Available at
www.ElsevierComputerScience.com
POWERED BY SCIENCE DIRECT
Theoretical Computer Science 313 (2004) 473–484
Theoretical Computer Science
www.elsevier.com/locate/tcs

# It is tough to be a plumber

Daniel Král'[a,*,1], Vladan Majerech[b,2], Jiří Sgall[a,c,1,3], Tomáš Tichý[c], Gerhard Woeginger[d]

[a] *Department of Applied Mathematics, Institute for Theoretical Computer Science (ITI), Charles University, Malostranské náměstí 25, Prague 118 00, Czech Republic*

[b] *Department of Theoretical Computer Science and Mathematical Logic (KTIML), Charles University, Malostranské náměstí 25, Prague 118 00, Czech Republic*

[c] *Mathematical Institute, AS CR, Žitná 25, CZ-11567 Praha 1, Czech Republic and Institute for Computer Science, Prague, Czech Republic*

[d] *Department of Mathematics, University of Twente, P.O. Box 217, AE 7500 Enschede, The Netherlands*

Received 30 April 2002; received in revised form 16 September 2002; accepted 1 December 2002

**Abstract**

In the Linux computer game `KPlumber`, the objective is to rotate tiles in a raster of squares so as to complete a system of pipes. We give a complexity classification for the original game and various special cases of it that arise from restricting the set of six possible tiles.

Most of the cases are NP-complete. One polynomially solvable case is settled by formulating it as a perfect matching problem; other polynomial cases are settled by simple sweepline techniques. Moreover, we show that all the unsettled cases are polynomial time equivalent.


*Keywords:* Combinatorial game theory; NP-completeness

* Corresponding author.

*E-mail addresses:* kral@kam.mff.cuni.cz (D. Král'), maj@ktiml.mff.cuni.cz (V. Majerech), sgall@math.cas.cz (J. Sgall), tichy@math.cas.cz (T. Tichý), g.j.woeginger@math.utwente.nl (G. Woeginger).

*URL:* http://www.math.cas.cz/∼sgall

[1] Institute for Theoretical Computer Science is supported as project LN00A056 by Ministry of Education of the Czech Republic.

[2] The author acknowledges support by GA CR Grant No. 201/98/1451.

[3] The authors acknowledge support by Grant 201/01/1195 of GA ČR, and cooperative research Grant KONTAKT-ME476/CCR-9988360-001 from the NSF and MŠMT ČR.

0304-3975/$ - see front matter 
doi:10.1016/j.tcs.2002.12.002

## 1. Introduction

The computer game `KPlumber` is included in standard packages of some of Linux distributions. `KPlumber` is a game for a single player. It is played on a rectangular (chessboard-like) board consisting of a number of rows and a number of columns that structure the board into small *squares* or *cells*. Every cell has (up to) four *adjacent* cells to the north, south, east, and west of it. In the initial configuration, every such cell contains one of the six tiles depicted in Fig. 1. Each of these six tiles contains several pipes that may cross each other, go around the corner, connect one side of the tile to the opposite side, and so on. If there is a pipe running from the center of a tile to the middle of one of its sides, then this side is called *open*. Otherwise, the side is called *closed*. If the pipes are filled with water, then the system will possibly leak at an open side of some tile. The only way of preventing this is to have another tile with an open side in the adjacent cell, so that the water can flow on into the open pipe in the adjacent cell. This motivates the following definition: Two tiles in adjacent cells form a *safe pair* if they either touch each other in open sides or touch each other in closed sides.

In the initial configuration, all the tiles are rotated arbitrarily. If the player clicks on one of the tiles, this tile makes a counter-clockwise rotation by 90°. Four clicks on the same tile bring it back into its initial state. The goal of the game is to bring the pipe system into a *safe* state where all the pairs of tiles in adjacent cells form safe pairs. The pipe system is allowed to consist of several connected components in a safe system. An instance of the plumber problem consists of a rectangular board and of rotated tiles in the cells. The question is to decide whether the system can be brought into a safe state.

Since there is no global connectivity condition, the plumber problem can be formulated as a simple constraint satisfaction problem (CSP): For every cell $C$, there is a corresponding variable $v(C)$ in the CSP instance. This variable can take four values that correspond to the four possible rotated states of the corresponding tile (Fig. 1). For every pair of adjacent cells $C_1$ and $C_2$, there is a constraint that forbids that $v(C_1)$ and $v(C_2)$ take values such that an open side touches a closed side. Every constraint

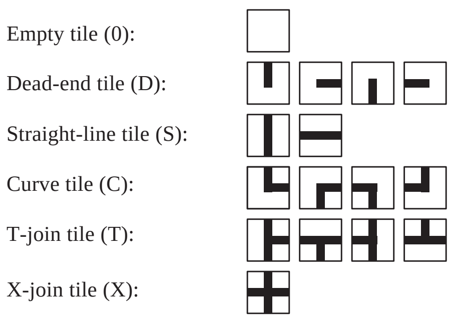


Fig. 1. Different types of tiles and their possible rotations.

involves only two variables, and every variable can only take four distinct values. Hence, the game is a special case of (4, 2)-CSP which is known in general to be NP-complete. Eppstein [1] gives fast (but exponential time) exact algorithms for (4, 2)-CSP.

The computational complexity of the game `KPlumber` is addressed. The reader is referred to [2] for introduction to computational complexity if necessary. The decision problem whether the game can be won is NP-complete (Theorem 1). On the other hand, if the straight line tiles are not permitted, the problem becomes polynomial-time solvable (Theorem 2). The other "polynomial" sets of tiles are identified in Theorems 3 and 4 and some NP-complete sets of tiles in Theorem 8. Finally, we prove in Theorem 9 that the remaining sets of tiles, i.e., those which are not shown to give either an NP-complete version or a polynomial time solvable one, have the same complexity. As pointed out by the referee, the hardness of the full version of the game was proved in [9], however, we include our proof for the sake of completeness. Another reason for including this proof is that our proof seems to be simpler than that of [9].

## 2. Notation

The following tiling problem based on the game `KPlumber` is studied in this paper: An instance of the problem is an $x \times y$ grid where each position in the grid has assigned one of six possible types of tiles (cf. Fig. 1): an *empty tile* (`O`-tile), a *dead-end tile* (`D`-tile), a *straight-line tile* (`S`-tile), a *curve tile* (`C`-tile), a *T-join tile* (`T`-tile) and an *X-join tile* (`X`-tile). The *types of tiles* are denoted by `O`, `C`, `D`, `S`, `T` and `X`, respectively. An expression *Y-tiles* for $Y \subseteq \{\mathtt{O}, \mathtt{C}, \mathtt{D}, \mathtt{S}, \mathtt{T}, \mathtt{X}\}$ means the tiles with types from $Y$.

Each of the tiles can be freely rotated in the grid (but not moved from its place). We refer to a grid with fixed rotations of its tiles as to a *tiling*. A tiling is *proper* if the tiles in each pair of the neighboring tiles either touch by their close sides or their open sides (i.e. they form safe pairs) and the tiles at the boundary of the grid touch the boundary by their empty sides. A rotation of the tiles which gives a proper tiling is a *proper rotation*. Neighboring tiles touching by their open sides are called *linked*.

In a *tiling problem*, you have to decide whether a given instance of the problem has a proper rotation. We also consider in the paper the tiling problems where only some types of tiles are allowed. An *Y-tiling problem* is a tiling problem where the types of tiles in a grid can be only from the set $Y \subseteq \{\mathtt{O}, \mathtt{C}, \mathtt{D}, \mathtt{S}, \mathtt{T}, \mathtt{X}\}$; e.g., the `CDT`-tiling problem is the tiling problem where the types of the tiles are restricted to the types `C`, `D` and `T`.

## 3. The general case

We settle the complexity of the general tiling problem in the next theorem.

**Theorem 1.** *The* `OCDSTX`*-tiling problem is* NP*-complete.*

**Proof.** Obviously, the problem is in NP. To prove its NP-hardness, we reduce the problem of planar (1, 3)-satisfiability to the `OCDSTX`-tiling problem. Planar (1, 3)-SAT

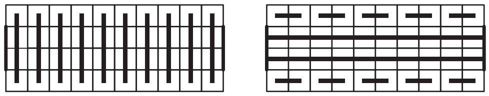

Fig. 2. Wires transporting the false (the left one) and the true signal (the right one).

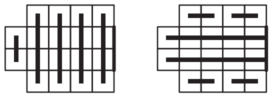

Fig. 3. The signal generator gadget.

is a satisfiability problem where the input formula is planar, all its clauses have sizes exactly three and we ask whether there is a variable assignment such that each clause contains exactly one true literal. A formula is said to be planar if its bipartite incidence graph is planar. The bipartite incidence graph of a formula is a graph whose vertices correspond to the variables and the clauses of the formula and a "variable vertex" is joined by an edge to a "clause vertex" if the corresponding variable is contained in the corresponding clause. The planar $(1,3)$-SAT is known to be NP-complete, see [4,5,7].

We construct various gadgets that are needed for the reduction. Throughout the proof, the straightforward and boring formal verification that the gadgets have the claimed properties is not presented in full detail. In the figures, we always draw all the possible proper orientations of the gadgets in order to assist the reader to figure out the omitted details. The gadget "contact" points are drawn with bold lines in the figures (see, e.g., Fig. 2). All the gadgets have even sizes, and they are placed so that the contact areas are at even distances from each other. Also, they are placed so that on all the sides with the exception of sides containing the contact points they touch empty tiles.

We first draw the incidence graph of the formula to a grid (the graph is planar and hence it can be also drawn in a grid). We replace the edges by "wires" from Fig. 2. These wires distribute the value of the variable from the variable gadgets to the clause gadgets. We use the two possible orientations of the S-tiles in the wire to represent the truth values: Let the orientation perpendicular to the direction of the wire represent the false value and the orientation parallel to the direction of the wire the true value (cf. Fig. 2). The wires guarantee that the signal is correct (i.e., all the S-tiles have the same orientation). In principle, the wires can be oriented so that their tiles adjacent to the contact areas also have a link to the adjacent gadgets. However, it can be verified that none of the other gadgets we use can be attached to such an orientation.

The wires can be terminated using the gadgets from Figs. 3 and 4, we further refer to these gadget as to the generator and true gadgets. The wires can be turned using the gadgets from Fig. 5; the top gadget in Fig. 5 turns the wire and preserves its value, the bottom one turns the wire and negates its value. The wire can be split into two wires with the same value using the gadget from Fig. 6. We can now describe the variable gadget: Start a "new" wire with the generator gadget, use several times

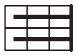

Fig. 4. The true signal generator gadget.

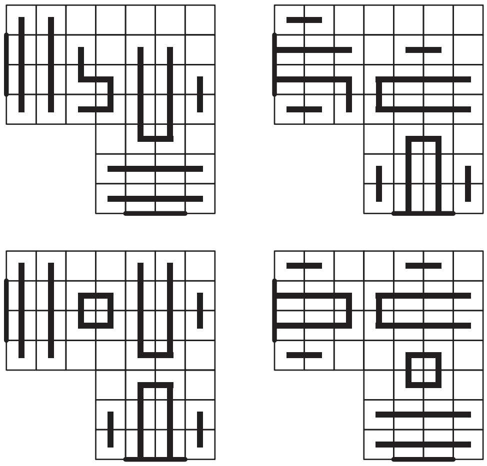

Fig. 5. The identity turn (the top two figures) and the negating turn (the bottom two figures).

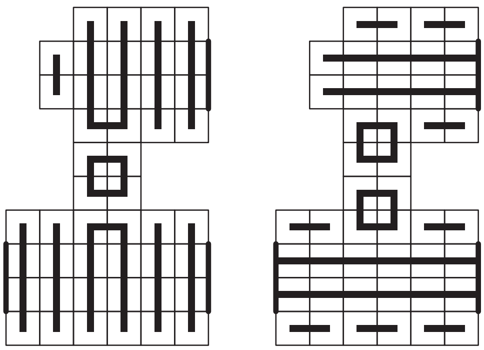

Fig. 6. The signal split gadget.

the split gadget (from Fig. 6) to make sufficiently many copies of the value of the variable (i.e., as many as the degree of the variable vertex in the bipartite incidence graph of the formula) and then conduct the values through the wires using the turn gadgets (those from Fig. 5) to the clause gadgets. If the variable is negated in the particular clause, negate it using the "negating turn gadget".

The clause gadget can be found in Fig. 7. This gadget has only the three states depicted in the figure. Exactly one of the inputs $X$, $Y$ and $Z$ is true in each of the admissible configurations and the remaining ones are false.

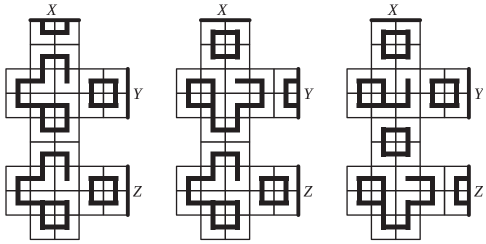


Fig. 7. The clause gadget.

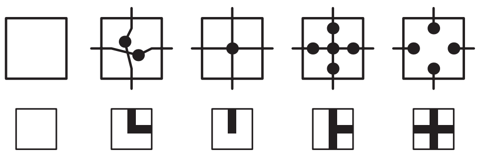

Fig. 8. The matching gadgets for the OCDTX-tiling problem.

The overview of the reduction is as follows: Construct for a given instance of the planar $(1,3)$-SAT an instance of the OCDSTX-tiling problem as shown above. This instance has polynomial size and can be constructed in polynomial time. If it can be properly oriented, then the values corresponding to the orientations of the wires give a satisfying assignment of the given formula. On the other hand, a satisfying assignment of the input formula provides a proper orientation of the tiling problem. Hence the OCDSTX-tiling problem is NP-complete. □

## 4. Polynomial cases

It might be surprising that forbidding only the S-tiles in input instances reduces the complexity of the problem.

**Theorem 2.** *The* OCDTX-*tiling problem can be solved in polynomial time*; *there is an algorithm which runs in time* $O((ab)^{3/2})$ *for an* $a \times b$ *grid.*

**Proof.** We reduce an instance of the OCDTX-tiling problem to the perfect matching problem in bipartite graphs which can be solved in polynomial time [3,6,8]. We replace each of the tiles of types $\{C, D, T, X\}$ by a gadget from Fig. 8. The edges in the formed graph correspond naturally to the pairs of neighboring tiles which can be linked: If the edge is included to a matching, then the neighboring tiles are linked. In order to decide whether a given instance of the OCDTX-tiling problem can be properly rotated, it is enough to construct the above described graph and check whether it contains a perfect matching. Its perfect matchings correspond one-to-one to the proper tilings.

We show that the constructed graph is bipartite: The tiles in the grid can be colored in white and black in such a way that no two adjacent tiles have the same color. The vertices in the gadgets are assigned the colors of the tiles which they replace with the following exception: The colors of the central vertices of the gadgets replacing the T-tiles are inverted. This is a proper coloring of the constructed graph and hence the graph is bipartite.

The estimate on the running time follows from the existence [3,6,8] of an algorithm for perfect matchings in bipartite graphs which runs in time $O(n^{1/2}m)$, where $n$ and $m$ are the numbers of vertices and edges of the input graph. □

The remaining two polynomial-time cases are rather easy.

**Theorem 3.** *The* OCSX*-tiling problem can be solved in linear time.*

**Proof.** Realize that the rotation of any tile whose type is among $\{\mathtt{O},\mathtt{C},\mathtt{S},\mathtt{X}\}$ is determined by the fact whether the tile is linked to the tile above it and to the tile left from it. Hence it is possible to determine the only possible rotations of all the tiles in linear time by sweeping a given instance of the problem from left to right and from top to bottom. □

**Theorem 4.** *An instance of the* OSTX*-tiling problem can be properly rotated iff it contains only the empty tiles. Hence the* OSTX*-tiling problem can be solved in linear time.*

**Proof.** An instance which contains only the empty tiles can be trivially properly rotated. Take an instance of the OSTX-tiling problem which contains a tile which is not the O-tile and consider such a tile $Z$ in the top most row and the left most column. This tile cannot be linked to the tile above and to the tile to the left (this neighboring tile either is an O-tile or does not exist). But then the tile $Z$ cannot be properly rotated. □

## 5. Reductions

In this section, we use simulation of tiles by larger gadgets using fewer types of tiles to classify the remaining problems. First, we define what type of simulation we use for other tiles. We say for an odd integer $k$ that a $k \times k$ gadget $G$ *simulates* a $Z$-tile, $Z \in \{\mathtt{O},\mathtt{C},\mathtt{D},\mathtt{S},\mathtt{T},\mathtt{X}\}$ if

- In every proper rotation of the gadget (we do not demand by definition that only the empty sides of the tiles may touch the boundary of the gadget), each tile touches the boundary of the gadget by its empty side, with possible exceptions for the tiles in the middle row and in the middle column.
- For each rotation of a $Z$-tile, there is a proper rotation of $G$ such that $G$ can be linked in the same directions (up, down, right, left) as the $Z$-tile, and vice versa, for each proper rotation of $G$, there is a corresponding rotation of a $Z$-tile.

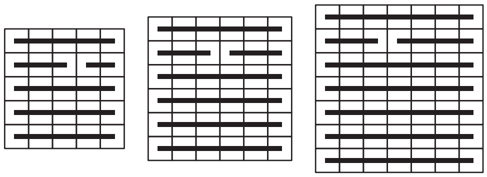

Fig. 9. The DS-gadgets which simulate the empty tile.

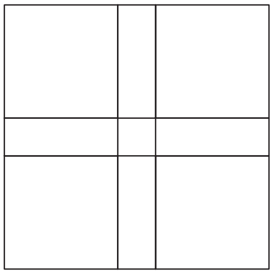

Fig. 10. Regions in the $(4k+1)\times(4k+1)$ grid from the proof of Lemma 6.

A $Z$-tile *can be simulated* by $k \times k$ gadget if there exists a gadget $G$ with the above described properties. If a gadget $G$ contains only tiles of types from a set $Y \subseteq \{\mathtt{0},\mathtt{C},\mathtt{D},\mathtt{S},\mathtt{T},\mathtt{X}\}$, $G$ is a *$Y$-gadget*.

First of all, we show that the empty tiles may be simulated by DS-gadgets.

**Lemma 5.** *For each integer $k \geqslant 5$, the empty tile may be simulated by a* DS-*gadget of order $k$.*

**Proof.** Consider the gadget from Fig. 9, the construction is extended for larger integers in a natural way. To prove that the gadgets simulate the empty tile, it is enough to prove that the configurations of the gadgets depicted in Fig. 9 are the only possible ones. The rotations of all the S-tiles have to be the same. They cannot be vertical, because of the D-tile in the second row and the third column. Hence, all the S-tiles are rotated as in the figure and the rotations of the remaining D-tiles are forced, too. □

Next, we prove other lemmas on existence of suitable types of gadgets.

**Lemma 6.** *For each integer $k \geqslant 3$ the $Z$-tile can be simulated by a* DS$Z$-*gadget of order $4k+1$ for any $Z \in \{\mathtt{0},\mathtt{C},\mathtt{D},\mathtt{S},\mathtt{T},\mathtt{X}\}$.*

**Proof.** Let $k \geqslant 3$ be a fixed integer. We divide the $(4k+1)\times(4k+1)$ grid into 9 regions (see Fig. 10): The corner regions have sizes $2k \times 2k$, the intermediate ones between them have sizes $1 \times 2k$ and $2k \times 1$ and the size of the middle one is just $1 \times 1$.

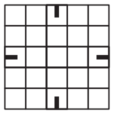

Fig. 11. The ODS-gadget of order 5 which simulates the X-tile.

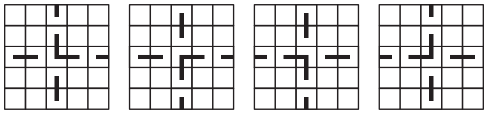

Fig. 12. The OCDS-gadget of order 5 which simulates the C-tile.

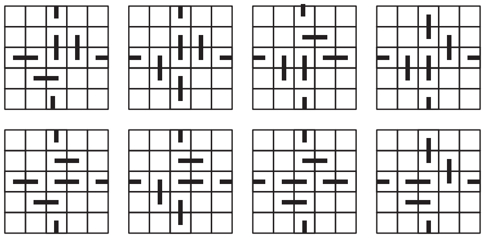

Fig. 13. The ODS-gadget of order 5 which simulates the T-tile.

We place in each of the four corner regions a DS-gadget of order $2k$ which simulates the O-tiles (Lemma 5) and we put in each of the four intermediate regions $2k$ D-tiles and we place in the middle region the $Z$-tile. Since the gadgets in the corner regions have only one possible rotation, the whole gadgets behave like if there were only the empty tiles in the corner regions and the tiles in the intermediate regions just transport the linking signal from the middle tile to the boundary; it is important that $2k$ is even in order not to negate the signal. Hence the gadget really simulates the $Z$-tile. □

**Lemma 7.** *The* ODSTX-*tiles can be simulated by* ODS-*gadgets of order* 5. *The* C-*tiles can be simulated by* OCDS-*gadgets of order* 5.

**Proof.** The ODS-tiles can be simulated by ODS-gadgets of order 5 in a way similar to Lemma 6; the corner $2 \times 2$ regions are filled by O-tiles and the remaining regions are filled exactly in the same way as in Lemma 6. The other simulation gadgets are depicted in Fig. 11–13. It is straightforward to verify that the configurations of the gadgets depicted in the figures are the only possible ones. □

We are now ready to prove the main theorem of this section.

**Theorem 8.** *Both the* CDS-*tiling problem and the* CST-*tiling problems are* NP-*complete*.

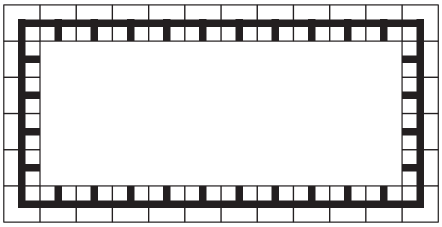

Fig. 14. The border around the complementary grid formed by the C-tiles and the T-tiles.

Table 1
The summary of our complexity results

| O | C | D | S | T | X | Complexity |
|---|---|---|---|---|---|---|
| ★ | ★ | ★ | × | ★ | ★ | Polynomial (Theorem 2) |
| ★ | ★ | × | √ | × | ★ | Polynomial (Theorem 3) |
| ★ | √ | √ | √ | ★ | ★ | NP-complete (Theorem 8) |
| ★ | √ | ★ | √ | √ | ★ | NP-complete (Theorem 8) |
| ★ | × | × | √ | ★ | ★ | Polynomial (Theorem 4) |
| ★ | × | √ | √ | ★ | ★ | The same (unknown) complexity (Theorem 9) |

The sign √ means that the corresponding type of tiles is present, the sign × means that it is not present and ★ means that it does not matter.

**Proof.** The general OCDSTX-tiling problem is NP-complete due to Theorem 1. The OCDS-tiling problem is NP-complete due to Lemma 7 and finally the CDS-tiling problem is NP-complete due to Lemmas 5 and 6 (used for $Z =$ C, D and S).

We reduce an instance of the CDS-tiling problem to an instance of the CST-tiling problem. First complement the tiles, i.e., replace the D- with T-tiles and keep both the C- and S-tiles. Next, create a boundary around the grid consisting of the C- and the T-tiles as shown in Fig. 14. The obtained instance of the problem has a proper rotation iff the original instance has one: It is enough to complement the tiles, e.g., if a curve tile is linked to the tiles above and to the right, then in the complement, the tile is linked to the tiles down and to the left. Hence, the CST-tiling problem is NP-complete. □

## 6. The unsettled cases

What cases remain unsettled? If S-tiles are not allowed, the tiling problem can be solved in polynomial time by Theorem 2. If neither D- nor T-tiles are allowed, the problem can be solved in polynomial time due to Theorem 3. The remaining cases are those where S-tiles are allowed together with D- or T-tiles. If even C-tiles, besides S- and D-tiles or T-tiles, are allowed, then the problem is NP-complete due to Theorem 8. Only the cases where C- are not allowed remain. If neither C-tiles nor D-tiles are allowed, the problem can be solved in polynomial time due to Theorem 4, see Table 1. Hence the unsettled problems are the $Y$-tiling problems where $\{\mathtt{D},\mathtt{S}\} \subseteq Y \subseteq \{\mathtt{O},\mathtt{D},\mathtt{S},\mathtt{T},\mathtt{X}\}$. We show that all these cases have the same complexity.

**Theorem 9.** *The complexity of the* DS-*tiling problem and the complexity of the* ODSTX-*tiling problem are the same.*

**Proof.** The ODSTX-tiling problem can be reduced to the ODS-tiling problem due to Lemma 7 and the ODS-tiling problem can be reduced to the DS-tiling problem due to Lemma 5 and 6 (used for $Z = \mathtt{D}$ and $\mathtt{S}$). □

## 7. Conclusion

The summary of the obtained complexity results can be found in Table 1. The unsettled case of the problem turns out to be surprisingly hard. We sketch the reasons why we think to be so.

First, we prove a claim which shows that a similar NP-completeness reduction based on gadgets is unlikely to exist because we cannot construct non-trivial gadgets, in particular an identity turn, or a negating gadget with inputs on the opposite sides. This also implies a non-existence of a gadget simulating a C-tile, since such a gadget would allow us to construct an identity turn. The proof uses a parity argument in conjunction with the planarity of the grid structure. The claim intentionally leaves vague the question of representation of the gadget inputs, since its variants seem to work for any representation. In fact, the claim seems to be pointing to some general invariant which we do not fully understand.

The statement of the claim is: Suppose we want to make a DS-gadget with two input points $A$ and $B$ belonging to different inputs of the gadget such that $A$ and $B$ are in even distance (i.e., leaving from the squares of the opposite colors in the chessboard coloring) and there is no other input point between $A$ and $B$ (i.e., going from $A$ clockwise along the border of the gadget, $B$ is the next input point). If there exist two proper orientations assigning to $A/B$ values false/true and true/false (in the notation of the proof of Theorem 1), respectively, then there also exists two proper orientations assigning to $A/B$ values false/false and true/true. (A similar claim can be shown for $A$ and $B$ on squares of the same color: If false/false and true/true configurations are both possible, then also false/true and true/false are possible.)

**Proof of the claim.** Take a false/true configuration and fill (draw) all the used pipes by blue and then take a true/false configuration and fill (draw) all the used pipes by red. A pipe which is both red and blue is purple. Erase all purple pipes. Each square either (i) is empty, or (ii) contains two crossing straight pipes of distinct colors (it origins from an S-tile), or (iii) two dead ends of distinct colors (the square origins from a D-tile). Furthermore, both the input pipe entering $A$ is red and the input pipe entering $B$ is blue. Start at the input pipe to $A$ and go left (i.e., in the direction to $B$) along the outside contour along the red and blue pipes. This gives you an alternating path: at a square of type (ii) you always turn and thus change color, or (iii) you change color no matter if you turn or not. By parity, the alternating path cannot end at $B$. Since it was the contour, it shows that the pipes of the inputs $A$ and $B$ are disconnected in the blue–red graph. This means that in each of the components we can switch the colors separately, and thus get the remaining configurations false/false and true/true.

On the other hand, our NP-completeness result can be extended to the case without C-tiles if we allow to draw a grid on any surface. More precisely, we allow a "board" to be glued from different rectangles with the restriction that it is locally correct. (In particular, the whole "board" can be non-planar.) On such a "board" it is easy to construct a gadget simulating a C-tile (simply replace the tile by two D-tiles, one connected to the right and left neighbor of the original C-tile and the other connected to the remaining two neighbors). Thus we can construct all the other needed gadgets and the NP-hardness proof applies. So any potential polynomial algorithm for the remaining open case needs to make an essential use of the planarity of the grid structure in the problem.

## Acknowledgements

Most of this research was done during the Dagstuhl Workshop on Algorithmic Combinatorial Game Theory; the authors are grateful to the workshop organizers for the opportunity to take part in this wonderful event. The third author acknowledges a support by the International Conference and Research Center Schloss Dagstuhl; the first and the fourth author acknowledge a support as young researchers from the program High Level Scientific Conferences of the European Union.

The authors would like to thank an anonymous referee for suggestions improving the readability of the paper.

## References

[1] D. Eppstein, Improved algorithms for 3-coloring, 3-edge-coloring and constraint satisfaction, in: Proc. 12th ACM–SIAM Symp. on Discrete Algorithms (SODA 2001), Washington, DC, pp. 329–337.
[2] M.R. Garey, D.S. Johnson, Computers and Intractability. A Guide to the Theory of NP-completeness, Freeman, San Francisco, CA, 1979.
[3] J. Hopcroft, R. Karp, An $n^{5/2}$ algorithm for matching in bipartite graphs, SIAM J. Comput. 2 (4) (1975) 225–231.
[4] J. Kratochvíl, A special planar satisfiability problem and some consequences of its NP-completeness, Discrete Appl. Math. 52 (1994) 233–252.
[5] D. Lichtenstein, Planar formulae and their uses, SIAM J. Comput. 11 (1982) 329–343.
[6] L. Lovász, M.D. Plummer, Matching Theory, North-Holland, Amsterdam, Netherlands, 1986.
[7] A. Mansfield, Determining the thickness of graphs is NP-hard, Math. Proc. Cambridge Philos. Soc. 93 (1983) 9–23.
[8] S. Micali, V.V. Vazirani, An $O(\sqrt{|V|}|E|)$ algorithm for finding maximum matching in general graphs, Proc. 21st Ann. Symp. Foundations of Computer Science (FOCS), Syracuse, New York, 1980, pp. 17–27.
[9] S. Rütter: NP-Vollständigkeit des "Plumber-Problems", available at http://www.itheoi.mu-luebeck.de/pages/jakoby/veranstaltungen/studienarbeiten/steffen_rueter/index.ht.