A

Placement & Routing for Tile-based Field-coupled Nanocomputing Circuits is $\mathcal{NP}$-complete

MARCEL WALTER, University of Bremen
ROBERT WILLE, Johannes Kepler University Linz
DANIEL GROSSE, University of Bremen
FRANK SILL TORRES, DFKI GmbH
ROLF DRECHSLER, University of Bremen

Field-coupled Nanocomputing (FCN) technologies provide an alternative to conventional CMOS-based computation technologies and are characterized by intriguingly low energy dissipation. Accordingly, their design received significant attention in the recent past. FCN circuit implementations like *Quantum-dot Cellular Automata* (QCA) or *Nanomagnet Logic* (NML) have already been built in labs and basic operations such as inverters, Majority, AND, OR, etc. are already available. The design problem basically boils down to the question how to place basic operations and route their connections so that the desired function results while, at the same time, further constraints (related to timing, clocking, path lengths, etc.) are satisfied. While several solutions for this problem have been proposed, interestingly no clear understanding about the complexity of the underlying task exists thus far. In this research note, we consider this problem and eventually prove that placement and routing for tile-based FCN circuits is $\mathcal{NP}$-complete. By this, we provide a theoretical foundation for the further development of corresponding design methods.

1. INTRODUCTION

Field-coupled Nanocomputing (FCN) [Anderson and Bhanja 2014] is a relatively young class of emerging computation technologies that might enable the design and realization of systems with highest processing performance and an intriguing low energy dissipation [Timler and Lent 2002; Srivastava et al. 2009; Porod et al. 2014]. FCN technologies allow for a completely different way of representing and processing Boolean values and functions based on local fields between nanoscale devices and repelling forces to transmit information without electric current flow. Consequently, numerous contributions on their physical realization have been made in the past, e. g. *molecular Quantum-dot Cellular Automata* (mQCA) [Eichwald et al. 2012], *atomic Quantum-dot Cellular Automata* (aQCA) [Wolkow et al. 2014; Huff et al. 2017], or *Nanomagnetic Logic* (NML) [Eichwald et al. 2012; Varga et al. 2013; Porod et al. 2014].

Regardless of how information is represented in the different FCN implementations, they all share certain attributes. Basic elements are so-called *cells* which store Boolean values and interact with adjacent cells via repelling forces [Lent and Tougaw 1997; Giri et al. 2018]. These cells can be based on atoms [Wolkow et al. 2014], molecules [Arima et al. 2012] or nanomagnets [Varga et al. 2013]. Furthermore, data transfer is controlled by external electric (QCA-like technologies) or magnetic clocks (NML technologies). For example, in the case of QCA-like technologies, binary information is represented by means of four *quantum dots* situated at the corners of a cell as well as two free and mobile *electrons* which are able to tunnel between adjacent dots. Since the electrons experience mutual repulsion due to Coulomb interaction, they tend to locate themselves as far as possible from each other. This yields to two stable states as shown in Fig. 1a (quantum dots occupied by an electron are drawn black, otherwise, they are drawn white) which are used to encode the Boolean values 0 and 1, respectively.

In order to conduct computations, several of such cells are placed next to each other as shown e. g. in Figs. 1b to 1d. Although the electrons of one cell may not tunnel to another cell due to a potential barrier, their polarizations can affect each other. This way, the charge of electrons leads to a mutual repulsion which causes them to avoid a quantum dot if the neighboring quantum dots of adjacent cells are occupied by other electrons. This can be

used to implement e. g. a wire as shown in Fig. 1b, an inverter as depicted in Fig. 1c, or a Majority gate as shown in Fig. 1d. The wire element simply propagates information from one end to the other by exploiting the Coulomb interaction as mentioned above. The inverter however is a bit more complex. The signal is assumed to traverse the gate from left to right. The actual inversion happens as QCA cells interact with one another over the corners. As depicted in Fig. 1c, the state of the output f is inverted compared to input a. The Majority gate works by three cells influencing the middle cell which adopts to the majority of the cells' states and propagates its value to the rightmost cell that is considered the output. The AND and OR gates can be derived from a Majority gate easily by fixing one input to 0 or 1, respectively.

Given this as basis, an important task for all FCN related technologies is the *design* of the desired circuit, i. e. how to arrange cells as reviewed above so that eventually an FCN circuit results which implements an initially given function. Recent developments treated this problem in a *tile-based* fashion [Huang et al. 2005; Campos et al. 2016; Giri et al. 2018]. That is, a predefined *floor plan* is given that is divided into *tiles* to which wires and basic operations such as those shown in Figs. 1b to 1d[1] can be assigned. This eventually boils down the design problem to a placement and routing task: how to place basic operations and route their connections (i. e. wires) so that the desired function results while, at the same time, further constraints (related to timing, clocking, path lengths, etc. which are reviewed in more detail in the next section) are satisfied.

The literature provides several solutions to this problem which enabled researchers to implement e. g. arithmetic functions [Perri and Corsonello 2012; Giri et al. 2016], FPGAs [Kianpour and Sabbaghi-Nadooshan 2014], or small processors [Fazzion et al. 2014]. Besides that, also several methods for the automatic placement and routing of FCN circuits have been introduced – including heuristic methods such as [Nath et al. 2017; Trindade et al. 2016] as well as exact methods such as [Walter et al. 2018]. However, all these endeavors have been conducted without a clear understanding about the complexity of the underlying task. As a consequence, we currently have heuristics which usually generate non-optimal results on the one hand and exact solutions which significantly suffer from huge run-times and severely limited scalability on the other, but do not know whether this is caused by insufficient methodology or complexity reasons.

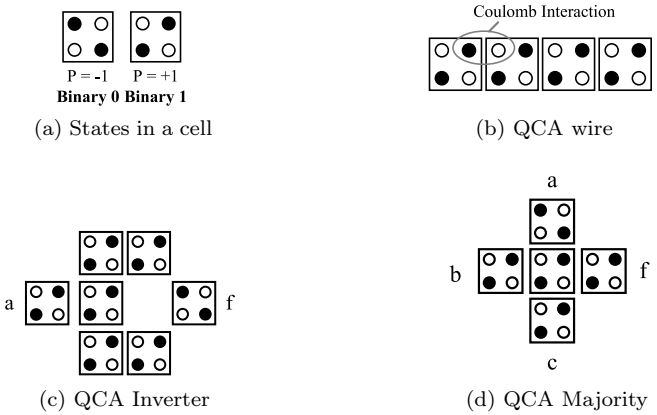
In this research note, we shed light on this. To this end, we provide the first formally sound definition of the tile-based FCN circuit placement and routing problem independent of specific physical implementations. Therefore, we utilize graph theoretical notation which allows us to formulate the problem in a way as close to applicability as possible. In other words, the notation directly points at related problems in the domain of graph theory which can be used as a starting point in the development of approximation algorithms. Then, we prove that the considered problem is $\mathcal{NP}$-complete and, hence, indeed a computationally complex task. By this, we provide a theoretical foundation for the further development of corresponding design methods.

The remainder of this work is structured as follows. While this section already provided the basics of FCN technology by means of QCA-like circuits, Section 2 illustrates the tile-based placement and routing problem – eventually resulting in a formal problem definition. Afterwards, Section 3 provides the proof for $\mathcal{NP}$-completeness. Finally, Section 4 summarizes the findings and discusses their possible implications.

2. PLACEMENT & ROUTING OF TILE-BASED FCN CIRCUITS

As reviewed above, the overall placement and routing task is formulated as determining an assignment of operations to floor plan tiles and connecting them by wires such that the

[1]Usually specified by a gate library [Giri et al. 2016; Torres et al. 2018b; Reis et al. 2016].



(a) States in a cell (b) QCA wire

(c) QCA Inverter (d) QCA Majority

Fig. 1: QCA states and operations

desired functionality is conducted. However, to ensure a correct physical realization, several constraints related to clocking, path lengths, etc. have to be considered during this process.

More precisely, external clocks are employed which regulate the intercellular tunneling barrier within QCA cells or the ability of NML cells to change their magnetization state. These clocks enable proper propagation of information among the cells and avoid metastable states [Hennessy and Lent 2001; Porod et al. 2014]. In case of tile-based floor plans, each tile is associated with an external clock which controls all cells within the specific tile [Huang et al. 2005; Giri et al. 2018]. Alternatively, the associated clock of each cell could be defined individually, as e. g. in [Fazzion et al. 2014; Roohi et al. 2015]. However, this raises serious concerns regarding the feasibility of the clock signal distribution and design complexity [Lent and Tougaw 1997; Huang et al. 2005; Vankamamidi et al. 2008] and, consequently, shall not be considered here.

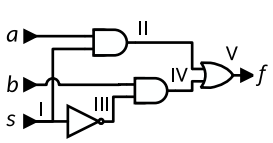
Instead, most FCN implementation models assume a 4-phase clocking with external clocks numbered from 1 to 4 and each clock shifted by 1 phase [Anderson and Bhanja 2014]. Here, information flows from cells controlled by clock 1 to cells controlled by clock 2 etc. and finally from clock 4 to clock 1. Nonetheless, several NML implementations employ a 3-phase clocking [Porod et al. 2014]. To keep the definition of the considered problem as generic as possible, we will assume that a clocking consists of C clock numbers.

An additional constraint that has to be considered is the throughput effect of equal path lengths. FCN circuits can be implemented like pipelines: new data are applied to the primary inputs in every clock cycle – eventually leading to corresponding results at the primary outputs. However, this is only the case for circuits that employ wire connections in a way that lead to exactly the same delay[2] of all signals to all multi-input gates on the floor plan. If this is not ensured, the FCN circuit's overall throughput drops and the same input signals must be applied over a period of multiple clock cycles as a consequence [Torres et al. 2018a].

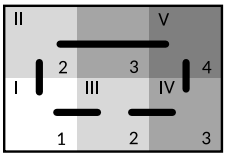
The following example illustrates some of these constraints.

Example 2.1. Consider the 2:1 multiplexer (MUX) function to be realized (in QCA) and whose (conventional) netlist is depicted in Fig. 2a. For all corresponding basic operations,

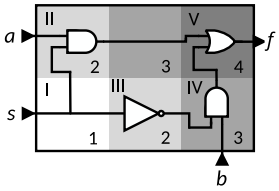
[2]Here, delay means the number of complete clock cycles a signal requires to traverse the circuit. A clock cycle lasts for the number of phases of the applied clocking and the minimum delay is 1.



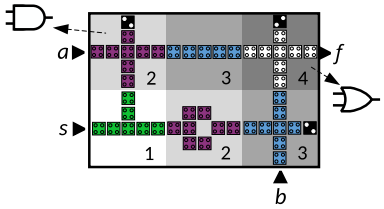
(a) 2:1 MUX netlist



(b) (p, r, c) mapping on $G_{3,2}$



(c) Gate level impl.



(d) QCA implementation

Fig. 2: Placement, routing, clocking, and final implementation of QCA 2:1 MUX

namely inverter, AND, OR, corresponding QCA realizations are available (see e. g. [Reis et al. 2016; Torres et al. 2018b]).

Fig. 2c and Fig. 2d depict a valid gate level implementation and the corresponding QCA circuit realizing that very function and also satisfying the constraints sketched above. As an example, consider the OR gate at the top right which is located in a tile controlled by clock 4. In order to enable a correct information flow, both inputs are driven by a wire and an AND gate respectively located in tiles controlled by clock 3.

There are pros and cons for satisfying a constraint or loosen it (as they e. g. affect area vs. throughput). Since we aim for maximum generality, we will prove both cases to be $\mathcal{NP}$-complete in the remainder.

Overall, this yields a formal definition of the placement and routing problem for tile-based FCN circuits (hereafter denoted FCNPR) as follows:

Definition 2.2. The FCNPR is a decision problem. It gets a floor plan $F = (T, A)$ as well as a netlist $N = (O, W)$ and a clock number $C \in \mathbb{N}$ as inputs.[3] F and N are both directed graphs with vertices and edges (which have been renamed as (T, A) and (O, W) respectively for ease of recognition) where F is composed of tiles T connected by an adjacency relation A and serves as a host graph, and N consists of operations O (given by a gate library) which fit on a single tile each and that are connected by wires W, and serves as the associated guest graph to F. FCNPR evaluates to `true` for F, N, and C iff there exists a 3-tuple of injective mappings (p, r, c), where $p : O \to T$ is the operation placement, $r : W \to \mathcal{P}_F$ is the wire routing,[4] and $c : T \to \{0, \ldots, C - 1\}$ is the clock number assignment, such that the following constraints hold:

[3]From a technical perspective, it makes sense to enforce $C \geq 3$ because no directed information flow is possible otherwise.

[4]Let $\mathcal{P}_F$ denote the set of all cycle-free paths in F and let paths be ordered (multi-)sets of vertices (i. e. tiles in this case).

(1) *Routing constraint:* Each wire between two operations o_1, o_2 in the netlist is routed on the floor plan in a way that it starts at the tile t_1 where o_1 is placed, ends at the tile t_n where o_2 is placed, and does not cross any other operation (crossing other wires is allowed though). More formally: $\forall (o_1, o_2) \in W : ((r((o_1, o_2)) = \mathcal{T} = \{t_1, \ldots, t_n\}) \implies (t_1 = p(o_1) \wedge t_n = p(o_2))) \wedge \forall o \in O : p(o) \notin \mathcal{T} \setminus \{t_1, t_n\}$.

(2) *Clock Number Assignment Constraint:* For each wire between two operations in the netlist that is routed on the floor plan, the clock numbers need to be assigned to the affected tiles in a way that they are consecutively numbered, where the highest possible clock number is followed by the lowest one (i. e. 0). More formally: $\forall (o_1, o_2) \in W : (r((o_1, o_2)) = \{t_1, \ldots, t_n\}) \implies \forall i \in \{2, \ldots, n\} : ((c(t_i) - c(t_{i-1})) \bmod C = 1)$.

(3) *Throughput Constraint:* All paths in the netlist starting at some primary inputs and sharing a common last vertex, must be equally long when routed on the floor plan while taking their clock numbers into account to compensate delay differences. More formally: $\forall p_1, p_2 \in \mathcal{P}_N$, where $p_1 = \{o_1, \ldots, o'\}$, and $p_2 = \{o_2, \ldots, o'\} : |r(o_1, o')| + c(p(o_1)) = |r(o_2, o')| + c(p(o_2))$, where $\forall o \in O : (o, o_1), (o, o_2) \notin W$, p_1 and p_2 are disjoint except for o', and let $\mathcal{P}_N$ be analogously defined to $\mathcal{P}_F$, and let $|P|$ denote the length of a path P.

Example 2.3. Consider the netlist of the 2:1 MUX given in Fig. 2a again. In Example 2.1, we have shown a possible QCA circuit realization for that netlist onto a 3×2 grid graph.[5] Possible mappings (p, r, c) are shown in Fig. 2b. The operation placement is depicted by Roman numbers associated with the operation numbering in Fig. 2a, while connections between operations (i. e. the wire routing) are drawn as bold lines. Note the difference between a wire in the netlist, which is an abstract connection between two operations o_1 and o_2, and the physical wire routing, which connects two placed operations $p(o_1)$ and $p(o_2)$. In Fig. 2b and 2d, these physical connections are depicted. As it can be seen, they can be of effective length 0 when a direct gate-to-gate connection is established without a wire element spanning in between. Additionally, a clocking that ensures correct information flow is given by Arabic clock numbers together with a grayscale background tile shade.[6] Overall, these mappings correspond to the QCA 2:1 MUX circuit given in Fig. 2d.

3. PROOF OF $\mathcal{NP}$-COMPLETENESS

In this section, we prove that the problem defined above is $\mathcal{NP}$-complete, i. e. we prove the following Lemmas 3.1 and 3.3 which lead to Proposition 3.5:

LEMMA 3.1. FCNPR *is in* $\mathcal{NP}$.

PROOF. Assume a non-deterministic Turing machine that *guesses* the 3-tuple (p, r, c) for a given netlist and floor plan and checks in polynomial time if it complies with all constraints. □

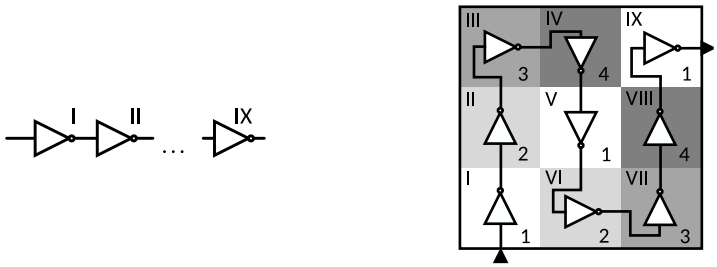
We will show the $\mathcal{NP}$-hardness by a polynomial time reduction from the well known *Hamiltonian path problem* (HPP) to FCNPR.

Definition 3.2. The HPP is a decision problem that was shown to be $\mathcal{NP}$-complete [Garey and Johnson 1979]. It gets a directed graph $G = (V, E)$ as input. Its output is `true` iff there exists a path in G that contains each vertex exactly once.

LEMMA 3.3. FCNPR *is* $\mathcal{NP}$*-hard.*

[5]Let an $x \times y$ grid graph formally be defined as $G_{x,y} := (V, E)$, where $V = \{1, \ldots, x\} \times \{1, \ldots, y\}$ and $E = \{((i, j), (i', j')), ((i', j'), (i, j)) \mid |i - i'| + |j - j'| = 1\}$.

[6]Note that we started clock numbering at 0 for ease of denotation using the modulo operation in our formal definition. In Fig. 2, however, we employed the commonly used numbering starting at 1. They can trivially be translated into one another by adding/subtracting 1 to/from every clock number.



(a) Single input/single output operation chain

(b) Finding a Hamiltonian path via embedding

Fig. 3: HPP $\leq_P$ FCNPR

PROOF. We prove this by providing a polynomial time reduction HPP $\leq_P$ FCNPR. That is, to a given instance of HPP, we can construct an instance of FCNPR in polynomial time so that they are *equisatisfiable*. It follows directly that iff an oracle provided a solution to that FCNPR instance, we can deduce a solution for the HPP instance easily. Thus, HPP cannot be harder than FCNPR.

Given a directed graph $G = (V, E)$ with n vertices. It is to be decided if HPP holds for G (i. e. if G is an element of the language $\mathcal{L}_{\text{HPP}}$) via reduction to FCNPR. Construct a netlist $N = (O, W)$ in a way that $O = V$ and $W = \{(o_1, o_2), (o_2, o_3), \ldots, (o_{n-1}, o_n)\}$, $o_i \in O$, i. e. a chain of all n operations.[7] This can be done in linear time. In the following, G serves as a floor plan for N. Since the number of vertices in both graphs is identical, a bijective operation placement p is possible (which also is injective by definition). If the wire routing r can also be employed while meeting the routing constraint (1), a Hamiltonian path has been found. This means, if the vertices are placed in a way that they are adjacent on the floor plan if they are adjacent in the netlist, the routing constraint can be fulfilled. Nonetheless, the clock number assignment constraint (2) must be met as well to truly satisfy FCNPR. If the embedding was possible, c can be chosen so that the tile onto which operation o_i is placed got clock number $i \bmod C$ assigned because propagating information through a chain is easily possible this way. Plus, assigning the clocking has only a linear overhead then. So the reduction stays polynomial. Finally, the throughput constraint (3) is trivially satisfied because there are no two paths which are disjoint except for their last vertex in the netlist.

As a consequence, HPP holds for G iff FCNPR holds for G and N, i. e. iff it is possible to embed the n-operation chain in G. Thus, FCNPR is $\mathcal{NP}$-hard. □

Example 3.4. To provide a visual intuition of how the operation chain embedding works, which is the core of the reduction, consider exemplarily the task of finding a Hamiltonian path in a 3×3 grid. The grid consists of 9 tiles, i. e. a 9-operation chain is created, like the one shown in Fig. 3a. If it is possible to embed this chain in the grid (i. e. determining mappings p and r), a Hamiltonian path has obviously been found. This is depicted in Fig. 3b. As it can be seen, determining the clocking c becomes trivial then by consecutively clock-numbering the tiles along with the chain, thus satisfying constraint (2). Furthermore, the throughput constraint (3) is satisfied because there are no two disjoint paths in the chain netlist.[8]

[7] We assume there is a single input/single output operation given by a gate library. Even the identity function (i. e. a wire) can be employed.

[8] Note that we have given a grid graph as an example for ease of visualization only. Any floor plan can be employed here in fact.

One might wonder if the permission to employ wire crossings does have any effect on the equisatisfiability of HPP and FCNPR during the reduction because this would imply that vertices could be "visited" twice which is obviously not allowed in the HPP. In fact, this is not a problem because we employ n-operation chains where the operation placement p is bijective. So there are no vertices in the floor plan which are occupied by wire nodes. Additionally, wires crossing operations is forbidden by definition. Therefore, no crossings are possible during the reduction and the equisatisfiability holds.

Based on the discussions from above, we now can conclude the following:

PROPOSITION 3.5. FCNPR *is $\mathcal{NP}$-complete.*

PROOF. Follows directly from Lemmas 3.1 and 3.3. □

PROPOSITION 3.6. FCNPR *remains $\mathcal{NP}$-complete when relieving the throughput constraint (3).*

PROOF. Straight-forward as constraint (3) was trivially satisfied in the proof of Lemma 3.3. □

PROPOSITION 3.7. FCNPR *remains $\mathcal{NP}$-complete when prohibiting wire-crossings.*

PROOF. Straight-forward as the chain embedding in the proof of Lemma 3.3 does not utilize wire crossings. □

Accordingly, we have shown $\mathcal{NP}$-completeness for the most general cases of the FCNPR. In the real world, however, the input floor plan is usually not an arbitrary graph. Instead, grid structures are often employed as shown by our examples. But this does not affect the complexity class as shown in our next proposition.

PROPOSITION 3.8. FCNPR *remains $\mathcal{NP}$-complete even if the input floor plan is restricted to grid graphs.*

PROOF. It was shown in [Itai et al. 1982] that HPP remains $\mathcal{NP}$-complete when the input is restricted to grid graphs. Thus, an analogous reduction as given in the proof of Lemma 3.3 can be conducted utilizing the findings from [Itai et al. 1982]. □

Furthermore, the clock number assignment c is often predefined to the floor plan prior to the layout step when using a scheme as e. g. in [Campos et al. 2016]. The clocking can therefore be considered part of the input.

PROPOSITION 3.9. FCNPR *remains $\mathcal{NP}$-complete when the clock number assignment c is part of the input.*

PROOF. Use the floor plan $F = (T, A)$ and the clocking map c to construct a clocking graph $G_c = (V_c, E_c)$, with $V_c := T$ and $E_c := \{(t_1, t_2) \mid (t_1, t_2) \in A \wedge (c(t_1) + 1) \bmod C = c(t_2)\}$. That is, the vertex set is equal to the floor plan's vertices and the edge set corresponds to all connections which (1) exist in the floor plan as well and additionally fulfill the requirement that they (2) allow information flow based on the clock number assignment constraint, i. e. have a consecutive numbering modulo the highest clock number C.

Then utilize G_c as the new input floor plan to FCNPR. This way, only connections allowed by information flow regarding F and c remain possible and the complexity of the problem does not change. □

One can prove that all possible combinations of the above restrictions still leave the problem's complexity unchanged.

Thus far, we have looked into the *existence* of valid solutions. Designers are often interested in *optimal* solutions for problem instances, too. Therefore, we introduce the optimization variant of FCNPR which we call FCNOPR.

Definition 3.10. FCNOPR is an optimization problem. Its inputs and constraints are those of FCNPR as given in Definition 2.2. Additionally, an optimization direction $type \in \{min, max\}$ and a *measure* m as a function that maps FCNPR solutions $S = (p, r, c)$ to a quality value $q \in \mathbb{N}$ is provided, whose computation must be in $\mathcal{FP}$, i. e. must be computable in polynomial time.

A suitable measure could be the number of occupied tiles, which one might want to minimize: $m(S) := \sum_{w \in W} |r(w)|$. In the definition, only the number of wires is respected, because this is the only variable factor (see e. g. [Torres et al. 2018c]), while the number of operations stays the same as by definition of FCNPR.

Alternatively, when working on a geometric floor plan like a grid, the area of the bounding box of the whole design can be used as a minimization measure, too. For grids, the definition would look like this: $m(S) := (max_x(T^*) - min_x(T^*)) \cdot (max_y(T^*) - min_y(T^*))$, where T^* is the set that contains all tiles occupied by either an operation or a wire, i. e. $T^* := \bigcup_{o \in O} \{p(o)\} \cup \bigcup_{w \in W} r(w)$. The functions $min_{x/y}$ and $max_{x/y}$ return the minimum or maximum x- or y-position of the tiles in the set respectively.

LEMMA 3.11. FCNOPR *is in* $\mathcal{NPO}$.

PROOF. To prove this, we have to show that (1) deciding whether a tuple (p, r, c) is a valid solution to FCNOPR is in $\mathcal{P}$, (2) $m \in \mathcal{FP}$, and (3) the solution size is bounded by a polynomial with respect to the input.

In proof of Lemma 3.1, we have already shown that deciding whether a tuple (p, r, c) is a valid solution to FCNPR is in $\mathcal{P}$. Since the problem definition of FCNOPR is identical, the same finding holds.

Additionally, the measure m is computable in polynomial time by definition, therefore $m \in \mathcal{FP}$ holds.

Finally, valid solutions are never larger in size than the input floor plan F, i. e. bounded by the polynomial $|F|$. □

PROPOSITION 3.12. FCNOPR *is* $\mathcal{NP}$*-hard.*

PROOF. Follows directly from Lemmas 3.3 and 3.11, i. e. the decision version of FCNOPR is $\mathcal{NP}$-hard and FCNOPR itself is in $\mathcal{NPO}$. □

4. CONCLUSION AND FUTURE WORK

In this research note, we have shown that placement and routing for the class of tile-based Field-coupled Nanocomputing circuits is $\mathcal{NP}$-complete while its corresponding optimization problem is in $\mathcal{NPO}$. By this, we provide a theoretical foundation for the further development of corresponding design methods for a new class of technologies. In fact, this result basically confirms that previously proposed heuristics and exact methods, which only generate non-optimal results or have a severely limited scalability, respectively, do not suffer from insufficient methodology but complexity reasons. For the design of a technology which comes with intriguing low energy properties and, hence, promising practical applications, this constitutes one of the first substantial theoretical results.

Moreover, this result will have a significant impact on future work. Besides the effect of the further development of corresponding design methods (now that the theoretical complexity of the problem has been shown, according reasoning methods such as theorem provers (e. g. [Nipkow et al. 2002]) or satisfiability solvers (e. g. [De Moura and Bjørner 2008]) seem a much more suitable choice to tackle this design problem), this result may also

trigger a further, more in-depth consideration of the computational theory behind FCN circuit design.

With the chosen graph-based formulation, several related problems like the *(directed) sub-graph homeomorphism problem* (denoted SHP) as introduced in [LaPaugh and Rivest 1980] and [Fortune et al. 1980] come into consideration for developing placers and routers for FCN technologies. Exploiting such similarities allow to transfer further results from the SHP domain (which has heavily been considered in the past, see e. g. [Lingas and Wahlen 2009; Matoušek and Thomas 1992; Chung 1987]) to the FCN domain. While SHP resembles FCNPR remarkably, recently, approximation algorithms for the loosely related *orthogonal graph drawing* (OGD) like [Biedl 1996] were found to approximate FCNOPR under certain restrictions based on the findings presented in this paper and have been successfully exploited in [Walter et al. 2019] demonstrating the significance of this work.

REFERENCES

N. G. Anderson and S. Bhanja. 2014. *Field-coupled Nanocomputing: Paradigms, Progress, and Perspectives* (1st ed.). Springer, New York.

Valentina Arima, Matteo Iurlo, and others. 2012. Toward Quantum-dot Cellular Automata Units: Thiolated-carbazole linked bisferrocenes. *Nanoscale* 4 (2012), 813–823. Issue 3.

Therese C. Biedl. 1996. Improved Orthogonal Drawings of 3-graphs.. In *CCCG*. 295–299.

C. A. T. Campos, A. L. P. Marciano, O. P. V. Neto, and F. S. Torres. 2016. USE: A Universal, Scalable, and Efficient Clocking Scheme for QCA. *TCAD* 35, 3 (2016), 513–517.

Moon Jung Chung. 1987. $O(n^{2.5})$ time algorithms for the subgraph homeomorphism problem on trees. *J. Algorithms* 8, 1 (1987), 106–112.

L. De Moura and N. Bjørner. 2008. Z3: An Efficient SMT Solver. In *TACAS/ETAPS*. 4.

I. Eichwald, A. Bartel, J. Kiermaier, S. Breitkreutz, G. Csaba, D. Schmitt-Landsiedel, and M. Becherer. 2012. Nanomagnetic Logic: Error-Free, Directed Signal Transmission by an Inverter Chain. *IEEE Trans. Magn.* 48, 11 (2012), 4332–4335.

E. Fazzion, O. L. Fonseca, J. A. M. Nacif, O. P. V. Neto, A. O. Fernandes, and D. S. Silva. 2014. A quantum-dot cellular automata processor design. In *SBCCI*.

Steven Fortune, John Hopcroft, and James Wyllie. 1980. The directed subgraph homeomorphism problem. *Theor. Comput. Sci.* 10, 2 (1980), 111–121.

Michael R. Garey and David S. Johnson. 1979. *Computers and Intractability – A Guide to the Theory of NP-completeness*. W. H. Freeman.

D. Giri, G. Causapruno, and F. Riente. 2018. Parallel and Serial Computation in Nanomagnet Logic: An Overview. *TVLSI* (2018), 1–11.

D. Giri, M. Vacca, G. Causapruno, M. Zamboni, and M. Graziano. 2016. Modeling, Design, and Analysis of MagnetoElastic NML Circuits. *IEEE Transactions on Nanotechnology* 15, 6 (Nov 2016), 977–985.

K. Hennessy and C. S. Lent. 2001. Clocking of molecular quantum-dot cellular automata. *J. Vac. Sci. Technol. B* 19, 5 (2001), 1752–1755.

J. Huang, M. Momenzadeh, L. Schiano, M. Ottavi, and F. Lombardi. 2005. Tile-based QCA Design Using Majority-like Logic Primitives. *JETC* 1, 3 (2005), 163–185.

Taleana R. Huff, Hatem Labidi, and others. 2017. Atomic White-Out: Enabling Atomic Circuitry through Mechanically Induced Bonding of Single Hydrogen Atoms to a Silicon Surface. *ACS Nano* 11, 9 (2017), 8636–8642.

Alon Itai, Christos H. Papadimitriou, and Jayme Luiz Szwarcfiter. 1982. Hamilton paths in grid graphs. *SIAM J. Comput.* 11, 4 (1982), 676–686.

M. Kianpour and R. Sabbaghi-Nadooshan. 2014. A novel quantum-dot cellular automata CLB of FPGA. *J. Comput. Electron.* 13, 3 (2014), 709–725.

Andrea S. LaPaugh and Ronald L. Rivest. 1980. The subgraph homeomorphism problem. *J. Comput. Syst. Sci.* (1980), 133 – 149.

C. S. Lent and P. D. Tougaw. 1997. A device architecture for computing with quantum dots. *Proc. IEEE* 85, 4 (1997), 541–557.

Andrzej Lingas and Martin Wahlen. 2009. An exact algorithm for subgraph homeomorphism. *JDA* 7, 4 (2009), 464–468.

Jiří Matoušek and Robin Thomas. 1992. On the complexity of finding iso-and other morphisms for partial k-trees. *Discrete Math.* 108, 1–3 (1992), 343–364.

R. K. Nath, B. Sen, and B. K. Sikdar. 2017. Optimal synthesis of QCA logic circuit eliminating wire-crossings. *IET-CDS* 11, 3 (2017), 201–208.

Tobias Nipkow, Lawrence C. Paulson, and Markus Wenzel. 2002. *Isabelle/HOL: a proof assistant for higher-order logic*. Vol. 2283. Springer Science & Business Media.

S. Perri and P. Corsonello. 2012. New Methodology for the Design of Efficient Binary Addition Circuits in QCA. *TNANO* 11, 6 (2012), 1192–1200.

Wolfgang Porod, Gary H. Bernstein, György Csaba, Sharon X. Hu, Joseph Nahas, Michael T. Niemier, and Alexei Orlov. 2014. *Nanomagnet Logic (NML)*. Springer Berlin Heidelberg, Berlin, Heidelberg, 21–32.

D. A. Reis, C. A. T. Campos, T. R. Soares, O. P. V. Neto, and F. S. Torres. 2016. A Methodology for Standard Cell Design for QCA. In *ISCAS*.

Arman Roohi, Ronald F. DeMara, and Navid Khoshavi. 2015. Design and evaluation of an ultra-area-efficient fault-tolerant QCA full adder. *Microelectron. J.* 46, 6 (2015), 531–542.

S. Srivastava, S. Sarkar, and S. Bhanja. 2009. Estimation of Upper Bound of Power Dissipation in QCA Circuits. *IEEE TNANO* 8, 1 (Jan 2009), 116–127.

J. Timler and C. S. Lent. 2002. Power gain and dissipation in quantum-dot cellular automata. *J. Appl. Phys.* 91, 2 (2002), 823–831.

Frank Sill Torres, Marcel Walter, Robert Wille, Daniel Große, and Rolf Drechsler. 2018a. Synchronization of Clocked Field-Coupled Circuits. In *IEEE-NANO*. 1–4.

Frank Sill Torres, Robert Wille, Philipp Niemann, and Rolf Drechsler. 2018b. An Energy-aware Model for the Logic Synthesis of Quantum-Dot Cellular Automata. *IEEE TCAD* 37, 12 (December 2018), 3031–3041.

Frank Sill Torres, Robert Wille, Marcel Walter, Philipp Niemann, Daniel Große, and Rolf Drechsler. 2018c. Evaluating the Impact of Interconnections in Quantum-Dot Cellular Automata. In *DSD*. 649–656.

A. Trindade, R. S. Ferreira, J. A. M. Nacif, D. Sales, and O. P. V. Neto. 2016. A Placement and routing algorithm for Quantum-dot Cellular Automata. In *SBCCI*.

V. Vankamamidi, M. Ottavi, and F. Lombardi. 2008. Two-Dimensional Schemes for Clocking/Timing of QCA Circuits. *TCAD* 27, 1 (2008), 34–44.

E. Varga, M. T. Niemier, G. Csaba, G. H. Bernstein, and W. Porod. 2013. Experimental Realization of a Nanomagnet Full Adder Using Slanted-Edge Magnets. *IEEE Trans. Magn.* 49, 7 (July 2013), 4452–4455.

Marcel Walter, Robert Wille, Daniel Grosse, Frank Sill Torres, and Rolf Drechsler. 2018. An Exact Method for Design Exploration of Quantum-dot Cellular Automata. In *DATE*. 503–508.

Marcel Walter, Robert Wille, Frank Sill Torres, Daniel Große, and Rolf Drechsler. 2019. Scalable Design for Field-coupled Nanocomputing Circuits. In *ASP-DAC*.

Robert A. Wolkow, Lucian Livadaru, and others. 2014. *Silicon Atomic Quantum Dots Enable Beyond-CMOS Electronics*. Springer-Verlag, 33–58.