MORPION SOLITAIRE

Erik D. Demaine
MIT Computer Science and Artificial Intelligence Laboratory
edemaine@mit.edu

Martin L. Demaine
MIT Computer Science and Artificial Intelligence Laboratory
mdemaine@mit.edu

Arthur Langerman
Langerman Diamonds
arthur@langerman.net

Stefan Langerman[1]
Université Libre de Bruxelles, Département d'informatique
stefan.langerman@ulb.ac.be

Abstract

We study a popular pencil-and-paper game called *morpion solitaire*. We present upper and lower bounds for the maximum score attainable for many versions of the game. We also show that, in its most general form, the game is NP-hard and the high score is inapproximable within $n^{1-\varepsilon}$ for any $\varepsilon > 0$ unless $P = NP$.

1. *Introduction*

The classic game of *morpion solitaire* starts with some configuration of points drawn on the intersections of a square grid, typically the cross shown on the left of Figure 1. In this game, the player makes a sequence of moves. Each move consists of placing a new point at a grid intersection and drawing a new line segment connecting 5 consecutive points that include the new one. The line can be drawn in any of the four directions: horizontal, vertical, or either diagonal. Moves are further constrained by one of two constraints. In the *disjoint model*, line segments with the same direction

[1] Chargé de recherches du FNRS.

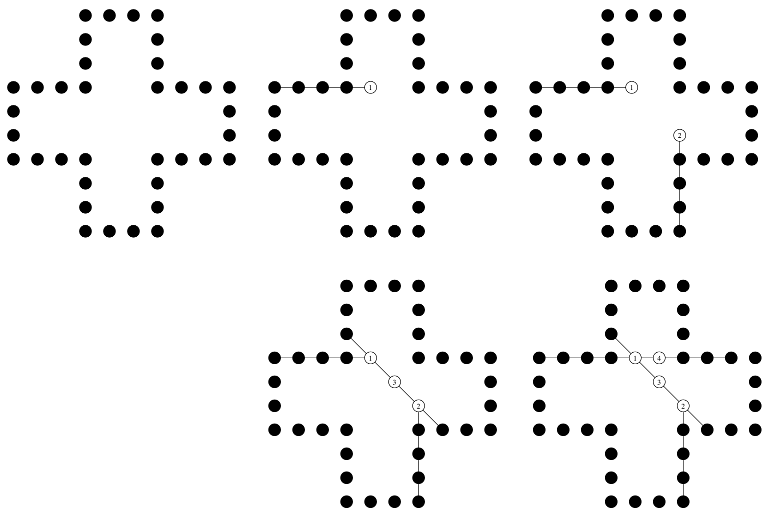

Figure 1: The cross starting configuration for $k = 4$, and four sample moves. The last move is permitted only in the touching model.

cannot share a point. However, line segments with different directions are always permitted to share points. In the *touching model*, line segments with the same direction are permitted to overlap just slightly, at a common endpoint, but cannot share more than one point. In other words, the touching model allows point overlap but disallows positive-length overlap of the line segments. The game is over when no further moves can be made. The goal of the game is to maximize the number of moves before the game ends.

The morpion solitaire game is famous in several European countries (mainly in Belgium and France), where every elementary school student is required to have graph paper in his schoolbag. The game is also commonly called “connector”, “petites croix” (“little crosses”), or “Malta cross”. The touching model is probably the most popular of the two models. The first published reference we could find about the game is in the magazine Jeux & Stratégie from September 1982 [3]. The article shows a solution of 164 moves and claim a record of 170 moves by Charles-Henri Bruneau without actually displaying it. The following two issues of the magazine mention that they have received a large number of proposed solutions, but those solutions have not been published either. Since then, several webpages have been dedicated to finding better solutions to the touching version of the game [1, 6, 7], and games of 170 moves due to Denis Excoffier, Charles-Henri Bruneau, and JB Bonté (bearing the date of 15 January 1982) have been published and verified. The game has also been used as a test case for an evolutionary algorithm by Hugues Juillé [5]. His program found a game of 122 moves.

The disjoint model is the one that appears under the name Connector in the excellent book by Walter Joris [4]. The book describes a two-player variant as well. A webpage maintaining the high scores for the disjoint-model solitaire game has been maintained by the fourth author since 1996 [6]. High scores have alternated between Stefan Schmieta, who used an implementation of a random-sampling algorithm with local search, and the third author, who used exclusively pencil and paper. The current record of 68 moves is held by the third author.

In this article, we consider combinatorial and computational issues for several variations on both the touching and disjoint variants of morpion solitaire. We first generalize the game so that, at every move, the drawn line segment joins $k+1$ points, rather than 5, for some specified value k, and scaling the initial cross configuration accordingly. We also consider the more general case where the starting configuration can be any given set of points. We present lower and upper bounds for the largest number of moves in all versions of the game, in particular partially characterizing when the number of moves can be infinite.

After the magazine Jeux & Stratégie received a large number of solutions, they were faced with the computational problem of verifying them. In [2] they write "It is horribly difficult, or even impossible to figure the order in which the line segments have been drawn, and thus to verify if the proposed game is valid. Indeed, after the 30th move, or even before, the addition of a new point allows 2, 3, or 4 alignment possibilities: which to choose? The number of possibilities grows as the game continues. Soon enough, it is a dead-end" (translated from French). In Section 4, we show that reconstructing a valid ordering from a drawing is not as difficult as it seems: we give a linear-time algorithm for this task. We then show that, on the other hand, determining the maximum number of moves that can be made from a given set of points is NP-hard and not approximable within $n^{1-\varepsilon}$ for any $\varepsilon > 0$ (unless $P = NP$).

2. *Notation*

Let $G_k(S)$ denote the maximum number of moves in a game starting with an initial set $S \subseteq \mathbb{Z}^2$ of points on the unit lattice, where at each step a line joining $k+1$ points is drawn through k existing points and a new one, and where two lines with the same direction cannot share a point (disjoint model). Let $G'_k(S)$ be the maximum number of moves in the variant where two lines are allowed to share one point but not two (touching model). Let A_k be the traditional initial set of points formed by a plus sign of thickness k. For example A_4 is the configuration shown on the left of Figure 1, and $G_4(A_4)$ is number of lines in the best possible solution of the original puzzle.

3. *Combinatorial Results*

In this section, we present upper and lower bounds on the values of $G_k(S)$, $G'_k(S)$, $G_k(A_k)$ and $G'_k(A_k)$. Because the touching model is less restrictive than the disjoint model, $G_k(S) \leq G'_k(S)$ for any S and any k.

3.1. *Potential Function*

Every point drawn on the board can be seen as having 8 *slots* to which line segments connect, coming from 8 directions. We define the potential of a point to be the number of directions in which it is not connected, i.e. the number of empty slots. The potential $\phi(D)$ of a drawing D is then the sum of the potential of all its points. Considering $G_k(S)$, the potential at the beginning is $8|S|$. Each move adds a new point, which adds 8 to the potential, and a line which removes $2(k+1)$ directions. No further moves can be made when the potential is less than $2k$. This implies that $8|S| - (2(k+1) - 8)(G_k(S) - 1) \geq 2k$, and so when $k > 3$,

$$G_k(S) \leq 1 + (4|S| - k)/(k-3).$$

For $G'_k(S)$, the situation is identical except that adding a line only removes $2k$ from the potential, and no further line can be added when the potential is less than $2k-1$. So, for $k > 4$,

$$G'_k(S) \leq 1 + (8|S| - 2k + 1)/2(k-4).$$

For example, this gives $G_4(A_4) \leq 141$. Unfortunately, this simple argument does not produce a bound for $G'_4(A_4)$. In fact, in that case, a move keeps the potential unchanged.

3.2. *Boundary Bound for* $G'_k(S)$

Let D be a drawing at some point in the game $G'_k(S)$, and let P be the set of points in D. Form a new drawing $\Gamma(P)$ by connecting every adjacent pair of points of P, horizontally, vertically or diagonally. $\Delta(P)$ forms a superset of the drawing D. This implies that $\phi(\Gamma(P)) \leq \phi(D)$. By extension, we define the potential of any set of points Q: $\phi(Q) = \phi(\Gamma(Q))$. We can assume $\Gamma(P)$ is connected. This assumption can be later removed by considering each connected component separately. Let $CH(P)$ be the convex hull of P, and let $\widehat{CH}(P)$ the set of all grid points contained in the convex hull of P. We first observe that

LEMMA 1 $\phi(P) \geq \phi(\widehat{CH}(P))$.

Proof: Consider an edge (u, v) of $CH(P)$, where u is before v in clockwise order. We assume no other point of P is incident to the edge (u, v), otherwise we split the edge. We define $\phi(u, v)$ to be the number of edges of the complete grid $\Gamma(\mathbb{Z}^2)$ which have an endpoint to the left of (u, v) and which intersect the interior of edge (u, v). Each of these edges connect to an empty slot of $\widehat{CH}(P)$, and so $\phi(P)$ is less or equal to the sum of $\phi(u, v)$ for all u and v consecutive on the convex hull of P. Since $\Gamma(P)$ is connected, we can walk between the two endpoints of e, walking along the edges of $\Delta(P)$. Furthermore, a path can be found using the right hand rule: starting at u, take the first edge (u, u') clockwise from (u, v), then the next edge (u', u'') clockwise from $(u.u')$, and so on until v is reached. The set of empty slots to the left of this path for all points encountered along the path is at least as large as $\phi(u, v)$. This proves the lemma because each empty slot can be counted by at most one edge of $CH(P)$. □

On the other hand, we have $|P| \le |\widehat{CH}(P)|$. Using Pick's Theorem [8], we can compute the area A of $CH(P)$: $A = \text{Area}(CH(P)) = |\widehat{CH}(P)| - B(CH(P))/2 - 1$, where $B(CH(P))$ is the number of points of $\widehat{CH}(P)$ on the boundary of $CH(P)$. Because $CH(P)$ is convex, we know that each of those points contribute at least 3 empty slots to the potential function, and so $B(CH(P)) \le \phi(\widehat{CH}(P))/3$. And $A \ge |\widehat{CH}(P)| - \phi(\widehat{CH}(P))/6 - 1$. The area A of $CH(P)$ can be bounded by a function of its perimeter L: $A \le L^2/4\pi$.

In turn, we can bound the perimeter L by a function of the potential. Consider an edge (u, v) of $CH(P)$ (in clockwise orientation), and let $\Delta x = v_x - u_x$, $\Delta y = v_y - u_y$, and suppose wlog that $\Delta x \ge 0$ and $\Delta y \ge 0$. There are $\Delta x - 1$ vertical lines, and $\Delta y - 1$ horizontal that contribute to $\phi(u, v)$. The number of lines of slope -1 contributing to $\phi(u, v)$ is $\Delta x + \Delta y - 1$, and the number of lines of slope 1 contributing to $\phi(u, v)$ is $\max(0, |\Delta x - \Delta y| - 1)$. Thus, assuming $\Delta x \ge \Delta y$, we have $\phi(u, v) \ge 3\Delta x + \Delta y - 4$. And if the slope of the edge (u, v) is c, $\Delta y = c\Delta x$, $\phi(u, v) \ge (3 + c)\Delta x - 4$. The length $L(u, v)$ of the edge (u, v) is $\sqrt{1 + c^2}\Delta x$ and so $\phi(u, v) \ge ((3 + c)/\sqrt{1 + c^2})L(u, v)$. Minimizing $(3 + c)/\sqrt{1 + c^2}$ over $c \in [0, 1]$, we obtain $\phi(u, v) \ge 2\sqrt{2}L(u, v)$. Summing over all edges, we have $L \le 2\sqrt{2}\phi(\widehat{CH}(P))$.

Putting it all together, we obtain $|\widehat{CH}(P)| - \phi(\widehat{CH}(P))/6 - 1 \le A \le L^2/4\pi \le \phi(\widehat{CH}(P))^2/(32\pi)$, $|\widehat{CH}(P)| \le \phi(\widehat{CH}(P))^2/(32\pi) + \phi(\widehat{CH}(P))/6 + 1$, which implies $|P| \le \phi(P)^2/(32\pi) + \phi(P)/6 + 1$. If n is the number of moves performed, we have $|P| = |S| + n$, and in the game $G'_k(S)$, $\phi(P) \le \phi(D) = 8|S| + n(8 - 2k)$. So, $|S| + n \le (8|S| + n(8 - 2k))^2/(32\pi) + (8|S| + n(8 - 2k))/6 + 1$. In particular, when $k = 4$, the maximum number of moves is $n \le |S|^2/(4\pi) + |S|/3 + 1$, which implies that the original $G'_4(A_k) \le 838$. This upper bound applies to any starting set of 36 points.

3.3. $k = 1$

Starting with one point, the game $G_1(P)$ can continue indefinitely, as shown in Figure 2. Thus, for any S with $|S| > 0$, $G_1(S) = G'_k(S) = \infty$.

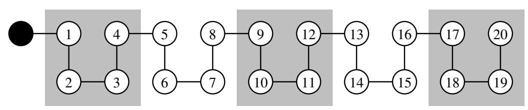

Figure 2: $G_1(S) = G'_k(S) = \infty$.

3.4. $k = 2$

A set of two points allow no more than one move: $G_2(2) = G'_2(2) = 1$. But there exists a starting set of three points with which one can play indefinitely (see Figure 3). So, $G_2(3) = G'_2(3) = \infty$ and in particular, $G_2(A_2) = G'_2(A_2) = \infty$.

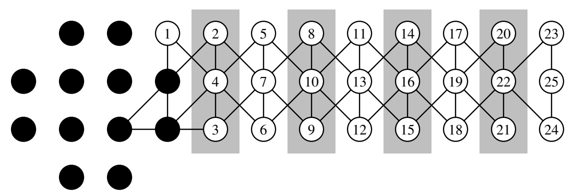

Figure 3: $G_2(A_2) = G'_2(A_2) = G_2(3) = G'_2(3) = \infty$.

3.5. $k = 3$

The case $k = 3$ is really the first interesting one. The potential argument does not help since for $G_3(S)$, the potential remains unchanged after a move. Moreover, there exists a starting set of 7 points with which one can play indefinitely (see Figure 4). So: $G_3(7) = G'_3(7) = \infty$.

Nevertheless, we can show that both $G_3(A_3)$ and $G_3(A_3)$ are bounded. Assume the bottom leftmost point of A_3 has coordinates $(1, 1)$. Notice that every point of A_3 has at least one of its coordinates odd. This implies that no point with both coordinates even can ever be played during the game. To see this, just notice that any segment of length 4 incident to one even point has to be incident to exactly two event points. This

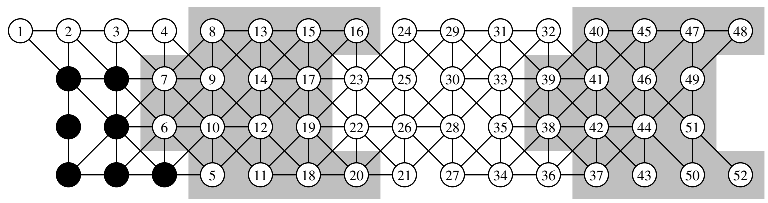

Figure 4: $G_2(A_2) = G'_2(A_2) = G_2(3) = G'_2(3) = \infty$.

reduces the number of slots available at every point: an (odd,odd) point has 4 slots and an (even,odd) point has 6 slots available. So the starting potential is 120.

We will split the potential into 3: ϕ_{oo} is the sum of all horizontal and vertical free slots at (odd,odd) points, ϕ_{oe} is the sum of all horizontal and vertical free slots at (odd,even) or (even,odd) points, and ϕ_d is the sum of all diagonal free slots (those never appear at (odd,odd) points). Let $m_{H,oo}$ be the number of moves placing a (odd,odd) point and drawing a horizontal or vertical line, $m_{H,oe}$ the number of moves placing a (odd,even) or (even,odd) point and drawing a horizontal or vertical line, and m_d the number of diagonal moves (those only place (even,odd) points). The potentials can be expressed by the following equations:

$$\begin{aligned}\phi_{H,oo} &= 48 - 4m_{H,oe}\\ \phi_{H,oe} &= 24 - 2m_{H,oe} - 4m_{H,oo} + 2m_d\\ \phi_{H,oe} &= 48 + 4m_{H,oe} - 4m_d\end{aligned}$$

Solving the linear program of maximizing $m_{H,oo} + m_{H,oe} + m_d$ subject to non-negativity constraints $\phi_{H,oo}, \phi_{H,oe}, \phi_d, m_{H,oo}, m_{H,oe}, m_d \geq 0$, we obtain $m_{H,oo} = 12, m_{H,oe} = 12, m_d = 24$, which imply that $G_3(A_3) \leq 48$. Figure 5 shows that $G_3(A_3) \geq 31$.

For the second variant $G'_3(A_3)$, the potentials are defined as follows:

$$\begin{aligned}\phi_{H,oo} &= 48 + m_{H,00} - 3m_{H,oe}\\ \phi_{H,oe} &= 24 - m_{H,oe} - 3m_{H,oo} + 2m_d\\ \phi_{H,oe} &= 48 + 4m_{H,oe} - 2m_d\end{aligned}$$

Solving the linear program of maximizing $m_{H,oo} + m_{H,oe} + m_d$ subject to non-negativity constraints $\phi_{H,oo}, \phi_{H,oe}, \phi_d, m_{H,oo}, m_{H,oe}, m_d \geq 0$, we obtain $m_{H,oo} = 60, m_{H,oe} = 96, m_d = 36$, which imply that $G'_3(A_3) \leq 192$. Figure 6 shows that $G'_3(A_3) \geq 56$.

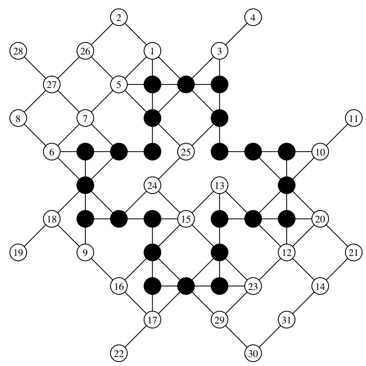

Figure 5: $G_3(A_3) \geq 31$.

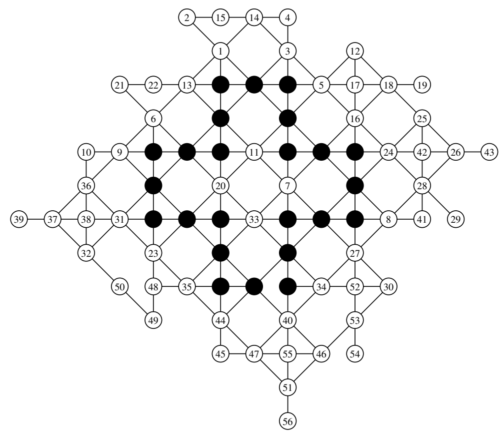

Figure 6: $G'_3(A_3) \geq 56$.

3.6. $k = 4$

The case $k = 4$ is the original game. The potential function argument from Section 3.1 shows that $G_4(n) \leq 4n - 3$, in particular, $G_4(A_4) \leq 141$. Figure 7 shows that $G_4(A_4) \geq 68$.

Several lower bounds on $G'_4(A_4)$ are described in the introduction; the longest known sequence of moves is 170. According to [7], Achim Flammenkamp has obtained an (unpublished) upper bound of 324 on $G'_4(A_4)$.

3.7. $k \geq 5$

Figure 8 shows all the new points that can be generated in the case $k \geq 5$. The addition of those points does not form any line of sufficient density to perform another move. Furthermore, only 12 of those 24 points can appear simultaneously in a game. This shows that $G_k(A_k) = G'_k(A_k) = 12$ for $k \geq 5$.

4. *Algorithmic Results*

4.1. *Verifying a Drawing*

In this section, we present algorithms for verifying a drawing without an ordering on the added points. We use a simple greedy algorithm: at every step, find a line in the drawing that covers k existing points and one point not yet played. Play that point

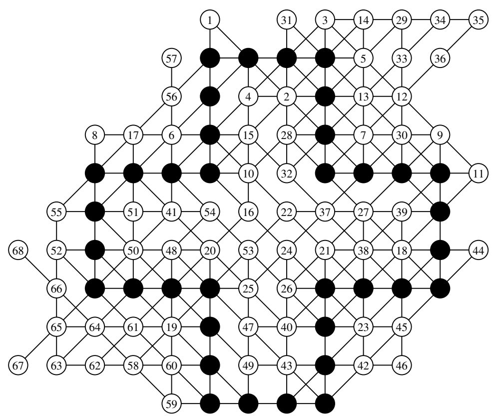

Figure 7: $G_4(A_4) \geq 68$.

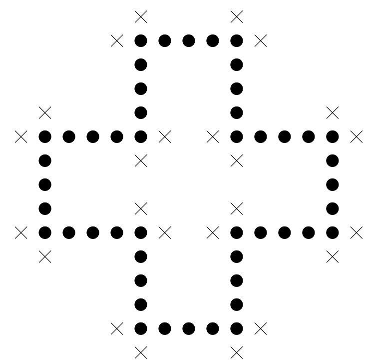

Figure 8: $k \geq 5$.

and line, repeat. If all lines have been played, report a success and the ordering of the lines. Otherwise, if no playable line exists, report a failure.

LEMMA 2 *The drawing is valid if and only if the greedy algorithm succeeds.*

Proof: Because the greedy algorithm obeys the rules of the game, if it is successful, then the drawing is valid. So we just have to show that if the drawing is valid, then the greedy algorithm will succeed. So suppose there is a valid ordering $\ell_1, \ell_2, \ldots, \ell_n$ for drawing the lines, that the greedy algorithm has already drawn $\ell_1, \ldots, \ell_i$, and let ℓ_j be a drawable line chosen by the greedy algorithm. We just have to show that $\ell_1, \ldots, \ell_i, \ell_j, \ell_{i+1}, \ldots, \ell_{j-1}, \ell_{j+1}, \ldots, \ell_n$ is also a valid ordering. If this is not the case, it would mean that some line $\ell_{j'}$, $i+1 \leq j' \leq j-1$ cannot be drawn because of the presence of ℓ_j. It cannot be because ℓ_j collides with $\ell_{j'}$ (i.e., they share a point in the same direction in the disjoint model, or they share two points in the touching model), because the collision is independent of the order in which the lines are drawn. he only other possible reason is that when drawing ℓ_j, we drew the point that was supposed to be drawn for $\ell_{j'}$. But then drawing $\ell_{j'}$ before ℓ_j would produce the same problem, contradicting that $\ell_1, \ell_2, \ldots, \ell_n$ is a valid ordering. □

THEOREM 1 *Given S and a set of n lines, it is possible to verify whether those n lines are a solution for either $G_k(S)$ or $G'_k(S)$ in $O(n + |S|)$ time, and if so, report an order in which the lines can be drawn.*

Proof: By the previous lemma, all we need to do is to be able to find drawable lines quickly. For this task, we preprocess the drawing, pointing each drawn point to the

≤ 8 lines it is covered by, and maintaining a counter for each line, initially zero. Whenever we draw a point, we increment the counter of all the lines pointed to by the point. If a counter reaches k, we put it in a *drawable* queue. First, the algorithm draws the points from the starting configuration. Then, at every step, a line is taken from the drawable queue, and drawn. The running time for the preprocessing is $O(|S| + n)$, and every of the n steps takes $O(1)$ time. □

4.2. *General Dot Patterns are Hard*

In this section we prove that maximizing the number of lines played starting from a general dot pattern is hard, even to approximate:

THEOREM 2 *For any $k \geq 3$, it is NP-hard to find the longest play from a pattern of n dots, or even to find a play of length within $n^{1-\varepsilon}$ of the longest play, for any constant $\varepsilon > 0$. This result holds for both variants of the game.*

To prove this theorem we reduce from 3-SAT. The reduction is identical for both variants of the game. The construction is only slightly different for different k; in the figures, we focus on $k = 3$.

Our construction represents boolean values by whether certain dots can be placed to make certain lines. The wire gadget in Figure 9 propagates this information across the construction. Specifically, one unit of a wire consists of $k - 1$ dots in a row (diagonally). If a dot is placed on one side of these $k - 1$ dots, then we can draw a line and create a dot on the other side of the $k - 1$ dots. By arranging several of these $k-1$ repeats to share the blank spaces on their ends, we obtain a wire that propagates a single dot placement at one end to a dot placement at the other end. To allow for the disjoint model, we do not allow two $k - 1$ repeats in a row to be collinear, but this restriction does not cause any difficulties routing.

To start the wires with values corresponding to variables, we use the variable gadget shown in Figure 10. This gadget simply consists of k dots in a row instead of $k - 1$. Thus the wire on either end can be started, but both wires cannot be started from this gadget because of the nonoverlapping constraint. Thus, one wire represents the variable being true and the other wire represents the variable being false.

To route the value of a variable to multiple clauses, we need the split gadget shown in Figure 11. This gadget consists of joining three wire gadgets together. However, to avoid multiple wires joining collinearly, we need to use some horizontal wires. Any of the wires can be the effective "input" that triggers the other two "outputs".

Before we can define the clause gadget, we need a one-way gadget that prevents information flow in one direction. Figure 13 shows such a gadget. The basic idea is to split the input wire into two so that two X's can be created in close proximity, enabling us to trigger the output wire. On the other hand, the output wire itself creates only one X, but the relevant row is lacking two X's before a line can be drawn. Thus the input wire cannot be triggered from this gadget even if the output wire is triggered.

The clause gadget is essentially three one-way input wires brought together, together with an output wire, as shown in Figure 14. Thus whenever any of the input wires is triggered, the output wire can be triggered, but the triggered input wire does not contaminate the other input wires.

We connect all of the output wires of clause gadgets to a final checker gadget, shown in Figure 15, which offers a large reward for setting all clause output wires correctly. The checker gadget is self-triggered by k dots in a row, but the trigger can continue at each stage only if another wire has triggered it. Thus the output wire in the lower-right can be triggered only if all clauses have been satisfied.

The output wire is connected to "treasure" which is a wire of length $n^{1/\varepsilon+O(1)}$. The reward of this treasure is so large compared to the $n^{O(1)}$ possible lines obtained elsewhere in the construction that even approximate solution to the instance requires solving the 3-SAT instance to gain the treasure.

Two technical issues not yet addressed in this construction are crossings and parity. Crossings in the wiring map can be handled with the crossover gadget in Figure 16. Parity issues arise when trying to connect gadgets whose sizes do not evenly divide each other. These issues can be resolved using the shift gadget in Figure 12, which moves a wire one step (modulo 3) in any desired direction. By repeating $O(1)$ shift gadgets, wires can be aligned horizontally or vertically to match any target gadget.

Acknowledgments. We thank Walter Joris for introducing us to some of the previous work on morpion solitaire; he also independently discovered an infinite play for the $k = 3$ touching variant. We thank Stefan Schmieta for helpful discussions and his script for drawing game executions, on which our figures are based. We thank KHBO Spellenarchief and Jean-Dominique Quinet for providing copies of the Jeux & Stratégies articles. We thank Barry Cipra for helpful comments on the paper.

References

[1] Enigma of malta cross game. http://croix2malte.free.fr/indexGB.php.

[2] Post-scriptum au no. 17. *Jeux & Stratégie*, 18:101, Dec. 1982 and Jan. 1983.

[3] Michel Brassine. Le morpion solitaire. *Jeux & Stratégie*, 16:28–29, Aug.–Sept. 1982.

[4] Walter Joris. *All Squared Up: Tic-Tac-Toe for Geniuses (100 Strategic Games for Pen and Paper)*. Carlton Books, 2002.

[5] Hugues Juille. Incremental co-evolution of organisms: a new approach for optimization and discovery of strategies. In *Proc. 3rd Europ. Conf. Artificial Life*, LNCS 929, June 1995.

[6] Stefan Langerman. A game. http://slef.org/jeu/.

[7] Jean-Charles Meyrignac. Morpion solitaire. http://euler.free.fr/morpion.htm.

[8] G. Pick. Geometrisches zur zahlentheorie. In *Sitzungsberichte des Vereines 'Lotos'*, vol. 19, pp. 311–319, Prague, 1899.

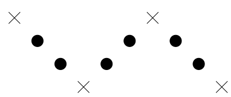

Figure 9: Wire gadget. The X in the upper-left can be covered if and only if the X in the lower-right can be covered.

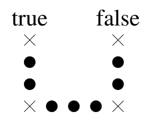

Figure 10: Variable gadget. The left or right wire can be triggered by this gadget, but not both.

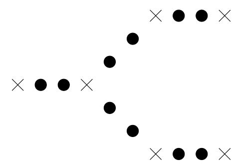

Figure 11: Split gadget. The X on the left can be covered if and only if either or both of the X's on the right can be covered.

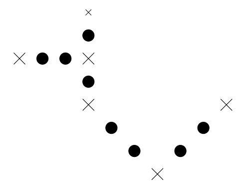

Figure 12: Shift gadget, shown here shifting a horizontal wire down by two steps.

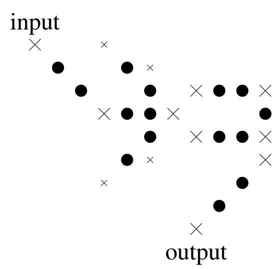

Figure 13: One-way gadget. If the X in the upper-left is covered, then the X in the lower-right can be covered, but there is no such implication in the reverse direction. Small X positions can be triggered but are irrelevant.

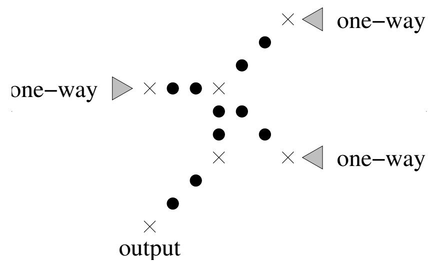

Figure 14: Clause gadget, which uses three one-way gadgets. The output wire on the bottom can be triggered if any of the three input wires can be triggered, but no other implications hold.

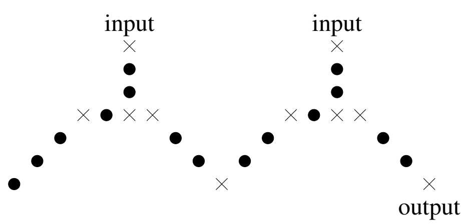

Figure 15: Checker gadget. The output wire can be triggered only if all input wires have been triggered.

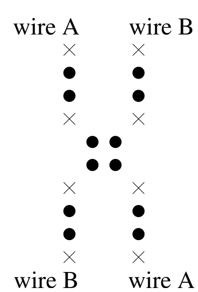

Figure 16: Crossover gadget. Wires A and B act as if they did not cross.