# A FRAMEWORK FOR COMPUTER MAHJONG COMPETITIONS

*Yi-Chang Shan[3], Ching-Hsuan Wei[3], Cheng-Hung Lin[3], I-Chen Wu[3], Li-Kai Chuang[3], and Shi-Jie Tang[3]*

Hsinchu, Taiwan

**ABSTRACT**

This contribution specifies a tournament framework for computer Mahjong competitions. A first version is described in Lin, Shan, and Wu (2011). The protocol between the framework and AI programs, is also described in Lin and Wu (2010). (In Chinese) The Appendices are not directly appended to the contribution; they are accessible at http://aigames.nctu.edu.tw/mahjong/protocol.html.

## 1. INTRODUCTION

In many computer game tournaments, most games are played onsite (face-to-face) without the help of computer networks. One of the reasons is to avoid cheating. However, this causes the following three problems.

1. It takes times for players to operate (make moves) on the tables. Thus, the time between the real clocks and the computer clocks are different due to the time consumed by the human operations.
2. Humans operate manually, and do so sometimes wrongly.
3. It lowers the possibility to play many games, in particular, in the area of games with imperfect information and high uncertainty which do not need a large amount of computation time. For example, Chinese Dark Chess, and Mahjong.

The first two problems are not critical, as long as the games in the tournaments are well operated. The tournaments in the Computer Olympiad have demonstrated that the current style of operation works reasonably well. Recently, the ICGA has accepted computer Go to be played in the Computer Olympiad over the KGS web site, since many of the latest computer Go programs need large clusters of computers which cannot be brought on-site.

The third problem becomes serious for the games with imperfect information and high uncertainty. For example, in the Chinese Dark Chess tournaments in the Computer Olympiad (see Yen, Chiu, and Wu, 2010; Chen, Shen, and Hsu, 2010), each pair of players is matched for two games or at most four games. Due to the uncertainty of the game, it is hard to identify from two or four games only which program is the stronger one. This can be seen in the latest two tournaments (see Chen, Shen, and Hsu, 2010; TCGA Computer Game Tournaments, 2011) where the winners were reordered.

Having observed this problem, we believe that the tournaments for Mahjong, a traditional Chinese game, has the same problem as Go. In addition, the rules of Mahjong are rather complicated, so it is hard for a human being to operate a program. For these reasons, the current contribution describes a framework for the Mahjong tournament.

Section 2 introduces Mahjong rules. Section 3 designs a framework for the tournament. Section 4 describes the protocol between the framework and the computer Mahjong programs. Section 5 discusses fairness supporting the Mahjong tournament in the TCGA computer game tournaments (2011) held in June 2011.

## 2. MAHJONG RULES

There are many different variations of Mahjong, with different rules. This document focuses on Mahjong as it is commonly played in Taiwan. The cards of Mahjong are called *tiles*, which are introduced in Subsection 2.1. The rules for a single game are described in Subsection 2.2, the rules for determining how many rounds to play are in Subsection 2.3, and the rules for scoring is in Subsection 2.4.

### 2.1 Tiles

Mahjong is played with a total of 144 tiles or pieces. These are classified into six kinds of suits. According to the definitions of the Mahjong competition rules of World Mahjong Organization (2006), the suits include *Wan*

[3] Department of Computer Science, National Chiao Tung University, Hsinchu, Taiwan. Email: {ycshan, miso, jhl, icwu}@java.csie.nctu.edu.tw

(ten thousand, or *萬* in Chinese), *Tiao* (rope, *條* or *索*), *Tong* (dots, or *筒*), *Wind* (*風*), *Dragon* (*箭*), and *Flower* (*花*).

Wan patterns include 1-Wan to 9-Wan, each of which has 4 tiles as listed in Table 1. The tiles of Tiao and Tong are similar as listed in Table 2 and Table 3, respectively. Wind patterns include East, West, South, and North Winds, each of which has 4 tiles. Dragon patterns include Red, Green, and White Dragons, each of which has 4 tiles shown in Table 4. Flower patterns include 8 tiles, labeled Spring, Summer, Autumn, Winter, Plum, Orchid, Chrysanthemum, and Bamboo, respectively as shown in Table 5. Each pattern of Mahjong consists of four identical tiles, except Flowers. The total number of tiles is 144.

Table 1: The patterns of Wan tiles.

| Wans | 1-Wan | 2-Wan | 3-Wan | 4-Wan | 5-Wan | 6-Wan | 7-Wan | 8-Wan | 9-Wan |
|---|---|---|---|---|---|---|---|---|---|
| Tile |  |  |  |  |  |  |  |  |  |

Table 2: The patterns of Tiao tiles.

| Tiaos | 1- Tiao | 2- Tiao | 3- Tiao | 4- Tiao | 5- Tiao | 6- Tiao | 7- Tiao | 8- Tiao | 9- Tiao |
|---|---|---|---|---|---|---|---|---|---|
| Tile |  |  |  |  |  |  |  |  |  |

Table 3: The patterns of Tong tiles.

| Tongs | 1-Tong | 2-Tong | 3-Tong | 4-Tong | 5-Tong | 6-Tong | 7-Tong | 8-Tong | 9-Tong |
|---|---|---|---|---|---|---|---|---|---|
| Tile |  |  |  |  |  |  |  |  |  |

Table 4: The patterns of Wind and Dragon tiles.

| Patterns | East | West | South | North | Red | Green | White |
|---|---|---|---|---|---|---|---|
| Tile |  |  |  |  |  |  |  |

Table 5: The patterns of Flower tiles.

| Patterns | Spring | Summer | Autumn | Winter |
|---|---|---|---|---|
| Tile |  |  |  |  |
| Patterns | Plum | Orchid | Chrysanthemum | Bamboo |
| Tile |  |  |  |  |

**2.2 A Game**

Mahjong is played by four players, normally sitting around a square table. In a game, all 144 tiles are initially faced down, shuffled randomly by players by hand, and piled up into a wall with width 18 and height 2 in front of each player. Four walls form a square on the table, and therefore they are called *squared walls*.

In a game, one of the four players serves as the *dealer* (sometimes called *banker*), who starts the game. The dealer takes 17 tiles, and the other three players take 16 tiles from the squared walls, according to some specific rule which is not described due to the irrelevancy to our framework. Note that the dealer initially takes 14 tiles and the other three players take 13 tiles in the rules of China and Hong Kong.

The goal of the game is to make a *winning pattern* including five *sets* of tiles and a *pair* of tiles. A set of tiles may include three or four identical tiles, e.g., three or four 2-Wans, or three sequentially-numbered tiles, e.g., a set of 2-Wan, 3-Wan, and 4-Wan. A pair of tiles consists of two identical tiles, e.g., two 2-Wans. For example, Figu shows a winning-tiles pattern of Mahjong.



Figure 1: A winning pattern of Mahjong.

A set of three identical tiles is called a set of *Ker* (*刻*) or simply a Ker; a set of four identical tiles is called a set of *Gong* (*槓*) or simply a Gong; a set of three sequentially-numbered tiles is called a set of *Shun* (*順*) or simply a Shun; and a set of two identical tiles is called a set of *Dui* (*對*) or simply a Pair.

After all four players have taken their tiles as described above, the dealer needs to discard a tile to the *pool*, normally placed in the middle of squared walls. Then, one of the other three players may bid for the discarded tile according to the following sequence.

1. *Hu* (*胡*): If one of the other three players can make a winning pattern by taking the discarded tile, then she[4] wins. This action is called Hu in Chinese. The player who discarded the tile is named *Chucker* in Chinese. If more than one player can make a winning pattern, the first player in the counter-clockwise manner starting from the chucker wins.

2. *Meld Gong (明槓)*: If one of the other three players can make a Gong by taking the discarded tile, she can bid for the tile. This action is called Meld Gong in Chinese. The player must meld (reveal) the Gong on the table.


3. *Pong (碰)*: If one of the other three players can make a Ker by taking the discarded tile, she can bid for the tile. This action is called Pong in Chinese. The player must meld the Ker on the table.


4. *Eat(吃)*: If the next player to the one discarding the tile can make a Shun by taking the discarded tile, she can bid for the tile. This action is called Eat in Chinese. The player must meld the Shun on the table.


If none of the players bid for the discarded tile following the above bidding, the next player to the one discarding the tile can take a new tile from the squared walls. This action is called *Mo* in Chinese, which may have the following two special situations.

- *Self-Mo* (*自摸*): If the player makes a winning pattern by taking this tile, she wins. This action is called Self-Mo in Chinese. The player must meld the winning pattern to the others.
- *Covered Gong* (*暗槓*): If the player can make a Gong by taking this tile, she leaves the Gong face down on the table, and is not allowed to move later. This action is called Covered Gong. After the Covered Gong, the player should take one tile from the squared walls.


In both cases, bidding and Mo, we see that the player takes one more tile, therefore she needs to discard a tile to the pool, unless she wins. Such an action, from bidding or Mo to discarding a tile, is called one *move* and will be repeatedly played, until a player wins or the game is drawn. The game is a draw when the number of tiles left in the squared walls is 16 or less.

Besides, whenever a player takes a Flower tile, the player always melds it on the table and takes one more tile. This implies that a Flower tile does not change the above behavior. The main purpose of Flower tiles is to have a different scoring, as described in Subsection 2.4 (below).

[4] For brevity, we use ‘she’ and ‘her’ whenever ‘she or he’ and ‘her or his’ are meant.

### 2.3 Rounds and Jongs

A game is also called a *hand* (*盤*) in Chinese. As described above, one of the four players serves as the dealer, who starts a game. If the dealer wins the game or nobody wins the game (a drawn game), the dealer will still serve as the dealer for the next game. This is called *consecutive-dealing*. Otherwise, the player next to the dealer in a counter-clockwise manner will serve as the dealer for the next game. When the player who serves as the first dealer serves again after all the others have served as dealers at least once, it is said that one *round* (*圈*) is finished and another round starts.

A **Jong***(將)* consists of four rounds, respectively called East Wind round, South Wind round, West Wind round, and North Wind round. These different round names are used for scoring as described in the next Subsection. Normally, when playing Mahjong, players play at least one Jong.

### 2.4 Scoring

In a Mahjong game, it is quite complicated to score the players. In practice, the amount of money a player wins determines the score. The score can be negative to indicate losing money, and the total scores of the four players should be 0. Before playing Mahjong, players normally need to make an agreement on the values of *base* (*底*) and *Tai* (*台*), denoted by *VB* and *VT*, respectively.

If player A wins by bidding the tile from a chucker B, A wins the score $SA = VB + NT * VT$ from B. Here, *NT* is the number of Tais, and is calculated based on complicated rules, where many factors are involved. These factors include (a) Kers with the Dragons, (b) Kers with the Winds matching the Wind round, (c) the Flower tiles, (d) whether the dealer is the winner or loser, and so on. The highest Tai, 16, are won when the dealer obtains the winning pattern initially from the squared walls. This case is named *Tian-Hu* (*天胡*) in Chinese. The details of calculating Tais can be found in The World Mahjong Organization (2006) or in (ThinkNewIdea Limited, 2005a) (in Chinese), and are omitted in this contribution.

If player A wins by Self-Mo, all the other three players should pay to A. So, normally, players try to get Self-Mo, if they can.

## 3. FRAMEWORK FOR TOURNAMENTS

Since the rules of Mahjong are quite complicated, the framework for Mahjong tournaments is designed by leveraging an existing Mahjong network game system of a web game server, currently supported by ThinkNewIdea Ltd (2005b). The architecture of the system, as shown in Figure 2 includes one Mahjong server and four clients, each for one player.

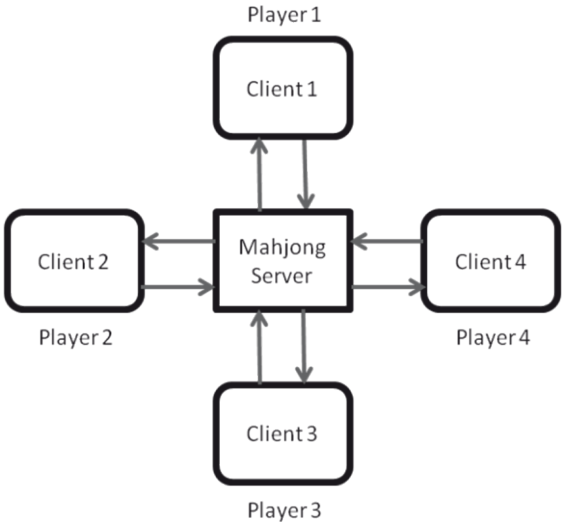


Figure 2: The architecture of Mahjong network game system.

The framework for Mahjong tournaments is designed mainly by modifying clients into wrappers which invoke the corresponding AI programs (see, e.g., Lin, 2008; Lin, 2011; Chuang, Chin & Lin, 2007). Thus, the framework includes one referee server and four wrappers for AI programs, as the gray boxes show in Figure 3.

AI program designers only need to design the protocol between their AI programs and the framework (or the wrapper).

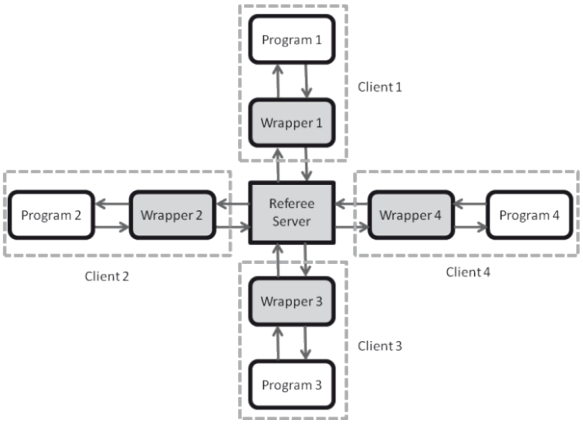


Figure 3: The framework for tournaments.

Initially, the referee of the tournament creates the referee server and sets up a seed which is used to generate all tiles of squared walls at random. Then the referee server waits for four AI players to login. A client logs in to the referee server via the wrapper. The client needs to specify the client ID and the file path of the executable for the AI program. The client ID indicates the location of the Mahjong table as shown in Figure 3. The file path is used to let wrapper invoke the executable. The AI program (the executable) can communicate with the referee server via the wrapper. In order to simplify the communication, the standard input of the AI program stands for the messages from the referee server to the AI program, while the standard output stands for those from the AI program to the referee server.

## 4. COMMUNICATION PROTOCOL

This section describes the communication protocol between the AI program (the executable) and the referee server (via the wrapper). The protocols are classified into eleven scenarios, which are now described.

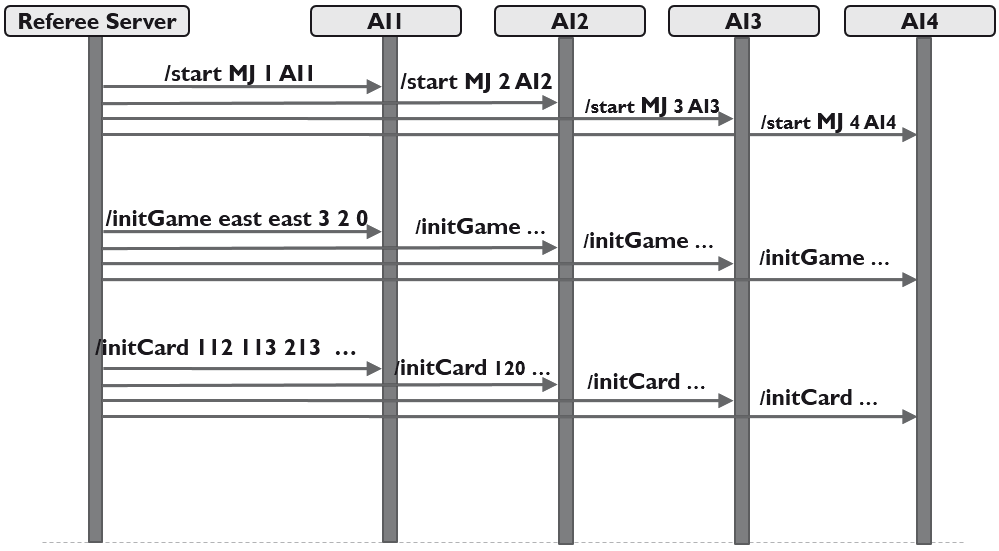


Figure 4: The sequence diagram for the initialization.

The first scenario is for initial setup. In Figure 4, the referee server sends each client the instruction `/start MJ` to indicate the client ID and the file path of the executable of the AI program. The details of the instruction formats are shown in CGI Lab (2014).Then, the server broadcasts the instruction `/initGame` to each client to decide the client who is the dealer to start the game. Finally, the server sends each client the instruction `/initCard` which includes the initial tiles, 17 tiles for the dealer and 16 tiles for others. In our system, each of 144 tiles has one ID, named *TID*, as shown in detail in CGI Lab (2014). For example, for the four 1-Wans, their TIDs are set to 110 to 113, respectively. (See Appendices on the given website.)

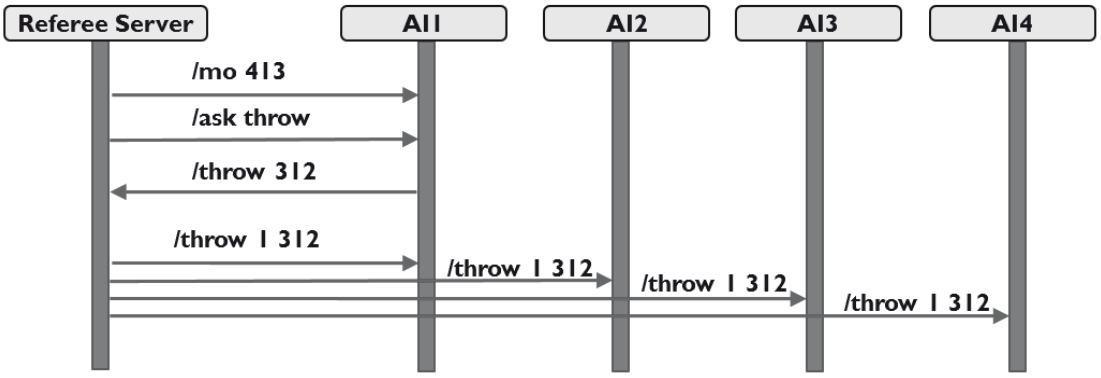


Figure 5: The sequence diagram for Mo.

The second scenario is for Mo. In Figure 5, the referee sever sends Client AI1 the instruction `/mo` that informs AI1 of taking the tile with TID 413 from the squared walls, and then the instruction `/ask throw` to inquire which tile to discard. Then, AI1 sends the instruction `/throw back` to the server to indicate the discarded tile. Finally, the server broadcasts the instruction `/throw to` indicate that AI1 discarded a tile to the pool.

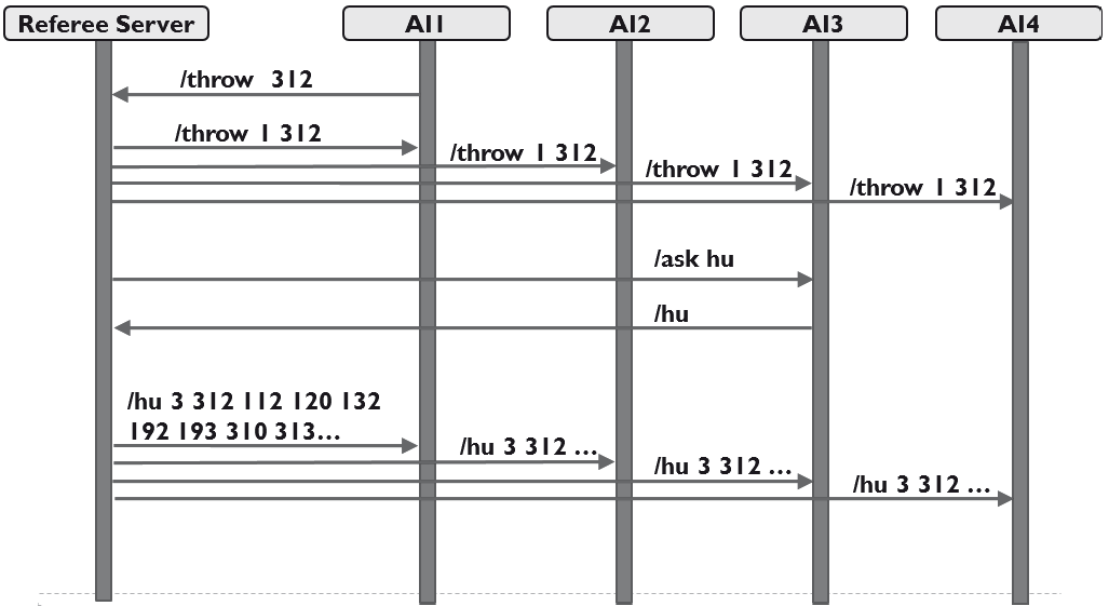


Figure 6: The sequence diagram for Hu.

The third scenario is for Hu, and a sequence diagram is illustrated in Figure 6. Assume that AI1 discarded a tile via `/throw` and that the referee server broadcasts the tile via `/throw`. If AI3 is able to Hu, the server sends AI3 the instruction `/ask hu` to inquire whether to Hu or not.

If AI3 wants to win by Hu, AI3 sends the server `/hu`. Then, the server broadcasts the message as well as AI3's winning pattern. If AI3 wants to ignore Hu, AI3 sends `/pass` to the server.

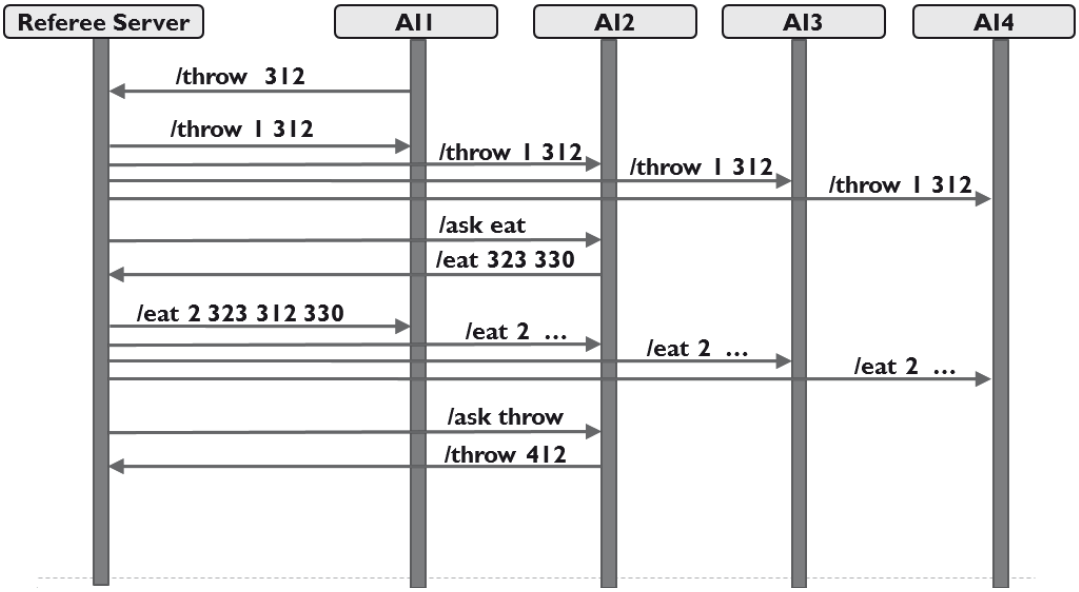


Figure 7: The sequence diagram for Eat.

The fourth scenario is for Eat, and a sequence diagram is illustrated in Figure 7. Similarly, assume that AI1 discarded a tile, say with TID 312 (indicating 1-Tong), and that the referee server broadcasts the tile. If AI2 is able to Eat, the server sends AI2 the instruction `/ask eat` to inquire whether to Eat or not.

If AI2 wants to Eat, AI2 makes a Shun by sending the server the instruction `/eat together` with the tiles 323 (indicating 2-Tong) and 330 (indicating 3-Tong). Then, the server broadcasts the message of Eat with tiles 312, 323 and 330. Finally, the server asks AI2 to discard one tile by the instructions `/ask throw` and `/throw`, as described the second scenario. If AI2 does not want to eat, AI2 sends `/pass` to the server.

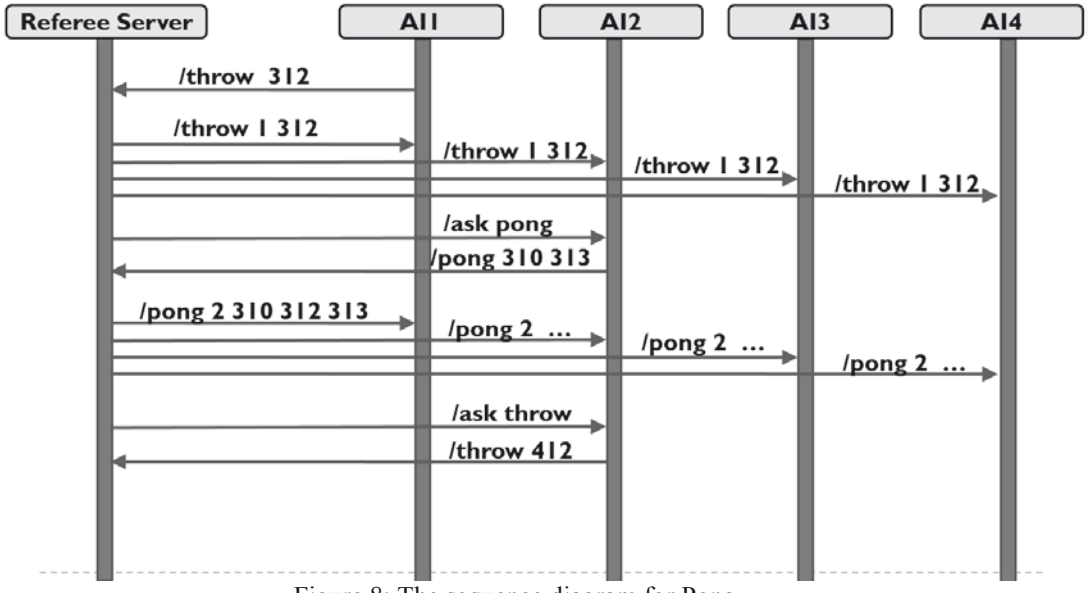


Figure 8: The sequence diagram for Pong.

The fifth scenario is for Pong, and a sequence diagram is illustrated in Figure 8. Similarly, assume that AI1 discarded a tile, say with TID 312 (indicating 1-Tong), and that the referee server broadcasts the tile. If AI2 is able to Pong, the server sends AI2 the instruction `/ask pong` to inquire whether to Pong or not.

If AI2 wants to Pong, AI2 makes a Ker by sending the server the instruction `/pong together` with the tiles 310 (indicating 1-Tong) and 313 (indicating 1-Tong). Then, the server broadcasts the Pong message with tiles 310, 312 and 313. Finally, the server asks AI2 to discard one tile by the instructions `/ask throw` and `/throw`, as described in the second scenario. If AI2 does not want to Pong, AI2 sends `/pass` the server.

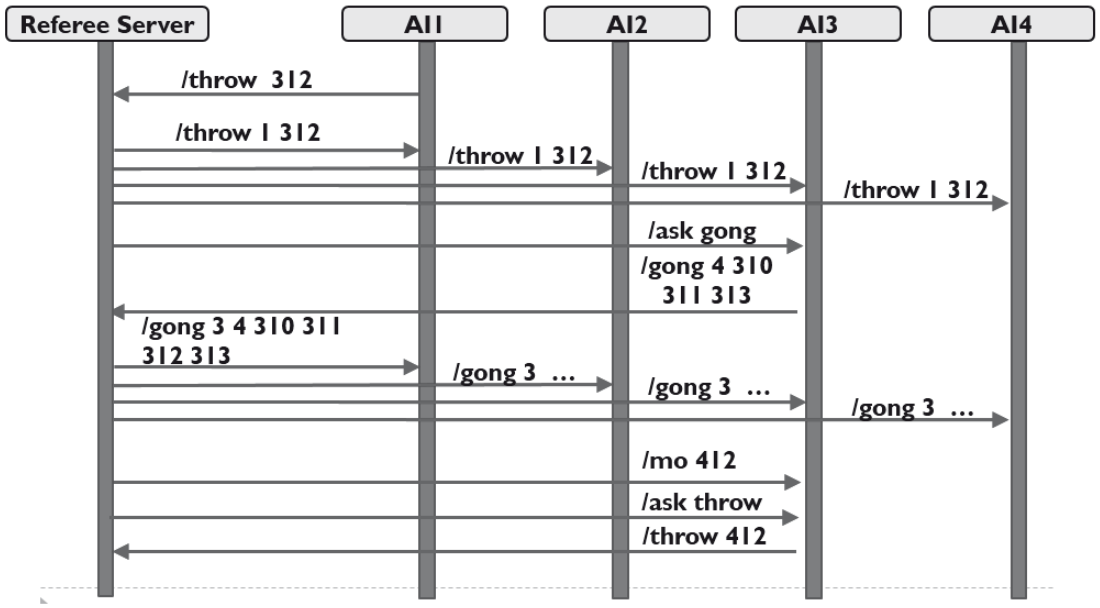


Figure 9: The sequence diagram for Meld Gong.

The sixth scenario is for Meld Gong, and a sequence diagram is illustrated in Figure 9. Similarly, assume that AI1 discarded a tile, say with TID 312 (indicating 1-Tong), and that the referee server broadcasts the tile. If AI3 is able to Meld Gong, the server sends AI3 the instruction `/ask gong` to inquire whether to Meld Gong or not.

If AI3 wants to Meld Gong, AI3 makes a Gong by sending the server the instruction `/gong 4` together with the tiles 310 (indicating 1-Tong), 312 (indicating 1-Tong) and 313 (indicating 1-Tong). Then, the server broadcasts the message of Gong with tiles 310,311, 312 and 313. Finally, the server asks AI3 to Mo (as described in the second scenario), and to discard one tile by the instructions `/ask throw` and `/throw`, as described in the second scenario. If AI3 does not want to Meld Gong, AI3 sends `/pass` to the server.

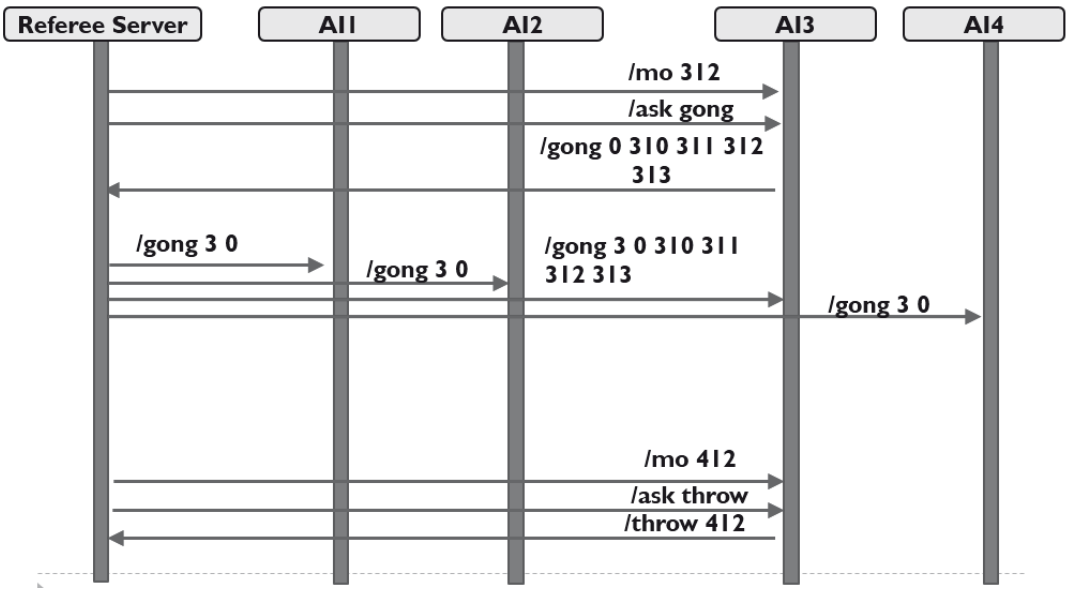


Figure10: The sequence diagram for Covered Gong.

The seventh scenario is for Covered Gong, and a sequence diagram is illustrated in Figure 10. Similarly, assume that AI3 receives a tile from Mo, say with TID 312 (indicating 1-Tong). If AI3 is able to make a Covered Gong, the server sends AI3 the instruction `/ask gong` to inquire whether to make a Covered Gong or not.

If AI3 wants to make a Covered Gong, AI3 sends the server the instruction /gong 0 together with the tiles 310, 311, 312 and 313, indicating four 1-Tongs. Then, the server sends AI3 the message of Covered Gong with tiles 310, 311, 312 and 313, and sends the message to other players of AI3 making a Covered Gong without tiles information. Finally, the server asks AI3 to Mo (as described in the second scenario), and discard one tile by the instructions /ask throw and /throw, as described in the second scenario. If AI3 does not want to Covered Gong, AI3 sends /pass to the server.

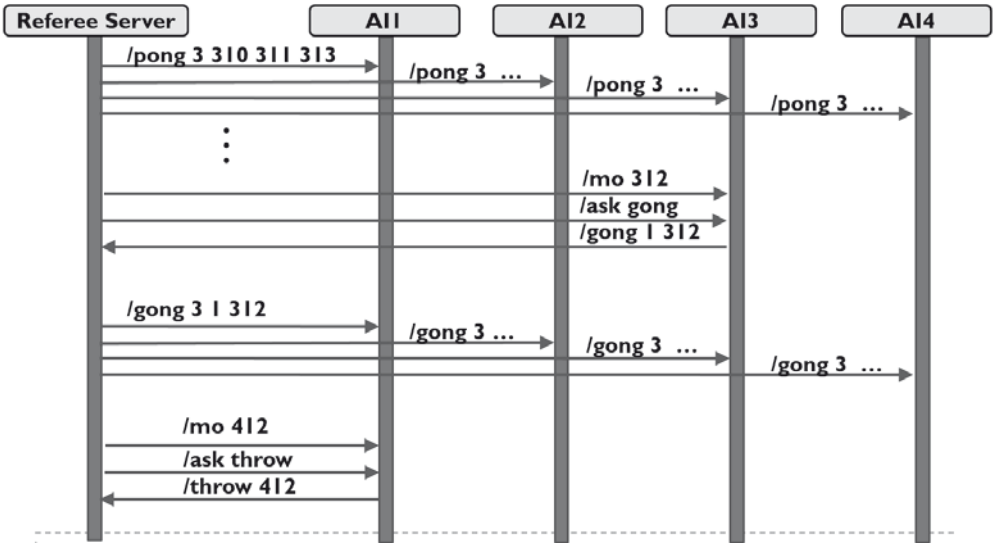


Figure11: The sequence diagram for Promote-to-Gong.

The eighth scenario is for Promote-to-Gong, and a sequence diagram is illustrated in Figure 11. Similarly, assume that AI3 makes a Ker of 1-Tong with TID 310, 311 and 313. After a period of time, AI3 receives a tile with TID 312 from a Mo. Then the referee server sends the instruction /ask gong to inquire whether to make a Promote-to-Gong or not.

If AI3 wants to make a Promote-to-Gong, AI3 upgrades a Ker to Gong by sending the server the instruction /gong 1 together with the tiles 312 (indicating 1-Tong). Then, the server broadcasts the message Promote-to-Gong with tiles 312. Finally, the server asks AI3 to Mo (as described in the second scenario), and discard one tile by the instructions /ask throw and /throw, as described the second scenario. If AI3 does not want to make a Promote-to-Gong, AI3 sends /pass to the server.

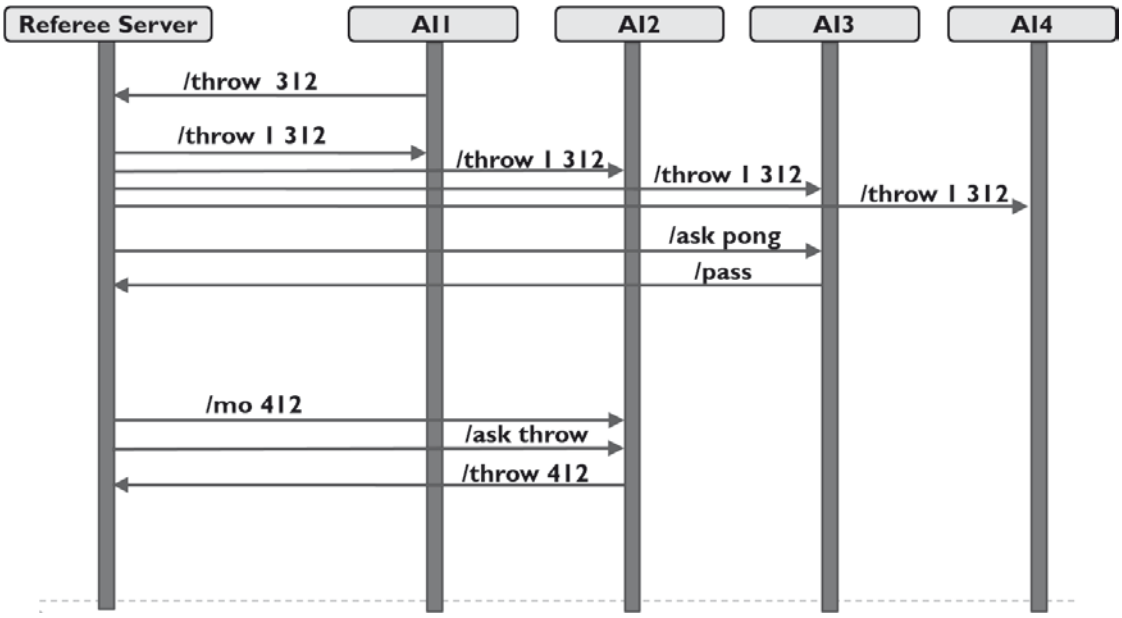


Figure12: The sequence diagram for Pass.

The ninth scenario is for Pass, and a sequence diagram is illustrated in Figure 12. Similarly, assume that AI1 discards a tile, say with TID 312 (indicating 1-Tong), and that the referee server broadcasts the tile. If AI3 is able to Pong, the server sends AI3 the instruction `/ask pong` to inquire whether to Pong or not.

In this case, AI3 does not want to Pong, so AI3 sends `/pass` to the server. Finally, the server asks AI2 to make a Mo (as described in the second scenario), and to discard one tile by the instructions `/ask throw` and `/throw`, as described in the second scenario.

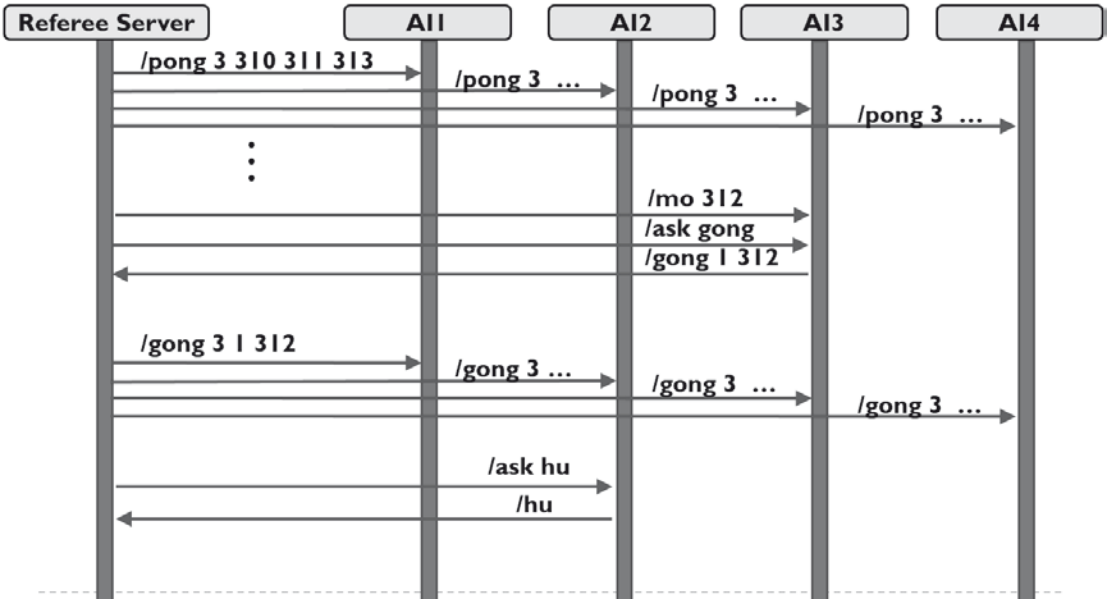


Figure 13: The sequence diagram for Hu-from-Intercept-Promote-to-Gong.

The tenth scenario is for Hu-from-Intercept-Promote-to-Gong, and a sequence diagram is illustrated in Figure 13. When a player is making a Promote-to-Gong for a tile, say with TID 312 (indicating 1-Tong), another player, say AI3, can intercept the tile, if AI3 can make a winning pattern after taking the tile.

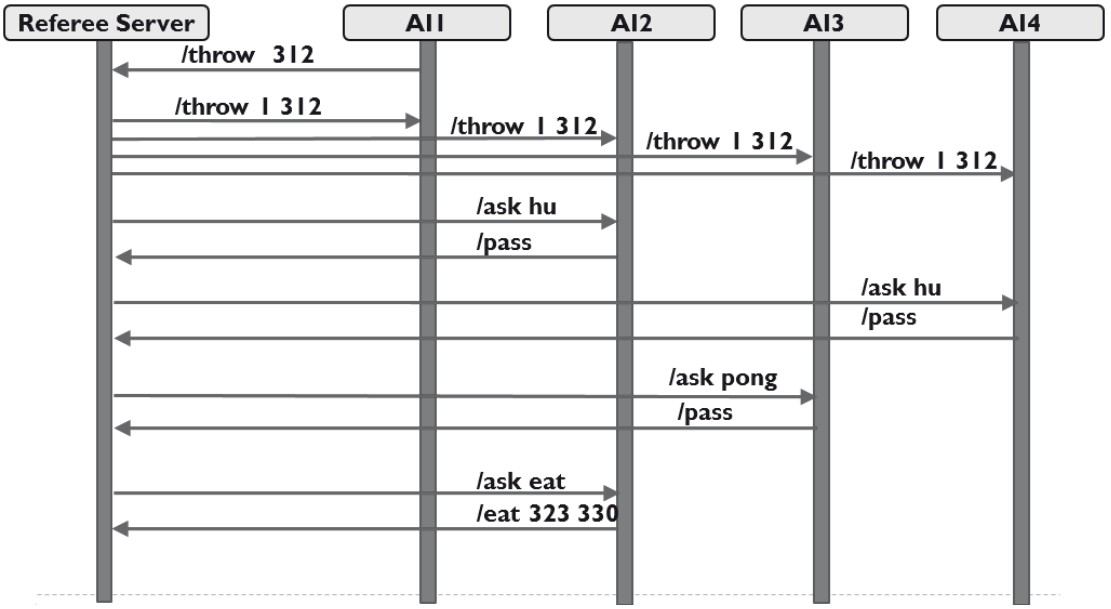


Figure 14: The sequence diagram for simultaneously bidding.

The eleventh scenario is for simultaneously bidding, and a sequence diagram is illustrated in Figure 14. Let AI1 send discarded a tile 312, and the server broadcasts the tile. Now, assume AI2 to Hu, AI4 to Hu, AI3 to Pong, and AI2 to Eat, respectively. Since AI2 is the first player (in the counter-clockwise manner starting from AI1)

to win, the server asks AI2 to Hu first. If AI2 says no, then the server asks AI4 to Hu. If AI4 says also no, the server asks AI3 to Pong. If AI3 says also no, then the system asks AI2 to Eat. At last, AI2 decides to Eat. Note that AI2 and AI4 refrain to win since they may want to win by Self-Mo.

All the protocol formats are summarized in both Table 6 and 7 (for details, see Lin & Wu, 2011). Table 6 lists the instructions from the referee server to clients, while Table 7 lists those from clients to the server.

Table 6:The instruction formats from the referee server to clients.

| ID | Instruction | Instruction Format |
|---|---|---|
| 1 | Start game | /start MJ <position> <AIPath> |
| 2 | Init game | /initGame <wind> <game> <open> <dealer> <continue> |
| 3 | Deal tiles | /initCard <card1> <card2> … <card16> |
| | | /initCard <card1> <card2> … <card16> <card17> |
| 4 | Mo | /mo <card> |
| 5 | Ask to discard a tile | /ask throw |
| 6 | Ask Pong | /ask pong |
| 7 | Ask Gong | /ask gong |
| 8 | Ask Eat | /ask eat |
| 9 | Ask Hu | /ask hu |
| 10 | Discard a tile | /throw <postion> <card> |
| 11 | Eat | /eat <postion> <card1> <card2> <card 3> |
| 12 | Pong | /pong <postion> <card1> <card2> <card3> |
| 13 | Meld Gong | /gong <postion> 4 <card1> <card2> <card 3> <card4> |
| 14 | Covered Gong | /gong <postion> 0 |
| | | /gong <postion> 0 <card1> <card2> <card 3> <card4> |
| 15 | Promote-to-Gong | /gong <postion> 1 <card> |
| 16 | Hu | /hu <postion> <hucard> <card1> <card2> … |
| 17 | Exit game | /exit <score1> <score2> <score3> <score4> |

Table 7: The instruction formats from clients to the referee server.

| ID | Instruction | Instruction Format |
|---|---|---|
| 1 | Throw | /throw <card> |
| 2 | Eat | /eat <card1> <card2> |
| 3 | Pong | /pong <card1> <card2> |
| 4 | Meld Gong | /gong 4 <card1> <card2> <card3> |
| 5 | Covered Gong | /gong 0 <card1> <card2> <card3> <card4> |
| 6 | Bu Gong | /gong 1 <card> |
| 7 | Hu | /hu |
| 8 | Pass | /pass |

Finally, we need to pay attention to the issue of fault tolerance. We do so as follows.

- If a client does not discard a tile within the time limitation, the framework will select the tile to discard at random. So, each client should change its internal state according to the message from the framework, not based on the message from itself.
- If the server receives any wrong instructions or formats, the framework gives a penalty by setting the client to the failure mode. In the failure mode, the client is forced to discard the tile which it gets from the squared walls, and is not allowed to Eat, Pong, Gong, and Hu.

## 5. DISCUSSION

This framework was used in the Mahjong tournament of the 17th Computer Olympiad (Tseng et al., 2013). In this tournament, there were four participants, named LONGCAT, GRANDSLAM, THOUSANDWIND, and MAJO, developed by three teams from National Chiao Tung University (NCTU), National Dong Hwa University (NDHU), and National Taiwan Normal University (NTNU), respectively. Note that the last two programs were developed at NTNU.

In order to make the game more balanced without affecting the uncertainty much, three rules were modified in order to make the tournaments more fair to all participants. The three suggested changes are as follows.

1. For fairness, play as many games as possible. In the TCGA (2011) tournaments, 12 Jongs were played.
2. For four players, A, B, C, and D, let each permutation be arranged in two Jongs, since the total number of different permutations around the table is six, {A, B, C, D}, {A, B, D, C}, {A, C, B, D}, {A, C, D, B}, {A, D, B, C} and {A, D, C, B}.
3. Let the dealers of the four games in each round have the same tiles in the squared walls by rotating all tiles in the squared walls.
   For example, for the permutation, {A, B, C, D}, when player A serves as the dealer, the tiles in front of {A, B, C, D} are the same as those in front of {B, C, D, A}, when B serves the dealer. Under that situation, we need to remove the rule of consecutive-dealing.

Below we wish to elaborate on the third change of rules. The reason of this rule is to avoid a serious uncertain case, as follows. Consider a problem: if a dealer gets Tian-Hu as described in Section 4, then the player wins a very large number of Tais and wins by a very large score. Thus, it is likely that the player wins the tournament due to the luck. The argument is somewhat similar to the game Bridge (American Contract Bridge League, 2008) where a pair of teams plays twice with different team members and with the same but exchanged cards.

A potential problem for the above is cheating. Each AI program is able to log the previous game and then many possibly guess the initial tiles in the earlier games in the same round. However, this issue is insignificant in a tournament with the protocol described in two aspects. First, the framework does not expose all tiles in the squared wall to all players. It is hard to guess the cards in the opponents, though it is possible. Second, we explicitly request that all participants not cheat in this way. We believe this might be an acceptable solution analogous to that we also trust that all participants in the Internet Go tournaments do not cheat by having a human player play, instead.

In the tournament, some settings are listed as follows. The values, *VB* and *VT*, are set to 1000 and 500, respectively. The time limit for each move is three seconds. So, it took about six hours to complete the whole tournaments for the 12 Jongs. Finally, LONGCAT won the champion of this tournament. In the future, we will modify the framework so that computer Mahjong programs are able to play against human players.

**ACKNOWLEDGEMENT**

The authors would like to thank ThinkNewIdea Ltd (群想網路科技) for supporting the Mahjong network game system of a web game server (including the design of some protocols), and thank the National Science Council of the Republic of China (Taiwan) for financial support of this research under contract numbers NSC 99-2622-E-009-010-CC3 and NSC 100-2622-E-009-011-CC3.

**References**

American Contract Bridge League (2008). *Laws of Duplicate Bridge* (North American Edition), Effective September 8, 2008.

CGI Lab (2014). The Protocol of the Framework for Computer Mahjong Competitions. Retrieved from http://www.aigames.nctu.edu.tw/mahjong/protocol.html.

B. N. Chen, B. J. Shen, and T. S. Hsu (2010). Chinese Dark Chess, *ICGA Journal*, Vol. 33, No.2, pp. 93-106.

K. M. Chuang, Y. J. Chen, S. S. Lin (2007). On the Study of Algorithm and Relevant Issues for Computer Mahjong, (in Chinese). *The 12th Conference on Artificial Intelligence and Applications TAAI 2007*.

TCGA Computer Game Tournaments (2011). http://tcga.ndhu.edu.tw/tcga2011/

The World Mahjong Organization (2006). *Mahjoan Competition rules*, Chinese Publishing Company 2006. Retrieved from http://www.mindmahjong.com/info/eshowinfo.asp?id=447.

C. H. Lin (2011). *The Study of Artificial Intelligence Program for Mahjong*, Master's thesis, NCTU, Taiwan.

C. H. Lin and I-C. Wu (2011). *The Communication Protocol between Mahjong Framework and Artificial Intelligence Programs*, (in Chinese), NCTU, Taiwan, unpublished.

C. H. Lin, Y.-C. Shan, and I-C. Wu (2011). Tournament Framework for Computer Mahjong Competitions, *The 2011 International Conference on Technologies and Applications of Artificial Intelligence* (TAAI), Chungli, Taiwan.

T. Y. Lin (2008). *The Study of Mahjong Artificial Intelligence* (2008). Master's thesis, NCTU, Taiwan.

ThinkNewIdea Limited (2005a). Mahjong Rules in CYC Game web site (in Chinese). Retrieved from http://cyc7.cycgame.com/cyc/cgi-bin/manual.php?i=manG&game=MJ16#anchor06.

ThinkNewIdea Limited (2005b). CYC Game (in Chinese). http://www.cycgame.com.

## NONOGRAM TOURNAMENT IN OLYMPIAD YOKOHAMA 2013

*Lung-Pin Chen[5], Der-Johng Sun,[6] and Wen-Jie Tseng[7]*

A Nonogram solver tournament was held as part of the 17th Computer Olympiad, which took place in Keio University, Yokohama, Japan, from Aug 12th to 18th, 2013. Table 1 lists the participants and the final standings.

| Rank | Program Name | Author(s) | Affiliation(s) | #Problem(s) Solved |
|---|---|---|---|---|
| 1 | LALAFROGKK | Kan-Yueh Chen, Ching-Hua Kuo, Hao-Hua Kang, and Der-Johng Sun | National Chiao Tung University (NCTU), Taiwan | 1000 |
| 2 | THUNONO | Lung-Pin Chen, Lin Ku Wei | Tunghai Univ, Taiwan | 297 |
| 3 | THEUNINITIATED | Kuo-Chen Huang | National Taichung University of Education, Taiwan | 1 |

**Table 1:** The programs and scores of the Nonogram solver tournament in ICGA 2013.

### Nonograms

Nonograms are logic grid puzzles in which cells in a grid must be painted according to the given *row clues* and *column clues*. Players are requested to paint each grid cell into either "white" or "black", such that the segments constituting of consecutive black cells in each row (or column) matches the corresponding row clue (or column clue). For example, Figure 1 shows a Nonogram puzzle and a feasible painting that matches all the clues. In this puzzle, the second row clue "3 3" indicates that row two contains two segments of size three that are separated by at least one white cell.

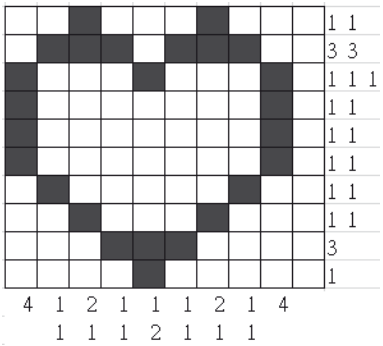


**Figure 1**: A Nonogram puzzle and a feasible painting.

[5] Dept. of Computer Science, Tunghai Univ., Taichung, Taiwan. Email: lbchen@thu.edu.tw.
[6] Dept. of Computer Science, National Chiao Tung Univ., Hsinchu, Taiwan. Email: derjohng.cs95g@nctu.edu.tw and icwu@csie.nctu.edu.tw.
[7] Dept. of Computer Science, National Chiao Tung Univ., Hsinchu, Taiwan. Email: wjtseng@csie.nctu.edu.tw.