Dots & Boxes Is PSPACE-Complete

Kevin Buchin
Department of Mathematics and Computer Science, TU Eindhoven, The Netherlands

Mart Hagedoorn
Department of Mathematics and Computer Science, TU Eindhoven, The Netherlands

Irina Kostitsyna
Department of Mathematics and Computer Science, TU Eindhoven, The Netherlands

Max van Mulken
Department of Mathematics and Computer Science, TU Eindhoven, The Netherlands

Abstract

Exactly 20 years ago at MFCS, Demaine posed the open problem whether the game of Dots & Boxes is PSPACE-complete. Dots & Boxes has been studied extensively, with for instance a chapter in Berlekamp et al. *Winning Ways for Your Mathematical Plays*, a whole book on the game *The Dots and Boxes Game: Sophisticated Child's Play* by Berlekamp, and numerous articles in the *Games of No Chance* series. While known to be NP-hard, the question of its complexity remained open. We resolve this question, proving that the game is PSPACE-complete by a reduction from a game played on propositional formulas.

2012 ACM Subject Classification Theory of computation → Complexity classes

Keywords and phrases Dots & Boxes, PSPACE-complete, combinatorial game

Digital Object Identifier 10.4230/LIPIcs.MFCS.2021.25

1 Introduction

Dots & Boxes is a popular paper-and-pencil game that is played by two players on a grid of dots. The players take turns connecting two adjacent dots. If a player completes the fourth side of a unit box, the player is awarded a point and an additional turn. When no more moves can be made, the player with the highest score wins the game.[1]

Originally described in 1883 [29], Dots & Boxes has since received a considerable amount of attention in the research community. In *Winning Ways for Your Mathematical Plays*, Berlekamp, Conway, and Guy [6] were among the first to discuss a number of interesting mathematical properties of the game. Later, Berlekamp [5] wrote an entire book *The Dots-and-Boxes game: Sophisticated Child's Play* about the game, in particular discussing winning strategies in particular positions. Since then, the mathematics of Dots & Boxes and variants have been discussed in many papers and books [1, 2, 7, 12, 16, 21, 22, 26, 30, 31, 33, 34]. There is also a rich body of work on solvers for Dots & Boxes [3, 4, 11, 27, 35].

Berlekamp et al. [6] argue that deciding the winner of a generalized version of Dots & Boxes, called *Strings-and-Coins*, is NP-hard. In this game, players take turns in removing edges of a given graph, scoring a point when they isolate a vertex. When restricted to the dual graph of a square grid, this corresponds to a dual formulation of Dots & Boxes. Eppstein [17] notes that the reduction given by Berlekamp et al. should extend to Dots & Boxes, and a formal proof of the NP-hardness is given in [8].

Exactly 20 years ago at MFCS, Demaine posed the open problem whether Dots & Boxes is PSPACE-complete [13]. Bounded two-player games, like Dots & Boxes, (that is, games in which the number of moves is bounded) naturally lie in PSPACE, since a Turing machine using space polynomial in the board size is able to search the entirety of the game

[1] For a visual explanation of the game see `https://youtu.be/KboGyIilP6k`, last accessed 6.5.2021

46th International Symposium on Mathematical Foundations of Computer Science (MFCS 2021).
Editors: Filippo Bonchi and Simon J. Puglisi; Article No. 25; pp. 25:1–25:18
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

space. PSPACE-hardness of many bounded two-player games is shown by a reduction from *Generalized Geography*, which is proven PSPACE-complete by Lichtenstein and Sipser [28]. For example, the PSPACE-completeness of Reversi [24], uncooperative UNO [14], and Tic-Tac-Toe [23] were shown by a reduction from Generalized Geography. However, unlike Dots & Boxes, the setting of Generalized Geography prescribes a stricter order on players' moves, making a reduction to Dots & Boxes challenging to obtain.

In their seminal work, Hearn and Demaine [20, 21] introduce *Constraint Logic*, a framework for analyzing complexity of games and puzzles. Inspired by Flake and Baum's proof of Rush Hour [18], it specifies a type of game played on a constraint graph. The framework includes bounded/unbounded state spaces and single/two-player variations. In the same work, Hearn and Demaine go on to provide a number of simpler reductions for various known PSPACE-complete games (including Rush Hour), as well as new proofs for several PSPACE-complete games. However, the Constraint Logic framework is intended for proving hardness of partisan games (games in which the moves available to the two players are different), whereas Dots & Boxes is not a partisan game.

Strings-and-Coins and the related game of *Nimstring* were very recently (while we were preparing this submission) proven to be PSPACE-complete by Demaine and Diomidov [15] by a reduction from a game on a DNF formula G_{pos}(POS DNF) [32]. But, as they point out, their results do not apply to Dots & Boxes, since the game positions they construct rely on multi-graphs (which additionally are neither planar nor have a maximum degree of 4). Specifically, they propagate signals through multi-edges consisting of a polynomial number of parallel edges, and the winner is the player who removes the last edge. As consequence, our reduction bears little commonalities with theirs.

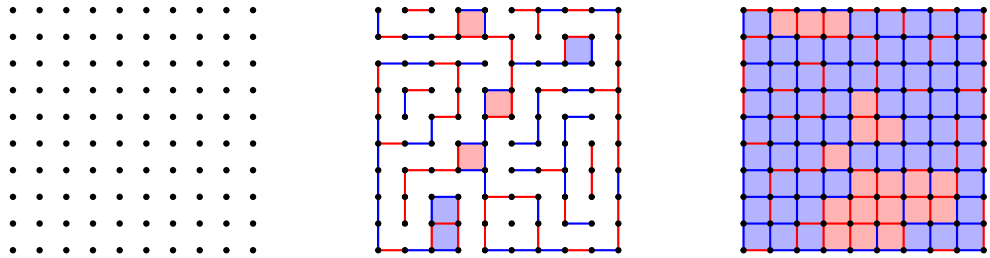
In this paper, we prove that Dots & Boxes is PSPACE-complete by a reduction from G_{pos}(POS CNF). The starting point of our construction are strategies for Dots & Boxes endgames that were also used to prove NP-hardness. However, the NP-hardness is proven by having one player be in control, such that therre is only one way for the other player to respond. This de facto makes the game into a 1-player game. For PSPACE-hardness we need both players to have choices, making it a true 2-player game. This gives a lot of freedom to the players, and makes it much more difficult to construct gadgets to control the gameplay, in particular because moves and scoring opportunities for one player – if not played immediately – are also available to the other player.

In Section 1.1 we discuss the gameplay of Dots & Boxes, and introduce terminology coined by Berlekamp et al. [6]. In Section 2 we present the general structure of our reduction, and then describe our gadgets in Section 3. In Section 4 we first show properties of optimal play for both players and finally prove PSPACE-hardness.

1.1 Dots & Boxes

On the surface, Dots & Boxes is quite a simple game. The starting and a typical final position for a 10×10 grid are shown in Figure 1. We refer to the players playing the blue and the red colors as Trudy and Fred, respectively. The color of a line connecting two dots indicates which player drew it, and the color of a box indicates which player closed it.

Consider a dual graph G of a board of Dots & Boxes, where a node in G corresponds to a box or the unbounded face, and a pair of nodes in G is connected with an edge if the corresponding move is still available, i.e., the line between the boxes has not been drawn. Let the *degree* of a box be the degree of the corresponding node in G.



(a) Starting position of a Dots & Boxes game with a 10×10 grid.

(b) An example of a loony end-game state of the game.

(c) A possible end position. Trudy (blue) won with 62 boxes versus Fred's (red) 19 boxes.

Figure 1 Typical starting, intermediate, and final position of a Dots & Boxes game.

In Dots & Boxes, a typical game usually results in a board state that consists exclusively of moves that open the possibility for the opponent to claim a number of boxes in their next turn (see Figure 1b). That is, in this state there are no degree-1 boxes, but any move made by a player creates a degree-1 box that can be immediately claimed by the opponent. Consider such a board configuration S and any available move ℓ in it. At least one box b incident to ℓ has degree two in S (before the move ℓ is made). Consider a maximal component σ of degree-2 boxes in S containing b. There are two cases, either σ is a *chain* ending in boxes of degree higher than two (or the outer face), or σ is a *cycle*. Then we say that a player making the move ℓ *opens* the chain (cycle) σ for the opponent.

To devise a good strategy for Dots & Boxes, it is important to note that a player is not obliged to claim a box whenever they can. It is sometimes beneficial for a player to sacrifice a small number of boxes for long-term gain. Consider the position in Figure 2, and let it be Fred's (red) turn. Here, Fred could claim the bottom three boxes (Figure 2 (top)). However, after doing so Fred has to make an extra move, allowing Trudy (blue) to claim the remaining four boxes and win the game. But by sacrificing two boxes (Figure 2 (bottom)), Fred can force Trudy to make another move and open the middle chain for him to claim. That way, Fred loses two boxes in the bottom chain, but gains all four boxes in the middle chain.

In *Winning Ways*, Berlekamp et al. [6] refer to the moves sacrificing a small number of boxes but passing the turn onto the opponent as *double-dealing* moves. Double-dealing moves can be made in chains of boxes, sacrificing two boxes, and in cycles, sacrificing four boxes (see Figure 3). Each double-dealing move is usually immediately followed by the opponent making at least one *double-cross* move, i.e., a move that closes two boxes at once. Note that

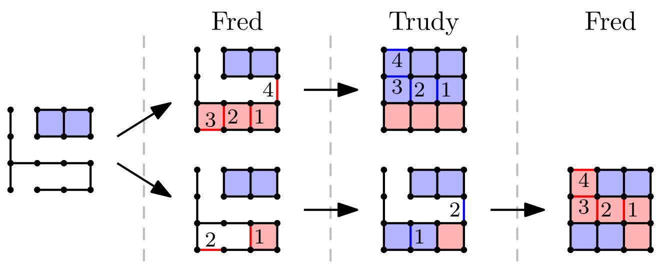


Figure 2 Two possible plays that Fred (red) can do. Fred can choose to claim all the available boxes (top) and lose the game, or to perform a double-dealing move sacrificing two boxes (bottom, second state, edge 2), and win the game. The order of the edges that are played by Trudy or Fred in one turn is indicated by edge labels. This example is borrowed from *Winning Ways, chapter 16* [6].

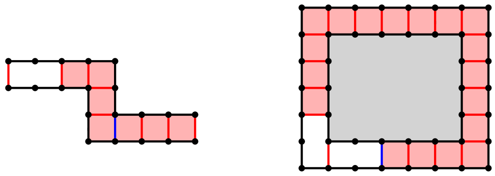


Figure 3 Double-dealing move by Fred (red). If Trudy (blue) opens a chain (or a cycle), Fred can claim a sequence of boxes. To pass the turn back to Trudy, Fred can leave two (or four) boxes unclaimed.

double-dealing moves are only possible in *long* chains of at least three boxes, and in cycles. (Chains of length one do not have enough boxes for a double-dealing move, and a chain of length two can be opened by selecting the middle edge, thus preventing the opponent from playing a double-dealing move.)

Berlekamp et al. [6] refer to moves opening a long chain or a cycle as *loony* moves. Making a loony move is not always a choice, since at some point in the game, all unclaimed boxes are part of long chains and cycles as in Figure 1b. Such positions are referred to as a *loony endgames*. Note that in chains of length ≥ 4 and cycles of length ≥ 8, the player making the double-dealing moves scores at least as many boxes as their opponent. Thus, in loony endgames with chains of length ≥ 4 and cycles of length ≥ 8, under optimal play, the game consists of one player making loony moves (opening chains and cycles), and the other player claiming all but two or four boxes, and making double-dealing moves to pass the turn back to the opponent [6]. Here, the player making the double-dealing moves is always better off, since each chain or cycle yields at least as many boxes to this player as it yields to their opponent. This player is thus referred to as being *in control* of the game. The benefit of being in control can be seen in Figure 1c, which is the end result of Trudy being in control of the loony endgame shown in Figure 1b.

In *Winning Ways*, Berlekamp et al. [6] argue that the player not in control has to maximize the number of disjoint cycles to maximize their score, because double-dealing moves in cycles yield twice as many boxes as double-dealing moves in chains. Since this property is important for our reduction, we restate it here and, for completeness, present the argument in the appendix.

▶ **Lemma 1.** *Let the configuration of a loony endgame contain k boxes with degree higher than 2, let T be the sum of the degrees of these boxes, and let c be the maximum number of disjoint cycles in the configuration. Then, the player who is* not *in control can claim at most $4c + T - 2k - 4$ boxes.*

Note that in this lemma we only count boxes; even though the outer face contributes to the degrees of adjacent boxes, it is not counted here.

2 Structure of the construction

To show that Dots & Boxes is PSPACE-hard we reduce from the game G_{pos}(POS CNF), introduced and proven PSPACE-complete by Schaefer [32]. The game is played by two players, Trudy and Fred, on a positive CNF formula $\mathcal{F}$. The players take turns picking a variable that has not yet been chosen. Variables picked by Trudy are set to TRUE, variables picked by Fred are set to FALSE. When all variables have been chosen, the game ends. Trudy wins if formula $\mathcal{F}$ evaluates to TRUE, and Fred wins if formula $\mathcal{F}$ evaluates to FALSE.

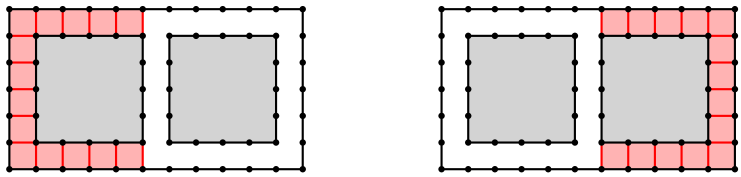


Figure 4 A choice of a cycle can encode the value of a signal.

Given a positive CNF formula $\mathcal{F}$ with n variables and m clauses, we construct an instance of Dots & Boxes in which Trudy has a winning strategy if and only if she also has a winning strategy in the corresponding instance of G_{pos}(POS CNF). For simplicity we assume that n is even, so that Trudy and Fred get to assign values to the same number of variables. If the number of variables in $\mathcal{F}$ is odd, we can introduce dummy variables without changing the outcome of a game such that the total number of the variables becomes even. For each variable and clause of $\mathcal{F}$ we construct a variable and a clause gadget, respectively. We place the variable gadgets in a row at the top of the board of Dots & Boxes, and the clause gadgets in a row at the bottom. We connect the variable gadgets to their corresponding clause gadgets using the wire gadgets, which transfer the values of the variables to the clauses. If a clause consists of more than one variable, the wires from these variables must pass through an or gadget. Since the signals propagating from the variables may need to cross each other, we construct a crossover gadget that preserves the values in the two crossing wires. In our instance of Dots & Boxes, only the gadgets contain available moves. The remaining boxes on the board have all the incident edges present.

As we detail in Section 4, after the values of the variables are set, the game enters a loony endgame where Fred is in control. Then Trudy's winning strategy reduces to selecting a maximum set of disjoint cycles $\mathcal{C}_{\max}$ in the remaining configuration (Lemma 1). In the remainder of this paper, we describe a strategy for both players referred to as *regular play*. Later, we will show that following the regular strategies is optimal for both players, in the sense that it maximizes their scores. Under regular play, Trudy opens all the chains outside of $\mathcal{C}_{\max}$ first, gaining two boxes per chain, and opens the chosen cycles last, gaining four boxes per cycle except the last one. Regular play for Fred is to ensure that he will be in control when the loony endgame starts. After entering the loony endgame, regular play for Fred is simply making double-dealing moves until his very last turn.

Most of our gadgets consist of partially overlapping cycles of boxes. The choice of a set of disjoint cycles determines the value of a signal. For example, in Figure 4 the choice of the left vs. right cycle can encode the value TRUE vs. FALSE. Of course, Trudy could join the cycles together to select the outermost cycle, but this, as we show later, will not be more beneficial.

As both players must have a choice in picking which variable to set, the instance of Dots & Boxes cannot yet be in a loony endgame. Thus, the variable gadgets, which we describe in Section 3.4, contain non-loony moves instrumental in setting the value of a variable. We ensure that under regular play the variables are set in alternating fashion, where Trudy sets them to TRUE, and Fred sets them to FALSE. Once all variables are set, the loony endgame is entered. At this point Fred is in control of the game, and it is up to Trudy to maximize her score by maximizing the number of disjoint cycles in $\mathcal{C}_{\max}$. The regular play by Trudy results in a correct propagation of the signals from the variables to the clauses.

To ensure that regular play by both players in the instance of Dots & Boxes corresponds to a valid G_{pos}(POS CNF) game, our gadgets need to give a specific number of boxes to

Trudy depending on the signal values. We will show that after the variable values have been set, under regular play, Trudy can maximize her score only if the signals are propagated correctly. Every gadget, except for the clause, yields the same number of disjoint cycles independent of the signals passing through the gadget. Only the clause gadget gives more cycles to Trudy if a TRUE signal reaches it. Exactly half of the variables are set to TRUE, and half to FALSE. Thus we can tune the starting score count such that the game is won by Trudy if and only if all the clauses are satisfied.

3 Gadgets

In this section we provide the details of the gadgets used in our reduction. When describing the gadgets below, for a simpler exposition, we assume that the moves that Trudy and Fred make follow the following sequence. First, in the first n moves Trudy and Fred set all the variables to TRUE and FALSE respectively. Afterwards, when the loony endgame is entered, the order in which Trudy selects which cycles to add to the disjoint set of cycles $\mathcal{C}_{\max}$ is from the top to bottom, that is, from the variables, through the outgoing wires, through the crossover and or gadgets, and finally down to the clause gadgets. Later, in Lemma 7, we will show that, indeed, under regular play Trudy and Fred start by setting all the variables. Furthermore, we will argue that the outcome of the game depends only on the choice of the cycles in $\mathcal{C}_{\max}$, and not on the order in which Trudy selects them.

3.1 Basic wiring

Signals from the variable gadgets are propagated to the clause gadgets through wires. A wire gadget consists of a chain of an even number of partially overlapping cycles (see Figure 5). The first cycle in the wire overlaps with the gadget from which the signal is propagated, and the last cycle overlaps with the gadget towards which the signal is propagated. Consider some wire w, let C_i be its first cycle overlapping with gadget G_i, and let C_o be its last cycle overlapping with gadget G_o. If C_i is disjoint from the cycles of G_i that Trudy adds to $\mathcal{C}_{\max}$, then we say that the input signal to the wire is TRUE; otherwise, if C_i overlaps with one of the cycles of G_i in $\mathcal{C}_{\max}$, the input value is FALSE. If Trudy does not add C_o to $\mathcal{C}_{\max}$, then the output signal is TRUE, and the output signal is FALSE otherwise.

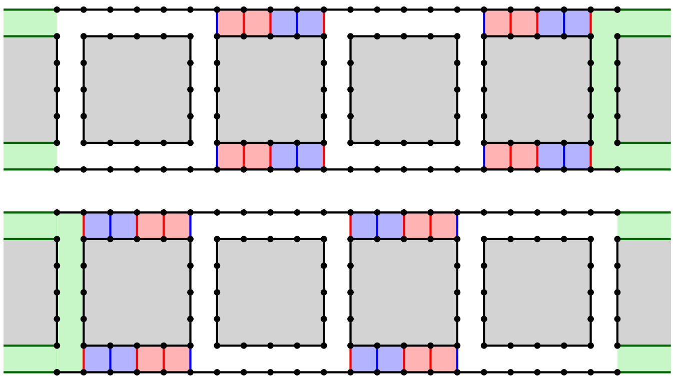


Figure 5 A wire gadget consisting of four overlapping cycles and two ways of selecting disjoint cycles. Shown in green are the connections to the adjacent gadgets. Selecting odd cycles in $\mathcal{C}_{\max}$ corresponds to TRUE (top), and selecting even cycles corresponds to FALSE (bottom).

To ensure that Fred always follows the strategy of double-dealing moves, we require that each maximal chain of degree-2 boxes in a wire gadgets contain at least four boxes. That way, Fred receives at least as many boxes in each chain (and cycle) as Trudy, and thus for Fred being in control is always beneficial [6].

Note that, besides the lower bound on the length of a chain, the size and the embedding of the overlapping cycles in a wire can be chosen freely. Thus wires are very flexible in connecting components together, which facilitates the construction.

▶ **Lemma 2.** *Let a wire w consist of $2k$ partially overlapping cycles. Then, under regular play, if the signal in w changes from* FALSE *to* TRUE*, then Trudy can select at most $k-1$ disjoint cycles from w to add to $\mathcal{C}_{\max}$. Otherwise, under regular play, Trudy can select k disjoint cycles from w to add to $\mathcal{C}_{\max}$.*

Proof. As we show in Lemma 7, after the first n moves, which Trudy and Fred make in the variable gadgets, the game enters a loony endgame with Fred in control. If the output signal in the wire matches the input signal, then only one of C_i or C_o of w are in $\mathcal{C}_{\max}$. Then Trudy can select all odd (if $C_i \in \mathcal{C}_{\max}$) or all even (if $C_o \in \mathcal{C}_{\max}$) cycles to add to $\mathcal{C}_{\max}$, which results in k disjoint cycles. If the the input signal is TRUE, and the output signal is FALSE, then both C_i and C_o are in $\mathcal{C}_{\max}$. Then Trudy can, for example, select $k-1$ odd cycles and C_o to add to $\mathcal{C}_{\max}$, which again results in k cycles in total.

If, however, the input signal is FALSE, and the output signal is TRUE, then neither C_i nor C_o can be in $\mathcal{C}_{\max}$. This leaves a chain of $2k-2$ cycles, of which at most $k-1$ disjoint cycles can be selected to be added to $\mathcal{C}_{\max}$. ◀

In our construction we ensure that Trudy can win only if she gets k disjoint cycles from a wire, and thus under regular play she cannot flip a signal propagating from a variable from FALSE to TRUE. Flipping a signal from TRUE to FALSE is not beneficial for Trudy, as her goal is to satisfy all the clauses. Nevertheless, flipping a signal from TRUE to FALSE leads to the same number of boxes for her (at least locally within a wire), and is thus allowed.

3.2 Crossover gadget

Since the graph representing G_{pos}(POS CNF) is not necessarily planar, wires may need to cross each other in our construction. We describe a crossover gadget that allows two signals to cross while preserving the signal values. The gadget has two inputs and two outputs

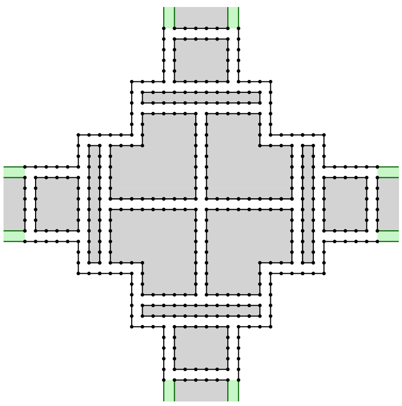


Figure 6 The crossover gadget. Connections to the adjacent wires are shown in green.

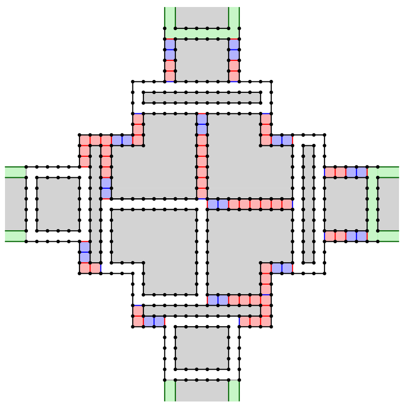


Figure 7 A possible choice of a set of disjoint cycles. The selection has fourth degree rotational symmetry.

on the opposite sides of the gadget. Let $C_{1,i}$ and $C_{2,i}$ be the input cycles of the gadget, and $C_{1,o}$ and $C_{2,o}$ be the output cycles (see Figure 6). An input cycle $C_{*,i}$ is in $\mathcal{C}_{\max}$ if the corresponding input signal is TRUE, and otherwise it is FALSE. An output cycle $C_{*,o}$ is not in $\mathcal{C}_{\max}$ if the output signal is TRUE, and otherwise it is FALSE.

There are four pairwise overlapping cycles C_a, C_b, C_c, and C_d in the middle of the gadget, forming a cross shape. Only one of these cycles can be added to $\mathcal{C}_{\max}$. A choice of which of these cycles is added to $\mathcal{C}_{\max}$ is in one-to-one correspondence to the input signal values (see Figure 7).

▶ **Lemma 3.** *Under regular play, if a signal in a* crossover *gadget changes from* FALSE *to* TRUE*, then Trudy can select at most* 4 *disjoint cycles from the gadget to add to* $\mathcal{C}_{\max}$*. Otherwise, under regular play, Trudy can select* 5 *disjoint cycles from the gadget.*

Proof. If the output signals in the crossover gadget match the input signals, then only one of each pair $\{C_{1,i}, C_{1,o}\}$ and $\{C_{2,i}, C_{2,o}\}$ are in $\mathcal{C}_{\max}$. Since the four center cycles C_a, C_b, C_c, and C_d all share a single square, only one of these four cycles can be chosen. Then Trudy can select a corresponding cycle from the middle of the gadget, and two more cycles from each signal. For example, a selection of five disjoint cycles for the case when the first input signal is FALSE and the second is TRUE is shown in Figure 7. If an input signal is TRUE, and the corresponding output signal is FALSE, then both $C_{*,i}$ and $C_{*,o}$ are in $\mathcal{C}_{\max}$. Then Trudy can, for example, make exactly the same choice as in the case where the output signal would have been TRUE.

Assume now, w.l.o.g., that the signal corresponding to $C_{1,i}$ and $C_{1,o}$ changes from FALSE to TRUE in the gadget. That is, neither $C_{1,i}$ nor $C_{1,o}$ are in $\mathcal{C}_{\max}$. Let C' and C'' be the cycles in the gadget adjacent to $C_{1,i}$ and $C_{1,o}$ respectively. Thus, among cycles C', C'', C_a, C_b, C_c, and C_d at most two cycles can be in $\mathcal{C}_{\max}$, and therefore at most four cycles can be chosen to be in $\mathcal{C}_{\max}$. ◀

3.3 Or gadget

The or gadget consists of three pairwise overlapping cycles (see Figure 8 (left)). Two of the cycles partially overlap with an end cycle of an input wire, and one cycle partially overlaps with the output cycle. Let $C_{1,w}$ and $C_{2,w}$ be the last cycles of the two input wire gadgets, and let $C_{1,i}$ and $C_{2,i}$ be the cycles of an or gadget adjacent to these two wires respectively. Let C_o be the third cycle of the or gadget, which is adjacent to an output wire. Cycles $C_{1,w}$ and $C_{2,w}$ are not in $\mathcal{C}_{\max}$ if the input from their corresponding wire is TRUE, and are in $\mathcal{C}_{\max}$ if their input is FALSE. If C_o is not in $\mathcal{C}_{\max}$ then the output of the or gadget is TRUE, and if it is in $\mathcal{C}_{\max}$ then the output value is FALSE. Only one of the three cycles in the or gadget can be selected to be added to $\mathcal{C}_{\max}$, and thus the output of the gadget can be TRUE only if one of $C_{1,w}$ or $C_{2,w}$ is in $\mathcal{C}_{\max}$.

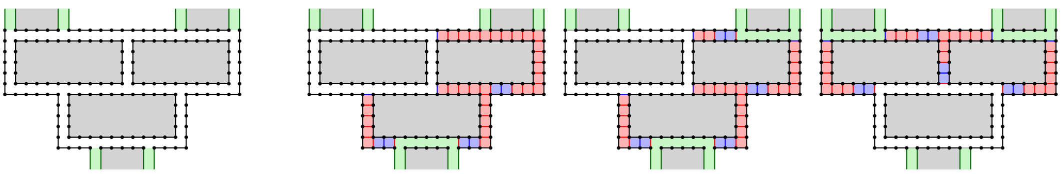


Figure 8 The or gadget (left) and the three possible combinations of the input values (right). From left to right: two TRUE inputs, one TRUE and one FALSE input, and two FALSE inputs. The boxes highlighted in green belong to a cycle in an adjacent wire gadget.

▶ **Lemma 4.** *Under regular play, if both input signals in an* or *gadget are* FALSE *but the output signal is* TRUE*, then Trudy cannot add any cycles from the gadget to $\mathcal{C}_{\max}$. Otherwise, under regular play, Trudy can select* 1 *cycle from the gadget to add to $\mathcal{C}_{\max}$.*

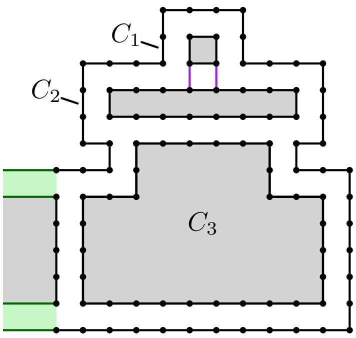
Proof. First consider the case when one of the input signals in the or gadget is TRUE. W.l.o.g., let the signal from the first wire be TRUE, that is $C_{1,w}$ is not in $\mathcal{C}_{\max}$. Then Trudy can select $C_{1,i}$ to add to $\mathcal{C}_{\max}$ and thus the output from the or gadget would correspond to TRUE. Trudy may as well choose C_o to add to $\mathcal{C}_{\max}$ and make the output of the gadget to be FALSE. In either case, one cycle from the gadget is in $\mathcal{C}_{\max}$.

If both input signals are FALSE, then both cycles $C_{1,w}$ and $C_{2,w}$ are in $\mathcal{C}_{\max}$. Thus none of $C_{1,i}$ and $C_{2,i}$ can be in $\mathcal{C}_{\max}$. Trudy can choose to add C_o to $\mathcal{C}_{\max}$ and make the output of the gadget to be FALSE. If, however, Trudy chooses to make the output of the or gadget TRUE, then C_o is not in $\mathcal{C}_{\max}$, and thus Trudy cannot select a single cycle to add to $\mathcal{C}_{\max}$ from this or gadget. ◀

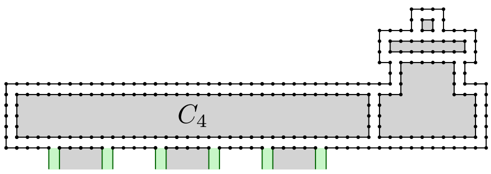
3.4 Variable gadget

The variable gadget is responsible for the assignment of TRUE and FALSE values to the variables of the G_{pos}(POS CNF) instance. It consists of two components: the *value-setting component* (see Figure 9) designed to set the value of the variable, and the *fan-out component* designed to duplicate the variable signal. The whole construction is presented in Figure 10. Let C_1, C_2, and C_3 be the three cycles in the value-setting component. The variable gadget is the only gadget that contains non-loony moves; there are two non-loony moves (shown in purple in the figure) at the intersection of C_1 and C_2.

Regular play by both Trudy and Fred is to set all the variables in the first n moves, such that Fred always sets a variable to FALSE and Trudy – to TRUE. Figure 11 shows the two possible value assignments of the variable gadgets. To set a variable to FALSE, Fred plays one of the non-loony moves in the corresponding variable gadget. Then Trudy responds by claiming the one box available (see Figure 11 (left)). This results in the cycles C_1 and C_2 getting merged. To set a variable to TRUE, Trudy opens a side chain of C_2 (see Figure 11 (right)). Then Fred responds by claiming every box in the opened chain, and proceeds to setting the next variable. Note that after Trudy's move the non-loony moves in the gadget become loony moves (as they are now a part of a long chain).



■ **Figure 9** The value-setting component of the variable gadget. There are two non-loony moves (purple) available, of which only one can be played as a non-loony move.



■ **Figure 10** The complete variable gadget consisting of the value-setting component and the fan-out component. Outgoing wires are shown in green.

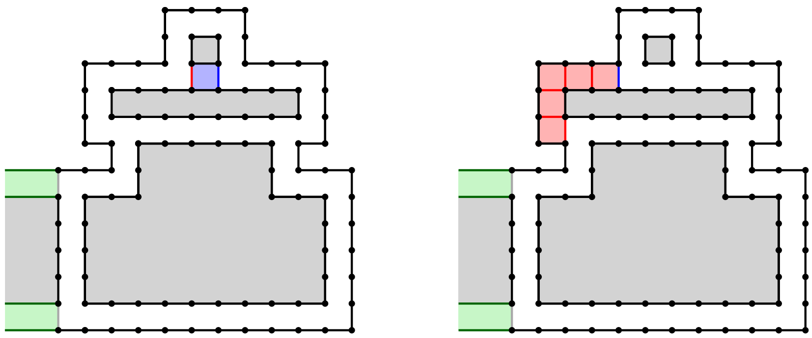


Figure 11 The variable is set to FALSE (left) and TRUE (right).

We make two observations which will be useful when proving correctness of the construction and the properties of the regular play in Section 4. First, observe that the non-loony moves come in pairs, one in each variable, such that, for each pair, either both moves in the pair are still non-loony or neither is anymore. We refer to them as *non-loony pairs*. Second, note that in the process of assigning values to the variable gadgets, Trudy gets a box for each variable set to FALSE by Fred, and zero boxes for each variable set to TRUE by herself.

Once the value of a variable is set, it propagates to the outgoing wires through the fan-out component of the variable gadget. The fan-out component simply consists of one cycle C_4 overlapping with the cycle C_3 (see Figure 10), to which multiple wires can be attached. After the variable is set, Trudy can add at most two cycles from it to $\mathcal{C}_{\max}$. Then, if the variable is set to FALSE, cycle C_4 has to be one of the two selected cycles, and thus the signal propagated into the wires is FALSE. If the variable is set to TRUE, Trudy can add C_1 and C_3 to $\mathcal{C}_{\max}$, and thus propagate the TRUE value into the wires. By considering the various cases under regular play, we obtain the following lemma.

▶ **Lemma 5.** *Under regular play, after a variable gadget is assigned a value, if it is set to FALSE but the output signal is TRUE, then Trudy can add at most* 1 *cycle from the gadget to* $\mathcal{C}_{\max}$*. Otherwise, under regular play, Trudy can add* 2 *cycles from the gadget to* $\mathcal{C}_{\max}$*.*

Proof. As we show in Lemma 7, optimal play of both Trudy and Fred results in them setting all the variables according to the rules described above in the first n moves. Afterwards the game enters a loony endgame with Fred in control.

If a variable gadget is set to TRUE, then there are three cycles left in the gadget: two overlapping cycles C_3 and C_4, and the cycle C_1 connected to C_3 by a chain. Then Trudy can select C_1 and one of C_3 or C_4 to add to $\mathcal{C}_{\max}$.

If the variable gadget is set to FALSE, then there are still three cycles left in the gadget, but now these cycles are forming a chain where each consecutive pair of cycles is overlapping. Thus, if the output value is FALSE, then C_4 and the merged cycle of C_1 and C_2 can be added to $\mathcal{C}_{\max}$. On the other hand, if the output value is TRUE then C_4 cannot be in $\mathcal{C}_{\max}$, and from the remaining two cycles, only one can be selected to be added to $\mathcal{C}_{\max}$. ◀

3.5 Clause gadget

Finally, we describe a clause gadget that yields more boxes to Trudy if the signal entering the clause corresponds to TRUE. A clause gadget is simply an extra cycle extending the end of a wire gadget to an odd length. Figure 12 shows the gadget, and the two possible assignments of this gadget. Whenever the signal is TRUE, it is possible for Trudy to create a disjoint cycle in the gadget which gives her four boxes. If the signal is FALSE, Trudy can only make a chain in this gadget which yields only two boxes. Again, by considering the various cases that can occur under regular play, we obtain the following lemma.

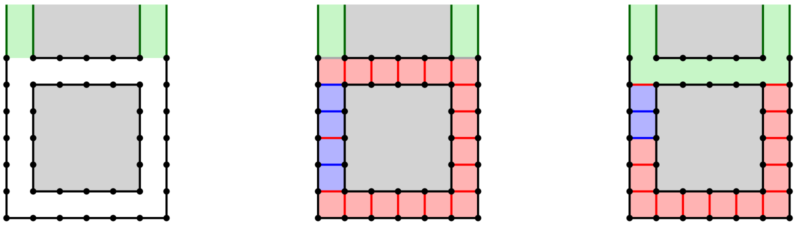


Figure 12 The clause gadget (left) yields four boxes to Trudy if the input signal is TRUE (middle), and only two boxes when the input is FALSE (right). Boxes highlighted in green belong to the last cycle in the adjacent wire.

▶ **Lemma 6.** *Under regular play, if the clause gadget is set to* TRUE*, then Trudy can add* 1 *cycle to* $\mathcal{C}_{\max}$*. Otherwise, under regular play, Trudy cannot add any cycle from the clause gadget to* $\mathcal{C}_{\max}$*.*

Proof. If the input signal to the clause gadget is TRUE, the adjacent cycle to the clause gadget is not in $\mathcal{C}_{\max}$. Therefore, a the cycle of the gadget can be added to $\mathcal{C}_{\max}$. When in the loony endgame, this cycle yields four boxes to Trudy after Fred makes a double-dealing move.

Otherwise, if the input signal is FALSE, the adjacent cycle is in $\mathcal{C}_{\max}$, and from the clause gadget only a chain is left. This chain yields only two boxes to Trudy after Fred makes a double-dealing move. ◀

4 Players' strategies and PSPACE-completeness

With the gadgets described above, we construct a Dots & Boxes instance for any G_{pos}(POS CNF) instance such that Trudy can win the Dots & Boxes instance if and only if she can win the corresponding G_{pos}(POS CNF) instance. We lay out the variable gadgets, attach a corresponding number of wire gadgets, pass the wires through or gadgets, using crossover gadgets to cross signals, and finally connect wires to the clause gadgets. An example of our construction is given in Figure 13 in the appendix.

The initial score we set to the Dots & Boxes instance depends on the number of gadgets of each type in the construction. By Lemma 1 the total score in the loony endgame depends on the number of disjoint cycles c, the number of boxes k with degree higher than two, and their total degree T. The configuration of the loony endgame, and thus the values k and T, is changed only when the variable gadgets are being assigned their values. We will argue below that, under regular play, exactly half of the variables are set to TRUE and half are set to FALSE. Thus the total values of k and T are the same, no matter which variables are assigned to which values. If Trudy can satisfy formula $\mathcal{F}$ of the G_{pos}(POS CNF) game, she can claim some $c_{\max} = |\mathcal{C}_{\max}|$ cycles in the corresponding Dots & Boxes instance. Therefore, by Lemma 1, she can claim $4c_{\max} + T - 2k - 4$ boxes in the loony endgame, and $n/2$ boxes from the variables set to FALSE. Let N be the total number of unclaimed boxes in our Dots & Boxes instance. Then, Fred gets $N - n/2 - (4c_{\max} + T - 2k - 4)$ boxes. We set the initial scores of Trudy and Fred such that Trudy's final score is one larger than Fred's if she can satisfy $\mathcal{F}$. Otherwise, her score will be strictly less than Fred's.

Next, we describe the regular strategies for Trudy and Fred, both before the loony endgame is entered and in the loony endgame.

Regular strategies for Trudy and Fred in the loony endgame. We first discuss both strategies in the loony endgame, assuming that all variables have already been assigned a value as described in Section 3.4. As we argue below, Fred can always ensure that he is in control of the loony endgame. It is always beneficial for Fred to stay in control, as then all the chains and cycles in the loony endgame configuration yield at least as many boxes to him as to Trudy.

In the loony endgame, Trudy can choose which chains and cycles to open. To maximize her score, Trudy is going to select a maximum number of disjoint cycles $\mathcal{C}_{\max}$ in the loony endgame (see Lemma 1). This can be done by first making a loony move in all chains, to which Fred responds by claiming all but two boxes, finishing with a double-dealing move in order to stay in control. Afterwards, Trudy makes loony moves in the remaining cycles, to which Fred responds again by claiming all but four boxes, finishing with a double-dealing moves each time, except for in the final cycle.

Regular strategy for Trudy before the loony endgame. Trudy's strategy before the loony endgame is to set enough variable gadgets to TRUE in order to satisfy all the clauses. By Lemmas 1 and 6, Trudy gains more boxes from each satisfied clause. Therefore, the regular strategy for Trudy is to claim the boxes opened by Fred when setting variables to FALSE, and to set variables to TRUE, by using a loony move in a side chain of cycle C_2 of the variables. As we show in Lemma 7, if Fred deviates from setting variables to FALSE, and plays a loony move when there are non-loony moves available, Trudy can adopt Fred's regular strategy and dominate the rest of the game by ensuring that she ends up in control when the loony endgame is entered.

Regular strategy for Fred before the loony endgame. Fred's strategy is to ensure that he is in control when the loony endgame starts, and it can be described completely as responses to what Trudy does. By our assumption the number of variables in $\mathcal{F}$ is even, thus initially the number of non-loony move pairs is even. Fred's strategy is then to keep the number of non-loony move pairs even at the start of every one of Trudy's turns. Then, once the number of non-loony moves reaches zero (and the loony endgame is reached), it is Trudy's turn, and Fred is in control. Specifically, Fred responds to Trudy's moves in the following way:

- If Trudy follows regular play and makes a loony move in a variable to set it to TRUE, then Fred simply claims all boxes in the chain opened by Trudy (without making a double-dealing move), and makes a non-loony move in another variable to set it to FALSE.
- If Trudy deviates from her strategy by making a non-loony move, setting a variable to FALSE, there must be at least one other non-loony move pair available to Fred. Therefore, Fred claims the box opened by Trudy, and makes a non-loony move, thereby setting another variable to FALSE. The number of non-loony pairs is again even at the start of Trudy's next turn.
- If Trudy deviates from her strategy by opening a chain with a loony move that does not remove a non-loony pair, Fred responds with claiming all but two (or four in case of a cycle) boxes and ends with a double-dealing move. The number of non-loony pairs remains even before Trudy's next turn.

Using this strategy, Fred can set a variable to FALSE each time Trudy sets a variable to any value, as well as gain control in the loony endgame.

Note that the order of moves in these strategies is not enforced. Trudy can play loony moves she would play in the loony endgame even if there are still non-loony moves available, as long as these moves do not interfere with the values set (or to be set) in the corresponding variables. For Fred, we consider it part of his regular strategy to simply respond to these moves as if the game was already in the loony endgame, since otherwise he would be in danger of losing control. Indeed, if Fred does not make a double-dealing move, the number of non-loony moves will no longer be even at the start of Trudy's turn, and Fred loses control of the loony endgame. Thus, it is not more beneficial for any player to make a move in any other gadget than the variable gadgets while there are still variables that have not been set.

▶ **Lemma 7.** *Deviating from the regular strategies described above is sub-optimal for Fred and cannot be more beneficial for Trudy.*

Proof. Trivially, Trudy and Fred always claim open boxes before making their move, except when Fred makes double-dealing moves. Otherwise the opponent can claim these boxes next turn.

First, consider the regular strategies in the loony endgame. If Trudy deviates from her strategy and does not select the maximum number of disjoint cycles, by Lemma 1 her score will be too low and she loses the game. Therefore, the regular loony endgame strategy for Trudy as described above is optimal. If, at any point in the loony endgame, except for his last move, Fred does not make a double-dealing move, he loses control. Since being in control is always beneficial in our construction, this play is sub-optimal.

The regular strategies described for before the loony endgame are also optimal. Observe that, under the described strategies, the value-setting component of a variable yields the same number of boxes to Trudy independent of whether it is set to TRUE or to FALSE. Indeed, if it is set to TRUE, the component contains three boxes with degree 3, while setting the variable to TRUE does not give any boxes to Trudy; if the variable is set to FALSE, the component contains two boxes with degree 3, but setting the value gives Trudy one box. Thus, the value-setting component contributes the same number of points to Trudy's final score independent of the value.

If Trudy deviates from her strategy by making a non-loony move and setting a variable to FALSE, she loses one box to Fred. Furthermore, setting a variable to FALSE can never help Trudy to satisfy formula $\mathcal{F}$. Thus, such a move is sub-optimal.

If Trudy deviates from her strategy by making a loony move in any other gadget than the variable gadget, there are two options: either she makes a move that leads to the same score as the strategy described above, or she makes a move that contradicts the setting of the variables and reduces her total score. The former case does not have any bad repercussions for Trudy. Fred will respond with a double-dealing move, otherwise Trudy would take control of the endgame. Thus, we can reorder the sequence of Trudy's moves and assume that she first sets all the variables. However, in the latter case, the move reduces the number of possible disjoint cycles, and thus leads to Trudy's loss in the game. Therefore, deviating from the regular strategy is never more beneficial for Trudy.

If Fred deviates from his strategy before the loony endgame, then Trudy can adopt his strategy and ensure that the number of non-loony move pairs is even at the start of each of Fred's turn. Since, if Fred is not in control of the loony endgame, he loses the game, deviating from his strategy is not optimal. ◀

Combining the lemmas above, we obtain the main theorem.

▶ **Theorem 8.** *Dots & Boxes is PSPACE-complete.*

Proof. A game of Dots & Boxes is finished after a polynomial number of turns. Thus, all possible sequences of moves can be explored using polynomial sized memory. This implies that Dots & Boxes is in PSPACE.

We now show that Dots & Boxes is PSPACE-hard. Given a G_{pos}(POS CNF) formula $\mathcal{F}$, we construct a Dots & Boxes instance δ following the description above. We argue that Trudy can win $\mathcal{F}$ if and only if Trudy can satisfy δ.

If Trudy can satisfy $\mathcal{F}$, then there must be a variable assignment following the G_{pos}(POS CNF) rules such that Trudy can ensure that every clause is connected to at least one variable which has been set to TRUE, regardless of what Fred does. Therefore, there can be at most $n/2$ variables that need to be set to TRUE by Trudy. Hence, Trudy can set the corresponding variable gadgets in δ to TRUE, and if needed set the remaining variables available to her to TRUE in any order. Thus, by Lemmas 2–6, Trudy can propagate the TRUE values down to all the clauses, that is, she can select the maximum number of disjoint cycles from all the gadgets, including all the clause gadgets, leading to the winning score in δ.

In order for Trudy to win δ, the set of disjoint cycles $\mathcal{C}_{\max}$ that she selects must contain a cycle from every clause gadget, and the maximum number of cycles from all the other gadgets. By Lemmas 2–7, this can be done only if the output signals from each gadget conform to their input signals, and thus there must be a set of variable gadgets set to TRUE whose signal is propagated all the way down to all the clause gadgets. In δ Trudy and Fred have to alternate choosing which variable gadgets get set to TRUE and FALSE, respectively. Thus, if Trudy has a winning strategy in δ, no matter how Fred plays, she can always pick a subset of variable gadgets to assign, such that every clause gadget obtains a TRUE signal. This results in a winning strategy for Trudy to win the G_{pos}(POS CNF) game on $\mathcal{F}$.

Thus, Dots & Boxes is PSPACE-complete. ◀

5 Conclusion

We proved that Dots & Boxes is PSPACE-complete, resolving a long-standing open problem. There exist a number of other intriguing open problems related to Dots & Boxes. Does restricting the game to a $k \times n$ grid for a small k make the game easier? How large does k need to be to make the problem PSPACE-hard or even just NP-hard? These are challenging questions, given that even for a $1 \times n$ grid Dots & Boxes is not yet fully understood [12, 19, 25]. Another direction of further research is the complexity of variants of Dots & Boxes, in particular of misère Dots & Boxes [12], of Dots & Boxes under normal play (where the last player to move wins), of Dots & Boxes on other grids, or even of Dots & Boxes with more than two players as it was originally described by Lucas [29]. One variant that our result resolves is *Dots & Polygons*, since the reduction from Dots & Boxes to *Dots & Polygons* that was used to prove NP-hardness [8] now directly also shows PSPACE-hardness.

Our result can be interpreted as proving that *Strings and coins* restricted to grid graphs is PSPACE-complete. What is the complexity of *Strings and coins* on other restricted graph classes, for instance outerplanar graphs (which generalize $1 \times n$ grids)?

This may also be a good moment to revisit other games, which are known to be PSPACE-complete on general graphs, but for which the complexity on grid graphs is open. This, for instance, includes *NoGo*, *Fjords* (on hexagonal grids), *Cats-and-Dogs* and *GraphDistance*, which are known to be PSPACE-complete for planar graphs [9, 10].

References

1 Oswin Aichholzer, David Bremner, Erik D. Demaine, Ferran Hurtado, Evangelos Kranakis, Hannes Krasser, Suneeta Ramaswami, Saurabh Sethia, and Jorge Urrutia. Games on triangulations. *Theoretical Computer Science*, 343(1):42–71, 2005. Game Theory Meets Theoretical Computer Science. doi:10.1016/j.tcs.2005.05.007.

2 Michael H. Albert, Richard J. Nowakowski, and David Wolfe. *Lessons in play: an introduction to combinatorial game theory.* CRC Press, 2019.

3 Joseph Barker and Richard Korf. Solving 4×5 dots-and-boxes. In *Proc. 25th AAAI Conference on Artificial Intelligence*, pages 1756–1757, 2011. doi:10.5555/2900423.2900680.

4 Joseph Barker and Richard Korf. Solving dots-and-boxes. In *Proc. 26th AAAI Conference on Artificial Intelligence*, pages 414–419, 2012. doi:10.5555/2900728.2900788.

5 Elwyn R. Berlekamp. *The Dots and Boxes Game: Sophisticated Child's Play.* A K Peters/CRC Press, 2000.

6 Elwyn R. Berlekamp, John H. Conway, and Richard K. Guy. Chapter 16: Dots-and-boxes. In *Winning Ways for your Mathematical Plays*, volume 3, pages 541–584. A K Peters/CRC Press, 2nd edition, 2003.

7 Elwyn R. Berlekamp and Katherine Scott. Forcing your opponent to stay in control of a loony Dots-and-Boxes endgame. In R. J. Nowakowski, editor, *More Games of No Chance,* Proc. MSRI Workshop on Combinatorial Games, volume 42, pages 317–330. Cambridge University Press, 2002.

8 Kevin Buchin, Mart Hagedoorn, Irina Kostitsyna, Max van Mulken, Jolan Rensen, and Leo van Schooten. Dots & polygons (media exposition). In *Proc. 36th Internat. Sympos. Computational Geometry (SoCG 2020)*, volume 164 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 79:1–79:4, 2020. doi:10.4230/LIPIcs.SoCG.2020.79.

9 Kyle Burke and Robert A. Hearn. PSPACE-complete two-color planar placement games. *International Journal of Game Theory*, 48(2):393–410, 2019. doi:10.1007/s00182-018-0628-8.

10 Kyle Burke, Silvia Heubach, Melissa A. Huggan, and Svenja Huntemann. Keeping your distance is hard. *CoRR*, 2016. arXiv:1605.06801.

11 Kevin Buzzard and Michael Ciere. Playing simple loony dots-and-boxes endgames optimally. *INTEGERS*, 14:2, 2014.

12 Sébastien Collette, Erik D. Demaine, Martin L. Demaine, and Stefan Langerman. Narrow misere Dots-and-Boxes. *Games of No Chance 4*, 63:57, 2015.

13 Erik D. Demaine. Playing games with algorithms: Algorithmic combinatorial game theory. In *International Symposium on Mathematical Foundations of Computer Science*, pages 18–33. Springer, 2001. doi:10.1007/3-540-44683-4_3.

14 Erik D. Demaine, Martin L. Demaine, Nicholas J.A. Harvey, Ryuhei Uehara, Takeaki Uno, and Yushi Uno. UNO is hard, even for a single player. *Theoretical Computer Science*, 521:51–61, 2014. doi:10.1016/j.tcs.2013.11.023.

15 Erik D. Demaine and Yevhenii Diomidov. Strings-and-Coins and Nimstring are PSPACE-complete. *CoRR*, 2021. arXiv:2101.06361.

16 Erik D. Demaine and Robert A. Hearn. Playing games with algorithms: algorithmic combinatorial game theory. In *Games of No Chance 3*, pages 3–56. Cambridge University Press, 2009. arXiv:cs/0106019.

17 David Eppstein. Computational complexity of games and puzzles. Last accessed on 06/05/2021. URL: https://www.ics.uci.edu/~eppstein/cgt/hard.html.

18 Gary William Flake and Eric B. Baum. Rush Hour is PSPACE-complete, or "Why you should generously tip parking lot attendants". *Theoretical Computer Science*, 270(1):895–911, 2002. doi:10.1016/S0304-3975(01)00173-6.

19 Richard K. Guy and Richard J. Nowakowski. Unsolved problems in combinatorial games. In R. J. Nowakowski, editor, *More Games of No Chance,* Proc. MSRI Workshop on Combinatorial Games, volume 42, page 457–473. Cambridge University Press, 2002.

20 Robert A. Hearn. *Games, puzzles, and computation.* PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, USA, 2006.

21 Robert A. Hearn and Erik D. Demaine. *Games, puzzles, and computation.* CRC Press, 2009.

22 Takashi Horiyama, Takashi Iizuka, Masashi Kiyomi, Yoshio Okamoto, Ryuhei Uehara, Takeaki Uno, Yushi Uno, and Yukiko Yamauchi. Sankaku-tori: An old western-japanese game played

on a point set. *Journal of Information Processing*, 25:708–715, 2017. doi:10.2197/ipsjjip.25.708.

23 Ming Yu Hsieh and Shi-Chun Tsai. On the fairness and complexity of generalized k-in-a-row games. *Theoretical Computer Science*, 385(1-3):88–100, 2007. doi:10.1016/j.tcs.2007.05.031.

24 Shigeki Iwata and Takumi Kasai. The Othello game on an $n \times n$ board is PSPACE-complete. *Theoretical Computer Science*, 123(2):329–340, 1994. doi:10.1016/0304-3975(94)90131-7.

25 Adam S. Jobson, Levi Sledd, Susan Calcote White, and D. Jacob Wildstrom. Variations on narrow dots-and-boxes and dots-and-triangles. *INTEGERS*, 17:#G2:1–8, 2017.

26 Will Johnson. The combinatorial game theory of well-tempered scoring games. *International Journal of Game Theory*, 43(2):415–438, 2014. doi:10.1007/s00182-013-0386-6.

27 Anthony Knittel, Terry Bossomaier, and Allan Snyder. Concept accessibility as basis for evolutionary reinforcement learning of dots and boxes. In *IEEE Symposium on Computational Intelligence and Games*, pages 140–145. IEEE, 2007. doi:10.1109/CIG.2007.368090.

28 David Lichtenstein and Michael Sipser. GO is polynomial-space hard. *Journal of the ACM*, 27(2):393–401, 1980. doi:10.1145/322186.322201.

29 Édouard Lucas. *Récréations mathématiques*, volume 2. Gauthier-Villars et fils, 1883.

30 Henry Meyniel and Jean-Pierre Roudneff. The vertex picking game and a variation of the game of dots and boxes. *Discrete Mathematics*, 70:311–313, 1988. doi:10.1016/0012-365X(88)90007-6.

31 Richard J. Nowakowski. ..., Welter's game, Sylver coinage, dots-and-boxes, ... In R. K. Guy, editor, *Combinatorial Games,* Proc. Symposia in Applied Mathematics, volume 43, pages 155–182. AMS, 1991. doi:10.1090/psapm/043/1095544.

32 Thomas J. Schaefer. On the complexity of some two-person perfect-information games. *Journal of Computer and System Sciences*, 16(2):185–225, 1978. doi:10.1016/0022-0000(78)90045-4.

33 Aaron N. Siegel. *Combinatorial game theory*, volume 146 of *Graduate Studies in Mathematics*. American Mathematical Society, 2013. doi:10.1090/gsm/146.

34 Julian West. Championship-level play of dots-and-boxes. In R. J. Nowakowski, editor, *Games of No Chance,* Proc. MSRI Workshop on Combinatorial Games, volume 29, pages 79–84. Cambridge University Press, 1996.

35 Yimeng Zhuang, Shuqin Li, Tom Vincent Peters, and Chenguang Zhang. Improving monte-carlo tree search for dots-and-boxes with a novel board representation and artificial neural networks. In *IEEE Conference on Computational Intelligence and Games*, pages 314–321. IEEE, 2015. doi:10.1109/CIG.2015.7317912.

A Example game

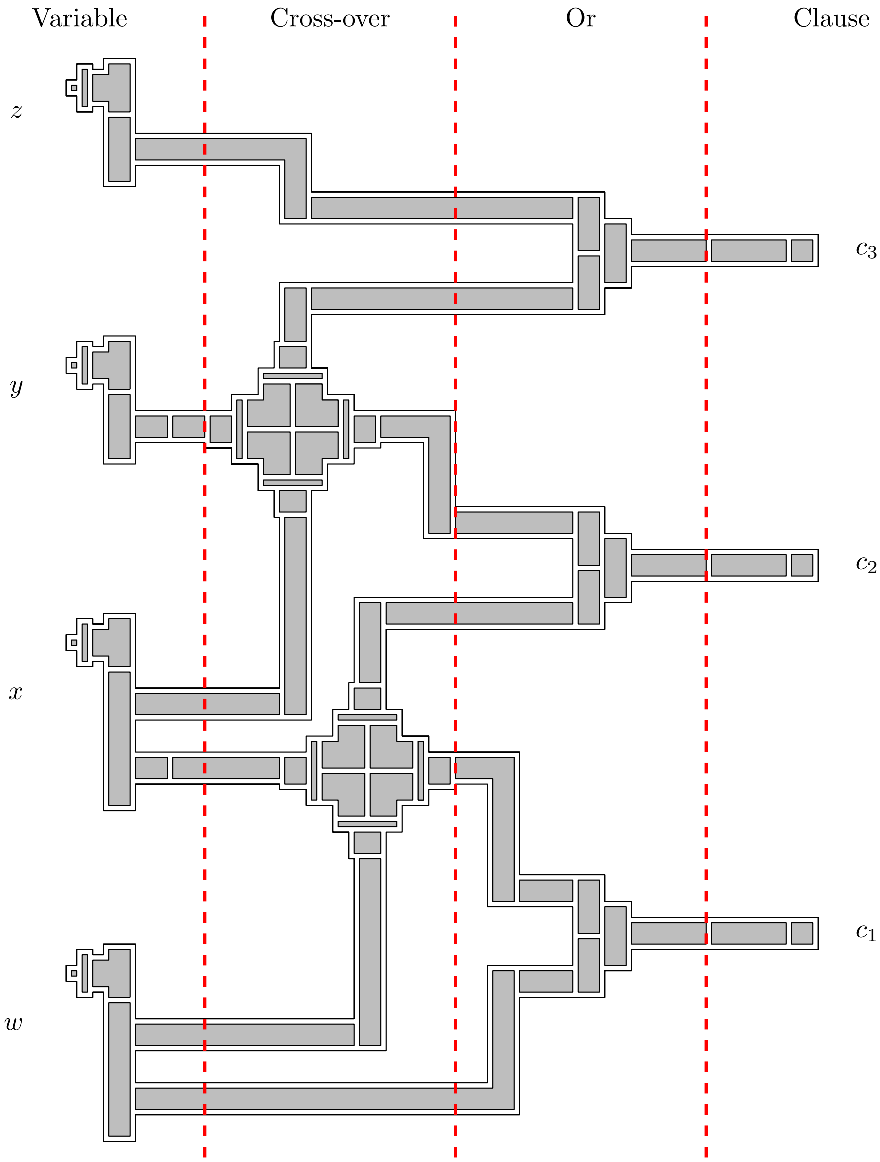


Figure 13 Example reduction from the G_{pos}(POS CNF) formula $(w \vee x) \wedge (w \vee y) \wedge (x \vee z)$. The construction can be divided into four sections: a variable, crossover, or, and clause section. Each section contains only the corresponding gadgets and wire gadgets that connect different gadgets together.

B Omitted proofs

▶ **Lemma 1.** *Let the configuration of a loony endgame contain k boxes with degree higher than 2, let T be the sum of the degrees of these boxes, and let c be the maximum number of disjoint cycles in the configuration. Then, the player who is* not *in control can claim at most $4c + T - 2k - 4$ boxes.*

Proof. Let Fred be in control of the game. To simplify the argument, w.l.o.g., we assume that the last move made by Trudy is made in a cycle. Let c denote the number loony moves made by Trudy in a disjoint cycle and let ℓ be the number of loony moves made by Trudy in chains. All but the last loony move in a disjoint cycle or chain yield 4 or 2 boxes for Trudy, respectively. Thus, the score gained by Trudy in the loony endgame is

$$4c + 2\ell - 4.$$

Consider the dual graph $G = (V, E)$ to the Dots & Boxes instance. In it, a node corresponds to a box, and an edge connects two nodes if the two corresponding adjacent boxes do not have a line drawn between them. Suppose G has k nodes with degree higher than 2. We define T to be the sum of the degrees of these nodes:

$$T = \sum_{\{v \in V \mid degree(v) > 2\}} degree(v).$$

A loony move on a disjoint cycle does not change T, since all disjoint cycles only contain boxes of degree 2. A loony move on a chain, however, decreases the degree of the box at both ends of the chain by 1. Furthermore, whenever the degree of a box reduces from 3 to 2 the degree of this box is no longer counted in T. Thus

$$T = 2\ell + 2k,$$

which means the score for Trudy will be

$$4c + T - 2k - 4.$$

Since T and k are fixed, the score is maximized when the number of loony moves in disjoint cycles is maximized. ◀