Money as a Medium of Exchange in an Economy With Artificial Intelligent Agents

November 10, 2022

Introduction

Kiyotaki and Wright 1989

1. Similar setup as in KW 1989

1. Similar setup as in KW 1989
2. Equal number of 3 types, say 1, 2, 3

Kiyotaki and Wright 1989

1. Similar setup as in KW 1989
2. Equal number of 3 types, say 1, 2, 3
3. Type 1 produce for 2, 2 produce for 3, 3 produce for 1. Type i wants i

Kiyotaki and Wright 1989

1. Similar setup as in KW 1989
2. Equal number of 3 types, say 1, 2, 3
3. Type 1 produce for 2, 2 produce for 3, 3 produce for 1. Type i wants i
4. No double coincidence of wants

1. Similar setup as in KW 1989
2. Equal number of 3 types, say 1, 2, 3
3. Type 1 produce for 2, 2 produce for 3, 3 produce for 1. Type i wants i
4. No double coincidence of wants
5. Intrinsic property - Cost of holding $s_1 < s_2 < s_3$

Kiyotaki and Wright 1989

1. Similar setup as in KW 1989
2. Equal number of 3 types, say 1, 2, 3
3. Type 1 produce for 2, 2 produce for 3, 3 produce for 1. Type i wants i
4. No double coincidence of wants
5. Intrinsic property - Cost of holding $s_1 < s_2 < s_3$
6. Type i receives utility when consumes good i

1. Similar setup as in KW 1989
2. Equal number of 3 types, say 1, 2, 3
3. Type 1 produce for 2, 2 produce for 3, 3 produce for 1. Type i wants i
4. No double coincidence of wants
5. Intrinsic property - Cost of holding $s_1 < s_2 < s_3$
6. Type i receives utility when consumes good i
7. Can only hold one unit of good each period

Kiyotaki and Wright 1989

1. Similar setup as in KW 1989
2. Equal number of 3 types, say 1, 2, 3
3. Type 1 produce for 2, 2 produce for 3, 3 produce for 1. Type i wants i
4. No double coincidence of wants
5. Intrinsic property - Cost of holding $s_1 < s_2 < s_3$
6. Type i receives utility when consumes good i
7. Can only hold one unit of good each period
8. Poisson meeting process

Kiyotaki and Wright 1989

1. Similar setup as in KW 1989
2. Equal number of 3 types, say 1, 2, 3
3. Type 1 produce for 2, 2 produce for 3, 3 produce for 1. Type i wants i
4. No double coincidence of wants
5. Intrinsic property - Cost of holding $s_1 < s_2 < s_3$
6. Type i receives utility when consumes good i
7. Can only hold one unit of good each period
8. Poisson meeting process

1. Similar setup as in KW 1989
2. Equal number of 3 types, say 1, 2, 3
3. Type 1 produce for 2, 2 produce for 3, 3 produce for 1. Type i wants i
4. No double coincidence of wants
5. Intrinsic property - Cost of holding $s_1 < s_2 < s_3$
6. Type i receives utility when consumes good i
7. Can only hold one unit of good each period
8. Poisson meeting process

It turned out to have two consistent equilibria

- Fundamental equilibrium : Good 1 as the medium of Exchange
- Speculative equilibrium : Type 1 also wants good 3

This paper explores the *emergence* of such equilibria with agent-based modeling.

Notations

- $a \in \mathscr{A} = \{1, 2, ...A\}$: agent
- At time t , a holds good x_{at}
- Trade with agent $\rho_t(a)$, who holds $x_{\rho_t(a)t}$. $z_{at} \equiv (x_{at}, x_{\rho_t(a)t})$
- a produces $f(a)$
- a receives utility $u_i(i) > 0$ after consumption. (Only consumes its own type)

Decisions: Trade or Not Trade, Consume or Not Consume.

$$\lambda_{at} = \begin{cases} 1 & \text{if} \quad a \text{ wants } x_{at} \text{ for } x_{\rho_t(a)t} \\ 0 & \text{if} \quad a \text{ refuses to trade} \end{cases} \tag{1}$$

$$\gamma at = \begin{cases} 1 & \text{if} \quad a \text{ wants to consume } x^+_{at} \\ 0 & \text{if} \quad a \text{ not consume} \end{cases} \tag{3}$$

$$x_{a,t+1} = \gamma_{at} f(a) + (1-\gamma_{at})\left((1-\lambda_{at}\cdot\lambda_{\rho_t(a)t})x_{at} + \lambda_{at}\cdot\lambda_{\rho_t(a)t}x_{\rho_t(a)t}\right) \tag{4}$$

Classifier System

Classifier System

- Allows agent to learn

Classifier System

- Allows agent to learn
- Good Rules accumulate higher *strength*

- Allows agent to learn
- Good Rules accumulate higher *strength*
- Higher strength rules are more likely to win the bid and become the decision

- Allows agent to learn
- Good Rules accumulate higher *strength*
- Higher strength rules are more likely to win the bid and become the decision

- Allows agent to learn
- Good Rules accumulate higher *strength*
- Higher strength rules are more likely to win the bid and become the decision

The following slides are going to cover

1. Encoding of CS and notations

Classifier System

- Allows agent to learn
- Good Rules accumulate higher *strength*
- Higher strength rules are more likely to win the bid and become the decision

The following slides are going to cover

1. Encoding of CS and notations
2. Evolution of strength

Classifier System

- Allows agent to learn
- Good Rules accumulate higher *strength*
- Higher strength rules are more likely to win the bid and become the decision

The following slides are going to cover

1. Encoding of CS and notations
2. Evolution of strength
3. Genetic algorithm

Encodings for a classifier system I

Code			Meaning
1	0	0	Good 1
0	1	0	Good 2
0	0	1	Good 3
0	#	#	Not good 1
#	0	#	Not good 2
#	#	0	Not good 3

1 : hold
0: not hold
#: either

Encodings for a classifier system II

Define an exchange classifier

$$\text{Holding of } a + \text{Holding of } \rho(a) + \text{Trade / Not Trade}$$

total of 3+3+1 codes

Desicion Making

$$M_e(z_{at}) = \{e : z_{at} \text{ matches the condition part of classifier } e\} \quad (5)$$

Members of $M_e(z_{at})$ form a collection of potential "bidders". e with the highest strength wins

$$\lambda_{at} = e_t(z_{at}) = \arg\max\{S_e^a(t) : e \in M_e(z_{at})\} \quad (6)$$

$$M_e(z_{at}) = \{e : z_{at} \text{ matches the condition part of classifier } e\} \tag{5}$$

Members of $M_e(z_{at})$ form a collection of potential "bidders". e with the highest strength wins

$$\lambda_{at} = e_t(z_{at}) = \arg\max\{S_e^a(t) : e \in M_e(z_{at})\} \tag{6}$$

Same for consumption desicion

$$M_c(z_{at}) = \{c : x_{at}^+ \text{ matches the condition part of classifier } c\} \tag{7}$$

$$\gamma_{at} = c_t(z_{at}) = \arg\max\{S_c^a(t) : c \in M_c(z_{at})\} \tag{8}$$

Along with the law of motion in Eq. 4

$$\tau_e^a(t) = 1 + \sum_{s=0}^{t} I_e^a(s) \tag{9}$$

- $\tau_e^a(t)$: the count of successful trade using the classifier e up to time t.
- $I_e^a(t)$: whether e wins the auction (between classifiers) and helps the agent successfully trade, i.e., $\lambda_{at} \cdot \lambda_{\rho_t(a)t} x_{\rho_t(a)t} = 1$

$$\tau_e^a(t) = 1 + \sum_{s=0}^{t} I_e^a(s) \tag{9}$$

- $\tau_e^a(t)$: the count of successful trade using the classifier e up to time t.
- $I_e^a(t)$: whether e wins the auction (between classifiers) and helps the agent successfully trade, i.e., $\lambda_{at} \cdot \lambda_{\rho_t(a)t} x_{\rho_t(a)t} = 1$

Analogously,

$$\tau_c^a(t) = 1 + \sum_{s=0}^{t} I_c^a(s) \tag{10}$$

The strength of a classifier is a function of successful wins.

$$S_e^a(t) = S^a_{e\tau_e^a(t)}$$
$$S_c^a(t) = S^a_{c\tau_c^a(t)}$$

Bid-paying Mechanism

Winning classifier e_t^a pays its bids to the winning classifier c_{t-1}^a
Winning classifier c_t^a pays its bids to the winning classifier e_t^a

This mechanism rewards the former classifier that lead to its success, reinforcing the right decision.

Bid-paying Mechanism

Winning classifier e_t^a pays its bids to the winning classifier c_{t-1}^a
Winning classifier c_t^a pays its bids to the winning classifier e_t^a

This mechanism rewards the former classifier that lead to its success, reinforcing the right decision.
The bid is set as $b_1(e)S_e^a(t)$ for a exchange classifier, and $b_2(c)S_c^a(t)$ for a consumption classifier.

$$b_1(e) = b_{11} + b_{12}\sigma_e \tag{11a}$$

$$b_2(c) = b_{21} + b_{22}\sigma_c \tag{11b}$$

$\sigma. = \frac{1}{1+\text{number of \#'s in the classifier}}$. Higher uncertainty, lower bid.

Evolution of Strength I

$$
\begin{aligned}
S^a_{c\tau^a_c(t)} =& S^a_{c\tau^a_c(t)-1} - \frac{1}{\tau^a_c(t)-1}\Bigg[(1+b_2(c))S^a_{c\tau^a_c(t)-1} \\
&- \sum_e I^a_e(t) b_1 S^a_{e\tau^a_e(t)} - U_a(\gamma^a_{ct})\Bigg]
\end{aligned}
\tag{12}
$$

$$
\begin{aligned}
S^a_{e\tau^a_e(t)+1} =& S^a_{e\tau^a_e(t)} - \frac{1}{\tau^a_e(t)}\Bigg[(1+b_1(e))S^a_{e\tau^a_e(t)} \\
&- \sum_c I^a_c(t) b_c S^a_{c\tau^a_c(t)}\Bigg]
\end{aligned}
\tag{13}
$$

Evolution of Strength II

The external payoff after making consumption desicion

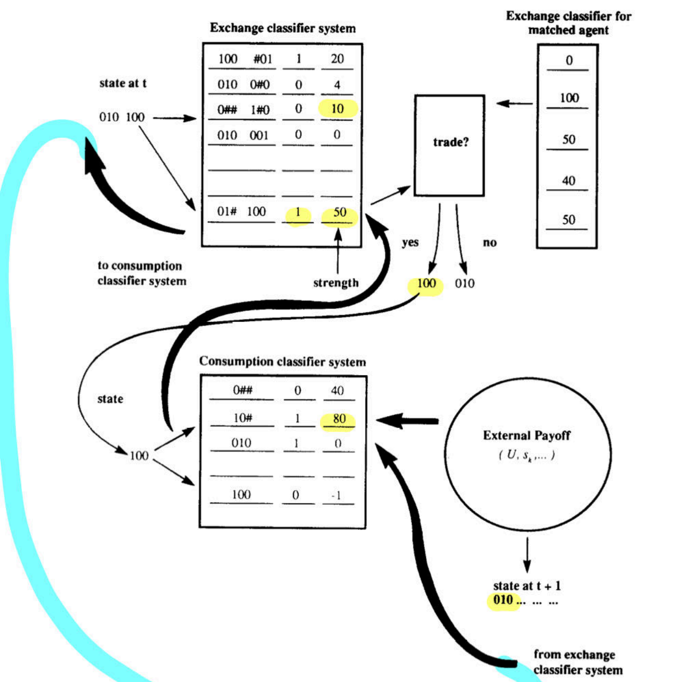
$$U_a(\gamma_{ct}^a) = \gamma_{ct}^a \left[u_i(x_{at}^+) - s(f(a)) \right] + (1 - \gamma_{ct}^a) s(x_{at}^+) \qquad (14)$$

Possible Typo

I think the authors might have a typo here. $(1 - \gamma_{ct}^a) s(x_{at}^+)$ should be a minus since it is the cost of holding.

Evolution of Strength III

- $S^a_{e\tau^a_e(t)}$ and $S^a_{c\tau^a_c(t)}$ evolves recursively
- Average of past payoff (External reward + bid from other classifier)
- minus payment (bids made to other classifier)
- Only winning classifier pays the bid (the sum term)
- Indexing with counter $\tau^a_{(\cdot)}(t)$: change is made only when successfully winning the auction.



Exchange classifier system
100 #01 1 20
010 0#0 0 4
0## 1#0 0 10
010 001 0 0
01# 100 1 50
state at t
010 100
to consumption
classifier system
strength
Exchange classifier for
matched agent
0
100
50
40
50
trade?
yes
no
100
010
Consumption classifier system
0## 0 40
10# 1 80
010 1 0
100 0 -1
state
100
External Payoff
(U, s_k,...)
state at t + 1
010 ... ...
from exchange
classifier system

Genetic Algorithm

Curse of Dimentionality

- As the variety of goods increase, the state space grows exponentially.
- Impossible to take all enumerations of classifiers into account when initializing.
- Genetic algorithm will be considered.

Curse of Dimentionality

- As the variety of goods increase, the state space grows exponentially.
- Impossible to take all enumerations of classifiers into account when initializing.
- Genetic algorithm will be considered.

Four steps for adding and deleting incumbent classifiers:

1. Creation
2. Diversification
3. Specialization
4. Generalization

Creation

Activation

There are no classifier that matches the current state z_{at}, i.e., $|M_e(z_{at})| = 0$.

Action

Assign random action to the current state z_{at} and add it to the collection of classifiers.

Diversification I

Activation

After the matched collection of classifiers $M_e(z_{at})$ are constructed.
If for all $e \in M_e(z_{at})$ have the same action.

Action

Assign opposite action to the current state z_{at} and add it to the collection of classifiers.
Assign the strength of the winning classifier to that of the new one.

There is also a deletion process

Deletion

Remove a "weak" classifier from the set of $M_e(z_{at})$. The weakness is defined jointly with the strength $S_e^a(t)$ and winning counter $\tau_e^a(t)$

Activation

After the winning bit has been determined. Activation probability decreases over time.
The winning classifier has some ambiguous position (#).

Action

Add a new classifier, which changes the #s in the condition part of the current winning classifier with some probability.
If the # is changed, it changes to the correspond value of the state.

Deletion

A weak rule is replaced by the new rule above.

Activation

Called randomly after the above variation steps are conducted.
The activation probability decreases over time.

Initialization

Draw *potential parents* and *potential exterminants*. The probability of drawing depends on some fitness criterion.

Generalizarion - Mating

1. Pick two parents to mate
2. Pick two position in the classifier
3. Pick to alter the *inner* or *outer* part of the slicing
4. Inconsistent positions are replaced with ambiguity symbol #

parent 1:	1	0	#	1	0	0	0
parent 2:	0	#	0	1	1	1	1
offspring 1:	1	0	#	1	#	0	0
offspring 2:	0	#	0	1	#	1	1

Fig. 4. The mating process for exchange classifiers who have drawn '3, 6' and 'in'.

5 Strength is set to be the average of its parents

Generalization - Exterminating

Remove one of the random selected classifier from the potential exterminants set.

I will report the simulation result next week.

Table 4

Description of the economies.

	Production					Storage cost					Utility	Initial	Equil.
	I	II	III	IV	V	1	2	3	4	5	u_i[a]	CS[b,c]	type[c]
A1.1	2	3	1	—	—	0.1	1	20	—	—	100	F	F
A1.2	2	3	1	—	—	0.1	1	20	—	—	100	R	F
A2.1	2	3	1	—	—	0.1	1	20	—	—	500	F	S
A2.2	2	3	1	—	—	0.1	1	20	—	—	500	R	S
B.1	3	1	2	—	—	1	4	9	—	—	100	F	F/S
B.2	3	1	2	—	—	1	4	9	—	—	100	R	F/S
C	2	3	1	—	—	0.1	20	70	0	—	100	R	F
D	3	4	5	1	2	1	4	9	20	30	200	R	—

[a] Utility levels u_i are set equal for $i = 1, 2, 3$.
[b] CS denotes 'classifier system'.
[c] 'F' implies fixed enumeration and 'R' implies randomly generated rules.