How to Make the Perfect Fireworks Display: Two Strategies for *Hanabi*

CHRISTOPHER COX
Carnegie Mellon University
Pittsburgh, PA 15213
cocox@andrew.cmu.edu

JESSICA DE SILVA
University of Nebraska-Lincoln
Lincoln, NE 68508
jessica.desilva@huskers.unl.edu

PHILIP DEORSEY
Emory & Henry College
Emory, VA 24327
pdeorsey@ehc.edu

FRANKLIN H. J. KENTER
Rice University
Houston, TX 77005
franklin.h.kenter@rice.edu

TROY RETTER
Emory University
Atlanta, GA 30322
tretter@emory.edu

JOSH TOBIN
University of California, San Diego
La Jolla, CA 92093
rjtobin@ucsd.edu

Hanabi is a card game by Antoine Bruza which won the prestigious *Spiel des Jarhes* (Game of the Year) in 2013 [**10**]. Named after the Japanese word for fireworks, the game is based upon creating the perfect fireworks display by playing cards, i.e. fireworks, in a desirable sequence. Unlike conventional card games, players see cards in other players' hands but not their own and work together as a team. Hence, gameplay is focused on the players discovering information about their own cards through the limited communication the game allows. Only through clever strategy, coordinated implementation, and a bit of luck can the team successfully create the perfect fireworks display.

In order to devise a strategy for *Hanabi* in which players communicate information effectively, we turn to *hat guessing games* for inspiration. Hat guessing games are a popular topic in recreational mathematics [**3, 4, 5, 7, 8, 11**]. Consider the following version of a hat guessing game. Five people each put on either a red or blue hat at random and stand so that each person can see the color of every other person's hat but

Math. Mag. **88** (2015) 323–336. doi:10.4169/math.mag.88.5.323.
MSC: Primary 00A08, Secondary 97A20.

not their own. If the people guess the color of their own hats sequentially out loud, how can they maximize the expected number of correct guesses? By guessing randomly, on average only 2.5 people will guess correctly. On the other hand, by implementing a clever strategy, the players can guarantee that 4 players will always guess correctly! We will discuss such a strategy and its application to *Hanabi* in later sections.

Hat guessing games are problems that arise in coding theory. Coding theory has many real world applications in communication theory [**9**], and it has also been used to study other games such as Sudoku [**1**]. Our approach to *Hanabi* is similar to the more modern topic of *network coding*, where multiple agents with asymmetric information try to communicate effectively. For example, suppose that two people wish to communicate simultaneously via a satellite. After the satellite receives the two messages A and B, which are numbers, the satellite can then broadcast $A + B$ once instead of broadcasting the two messages in sequence. Each person can then use the message she sent along with the broadcasted message to deduce the the message sent to her, i.e., $B = (A + B) - A$. As a result, the satellite becomes more efficient by broadcasting half as many messages!

This article presents two strategies for *Hanabi* that incorporate ideas from hat guessing games and network coding. These strategies work similarly to the satellite example above, where a player who is giving hints acts as the satellite and the contents of the other players' hands act as messages. In turn, each hint is more effective and hence, more fireworks are made with fewer hints.

In the first strategy, hints are used to recommend actions to players. In the second strategy, hints are used to tell the players information about their cards. Results from computer simulations demonstrate that both strategies perform well, and that the more advanced information strategy achieves a perfect score over 75 percent of the time. In comparison, this is only slightly worse than a scenario in which the players cheat by looking at their own hands and play by a simple heuristic.

This article begins with an overview of the rules of *Hanabi* followed by a discussion of the hat guessing game ideas incorporated into both of our strategies. We then describe our two strategies and conclude by discussing results of computer simulations.

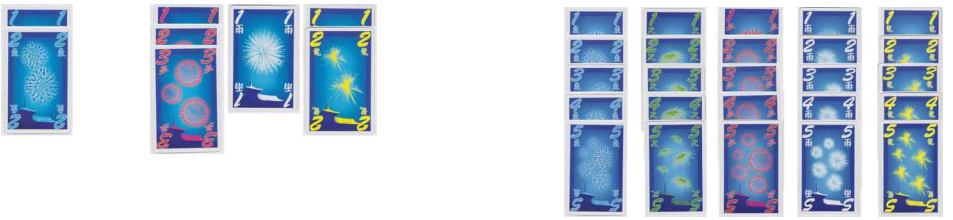
Overview of the rules of *Hanabi*

Here we give a brief overview of the rules of *Hanabi*. We focus on the original variation of the game with five players although most of the concepts throughout can be adapted to other variations.

The game *Hanabi* is played with a special deck of cards, each card representing a specific firework. Each card has a *rank*, which is a number 1 through 5, and a *suit*, which is one of five colors. The deck consists of 50 cards with 10 cards in each of the five suits. Within each suit, there are three '1's, two '2's, two '3's, two '4's, and one '5'.

Each player begins with a hand of four cards which are held so that all other players can see them but she cannot. The team also begins with eight *hint tokens*. Starting with the youngest player, players take turns in the clockwise direction performing one of three actions: play a card, discard a card, or give a hint.

To play a card, a player selects a card in her hand, declares she is playing the card, and then reveals the card. The cards that have already been played are placed in piles based on suit, with the most recent, i.e., highest rank, card on top as demonstrated in Figure 1a. The new card will be successfully played if its rank is exactly one higher than the last successfully played card of the same suit and a '1' is successful if no cards of that suit have been previously played. For example, in Figure 1a the following cards



(a) An imperfect fireworks display.

(b) The perfect fireworks display.

Figure 1 Examples of different fireworks displays.

can be successfully played at the current game state: blue 3, green 1, red 4, white 2, and yellow 3. If the card is successfully played, it is then added to the appropriate pile and the fireworks display becomes one firework brighter! A perfect score corresponds to a fireworks display of 25 cards as seen in Figure 1b. An unsuccessfully played card is removed from the game and the team makes an *error*; that is, the team tried to launch a firework at the wrong time. In either the event of a successful or unsuccessful play, the player draws a card if the deck is nonempty.

To discard a card, the player selects a card in her hand, declares she is discarding the card, and then reveals the card. The card is removed from the game and the team is awarded a hint token, provided the team has fewer than eight hint tokens. She then draws a card, provided the deck is not empty.

To give a hint, one player selects another player and identifies all cards in the other player's hand of a particular rank or suit, e.g., "These two cards are blue." Giving a hint costs the team one hint token, so if no hint tokens remain a hint cannot be given. Also, whenever a player gives a hint, the recipient must have at least one card of the chosen rank or suit, e.g., "You have no yellow cards in your hand" is not a legal hint.

There are three ways in which the game may end. If the team makes a third error, the game ends with a score of 0. If the the team successfully plays 25 cards, the game ends with perfect score of 25 points. Otherwise, once the deck becomes empty, each player makes one final turn and the game ends with a score equal to the number of fireworks successfully launched.

Hat guessing

We now discuss a strategy for the hat guessing game described in the introduction. We will then generalize this strategy for two colors to an eight color version. The strategy for the eight color hat guessing game will be implemented in both of our strategies for *Hanabi*.

Two color hat guessing game Each of five players will be assigned either a red hat or blue hat at random. Each player will be able to see the color of every other player's hat but will be unable to see the color of her own hat. In some order, for instance from youngest to oldest, each player will then be asked to guess the color of the hat on her own head. The players will be able to hear the guesses made by the previous players, but no other communication is allowed. The objective of the game is for the players to devise a strategy before the hats are assigned that maximizes the expected number of correct guesses they will make as a team.

Since the youngest player has no information about her own hat and has not heard any of the other players' guesses, she can only guess her hat correctly, on average, half of the time. It follows that the expected value of the game can be no greater than 4.5/5 correct guesses.

There is, perhaps surprisingly, a strategy in which the expected number of correct guesses is 4.5/5. Indeed, suppose that the first player guesses 'blue' if the number of blue hats on the heads of the other four players is odd and guesses 'red' otherwise. As noted before, this first guess will be correct, on average, half of the time. Once this first guess has been made, every other player can now deduce the color of her own hat! For example, suppose that the second player observes exactly one blue hat on the heads of players three, four, and five. She can now reason that if her hat were blue, the first player would have responded 'red' since that indicates there are two blue hats on players two, three, four, and five. By similar reasoning, if her hat were red, the first player would have guessed 'blue.' Similarly, players three, four, and five can also deduce the color of the hat they are wearing based upon the first player's guess.

We now describe a generalized version of this strategy for five players and eight hat colors which we will incorporate into our strategies for *Hanabi*.

Multiple color hat guessing game The following notation will aid in our description of this eight color version of the above hat guessing game. Label the players $P_1, P_2, \ldots, P_5$ and suppose that there are eight different "colors": $0, 1, \ldots, 7$. Let c_i be the color of the hat placed on the head of player P_i. The following generalization of the previous strategy can guarantee at least four of the players will guess correctly.

Player P_1 will guess, or rather respond, with color

$$r_1 := \sum_{i \neq 1} c_i \pmod 8,$$

which is computed by finding the sum of the hat colors of every player who is not the first player and then determining the remainder when divided by 8. Hence, player P_1 is not actually guessing at her own hat color, nor is she guaranteed to respond correctly. However, her response will guarantee that all other players can respond correctly.

For $i > 1$, player P_i will respond

$$r_i := r_1 - \sum_{j \neq 1, i} c_j \pmod 8,$$

where the sum $\sum_{j \neq 1,i} c_j$ adds together the colors of the hats worn by all players other than players P_1 and P_i. Since

$$r_i \equiv \sum_{j \neq 1} c_j - \sum_{j \neq 1, i} c_j \equiv c_i \pmod 8,$$

every player other than the first player is guaranteed to have responded correctly. We remark that this strategy generalizes to any number of colors and any number of players.

We also note that this type of scheme is similar to "check digits" on bar codes such as UPC and ISBN where the final digit of the code serves to verify the parity of the other digits (see, for example, [**12**]). In the above hat guessing game, each player is missing a different piece of information, and the initial response serves as a "check digit" which allows the other players to deduce the information they are missing.

For more information about this sequential hat guessing game, in addition to other variations, see [**3, 4, 8, 11**]. While sequential hat guessing games are well understood, many interesting and open problems remain when players guess the colors of their hats simultaneously. See [**5, 7, 11**] for more information on simultaneous hat guessing.

Overview of applying hat guessing to *Hanabi* The main idea of applying hat guessing to *Hanabi* is that each player's hand is a "hat" and the contents correspond to a

"color." While there are more than eight possible hands, we assign each possible hand a "color" 0 through 7. As a result, when a player gives a hint, all other players can determine the colors of their hands, thereby deducing information about its contents.

The two strategies presented assign colors to hands differently. In the first, the color represents which card should be played or discarded. That is, loosely speaking, the colors recommend a particular move to each other player. For the second, more complicated strategy, the colors correspond to a set of possible rank and suit values for a particular card. Hence, each hint narrows down the possibilities for the identity of a particular card.

Strategy 1: The recommendation strategy

In the recommendation strategy, players recommend actions to each other by encoding the recommendations as hints. For example, if player P_1 gives a hint to player P_2 telling her which of her cards are red, *all* of the other players will interpret this as a custom recommended action. For instance, a player may decode this hint to determine that she should discard a certain card in her hand while another player will learn that one of her cards should be played. By implementing a hat guessing scheme, a single hint communicates custom recommendations to each of the other players.

For this strategy, the cards in each player's hand are indexed from left to right (as seen by the other players), C_1, C_2, C_3, C_4. Each time a player plays or discards a card, the indices of cards with higher index will shift their indices down by 1, and the new card drawn will be indexed as C_4.

The key to the recommendation strategy is the following encoding scheme which assigns a number 0 through 7 to each player's hand. The possible recommendations and their corresponding numbers are as follows:

0. Play card C_1
1. Play card C_2
2. Play card C_3
3. Play card C_4
4. Discard card C_1
5. Discard card C_2
6. Discard card C_3
7. Discard card C_4

Giving recommendations Before we describe how to determine which recommendation to give to each player, we define three types of cards:

- **Playable**: a card that can be successfully played with the current game state.
- **Dead**: a card that has the same rank and suit of a successfully played card.
- **Indispensable**: a card for which all other identical copies have been removed from the game, i.e., a card that if removed from the game will imply a perfect score cannot be obtained.

In Figure 2, player P_2's card C_1, the white 2, can be successfully played which deems player P_2's card C_1 as *playable*. Player P_4's card C_1, the green 1, has already been successfully played so player P_4's card C_1 is a *dead card*. Player P_5's card C_4, the red 5, is *indispensable* since 5's of each color are unique in the deck. The cards which are playable, dead, and indispensable in a player's hand will determine the recommendation given to her as described below.

The recommendation for a hand will be determined with the following priority:

1. Recommend that the playable card of rank 5 with lowest index be played.

2. Recommend that the playable card with lowest rank be played. If there is a tie for lowest rank, recommend the one with lowest index.
3. Recommend that the dead card with lowest index be discarded.
4. Recommend that the card with highest rank that is not indispensable be discarded. If there is a tie, recommend the one with lowest index.
5. Recommend that C_1 be discarded.

With this, each player's hand is assigned a number 0 through 7. Viewing the value of the recommendation for each player's hand as the "color of her hat," we see that the player giving a hint would like to tell every other player the color of her hat. Since every other player knows the color of every other player's hat, we can think of this as a multiple color hat guessing game. As discussed in the hat guessing section, if the player giving the hint can communicate a number 0 through 7 to the other players, she can simultaneously tell every other player the color of her hat and thus their custom recommendation. This is possible using the following encoding scheme.

Let position j denote the jth position in the clockwise direction from the player giving the hint.

0. Rank hint to the player in position 1
1. Rank hint to the player in position 2
2. Rank hint to the player in position 3
3. Rank hint to the player in position 4
4. Suit hint to the player in position 1
5. Suit hint to the player in position 2
6. Suit hint to the player in position 3
7. Suit hint to the player in position 4

By choosing an appropriate rank or suit, it is indeed always possible to give any of these hints.

Thus when a player gives a hint, she tells every other player the current recommendation for their hands. For example, in Figure 2, player P_1 needs to communicate that the sum of the "colors" of the other players' "hats." She looks at the other players' hands which have values 2, 0, 4, and 0. Since the sum is congruent to 6 modulo 8, she will give a suit hint to player P_4. From this hint, every other player can determine the color of her hat and know what action has been recommended. For example, player P_2 can deduce the value of her hand is $2 \equiv 6 - (0 + 4 + 0) \pmod 8$.

It is important to keep in mind that as actions take place after a hint is given, the recommendation made to a player may no longer be appropriate. Consider the situation shown in Figure 3; on her turn, player P_4 will recommend to *both* players P_5 and P_2 to play their blue 3 cards. After which, player P_5 will then play her blue 3 card. In a following turn, player P_2 will also play her blue 3, resulting in an error. Although the player giving the hint could have realized that this conflict was going to occur, our method only allows her to communicate the current state of each hand since this is the only information known to all players. As it is more important for players to communicate in a consistent reliable manner, we must accept these duplicate recommendations as a possibility.

To use the recommendations given to them in a way which reduces the number of errors that will be made, the following algorithm is used:

Action algorithm The action a player will take will be decided in the following order of priority.

1. If the most recent recommendation was to play a card and no card has been played since the last hint, play the recommended card.

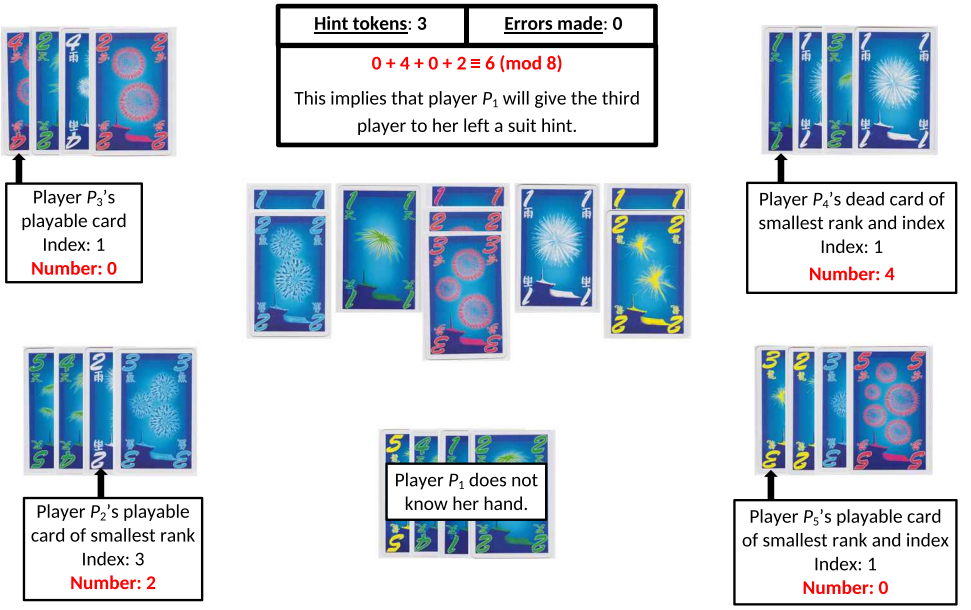


Figure 2 Recommendation strategy: An example of a hint using the recommendation strategy. The updated game state based on this hint and the action algorithm can be found in Figure 3.

2. If the most recent recommendation was to play a card, one card has been played since the hint was given, and the players have made fewer than two errors, play the recommended card.
3. If the players have a hint token, give a hint.
4. If the most recent recommendation was to discard a card, discard the requested card.
5. Discard card C_1.

To see the how the hints and action algorithm fit together, consider Figure 2. Player P_1 gives a suit hint to player P_4. Recall that player P_2 can decode the hint as $6 - 0 - 4 - 0 \equiv 2 \pmod 8$. As a result, she knows it is recommended to play her card C_3.

Similarly, players P_3 and P_5 decode the hint as 0 and are recommended to play C_1, and player P_4 decodes the hint as 4 receiving the recommendation to discard C_1. The action algorithm indicates that the next move is for player P_2 to play her card C_3, which is successful. Since the team has less than two errors, the action algorithm indicates that player P_3 will play her card C_1, which is also successful. Now, P_4 was recommended to discard C_1, however, the team has three hint tokens left so the action algorithm tells player P_4 to give a hint. This new hint will give each of the players P_1, P_2, P_3, and P_5 a new recommended action. The updated game state can be seen in Figure 3.

In summary, the recommendation strategy uses hints to tell other players what actions to take. We believe players can implement this strategy with just a little practice, so give it a try at your next game night! Its performance is discussed in the simulation section.

Strategy 2: The information strategy

In the information strategy, hints give players information about the ranks and suits of their cards. Players then decide how to play based upon this knowledge. Once again,

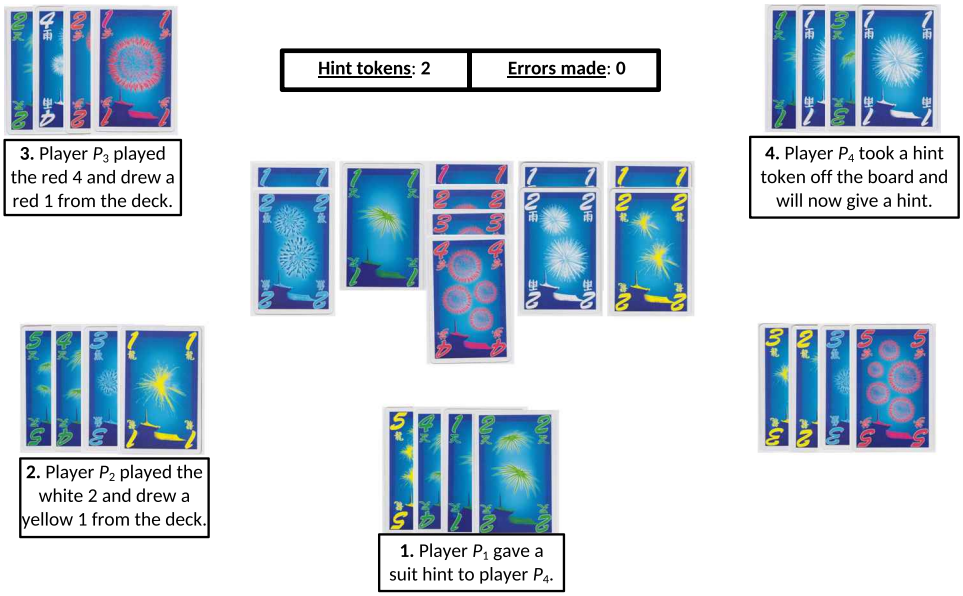


Figure 3 Recommendation strategy: Updated game state three turns after player P_1 gave a hint in Figure 2. It is the beginning of player P_4's turn after players P_2 and P_3 played the cards based on the recommendations of player P_1's hint.

a hat guessing scheme will be used so that the player giving the hint can communicate information to all the other players simultaneously. In what follows, we give a brief description of the concepts used in the strategy. The precise implementation of the strategy can be read in the simulation code available online [**6**].

As with the recommendation strategy, each player's hand will be assigned a value 0 through 7 and the same encoding scheme will be used as in the recommendation strategy. However, unlike the recommendation strategy, the value assigned to the hand of player P_i will not only be a function of the cards in P_i's hand and the cards played previously, but will also be based upon what P_i has already been able to deduce about the cards in her hand. An important aspect to this deduction will be what we refer to as *public* and *private information*.

Public and private information We refer to two types of information: public and private. Public information is information that all players know; that is, every player knows the same public information. Private information is information that a player can deduce based upon seeing the other players' hands.

For example, in Figure 3, player P_1 has a yellow 5. All of the other players know that they do not have a yellow 5, since there is a unique yellow 5 in the game. However, player P_1 does not know that the other players can deduce that they do not have a yellow 5. Hence, the other players' knowledge that they do not have a yellow 5 is private information. On the other hand, if a hint is given to player P_1 that allows her to deduce that she has a yellow 5, then this knowledge becomes public information.

Another important part of the information strategy is what we call the possibility table, which is based solely on public information.

Possibility table Consider a card in *Hanabi* whose rank and suit are unknown. If no information can be deduced, this card may take on one of five possible ranks and five possible suits. We visualize these possibilities as a 5×5 table that we call the

possibility table. As public information is revealed or deduced about the card, some of these possibilities are eliminated. The table evolves with new information by placing an N in each entry corresponding to a rank and suit combination that is no longer possible and a P in each entry corresponding to rank and suit combination that is possible. Figure 4 depicts what two possibility tables might look like at a certain point in the game.

	1	2	3	4	5
Blue	N	P	P	P	N
Green	N	N	N	N	N
Red	N	N	N	N	N
White	N	N	N	N	N
Yellow	N	N	N	N	N

(a) Possibility table for C_2.

	1	2	3	4	5
Blue	N	P	P	P	N
Green	P	P	P	P	P
Red	P	P	N	P	P
White	P	N	P	P	P
Yellow	N	N	N	N	N

(b) Possibility table for C_3.

Figure 4 Possibility tables

Targeting a card When a player gives a hint, that hint will consider exactly one card in every other player's hand. We say the hint *targets* those cards. However, the exact card targeted by each hint may vary from hint to hint. Hence, it is important for all players to know which card is targeted for each hand.

To achieve this, we estimate the probability that each card is playable based upon the public information. To this end, suppose that the card C_i can take on t_i total different values; that is, there are t_i P's in the possibility table for C_i, and a_i of the possibilities for C_i are immediately playable. The probability that card C_i is playable can be estimated by a_i/t_i. However, since there are not the same number of cards of each type in the deck, we use a slightly more complicated scheme to better estimate the probability that the card is playable.

Our more complicated scheme utilizes the publicly available knowledge in the following way. For a card C_i specified by suit and rank, let T_i be the set of elements in the possibility table for C_i with entry 'P' and let S_i be the set of elements in T_i that correspond to playable cards. Now for $c \in T_i$, let m_C be the number of copies of C that have not been fully determined based on publicly available knowledge. The probability that C_i is immediately playable can be estimated by

$$\frac{\sum_{C \in T_i \cap S_i} m_C}{\sum_{C \in T_i} m_C}.$$

The card within a player's hand with the highest such probability is targeted, with the exception that a card with only one P in the possibility table will never be targeted. In the event of a tie, the lowest indexed card is targeted. We remark that in Figure 7 we use the estimate of a_i/t_i for simplicity.

Partition of the possibilities When a player gives a hint, each other player will receive information about her targeted card thereby eliminating some of the P's in the card's possibility table. To establish the meaning of each hint, the set of possibilities for each targeted card is partitioned into 8 sets, and this partition is determined by the public information.

Consider the possibility table for C_3 in Figure 4. There are 16 possible values for C_3 and each hint is one of eight different values, the numbers 0 through 7. As an example,

the players might agree that if the hint is 0, this means that the true value of C_3 is one of the first two possible values in this table, ordered from the top left; that is, either a blue 2 or a blue 3. Similarly, if the hint is a 1, this means that the true value of the card is either a blue 4 or a green 1, and so on. Then when this player receives a hint about C_3, she is able to eliminate all but two possibilities from the possibility table for C_3. For example, if the player receives hint 7, then she knows that the true value of C_3 is one of the two last P's in the possibility table. See Figure 5 for an illustration of this.

Note that there many ways to partition the possibility table. While it may seem advantageous to partition it as evenly as above, we will, in fact, partition it differently as we discuss later.

	1	2	3	4	5
Blue		0	0	1	
Green	1	2	2	3	3
Red	4	4		5	5
White	6		6	7	7
Yellow					

(a) The meaning of each hint value for C_3.

	1	2	3	4	5
Blue	N	N	N	N	N
Green	N	N	N	N	N
Red	N	N	N	N	N
White	N	N	N	P	P
Yellow	N	N	N	N	N

(b) Possibility table after receiving the hint 7.

Figure 5 Partition of the possibility table into hint sets for the nearly equal case. Our information strategy uses a slightly different partition shown in Figure 6.

The different hint values partition the possibilities for a card. In the above example, we partitioned the 16 possibilities into eight sets of size two. We will refer to the sets in this partition as *hint sets*. Given any possibility table, there are many different partitions into hint sets. In the C_3 example, as seen in Figure 5, we gave one possible partition for the possibility table of C_3. Our choice of partition scheme for the information strategy is a little more complex but follows three principles:

1. All possibilities that correspond to dead cards are grouped together in a single hint set. This is because if the card is dead its rank and suit are unimportant.
2. We want many hint sets to contain a single element, which we call *singleton hint sets*. The virtue of this is that whenever a hint specifies a hint set with a single element, the hint recipient learns the exact rank and suit of that card in a single hint. Consequently, we make as many singleton hint sets as possible.
3. Two hints about the same card should always completely determine both the suit and rank of that card. This is accomplished by ensuring that there are no more than eight elements in any hint set. Then after a single hint has been received about a card, there must be fewer than eight possibilities left, at which point a second hint will completely determine the card. We exclude the hint set comprised of the dead cards, which is allowed to have more than eight elements.

As an example, consider again C_3 whose possibility table is given in Figure 4b, and assume that the only dead cards are the 1's of each suit. Then the first hint set consists of the green, red, and white 1's. This leaves 13 possibilities. We can make six singleton hint sets, leaving the seven remaining possibilities for our final hint set. This partition is illustrated in Figure 6.

Value of a hand The *value* of a player's hand, i.e., the hint 0 through 7 that she will be given, is the number assigned to the targeted card by the partition of the possibility table. Note that the possibility table was constructed from public information, so each

	1	2	3	4	5
Blue		1	4	7	
Green	0	2	5	7	7
Red	0	3		7	7
White	0		6	7	7
Yellow					

Figure 6 Partition of the possibility table into hint sets according to the information strategy.

player can determine the value of every other player's hand. Moreover, every player can determine which card in their hand will be targeted and can construct the possibility table for this card. From this information, each player can deduce the partition their card falls into based upon the hint given.

Action algorithm A player will act using her private information with the following priority:

1. Play the playable card with lowest index.
2. If there are less than 5 cards in the discard pile, discard the dead card with lowest index.
3. If there are hint tokens available, give a hint.
4. Discard the dead card with lowest index.
5. If a card in the player's hand is the same as another card in any player's hand, i.e., it is a duplicate, discard that card.
6. Discard the dispensable card with lowest index.
7. Discard card C_1.

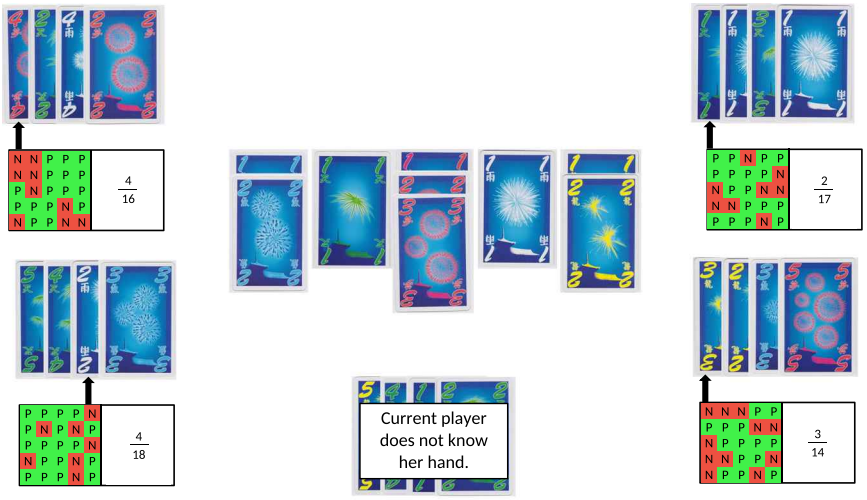
Figure 7 illustrates an example of a player giving a hint using the information strategy with the probability estimate of a_i/t_i. The information strategy is not easily implemented in practice, however a computer implementation is discussed in the following section.

Simulations and their interpretation In this section, the results of simulating the recommendation and information strategies are presented. We also simulate a *cheating strategy* for the purpose of comparison. In this cheating strategy, each player cheats by looking at the cards in their hand and follows the action algorithm presented in the information strategy. The results are presented in Figure 8. The recommendation strategy averages 23.00 points out of 25 and the information strategy averages 24.68 points. By comparison, the perfect information cheating strategy averages 24.87.

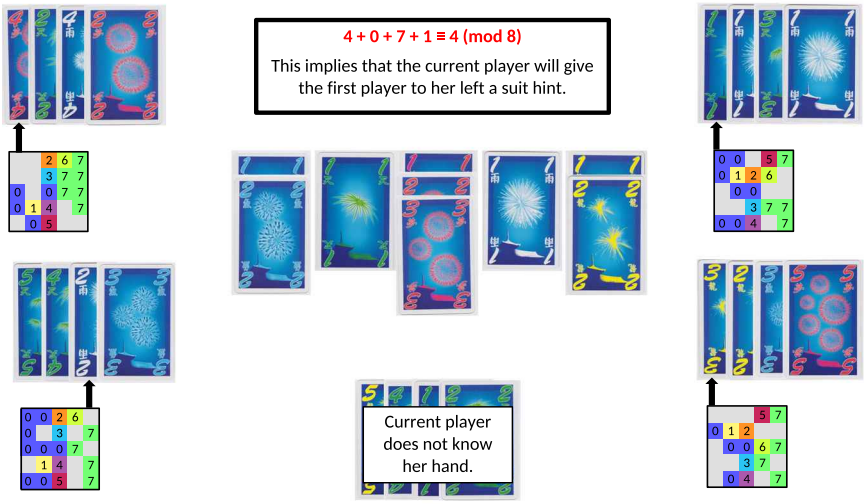
We also remark that in simulation the recommendation strategy frequently makes two errors, but will never make a third. The information strategy will never make any errors.

The simulations were written in C++, and the documented code is available online [**6**]. The code was designed to be modular, separating the game mechanics from the implementation of player strategies. We would like to encourage any interested readers to improve our strategies or implement their own. We made an effort to make our code accessible and be a versatile foundation upon which any strategy could be implemented.

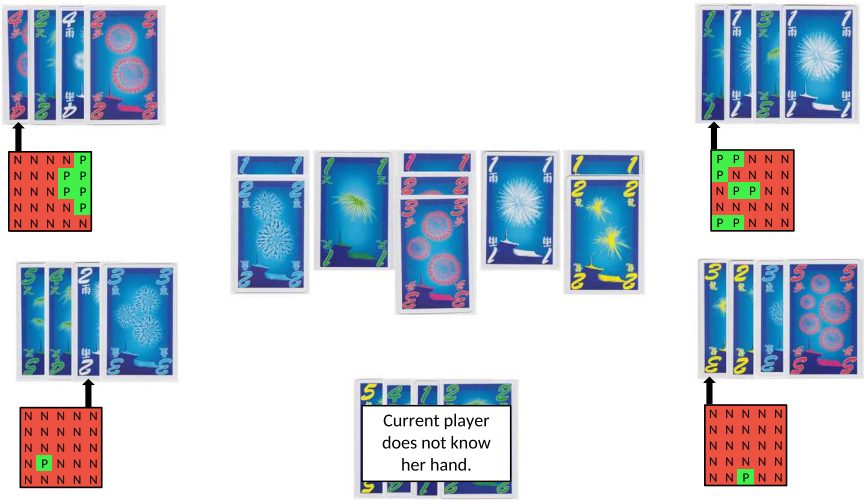
We recognize that our strategies are not optimal. In particular, we see some improvements that could be made but they come at the expense of increased complexity and



(a) The possibility tables and ratio of playable to possible cards for each targeted card.



(b) The corresponding partitions for each targeted card.



(c) Updated possibility tables for each targeted card.

Figure 7 An example of a hint using the information strategy.

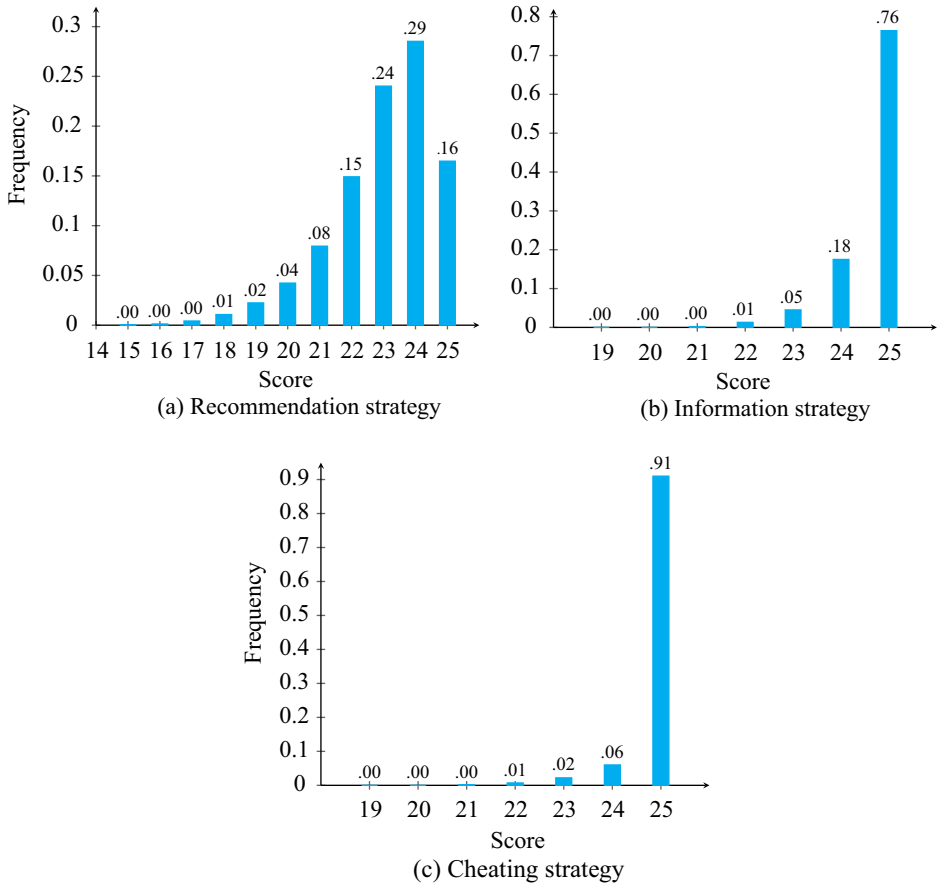


Figure 8 Histograms of the scores after simulating each strategy 10^6 times.

appear to offer only small gains. Although any improvement would be of interest, we would be particularly interested in a strategy that performs significantly better or a simpler strategy with similar performance. We find it important to mention that no strategy can achieve a perfect score every time, as there are permutations of the deck for which a perfect score is impossible. One such permutation occurs when all fifteen 1's are on the bottom of the deck. Thus, there is some upper bound on the average score, which is less than 25. It would be interesting to know a good estimate on the expected value of the game. In particular, we wonder if it is possible for a legal strategy to outperform the average score of 24.87 achieved by our cheating strategy.

Although some of our techniques generalize to other variants, including fewer players, many do not. As such, we leave it to the readers to come up with other strategies for *Hanabi* when playing with fewer than five players. In general, there is still much that could be done regarding the mathematics of *Hanabi*. We hope that fans of *Hanabi* try to implement our strategies and come up with some of their own. Now go out and create your own perfect fireworks display!

Acknowledgment This work was done as part of the inaugural *Rocky Mountain–Great Plains Graduate Research Workshop in Combinatorics* at the University of Denver and the University of Colorado Denver in August 2014. This workshop was additionally co-sponsored by Iowa State University, the University of Nebraska-Lincoln, and the University of Wyoming. Grants supporting this research include National Science Foundation grants CMMI-1300477, CMMI-1404864, DGE-1041000, DMS-1102086, DMS-1301698, and DMS-1427526. We would also like to thank the referees and the editor for their helpful comments.

REFERENCES

1. R. A. Bailey, P. J. Cameron, R. Connelly, Sudoku, gerechte designs, resolutions, affine space, spreads, reguli, and Hamming codes, *Amer. Math. Monthly* **115** no. 5 (2008) 383–404.
2. Board Game Geek, How good is your score really? The statistics (2013), http://boardgamegeek.com/thread/1005947/how-good-your-score-really-statistics.
3. E. Brown, J. Tantan, A dozen hat problems, *Math Horizons* **16** no. 4 (2009) 22–25.
4. J. Bushi, *Optimal Strategies for Hat Games*, Master Thesis, Portland State University, Portland, OR, 2012.
5. S. Butler, M. T. Hajiaghayi, R. D. Kleinberg, T. Leighton, Hat guessing games, *SIAM Rev.* **51** no. 2 (2009) 399–413.
6. C. Cox, J. de Silva, P. DeOrsey, F. Kenter, T. Retter, J. R. Tobin, Hanabi Simulation (2015), https://github.com/rjtobin/HanSim.
7. U. Feige, *You can leave your hat on (if you guess its color)*. Technical Report MCS04-03, Computer Science and Applied Mathematics, The Weizmann Institute of Science, 2004.
8. J. Havil, *Impossible?: Surprising Solutions to Counterintuitive Conundrums*. Princeton Univ. Press, Princeton, NJ, 2011, 50–59.
9. S. Robinson, Why mathematicians now care about their hat color, *New York Times*, April 2001.
10. S. des Jahres, Spiel des Jares 2013: Hanabi (2013), http://www.spiel-des-jahres.com/.
11. P. Winkler, Games people don't play, *Puzzlers Tribute: A Feast for the Mind* (2002) 301–313.
12. J. Kirtland, *Identification Numbers and Check Digit Schemes*. Mathematical Association of America, Washington, DC, 2001.

Summary. The game of *Hanabi* is a multiplayer cooperative card game that has many similarities to a mathematical "hat guessing game." In *Hanabi*, a player does not see the cards in her own hand and must rely on the actions of the other players to determine information about her cards. This article presents two strategies for *Hanabi*. These strategies use different encoding schemes, based on ideas from network coding, to efficiently relay information. The first strategy allows players to effectively recommend moves for other players, and the second strategy allows players to determine the contents of their hands. Results from computer simulations demonstrate that both strategies perform well. In particular, the second strategy achieves a perfect score more than 75 percent of the time.

CHRISTOPHER COX (MR Author ID: 1124471) is currently a graduate student at Carnegie Mellon University and enjoys working on problems in extremal and probabilistic combinatorics. If he is not busy being confused by math, then he is probably not working hard enough!

JESSICA DE SILVA (MR Author ID: 1030014) is a graduate student at the University of Nebraska-Lincoln under the supervision of Jamie Radcliffe. If she is not teaching College Algebra at 8AM, she is working out at the university's recreation center. The calories she burns off at the gym are replenished on the weekends when she tries out new restaurants around Lincoln.

PHILIP DEORSEY (MR Author ID: 1124472) received his Ph.D. from the University of Colorado Denver in May 2015 under the direction of William Cherowitzo. He is a new faculty member at Emory & Henry college in Emory, VA. When he is not busy doing math you can usually find him playing games, or hiking on the beautiful trails in southwestern Virginia.

FRANKLIN H. J. KENTER (MR Author ID: 1058998) received his Ph.D. from UC San Diego in 2013 under the supervision of Fan Chung and is currently a Pfeiffer Postdoctoral Instructor in the Computational and Applied Mathematics Department at Rice University. When he is not thinking about spectral graph theory, he is busy devising crazy strategies for games of all sorts.

TROY RETTER (MR Author ID: 1081998) will complete his Ph.D. in spring 2015 under the supervision of Vojtech Rödl at Emory University. Troy is interested in an eclectic mix of discrete mathematics, especially the insights offered by probabilistic combinatorics.

JOSH TOBIN (MR Author ID: 1124473) is a graduate student at UC San Diego, and will be Franklin Kenter's academic brother after graduation. When he is not working on his thesis, he is trying to make his computer write his thesis for him.