Proof of a Fair Opening

Gu and Giorgos

June 2020

1 Definition:

1.1 Items

1. **Player Set**.$P(k) := \{p_1, \ldots, p_j, \ldots, p_k\}$. If we have 3 players(player 1,2 and 3), the player set would be $P(3) = \{p_1, p_2, p_3\} = \{1, 2, 3\}$
2. **Grid Set**. $G(n) := \{g_1, \ldots, g_j, \ldots, g_n\}$. In Fig.1, the board game just have 7 grids, $G(7) = \{a, b, c, d, e, f, g\}$.
3. **Feasible Opening Move Set**. $M(k, n) := \{m_1, \ldots, m_j, \ldots, m_l\}$. m_j means the opening moves of all players(after swap round, just before every one choose their second move). m_j could be $(a, c, d), (b, c, e)$ *or* (c, e, b). If $m_j = (c, e, b)$, it means the first player after swap moves c but not the player 1. The number of elements in $M(k, n)$ is $l = A_n^k = \frac{n!}{(n-k)!}$.
4. **State**. $s(m) := \{s_1^m, \ldots, s_j^m, \ldots, s_k^m\}$. If the k players finish their opening moves m, they will have their winning rates. s_j^m means if all players move m, the jth player's winning rate. $\sum_{j=1}^{k} s_j^m = 100$, the total win rate will be 100 percent.
5. **Feasible State Set**. $S(M) := \{s(m_1), \ldots, s(m_j), \ldots, s(m_l)\}$.
6. **Game Level**. Every player's game level is different. Someone is master and someone is a newbie. We can use AI to simulate it. $S(M) = S'(M)$ if and only if the players' game level of S, S' are the same.

1.2 Hypothesis

1. **Independent**: There is no union and everyone fights for himself.
2. **Selfish**: Everyone wants a higher winning rate for himself but not a balanced state for the others.
3. **Compact**: Everyone knows the game's feasible state set S.
4. **With Same Level**: Every player has the same game level. so if $s(d, b, e) = (50, 20, 30)$, then $s(b, e, d) = (20, 30, 50)$.

Figure 1: The Board Grids

1.3 Criteria

To judge a state is fair or not, we should come up with some criteria for fairness.

1. Weak-d : $d(s) := 100/k - min_j s_j$, the ideal winning rate for each one is 100/k for k players. and $min_j s_j$ means the the lowest win rate of all players.
2. Weak-D : $D(s) := max_j s_j - 100/k$.
3. Strong : $\bar{D}(s) = max(d(s), D(s))$.
4. Semi-complete: $C(s) = max_{ij}|s_i - s_j|$
5. Complete: $\bar{C}(s) = ||1/K, s||_{L1}$, if we see the s is a k-vector, then the ideal vector would be 1/K that every element is 1/k.

1.3.1 Example

If $s = (50, 25, 25)$, so $D = 50 - 33 = 17, d = 33 - 25 = 8, \bar{D} = 17, C = 50 - 25 = 25, \bar{C} = 50 - 33 + 33 - 25 + 33 - 25 = 33$.

1.3.2 Most fair state

We try to find the most fair state s out of the whole state set S according to different criteria, $min_{s \in S} Criteria(s)$. It will find different "most fair state" using each criteria in the S. Because of the **Selfish Hypothesis**, intuitively, we can most likely find a Weak-d fairness.

1.3.3 Transformation

There is a transformation from s to s', $T(s_j) = 100 - s_j - 100 \times (k-2)/k$. For example, a state $s = (40, 20, 20, 20)$ with $d = 5, D = 15$ can be transformed to a virtual state $s' = (10, 30, 30, 30)$ with $d = 15, D = 5$. It convert a problem of finding the maximum value into a problem for minimum value. So if we have a strategy to find a Weak-d Fairness, we can just build a reverse state set RS where the reverse state means the losing rates but not the win rates. So using the strategy in the new set RS can find a Weak-D fairness.

2 Game

2.1 Matrix Game

In player j's turn, he will choose the win rate for himself. The latter player's choice will not affect the winning rate of the previous players' if the previous player has completed his move. The state set of matrix game is infinite and continuous.

2.2 Board Game

Only when every players finish their opening moves, a state can be determined, Fig.2. The state set of board game is finite and discrete.

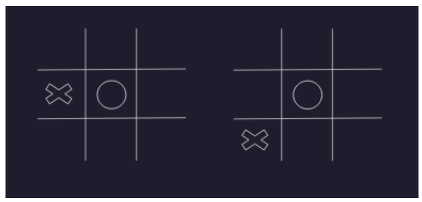

Figure 2: Consider the tic-tac-toc game and we assume player 1 moves in the center. If player 2 moves as left, player 1 has 100% win rate. If player 2 moves as right, the game ends up with a draw. So the win rate of every player can only determined after all players finish their move

3 Swap Rule

The significance of the swap rule is that when the players consider the existence of a swap rule, he will choose the best move as much as possible and ensure that he will not be swapped by others. So if every player is really smart(**Compact Hypothesis**), no one needs to swap with others for their opening move.
Since the goal of swap rule is to get a fair opening state, so:

1. **Move first, Lowest win rate**. In 4 players game, if player 1 can guarantee the second lowest win rate, he will choose a state with (33,0,33,34) rather than a (25,25,25,25)

2. **Degradable**. A k-player swap rule can degenerate into k-1 player swap rule.

These two point can filter the states from S with win rate order from lowest to highest. $f(s) = s \cdot I(s_i \leq s_j | i \leq j)$, I is the indicator function.

3.1 Matrix Game

3.1.1 "In place with previous" swap rule(IPWP)

The swap rule tells that every player can choose to swap with his previous player. If the previous player is swapped, he needs to select his move again and change his move order.

when it is a 2 player game, player 1 can only get $min(p, 100 - p)$, so he will make the p is close to 50.

When it comes to 3 player game, player 1 first choose p as his win rate. If player 2 does not swap, player 2 and 3 will play a 2 player swap game with the whole win rates are $100 - p$. So player 2 can only get $0.5 \cdot (1 - p)$ at most. So if $p > \frac{100}{3}$, player 2 will swap with player 1. Then player 1 and 3 plays a 2-player swap game again. In any case player 1 will get the smallest winning rates, so he will make p is close to 33.

It is a good swap rule to extend to k+1 players because it make sure the first player in k players can only get minimum.

3.2 Board Game

3.2.1 2 player game with swap rule

Let us consider a simple board game as follow(Fig.3).

	a	b	c
a		(90,10)	(70,30)
b	(10,90)		(40,60)
c	(30,70)	(60,40)	

Figure 3: A board game with only 3 grids and 2 players. The column means the player 1's move and the row is that of player 2.

The existing swap rule for 2 player board game is that: Player 1 moves first, Player 2 decides whether to swap. If he swaps, he will become the first player to move with the occupied grid and Player 1 will move another grid.

So if we use this swap rule for the board game above, the search tree would be like Fig.4. We can not find the most fair state out of the whole state set, so the existing swap rule is not enough.

3.2.2 Solution of board game

Let's take a closer look at the search tree, and we can know how each player determines their choice. From the root node to each leaf node, we continuously select subsets from the state set. When the player moves on 'a', all the states

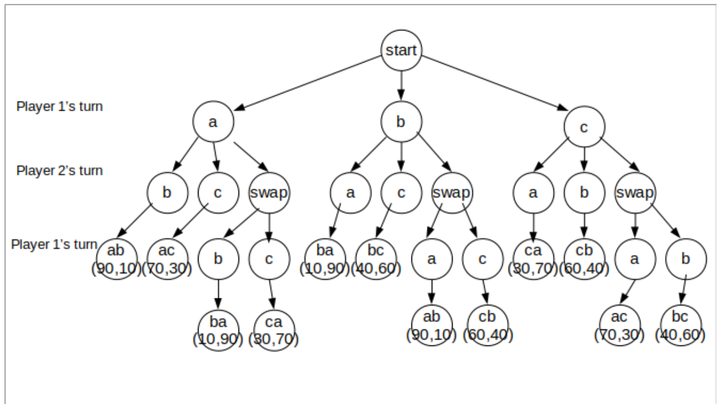

Figure 4: The search tree with swap rule.Intermediate node means the player's choice. In each player's turn, he chooses the maximum of his own win rates from child nodes. The leaf node contains the moves m and state s. The final opening result would be 'ac' with (30,70), but there is another more fair state (40,60)

where the first element is 'a' is filtered from the state set. We call this process as **Projection**. We use the operators $P_j(g)$ to represent the projection that the player j moves on grid g.

In the reverse path, the parent node decides which child node to choose. The only judging criterion is to maximum his own winning rate. We call this process as **Max**. We use the operators $Max_j(g)$ to show that player j moves on grid g and it's his turn to maximum his win rate.

So we can get the objective function without a swap rule like:

$$Max_1(x)Max_2(y)Max_3(z)P_1(x)P_2(y)P_3(z), x \neq y \neq z$$

3.2.3 3 player board game with IPWP

When there is a swap rule, it actually reprocesses the child nodes before the Max process, We just call it as **Swap**. If the swap rule is IPWP, so we reorder all child states with operator $S(i, i+1)$: if $s_{i+1} < s_i$, swap s_{i+1} and s_i. The corresponding objective function would be:

$$Max_1(x)S(1,2)Max_2(y)S(2,3)Max_3(z)P_1(x)P_2(y)P_3(z), x \neq y \neq z$$

3.2.4 In Place with Any swap rule(IPWA)

This swap rule means that every player can choose to swap with any player ahead. The player is swapped needs to select his move again and change his move order.

Just llike the IPWP, we can also use a operator to describe it. $S(i)$: reorder all elements after i-th element of the state s from lower to higher.

So we can see obviously that the two swap rule for 3 player game are equivalent! because they both limit one player to get a lower win rate than later players. In another word, they filter the states out of S with elements' order from lower to higher.

It is asymmetric because there is no constraint on the last player, just like he 2 player one. So it can not guarantee a most fair state.

4 Equal Status Swap Rule(ESSR)

For now, the last player has a clear advantage. The swap rule should also achieve the same status of each player. That is to say we should also make the last player restricted. A simple idea is if the first player has not be swapped, he can decided to swap with the last player in the end. Let us consider the simple board game above again.

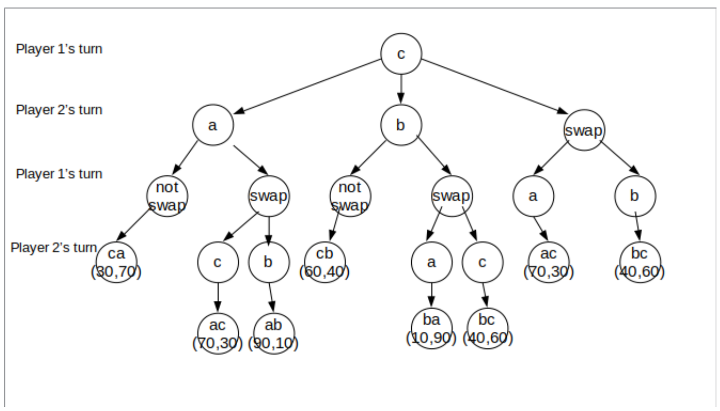

Figure 5: The sub-tree with equal status swap rule. We only show you the situation when player 1 moves on c. The final state will be 'cb':(60,40). The final state in other two cases are 'bc':(40,60) and 'ca':(30,70). So the player 1 would choose the 'cb' in the end, which is the most fair in state set S.

Using the search tree above, we can get a generalized conclusion. If we have 3 grids a,b,c, $(a, b) = (x_1, x_2), (b, a) = (x_2.x_1)$,$(a, c) = (x_3, x_4), (c, a) = (x_4.x_3)$, $(b, c) = (x_5, x_6), (c, b) = (x_6.x_5)$. We assume that $x_1 \leq x_3 \leq x_5 \leq x_2, x_4, x_6$. $x_1 + x_2, x_3 + x_4, x_5 + x_6$ may not equal to 100. If player 1 moves on a, the final result would be (x_5, x_6), b with (x_3, x_4), c with(x_6, x_5). So by ESSR, from all 3 grids and 2 players cases, we can find (x_6, x_5) in the end, which is a $max_s(min_j s_j)$.

From the generalized conclusion, we can easily extend it to n-grid and 2 player case. We can describe the game with a $n \times n$ matrix, with diagonal blank. Then we can select every 3 rows and corresponding columns to reconstruct a 3×3 game. It would find the $max_s(min_j s_j)$ of the 3×3 game. So we can find the $max_s(min_j s_j)$ of the whole state set of $n \times n$ game, which is a **Weak-d Fairness State**.

To extend it into a k-player case. The game becomes a k-dimension matrix, but if we fixed moves of player 2 to player k-1, the projection would be a 2-d matrix again when we only consider the first and the last player. Of course the project matrix do not include the grids that the other players fix.

So using recursion and projection methodologies, we can prove the the ESSP can guarantee a Weak-d Fairness State of the k-player board game.

5 fairness

6 Cut and choose

7 Conclusion

References