Computational Verb Game Theory: Part I–Strategic Games

Tao Yang

Abstract— **In computational verb games, the actions of players are modeled by using computational verb rules. Computational verb games extend conventional games by introducing premature payoffs and payoffs of continuum of actions. Strategic computational verb games are studied and some examples are provided.** *Copyright © 2009 Yang's Scientific Research Institute, LLC. All rights reserved.*

Index Terms— **Computational verb, computational verb game, strategic game.**

I. Introduction

COMPUTATIONAL verb rules can be used to model the rules/strategies of games, which are formal descriptions of strategic situations. Therefore, the game theory can be easily transformed into computational verb game theory (CVGT) by extending the rules/strategies of games into computational verb rules. Computational verb theory(CVT), which has been developed as an independent discipline in the framework of modern science and technology, shows again its strong penetrating power into other disciplines. This time, CVT penetrates into the game theory. This paper is the first part of a series of papers on the topics of computational game theory. In this paper, I will give no references of game theory because there are many standard textbooks available to the public and materials of course works in college level can be downloaded from the Internet.

The organization of this paper is as follows. In Section II, the brief history of computational verb theory will be given. In Section III, strategic games will be verbified. In Section IV, examples of two-player games will be presented. In Section V, some concluding remarks will be included.

II. A Brief History of Computational Verb Theory

As the first paradigm shift for solving engineering problems by using verbs, the computational verb theory[31] and physical linguistics[34], [51], [25] have undergone a rapid growth since the birth of computational verb in the Department of Electrical Engineering and Computer Sciences, University of California at Berkeley in 1997[16], [17]. The paradigm of implementing verbs in machines was coined as *computational verb theory*[31]. The building blocks of computational theory are *computational verbs*[26], [20], [18], [27], [32]. The relation between verbs and adverbs was mathematically defined in [19]. The logic operations between verb statements were studied in [21]. The applications of verb logic to verb reasoning were addressed in [22] and further studied in [31]. A logic paradox was solved based on verb logic[28]. The mathematical concept of set was generalized into verb set in[24]. Similarly, for measurable attributes, the number systems can be generalized into verb numbers[29]. The applications of computational verbs to predictions were studied in [23]. In [33] fuzzy dynamic systems were used to model a special kind of computational verb that evolves in fuzzy spaces. The relation between computational verb theory and traditional linguistics was studied in [31], [34]. The theoretical basis of developing computational cognition from a unified theory of fuzzy and computational verb theory is the theory of the UNICOGSE that was studied in [34], [39]. The issues of simulating cognition using computational verbs were studied in [35]. In [62] the correlation between computational verbs was studied. A method of implementing feelings in machines was proposed based on grounded computational verbs and computational nouns in [41]. In [48] a theory of how to design stable computational verb controllers was given. In [42] the rule-wise linear computational verb systems and their applications to the design of stable computational verb controllers and chaos in computational verb systems were presented. In [46] the concept of computational verb entropy was used to construct computational verb decision tree for data-mining applications. In [45] the relation between computational verbs and fuzzy sets was studied by using computational verb collapses and computational verb extension principles. In [47] the distances and similarities of saturated computational verbs were defined as normalized measures of the distances and similarities between computational verbs. Based on saturated computational verbs, the verb distances and similarities are related to each other with a simple relation. The distances and similarities between verbs with different life spans can be defined based on saturated computational verbs as well. In [49] the methods of using computational verbs to cluster trajectories and curves were presented. To cluster a bank of trajectories into a few representative computational verbs is to discover knowledge from database of time series. We use cluster centers to represent complex waveforms at symbolic levels. In [14] computational verb controllers were used to control a chaotic circuit model known as Chua's circuit. Computational verb controllers were designed based

Manuscript received April 17, 2009; revised April 26, 2009, May 09, 2009.
Tao Yang, Department of Electronic Engineering, Xiamen University, Xiamen 361005, P.R. China. Department of Cognitive Economics, Department of Electrical Engineering and Computer Sciences, Yang's Scientific Research Institute, 1303 East University Blvd., #20882, Tucson, Arizona 85719-0521, USA. Email: taoyang@xmu.edu.cn,taoyang@yangsky.com,taoyang@yangsky.us.
Publisher Item Identifier S 1542-5908(09)10310-X/$20.00

on verb control rules for different dynamics of the region-wise linear model of the control plant. In [13] computational verb controllers were used to synchronize discrete-time chaotic systems known as Hénon maps. Different verb control rules are designed for synchronizing different kinds of dynamics. In [53], how can computational verb theory functions as the most essential building block of cognitive engineering and cognitive industries was addressed. Computational verb theory will play a critical important role in personalizing services in the next fifty years. In [50], [52] computational verb theory was used to design an accurate flame-detecting systems based on CCTV signal. In [56] the learning algorithms were presented for learning computational verb rules from training data. In [54] the structures and learning algorithms of computational verb neural networks were presented. In [61] the ambiguities of the states and dynamics of computational verbs were studied. In [55] the history and milestones in the first ten years of the studies of computational verb theory were given. In [4] a case study of modeling adverbs as modifiers of computational verbs was presented. In [15] computational verb rules were used to improve the training processes of neural networks. In [57] the classifications and interactions between computational verb rule bases were presented. In [58] the simplest verb rules and their verb reasoning were connected to many intuitive applications of verb rules before the invention of computational verbs. In [59] the interactions between computational verbs were used as a powerful tools to understand the merging and splitting effects of verbs. In [3] computational verb rules were trained by using prescribed training samples of functions. In [60] the trend-based computational verb similarity was given as a way to decrease the computational complexities of verb similarities. In [5] computational verb PID controller was used to control linear motors. In [12] computational verb controller was used to control an auto-focusing system.

The theory of computational verb has been taught in some university classrooms since 2005[1]. The latest active applications of computational verb theory are listed as follows.

1) Computational Verb Controllers. The applications of computational verbs to different kinds of control problems were studied on different occassions[30], [31]. For the advanced applications of computational verbs to control problems, a few papers reporting the latest advances had been published[37], [36], [48], [42], [63]. The design of computational verb controllers was also presented in a textbook in 2005[1].
2) Computational Verb Image Processing and Image Understanding. The recent results of image processing by using computational verbs can be found in[38]. The applications of computational verbs to image understanding can be found in [40]. The authors of [2] applied computational verb image processing to design the vision systems of RoboCup small-size robots.
3) Stock Market Modeling and Prediction based on computational verbs. The product of Cognitive Stock Charts[8] was based on the advanced modeling and computing reported in [43]. Computational verb theory was used to study the trends of stock markets known as Russell reconstruction patterns [44].

Computational verb theory has been successfully applied to many industrial and commercial products. Some of these products are listed as follows.

1) Visual Card Counters. The *YangSky-MAGIC* card counter[10], developed by Yang's Scientific Research Institute and Wuxi Xingcard Technology Co. Ltd., was the first visual card counter to use computational verb image processing technology to achieve high accuracy of card and paper board counting based on cheap webcams.
2) CCTV Automatic Driver Qualify Test System. The *DriveQfy* CCTV automatic driver qualify test system[11] was the first vehicle trajectory reconstruction and stop time measuring system using computational verb image processing technology.
3) Visual Flame Detecting System. The *FireEye* visual flame detecting system[6] was the first CCTV or webcam based flame detecting system, which works under color and black & white conditions, for surveillance and security monitoring system.
4) Smart Pornographic Image and Video Detection Systems. The *PornSeer*[9] pornographic image and video detection systems are the first cognitive feature based smart porno detection and removal software.
5) Webcam Barcode Scanner. The *BarSeer*[7] webcam barcode scanner took advantage of the computational verb image processing to make the scan of barcode by using cheap webcam possible.
6) Cognitive Stock Charts. By applying computational verbs to the modeling of trends and cognitive behaviors of stock trading activities, cognitive stock charts can provide the traders with the "feelings" of stock markets by using simple and intuitive indexes.
7) TrafGo ITS SDK. Computational verbs were applied to model vehicle trajectories and dynamics of optical field and many other aspects of dynamics in complex environments for applications in intelligent transportation systems (ITS).

III. Strategic Games

In a strategic game, there are N players, the ith of which chooses an action a_i from a set of actions A_i. Let us define the set of all possible action vectors as

$$A \triangleq A_1 \times \ldots \times A_N. \tag{1}$$

The outcome of the game is an action vector $\boldsymbol{a} \in A$. Given two outcome vectors $\boldsymbol{a}_1 \in A$ and $\boldsymbol{a}_2 \in A$, if the player i

[1]Some computational verb theory related college courses are

- Dr. G. R. Chen, EE 64152 - Introduction to Fuzzy Informatics and Intelligent Systems, Department of Electronic Engineering, City University of Hong Kong.
- Dr. D. H. Guo, Artificial Intelligence, Department of Electronic Engineering, Xiamen University.
- Prof. T. Yang, Computational Methodologies in Intelligent Systems, Department of Electronic Engineering, Xiamen University.
- Dr. Mahir Sabra, EELE 6306: Intelligent Control, Electrical and Computer Engineering Department, The Islamic University of Gaza.

prefers $\boldsymbol{a}_1$ over $\boldsymbol{a}_2$ then we denote the preference relation as

$$\boldsymbol{a}_2 \preccurlyeq_i \boldsymbol{a}_1. \tag{2}$$

Definition 1 (Strategic game): A *strategic game* is a 3-tuple $\langle N, \{A_i\}, \{\preccurlyeq_i\}\rangle$ where N is the number of players, A_i and $\preccurlyeq_i$ are the finite set of actions and the preference relation for the ith player, respectively.

A *payoff function* for the ith player, $u_i : A \to \mathbb{R}$, can be used to transfer a preference relation $\boldsymbol{a}_2 \preccurlyeq_i \boldsymbol{a}_1$ into $u_i(\boldsymbol{a}_2) \leq u_i(\boldsymbol{a}_1)$. Therefore, the strategic game can be represented as $\langle N, \{A_i\}, \{u_i\}\rangle$ as well.

Definition 2 (Pareto optimal): An outcome of a game $\langle N, \{A_i\}, \{u_i\}\rangle$, $\boldsymbol{a}$, is *Pareto optimal* if there is no outcome $\boldsymbol{b}$ such that

$$\forall_{i\in\{1,\ldots,N\}} u_i(\boldsymbol{a}) \leq u_i(\boldsymbol{b}), \tag{3}$$

and

$$\exists_{i\in\{1,\ldots,N\}} u_i(\boldsymbol{a}) < u_i(\boldsymbol{b}). \tag{4}$$

Definition 3 (Nash equilibrium): An outcome of a game $\langle N, \{A_i\}, \{\preccurlyeq_i\}\rangle$, $\boldsymbol{a}^*$, is a *Nash equilibrium* if

$$\forall_{i\in\{1,\ldots,N\}}\forall_{b_i\in A_i}(a^*_{-i}, b_i) \preccurlyeq (a^*_{-i}, a^*_i), \tag{5}$$

where

$$(a_{-i}, x) = (a_1, a_2, \ldots, a_{i-1}, x, a_{i+1}, \ldots, a_N). \tag{6}$$

Example 1 (Payoff matrix): Assume a game of two players with actions $A_1 = \{\mathsf{V}_{11}, \mathsf{V}_{12}\}$ and $A_2 = \{\mathsf{V}_{21}, \mathsf{V}_{22}\}$, then the matrix representation of a game is given by Eq. (7).

In a conventional game, the actions V_{11}, V_{12}, V_{21}, and V_{22} are not modeled as computational verbs, they are only the name of these actions. Therefore, the payoff matrix in Eq. (7) is equivalent to the computational verb collapse of the computational verb rule set (8). The computational verb rule set (8) provides a way of modeling the dynamics of the game and generate dynamics of the payoffs.

The difference between a conventional game and computational verb game is the way of calculating the payoff of an action vector. In a conventional game, an action vector $\boldsymbol{a}$ is time-invariant, therefore, the payoff of the ith player, $u_i(\boldsymbol{a})$, is time-invariant as well. In a computational verb game each action a_i is a computational verb, therefore, $\boldsymbol{a}$ is not a vector of symbols but a vector of functions. To avoid confusing, we explicitly represent the action vector in the computational verb game as $\boldsymbol{a}(t)$. The payoff of the ith player in a computational verb game is represented as $u_i(\boldsymbol{a}(t))$ to differentiate it from that in the conventional game. Let's put the concepts of computational verb games under scrutiny as follows.

In a computational verb strategic game, an action a_i is a computational verb. Therefore, a computational verb strategic game is defined as follow.

Definition 4 (Computational verb strategic game): A *computational verb strategic game*(CVSG) is a 3-tuple $\langle N, \{A_i(t)\}, \{\preccurlyeq_i\}\rangle$ where N is the number of players, $A_i(t)$ and $\preccurlyeq_i$ are the finite set of computational verbs and the preference relation for the ith player, respectively.

The payoff of a CVSG is calculated as follow. Let's assume that the ith player takes an action modeled by a computational verb v_i, then the outcome vector is given by $\boldsymbol{v} = (\mathsf{v}_1, \ldots, \mathsf{v}_N)$. The payoff for the ith player is calculated as

$$\begin{aligned} u_i(\boldsymbol{v}) &= \frac{\displaystyle\sum_{j\in\{1,\ldots,N\}} u_i(\boldsymbol{a}_j) S(\boldsymbol{a}_j(t), \boldsymbol{v})}{\displaystyle\sum_{j\in\{1,\ldots,N\}} S(\boldsymbol{a}_j(t), \boldsymbol{v})}, \\ S(\boldsymbol{a}_j(t), \boldsymbol{v}) &= \bigwedge_{p=1}^{N} S(a_{jp}(t), \mathsf{v}_p) \end{aligned} \tag{9}$$

where $S(a_{jp}, \mathsf{v}_p)$ is a computational verb similarity. For an arbitrary computational verb similarity, the Nash equilibrium of the conventional game might not be the same as that of the corresponding CVSG. In order to guarantee that the Nash equilibrium of CVSG keeps the same as that of the conventional game, the computational verb similarity must be specially designed to satisfy the following conditions.

$$\begin{aligned} &S(\boldsymbol{a}_i(t), \boldsymbol{a}_i(t)) = 1, S(\boldsymbol{a}_i(t), \boldsymbol{a}_j(t)) = 0, i \neq j, \\ &\forall \boldsymbol{a}_i(t) \in A(t), \forall \boldsymbol{a}_j(t) \in A(t). \end{aligned} \tag{10}$$

Definition 5 (Consistent CVSG): A CVSG, $\langle N, \{A_i(t)\}, \{\preccurlyeq_i\}\rangle$, is *consistent* if its computational verbs satisfy Eq. (10).

Theorem 1: The computational verb collapse of a Nash equilibrium of a consistent CVSG $\langle N, \{A_i(t)\}, \{\preccurlyeq_i\}\rangle$ is a Nash equilibrium of the conventional game $\langle N, \{A_i\}, \{\preccurlyeq_i\}\rangle$.

Proof: It follows from Eqs. (9) and (10) that

$$\begin{aligned} u_i(\boldsymbol{v} = \boldsymbol{a}_k(t)) &= \frac{\displaystyle\sum_{j\in\{1,\ldots,N\}} u_i(\boldsymbol{a}_j) S(\boldsymbol{a}_j(t), \boldsymbol{a}_k(t))}{\displaystyle\sum_{j\in\{1,\ldots,N\}} S(\boldsymbol{a}_j(t), \boldsymbol{a}_k(t))} \\ &= \frac{u_i(\boldsymbol{a}_k) S(\boldsymbol{a}_k(t), \boldsymbol{a}_k(t))}{S(\boldsymbol{a}_k(t), \boldsymbol{a}_k(t))} \\ &= u_i(\boldsymbol{a}_k). \end{aligned} \tag{11}$$

Therefore, the payoff of the CVSG are the same as the corresponding conventional game. ■

Similarly, we have the following theorem.

Theorem 2: If the computational verb collapse of an outcome of a consistent CVSG $\langle N, \{A_i(t)\}, \{\preccurlyeq_i\}\rangle$ is Pareto optimal, then the outcome is Pareto optimal.

Therefore, the static properties of a consistent CVSG are the same as those of the corresponding conventional game. However, the dynamics of CVSG can't find from the conventional game.

IV. Examples of Two-player Games

As shown in Fig. 1, the normal form of game is represented by a *payoff matrix*. In the payoff matrix the players, strategies and payoffs are shown. In the existing normal form of game, the actions of players are modeled as static events, which are collapses of computational verbs. Furthermore, the normal form in Fig. 1 can be expressed as the following conventional

	V_{21}	V_{22}
V_{11}	$(u_1(\mathsf{V}_{11},\mathsf{V}_{21}),u_2(\mathsf{V}_{11},\mathsf{V}_{21}))$	$(u_1(\mathsf{V}_{11},\mathsf{V}_{22}),u_2(\mathsf{V}_{11},\mathsf{V}_{22}))$
V_{12}	$(u_1(\mathsf{V}_{12},\mathsf{V}_{21}),u_2(\mathsf{V}_{12},\mathsf{V}_{21}))$	$(u_1(\mathsf{V}_{12},\mathsf{V}_{22}),u_2(\mathsf{V}_{12},\mathsf{V}_{22}))$

(7)

$$\begin{aligned}
&\text{IF player 1 } \mathsf{V}_{11} \text{ AND player 2 } \mathsf{V}_{21}\text{, THEN the payoff for player 1 } \textsf{is } u_1(\mathsf{V}_{11},\mathsf{V}_{21}) \text{ AND for player 2 } \textsf{is } u_2(\mathsf{V}_{11},\mathsf{V}_{21});\\
&\text{IF player 1 } \mathsf{V}_{11} \text{ AND player 2 } \mathsf{V}_{22}\text{, THEN the payoff for player 1 } \textsf{is } u_1(\mathsf{V}_{11},\mathsf{V}_{22}) \text{ AND for player 2 } \textsf{is } u_2(\mathsf{V}_{11},\mathsf{V}_{22});\\
&\text{IF player 1 } \mathsf{V}_{12} \text{ AND player 2 } \mathsf{V}_{21}\text{, THEN the payoff for player 1 } \textsf{is } u_1(\mathsf{V}_{12},\mathsf{V}_{21}) \text{ AND for player 2 } \textsf{is } u_2(\mathsf{V}_{12},\mathsf{V}_{21});\\
&\text{IF player 1 } \mathsf{V}_{12} \text{ AND player 2 } \mathsf{V}_{22}\text{, THEN the payoff for player 1 } \textsf{is } u_1(\mathsf{V}_{12},\mathsf{V}_{22}) \text{ AND for player 2 } \textsf{is } u_2(\mathsf{V}_{12},\mathsf{V}_{22}).
\end{aligned} \tag{8}$$

rule set:

$$\begin{aligned}
&\text{IF } a_1 \textsf{ is } 0\text{m AND } a_2 \textsf{ is } 0\text{m,}\\
&\qquad\text{THEN } p_1 \textsf{ is } 0 \text{ AND } p_2 \textsf{ is } 0;\\
&\text{IF } a_1 \textsf{ is } 0\text{m AND } a_2 \textsf{ is } 100\text{m,}\\
&\qquad\text{THEN } p_1 \textsf{ is } -1 \text{ AND } p_2 \textsf{ is } 1;\\
&\text{IF } a_1 \textsf{ is } 100\text{m AND } a_2 \textsf{ is } 0\text{m,}\\
&\qquad\text{THEN } p_1 \textsf{ is } 1 \text{ AND } p_2 \textsf{ is } -1;\\
&\text{IF } a_1 \textsf{ is } 100\text{m AND } a_2 \textsf{ is } 100\text{m,}\\
&\qquad\text{THEN } p_1 \textsf{ is } 0 \text{ AND } p_2 \textsf{ is } 0
\end{aligned} \tag{12}$$

where a_1 and a_2 are noun phrases that are used to name the actions taken by the players 1 and 2, respectively. p_1 and p_2 are the payoffs of the players 1 and 2, respectively.

Player 2

Player 1	0m	100m
0m	0,0	-1,1
100m	1,-1	0,0

Fig. 1. The payoff matrix of a two-player and two-strategy game.

Figure 1 represents a simultaneous game of two players and two strategies. The payoffs of both players can be calculated based on Boolean rule set (12). For example, if the action of the player 1 is $a_1 = 0$m and the action of the player 2 is $a_2 = 100$m, then it follows from the second rule of Eq. (12) that the payoff of the player 1 is -1 and that of the player 2 is 1. Although it is simple to find the payoffs from Boolean rule set (12), it is unclear how to find the possible payoffs if the game is premature. For example, if the a_1 is not 0m and we know it is approaching 0m and a_2 is approaching 100m, then what payoffs should we predict or should we assign to both players if the game stops before it is mature?

In many games, it is not an issue because these games don't allow premature. However, in many other games, it is very natural to accept premature results. It is impossible to use the rule set (12) to find the payoffs for a premature game because Boolean rules define only mature results. To overcome this problem, by using computational verb extension principles presented in [51], we can extend the rule set (12) into the following computational verb rule set.

$$\begin{aligned}
&\text{IF } a_1 \textsf{ become } 0\text{m AND } a_2 \textsf{ become } 0\text{m,}\\
&\qquad\text{THEN } p_1 \textsf{ become } 0 \text{ AND } p_2 \textsf{ become } 0;\\
&\text{IF } a_1 \textsf{ become } 0\text{m AND } a_2 \textsf{ become } 100\text{m,}\\
&\qquad\text{THEN } p_1 \textsf{ become } -1 \text{ AND } p_2 \textsf{ become } 1;\\
&\text{IF } a_1 \textsf{ become } 100\text{m AND } a_2 \textsf{ become } 0\text{m,}\\
&\qquad\text{THEN } p_1 \textsf{ become } 1 \text{ AND } p_2 \textsf{ become } -1;\\
&\text{IF } a_1 \textsf{ become } 100\text{m AND } a_2 \textsf{ become } 100\text{m,}\\
&\qquad\text{THEN } p_1 \textsf{ become } 0 \text{ AND } p_2 \textsf{ become } 0.
\end{aligned} \tag{13}$$

The verb rule set (13) defines a computational verb game as shown in Fig. 2.

Player 2

Player 1	Become 0m	Become 100m
Become 0m	Become 0, become 0	Become -1, become 1
Become 100m	Become 1, become -1	Become 0, become 0

Fig. 2. The payoff matrix of a two-player and two-strategy computational verb game.

To extend the rule sets of strategies of games from conventional rules to computational verb rules has the following implications to the extension of existing game theory.

- To extend games from strategies about static points into strategies about trends allows us to play a game continuously without waiting for all players finishing their moves.
- To find the payoff of a game when the players are making decisions or taking actions. This is of great importance in real-life applications when games must have a result whenever a condition is satisfied.
- To develop a game theory that can cope with premature conventional games.
- To develop a game theory that can cope with continuum of actions based on a few sampled actions of players.

Computational verb reasoning must be used to find the payoffs of verb games as that shown in Eq. (13). The basic method is to define the template computational verbs in Eq. (13)

first. Then using the time series of a_1 and a_2 to find the computational verb similarities between observed waveforms and template verbs. The computational verb similarities are used to determine the dynamics of the payoffs. If we assume that the actions shown in Fig. 2 are four sampling points of a 2D continuum of dynamics, then the payoffs of arbitrary actions of both players might be interpolated from the existing sampling points by using Eq. (9). Examples of such outcomes are given as the following computational verb rules.

$$\begin{aligned} &\text{IF } a_1 \textsf{ become } 10\text{m AND } a_2 \textsf{ become } 50\text{m}, \\ &\qquad \text{THEN } p_1 \textsf{ become } x_1 \text{ AND } p_2 \textsf{ become } x_2; \\ &\text{IF } a_1 \textsf{ become } 70\text{m AND } a_2 \textsf{ become } 80\text{m}, \\ &\qquad \text{THEN } p_1 \textsf{ become } x_3 \text{ AND } p_2 \textsf{ become } x_4; \\ &\text{IF } a_1 \textsf{ become } \texttt{big} \text{ AND } a_2 \textsf{ become } \texttt{small}, \\ &\qquad \text{THEN } p_1 \textsf{ become } x_5 \text{ AND } p_2 \textsf{ become } x_6 \end{aligned} \tag{14}$$

where x_1, $i = 1, \ldots, 6$, are payoffs to be found. Observe that the actions modeled as computational verbs in the premises of these computational verb rules are not exactly those computational verbs in the premises in rule set (13).

In the rest of this section, some examples are used to illustrate how computational verb games are constructed.

A. Prisoner's Dilemma

Let's consider the canonical payoff matrix of prisoner's dilemma as in the following equation

	cooperate	defect
cooperate	R, R	S, T
defect	T, S	P, P

(15)

where the parameters T, R, P and S represent four kinds of payoffs for different strategies. The computational verb rule set for the payoff matrix (15) is as follow.

$$\begin{aligned} &\text{IF } a_1 \textsf{ cooperate} \text{ AND } a_2 \textsf{ cooperate}, \\ &\qquad \text{THEN } p_1 \text{ is } R \text{ AND } p_2 \text{ is } R; \\ &\text{IF } a_1 \textsf{ cooperate} \text{ AND } a_2 \textsf{ defect}, \\ &\qquad \text{THEN } p_1 \text{ is } S \text{ AND } p_2 \text{ is } T; \\ &\text{IF } a_1 \textsf{ defect} \text{ AND } a_2 \textsf{ cooperate}, \\ &\qquad \text{THEN } p_1 \text{ is } T \text{ AND } p_2 \text{ is } S; \\ &\text{IF } a_1 \textsf{ defect} \text{ AND } a_2 \textsf{ defect}, \\ &\qquad \text{THEN } p_1 \text{ is } P \text{ AND } p_2 \text{ is } P \end{aligned} \tag{16}$$

Assume that the observation verbs for a_1 and a_2 are, respectively, V_1 and V_2, then the payoffs for both prisoners are calculated by using the following computational verb reasoning procedure.

$$\begin{aligned} S_1 &= S(\mathsf{cooperate}, \mathsf{V}_1) \wedge S(\mathsf{cooperate}, \mathsf{V}_2), \\ S_2 &= S(\mathsf{cooperate}, \mathsf{V}_1) \wedge S(\mathsf{defect}, \mathsf{V}_2), \\ S_3 &= S(\mathsf{defect}, \mathsf{V}_1) \wedge S(\mathsf{cooperate}, \mathsf{V}_2), \\ S_4 &= S(\mathsf{defect}, \mathsf{V}_1) \wedge S(\mathsf{defect}, \mathsf{V}_2), \\ p_1 &= \frac{S_1 R + S_2 S + S_3 T + S_4 P}{S_1 + S_2 + S_3 + S_4}, \\ p_2 &= \frac{S_1 R + S_2 T + S_3 S + S_4 P}{S_1 + S_2 + S_3 + S_4} \end{aligned} \tag{17}$$

where $\wedge$ denotes a t-norm. $S(\cdot, \cdot)$ might be any kind of computational verb similarity.

B. Chicken Game

Let's consider one example of the payoff matrix of chicken game as in the following equation

	swerve	stay straight
swerve	$0, 0$	$-1, +1$
stay straight	$+1, -1$	$-10, -10$

(18)

The computational verb rule set for the payoff matrix (18) is as follow.

$$\begin{aligned} &\text{IF } a_1 \textsf{ swerve} \text{ AND } a_2 \textsf{ swerve}, \\ &\qquad \text{THEN } p_1 \text{ is } 0 \text{ AND } p_2 \text{ is } 0; \\ &\text{IF } a_1 \textsf{ swerve} \text{ AND } a_2 \textsf{ stay straight}, \\ &\qquad \text{THEN } p_1 \text{ is } -1 \text{ AND } p_2 \text{ is } +1; \\ &\text{IF } a_1 \textsf{ stay straight} \text{ AND } a_2 \textsf{ swerve}, \\ &\qquad \text{THEN } p_1 \text{ is } +1 \text{ AND } p_2 \text{ is } -1; \\ &\text{IF } a_1 \textsf{ stay straight} \text{ AND } a_2 \textsf{ stay straight}, \\ &\qquad \text{THEN } p_1 \text{ is } -10 \text{ AND } p_2 \text{ is } -10 \end{aligned} \tag{19}$$

Assume that the observation verbs for a_1 and a_2 are, respectively, V_1 and V_2, then the payoffs for both drivers are calculated by using the following computational verb reasoning procedure.

$$\begin{aligned} S_1 &= S(\mathsf{swerve}, \mathsf{V}_1) \wedge S(\mathsf{swerve}, \mathsf{V}_2), \\ S_2 &= S(\mathsf{swerve}, \mathsf{V}_1) \wedge S(\mathsf{stay\ straight}, \mathsf{V}_2), \\ S_3 &= S(\mathsf{stay\ straight}, \mathsf{V}_1) \wedge S(\mathsf{swerve}, \mathsf{V}_2), \\ S_4 &= S(\mathsf{stay\ straight}, \mathsf{V}_1) \wedge S(\mathsf{stay\ straight}, \mathsf{V}_2), \\ p_1 &= \frac{-S_2 + S_3 - 10 S_4}{S_1 + S_2 + S_3 + S_4}, \\ p_2 &= \frac{S_2 - S_3 - 10 S_4}{S_1 + S_2 + S_3 + S_4}. \end{aligned} \tag{20}$$

V. Concluding Remarks

When I set up world's first *Department of Cognitive Economics* a few years ago in Yang's Scientific Research Institute, I had a strong feeling that computational verb theory would play a critical role in bringing irrational aspects of human cognition into the study of economics. One day in April 2009, when I was swimming under the bright Sun of Tucson, a brain storm hit me, I knew I could turn my feeling into a real mathematical implementation by verbifying game theory.

When I write these lines, the world's economics is in a deep trouble. I just read some news about the bankrupts of four more US banks. I get a feeling that the mainstream economics ignored too much and too long the facts that economics is essentially the study of the relation among human cognitions. I told my students once that the mainstream economics is not yet a science rather than an art because the core concept of *value* is ill-defined. From my perspective, the *value* is about the work can be done by a COGNITION information body. Without setting up a measurable framework for cognition, it is impossible to make *value* measurable in physical sense. Without a physically measurable *value*, economics will remain a social science rather than a natural science.

To make economics a natural science, the key is to make cognition *measurable* and furthermore, *computable*. As I had addressed in [51], the only way to make cognition measurable is via *physical linguistics*. Therefore, computational verb theory is the key to making the irrational aspects of cognition in economics measurable. This paper is the first of its kind and will pave the way for my students to explore the unknowns in *cognitive economics*. My students and assistants will join me to continue the researches in different directions of computational verb game theory.

REFERENCES

[1] Guanrong Chen and Trung Tat Pham. *Introduction to Fuzzy Systems*. Chapman & Hall/CRC, November 2005. ISBN:1-58488-531-9.

[2] Wanmi Chen, Yanqin Wei, Minrui Fei, and Huosheng Hu. Applications of computational verbs to image processing of RoboCup small-size robots. In *Intelligent Control and Automation*, volume 344/2006 of *Lecture Notes in Control and Information Sciences*, pages 494–499. Springer, Berlin / Heidelberg, 2006.

[3] Y. Guo and T. Yang. Training computational verb neural networks with computational verb rule bases,. *International Journal of Computational Cognition*, 6(4):12–23, December 2008 [available online at $http://www.YangSky.us/ijcc/ijcc64.htm$, $http://www.YangSky.com/ijcc/ijcc64.htm$].

[4] Yi Guo. A study of adverbs as modifiers of computational verbs. *International Journal of Computational Cognition*, 6(1):31–35, March 2008 [available online at $http://www.YangSky.com/ijcc/ijcc61.htm$, $http://www.YangSky.us/ijcc/ijcc61.htm$].

[5] J. Li. Research and application of computational verb PID controller of linear motor. Master's thesis, Kunming University of Science and Technology, Feb. 2008 [available online at $http://www.YangSky.com/researches/computationalverbs/verbfuzctrl/vbPIDMotor.pdf$]. (in Chinese).

[6] Yang's Scientific Research Institute LLC. **FireEye** *Visual Flame Detecting Systems*. http://www.yangsky.us/products/flamesky/index.htm, http://www.yangsky.com/products/flamesky/index.htm, 2005.

[7] Yang's Scientific Research Institute LLC. **BarSeer** *Webcam Barcode Scanner*. http://www.yangsky.us/demos/barseer/barseer.htm, http://www.yangsky.com/demos/barseer/barseer.htm, 2006.

[8] Yang's Scientific Research Institute LLC. *Cognitive Stock Charts*. http://www.yangsky.us/products/stock/, http://www.yangsky.com/products/stock/, 2006.

[9] Yang's Scientific Research Institute LLC. **PornSeer** *Pornographic Image and Video Detection Systems*. http://www.yangsky.us/products/dshowseer/porndetection/PornSeePro.htm, http://www.yangsky.com/products/dshowseer/porndetection/PornSeePro.htm, 2006.

[10] Yang's Scientific Research Institute LLC. and Wuxi Xingcard Technology Ltd. **YangSky-MAGIC** *Visual Card Counters*. http://www.yangsky.us/products/cardsky/cardsky.htm, http://www.yangsky.com/products/cardsky/cardsky.htm, 2004.

[11] Yang's Scientific Research Institute LLC. and Chinese Traffic Management Research Institute of the Ministry of Public Security(TMRI-China). **DriveQfy** *Automatic CCTV Driver Qualify Testing Systems*. http://www.yangsky.us/products/driveqfy/driveqfy.htm, http://www.yangsky.com/products/driveqfy/driveqfy.htm, 2005.

[12] Weibin Tang, Yinghao Liao, Zhicong Chen, Lihuan Cai, Tao Yang, and Donghui Guo. Auto-focusing system for microscope based on computational verb controllers. In *Anti-counterfeiting, Security and Identification, 2008. ASID 2008. 2nd International Conference on*, pages 84–87, Guiyang, China, 20-23 Aug. 2008. IEEE, IEEE Press.

[13] R. Tonelli and T. Yang. Synchronizing Hénon maps using computational verb controllers. *Phys. Rev. E.*, 2007. submitted.

[14] R. Tonelli and T. Yang. Controlling Chua's circuits using computational verb controllers. *International Journal of Robust and Nonlinear Control*, 18(17):1622–1636, Nov. 25 2008.

[15] H.-B. Wang and T. Yang. Training neural networks using computational verb rules. *International Journal of Computational Cognition*, 6(2):17–32, June 2008 [available online at $http://www.YangSky.us/ijcc/ijcc62.htm$, $http://www.YangSky.com/ijcc/ijcc62.htm$].

[16] T. Yang. Verbal paradigms—Part I: Modeling with verbs. Technical Report Memorandum No. UCB/ERL M97/64, Electronics Research Laboratory, College of Engineering, University of California, Berkeley, CA 94720, 9 Sept. 1997. page 1-15.

[17] T. Yang. Verbal paradigms—Part II: Computing with verbs. Technical Report Memorandum No. UCB/ERL M97/66, Electronics Research Laboratory, College of Engineering, University of California, Berkeley, CA 94720, 18 Sept. 1997. page 1-27.

[18] T. Yang. Computational verb systems: Computing with verbs and applications. *International Journal of General Systems*, 28(1):1–36, 1999.

[19] T. Yang. Computational verb systems: Adverbs and adverbials as modifiers of verbs. *Information Sciences*, 121(1-2):39–60, Dec. 1999.

[20] T. Yang. Computational verb systems: Modeling with verbs and applications. *Information Sciences*, 117(3-4):147–175, Aug. 1999.

[21] T. Yang. Computational verb systems: Verb logic. *International Journal of Intelligent Systems*, 14(11):1071–1087, Nov. 1999.

[22] T. Yang. Computational verb systems: A new paradigm for artificial intelligence. *Information Sciences—An International Journal*, 124(1-4):103–123, 2000.

[23] T. Yang. Computational verb systems: Verb predictions and their applications. *International Journal of Intelligent Systems*, 15(11):1087–1102, Nov. 2000.

[24] T. Yang. Computational verb systems: Verb sets. *International Journal of General Systems*, 20(6):941–964, 2000.

[25] T. Yang. Computational verb systems: Towards a unified paradigm for artificial languages. In *Proceedings of the Fifth Joint Conference on Information Sciences(JCIS 2000)*, pages 29–32, Atlantic City, NJ, Feb. 27-Mar. 03 2000.

[26] T. Yang. *Advances in Computational Verb Systems*. Nova Science Publishers, Inc., Huntington, NY, May 2001. ISBN 1-56072-971-6.

[27] T. Yang. Computational verb systems: Computing with perceptions of dynamics. *Information Sciences*, 134(1-4):167–248, Jun. 2001.

[28] T. Yang. Computational verb systems: The paradox of the liar. *International Journal of Intelligent Systems*, 16(9):1053–1067, Sept. 2001.

[29] T. Yang. Computational verb systems: Verb numbers. *International Journal of Intelligent Systems*, 16(5):655–678, May 2001.

[30] T. Yang. *Impulsive Control Theory*, volume 272 of *Lecture Notes in Control and Information Sciences*. Spinger-Verlag, Berlin, Aug. 2001. ISBN 354042296X.

[31] T. Yang. *Computational Verb Theory: From Engineering, Dynamic Systems to Physical Linguistics*, volume 2 of *YangSky.com Monographs in Information Sciences*. Yang's Scientific Research Institute, Tucson, AZ, Oct. 2002. ISBN:0-9721212-1-8.

[32] T. Yang. Computational verb systems: Verbs and dynamic systems. *International Journal of Computational Cognition*, 1(3):1–50, Sept. 2003.

[33] T. Yang. *Fuzzy Dynamic Systems and Computational Verbs Represented by Fuzzy Mathematics*, volume 3 of *YangSky.com Monographs in Information Sciences*. Yang's Scientific Press, Tucson, AZ, Sept. 2003. ISBN:0-9721212-2-6.

[34] T. Yang. *Physical Linguistics: Measurable Linguistics and Duality Between Universe and Cognition*, volume 5 of *YangSky.com Monographs in Information Sciences*. Yang's Scientific Press, Tucson, AZ, Dec. 2004.

[35] T. Yang. Simulating human cognition using computational verb theory. *Journal of Shanghai University(Natural Sciences)*, 10(s):133–142, Oct. 2004.

[36] T. Yang. Architectures of computational verb controllers: Towards a new paradigm of intelligent control. *International Journal*

of Computational Cognition, 3(2):74–101, June 2005 [available online at http://www.YangSky.com/ijcc/ijcc32.htm, http://www.YangSky.us/ijcc/ijcc32.htm].

[37] T. Yang. Applications of computational verbs to the design of P-controllers. *International Journal of Computational Cognition*, 3(2):52–60, June 2005 [available online at http://www.YangSky.us/ijcc/ijcc32.htm, http://www.YangSky.com/ijcc/ijcc32.htm].

[38] T. Yang. Applications of computational verbs to digital image processing. *International Journal of Computational Cognition*, 3(3):31–40, September 2005 [available online at http://www.YangSky.us/ijcc/ijcc33.htm, http://www.YangSky.com/ijcc/ijcc33.htm].

[39] T. Yang. Bridging the Universe and the Cognition. *International Journal of Computational Cognition*, 3(4):1–15, December 2005 [available online at http://www.YangSky.us/ijcc/ijcc34.htm, http://www.YangSky.com/ijcc/ijcc34.htm].

[40] T. Yang. Applications of computational verbs to effective and realtime image understanding. *International Journal of Computational Cognition*, 4(1):49–67, March 2006 [available online at http://www.YangSky.com/ijcc/ijcc41.htm, http://www.YangSky.us/ijcc/ijcc41.htm].

[41] T. Yang. Applications of computational verbs to feeling retrieval from texts. *International Journal of Computational Cognition*, 4(3):28–45, September 2006 [available online at http://www.YangSky.com/ijcc/ijcc43.htm, http://www.YangSky.us/ijcc/ijcc43.htm].

[42] T. Yang. Rule-wise linear computational verb systems: Dynamics and control. *International Journal of Computational Cognition*, 4(4):18–33, December 2006 [available online at http://www.YangSky.com/ijcc/ijcc44.htm, http://www.YangSky.us/ijcc/ijcc44.htm].

[43] T. Yang. Applications of computational verbs to cognitive models of stock markets. *International Journal of Computational Cognition*, 4(2):1–13, June 2006 [available online at http://www.YangSky.us/ijcc/ijcc42.htm, http://www.YangSky.com/ijcc/ijcc42.htm].

[44] T. Yang. Applications of computational verbs to the study of the effects of Russell's annual index reconstitution on stock markets. *International Journal of Computational Cognition*, 4(3):1–8, September 2006 [available online at http://www.YangSky.us/ijcc/ijcc43.htm, http://www.YangSky.com/ijcc/ijcc43.htm].

[45] T. Yang. Bridging computational verbs and fuzzy membership functions using computational verb collapses. *International Journal of Computational Cognition*, 4(4):47–61, December 2006 [available online at http://www.YangSky.us/ijcc/ijcc44.htm, http://www.YangSky.com/ijcc/ijcc44.htm].

[46] T. Yang. Computational verb decision trees. *International Journal of Computational Cognition*, 4(4):34–46, December 2006 [available online at http://www.YangSky.us/ijcc/ijcc44.htm, http://www.YangSky.com/ijcc/ijcc44.htm].

[47] T. Yang. Distances and similarities of saturated computational verbs. *International Journal of Computational Cognition*, 4(4):62–77, December 2006 [available online at http://www.YangSky.us/ijcc/ijcc44.htm, http://www.YangSky.com/ijcc/ijcc44.htm].

[48] T. Yang. Stable computational verb controllers. *International Journal of Computational Cognition*, 4(4):9–17, December 2006 [available online at http://www.YangSky.us/ijcc/ijcc44.htm, http://www.YangSky.com/ijcc/ijcc44.htm].

[49] T. Yang. Using computational verbs to cluster trajectories and curves. *International Journal of Computational Cognition*, 4(4):78–87, December 2006 [available online at http://www.YangSky.us/ijcc/ijcc44.htm, http://www.YangSky.com/ijcc/ijcc44.htm].

[50] T. Yang. Accurate video flame-detecting system based on computational verb theory. AS *Installer*, (42):154–157, August 2007. (in Chinese).

[51] T. Yang. *The Mathematical Principles of Natural Languages: The First Course in Physical Linguistics*, volume 6 of *YangSky.com Monographs in Information Sciences*. Yang's Scientific Press, Tucson, AZ, Dec. 2007. ISBN:0-9721212-4-2.

[52] T. Yang. Applications of computational verb theory to the design of accurate video flame-detecting systems. *International Journal of Computational Cognition*, 5(3):25–42, September 2007 [available online at http://www.YangSky.us/ijcc/ijcc53.htm, http://www.YangSky.com/ijcc/ijcc53.htm].

[53] T. Yang. Cognitive engineering and cognitive industry. *International Journal of Computational Cognition*, 5(3):1–24, September 2007 [available online at http://www.YangSky.us/ijcc/ijcc53.htm, http://www.YangSky.com/ijcc/ijcc53.htm].

[54] T. Yang. Computational verb neural networks. *International Journal of Computational Cognition*, 5(3):57–62, September 2007 [available online at http://www.YangSky.us/ijcc/ijcc53.htm, http://www.YangSky.com/ijcc/ijcc53.htm].

[55] T. Yang. Computational verb theory: Ten years later. *International Journal of Computational Cognition*, 5(3):63–86, September 2007 [available online at http://www.YangSky.us/ijcc/ijcc53.htm, http://www.YangSky.com/ijcc/ijcc53.htm].

[56] T. Yang. Learning computational verb rules. *International Journal of Computational Cognition*, 5(3):43–56, September 2007 [available online at http://www.YangSky.us/ijcc/ijcc53.htm, http://www.YangSky.com/ijcc/ijcc53.htm].

[57] T. Yang. Computational verb rule bases. *International Journal of Computational Cognition*, 6(3):23–34, September 2008 [available online at http://www.YangSky.us/ijcc/ijcc63.htm, http://www.YangSky.com/ijcc/ijcc63.htm].

[58] T. Yang. Simplest computational verb rules and their reasoning. *International Journal of Computational Cognition*, 6(3):35–41, September 2008 [available online at http://www.YangSky.us/ijcc/ijcc63.htm, http://www.YangSky.com/ijcc/ijcc63.htm].

[59] T. Yang. Interactions between computational verbs. *International Journal of Computational Cognition*, 6(4):1–11, December 2008 [available online at http://www.YangSky.us/ijcc/ijcc64.htm, http://www.YangSky.com/ijcc/ijcc64.htm].

[60] T. Yang. Trend-based computational verb similarity. *International Journal of Computational Cognition*, 6(4):24–33, December 2008 [available online at http://www.YangSky.us/ijcc/ijcc64.htm, http://www.YangSky.com/ijcc/ijcc64.htm].

[61] T. Yang and Y. Guo. Measures of ambiguity of computational verbs based on computational verb collapses. *International Journal of Computational Cognition*, 5(4):1–12, December 2007 [available online at http://www.YangSky.us/ijcc/ijcc54.htm, http://www.YangSky.com/ijcc/ijcc54.htm].

[62] Jian Zhang and Minrui Fei. Determination of verb similarity in computational verb theory. *International Journal of Computational Cognition*, 3(3):74–77, September 2005 [available online at http://www.YangSky.us/ijcc/ijcc33.htm, http://www.YangSky.com/ijcc/ijcc33.htm].

[63] Sheng Zhu, Zhong-Jie Wang, Yong Liu, and Bao-Liang Xia. An improvement of the design of computational verb PID-controllers. *System Simulation Technology*, 2(1):25–30, Jan. 2006. (in Chinese).