An Efficiency Improvement of the Equilibrium Solution Search on the Selfish Routing Game by Removing Redundant Paths

Koichi YOSHIDA \textsuperscript{*}, Takashi OKAMOTO \textsuperscript{*}, and Seiichi KOAKUTSU \textsuperscript{*}

Abstract: The selfish routing game is a mathematical model to represent the behavior of selfish players who select a path in a congested network. In the equilibrium solution search on the selfish routing game, the amount of flow passing through each path is designated as the decision variable. Therefore, it is difficult to obtain the equilibrium solution of the selfish routing game in large-scale networks with a vast number of paths in a realistic time. In many cases, flows pass through a few part of the paths only and no flow passes through the other paths in the equilibrium solution of the selfish routing game in large-scale networks. If some of the paths which are zero-flow paths in the equilibrium solution can be removed from the decision variables in advance, the efficiency of the equilibrium solution search is expected to be improve. This paper proposes a new solution search method to improve the efficiency of the equilibrium solution search by removing redundant paths which can be detected in advance by considering a condition with respect to the equilibrium solution. The effectiveness of the proposed method is confirmed through numerical experiments.

Key Words: network routing, selfish routing, replicator dynamics, Nash equilibrium solution.

1. Introduction

In social and biological systems, there exist many populations which consist of a large number of selfish players interfering with each other. In such populations, the objective of each player often conflicts with the total objective of the population, and thereby the problem called social dilemma arises. The selfish routing game introduced by Wardrop in 1950s \cite{1} is a mathematical model to represent the social dilemma on the network routing. In the selfish routing game, the result of the selfish routing are known to converge to a Nash equilibrium solution \cite{2}. For the network routing under the same condition, it has been shown that the average cost of all players in the equilibrium solution of the selfish routing game can be worse than the average cost of all players in the optimal network routing \cite{3,4}. In the selfish routing game, the arise of the Braess’s paradox \cite{5,6} is also well-known. The paradox shows more network resources may cause worse traffic performance contrary to expectations. It has been reported that some phenomena similar to Braess’s paradox arise in realistic networks \cite{7}–\cite{10}. Hence, the computer simulation of the selfish routing game is expected to contribute to the optimal network design and the traffic control planning.

The equilibrium solution of the selfish routing game can be obtained without violating any constraint by using the replicator dynamics \cite{11}–\cite{13}. In the equilibrium solution search, the amount of flow passing through each path is designated as the decision variable. Therefore, it is difficult to obtain the equilibrium solution of the selfish routing game in large-scale networks with a vast number of paths in a realistic time. In many cases, flows pass through a few part of the paths only and no flow passes through the other paths in the equilibrium solution of the selfish routing game in large-scale networks. If some of the paths which are zero-flow paths in the equilibrium solution can be removed from the decision variables in advance, the efficiency of the equilibrium solution search is expected to be improve.

This paper proposes a new method to improve the efficiency of the equilibrium solution search by removing the redundant paths which can be detected in advance by considering a condition with respect to the equilibrium solution. The effectiveness of the proposed method is confirmed through numerical experiments.

2. Selfish Routing Game

2.1 Selfish Routing Game and Its Equilibrium Solution

Consider the network like Fig. 1. The network consists of $E$ edges. The flow passes from the source node $s$ to the sink node $t$. There are $P$ paths that connect $s$ to $t$. Let $x_p$, $p = 1, \ldots, P$, be the amount of flow passing through the $p$th path. Let $X$ be the amount of the total flow. $x_g$ has to satisfy the linear equality constraint and the non-negative constraint defined by

\begin{equation}
\sum_{p=1}^{P} x_p = X, \quad (1a)
\end{equation}
\[ x_p \geq 0, \quad p = 1, \ldots, P. \] (1b)

Let \( \mathcal{P}_e, e = 1, \ldots, E \) be the set of path numbers including the \( e \)th edge. Let \( f_e(x) = \sum_{p \in \mathcal{P}_e} x_p \) be the amount of the flow passing through the \( e \)th edge. Each edge \( e \) has a non-negative cost function \( c_e(f_e(x)), e = 1, \ldots, E \), and the cost function monotonically increases with respect to \( f_e(x) \). The cost on use of the \( p \)th path is defined by

\[ \tilde{c}_p(x) = \sum_{e \in \mathcal{E}_p} c_e(f_e(x)), \quad p = 1, \ldots, P, \] (2)

where \( \mathcal{E}_p \) is the set of edge numbers that are included in the \( p \)th path. The average cost of the network is defined by

\[ C(x) = \frac{1}{X} \sum_{p=1}^{P} x_p \tilde{c}_p(x). \] (3)

In the selfish routing problem, each selfish player chooses the path so as to minimize the passing cost. The equilibrium state is that each player cannot reduce his/her passing cost by changing his/her own path. That is, in the equilibrium solution \( x^* \), for \( p \) satisfying Eq. (1) and \( x_p > 0 \),

\[ \forall q \in \{1, \ldots, P\} \quad \tilde{c}_p(x^*) \leq \tilde{c}_q(x^*) \] (4)

holds. Let \( \hat{P} = \{ p \mid x_p > 0 \} \) be the set of path numbers whose amount of flow is not zero in the equilibrium solution. Then, the following equations hold for the equilibrium solution \( x^* \).

The optimization problem whose optimal solution is the equilibrium solution for which Eq. (5) holds is formulated as the following potential function minimization problem [2]:

minimize \( \Phi(x) \)

subject to \( \sum_{p=1}^{P} x_p = X \), \( x_p \geq 0, \quad p = 1, \ldots, P \),

where \( \Phi(x) = \sum_{e=1}^{E} \int_{0}^{\gamma_e(x)} c_e(y) dy \).

The following equations are derived from the optimality condition for the optimization problem (6):

\[ \left( \frac{\partial \Phi(x^*)}{\partial x_p} - \phi^* \right) x^*_p = 0, \quad p = 1, \ldots, P, \] (7a)

\[ \frac{\partial \Phi(x^*)}{\partial x_p} - \phi^* \geq 0, \quad p = 1, \ldots, P, \] (7b)

where \( \phi^* \) is the optimal Lagrange multiplier with respect to the equality constraint (6b). Here, \( \frac{\partial \Phi(x)}{\partial x_p} \) is given by

\[ \frac{\partial \Phi(x)}{\partial x_p} = \sum_{e \in \mathcal{E}_p} c_e(f_e(x)) = \tilde{c}_p(x), \quad p = 1, \ldots, P. \] (8)

When \( x^*_p > 0, \quad p = 1, \ldots, P \), by substituting Eq. (8) into Eq. (7a),

\[ \tilde{c}_p(x^*) = \phi^*, \quad p = 1, \ldots, P \] (9)

is obtained. When \( x^*_p = 0 \) holds for some \( q \in \{1, \ldots, P\} \), Eq. (9) holds for the components except \( x^*_q \). For \( x^*_p \) from Eq. (7b),

\[ \tilde{c}_p(x^*) \geq \phi^* \] (10)

holds. From Eq. (3), the passing cost of the path whose amount of flow is zero does not affect the average cost \( C(x) \), and thereby the following equation:

\[ \phi^* = C(x^*) \] (11)

is obtained. Thus, the optimal solution of the optimization problem (6) gives the equilibrium solution for which Eq. (5) holds.

2.2 Solution Search Method by Using Replicator Dynamics

The solution of the optimization problem (6) can be obtained by using the replicator dynamics, which is used to search the evolutionarily stable strategy solution of the evolutionary game [11]–[13]. The replicator dynamics for the selfish routing game, that solves the optimization problem (6), is given by

\[ x_p(k + 1) = x_p(k) - \alpha x_p(k) \left( \tilde{c}_p(x(k)) - C(x(k)) \right), \quad p = 1, \ldots, P. \] (12)

The total amount of change in all variables at an iteration of the replicator dynamics is given by

\[ \sum_{p=1}^{P} x_p(k + 1) - x_p(k) \] (13)

Hence, the replicator dynamics searches an optimal solution without violating the equality constraint (6b) when the initial point satisfying the constraint (6b).

In the stationary point of the replicator dynamics (12) represented by \( x^* \), the \( p \)th variable \( (p = 1, \ldots, P) \) satisfies either of the following equations:

\[ x^*_p = 0 \quad (14a) \]

\[ \tilde{c}_p(x^*) = C(x^*), \] (14b)

Hence, the stationary point of the replicator dynamics (12) satisfies the optimality conditions (7) when the initial point satisfying the constraint (6b) and \( x_p > 0, \quad p = 1, \ldots, P \).

3. Equilibrium Solution Search with Removing Redundant Paths

In the equilibrium solution search method using the replicator dynamics, the number of decision variables is equal to the number of paths connecting source \( s \) to sink \( t \). Therefore, it is difficult to obtain the equilibrium solution of the selfish routing game in large-scale networks with a vast number of paths in a realistic time. For example, when the network has at least one closed-loop, an infinite number of paths that goes around the closed-loop exists. Furthermore, even if the closed-loop does not exist, the number of paths in the complete graph of \( N \) nodes becomes \( P = \sum_{k=0}^{N-2} (N-2)! / n! \).

In the equilibrium solution in the large-scale networks, in many cases, only a small number of paths have flow, and
3.1 Removing the Path Including the Closed-Loop

Consider the network shown in Fig. 2. Let the first path \( p = 1 \) be the path \( s \to v_1 \to t \) which does not include the closed-loop. Let the second path \( p = 2 \) be the path \( s \to v_1 \to v_2 \to v_1 \to t \) which includes a closed-loop trip. The passing cost of each path is given by

\[
\tilde{c}_1(x) = c_1(x) + c_4(x), \quad (15a)
\]

\[
\tilde{c}_2(x) = c_1(x) + c_2(x) + c_3(x) + c_4(x). \quad (15b)
\]

Considering the condition with respect to the equilibrium solution given by Eq. (5), \( c_k(x) = c_3(x) = 0 \) has to be satisfied in order for \( x_2 > 0 \). However, the zero-cost closed-loop is negligible in searching the equilibrium solution. Hence, the path including closed-loops can be removed from the decision variables of the selfish routing game.

3.2 Removing the Path with Redundant Constant Cost

The passing cost of the \( p \)-th path \( \tilde{c}_p(x) \) can be regarded as

\[
\tilde{c}_p(x) = \tilde{c}_p^{cs} + \tilde{c}_p^{cp}(x), \quad p = 1, \ldots, P, \quad (16a)
\]

\[
\tilde{c}_p(0) = \tilde{c}_p^{cs}, \quad p = 1, \ldots, P, \quad (16b)
\]

\[
\tilde{c}_p^{cp}(x) \geq 0, \quad p = 1, \ldots, P. \quad (16c)
\]

\( \tilde{c}_p^{cp} \) is the constant cost to pass the \( p \)-th path. The constant cost is the passing cost irrespective of the amount of flow through the path. \( \tilde{c}_p^{cp}(x) \) is the congestion cost to pass the \( p \)-th path. The congestion cost increases with the increase of the flow in the path. From the condition with respect to the equilibrium solution given by Eq. (5) and Eq. (16), \( x_p^0 = 0 \) holds when \( \tilde{c}_p^{cs} > C(x^*) \). Hence, the path with the constant cost which is larger than the average cost of the equilibrium solution can be removed from the decision variable of the selfish routing game. In other words, if the following condition:

\[
\tilde{c}_p^{cs} \leq C(x^*), \quad (17)
\]

is satisfied, then the \( p \)-th path can be active in the equilibrium solution of the selfish routing game, and thereby the \( p \)-th path has to be considered in the solution search.

Generally, \( C(x^*) \) is not known in advance. Instead, an estimation method of the cut-off cost \( \tilde{C}_{cut} \) to detect the redundant path is considered. The target value of \( C_{cut} \) is \( C(x^*) \). When \( C_{cut} = C(x^*) \), the effect of the removal of redundant paths is maximized. From Eq. (17), the sufficient condition for obtaining the equilibrium solution is given by

\[
C_{cut} \geq C(x^*). \quad (18)
\]

An estimation method that gives the cut-off cost \( C_{cut} \) satisfying Eq. (18) using the following equations:

\[
\begin{align}
C_{cut} &= \min_{p \in \{1, \ldots, P\}} \tilde{c}_p(\tilde{x}^p), \quad (19a) \\
\tilde{x}_p^q &= \begin{cases} X & (q = p) \\ 0 & (q \neq p) \end{cases}, \quad q = 1, \ldots, P. \quad (19b)
\end{align}
\]

can be considered. \( \tilde{x}_p^q \) indicates that the maximum flow \( X \) passes through the \( p \)-th path only and no flow passes through the other paths. In Eq. (19), the most extreme case that all flow concentrates to a path is assumed for each path, then \( C_{cut} \) is estimated as the minimum cost of all paths under the assumption. For the cut-off cost estimated by Eq. (19), the following proposition, which indicates that the equilibrium solution can be obtained by using the cut-off cost, is proven.

**Proposition 1** The cut-off cost estimated by Eq. (19) satisfies the condition given by Eq. (18).

**Proof:** Let \( r \) be the argument at which \( \tilde{c}_r(\tilde{x}^r) \) takes the minimum value. Since the passing cost monotonically increases with the increase of the flow and the maximum flow passes through the path \( r \), the congestion cost \( \tilde{c}_r^{cp}(\tilde{x}^r) \) is maximized. Hence, for any decision variable \( x \) satisfying the constraints represented by Eqs. (6b) and (6c), the following equation:

\[
\tilde{c}_r(x) \leq \tilde{c}_r(\tilde{x}^r) \quad (20)
\]

holds. From Eq. (5) and Eq. (20),

\[
C_{cut} = \tilde{c}_r(\tilde{x}^r) \geq C(x^*) \quad (21)
\]

holds.

When the cut-off cost \( C_{cut} \) is estimated by using Eq. (19), \( C_{cut} - C(x^*) \) tends to increase with the increase of the total flow \( X \), and thereby the effect of the removal of redundant paths tends to decrease. Consequently, this paper proposes to estimate \( C_{cut} \) as

\[
C_{cut} = \min_{p \in \{1, \ldots, P\}} \tilde{c}_p(\tilde{x}^p(k)), \quad (22a)
\]

\[
\tilde{x}_p^q(k) = \begin{cases} kX & (q = p) \\ 0 & (q \neq p) \end{cases}, \quad q = 1, \ldots, P. \quad (22b)
\]

where \( k \in (0, 1] \). When \( k = 1 \), Eq. (22) is equivalent to Eq. (19), and the satisfaction of Eq. (18) is guaranteed. As \( k \) is decreased more, more redundant paths can be removed. On the other hand, when \( k < 1 \), \( C_{cut} < C(x^*) \) can arise, and thereby the obtained solution is not necessarily the equilibrium solution. However, for the estimation method using Eq. (22), the following proposition can be proven.

**Proposition 2** Assume at least one path be detected by using the cut-off cost estimated by Eq. (22) \(^1\). Let \( x^*_{cut} \) be the solution of the optimization problem (6) without the redundant paths detected by using the cut-off cost. If \( C(x^*_{cut}) \leq C(x^*) \), then \( C_{cut} \geq C(x^*) \), that is, \( x^*_{cut} \) is the equilibrium solution.

\(^1\) If no path is detected, the obtained problem by the cut-off cost is the original problem.
Proof Let $\mathcal{P}_{\text{cut}}$ be the set of cut path numbers, i.e.,
\[ \mathcal{P}_{\text{cut}} = \{ p \in \mathcal{P} | C_{\text{cut}} < c_p^\text{cut} \}. \] (23)

Since $C(x_{\text{cut}}^*) \leq C_{\text{cut}}, C(x_{\text{cut}}^*) \leq C_{\text{cut}} < c_p^\text{cut}, p \in \mathcal{P}_{\text{cut}},$ and thereby $x_{\text{cut}}^* = 0, p \in \mathcal{P}_{\text{cut}}$ holds. Hence, all cut paths are removed correctly, and thereby $C_{\text{cut}} \geq C(x_{\text{cut}}^*) = C(x^*)$ holds. \[ \square \]

3.3 Algorithm of the Proposed Method

This subsection proposes algorithms of the equilibrium solution search methods using the cut-off cost estimated by Eqs. (19) and (22).

In the first proposed method, which is called P. M. 1 hereinafter, the cut-off cost estimated by Eq. (19) is used. As explained in the foregoing subsection, the solution of the optimization problem (6) without the redundant paths detected by using the cut-off cost estimated by Eq. (19) is the equilibrium solution. The algorithm of the P. M. 1 is given as follows.

Algorithm of P. M. 1

1. Give the network consisting of $N$ nodes and designate the source $s$ and the sink $t$.
2. Compute the minimum constant cost from the source to each node (pruning operation).
3. Compute the cut-off cost $C_{\text{cut}}$ by using Eq. (19) with applying Dijkstra’s algorithm to the network where the cost of all edges are $c_i(x), x = 1, \ldots , E$.
4. Search all paths from the source $s$ to the sink $t$ by using the depth-first search with forbidding the duplex passages of the same node. In the search process, if the constant cost from the source $s$ to the $n$th node is larger than $C_{\text{cut}} - c_{n,j}^\text{min}$, then abort the search with respect to the $n$th node (pruning operation). Detect the redundant path whose constant cost is larger than $C_{\text{cut}}$. Designate the found paths without the detected redundant paths as the decision variables of the optimization problem (6).
5. Solve the optimization problem (6) with respect to the designated decision variables to obtain the equilibrium solution $x^*$. Output the obtained solution $x^*$ and terminate.

In the second proposed method, which is called P. M. 2 hereinafter, the cut-off cost estimated by Eq. (22) is used. In the proposed method, $k$ of Eq. (22) decreases from 1, until the number of remaining variables after removing redundant paths is lower than a given parameter $P_{\text{max}}$. Then, the solution of the optimization problem (6) with respect to the remaining variables, that is represented by $x_{\text{cut}}^*$, is obtained. Based on the proposition 2, if $C(x_{\text{cut}}^*) \leq C_{\text{cut}}$, then $x_{\text{cut}}^*$ is guaranteed to be the equilibrium solution. Otherwise, $C(x_{\text{cut}}^*) > C_{\text{cut}}$ holds. Hence, setting $C(x_{\text{cut}}^*)$ to the new cut-off cost $C_{\text{cut}}$ makes the new cut-off cost approach suitable cut-off cost. Then, the new obtained solution by the solution search with the new cut-off cost approaches or attains the equilibrium solution. Thus, even if an incorrect solution is obtained in the first search, the equilibrium solution is surely obtained by additional searches with setting the average cost of the obtained solution to the new cut-off cost. The algorithm of the P. M. 2 is given as follows.

Algorithm of P. M. 2

1. Give the network consisting of $N$ nodes and designate the source $s$ and the sink $t$. Give the parameter $P_{\text{max}}$ and a parameter $0 < \beta < 1$ which is the decrease ratio of $k$. Set $k = 1$ and $r = 1$.
2. Compute the minimum constant cost from the $n$th node to the sink $t c_{n,j}^\text{min}, n = 1, \ldots , N$ with applying Dijkstra’s algorithm to the network where the costs of all edges are $c_i(x), e = 1, \ldots , E$.
3. Compute the cut-off cost $C_{\text{cut}}$ by using Eq. (22) with applying Dijkstra’s algorithm to the network where the cost of all edges are $c_i(kX), e = 1, \ldots , E$.
4. Search all paths from the source $s$ to the sink $t$ by using the depth-first search with forbidding the duplex passages of the same node. In the search process, if the cost from the source $s$ to the $n$th node is larger than $C_{\text{cut}} - c_{n,j}^\text{min}$, then abort the search with respect to the $n$th node (pruning operation). Detect the redundant path whose constant cost is larger than $C_{\text{cut}}$. Designate the found paths without the detected redundant paths as the decision variables of the optimization problem (6).
5. If $r = 0$ or the number of the designated decision variables is lower than the parameter $P_{\text{max}}$, then go to the step 6. Otherwise, $k \leftarrow k\beta$ and go to the step 3.
6. Solve the optimization problem (6) with respect to the designated decision variables. Let the obtained solution be $x_{\text{cut}}^*$.
7. If $C(x_{\text{cut}}^*) \leq C_{\text{cut}}$, then output $x_{\text{cut}}^*$ as an equilibrium solution and terminate. Otherwise, $C_{\text{cut}} \leftarrow C(x_{\text{cut}}^*)$ and $r \leftarrow 0$. Then, go to the step 4.

In the P. M. 2, the solution search is implemented several times at worst in contrast to the P. M. 1. However, the number of variables for each solution search in the P. M. 2 is equal to or smaller than the number of variables in P. M. 1. If the number of the removed variables is sufficiently large, the computational cost of the P. M. 2 can be smaller than the P. M. 1. In addition, when the solution search is implemented more than twice, the following inequality holds with respect to the solution of the first search $x_{\text{cut}}^*$:
\[ C(x_{\text{cut}}^*) < C(x^*). \] (24)

In this case, the average cost of the network without removed paths is better than the average cost of the equilibrium state. This corresponds to the Braess’s paradox when the removed paths correspond to additional paths. The Braess’s paradox arises when the constant costs of the additional paths are less than the existing paths in general. However, the constant costs of the removed paths are larger than the constant costs of the existing paths. To the authors’ knowledge, the paradox seldom arises under this situation.

4. Numerical Experiments

This section verifies the effectiveness of the proposed methods through numerical experiments. The first experiment shows the computational cost of the solution search against the number of paths. The second experiment shows the effectiveness of the removal of redundant paths against the network size. In the last experiment, the effectiveness of the proposed methods is verified.
All numerical experiments are carried out under the following environment — CPU: Intel Core i7 4770K 3.5 GHz, RAM: 16 GB, OS: Windows 7 Professional, Programming Language: C/C++, Compiler: GNU g++ 4.9.2.

4.1 The Computational Cost of Solution Search against the Number of Paths

In this experiment, the computational cost of the replicator dynamics given by Eq. (12), which is used for the solution search, against the number of paths is investigated. Note that the results of this experiment is valid irrespective of the removal of the redundant paths, which is the proposed method. This experiment uses a layered network in which each node is connected with all the nodes of the adjacent deeper layer. The input (first) layer consists of a source node and the output (last) layer consists of a sink node. The structure of the middle layers are determined so that the number of paths is desirable. For example, when the desired number of paths is 100, the network has 2 middle layers and each layer consists of 10 nodes. The computational time is almost proportional to the number of paths. As can be seen from in Fig. 4, the computational cost increases with increasing of the number of paths. Figure 4 shows the computational time to obtain the average of search steps and computational cost.

The results of this experiment is valid irrespective of the results for the random graph is the estimated value based on its theoretical value due to vast computational time to obtain the results for the plot. The results of Ex. 1 show that the number of paths at 20 nodes of Ex. 1 for the random graph is the estimated value based on its theoretical value due to vast computational time to obtain the results for the plot. The results of Ex. 1 show that the number of paths is increased sharply with increasing of the number of nodes in all network models. In contrast, the number of paths is decreased significantly in the results of Ex. 2 compared to the results of Ex. 1. The larger the networks size becomes, the more remarkable the effectiveness becomes. In addition, the effectiveness...
models in Ex. 2 and Ex. 3. There are a large number of paths with a constant cost approximately equivalent to the minimum constant cost in the grid network, and thereby the removal of the path with redundant constant cost is somewhat less effective in the grid network.

4.3 Effectiveness of the Proposed Method

This experiment verifies the effectiveness of the proposed methods explained in the subsection 3.3. In this experiment, the proposed methods are compared with the equilibrium solution search method with only removing paths including the closed-loops. The algorithm of the compared method (C. M.) is given as follows.

**Algorithm of C. M.**

1. Give the network consisting of \( N \) nodes and designate the source \( s \) and the sink \( t \).
2. Search all paths from the source \( s \) to the sink \( t \) by using the depth-first search with forbidding the duplex passages of the same node. Designate the found paths as the decision variables of the optimization problem (6).
3. Solve the optimization problem (6) with respect to the designated decision variables to obtain the equilibrium solution \( x^* \). Output the obtained solution \( x^* \) and terminate.

C. M., P. M. 1, and P. M. 2 are applied to the selfish routing game under the same conditions. Their computational time required to obtain the equilibrium solution are compared. In addition, the variation of the effectiveness of P. M. 2 by changing the parameter \( P_{\text{max}} \) is investigated. Verified in the subsection 4.2, in the grid network, the difference of performances between C. M., P. M. 1, and P. M. 2 is smallest. Therefore, this experiment uses the grid network. The grid networks used in this experiment consist of \( 4 \times 4, 6 \times 6, \) or \( 10 \times 10 \) nodes. The cost function of the edge \( e, e = 1, \ldots, E \) is given randomly as Eq. (25). The amount of the total flow \( X \) is 1. The initial point of the replicator dynamics is set as \( x(0) = X/P, p = 1, \ldots , P \).

The step width \( \alpha \) of the replicator dynamics is 0.1. When the absolute update value of the variables are all below the tolerance \( \epsilon = 10^{-9} \), the dynamics is deemed to converge and the solution search is terminated. \( \beta \) of P. M. 2 is set to 0.5.

The results are shown in Tables 1–3. \( C_i(x^*) \) is the estimated cut-off cost for the \( i \)th solution search. \( C_i(x^*) \) is the average cost of the solution obtained in the \( i \)th solution search. Note that when \( i > 1, C_i^{\text{cut}} \) is equal to \( C_{i-1}(x^*) \). \( P_i \) is the number of variables (paths) remaining after removing redundant paths in the \( i \)th solution search. \( P_{\text{cut}} \) is the number of variables (paths) which have non-zero flow in the solution obtained in the \( i \)th solution search. C. T. is the computational time required to obtain the equilibrium solution. Note that the results of C. M. in Table 3 are the estimated value based on the results shown in Figs. 4 and 5, because it is difficult to obtain the equilibrium solution in a realistic time for the \( 10 \times 10 \) grid network.

As can be seen from Tables 1–3, P. M. 1 obtains the equilibrium solution faster than C. M. in all problems. In Table 1, P. M. 2 obtains the equilibrium solution faster than P. M. 1 when \( P_{\text{max}} = 1,000 \). In Table 2, P. M. 2 obtains the equilibrium solution faster than P. M. 1 except \( P = 10,000 \). In Table 3, P. M. 2 obtains the equilibrium solution faster than P. M. 1 in all cases. Thus, the effectiveness of P. M. 2 is confirmed, especially, in

![Fig. 5 Number of paths in the grid network.](image)

![Fig. 6 Number of paths in the random graph.](image)

![Fig. 7 Number of paths in the scale-free network.](image)
with estimated by Eq. (22). As can be seen from Table 2, P. M. 2 obtains the equilibrium solution actually in the first solution search, but the obtained solution is not guaranteed to be the equilibrium solution, since there are not many removable paths by using the cut-off cost before the first solution search without detected redundant paths in the step 6. As can be seen from Fig. 9, when β = 0.1 and β = 0.2, the cut-off for the first search is obtained by two estimations. With small β, the costs of edges $c_{ij}(kX)$ decrease is implemented under the flows concentrating to $P_1 = 3$ paths, and thereby $C_1(x^*)$ becomes large; therefore, $P_2$ becomes large and the computational time becomes large. Thus, it is desirable to set $P_{\text{max}}$ of P. M. 2 to a suitable value depending on the network size. Concerning this issue, the authors recommend setting $P_{\text{max}}$ to a larger value proportional to the performance of the computer so as to yield the effective performance of P. M. 2.

The variation of the effectiveness of P. M. 2 by changing the parameter β is investigated. In this experiment, the 10 × 10 grid network is used, $P_{\text{max}}$ is set to 1,000, and the other settings are same in the experiments of this subsection. The results are shown in Fig. 8. As can be seen from Fig. 8, the result with β = 0.5 is best. Computational time does not vary much when β ≥ 0.5. The variations of cut-off cost $C_{\text{cut}}$ by updates in the step 3 of the algorithm P. M. 2 are shown in Fig. 9. This graph shows variations with the applications of Dijkstra’s algorithm in the step 3 in the estimation process of the cut-off cost before the first solution search without detected redundant paths in the step 6.


