Reduction From Hypergraphic Connect Game to Generalized Geography

Yiyang Zhou

September 2023

Abstract

In this paper, we consider a generalized version of the (m, n, k)-Games, or **Connect**(m, n, k), called **Hypergraphic-Connect**, where the game board is represented by a hypergraph, in which the win conditions are encoded as the hyperedges. The decision problem asks whether the first player is always winning, losing, or tying. We demonstrate a reduction from the decision problem to a modified version of **Generalized Geography** with a special node that allows the tying of a game.

1 Introduction

Connect(m, n, k) is a classic board game where two players put their pieces (usually black and white) into a $m \times n$ grid in turn, so that the player who connects k of their pieces horizontally, vertically, or diagonally first wins. Popular specializations of **Connect** include **Tic-Tac-Toe** and **Go-Moku**, which are **Connect**$(3, 3, 3)$ and **Connect**$(19, 19, 5)$ respectively. Other interesting variations, like **Connect**(m, n, k, p, q), where each player places p pieces per turn except that first player places q pieces in their first turn, have also been studied[1]. Here, we consider an even generalized version of the **Connect** game, to which we tentatively refer as **Hypergraphic-Connect** (**HC**), where the board is no longer restricted to a rectangular shape, and the winning condition may be any combination of the "squares" on the board. Furthermore, the turn sequence is allowed to be arbitrarily defined

instead of a simple alternation between the two players[1]. We wil demonstrate a reduction from **HC** to the classic **Generalized Geography** game (**GG**). Normally, the decision problem of two-player games asks whether the first player (P_1) has a winning strategy. However, it is not uncommon to formulate the question by asking not just whether the P_1 can win, but also whether they can win, lose, or tie exactly[2].

It is worth noting that the implications of the reduction have little theoretical significance, as all it shows is that **Hypergraphical-Connect** $\in$ PSPACE, which should pose no challenge to prove by devising an explicit brute-force algorithm. The reason why this paper exist is very simple: the proof looks cool.

2 Definition of the problems

Definition 1 (Hypergraphic-Connect, **HC**)**.**
Input: $(H = (V, E), A, B)$, where H is a *hypergraph* with vertices V and *hyperedges* E, and A (B) is the turn numbers when P_1 (P_2) is moving. The players color one vertice per turn, in either black (for P_1) or white (for P_2). The goal is to completely cover an (hyper-)edge using the players' own color.
Question: Is P_1 always winning, losing, or tying the game?

Example 1. The standard **Tic-Tac-Toe** can be expressed as an **HC** instance as in Fig. 1.

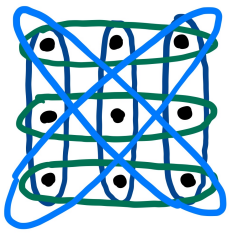

Figure 1: Tic-Tac-Toe represented by 9 vertices and 8 hyperedges.

[1]That being said, examples and diagrams in the paper will still assume a simple alternation of turns to avoid over-generalization and causing confusion.

[2]Readers familiar with Game Theory may suggest that the second player (P_2) could never have a winning strategy in **Connect** games because of the *strategy stealing* argument; However, thanks to the freedom in the definition of the turn sequence, the argument may not always work (for example, **Connect**(m, n, k, p, q) where $p > q$).

Example 2. Figure 2 is an alternative **Tic-Tac-Toe** board designed by me long ago (after child me finally got bored of my classmates capturing the center on their first move with no variation in their tactics whatsoever) that contains "bridges" and banned squares. It seems like P_1 always wins after some play testing.

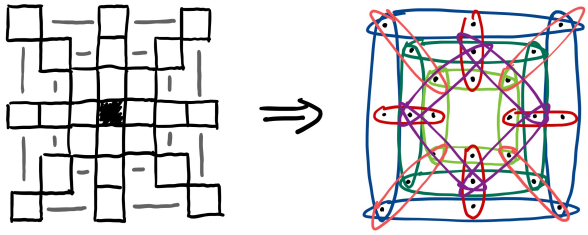

Figure 2: Alternative Tic-Tac-Toe board.

Geography games does not have the possibility for the players to tie a game without modification. To remedy this, we artificially create a special node called *tie node* such that whenever either player enters this node, the game ends with a tie. Note that it is trivial to see the equality between the alternative definition where multiple *tie nodes* are allowed.

Definition 2 (Special-Generalized-Geography , **GG**)**.**
Input: $(G = (N, A), s, t)$, where G is a directed graph with node-set N and arc-set A. s is the starting node, and t is the *tie node*. On the first move P_1 chooses an out-neighbour of s, then P_2 does the same w.r.t. the node chosen by P_1, so on and so forth. The goal is to force the other player into a dead end that they have nowhere to go, or to have either player enter the *tie node* if there is no winning strategy.
Question: Is P_1 always winning, losing, or tying the game?

Example 3. This is a simple instance of our special generalized geography game, where P_1 has a winning strategy:

Example 4. In this instance, P_2 does not have a winning strategy, but they can force a tie. I.e. P_1 is always tying.

Next, we will define concepts and terminologies that will be referred to frequently in our reduction.

Definition 3 (Piece)**.** We agree to use the word *piece(s)* to refer to the tokens used by the players while playing a **Connect** game. WLOG, pieces

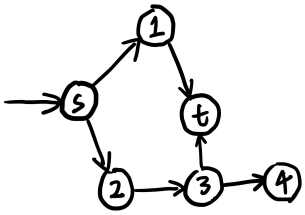

Figure 3: Winning strategy for P_1: $(s \rightarrow)2 \rightarrow 3 \rightarrow 4$

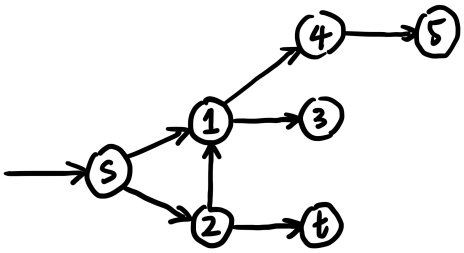

Figure 4: Forced route that leads to a tie: $(s \rightarrow)2 \rightarrow t$

used by the first player are black, and those used by their opponent are white.

Definition 4 (Square)**.** A *square* is a vertex in the hypergraph of a **HC** game. A square is *occupied* if it contains a piece.

Definition 5. (Win condition) A *win condition* refers to the set of squares covered by a particular hyper-edge in the hypergraph.

Definition 6. (Move) A *move* (or sometimes a *turn*, which still refers to one move, instead of two) refers to the action of a player placing their pieces in a square. The player who makes the move is called the *active player*, and the other player is the *opponent*. A move is *invalid* if the active player attempts to place their pieces in an occupied square. We say a move *forms a match* if by making this move, the active player completes a win condition; However, this does not necessarily mean that the player wins the game. A move *wins the game* if no earlier moves have formed a match. Realistically, **Connect** games ends as soon as one of the player forms a match, but in our reduction, we WLOG let the players keep playing the game until the board is completely filled up. As a result, there may be multiple matches through out a game. On the contrary, a move is called a *false win* if some earlier moves have formed a match already.

Finally, we let $n = |V|$ and $m = |E|$ for $H = (V, E)$ in the **HC** instance.

3 The reduction

The **HC** game can be abstracted into two logical steps. The first step is where the players place their pieces on the board until it is filled up, and the second step is where we resolve the game by finding out which player, if any, formed a match first. We simulate the first step in the *Record Zone* and the *Validation Zone*, and the second step in the *Declaration Zone* and the *Proof Zone* (Fig. 3).

In the Record Zone, we simulate players making moves. Abstractly, each player making a move is just them jotting down a number from 1 to n, and the Record Zone keeps a record of all such numbers jotted down in each turn. However, we need a way to check or prevent invalid moves being played, which is handled by the Validation Zone.

The Validation Zone structurally overlaps much with the Record Zone. Its role in the reduction, narratively, is to provide the opponent of the active player a chance to challenge an invalid move at the end of each turn. If the opponent manages to prove that the active player did indeed make an invalid move, they immediately win. Because of this mechanism, optimal players will never make invalid moves.

After we are done simulating the first major step of **HC**, the players are given a chance each to declare whether they have managed to win the game in the Declaration Zone. If both players give up the chance, then the game will be tied. If one of the players takes the chance, then they must prove that they have indeed won the game in the Proof Zone (a *proof of victory* so to speak). If they cannot do it (which means they were bluffing), then their opponent wins.

The case where both players declare themselves as the winner is a bit more complicated, as we have to find out which player formed a match first. To this end, we have the players declare integers k_1 and k_2, respectively, such that $1 \leq k_1, k_2 \leq n$, and $k_1 \in A$, $k_2 \in B$. Now, the players are not only declaring whether they have won, but also specifying a time limit within which they managed to win. As a consequence, if they are to show a proof of victory, they must use only the first k_1 or k_2 moves of the game. Whoever declared the smaller number bears the responsibility of proving their claim. We will show the correctness of this process later when we get to the details of the Declaration Zone.

Finally, the Proof Zone is where players actually prove that they did win

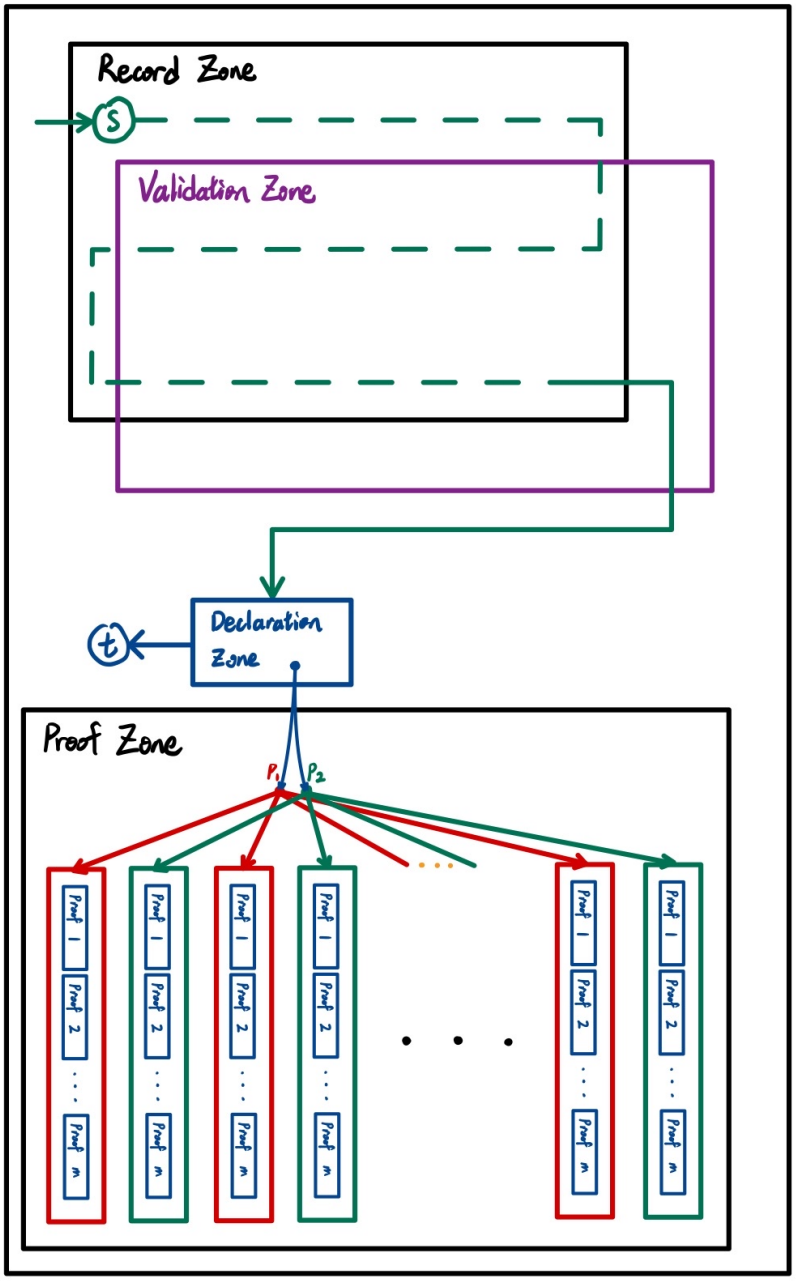

Figure 5: Blue print for the reduction.

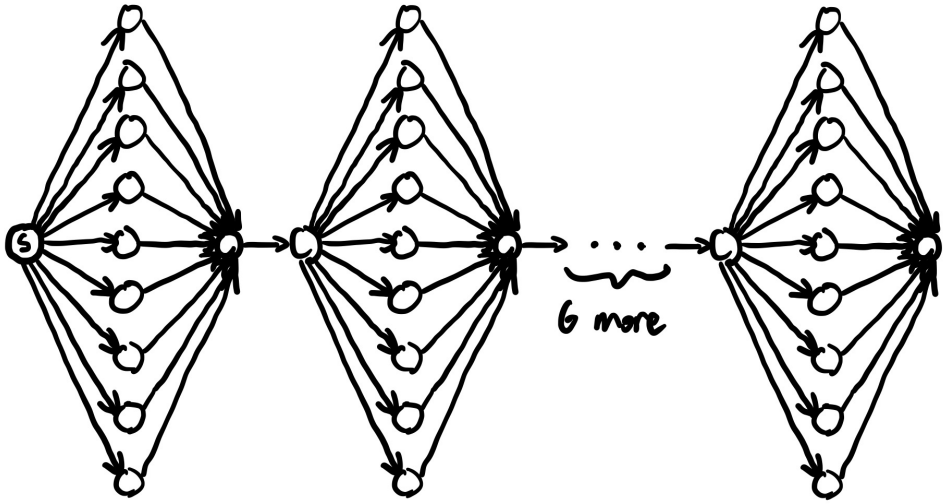

Figure 6: Example on a game board with 9 squares.

the game. Depending on the number k the player declared in the previous step, they enter one of the n *cutoff* blocks, each of which contains m *proof widgets*. Each proof widget corresponds to a win condition encoded by E. The player must choose one proof widget to proceed. The widget is designed in such a way that the proving player wins if and only if they have pieces on the board that *forms a match*. Note very carefully that the proof widgets alone cannot check if the match used in the player's proof actually *wins* the game; however, we will show that, with the help of the Declaration Zone, this is not an issue.

3.1 Construction of the Record Zone

For the Record Zone, we use a generalized version of the diamond widget that also appears in the TQBF $\leq_p$ GG reduction by Sipser as shown in Figure 6[2]. Each diamond widget has n middle nodes, and there are n widgets strung together. For the convenience of referencing the nodes, lets call the j^{th} node in the column of i^{th} widget (i, j), and the node that "spreads out" the edges i_{pre}, and the node that "collects" the edges i_{post}. We can think of the variable i as the turn number, and j as the active player of that turn choosing the j^{th} square (in some natural ordering of V that is consistently used in all widgets of the construction) to place their pieces in. Through one game, P_1 and P_2 take turns to pick exactly one node for each $1 \leq i \leq n$. The

Record Zone alone does not ensure each j is unique; We need to supplement it with the Validation Zone.

Complexity of the Record Zone

Node count: $O(n^2)$. Edge count: $O(n^2)$

3.2 Construction of the Validation Zone

The main purpose of the Validation Zone is to prevent players from making illegal moves. After each move, the opponent of the active player is given a choice: Either proceed to the next widget as normal (by going into the node that collects all choices in the currect widget), or accuse the active player of making an illegal move by specifying which previous move has already placed a piece in the j they just chose. To this end, we attach an auxiliary node to each (i, j), called $a_{i,j}$, and add an arc $a_{i,j} \to (i, j)$ (Fig. 7). Note that we can omit the last set of auxiliary nodes, since, as evident from the construction following, they are not reachable from anywhere. Next, for all (i, j), we connect (i, j) to all $a_{i',j'}$ such that $i' < i$ and $j' = j$ (Fig. 8, 9, 10).

Proposition 1. *The Record Zone always will encode a valid game if both players play optimally*[3]*.*

Proof. Suppose that either player have played (i, j) already, and now WLOG P_1 plays (i', j), where of course $i < i'$. This is an invalid move, so P_2 now decides to go to $a_{i,j}$. The only node accessible from $a_{i,j}$ is (i, j), which is already taken. Thus, P_1 loses the game, and did not play optimally. Now suppose the contrary, that (i, j) has not been played, and P_2 decides to choose $a_{i,j}$, falsely accusing P_1. Then, P_1 will be able to go to (i, j), while P_2 will meet a dead end, since by now i_{post} must have already been occupied (by the opponent of the player who occupied (i, j)). Therefore, falsely accusing the other player is also an undesirable strategy.

Since neither player could make an invalid move, and neither player could interrupt the game if their opponent does not make an invalid move, we know that the moves recorded in the Record Zone must be valid. □

[3]Playing optimally means the player 1) does not make a decision that would cause the game to transition into a worse state, 2) wins the game as quickly as possible, and 3) prolongs the game as much as possible if it is in a losing or tying state.

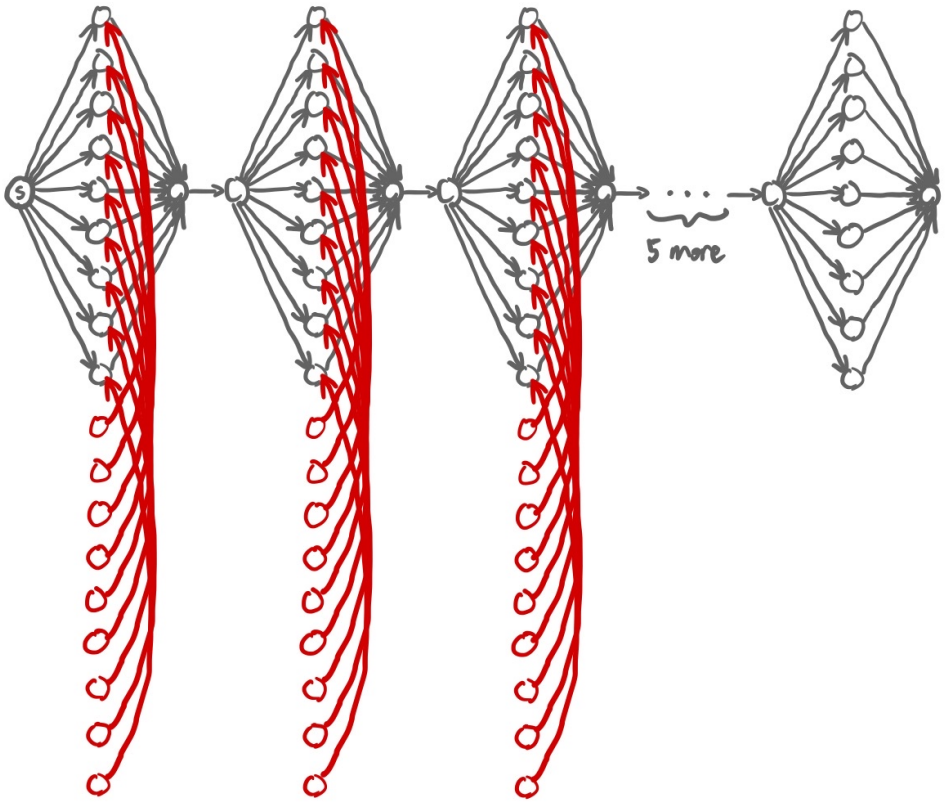

Figure 7: Adding $a_{i,j}$ for each i and j except $i = n$.

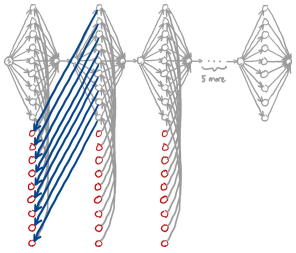

Figure 8: First step.

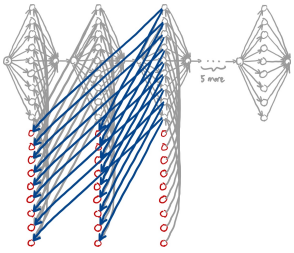

Figure 9: Second step.

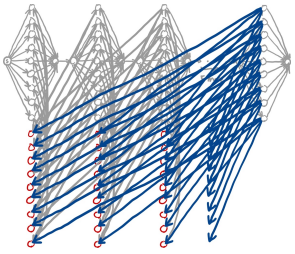

Figure 10: Final step.

Complexity of the Validation Zone

Node count: $O(n^2)$. Edge count: $O(n^3)$

3.3 Construction of the Declaration Zone

We now have a valid **HC** game encoded in the Record Zone. Next, we show a way of constructing the Declaration Zone.

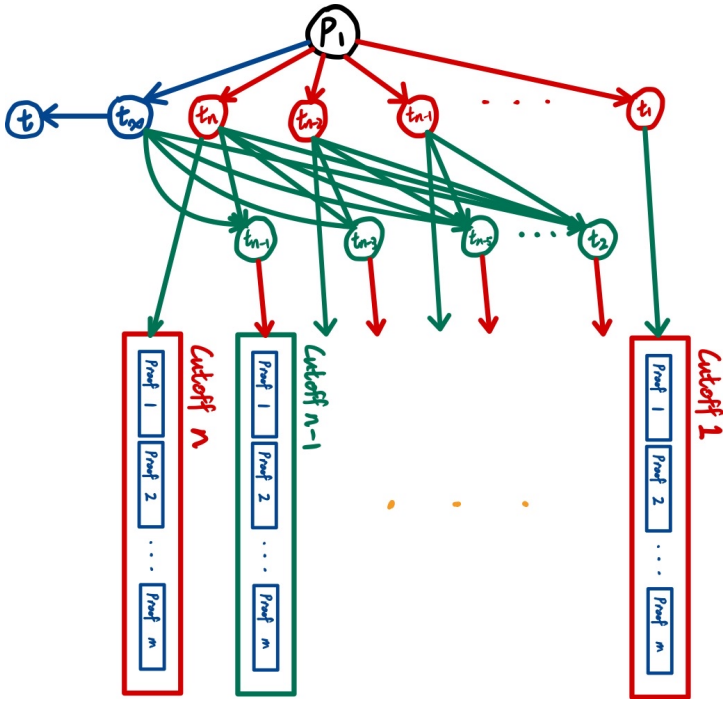

Figure 11: The Declaration Zone with part of the Proof Zone.

First of all, we may need to add one buffer node between the previous zone and this zone, so that P_2 chooses the node labeled "P_1" in Figure 11, and P_1 has the freedom of choosing the next node in this zone. This zone consists of two rows of nodes (colored red and green in Fig. 11). Let's label the nodes $t_1 \sim t_n$. All t_i such that $i \in A$ are in the first row, and those $\in B$ are in the second row. There are also a t_∞ node and the special t node (the tie node), colored in blue.

Connect the "P_1" node to every node in the first row, as well as the t_∞ node. Then, for every t_i in the first row (including the t_∞ node), connect it to all t_j in the second row, such that $i > j$. Finally, connect t_∞ to t.

Though we are not quite there yet, in order to explain the idea behind this construction, we include parts of the Proof Zone in Figure 11. For now, we state without proof that:

Proposition 2. *Let's call the opponent of the player that enters the i^{th} cutoff-block the* ***prover****. The prover wins the* ***GG*** *game if and only if they had formed a match in the Record Zone by move i.*

As explained in Section 3, each player is required to declare on which turn they won the game, essentially a number k between 1 and n, or give up the chance to do so and hope for a tie, which we encode as declaring the number ∞. Upon actual implementation, P_1 is the first to declare a number k, followed by P_2, who is only allowed to declare a number m that is smaller than k, or infinity; Declaring $m = \infty$ would mean the game is tied if $k = \infty$, or entering the k^{th} cutoff block if otherwise.

Now, we claim that:

Proposition 3. *Assuming optimal play from both players, as well as Prop.2, the player who won the* ***HC*** *game wins the* ***GG*** *game. Furthermore, the* ***GG*** *game is tied if no one wins the* ***HC*** *game.*

Proof. Assume for context that P_1 forms their first match on move $i \in A$, and P_2 on move $j \in B$ (unless the player never formed a match, in which case the value of ∞ is assumed). First, we claim that it is not optimal for P_1 to declare a number $k < i$. Suppose they do, then P_2 can simply go to the k^{th} cutoff block, and win the game by Proposition 2. The same can be said about P_2's strategy.

Now, suppose $i < j$, then P_1 can force a win by declaring a $i \leq k < j$. Next, suppose $i > j$, since the number k declared by P_1 is no less than i, P_2 can declare a $j \leq m < k$ and force a win. Finally, suppose $i = j$ ($= \infty$), then since neither player can afford to declare anything that is less than infinity, the game is tied. □

Complexity of the Declaration Zone

Node count: $O(n)$. Edge count: $O(n^2)$.

3.4 Construction of the Proof Zone

As seen in Fig. 3 and Fig. 11, the Proof Zone consists n almost identical copies of *cutoff blocks*. Each cutoff block contains a starting node and m *proof widgets*, one for each $e \in E$. The player at the starting node (whom we

call the "prover") is expected to choose one widget such that the encoded piece placement of theirs in the Record Zone completes the hyperedge e corresponded to that widget if they indeed formed a match as they claimed in the Declaration Zone.

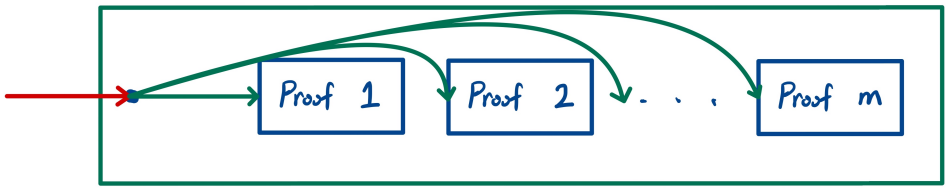

Figure 12: A single cutoff block.

We use a similar method that was employed previously in the construction of the Validation Zone, that is, augmenting each record widget with an additional column of nodes, to check whether a particular move did place a piece in a specified squre. Call the added nodes $b_{i,j}$, and add edges from all $b_{i,j}$ to (i, j). These nodes can be shared among all proof widgets.

Let the cutoff number be c, and $e \subseteq V$ be the hyperedge corresponding to the proof widget chosen. WLOG, assume $c \in A$; If this is not the case, replace every A with B. Call this particular proof widget "Cutoff c – Proof e". It consists of $|e|$ nodes[4]. Let's label them $P_{c\text{-}e\text{-}1}$, $P_{c\text{-}e\text{-}2}$, ..., $P_{c\text{-}e\text{-}|e|}$ (see example in Fig. 13). Denote the n^{th} node in e as $e[n]$. For every $1 \leq n \leq |e|$, connect $P_{c\text{-}e\text{-}n}$ to every $b_{i,e[n]}$ such that $i \leq c$ and $i \in A$. Next, connect all such $b_{i,e[n]}$ to $P_{c\text{-}e\text{-}(n+1)}$ unless $n = |e|$.

Proposition 4. *The prover of Cutoff c – Proof e wins the **GG** game if and only if they had completely covered the hyperedge e by move c in the **HC** game.*

Proof. Suppose the prover did have covered e by move c, say, they captured $e[i]$ on move $m_i (\leq c)$ for all $1 \leq i \leq |e|$. A winning strategy for the prover is:

$$(P_{c\text{-}e\text{-}1}) \longrightarrow b_{m_1,e[1]} \xrightarrow{\text{forced}} P_{c\text{-}e\text{-}2} \longrightarrow b_{m_2,e[2]} \xrightarrow{\text{forced}} P_{c\text{-}e\text{-}3} \longrightarrow \cdots$$

$$\cdots \longrightarrow b_{m_{|e|-1},|e|-1} \xrightarrow{\text{forced}} P_{c\text{-}e\text{-}|e|} \longrightarrow b_{m_{|e|},|e|}$$

This works because in each step, the only other node reachable from $b_{m_i,e[i]}$ – that is $(m_i, e[i])$ – has already been taken by the prover, so the opponent

[4] An additional buffer node is needed before the first node.

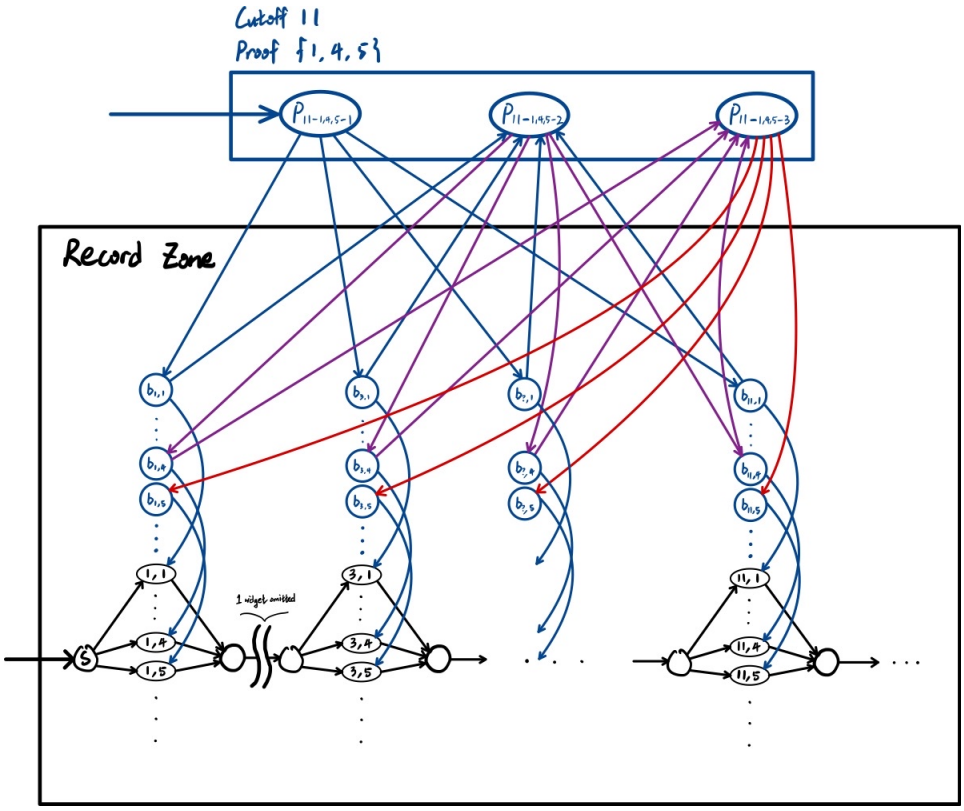

Figure 13: Diagram of an exemplar Cutoff 11 – Proof $\{1, 4, 5\}$.

can only go to $P_{c\text{-}e\text{-}(i+1)}$ – the next node in the proof widget. This goes on until the very last node, which is connected only to the node in the record widget, thus the opponent loses the game.

Now suppose the contrary, that the prover had not covered all nodes in e by move c. WLOG, suppose the $e[j]$ is not captured by the prover within the first c moves, which means all $(i, e[j])$ are vacant for $i \leq c$. The prover must choose a node to go from $P_{c\text{-}e\text{-}(j)}$, say $b_{i,e[j]}$, but then the opponent can now go to $(i, e[j])$ from there, and win the game. □

Finally, we can prove Proposition 2:

Proof. If the prover had formed a match, say e, by move i, then by Prop. 4, they can win the **GG** game by going into the Cutoff i – Proof e widget. If the prover had not formed any match by move i, then for whichever proof widget they choose, they lose the **GG** game by Prop. 4. □

Complexity of the Proof Zone

Node count: $O(n^2)$ for the shared part; $O(mn^2)$ for n cutoff blocks, m proof widgets per block, and up to n nodes per proof widget. Edge count: $O(n^2)$ for the shared part; Plus $O(mn^3)$ since each node has $O(n)$ edges.

3.5 Putting it all together

With our careful design of the **GG** game board, we make the players essentially play the original **HC** game on a different medium. While the players are in the Record Zone, they are essentially placing their pieces into the squares. The Validation Zone, Declaration Zone and Proof Zone do not correspond to any explicit actions taken by the players; Instead, they encode game rules that limit the players' action, and rules that decide the winner. The important theorem is prop. 3, which essentially shows the correctness of the reduction.

4 Conclusion and discussion

This main purpose of this paper is to demonstrate several techniques and ideas when constructing reductions between gaming problems. This includes abstraction of the game, extraction and encoding of the game's relevant information, validate players' action by mutual supervision between the players themselves, and the concept of proof of victory.

Alternative construction

In our reduction, the **HC** game is played in full before a winner is decided. Is it possible to devise a reduction, where the simulation of the game stops as soon as the winning move is made? It is easy to just add an edge that *escapes* the Record Zone, but the uninitialized record widgets would require a redesign of the proof widgets.

Further questions

How to encode special rules? In professional competitions of GoMoku, special restrictions that prevents certain plays like 4 by 4 from being made are usually applied to P_1. More exotic rule variations, like swapping pieces after move 1, also exist. Adding these restrictions does not seem to affect the hardness of the problem, so it is theoretically possible to devise a reduction regardless.

References

[1] Ming Yu Hsieh and Shi-Chun Tsai. On the fairness and complexity of generalized k-in-a-row games. *Theoretical Computer Science*, 385(1):88–100, 2007.

[2] Michael Sipser. *Introduction to the Theory of Computation*. Course Technology, Boston, MA, third edition, 2013.