On Role of Implicit Interaction and Explicit Communications in Emergence of Social Behavior in Continuous Predators-Prey Pursuit Problem

Ivan Tanev[1] and Katsunori Shimohara[1, 2]

[1] ATR Human Information Science Laboratories,
2-2-2 Hikaridai, "Keihanna Science City", Kyoto 619-0288, Japan
[2] Graduate School of Informatics, Kyoto University,
Yoshida-Honmachi, Sakyo-ku, Kyoto 606-8501, Japan
{i_tanev, katsu}@atr.co.jp

Abstract. We present the result of our work on use of genetic programming for evolving social behavior of agents situated in inherently cooperative environment. We use predators-prey pursuit problem to verify our hypothesis that relatively complex social behavior may emerge from simple, implicit, locally defined, and therefore – robust and highly-scalable interactions between the predator agents. We propose a proximity perception model for the predator agents where only the relative bearings and the distances to the closest predator agent and to the prey are perceived. The instance of the problem we consider is more realistic than commonly discussed in that the world, the sensory and moving abilities of agents are continuous; and the sensors of agents feature limited range of "visibility". The results show that surrounding behavior, evolved using proposed strongly typed genetic programming with exception handling (STGPE) emerges from local, implicit and proximity-defined interactions between the predator agents in both cases when multi-agents systems comprises (i) partially inferior predator agents (with inferior moving abilities and superior sensory abilities) and with (ii) completely inferior predator agents. In the latter case the introduction of short-term memory and explicit communication contributes to the improvement of performance of STGPE.

1 Introduction

Over the past few years, multi-agent systems (MAS) have become more and more important in many aspects of computer science such as distributed artificial intelligence, distributed computing systems, robotics, artificial life, etc. MAS introduce the issue of collective intelligence and of the emergence of behavior through interactions between the agents. An agent is a virtual entity that can act, perceive the proximity of its environment and communicate with others; it is autonomous and has abilities to achieve its objectives. MAS contain a world (environment), entities (agents), relations between the entities, a way the world is perceived by the entities, a set of operations that can be performed by the entities and the changes of the world as a result of these actions. Currently, the main application areas of MAS are problem solving, simula

E. Cantú-Paz et al. (Eds.): GECCO 2003, LNCS 2723, pp. 74–85, 2003.

tion, collective robotics, software engineering, and construction of synthetic worlds [4]. Considering the latter application area and focusing on the autonomy of agents and the interactions that link them together [14], the following important issues can be raised: What is minimum amount of perception information needed to agents in order to perceive the world? How can agents cooperate? What are the methods, and what are the lower bounds of communications, required for them to coordinate their actions? What is the architecture they should feature so that they can achieve their goals? What approaches can be applied to automatically construct the agents' functionality, with the quality of such a design being competitive to the design handcrafted by human? These issues are of special interest, since the aim is to create MAS which is scalable, robust, flexible, and the able to automatically adapt to changes. These features of MAS are believed to be particularly important in real world applications where the approaches to construct synthetic worlds can be viewed as a practical methods, a techniques towards creating complex "situational aware" multi-computer, multi-vehicle, or multi-robot systems based on the concepts of agents, communication, cooperation and coordination of actions.

Within considered context, the *objective* of our research is an automatic design of autonomous agents which situated in inherently cooperative environment are capable of accomplishing complex tasks through interaction. We adhere to the methodological holism based on the belief that any complex system or society (Heraclitus, Aristotle, Hegel, and more recently – [12][13]), and multi-agent society in particular [7] is more than the sum of its individual entities, more than the sum of the parts that compose it. The social behavior, needed to accomplish the complex task might emerge in MAS from relatively simply defined interactions between the agents. We are particularly interested in the ultimate case of such simplicity – local, implicit, proximity defined, and therefore, robust, flexible and highly scalable interactions between the agents, situated in more realistic (than commonly considered), inherently cooperative environments. This document is intended to highlight the issues of applying genetic programming for investigating the sufficiency of implicit interaction and the role of explicit communication in emergence of social behavior in MAS.

The remaining of the document is organized as follows. Section 2 introduces the task which we use to test our hypotheses – an instance of the general, well-defined yet difficult to solve predator-prey pursuit problem. The same section addresses the issue of developing the software architecture of the agents. Section 3 elaborates the strongly typed genetic programming with exceptions handling (STGPE), proposed as an algorithmic paradigm used to evolve the functionality of agents. Empirical results are presented in Section 4 and conclusion is drawn in Section 5.

2 The Problem and the Agents Architecture

2.1 Instance of Predator Prey Pursuit Problem

The general, well-defined and well-studied yet difficult to solve predator-prey pursuit problem [2] is used to verify our hypothesis that relatively complex social behavior might emerge from simple, local, implicit, proximity-defined, and therefore – robust and highly-scalable interactions between the predator agents. The problem comprises four predator agents whose goals are to capture a prey by surrounding it on all sides in a world. In our work we consider an instance of the problem, which is more realistic than commonly considered in the previous work [5][6][9]. The world is a simulated two-dimensional continuous torus 1600mm x 1000mm. The moving abilities of four predator agents are continuous too – the predators can turn left and right to any angle from their current heading and can run with speed equal to 0, 0.25, 0.5, 0.75 and 1.0 of maximum speed. In addition, we introduce an proximity perception model for predator agents in that they can see the prey and *only the closest predator agent*, and only when they are within the *limited range* of visibility of their simulated (covering an area of 360 degrees) sensors. The prey employs random wandering if there is no predator in sight and a priory handcrafted optimal escaping strategy as soon as predator(s) become "visible". The maximum speed of prey is higher than the maximum speed of predator (i.e. predator-agents feature inferior moving abilities). In order to allow for predators to stalk and collectively approach the prey, in the first of the two considered cases the range of visibility of predators is more than the range of visibility of the prey (i.e. superior are only the sensory abilities of predators, therefore they can be considered *partially inferior*). In the second case the range of visibility of predators is equal to the range of visibility of the prey (i.e. *completely inferior* predator agents). We consider these two cases in order to create an inherently cooperative environment in that the mission of predators is nearly impossible unless they collaborate with each other. We are not interested in cases when predators are superior in their moving abilities, since the capturing of the prey in this case seems trivial, can be accomplished by single agent, and therefore does not require collective behavior from MAS. Analogically, the situation comprising completely inferior agents who besides being slower feature more myopic (rather than equal) sensory abilities than the prey is intractable for the conditions give above, and therefore is beyond our current consideration.

2.2 Architecture of the Agents

We adopted the subsumption architecture [3] of the agents comprising of functional modules distributed in three levels, corresponding to the three different aspects ("levels of competence") of agent's behavior: wandering/exploring, greedy chase and social behavior - surrounding (Figure 1a). Given that we focus our attention on evolving the top-level, highest priority module – surrounding the prey (assuming that the rest two modules are handcrafted), our objective of automatic design of autonomous agents via simulated evolution, which we declared earlier, can be rephrased as evolving the surrounding module in subsumption architecture of the agents.

In order to coordinate the functionalities of each of architectural modules we introduce the notion of agent's state. At every instant, the agent can be in one of the three states, corresponding to the module which is currently governing the agent's behavior: *surrounding*, *greedy chase* and *wandering/exploring*. The agent is in *surrounding* state if and only if there is a match (response) for the currently perceived proximity of the world (stimuli) in the functionality of evolved surrounding module. Being in *sur-*

rounding state, the agent is fully controllable by the functionality of the surrounding module, while functionalities of hunting and wandering/exploring modules are inhibited. If there is no match of the perceived proximity of the world in the functionality of evolved surrounding module, the agents state switches to *greedy chase* if prey is in sight or to *wandering/exploring* state otherwise (Figure 1b).

We would like to emphasize that the proposed implicit interstate-transition scheme is fully controllable by evolved functionality of the surrounding module, which allows to simultaneously evolve (i) the capability of agents to resolve social dilemmas, determined by the way *social behavior* overrides *greedy chase* when prey is in sight, and (ii) the capability to resolve the exploration-exploitation dilemma, determined by the ability of *social behavior* to override *wandering/exploring* when prey is invisible.

3 Algorithmic Paradigm Employed to Evolve Predator Agents

3.1 Strongly-Typed Genetic Programming with Exception Handling

Limiting the Search Space of Genetic Programming. We consider a set of stimulus-response rules as a natural way to model the reactive behavior of predator agents [7] which in general can be evolved using artificial neural networks, genetic algorithms, and genetic programming (GP). GP is a domain-independent problem solving approach in which a population of computer programs (individuals) is evolved to solve problems [8]. The simulated evolution in GP is based on the Darwinian principle of reproduction and survival of the fittest. In GP genetic programs (individuals) can be represented as parsing trees whose nodes are functions, variables or constants. The nodes that have sub-trees are non-terminals - they represent functions where the sub-trees represent the arguments to function of that node. Variables and constants are terminals – they take no arguments and they always are leaves in the parsing tree. The set of terminals for evolving agent's behavior includes the perceptions (stimuli), and the actions (response) which the agent is able to perform. The set of functions comprises the arithmetical and logical operators, and the `IF-THEN` function, establishing the relationship between certain stimulus and corresponding response(s). Without touching the details of representation of genetic programs, which will be elaborated later in Section 3.2, a human readable form of sample stimulus-response rule is shown in Figure 2. It expresses a reactive behavior of turning to the bearing of the peer agent (`Peer_a`) plus `10` (degrees) as a result of stimulus of its own speed being less than `20` (mm/s).

The strength of GP to automatically evolve a set of stimulus-response rules of arbitrary complexity without the need to a priory specify the extent of such complexity might imply an enormous computational effort caused by the need to discover a huge search space while looking for potential solution to the problem. Agreeing with [13] that for huge and multidimensional search spaces the introduction of "pruning algorithms" is a critical step towards efficient solution search, we impose a restriction on the syntax of evolved genetic programs based on some a priory known semantics. The approach is known as strongly typed genetic programming and its advantage over canonical GP in achieving better computational effort is well proven [11].

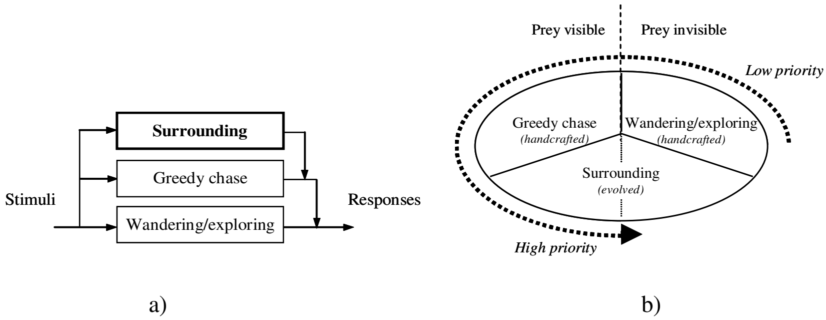


Fig. 1. Subsumption architecture of the agents: functional structure (a) and states (b)

```
IF (Speed<20) THEN Turn(Peer_a+10)
```

Fig. 2. Sample stimulus-response rule

Considering the sample rule, shown in Figure 2 it is noticeable that both the functions and their operands are associated with *data types* such as *speed* (e.g. `Speed`, `20`), *angle of visibility (bearing)* (`Peer_a, 10`), and *Boolean* (`Speed<20`). An eventual arbitrary creation or modification of genetic program semantically would make little sense: indeed, it is unfeasible to maintain Boolean expressions comparing operands of different data types at least because they have different physical units. Moreover, since we introduce sensor's range limits, there is a clear possibility of maintaining introns in genetic programs when, for example Boolean expressions include comparison of perception variable of certain data type with constant value beyond the limits of the data type of that variable (e.g. `Peer_d>1000`, in case that sensor range is only `400`). Analogically, the semantics of action `Turn()` implies a parameter of data type *angle*. And allowing just addition and subtraction as arithmetical operations implies that all the operands involved in the expression which defines the turning parameter should have the same data type *angle*. Addressing the mentioned concerns, the grammar of STGPE establishes generic data types of *visible angle*, *distance*, *speed*, and *Boolean* with the corresponding allowed ranges of the values for their respective instances (variables and ephemeral constants). In addition, it stipulates the data type of the results of arithmetical and logical expressions, and the allowed data type of operands (perception variables and ephemeral constants) involved in these expressions.

We would like to emphasize that proposed approach is not based on domain-specific knowledge, and therefore STGPE can not be considered as a "stronger" approach compromising the domain-neutrality of the very GP paradigm itself. The limitations imposed to the syntax of genetic programs are solely based (i) on the natural presumption that the predator agents are fully aware of their physically reasonable limits of his perception- and moving abilities; and (ii) on the common rule in strongly-typed 3G algorithmic languages that all the operands in addition, subtraction and comparison operations should have the same data types. In no way these limitations incorporate a priory obtained knowledge, specific for the domain or for the world where the agents are situated.

Exception Handling. The notion of exception handling is introduced in a way, much similar to the 3G algorithmic languages. In our approach, an *exception* is an event, raised when a runtime error occurs in an evaluation of the Boolean expression in the conditional part of `IF-THEN` rule. Due to the limited range of simulated sensors of predator agents, such an error would happen when the Boolean condition involving perception variable(s) related to perceiving another entity in the world (e.g. closest predator agent and/or the prey) can not be evaluated because the corresponding entity is currently "invisible". In addition to `IF-THEN` we introduce `IF-THEN-NA` (`IF-THEN-"not available"`) type of stimulus-response rule with exception handling capabilities. The human readable syntax and the corresponding semantics of sample stimulus response rule with exception are shown in Figure 3.

3.2 Main Attributes of STGPE

Function and Terminal Sets. Function and terminal sets of adopted STGPE are summarized in Table 1. Notice the local, proximity defined sensory abilities of agents.

Representation of Genetic Programs. Inspired by flexibility and recently emerged widespread adoption of document object model (DOM) and extensible markup language (XML), we represent genetic programs as a DOM-parsing trees featuring corresponding flat XML text. Our additional motivation stems from the fact that despite of the recently reported use of DOM/XML for representing computer architectures, source codes, and agents' communication languages we are not aware about any attempts to employ XML technology for representing evolvable structures such as genetic programs in generic, standard, and portable way. Our approach implies performing genetic operations on DOM-parsing tree using off-the shelf, platform- and language neutral DOM-parsers, and using XML-text representation (rather than S-expression) as a flat format, feasible for migration of genetic programs among the computational nodes in eventual distributed implementation of STGPE. The fragment of XML representation of the above discussed sample stimulus-response rule (refer to Figure 3) is shown in Figure 4. The benefits of using DOM/XML-based representations of genetic programs, as documented in [15] can be briefly summarized as follows: (i) XML tags offer a generic support for maintaining data types in STGPE; (ii) W3C-standard XML schema offers generic way for representing the grammar of STGPE; (iii) using standard built-in API of DOM-parsers for maintaining and manipulating genetic programs; (iv) OS neutrality of parsers; (v) algorithmic language neutrality of DOM-parsers, and (vi) inherent Web-compliance of eventual parallel distributed implementation of STGPE.

Genetic Operations. Binary tournament selection is employed – a robust, commonly used selection mechanism, which has proved to be efficient and simple to code.

```
TRY IF (Peer_d<20)
   THEN Turn(Peer_a+10)
EXCEPT  Turn(10);
```

a)

```
IF (Peer is Visible) THEN BEGIN
  IF (Peer_d<20) THEN Turn(Peer_a+10);
END ELSE Turn(10); // invisible preda-
tor agent
```

b)

Fig. 3. Syntax (a) and semantics (b) of sample stimulus-response rule with exception handling

Table 1. Function Set and Terminal Set of STGPE

Category		Designation	Remarks
Function set		IF-THEN, IF-THEN-NA LE, GE, WI, EQ, NE, +, -	IF-THEN without/with exception handling $\leq$, $\geq$, Within, =, $\neq$, +, -
Terminal set	Sensory abilities	Prey_d; Peer_d Prey_a; Peer_a PreyVisible; PeerVisible	Distance to the prey and to the closest agent, *mm.* Bearing of the prey and of the closest agent, *degrees* True if prey / agent is "visible", false otherwise
	State variable	Speed	Speed of the agent, *mm/s*
	Ephemeral constants	Integer	
	Moving abilities	Turn(α) Stop, Go_1.0 Go_0.25, Go_0.5, Go_0.75	Turns relatively to α degrees (α>0: clockwise) Sets speed to 0, Sets speed to maximum Sets speed to 25%, 50%, 75% of maximum

```
<IF-THEN-NA>
   <COND-THEN-NA><COND_TDistance>
       <VAR_TDistance>Peer_d</VAR_TDistance >
       <OPER TDistance>LE</OPER_TDistance >
       <CONST TDistance>20</CONST_TDistance >
   </COND_TDistance ></COND-THEN-NA>
   <THEN> ... </THEN>
   <NA>   ... </NA>
</IF-THEN-NA>
```

Fig. 4. Fragment of XML representation of sample stimulus-response rule

Crossover operation is defined in a strongly typed way in that only the nodes (and corresponding subtrees) of the same data type (i.e. labeled with the same tag) from parents can be swapped. The sub-tree mutation is also allowed in strongly typed way in that a random node in genetic program is replaced by syntactically correct sub-tree. The routine refers to the type of node it is going to currently alter and applies the randomly chosen rule from the set of applicable rules as defined in the grammar of STGPE. The transposition mutation also operates on single genetic program by swapping two random nodes having the same data type.

Breeding Strategy. We adopted a homogeneous breeding strategy in which the performance of single genetic program, cloned to all the agents is evaluated. Anticipating that the symmetrical nature of the world, populated with identical predator agents is unlikely to promote any specialization in the behavior of agents, we consider the features of such a homogeneous multi-agent society as (i) adequate to the world and (ii) consistent with our previously declared intention to create robust and well scalable multi-agent system.

Fitness Function. In order to obtain more general solutions to the problem the fitness of the genetic program is evaluated as average of the fitness measured over 10 different initial situations. However, based on the empirically proven data that on the initial stages of evolution agents are hardly able to successfully resolve more than few (out of 10) initial situations in order to enhance the computational performance of STGPE we applied the approach of noisy evaluation of the fitness function [10]. The amount of initial situations used to evaluate the genetic programs in population gradually increases as population evolves. Starting from 4 for the first generation of each run, the amount of situations is revised on completion of each generation and it is set to exceed by number of 2 the amount of situations, successfully solved by the best-of-generation genetic program. Given that with addition of another initial situation(s) they have to resolve, the agents would perform either better or, most probably worse, the fitness of the best-of-current generation could be occasionally somewhat worse than fitness of best genetic program from previous generation. Therefore, it is reasonable to expect non-monotonous fitness convergence characteristics of STGPE.

The fitness F measured for the trial starting with particular initial situation is evaluated as a length of the radius vector of the derived agents' behavior in the virtual energy-distance-time space as:

$$F = \sqrt{dE_A^2 + D_A^2 + T^2} \quad (1)$$

where dE_A is the average energy loss during the trial, D_A is the average distance to the prey by the end of the trial, and T is the elapsed time of the trial. The quantities dE_A and D_A are averaged over the all predator agents. The energy loss estimation dE for each of predator agents takes into account both the basal metabolic rate, equal to 0.05 units per second, and the energy loss for moving activities equal to 0.01 units per mm of path traversed during the trial. The trial is limited to 300s of "real" time or to the instance when prey is captured; and with sampling rate of 500ms it is simulated with up to 600 time steps. Smaller values of fitness function correspond to better performing predator agents.

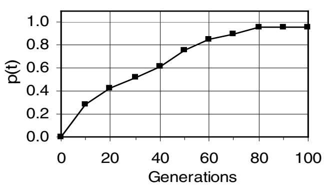
4 Empirical Results

4.1 Parameter Values of STGPE

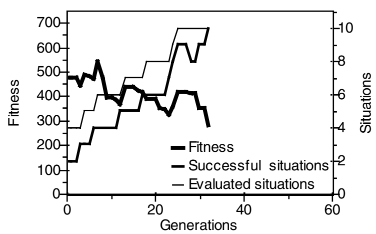
The parameters of STGPE used in our experiments are as follows: the population size is 400 genetic programs, the selection ratio is 0.1 (including 0.01 elitism), and the mutation ratio is 0.02, equally divided between sub-tree mutation and transposition. The termination criterion is defined as a disjunction of the following three termination conditions: (i) fitness of the best genetic program in less than 300 and the amount of initial situations in which the prey is captured (successful situations) equals 10 (out of 10), (ii) the amount of elapsed generations is more than a 100, and (iii) the amount of recent generations without fitness improvement is more than 16.

4.2 Partially Inferior Predator Agents

In the first case a superior sensory abilities of predators (range of visibility 400mm vs. 200mm for prey) and inferior moving abilities are considered (20mm/s vs. 24mm/s). The computational effort (amount of genetic programs needed to be evaluated in order to obtain the solution with specified probability, e.g. 0.95), is obtained from the probability of success p(t) by each of 20 independent runs in a way as suggested in [8]. The result, shown in Figure 5a indicates that p(t)=0.95 by generation 80 which yields a computational effort of about 32,000 genetic programs. Typical fitness convergence characteristic is shown in Figure 5b.



a)



b)

Fig. 5. Probability of success (a) and typical fitness convergence characteristic (b)

Human-readable representation of sample best-of-run genetic program is shown in Figure 6, and the traces of the entities in the world for one of the 10 initial situations is shown in Figure 7. The prey, originally situated in the center of the world, is captured by time step 140. The emergence of following behavioral traits of predator agents are noticeable: (i) switch from greedy chase into surrounding approach of agents #3 (time step 65, on right part of the world) and agent #2 (time step120, top left) as soon as other agents appear in sight; (ii) zigzag move by agent #0 which results a lower chasing speed indicating "intention" to trap the prey (after time step 100, far right and far left) and (iii) surrounding approach demonstrated by agents #1 and #3 during the final stages of the trial (top left).

4.3 Completely Inferior Predator Agents

In this case the same range of visibility of predators (400mm vs. 400mm respectively) and inferior moving abilities are considered (20mm/s vs. 24mm/s). These conditions definitely render the task more difficult for predator agents. As empirical results indicate that for the same values of GP-parameters (as sated in Section 4.1) a probability of success of 0.95 is hardly achievable. In order to illustrate the very ability of STGPE to discover the solution in these conditions we present plotted values of fitness of the best-of-run genetic program and the amount of successfully resolved situations for 20 independent runs. The results are shown in Figure 8a.

```
Program Main;
type TDistance =    0..400;
     TVisAngle = -180..180;
     TSpeed    =    0..22;
var  Peer_d, Prey_d          : TDistance;
     Peer_a, Prey_a          : TVisAngle;
     Speed                   : TSpeed;
     PreyVisible, PeerVisible : Boolean;
Procedure GP;
begin
 try if (Prey_a >= -26)
   then try if (Prey_a within -5)
     then begin
       if (not PeerVisible) then Turn(Prey_a);
       try if (Prey_a <= -139)
         then begin
           Null;
           Go_0.25;
         end;
       except Turn(-24-Peer_a+7-24);
     end;
   except Null;
 except try if (Peer_d <= 136) then Null; except Turn(Prey_a);
end;
begin // main program
  GP;
end.
```

Fig. 6. Human-readable representation of sample best-of-run genetic program

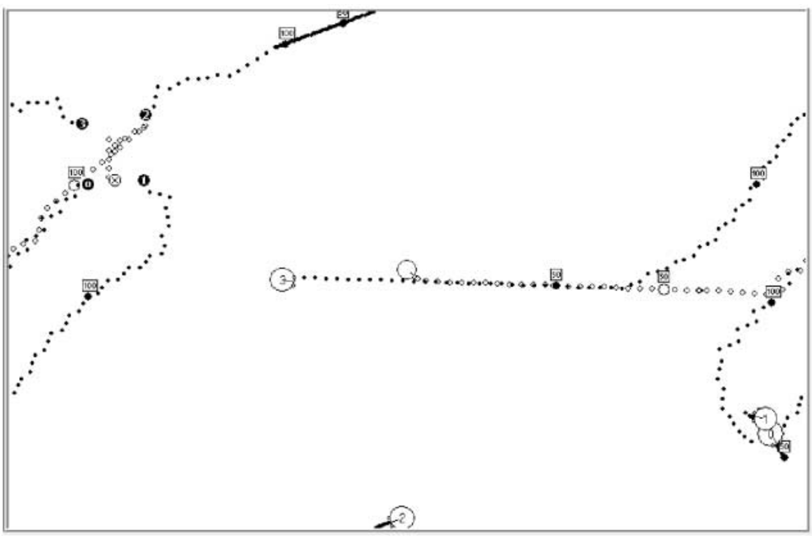


Fig. 7. Traces of the entities with agents governed by the genetic program shown in Figure 6. The prey is captured in 140 simulated time steps (top left). Larger white and small black circles denote the predator agents in their initial and final position respectively. The small white circle indicates the prey, initially situated in the center of the world. The numbers in rectangles show the timestamp information.

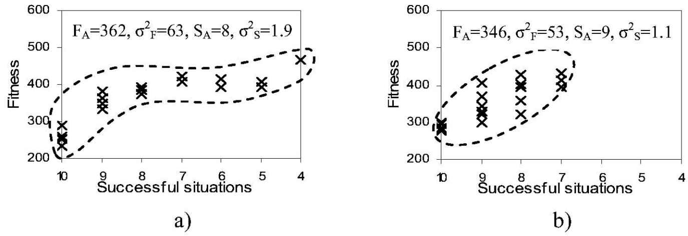


Fig. 8. Fitness and amount of successful situation of the best-of-run genetic program in 20 independent runs when agents are implicitly interacting (a) and when agents are employing short term memory with explicit communication.

As figure illustrates, only 25% of runs have been successfully completed (i.e. terminated by criteria (i) as described in Section 4.1) and the average and the standard deviation of fitness and successful situations are F_A=362, σ^2_F=63, S_A=8 and σ^2_S=1.9. The results with introduced short term (working) memory [1], storing the direction where in which prey has been recently seen and explicit communication allowing the exchange the currently or recently seen direction to the prey (i.e. predator agents still remain *mechanically* inferior) are shown in Figure 8b. Although the solution can be evolved by STGPE even with implicit interactions between the predator agents, introducing short term memory and explicit communication improves the performance of simulated evolution: 35% of 20 runs have been successfully completed with more favorable statistical results of F_A=346, σ^2_F=53, S_A=9 and σ^2_S=1.1.

5 Conclusion

We presented the result of our work on use of genetic programming for evolving social behavior of agents situated in inherently cooperative environment. We use predators-prey pursuit problem to verify our hypothesis that relatively complex social behavior may emerge from simple, implicit, locally defined, and therefore – robust and highly-scalable interactions between the predator agents. We proposed a proximity perception model for the predator agents where only the relative bearings and the distances to the closest predator agent and to the prey are perceived. The instance of the problem we consider is more realistic than commonly discussed in that the world, the sensory and moving abilities of agents are continuous; and the sensors of agents feature limited range of "visibility". Adopted subsumption architecture and developed implicit inter-state transition model allow for simultaneous evolution of the capabilities of predator agents to resolve both the social dilemma and the dilemma between exploration and exploitation. The empirical results show that surrounding behavior, evolved using proposed strongly typed genetic programming with exception handling (STGPE) emerges from local, implicit and proximity-defined interactions between the predator agents in both cases when multi-agents systems comprises (i) partially inferior predator agents (with inferior moving abilities and superior sensory abilities) and with (ii) completely inferior predator agents. In the latter case the introduction of short-term memory and explicit communication contributes to the improvement of performance of STGPE.

Acknowledgements. This research was conducted as part of "Research on Human Communication" with funding from the Telecommunications Advancement Organization of Japan.

References

1. Barborica, A., Ferrera, V.P., Estimating Invisible Target Speed from Neuronal Activity in Monkey Frontal Eye Field, Nature Neuroscience Vol.6, No.1 (2003) 66–74
2. Benda, M., Jagannathan, B. , Dodhiawala, R.: On Optimal Cooperation of Knowledge Sources. Technical Report BCS-G2010-28, Boeing AI Center, Boeing Computer Services, Bellevue,WA (1986)
3. Brooks, R.A.: A Robust Layered Control System for a Mobile Robot. IEEE Journal of Robotics and Automation, Vol.2, No.1 (1986) 14–23
4. Ferber, J.: Multi-Agent Systems: An Introduction to Distributed Artificial Intelligence, Harlow: Addison Wesley Longman (1999)
5. Haynes, T., Sen, S.: Evolving Behavioral Strategies in Predators and Prey (1996)
6. Haynes, T., Wainwright, R., Sen, S., Schoenefeld, D.: Strongly Typed Genetic Programming in Evolving Cooperation Strategies (1997)
7. Holand, J.H.: Emergence: From Chaos to Order, Cambridge, Perseus Books (1999)
8. Koza, J.R.: Genetic Programming: On the Programming of Computers by Means of Natural Selection, Cambridge, MA, MIT Press (1992)
9. Luke, S., Spector, L.: Evolving Teamwork and Coordination with Genetic Programming
10. Miller, B.,L., Goldberg, D., E.,: Genetic Algorithms, Tournament Selection, and the Effects of Noise, Illigal Report No. 95006, University of Illinois (1995).
11. Montana, D.: Strongly Typed Genetic Programming, Evolutionary Computation, Vol.3, No.2 (1995) 199–230
12. Morgan, C.: Emergent Evolution, New York (1923)
13. Morowitz, H.,J.: The Emergence of Everything: How the World Became Complex, Oxford University Press, New York (2002)
14. Parunak, H. Van D., Brueckner, S., Fleischer, M., Odell, J.: Co-X: Defining what Agents Do Together, Proceedings of the AAMAS 2002 Workshop on Teamwork and Coalition Formation, Onn Shehory, Thomas R. Ioerger, Julita Vassileva, John Yen, eds., Bologna, Italy, (2002)
15. Tanev, I.: DOM/XML-Based Portable Genetic Representation of Morphology, Behavior and Communication Abilities of Evolvable Agents, Proceedings of the 8th International Symposium on Artificial Life and Robotics (AROB'03), Beppu, Japan (2003), 185–188