). A Computational Model of Life Cycle of Game Actions and Effects. KCL School of Engineering.
2. Michael Sipser(1997). Introduction to Theory of Computations. PWS Publishing Company.
3. HopCroft, J. E. and Ulmann, J. D(1979). Introduction to Automata theory, Languages and Computation. Addison Wesley.
4. Roche, E. and Schabes, Y(1997). Finite-State Language Processing. MIT Press.
5. GUI (2020). Java AWT, Oracle Corporation USA.
6. Enderton H. B (1972). A mathematical Introduction to Logic. Academic Press.
7. Micheal Sipser(1997). An Introduction to Computation Theory. PWS Publishing House.