A Rewriting Game Theory Analysis of a Dynamic Router-Layer

By Ichiro Kokai

A thesis submitted to
School of Information Science,
Japan Advanced Institute of Science and Technology,
in partial fulfillment of the requirements
for the degree of
Master of Information Science
Graduate Program in Information Science

Written under the direction of
Associate Professor Rene Vestergaard

March, 2006

A Rewriting Game Theory Analysis of a Dynamic Router-Layer

By Ichiro Kokai (410043)

A thesis submitted to
School of Information Science,
Japan Advanced Institute of Science and Technology,
in partial fulfillment of the requirements
for the degree of
Master of Information Science
Graduate Program in Information Science

Written under the direction of
Associate Professor Rene Vestergaard

and approved by
Associate Professor Rene Vestergaard
Professor Kokichi Futatsugi
Professor Hiroakira Ono

February, 2006 (Submitted)

Abstract

Game theory was formulated by von Neumann and Morgenstern as a general theory of rational behavior. The theory aims to give insight into how strategies are formed and equilibrium points arise when agents interact. Equilibrium points are situation where all agents are happy, especially, the non-cooperative game of the equilibrium point are called Nash equilibrium point. Game theory is nowadays used in the fields of Economics, Biology, Law, Networking and so on. We are interested in the non-cooperative simultaneous game. The game theory in the simultaneous game is matrix-representation, so the game is real-valued preference and non-interference. Computer Network has a lot of dynamic aspects(i.e. P2P communication of router layer). We want to model such a network game to analyze, however, the network game with those dynamic aspects are non-uniformed and the interference relation will arise, so that the classic game theory is not appropriate, instead we aim to use abstract game and rewriting game theory, as proposed recently by LeRoux, Lescanne, and Vestergaard to analyze such a network.

In this thesis, we proposed the dynamic router-layer of network game, and we use the rewriting game theory to apply the game to analyze the 2routers and 2clients' network game of equilibrium points. And, we also used the LEDA and Mathematica as a tool to compute the equilibrium point automatically.

In the first chapter we will mention about the motivation and the aim. We will introduce the game theory and rewriting game theory in the chapter2 and chapter3. The automatic computation of rewriting equilibrium method will introduce in the chapter4, we will show you the algorithms and the methods. Finally we use model the router-lay of network game to analyze the dynamic aspects of the game. The knowledge of chapter3 and chapter4 will be used in this chapter. you will also see how the computation goes and what the results are. We conclude the 2routers&2clients network game in the last chapter and also mention the future work.

Contents

Chapter 1

Introduction

1.1 Background and Aim

Game Theory is nowadays used in the fields of Economics, Biology, Law, Networking and so on. Game Theory is a general theory of rational behavior. In Game Theory the model as a game, agent as a player and each agent has its own strategies to behave. The solution of Game Theory is the equilibrium points. Especially, the equilibrium point of the non-cooperative game are called Nash equilibrium point. Nash equilibrium point is a point making all agents' are happy. We are interested in the network game. Network usage has a lot of dynamic aspects such as the router-layer of the network game. However, the Classic Game Theory is inappropriate to model the dynamic network game, instead, we aim to use Abstract Game and Rewriting Game Theory, as proposed recently by LeRoux, Lescanne, and Vestergaard to study such the dynamic aspects of the network usages(e.g.P2P communication) in our network model. Our aim is to analyze where the dynamic game of the rewriting equilibrium points arise and what the situations are. The Mathematica and LEDA are used as tools for finding the rewriting equilibrium points of our model.

1.2 Structure of Thesis

The structure of our thesis is as follows, at first, in the chapter2 we explain the game theory which is the basic knowledge of our study. In the chapter3 we explain the rewriting game theory, rewriting game theory is the essential part of our thesis. We use this theory to study our model. In the chapter4 we explain how to find out the solution of rewriting game using tools. The solution is called rewriting equilibrium. In the chapter5 we define the router-layer of network model and we analyze the dynamic equilibria of 2by2 network game. In the final chapter we conclude our work and future work. We also include the results of our experiments and the tools' programming in appendix.

Chapter 2

Game Theory

We are doing various decision-makings in daily life. Our decision-making influence other people's decision-makings and vice versa. In the social life, our decision-making is interdependent.

let us take the personal computer as an example to see the decision-making in such an economic society. We are thinking whether to buy the personal computer now. First of all, we will see our income and a price of a computer to decide whether to buy. Secondly, we will decide which brands and computer shops are a good buy. It mainly depends on how we use the computer. But also, the reputation of personal-computer makers, the past sales performance, the design and the specification will influence our decision-makings. Finally, we care about the support service and the warranty period.

On the other hand, the personal-computer makers should decide the design, the performance the sales price and the business strategy of the new computer while searching for consumer's demand trend and the strategy of the same market of competitors. Moreover, the computer maker exports the computer not only the domestic market but also outside the country, a foreign market also will be influenced by the business strategy of the computer maker. Sometimes, an intergovernmental international negotiation is done about the tariff duty and the amount of export when the trade friction between two countries happened. The industrial policy of the government has a big influence on computer maker's business strategy.

Thus, consumer, a computer maker, a computer shop, a foreign firm, a home country government, and foreign governments, those decision-makings are complicated interdependence in various ways in greater or lesser degrees. So, Two or more players compete or interact with each other toward each target as if a kind of game in the situation of the decision-making in such an economic society. From this resemblance, game theory was formulated by John von Neumann and Oskar Morgenstern.

Game theory was established as a field in its own right after the 1944 publication of *Theory of games and Economic Behavior* by mathematician John von Neumann and economist Oskar Morgenstern. Game Theory nowadays can apply to not only economics or politics, but also psychology, biology and even networking. The concepts of game theory provide a language to formulate, structure, analyse, and understand strategic scenarios. So, Game theory is a good tool to study a mathematical model. A model is

usually based on a set of rules.

Model's basic requirements(rules)

- Game - A game(model) is a formal description of a strategic situation
- Player - A player is an agent who makes rational decisions in a game
- Strategy - In a game in normal form, a strategy is one of the given possible actions of a player
- Payoff - A payoff is a number, also called utility, that reflects the desirability of an outcome to a player

Game theory is generally divided roughly into the cooperative game and the non-cooperative game. In the cooperative game theory, on the assumption that the cooperation between players, it analyses the result of the player of the tie-up action and the cooperation. In the non-cooperative game theory, on the assumption that the non-cooperation between players, it analyses a variety of economic actions of competition and cooperation at a level of decision-making of individual player. We are going to see the non-cooperative game in detail. Because we are interested in Nash equilibrium point.

2.1 Glossary of Symbols

- $i \in \mathcal{A}$ means i is a member of the set $\mathcal{A}$
- $\bigotimes$ is for Cartesian product
- $\mathbb{R}$ is the set of real number
- $\succ_i$ means i can convert left to right
- $\prec_i$ means i can convert right to left
- $\succ\prec_i$ is equivalent strategies for i
- Eq^{N} is Nash Equilibria
- Se-Eq^{N} is sequential game of Nash Equilibria
- Sim-Eq^{N} is simultaneous game of Nash Equilibria
- $\lhd_i$ is for preference symbol, means i prefers right rather than left
- $\sum$ is for summation of numbers
- $\prod$ is for multiplication of numbers
- $\bigcup$ is for set union
- $\emptyset$ is the empty set

2.2 Non-cooperative Game

Non-cooperative game has two kinds of games, sequential games and simultaneous games. Sequential games are games in an extensive form. Simultaneous games are games in a strategic(normal) form. Sequential games are a game where one agent chooses his action

before the others choose theirs. Simultaneous games are a game that agents take action at the same time to select their strategies.

2.2.1 Sequential Games

A sequential game is one in which agents make decisions following a certain predefined order, and in which at least some agents can observe the moves of players who preceded them. A sequential game is a form of game tree with the pay-off functions in the leaves and agents(players) in internal nodes. In addition, edges from nodes to leaves make up the game tree. We assume a simple sequential game with the game tree is following.
- A binary game tree **bG** with leaves **gL** and nodes **gN**
- A function, called the pay-off functions $\mathcal{P}$
- Agents $\mathcal{A}$ make choices, called strategies **bS**

The sequential game in the extensive game as we mentioned is a game tree. The games is in the leaves. The agent receive pay-off as an outcome. So, the pay-offs can be defined as follows.

Definition1(Games)

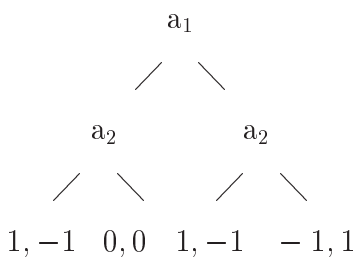
$$\mathbf{bG} ::= \mathbf{gL}\ \text{Payoffs} \mid \mathbf{gN}\ \mathcal{A}\ \mathbf{bG}\ \mathbf{bG}$$
$$\text{Payoffs} : \mathcal{A} \longrightarrow \mathcal{P}$$

The agents make choices from a root or nodes to the left edge or the right edge in the tree that depends on their strategies. So, the strategy and the choice can be defined as follows.

Definition2(Strategies, Choices)

$$\mathbf{bS} ::= \mathbf{sL}\ \text{Payoffs} \mid \mathbf{sN}\ \text{Choice}\ \mathcal{A}\ \mathbf{bS}\ \mathbf{bS}$$
$$\text{Choice} ::= \mathbf{l} \mid \mathbf{r}$$

Finally we use those definitions to make a game tree. We will always assume that the pay-off is given by a real number in the classic game theory.
Therefore, in our game tree you could say the outcome is 1 if you won the game, 0 if it is a draw, -1 if you lose it. However, in other games the result is more complicated or intangible. Let us see how the game tree creates.



In order to let the game to be as a draw(0,0), agent a_1 choses left strategy then agent a_2 choses right strategy, it can also represent like; a_1 **l** [a_2 **r** $\{(1,-1),(0,0)\}$] by definition2.

2.2.2 Simultaneous Games

A simultaneous game is a game in normal form. A simultaneous game is a matrix game in which players simultaneously choose their strategies. The resulting payoffs are presented in a table with a cell for each strategy combination.

A simultaneous game consists of:
- $\mathcal{A}$ is a nonempty set of agents
- $\{S_i\}_{i\in\mathcal{A}}$ is a nonempty set of agents' available strategies.
- P_i: $S_\mathcal{A} = \bigotimes_{i\in\mathcal{A}} S_i \to \mathbb{R}$, is a pay-off function of agent i. $S_\mathcal{A}$ is the strategy profiles and $\mathbb{R}$ is the set of real numbers.

Therefore, the simultaneous game can be defined with 3-tuples, $\mathcal{A}$, $\{S_i\}_{i\in A}$ and P_i.

In the simultaneous game, when the agents' a nonempty set of strategies S_i is finite, the game is called a finite game otherwise an infinite game. We are mainly study the finite game.

$$Sum = \sum_{i=1}^{n} P_i(\mathrm{s}_1, ..., \mathrm{s}_n)$$

P_i is the agent i of the pay-off function. All sets of strategies $s = (\mathrm{s}_1, ..., \mathrm{s}_n)$
The Sum on the above equation is 0 the game is called a zero-sum game. When sum is not 0, the game is called a non-zero-sum game. Especially the non-zero-sum game can be expressed as a bimatrix.

The two agents of the non-zero-sum game can be expressed as follows. The sets of strategies of agent1 and agent2 are $S_1 = \{1, ..., \mathrm{s}_m\}$ and $S_2 = \{1, ..., \mathrm{s}_n\}$.

$$H = \begin{pmatrix} (a_{11}, b_{11}) & \dots & (a_{1n}, b_{1n}) \\ \vdots & (a_{ij}, b_{ij}) & \vdots \\ (a_{m1}, b_{m1}) & \dots & (a_{mn}, s_{mn}) \end{pmatrix}$$

In the bimatrix above, the agent1 selects a row and the agent2 selects a column. In the (i,j) combination of (a_{ij}, b_{ij}), the left of the combination a_{ij} is the pay-off of the agent1 and the right of the combination b_{ij} is the pay-off of the agent2. Bimatrix H is called a payoff matrix of the non-zero-sum game of two agents. We assume a simple bimatrix game G with the resulting pay-off table to have a better understanding of the simultaneous game.

G	H_1	H_2
V_1	(0,0)	(1,1)
V_2	(1,0)	(2,2)

How the 3-tuples that we defined takes place in this game G.

- $\mathcal{A} = \{V, H\}$
Agent$V$ and agent$H$ are existing in game G.
- $\{S_V\}_{V \in \mathcal{A}} = \{V_1, V_2\}$,$\{S_H\}_{H \in \mathcal{A}} = \{H_1, H_2\}$
Agents have two strategies each.
- P(0,0) in game G is: $(P_{S_V}, P_{S_H})_{V,H \in \mathcal{A}}$, when the $S_V = V_1$, $S_H = H_1$ The pay-off function with the real numbers P.

2.3 Nash Equilibria of Non-cooperative Game

The solution of non-cooperation game is called equilibrium points. Equilibrium points in classic game theory especially are called Nash Equilibria. Nash Equilibria was proposed by John Nash. The concept of Nash Equilibria is a situation where all agents(players) are happy in the game. Technically speaking, Nash Equilibria is a kind of optimal collective strategy in a game involving two or more players, where no player has anything to gain by changing only his or her own strategy. In this section, we would see how Nash Equilibrium points arise when agents interact in the sequential game and simultaneous game.

2.3.1 Nash Equilibria of Sequential Games

We have already seen the sequential game in the last section. We defined two definitions for creating the game tree. The definitions of induced pay-off and convertibility are needed in the definition of Nash Equilibria.

Definition3(Induced pay-off)

$$\mathrm{OC}(\mathbf{sL}\ \mathrm{po}) \triangleq \mathrm{po}$$
$$\mathrm{OC}(\mathbf{sN}\ \mathrm{i}\ c\ \mathrm{s}_l\ \mathrm{s}_r) \triangleq \mathrm{OC}(\mathrm{s}_c)$$

An agent i will receive the outcome as a real number at the end of the game. c is the decision of the choice.

Definition4(convertibility)

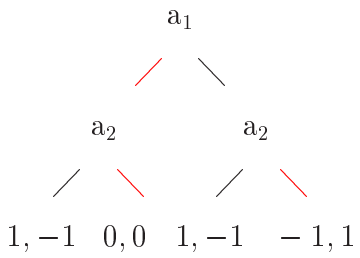
$$\frac{}{(\mathbf{sL}\ \mathrm{OC}) \succ\prec_i (\mathbf{sL}\ \mathrm{OC})} \qquad \frac{(i = i' \vee c_1 = c_2)\ \ s_l^1 \succ\prec_i s_l^2\ \ s_r^1 \succ\prec_i s_r^2}{(\mathbf{sN}\ i'\ c_1\ s_l^1\ s_r^1) \succ\prec_i (\mathbf{sN}\ i'\ c_2\ s_l^2\ s_r^2)}$$

The sequential Nash Equilibria in which the situation that no agent can get a better payoff by changing his or her strategies to left or right can be defined by Def.3 and Def.4 as follows. Please also refer the Glossary of symbols for your understanding.

Definition5(Nash equilibrium of sequential games)

$$\text{Se-Eq}^{\mathrm{N}}(s) \triangleq \forall i, \forall s'\, s \succ\prec_i s' \Rightarrow \neg(\mathrm{OC}(s)(i) < \mathrm{OC}(s')(i))$$

Consider, the Nash Equilibria of the following game tree.



We have seen this game tree in the section of sequential game. The pay-off is 1 if you won the game, 0 if it is a draw, -1 if you lose it. By the pay-offs, we can identify this game as a chess game. Nash Equilibria in sequential game usually think about agents' strategies(move) from the bottom to the top of the tree. This method is called backward induction. So, by the backward induction, the agent2 on the left node goes to right to get his or her outcome as a "draw" rather than the left of "lose" and the agent2 on the right node goes to left to get his or her outcome as a "win" rather than the right of "lose". By prediction of agent2's strategies the agent1 goes to left to get his or her outcome as a

"draw" rather than the right of "lose". Those three red edges are the Nash Equilibrium of this game. In this way, the sequential game of pure equilibrium can be defined by Nash Equilibria.

2.3.2 Nash Equilibria of Simultaneous Games

There are pure strategies and mixed strategies in the simultaneous games. The pure strategies in simultaneous game has a clear result of Nash Equilibria(pure Nash Equilibria), but mixed strategies does not. We are especially interested in Nash Equilibria of simultaneous games, because finding the Nash Equilibria in simultaneous games with mixed strategies is to assign probabilities to every pure strategies. Let us explain one by one. First, let us define the Nash Equilibirum of the pure strategy in the simultaneous games. The convertibility of the pure strategies of simultaneous games is to change his or her individual strategy. It means that agent with the pay-off of the vertical element is able to change row and agent with the horizontal element is able to change column.

Definition6(Nash Equilibrium of simultaneous games with **pure strategies**)

$$\text{Sim-Eq}^{\text{N}}_{S_{\mathcal{A}}}(ps) \triangleq \forall i, \forall ps' \, ps \succ \prec_i^{S_{\mathcal{A}}} ps' \Rightarrow \neg(P(ps)(i) < P(ps')(i))$$

ps is a strategy profile, ps $= \bigotimes_{i \in \mathcal{A}} S_i$. ps_i is the i-projection of ps. P(ps) indicates the one of cells of the pay-off table. Please refer the glossary for other symbols.

Let us consider, the following pay-off table.

H / V	H_1	H_2
V_1	$\Downarrow$ (0,0) $\Rightarrow$	(1,1) $\Downarrow$
V_2	(1,0) $\Rightarrow$	(2,2)

- $\mathcal{A} = \{V, H\}$
Agent$V$ and agent$H$ are existing in game G.
- $\{S_V\}_{V \in \mathcal{A}} = \{V_1, V_2\}$,$\{S_H\}_{H \in \mathcal{A}} = \{H_1, H_2\}$
Agents have two strategies each.
- Pay-off cell: $(P_{S_V}, P_{S_H})_{V,H \in \mathcal{A}}$, $S_V = \{V_1, V_2\}$, $S_H = \{H_1, H_2\}$

In this pay-off table: In the cell (0,0), the agent$H$ changes his or her strategy to the right cell for getting a better pay-off and on the right cell (1,1) the agent$V$ changes his or her strategy to the lower cell (2,2) to be a terminal strategy. In the cell (1,0), the agent$\{H, V\}$ changes his strategy to the right cell (2,2) to be a terminal strategy. In this way, the colored cell (2,2) is the pure Nash Equilibria in this game.

Finally, we consider the mixed strategies in simultaneous games. We study how to compute the mixed Nash Equilibrium with the probability distribution. The pay-off table below is the mixed strategies of simultaneous game.

V \ H	H_1	H_2
V_1	(1,0) $\Rightarrow$	(0,1) $\Downarrow$
V_2	$\Uparrow$ (0,1)	$\Leftarrow$ (1,0)

In each cell either agent$H$ or agent$V$ wants to change his or her strategies circularly. You can easily recognize that by the red-arrows in the pay-off table above. The equilibrium does not exist in the mixed strategies game. Therefore, Nash introduced the probabilities for finding equilibria in the mixed strategies of simultaneous game. The method is to distribute the probabilities to agents' individual strategies. The definition of mixed Nash Equilibria is following.

A mixed strategies of simultaneous game consists of:
- $\mathcal{A}$ is a nonempty set of agents($i \in \mathcal{A}$)
- $M_i \triangleq \{m_i : ps_i \to [0,1] \mid \sum_{ps_i \in S_i} m_i(ps_i) = 1\}$, for agent$i$,

the assignment of probability is from 0 to 1 , and the total sum is 1.
- $M_{\mathcal{A}} \triangleq \bigotimes_{i \in \mathcal{A}} M_i$, $M_{\mathcal{A}}$ is the set of probability-distribution profile.
- P_i: $S_{\mathcal{A}} = \bigotimes_{i \in \mathcal{A}} S_i \to \mathbb{R}$, is a pay-off function of agent i. $S_{\mathcal{A}}$ is the strategy profiles and $\mathbb{R}$ is the set of real numbers.
- $\mu(m, ps) \triangleq \prod_{i \in \mathcal{A}} w_i(ps_i)$, $w \in W_{\mathcal{A}}$, are the multiplication of the probability-distribution of the ps_i (a cell).
- Ex-P$(m)(i) \triangleq \sum_{ps \in S_{\mathcal{A}}} \mu(m, ps) \cdot \mathrm{P}(ps)(i)$, the expected pay-off function(Ex-P) is the summation of the multiplication of agent$i$'s pay-off(a real number) and the probability on the cell.

Now we can define the Nash Eauilibria of the mixed strategies by above elements. The convertibility in this mixed strategies is redistribution of the probabilities assigned by agents.

Definition7(Nash Equilibrium of simultaneous games with **mixed strategies**)

$$\text{Sim-Eq}^{\mathrm{N}}_{M_{\mathcal{A}}}(m) \triangleq \forall i \in \mathcal{A}. m \succ \prec_i^{M_{\mathcal{A}}} m' \Rightarrow \neg(\text{ex-P}(m)(i) < \text{ex-P}(m')(i))$$

Let us use Def.7 to compute and distribute the probability of the following simultaneous game.

		Y	$1-Y$
	H / V	H_1	H_2
X	V_1	(1,0)	(0,1)
$1-X$	V_2	(0,1)	(1,0)

The total of the probability of H_1 and H_2 is 1, therefore we assign Y probability to H_1 and $1-Y$ probability to H_2. Likewise, we assign X probability to V_1 and $1-X$ probability to V_2. $(0 \leq X \leq 1, 0 \leq Y \leq 1)$

Therefore, the expected pay-offs(Ex-P) of each pure strategies are;

$$\text{Ex-P}(V_1) = XY \cdot 1 + X(1-Y) \cdot 0 = XY$$

$$\text{Ex-P}(H_1) = XY \cdot 0 + X(1-Y) \cdot 1 = X(1-Y)$$

$$\text{Ex-P}(V_2) = (1-X)Y \cdot 0 + (1-X)(1-Y) \cdot 1 = (1-X)(1-Y)$$

$$\text{Ex-P}(H_2) = (1-X)Y \cdot 1 + (1-X)(1-Y) \cdot 0 = (1-X)Y)$$

So the expected pay-off of Agent$V$ and Agent$H$ are;

$$\text{Ex-P}(V) = \text{Ex-P}(V_1) + \text{Ex-P}(V_2) = XY + (1-X)(1-Y)$$

$$\text{Ex-P}(H) = \text{Ex-P}(H_1) + \text{Ex-P}(H_2) = X(1-Y) + (1-X)Y$$

We assign $X = 1$, if the probability of Y_1 is more than $\frac{1}{2}$ (which means $Y > \frac{1}{2}$), because that is the optimal probability-distribution for agent$H$.
We assign $X = \{0 \leq X \leq 1\}$, if the probability of Y_1 is $\frac{1}{2}$ (which means $Y = \frac{1}{2}$), because that is the optimal probability-distribution for agent$H$.
We assign $X = 0$, if the probability of Y_1 is less than $\frac{1}{2}$ (which means $Y < \frac{1}{2}$), because that is the optimal probability-distribution outcome for agent$H$.

In a similar way, we assign $Y = 1$, if the probability of X_1 is more than $\frac{1}{2}$ (which means $X > \frac{1}{2}$), because that is the optimal probability-distribution for agent$V$.
We assign $Y = \{0 \leq Y \leq 1\}$, if the probability of X_1 is $\frac{1}{2}$ (which means $X = \frac{1}{2}$), because that is the optimal probability-distribution for agent$V$.
We assign $Y = 0$, if the probability of X_1 is less than $\frac{1}{2}$ (which means $X < \frac{1}{2}$), because that is the optimal probability-distribution for agent$V$.

		$\frac{1}{2}$	$\frac{1}{2}$
	H / V	H_1	H_2
$\frac{1}{2}$	V_1	(1,0)	(0,1)
$\frac{1}{2}$	V_2	(0,1)	(1,0)

So that, the Nash Equilibria of probability distribution is the intersection of X and Y (which means $Inter(X, Y) = Inter(\frac{1}{2}, \frac{1}{2})$)

2-player's probability of calculation can compute above, but more players in the game, then the computation is very complicate and take a long time. Instead, we would like to use an abstract game of rewriting game theory. In the next chapter, you will see how the simultaneous game of classic game theory changed to and enlarged.

Chapter 3

Rewriting Game Theory

A rewriting game theory is proposed by LeRoux, Lescanne, and Vestergaard. It is a new approach to study Nash Equilibria especially in simultaneous games. The rewriting game theory consists of abstract games. Simultaneous game in classic game theory needs outcomes as number in order to use probability theory that Nash proposed. But, equilibria in abstract game do not use probability theory and therefore, rewriting equilibria are discrete and dynamic in nature. Advantage of rewriting game theory can see the equilibrium point of cases of non-uniform(causality) and remove the complex probability computation. In this chapter, we will show, how the rewriting game theory is defined.

3.1 Abstract Games

Abstract games is kind of rewriting and enlarging simultaneous games. Abstract games removed two rules of simultaneous games of classic game theory; the use of a vector space for pay-offs and the insistence that agents' choices are independent of each other. Instead, Abstract game derived two new notations; the convertibility relationship and the preference relationship, so that, the agents' choice are interference with each other. Let us see the definition of abstract games next.

Definition8(Abstract games)

-$\mathcal{A}$ is a non-empty set of agents
-S is a non-empty set of synopses, synopses in classic game theory is called strategies
-$\{\succ_i\}_{i\in\mathcal{A}}$, two synopses if agent i can convert the first to the second
-$\{\lhd_i\}_{i\in\mathcal{A}}$, two synopses if agent i prefers the second to the first

In this way, the abstract games can be defined with 4-tuples, $< \mathcal{A}, S, \{\succ_i\}_{i\in\mathcal{A}}, \{\lhd_i\}_{i\in\mathcal{A}} >$.

Nash Equilibria in abstract games can be rewritten as follows;

$$\mathrm{Eq}^N(s) \triangleq \forall s', i.s \succ_i s' \Rightarrow \neg(s \lhd_i s')$$

Nash equilibria in abstract games means that, for all strategies s', agent i can change his or her strategy from s to s', even though agent$i$ does not change, because agent i does not prefer strategies s'.

Abstract games has the change-of-mind relation. It can be defined by the convertibility relation and preference relation as follows.

Definition9(Change-of-mind relation of agent i)

$$\frac{s \succ_i s' \quad s \lhd_i s'}{s \rightsquigarrow_i s'}$$

$\rightsquigarrow_i$ is the notation of change-of-mind relation. From the Def.9, we can easily take notice of that, the change-of-mind relation is the intersection of the convertibility relation and the preference relation, in other words, if agent i can convert his or her strategies from s to s' and agent i prefers strategy s', then agent i changes his or her strategies to s'.
Abstract games proposed the irreducible of change-of-mind relation as follows:

Proposition 1(Irreducible of change-of-mind relation)

$$\rightsquigarrow \triangleq \bigcup_{i \in \mathcal{A}} \rightsquigarrow_i$$

$$s \in IrR_{\rightsquigarrow} \Leftrightarrow \forall s', i.s \succ_i s' \Rightarrow \neg(s \lhd_i s')$$

The irreducible of change-of-mind relation is the point, in which, agent i can convert his or her strategies but agent$i$ does not prefer. Straightforward, we can say that the irreducible of change-of-mind relation is Nash equilibria of the classic games. In the next section, we will define the rewriting equilibria.

3.2 Rewriting Equilibria

Simultaneous games in classic game theory do not only have a pure strategy profile, but also have a mixed strategy profile. So, prop.1 can be only applied to the pure strategies of simultaneous games. Because the choices of mixed strategies are dynamic. In this section we are going to define the rewriting equilibria(complete equilibria) in which guarantees to all strategies of simultaneous games.

V \ H	H_1	H_2
V_1	(1,0) $\Rightarrow$	(0,1) $\Downarrow$
V_2	$\Uparrow$ (0,1)	$\Leftarrow$ (1,0)

The pay-off table of the mixed strategies

The change-of-mind relation that agents made constructs a cycle in the table above. A cycle means that the pay-off is eternally dynamic. So, Nash assigned the probabilities for each pure strategies and defined that as the Nash equilibrium points. However, in the rewriting game theory, we do not assign the probabilities. Instead, we defined those a cycle of change-of-mind relation as one group. We defined that a cycle of change-of-mind is an equivalent class of the synopses(strategies).

V \ H	H_1	H_2
V_1	(1,0) $\Rightarrow$	(0,1) $\Downarrow$
V_2	$\Uparrow$ (0,1)	$\Leftarrow$ (1,0)

The part of the individual colored cell is the equivalent class of the synopses

The form of an equivalent class of synopses are called a strongly connected component. We will explain this property in detail later on. Actually, the rewriting equilibria is the colored part in this simultaneous game. The definition of the strongly connected components is following.

Definition10(Strongly Connected Components)

$$\lfloor s \rfloor \triangleq \{ s' \mid s \rightsquigarrow^* s' \wedge s' \rightsquigarrow^* s \}$$

$\rightsquigarrow^*$ is reflexive, transitive closure of $\rightsquigarrow$. In other words, it is ok to take any steps to reach to s' from s, and vice versa. $\lfloor s \rfloor$ is an equivalent class of the synopses. The complicate mixed strategies of simultaneous games have many equivalent classes. Given this factor, we define a progressive change-of-mind relation in which a change-of-mind step goes between two strongly connected components. The definition is as follows:

Definition11(Progressive Change-of-Mind Relation)

$$\frac{s \rightsquigarrow s' \quad \lfloor s \rfloor \neq \lfloor s' \rfloor}{\lfloor s \rfloor \curvearrowright \lfloor s' \rfloor}$$

The progressive change-of-mind relation($\curvearrowright$) is that, the strategy changes from s to s' and s and s' are in the different strongly connected components then, change the classes from $\lfloor s \rfloor$ to $\lfloor s' \rfloor$. The graph of ($\curvearrowright$) is acyclic. The rewriting equilibria is the terminal of strongly connected components.

Definition12(Rewriting Equilibrium)

$$\mathrm{Eq}^R(s) \triangleq \forall s' \in \mathcal{S}.(s \rightsquigarrow s' \Rightarrow s' \in \lfloor s \rfloor)$$

The rewriting equilibria is the terminal of strongly connected components.

3.3 Comparison of $\mathbf{Eq^N}$ and $\mathbf{Eq^R}$

The irreducible of change-of-mind is the pure Nash equilibrium.

$$Eq^N(s) \Leftrightarrow s \in IrR_{\rightsquigarrow}$$

The irreducible of progressive change-of-mind is the rewriting equilibria.

$$\mathrm{Eq}^R(s) \Leftrightarrow \lfloor s \rfloor \in IrR_{\curvearrowright}$$

We are especially interested in the progressive change-of-mind relation, because we use this way of finding equilibrium points for a new approach to games. In the next chapter, you will see how we compute the rewriting equilibrium generally.

Chapter 4

Computing $\mathbf{Eq}^R$

Prologue

We introduce the concepts and techniques involved in understanding and computing strongly connected component(SCC). We do it because rewriting GT uses graphs and identifies rewriting Eq^R as terminal nodes in a scc graphs. Our SCC algorithm is due to Tarjan. The algrithm fro finding SCC graph is documented in the LEDA book.

4.1 Create Graph

This section is the starting point of computing the rewriting equilibria. To be able to compute the rewriting equilibria, we use graph to compute. We assume the structure of the graph mainly has two elements, a node and an edge. We also assume that the graph as G, set of nodes as V and set of edges as E, is $G\{V, E\}$. A node v_1 belongs to V, is $\{v_1 \in V(G)\}$ and a edge e_1 belongs to E, is $\{e_1 \in E(G)\}$. An edge e contains two nodes, is $e(v_1, v_2)$. Let $e(v_1, v_2)$ to be $v_1 \to v_2$. See the following example.

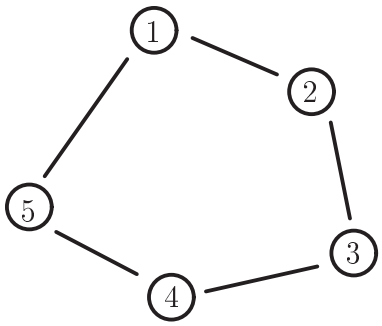


Figure.1 $G:\quad V = \{1, 2, 3, 4, 5\}$
$E = \{\{1, 2\}, \{2, 3\}, \{3, 4\}, \{4, 5\}, \{5, 1\}\}$

Before using these compositions to create graph, let us go back to the rewriting game

theory. We have studied rewriting game theory in the last chapter. And it says, the pure strategies of simultaneous games if the strategy in the cell is the irreducible of the change-of-mind($s \in IrR_{\rightsquigarrow}$), then that is the kind of the rewriting equilibria, because ($s \in IrR_{\rightsquigarrow}$) can not always be guaranteed to be able to reach an irreducible. What if the game includes the strongly connected components, such as, the mixed strategies of simultaneous games. By this factor, both change-of-mind relation and progressive change-of-mind relation are important when we create graph. First, let us create the graph without the strongly connected components.

H / V	H_1	H_2
V_1	$\Downarrow$ (0,0) $\Rightarrow$	(1,1) $\Downarrow$
V_2	(1,0) $\Rightarrow$	(2,2)

This is the pay-off table of pure strategies in which we have seen in the last chapter. Let us use matrix table to explain. First of all, let each cell(pure individual strategy) be separated as follows. And we assign the numbers on each cell.

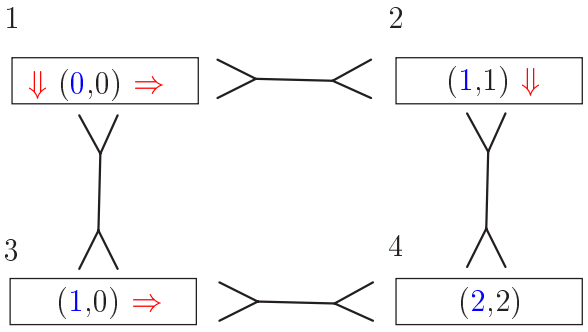


Figure.2 The relation of separated cells

Next, we use those elements to create graph. Rewriting game uses this notation$\{> - <\}$ to describe the convertibility of each cell. The arrows are change-of-mind relation and they give the direction to the graph. The numbers that we assigned are for defining nodes. Finally we are ready to create the graph. Let us see the Fig.3

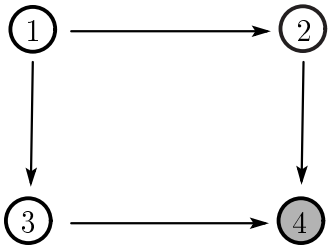


Figure.3 Create graph

- $G = \{V, E\}$
- $V = \{1, 2, 3, 4\}$, $E = \{\{1, 2\}, \{1, 3\}, \{2, 4\}, \{3, 4\}\}$

e(1,2) means player changes his strategy from node1 to node2. The colored node is the rewriting equilibria. We have created the graph without the strongly connected components. The graph with the strongly connected components is as follows.

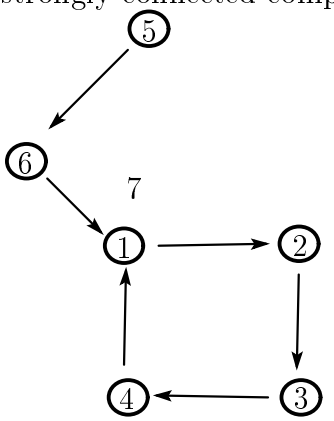


Figure.4 Graph with a strongly connected component

In this graph, the irreducible of the change-of-minds$s \in IrR_{\rightsquigarrow}$ can not be found. Because of $s \in IrR_{\rightsquigarrow} \Rightarrow \lfloor s \rfloor \in IrR_{\curvearrowright}$. Therefore, we need to let our graph be the progressive of change-of-mind graph in order to use the definition of $\lfloor s \rfloor \in IrR_{\curvearrowright}$. So, we follow the definition of classifying SCC and let the SCCs with progressive chinge-of-mind relation to be the acyclic graph is as follows. The Fig.5 is the shrunken graph of Fig.4

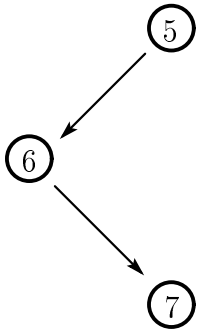


Figure.5 Shrunken graph

Again, the concept of creating a graph is to let Eq^R readable for the tool.

Summary of creating a graph

- Nodes - We assign numbers for each cell of matrix table. i.e. $V = \{1, 2, 3\}$
- Edges - The change-of-mind relations are edges. i.e. $e\{1, 2\}$ Agent changes strategies from node1 to node2.
- Shrink - If we have the cyclic of change-of-mind relation(strongly connected components) in the graph, we have to shrink the cyclic of change-of-mind to one node. We want the graph always to be acyclic, because the progressive change-of-mind graph can always guaranteed to be able to reach an irreducible, in other words, we can always get all of rewriting equilibrium.

In the next section, we will use tool to apply to graphs.

4.2 Tools and Computation

Our starting point of computing the rewriting equilibira was used Mathematica. Mathematic is a widely-used computer algebra system originally developed by Wolfram. It has a lot of packages, such as the graph theory package and the combinatorics theory package. We used the graph theory in Mathematica to analyze the rewriting equilibirua with the graph, and we succeeded to let it compute the strongly connected components(SCCs). However, we could not let it shrink the SCCs and compute the acyclic graph. So, we made a pattern matching algorithm for finding the arrows between SCCs, so that we had a kind of shrunken graph, because the algorithm and programming are not based on the algorithm of defining progressive change-of-mind relation, so we can not guarantee that we can always find all of the rewriting equilibrium points, and what's more, we do not know the complexity of our programming. Instead, we studied LEDA. LEDA is the C++ class library of efficient data types and algorithms. The concept of LEDA library is to provide algorithms as objects(object-oriented language) and data types. So, LEDA already has a lot of algorithms are implemented for dealing with graphs, moreover LEDA has a powerful graphic-user-interface is called Graphwin, so we can visually look at the graph. For that reason, we could find out an efficient algorithm for dealing with our graph, such that its complexity is linear . In this section, we use the flow chart of our programming(based on the LEDA algorithm) file to "graph.cpp" to introduce finding the rewriting equilibria from our graph.

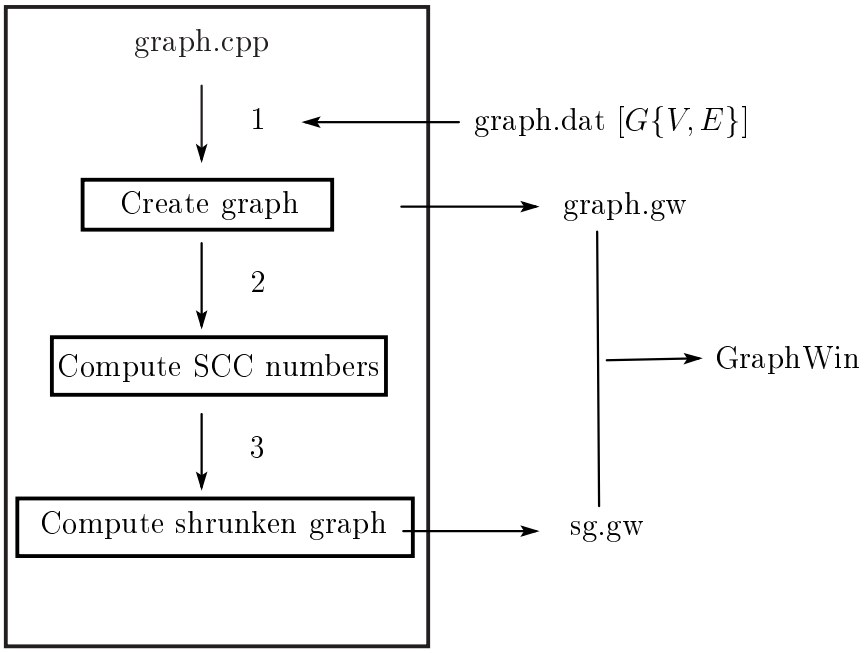


Figure.6 Flow chart of graph.cpp

The programming of computing the terminal SCCs that we wrote in the "graph.cpp file". The programming was written by C programming language with LEDA library. For the full of programming, please refer the appendix at the end. Let us follow the numbers in Fig.6 to introduce the detail of the algorithms.

1. First of all we create a "graph.dat file", this file contains the information of nodes and edges($G\{V, E\}$). Next, Let "graph.cpp" return the "graph.gw" from computation of "graph.dat". The "graph.gw" is the file in which the graphic-user-interface of GraphWin can display.
2. Then, we compute the SCC numbers, SCC numbers are the numbers for computing the shrunken graph later. Let us explain the algorithm of computing SCC numbers. Consider an directed graph, first of all, we distribute depth-first-search number -1 to all nodes and we do DFS, we increase the DFS number to 0 when we searched the node. In other words, DFS number of the reached nodes are 0 and unreached node are -1. We also distribute the component number -1 for all of nodes which are still tentative. Tentative means the nodes are still not either the SCC or a part of the SCC. Finally, we use two stacks, the stack of roots and the stack of unfinished to compute the SCC number. Stacks are stacking with reached nodes.

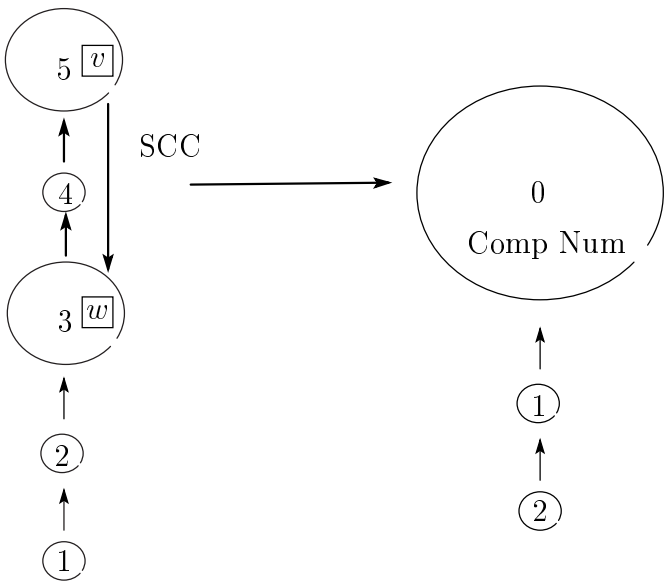


Figure.7 Computation of the SCC numbers

Let us consider the Fig.7. Now we are in the node5(v), and the next step of the DFS is to go back to node3(w). The DFS number in the w is 0, we know the nodes3,4,5 are the strongly connected components because the DFS number is 0 means we have already reached the node w. Now, situation of stacks are unfinished $\{1,2,3,4,5\}$ and roots $\{1,2,3,4,5\}$. Let us move to the next step of DFS. We are in the node w now. Our programming says when the node with DFS number is 0 and component number is -1, we pop the stack of roots' nodes until the node's number is smaller than node's number w. From Fig.7, the stack of roots are $\{1,2,3\}$. Finally we are ready to compute the SCC numbers. If v is the top of the stack of roots, we let the stack of unfinished pop while the v is not w. At the same time, we also assign the same SCC number to popped nodes from the stack of unfinished. Therefore, in the Fig.7, the top of the roots' stack number is 3, so we pop the nodes in the unfinished stack are $\{3,4,5\}$ and we distributed the same number to popped nodes as the SCC number. In other words, we are using the stack to let the strongly connected components pop up and give them a same number to classify. We use this kind of computation to find out strongly connected components.

3. Finally we compute the shrunken graph. We have given the SCC numbers to all of the nodes. First of all, we compute a new graph is call shrunken graph. If the highest SCC number is 5 then the shrunken graph will have 6 nodes and with number $\{0,1,2,3,4,5\}$. The algorithm of computing SCC will let the shrunken graph always be the partial order. The partial order guarantees that there is no directed edges from the lower number of nodes to the higher number of nodes. From this nature, we use the edges' data of "graph.gw" to check whether there exists the edges of source's SCC numbers are bigger than the edges of target's SCC numbers. If there exists, we add those edges in to the shrunken graph to complete. The nodes of shrunken graph are SCCs and it is acyclic. Therefore, Eq^R(terminal SCC) is the node with the none of outedge. Mathematica has the SCC computation as well, but the algorithm of distributing SCC numbers does not

exist. So, we could not compute the shrunken graph. However, I succeed to implement these kind of algorithms in Mathematic. We can use these tools and the algorithms for finding Eq^R in our network usage graph. In generally, this method can find terminal SCCs in any graphs with nodes and edges.

Chapter 5

Dynamic Network Model

5.1 Modeling

Rewriting game theory says that the rewriting equilibria in the game are dynamic. Network has many dynamic aspects of uses. Therefore, We model the network with this nature. In this section, you would see how our model are defined and what usage profiles are formed.

5.1.1 Model

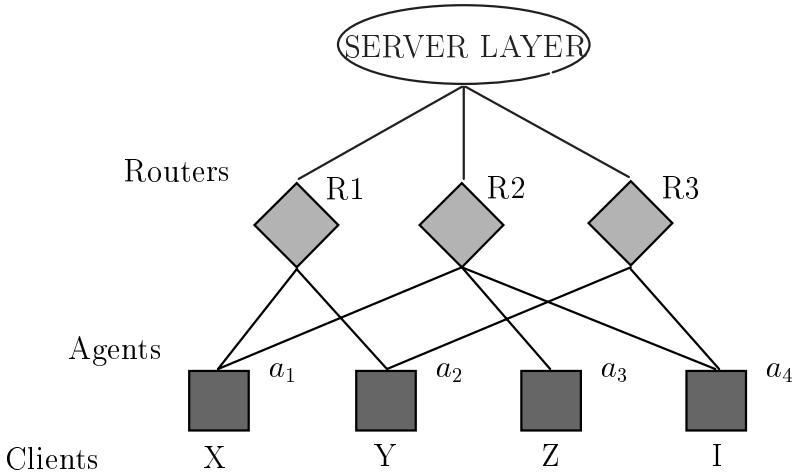


Figure.8 Network

The network structure of Fig.8 contains a set of Routers, a set of Agents and a set of Clients. We consider this kind of network structure in our network model. We assume that the agents do the jobs for clients. We also assume that the use of routers for the agents are synopses(strategies). Finally we are defined our network game by the abstract game. So, the rewriting game theory is used.

Definition8(Network Game)

- We defined the game with next 4-tuples $< A, S, (\succ_a)a \in A, (\lhd_a)a \in A >$
- A is the set of Clients.
- a is an agent $a \in A$, each client has an agent for dealing with their moves.
- $S = \bigotimes_{a \in A} Power(Connect(a))$, is the power set of the subset of connectivity to Routers.
  - R: set of Routers.
  - Connect: $A \to Power(\mathrm{R})$, is Clients connect to Routers.

  S is the power set of the subset of the connectivity of Routers for client a.
- $\{(\succ_a)a \in A\} \subseteq S \times S$, is the convertibility relation of client a.
  For example, ($s_1 \succ_a s_2$, $\{s_1, s_2\} \in S$), is s_1 and s_2 are the connectivity of routers for client a, so $(\succ_a)$ means agent a can convert from s_1 to s_2.
- $(\lhd_a)a \in A$, is the preference relation of client a.

Let us consider, how to define the situations of "Client X uses R1" and "Client X stops using R1", see also Fig.8.
- $[X] \succ_{a_1} [X \cup R1]$, "Client X uses R1". We defined $\{\cup\}$ as an action of using a router.
- $[X] \succ_{a_1} [X \setminus R1]$, "client X uses R1". We defined $\{\setminus\}$ as an action of stop using a router.

5.1.2 Usage Profiles(UP)

The rewriting games and the classic games have two differences. One is that the rewriting equilibria can be dynamic, so that lets the game to have not only the real-valued preference but also the arbitrary preference. Let us difference is the relation between agents in the classic games are non-interference, however the abstract games are able to express the interference relation. So, Our usage profiles are formed including the arbitrary preference and the interference relation.

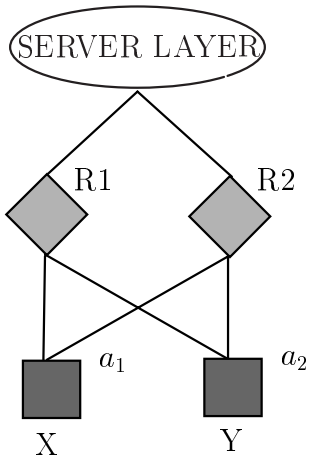


Figure.9 Simplest Network

We used the 2 clients and 2 routers network model(2by2) to consider the usage profiles, also see Fig.9. First of all, let us consider, what kind of network usages are the classic game form, and what kind of network usages are the abstract game form.

- Classic game form(Normal-form)
  $[X, Y] \succ_{a_1} [X \cup R1, Y]$, when client X uses router R1, client Y uses nothing.
  $[X, Y] \succ_{a_1} [X \setminus R1, Y]$, when client X stops using router R1, client Y uses nothing.

- Abstract game form(Dynamic-form)
  $[X, Y] \succ_{a_1} [X \cup R1, Y \setminus R1]$, when client X uses router R1, client Y stops using R1.
  $[X, Y] \succ_{a_1} [X \setminus R1, Y \cup R1]$, when client X stops using router R1, client Y uses R1.

Once again, the classic game theory can only express the simple situation such as the normal-forms above. It says, when X uses router R1 or X stops using R1, Y does noting. Which means the action of client X will never interfere the action of client Y. However, the rewritng game theory can express the situation of interference. The dynamic form above says that when the client X stops using router R1 client Y stops using R1. Which means the action of client X interfered the action of client Y. And, this situation means the monopolizing router R1. So, rewriting game theory can let the game be more arbitrary and involving the causality relation. Our usage profiles of 2by2 network model are formed by these natures of rewriting game theory as follows.

Basic step		**Interference**	**Typical Example**
$[X, Y] \succ_{a_1} [X \cup R1, \_\_\_]$	1	Y	Normal-formed
	2	$Y \cup R1$	P2P request
	3	$(Y \cup R1) \cup R2$	P2P relayed request
	4	$(Y \cup R1) \setminus R2$	Overloaded P2P relay
	5	$Y \cup R2$	?
	6	$Y \setminus R1$	Denial of Service(DoS)
	7	$(Y \setminus R1) \cup R2$	DoS w/fall-back
	8	$(Y \setminus R1) \setminus R2$	DoS on relay
	9	$Y \setminus R2$	?

Figure.10 Usage Profiles of Table(1)

When the client X uses the router, this usage profiles table covers the all cases of the client Y's actions. Let us see the interference situations one by one.
1. When the client X uses router R1, the client Y does noting is the normal-form and simply says "X uses router R1".
2. When the client X uses router R1, the client Y starts using the same router is the situation of P2P request. P2P communication is using for the file sharing between client and client through the router. So, client X requests using router R1 to do some file sharing

and client Y responds.
3. P2P relayed request is the situation of that client X requests to do the file sharing using router R1 with the other client, however the distance is faraway and it needs to via client Y, so client Y connects to router R1 and then connects to router R2 for responding client X's P2P communication.
4. Overloaded P2P relay is the situation of that when the P2P communication too much traffic goes through the router R1, and the data traffic is larger than the traffic capacity of router R2, then client Y got dumped.
5. We still do not know the situation of the client X uses router R1 that interferes client Y uses R2.
6. Denial of service is the situation that client X monopolizing router R1(e.g.using full capacity of router R1 to do the big data transfer or downloading), so that client Y can not connect to router R1.
7. Denial of service with fall-back is the situation of 6, but client Y starts his job uses router router R2 instead router R1.
8. Denial of service on relay is the situation of the client X starts monopolizing router R1 and client Y has to stop the P2P communication.
9. We still do not know the situation of the client X uses router R1 that interferes client Y uses R2.

Basic step		**Interference**	**Typical Example**
$[X, Y] \succ_{a_1} [X \setminus R1, \_\_]$	1	Y	Normal-formed
	2	$Y \cup R1$	Unblocking
	3	$(Y \cup R1) \cup R2$	Unblocking relay
	4	$(Y \cup R1) \setminus R2$	Unblocking w/fall-back
	5	$Y \cup R2$	?
	6	$Y \setminus R1$	Kill router / stop P2P
	7	$(Y \setminus R1) \cup R2$	Stop P2P w/fall-back
	8	$(Y \setminus R1) \setminus R2$	Stop relayed P2P
	9	$Y \setminus R2$	?

Figure.10 Usage Profiles of Table(2)

When the client X stop using the router, this usage profiles table covers the all cases of the client Y's actions. Let us see the interference situations one by one.
1. This situation is the normal-formed and it says the client X stops using the router R1.
2. This situation can be that client X stops monopolizing router R1, so the client Y starts using router R1 again.
3. This situation can be that client X stops monopolizing router R1, so the client Y starts to respond the P2P request from the other agents.
4. This situation can be that client X stops monopolising router R1, so the client Y starts to use the router R1 back instead using router R2, because client Y prefers using router R1.

5. We still do not know the situation of the client X stops using router R1 that interferes client Y starts using router R2.
6. This situation can be that client X stops doing P2P communication with the client Y using router R1.
7. This situation can be that client X stops doing P2P communication with the client Y using router R1, then client Y stops using router R1 and starts using router R2 for doing the other jobs.
8. This situation can be that client X stops doing P2P communication via the client Y, so that the client Y stops using routers R1 and R2.
9. We still do not know the situation of the client X stops using router R1 that interferes client Y stops using router R2.

We have seen the situation of the usage profiles. The general usage profiles are actually the following.

Definition9(Usage Profiles of 2by2 Network Game)

- $Z = \emptyset \ \vee \ Z = \{R1\} \ \vee \ Z = \{R2\}$, $\emptyset$ is the emptyset
- $\succ_{a_1,a_2} \subseteq S \times S$
- $< X, Y > \succ_{a_1,a_2} < X \cup Z, Y \cup Z >$
- $< X, Y > \succ_{a_1,a_2} < X \cup Z, Y \setminus Z >$
- $< X, Y > \succ_{a_1,a_2} < X \setminus Z, Y \cup Z >$
- $< X, Y > \succ_{a_1,a_2} < X \setminus Z, Y \setminus Z >$

By this definition, the usage profiles are the directed edges E in the usage graph G.

5.2 Usage Graphs(UG)

So far, we defined the abstract network game and the usage profiles(with situation) of the 2by2 network game. Finally, we create graph for study(compute) where the rewriting equilibrium with convertibility relation arise. The usage graph $G\{V, E\}$ is need for Eq^R computation. Therefore in this section, you will see how we create our network model by the graph and its rewriting equilibria computation.

5.2.1 Creation of Usage Graphs

Definition9(Usage Graph of 2by2 Network Game)

- We define Usage Graph(UG) as follows::
- $V = S \times S$, the all possible connection of routers for the agent a_1 and the agent a_2.
- E is the binary relation of V and also is the usage profiles, also see Def.9.

e.g. $([\emptyset, \emptyset], [\{R1\}, \{R2\}]) \in E(UG)$
e.g. $[\{R1\}, \{R1, R2\}] \in V(UG)$

5.2.2 Finding Rewriting Equilibrium points in usage graphs

We have studied LEDA and Mathematica. Through our study, we found out that LEDA had the line order of the algorithm for computing shrunken graph. However, the Mathematica of combinatorics theory can help us to compute the usage graph. So, we programmed the shrunken module into Mathematica based on the LEDA's algorithm of shrunken graph. Finally, we use the Mathematica to compute the terminal SCCs. The flow chart of finding terminal SCC as follows.

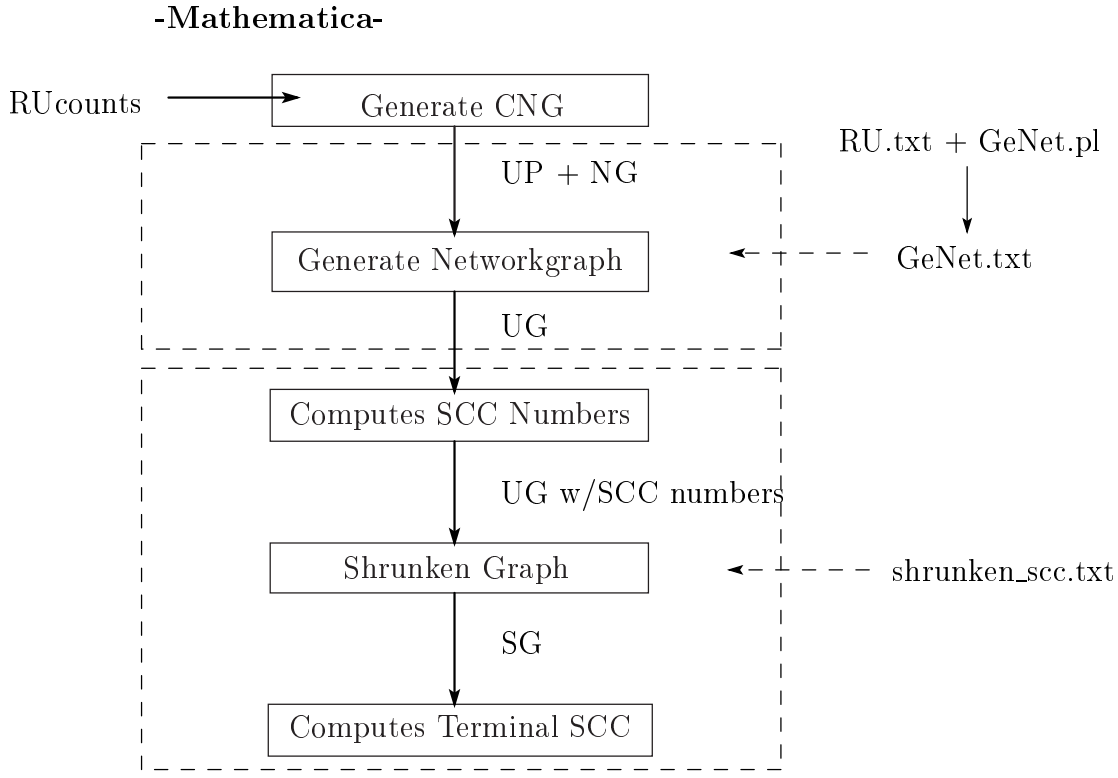


Figure.11 Mathematica Flow Chart

Let us use the flow chart above to explain how we compute the terminal SCC using Mathematica. First of all, we give the number of users(clients) and routers to start generating. We give the usage profiles and network game to generate networkgraph. The programming of generating networkgraph is saved in "GeNet.txt". Mathematica is using the user interface called front-end. The front-end is the note book form. And, "<< + filename.txt" is a command in which, let the front-end read the text and return back the result of computation. So, we wrote the program in the GeNet.txt. The "<<GeNet.txt" command gives us the networkgraph(UG) result back, see also Fig.12.

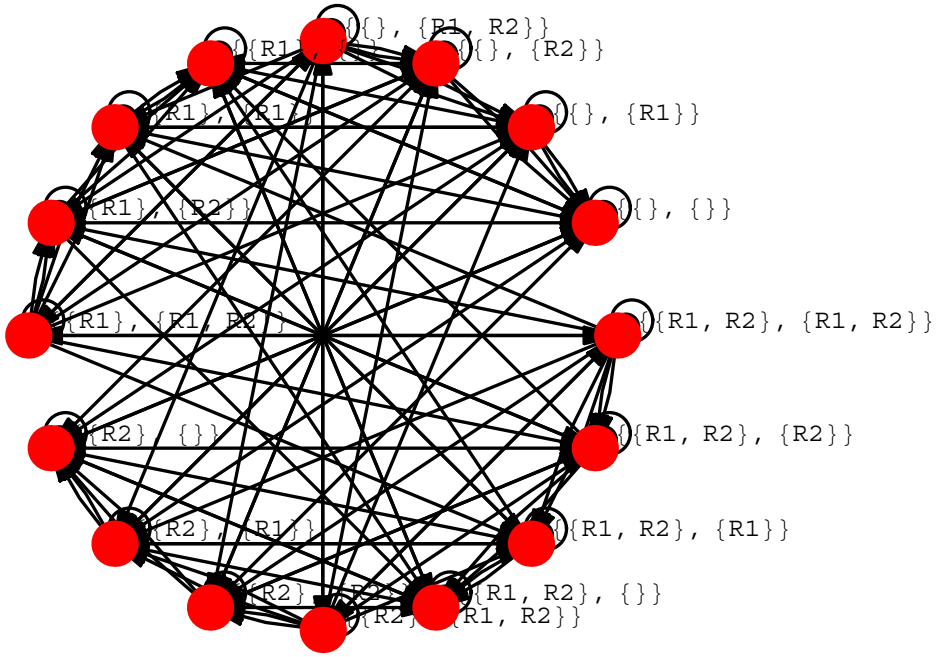


Figure.12 Networkgraph(UG)

The "GeNet.txt" returns the networkgraph(UG). In the "GeNet.txt" also deals with the UP computation. That gives us the reason of why we use the Mathematica rather than LEDA, because the computation of UP is the List of List structure, in C language, the list of list computation is difficult, however, the module type language of Mathematica is not as difficult as C language, the programming is as follows.

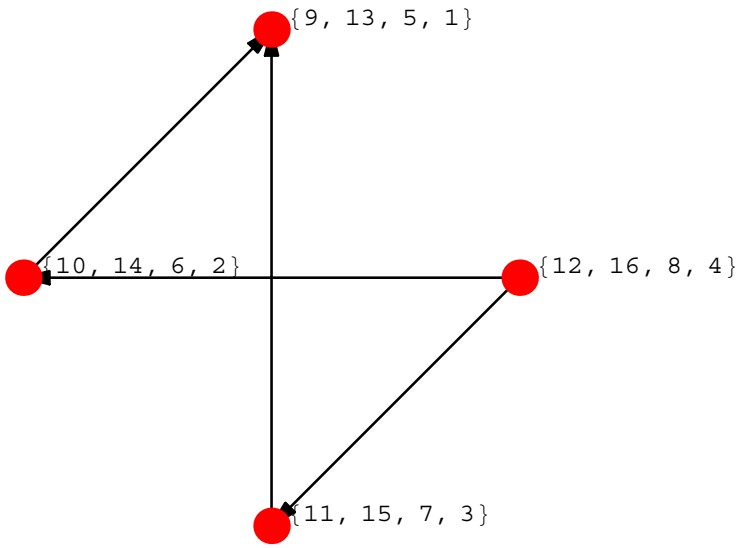
Computation of UP(Usage Profile)

unicom[{$x\_, y\_$}] :=
MapAt[Complement[#,y] &, (MapAt[Union[#,x] &, vertex,
Table[{$i, 1$}, {$i, lengthvertex$}]]), Table[{$i, 2$}, {$i, lengthvertex$}]]

The implementation of usage profile of $< X, Y > \succ_{a_1,a_2} < X \cup Z, Y \setminus Z >$

Next, we compute the SCC numbers using usage graph. We assigned SCC numbers to all nodes for identifying the strongly connected components when we compute the shrunken graph later. Now, we are ready for shrinking the graph. We programed the computing SCC numbers and shrunken graph based on the LEDA algorithm. After

shrunken the graph, we looked at all SCC nodes in which, the nodes do not have the out-edge as the terminal SCCs. These precessing all have done in the "shrunken_scc.txt", see also Fig.13.



$\{\{\{\},\{\}\} \to 1, \{\{\},\{R1\}\} \to 2, \{\{\},\{R2\}\} \to 3, \{\{\},\{R1,R2\}\} \to 4,$
$\{\{R1\},\{\}\} \to 5, \{\{R1\},\{R1\}\} \to 6, \{\{R1\},\{R2\}\} \to 7,$
$\{\{R1\},\{R1,R2\}\} \to 8, \{\{R2\},\{\}\} \to 9, \{\{R2\},\{R1\}\} \to 10,$
$\{\{R2\},\{R2\}\} \to 11, \{\{R2\},\{R1,R2\}\} \to 12, \{\{R1,R2\},\{\}\} \to 13,$
$\{\{R1,R2\},\{R1\}\} \to 14, \{\{R1,R2\},\{R2\}\} \to 15, \{\{R1,R2\},\{R1,R2\}\} \to 16\}$

Figure.13 Shrunkengraph

In the next section, we use Mathematica to study the rewriting equilibrium points among all usage profiles of the graphs.

5.3 Experiment and Results

We computed the all general usage profiles of usage graphs and those shrunken graphs of the 2by2 network game, and analyzed the equilibrium points. The results of computations are in the appendix.

Consider the "Shrunken_UG1.nb" of the **Appendix**.
- $< X, Y > \succ_{a_1,a_2} < X \cup Z, Y \cup Z >$ is the general UP(usage profile) of UG1
UG1 is the situation of the clients X and Y never stop using the routers(e.g. ClientX and

ClientY are doing P2P communications). The equilibrium point is $(\{R1, R2\}\{R1, R2\})$, also no nodes could be shrunken. The UG1 has the pure Eq^N(node16).

Consider the "Shrunken_UG2.nb" of the **Appendix**.
- $< X, Y > \succ_{a_1,a_2} < X \cup Z, Y \setminus Z >$ is the general UP of UG2
UG2 is the situation of the client X never stops using routers and client Y keeps stopping using routers(e.g. ClientX keeps monopolising routers, because his computer has broken down, the computer will send out as many files as possible while time allows). The equilibrium point is $(\{R1, R2\}\{\})$, also no nodes could be shrunken. The UG2 has the pure Eq^N(node13).

Consider the "Shrunken_UG3.nb" of the **Appendix**
- $< X, Y > \succ_{a_1,a_2} < X \setminus Z, Y \cup Z >$ is the general UP of UG3
UG3 is the situation of the client X keeps stopping using routers and client Y never stops using routers(e.g. ClientX stops monopolising routers and clientY is willing to use the routers in which clientX has been monopolised) . The equilibrium point is $(\{\emptyset\}\{R1, R2\})$, also no nodes could be shrunken. The UG3 has the pure Eq^N(node4).

Consider the "Shrunken_UG4.nb" of the **Appendix**
- $< X, Y > \succ_{a_1,a_2} < X \setminus Z, Y \setminus Z >$ is the general UP of UG4
UG3 is the situation of the client X and Y keep stopping using routers(e.g. When the router is broken, clients are dumped by routers, or clients are stopping doing P2P communications . The equilibrium point is $(\{\emptyset\}\{\emptyset\})$, also no nodes could be shrunken. The UG4 has the pure Eq^N(node1).

Let us consider, what if those general usage profiles of the usage graphs are combined.
Consider the "Shrunken_UG1_UG2.nb" of the **Appendix**
- $< X, Y > \succ_{a_1,a_2} [< X \cup Z, Y \cup Z > \& < X \cup Z, Y \setminus Z >]$ are the general UP combination of UG1 & UG2
This is the situation of the client X keeps using routers and client Y sometimes uses and stops using routers. Our shrunken graph says the nodes has been shrunken, so that, we have the SCCs and the irreducible SCC is the equilibrium point of the game are called Eq^R. So, in this kind of combined usage graph, client Y has a dynamic moves and the equilibrium points is dynamic. The rewriting game theory is a theory for dealing with the dynamic aspect like this kind of usage graph. Eq^R 13,14,15,16 is $[(\{R1, R2\}\{\emptyset\})$, $(\{R1, R2\}\{R1\})$, $(\{R1, R2\}\{R2\})$, $(\{R1, R2\}\{R1, R2\})]$.

Consider the "Shrunken_UG1_UG3.nb" of the **Appendix**
- $< X, Y > \succ_{a_1,a_2} [< X \cup Z, Y \cup Z > \& < X \setminus Z, Y \cup Z >]$ are the general UP combination of UG1 & UG3
This situation is similar to UG1 & UG2 case. The client X has the dynamic moves. Therefor the shrunken graph of terminal scc is Eq^R (4,8,12,16) is $[(\{\emptyset\}\{R1, R2\})$, $(\{R1\}\{R1, R2\})$, $(\{R1, R2\}\{R1\})$, $(\{R2\}\{R1, R2\})$, $(\{R1, R2\}\{R1, R2\})]$

Consider the "Shrunken_UG1_UG4.nb" of the **Appendix**
- $< X, Y > \succ_{a_1,a_2} [< X \cup Z, Y \cup Z > \& < X \setminus Z, Y \setminus Z >]$ are the general UP combination of UG1 & UG4
The UG1 and UG4 are taking opposite jobs which means both client X and Y are dynamic. The graph shrinks to one nodes, the equilibrium points are all nodes. Eq^R(1,2,3,4,5,6,7, 8,9,10,11,12,13,14,15,16) is $[(\{\emptyset\}, \{\emptyset\}),$
$(\{\emptyset\}, \{R1\}), (\{\emptyset\}, \{R2\}), (\{\emptyset\}, \{R1, R2\}), (\{R1\}, \{\emptyset\}), (\{R1\}, \{R1\}), (\{R1\}, \{R2\}),$
$(\{R1\}, \{R1, R2\}), (\{R2\}, \{\emptyset\}), (\{R2\}, \{R1\}), (\{R2\}, \{R2\}), (\{R2\}, \{R1, R2\}), (\{R1, R2\},$
$\{\emptyset\}), (\{R1, R2\}, \{R1\}), (\{R1, R2\}, \{R2\}), (\{R1, R2\}, \{R1, R2\})]$.

Consider the "Shrunken_UG2_UG3.nb" of the **Appendix**
- $< X, Y > \succ_{a_1,a_2} [< X \cup Z, Y \setminus Z > \& < X \setminus Z, Y \cup Z >]$ are the general UP combination of UG2 & UG3
The UG2 and UG3 are taking opposite jobs which means both client X and Y are dynamic. The graph shrinks to one nodes and Eq^R is same as the UG1 & UG4 case.

Consider the "Shrunken_UG2_UG4.nb" of the **Appendix**
- $< X, Y > \succ_{a_1,a_2} [< X \cup Z, Y \setminus Z > \& < X \setminus Z, Y \setminus Z >]$ are the general UP combination of UG2 & UG4
This situation is the client Y keeps stopping using the routers, and the client X is dynamic. The Eq^R(1,5,9,13) is $[(\{\emptyset\}, \{\emptyset\}), (\{R1\}, \{\emptyset\}), (\{R2\}, \{\emptyset\}), (\{R1, R2\}, \{\emptyset\})]$

Consider the "Shrunken_UG3_UG4.nb" of the **Appendix**
- $< X, Y > \succ_{a_1,a_2} [< X \setminus Z, Y \cup Z > \& < X \setminus Z, Y \setminus Z >]$ are the general UP combination of UG2 & UG4
This situation is the client X keeps stopping using the routers, and the client Y is dynamic. The Eq^R(1,5,9,13) is $[(\{\emptyset\}, \{\emptyset\}), (\{R1\}, \{\emptyset\}), (\{R2\}, \{\emptyset\}), (\{R1, R2\}, \{\emptyset\})]$

Epilogue

We see that is all the examples we have our expectation to what the likely usage of a network based on different usage proflies matches what we compute to be Eq^R using mathematica on our network.

Chapter 6

Summary

6.1 Conclusion

In the 2by2 network game, the usage graph with one general usage profile can not shrink, so that the pure Nash equilibrium point always exists. However, the usage graph with two or more general usage profiles can be shrunken and a terminal scc always exists, for that reason, the rewriting equilibrium point always exists in the combination of UPs graph. From those factors we can conclude the network game of router-lay game can not be analyzed by classic game theory without Nash probability theory. As we mentioned in the chapter2, the computation of probability is very complicate and the probability distribution makes us difficult to understand the situation of why those pay-offs are cyclic. By contraries, the rewriting game theory will treat the terminal SCC as a rewriting equilibrium, so that makes us easy to define as a situation. We also found out If the client has the opposite general usage profiles in the same usage graph, then the client will always have the dynamic moves. Especially if the case of the both client X and client Y have the opposite general usage profiles, then the graph will shrink to the one node, therefore, the rewriting equilibrium is the usage graph itself. What the equilibrium point says is, if the client can only keep using routers or stopping using routers, then the game will have the Nash equilibrium point and, if the client can use and stop using, then the game will have the dynamic aspect, so that rewriting equilibrium point arise. See also appendix.

6.2 Related Work

In classic game theory the network model consists of m agents and a source and a sink computer connected through n wires and using Nash equilibria, Papadimitriou and others have established a O(ln n / ln ln n) upper bound between router and idealized regulated network traffic. Also the study the probability of Nash theory to analyze the latency. See also reference of [11],[12].

6.3 Future Work

We studied the general 2by2 network game usage profiles with the usage graph, however, the specific usage profiles might have a lot of interesting cases to analyze (such as two or more terminal sccs exists in the same usage graph, the equilibrium point does not exist and so on). The computation might take time, however, to study ..by.. network game is good for understanding network usage profiles and graphs. The finding terminal SCC program written by Mathematica can use to any kinds of games which are able to create the graphs, hence, to polish the tool helps us to study rewriting equilibrium.

Acknowledgment

To be able to accomplish this study, Our supervisor, Professor Rene gave me a lot of good advice, his advice is gratefully acknowledged. We also thank LDL members, assistant Iwagaki-san and Professor Futatugi.

Reference

[1] S. スキエン=著/植野義明=訳 「Mathematica 組み合わせ論とグラフ理論 離散数学を実現する」 株式会社トッパン, 1992

[2] Martin J. Osborne and Ariel Rubinstein, editors. 「A Course in Game Theory」 Mit Press, 1994

[3] 浅野哲夫, 小保方幸次=共著 「LEDA で始める C/C++プログラミング-入門からコンピュータ・ジオメトリまで-」 株式会社 サイエンス社, 2002

[4] Kurt Mehlhorn and Stefan Naher, editors. 「LEDA: A Platform for Combinatorial and Geometric Computing」 Cambridge Univ Press, 1999

[5] R. ディーステル=著/根上生也・太田克弘=訳 「グラフ理論」 ジュプリンガー・フェアラーク東京 株式会社, 2000

[6] Guillermo Owen, editor. 「Game Theory」 Academic Press, 1995

[7] Andrew S.Tanenbaum, editor.「Computer Networks International Edition」 Prentice Hall, 2002

[8] 鈴木光男=著 「新ゲーム理論」 株式会社 けい草書房, 2000

[9] 石畑清=著 「アルゴリズムとデータ構造」株式会社 岩波書店, 1998

[10] Stephane Le Roux, Pierre Lescanne, and Rene Vestergaard. Rewriting Games and their Equilibrium Points

[11] Christos H. Papadimitriou. Algorithms,Games,and the Internet

[12] Marios Mavronicolas, Paul Spirakis. The Price of Selfish Routing

Appendix

- 1. graph.cpp
- 2. GeNet.txt
- 3. shrunken_scc.txt
- 4. UG