Apply Cultural Algorithm to Coevolution of Multi-Agent System*

Jong Yih. Kuo [1] Hsing Chang Lai [2]

[1]Department of CSIE, National Taipei University of Technology, Taipei, Taiwan, Email: jykuo@csie.ntut.edu.tw
[2] Department of CSIE, Fu Jen Catholic University, Hsien Chung, Taiwan, Email: peice93@csie.fju.edu.tw

Abstract

We proposed a framework called Cultural Coevolutionary Multi-Agent Model (CCMAM) which combines cultural algorithm model and Coevolutionary model with a 3APL-based Multi-Agents platform. Our purpose is to develop coevolvable multi-agent model, which includes two features; (1) to improve the capability of agent or agent group, (2) to improve the cooperation between agents, via coevolvable cultural algorithm model. The application domain is a pursuit-evasion game includes three groups of teams: pursuers, evaders and obstacles. Results indicate that the approach applies cultural algorithm to both competitive and cooperative coevolution process at the same time yields ascendant performance comparative to apply only either competitive or cooperative approach.

Keywords: 3APL, Coevolution, Cultural Algorithm, Pursuit-Evasion Games

1. Introduction

On recent years, agent's evolution was fast developed and applied to many different domains, such as agent learning, agent group, and agent negotiation [5], etc. In this paper, we propose a Cultural Coevolutionary Multi-Agent Model (CCMAM) which combines coevolution with cultural algorithm and groups then into a model. All cultural algorithms were introduced by Reynolds as a vehicle for modeling social evolution and learning. Those algorithms include two spaces, belief space and population space, respectively, where the population space is an evolutionary population whose experience of individuals are integrated into belief space consisting of various forms of symbolic knowledge. We develop a cultural coevolutionary algorithm by employing cultural algorithm as the basic mechanism for agent evolution and obtain experience of individuals by means of cooperating or competing with other agents. Moreover, we apply our approach to 3APL platform which its mental state is implemented base on BDI theory [1] and has robust mechanism of agents' communication mechanism. Beside, we adopt a pursuit-evasion game to be the problem domain; the pursuit-evasion game includes two competitive groups, pursuers and evaders, where the pursuers need to cooperate so as to pursue evaders. Finally, the experiment result compares both cooperative and competitive approach with either only single approach; and encourages in many respects to discuss. We conclude our approach and illustrate the possibility of development in future.

2. Cultural Coevolutionary Multi-Agent Model

2.1. Model analysis

The cultural coevolutionary multi-agent model can be divided into two parts- internal of agent and external of agent. In the internal part, the individual agent's mental state is evolved by the cultural algorithm [9] model; and the external part is implements by coevolved coevolutionary Multi-Agent framework.

In internal part of agent, we delegate cultural algorithm to evolve goals and plans attitudes in the data structure of 3APL agent, i.e. for specific problem domain we apply cultural algorithm to obtain appreciate substitution of goal or plan instead using goal planning rule and plan reasoning rule. Every goal and plan in the data structure of 3APL agent is a concrete class; here is the definition of mental attitudes in [6],

<belief> ::=
  <ground_atom>“." |
  <atom> ": − "<literals>“."
<goal> ::=
  <ground_atom>("and"*<ground_atom>*)*
<plan> ::=
  <basicaction> | *<composedplan>*
<basicaction> ::=

* This work was supported by the Ministry of Economic Affair under grant 95-EC-17-A-02-S1-029 and Ministry of Education under grant EX-91-E-FA06-4-4.

"€" | <*Atom*> |
"Send("<*iv*>,<*iv*>,<*atom*>")"|
"Java("<*ident*>,<*atom*>,<*var*>")"|
<*wff*>"?"|<*atom*>
<*literal*> ::=
<*atom*> | "not("<*atom*>")"

The goal could be a (set) of ground atom which stand for some situation the agent want to realize; the basic action can be composed to build plans through so-called program operators (sequential operator, iteration operator and conditional choice) and a plan could be a composed plan represents a abstract plan. Therefore, we encode these goals and plans with their problem domain into the population space of cultural algorithm and then derive belief space (schemata) from the population space.

Moreover, we deliver the process of evolving agent to the mechanism of external actions on 3APL platform. It can be seen as a function call because of the mechanism of external action of 3APL platform. Fig 1 illustrate the cultural deliberation cycle which extend form original deliberation cycle in [3].

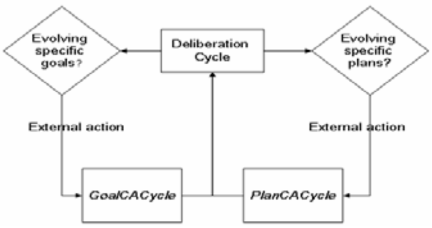

Fig. 1: Cultural deliberation cycle.

In the cultural algorithm model described above, the population space should need to obtain the fitness of individuals form environment. Our purpose is to change the cultural algorithm model and make it coevolvable; therefore, population should obtain its fitness by either directly compete with others or indirectly cooperate with others. According to the reason, we modify the original cultural algorithm model into coevolutionary cultural model, Fig 2 shows that the population space directly compete or indirectly cooperate with other population. In our coevolutionary cultural algorithm cycle, it not only supports both competitive and cooperative coevolution but also apply them in conjunction for yield superior performance comparative to single approach.

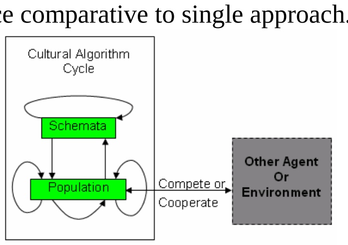

Fig. 2: Coevolutionary cultural Algorithm cycle.

Our approach is different with general coevolution approach where the cooperative and competitive coevolutions are proceeding at the same time. The cooperative mechanism is extended from Potter's standard CCGA [8] but sharing the same cultural network within homogeneous agent group, Fig 3 describes the pseudo code of cultural coevolution algorithm (CCEA).

```
Begin
  t=0;
  for each homogeneous subpopulation s
    initialize Pops(t);
    evaluate fitness in Pops(t);
  end for
  for each agent group g
    initialize Cultural Component Cg(t);
    end for
  while termination condition =false do
    t=t+1;
    for each homogeneous subpopulation s
      evaluate fitness of each individual on Pops(t);
      update(Cg(t-1), accept(Pops(t-1)));
      generate(Pops(t-1), influence(Cg(t-1))) ;
      select Pops(t) from Pops(t-1);
    end for
  end while
end begin
```

Fig. 3: Cultural coevolution algorithm (CCEA).

Both of competitive and cooperative behavior can be represents by following fitness function,

$$[Obs()*\alpha_1 + Cooperation(I)*\alpha_2 + Compete()*\alpha_3]^2$$

where the value of Obs() function is inverse proportioned to the average distance of obstacles and wall; the Cooperation(I) means how well the individual I cooperate with other agents; and the Cap()/Free() illustrates the average capture and free time on each round.

2.2. Model design

Fig 4 represents the architecture of cultural coevolutionary multi-agent model. The 3APL agent delegates the evolution of goals and plans to agent evolution component via external action. The agent evolution component include three subcomponents, Cycle Composition Component, Population Evolution Component and Belief Revision Component; the first component deals with incoming message and deliver to correct evolution cycle; the second component encodes incoming goals or plans with problem domain to construct population space and then derive belief space (schemata-oriented) from these individuals of population use [11] ; final component refines current belief base in 3APL agent via adaptation methodology.

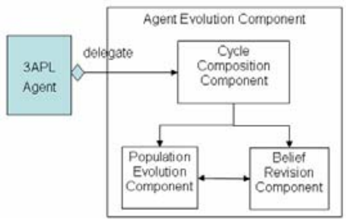

Fig. 4: Architecture of Coevolutionary Multi-Agent Model.

3. Case Study

In this Section we will introduce the problem pursuit-evasion game, formalize the problem domain and design a pursuit-evasion game system later.

3.1. A Pursuit-Evasion Game

We consider a two-player pursuit-evasion game [2] [11] between a team of pursuers and another team of evaders. The team of pursers needs to cooperate to pursuit evaders for win; and every evader is a self evolved agent which runs away over finite game area. Even though each agent is a single individual on the map, but the whole game is constructed by two competitive groups, team of pursuers and team of evaders. Fig 5 shows a visual view of pursuit-evasion game; the circular means pursuers, triangle represents obstacles and the X means evaders.

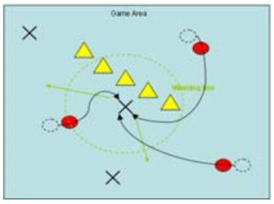

Fig. 5: Graphic view of Pursuit-Evasion game.

3.2. Formalization

The formalization focuses on three key point, the game area, evaders, pursuers and obstacle, and measurement space, respectively. We consider a 2-D environment X with n_c square cells and all event take place on a set of equally space event times $T=\{1,2,\ldots\}$ and a team of n_p pursuers, called player U and a team of n_e evaders, called player D. X_p (X_e) be the set of cells occupied by n_p pursuers (n_e evaders). Every pursuer and evader collects information at discrete time instant $t \in T=\{1,2,\ldots\}$, we denote X_p (t) $=(X_p^0$ (t), X_p^1 (t),..., X_p^{nc}(t))$\in X^{n_p}$ and similarity to X_e (t) the position at time t T of players U and D, respectively. Obstacles denote X_p(t)=(X_o^0 (t), X_o^1 (t),..., X_o^{nc}(t)), where X_o^i (t)=1 if cell i contains an obstacle at time t. The action $p \in P, e \in E$, where P and E denote the sets of actions available to player U and D. We denote Y and Z the measurement space for player U and D, respectively. In measurement space Y, $y(t)=\{p(t),e(t),o(t)\}, where\ y(t) \in Y$, where p(t) denotes the measured position of pursuers, e(t) and o(t) is a set of cells where the evaders and obstacles are detected, similarity to measurement space Z and z(t). y* means the set of all finite sequences of elements in Y, and $Y_t \in$ y* be the sequence of measurements { y(1), ⋯, y(t)} take up to time t; similarity to z*. Moreover, we denote the visibility region of pursuer k (evader i) as V_{pk} (t) (V_{ei} (t)). Fig 6 illustrates the game after formalization, where Y_0:={[(2,7)],[(2,4),(5,6)],[ ]};

{[(6,2)],[ ],[(3,3),(4,4),(5,5),(6,6)] };

{[(9,8)],[ ],[ ]};

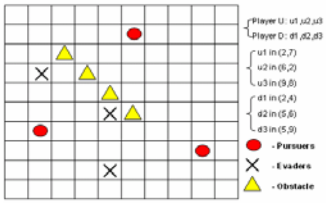

Fig. 6: Game environment after formalization.

3.3. System Architecture

We introduce the system architecture which consists of 3APL platform, agent evolution component and game environment in Fig 7.

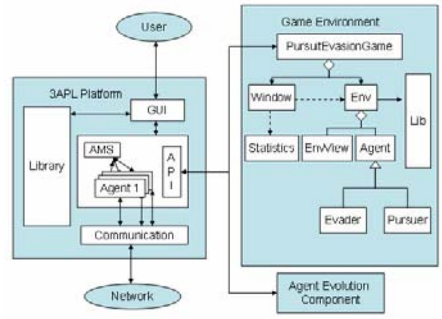

Fig. 7: System Architecture.

3APL platform consists of a number of agents, a directory facilitator called agent management system (AMS), a communication mechanism which delivers messages between agents and a plugin interface that allows agents to execute actions in the shared environment. Game environment consists of a bridge which implements plugin interface and connect environment with 3APL platform, called PursuitEvasionGame; moreover, classes Window, Env, EnvView, and Statistics realize the user interface of game.

4. Experiments

Fig 8 shows the realistic working environment of pursuit-evasion game, the green square means pursuers and the red square represents evaders; all the obstacles are shows by blue squares.

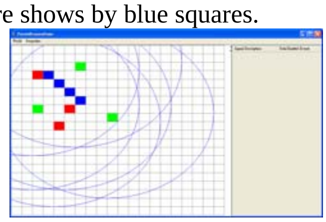

Fig. 8: Game Environment.

The application of cultural algorithm provides success moving strategies for pursuers to catch evaders and for evaders to run away over map within acceptable duration (every generation is completed within 42 ms and the evolution is stable after almost 6 generations). Fig 9 illustrates all the comparison of traditional GA and coevolved multi-agent methodology.

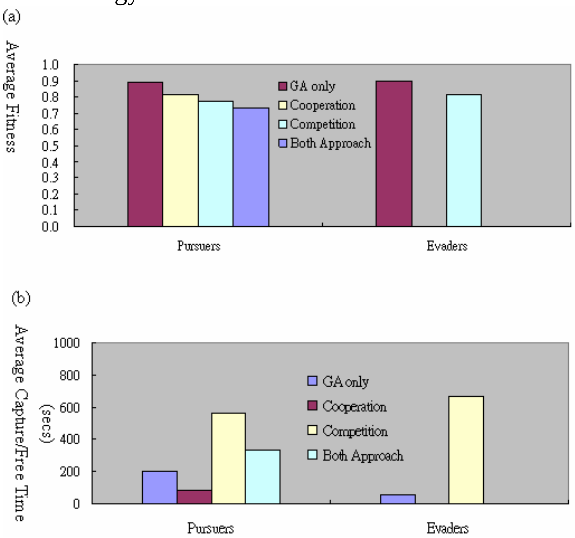

Fig. 9: (a) Average Fitness (b) Average Capture/ Free time

The result of experiment tells us that competitive coevolution approach is superior to cooperative coevolution approach; and the application of both approaches at the same time obtains the best performance. In comparison to the experiment result of [7], we can find a special situation that the average fitness value of applying both approaches within pursuers is lower than only applying single approach; the reason is when we apply more approaches to support coevolution process, the fitness value of poor individuals will pull down the whole average fitness value. Moreover, the average free time shows that competitive approach can greatly increase the survived time of evaders against to pursuers.

5. Conclusions

In this paper we introduce a coevolutionary multi-agent system to apply to a pursuit-evasion game. The framework includes a cultural based evolutionary methodology for evolving and improving of capabilities both agent individual and agent group. The experiment result illustrates that our approach is benefiting either competitive or cooperative coevolution of pursuit-evasion problem. In the future, we will focus on elasticizing the agent evolution component, relax restriction of the cooperation, and enhance power of competition.

6. References

[1] B. Frances, D.K. Barbara, T. Jan, V. Rineke, “Beliefs, Intentions and DESIRE”, *Proc. Of the 10th Workshop on Knowledge Acquisition*, 1996.

[2] J.P. Hespanha, M. Prandini, S. Sastry, “Probabilistic Pursuit-Evasion Games: A One-Step Nash Approach”, *Proc. Of the 39th IEEE Conference on Decision and Control*, Vol. 3, pp. 2272-2277, 2000.

[3] K. Hindriks, M.d' Inverno, M. Luck, “Architecture for Agent Programming Languages”, *Proc. Of the Fourteenth European Conference on Artificial Intelligence*, 2000.

[4] M.d’ Inverno, K. Hindriks, and M. Luck, “A Formal Architecture for the 3APL Agent Programming Language”, *Proc. Of the first International Conference of B and Z users*, 2000.

[5] R.Y.K. Lau, B. Essam, S.Y. Chan, “Belief revision for adaptive negotiation agents”, *Proc Of the IEEE/WIC International Conference on Intelligent Agent Technology*, pp. 196-202, 2003.

[6] D. Mehdi, M. Birna van Riemsdijk, M. John-Jules, “Programming Multi-Agent Systems in 3APL”, Available: http://www.computingscience.nl/people/mehdi/publication/ProMASin3APL.pdf

[7] G. Nitschke, “Co-evolution of cooperation in a pursuit evasion game”, *Proc. Of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, Vol. 2, pp. 2037-2042, 2003.

[8] M.A. Potter, K.A.D. Jong, “A Cooperative Coevolutionary Approach to Function Optimization”, *Proc. Of the International Conference on The Third Conference on Parallel Problem Solving from Nature of Evolutionary Computation*, pp. 249-257, 1994.

[9] R.G. Reynolds, B. Peng, “Cultural algorithms: knowledge learning in dynamic environments”, *Proc. Of the 2004 Congress on Evolutionary Computation*, Vol. 2, pp1751-1758, 2004.

[10] C. D. Rosin, Richard K. Belw, “New Methods for Competitive Coevolution”, *Jour. Of Evolutionary Computation*, 5(1), pp. 1-29 1997

[11] W. Sverdlik, R.G. Reynolds, “Incorporating Domain Specific Knowledge into Version Space Search”, Proceeding of TAI Fifth International Conference on Tools with Artificial Intelligence, pp. 216-223, 1993

[12] R. Vidal, O. Shakernia, H.J. Kim, D.H. Shim, S. Sastry, “Probabilistic pursuit-evasion games: theory, implementation, and experimental evaluation”, *Transactions on Robotics and Automation*, 18(5), pp. 662 – 669, 2002.