The Complexity of Determining Critical Sets

Fermi Ma, Ariel Schvartzman, and Erik Waingarten

MIT,
77 Mass Ave., Cambridge, MA 02139, USA,
{fermima,arielsc,eaw}@mit.edu

1 SAT

- f is the Cook-Levin construction, and add variables $a_{l,q,i,j,k}$ to indicate whether took left branch at time k from state q and wrote at i character j.
- C'_B is the set of clues involving only true values of $a_{l,q,i,j,k}$ and full solutions.
- M takes variables which are false and finds the ones that are true. This is done in polytime since a variable being true, implies one being false. We know that one must be true, so we can look for it. M sets $a_{l,q,i,j,k}$ while removing other variables.
- g sends $a_{l,q,i,j,k}$ which is true into took l branch at time k.

Note that algorithm M takes minimal to minimal since any false variable can be replaced, and any non-a variable can be replaced with an a variable which implies it.

$g$is surjective, since for each right-branch, left-branch, there exists a set of a's set to True. g is bijective on the solutions since Cook-Levin is parsimonious. If $c_1 \subset c_2$, then $g(c_1) \subset g(c_2)$ since g maps each element for c.

2 3SAT

- f takes all clauses like this: if below 4 variables, then the obvious. If clause C has n variables, $x_1, ..., x_n$, then map to

$$(x_1 \vee x_2 \vee z_2) \wedge (\overline{z_2} \vee \overline{x_2}) \wedge (z_2 \vee x_2) \wedge f(\overline{z_2}, x_3, ..., x_n)$$

- C'_B is the set of clues involving x_i and full solutions.
- M: switch z_i to x_i.
- g projects

Note that f works, and its parsimonious. M takes a minimal to a minimal since we add an element and remove an element.

g is actually bijective, since $x_i \leftrightarrow z_i$. Subsets hold since g maps individual elements.

2.1 NAE-3SAT

Note that the same reduction holds, since the instance created will never have all the elements in a clause set to true.

2.2 1-in-3 SAT

Also, we if we add the modification to the above reduction where instead of $(x_1 \vee x_2 \vee z_2)$, we write $(x_1 \vee z_1) \wedge (\overline{z_1} \vee \overline{x_1}) \wedge (\overline{z_1} \vee x_2 \vee z_2)$, then we never have more than one thing set to true in each clause.

Therefore, this reduction would show that FCP 1-in-3SAT is FCP-hard.

3 Triangle Partition

- f is Holyer construction.
- C_B is the clues with at most one particular triangle per $H_{3,p}$ graph, not in unions, the first true literal of each clause is set to F-partition.
- M translates the triangles to particular triangle. It also moves the triangles in the clauses so that the first true literal is set F-partition.
- g projects variables.

Note that M is fine becuase if a clause is set to T partition, it doesn't add information, so we put whichever literal to F partition that we need.

g is a surjection since it projects. g is also a bijection on solutions since we defined designated how C'_B will assign clauses in light of ambiguity. g preserves subsets since each element maps to another element.

4 Latin Squares

In Latin Squares we are given a $n \times n$ grid, some of whose entries have been filled in with numbers from 1 to n. The goal of the game is to fill the rest of the grid with numbers from 1 to n while enforcing that no row or column repeats a number. It is known that such problem is NP-complete and it can be shown that its FCP version is $\mathsf{FCP} - complete$.

Theorem 1. FCP *Latin Squares is* FCP*-complete.*

Proof. We do this via reduction from FCP TrianglePartition. Throughout this proof we will be using constructions and results from [?].

Given a tripartite graph G, we first check whether or not it is uniform. If it is not uniform, then there does not exist a triangle partition. If it is uniform, we can write down a Latin framework $LF(G; 2n, 2n, 2n)$ in polynomial time [?]. This Latin framework is constructed so that G is the defect graph of $LF(G; 2n, 2n, 2n)$. We then simply map via f instances of defect graphs G to partial Latin Squares. The Latin Squares clues are mapped to triangles with vertices on different parts of the partition. Each clue is mapped, by construction, bijectively to a triangle (since the triangle takes into account the row, column and number of the clue). As Colbourn shows, there is a triangle partition of G if and only if we can complete the partial Latin Square $LF(G; 2n, 2n, 2n)$. This implies that the triangle partitions of a defect graph G are in one to one correspondence with the solutions to its Latin Square. This completes the proof. □

5 Sudoku

In Sudoku, or Number Place, we are given a $n^2 \times n^2$ grid consisting of n^2 $n \times n$ bolded blocks, some of whose entries have been filled in with number from 1 to n^2. The goal of the game is to fill in the rest of the grid with numbers from 1 to n^2 while enforcing that no row, column or bolded square repeats a numbers. We will show this problem is FCP-Complete by providing a reduction from FCP-Latin Squares.

Before showing the main result of this section, we will exhibit a particular way to fill a Sudoku puzzle. Next, we will show how this particular solution can be used to explicitly map instances of Sudoku to instances of Latin Squares. For simplicity, throughout this argument we will be using numbers in the range 0 through $n^2 - 1$. We will refer to the entry on the i-th row and j-th column of a Number Place puzzle as $S(i, j)$.

Proposition 1. *Let S_0 be defined as*

$$S_0(i,j) = ((i \mod n)n + \lfloor i/n \rfloor + j) \mod n^2.$$

Then S_0 represents a solution to an order n Number Place.

Proof. Omitted. See [Yato and Seta] □

Lemma 2. *Let S be a Sudoku instance of order n such that*

$$S(i,j) = \begin{cases} \perp & : (i,j) \in B \\ S_0(i,j) & : \textit{otherwise} \end{cases}$$

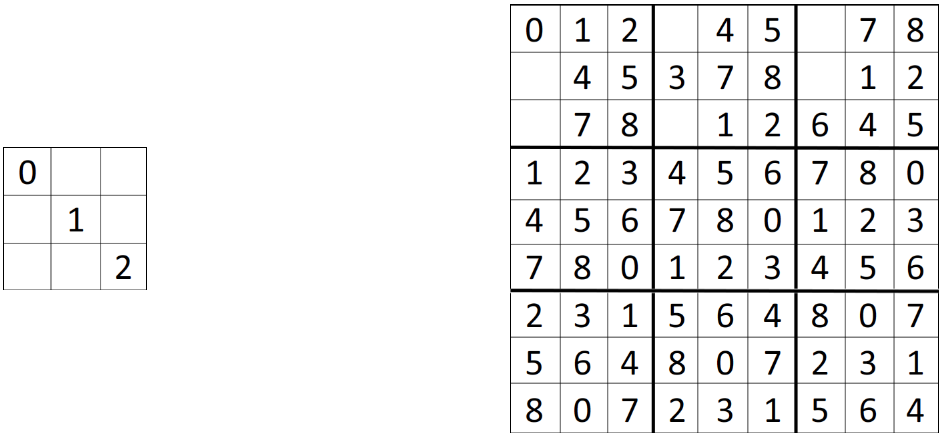
where $B = \{(i,j) | i < n \text{ and } (j \equiv 0) \mod n\}$. Then a square S' obtained by filling in the blanks of S is a solution to S if and only if

- *For any $(i,j) \in B$, $S'(i,j) \equiv 0 \mod n$*
- *A square L defined by $L(i, j/n) = S'(i,j)/n$ for all $(i,j) \in B$ is a Latin Square.*

Proof. Omitted. See [Yato and Seta] □

Lemma 3. *FCP-Sudoku is FCP-Complete*

Proof. We will show a reduction via FCP-Latin Squares. We use the injective map from instances of Latin Square to instances of Sudoku described above. Let C_B be the set of set of clues of Sudoku. Then we can let $g(C_B) = C_B$, mapping the clues from $S(i,j)$ (where the $(i,j) \in B$) to $L(i, j/n)$.This mapping is surjective, bijective on solutions by the lemma above and preserves subsets. Therefore, FCP-Sudoku is FCP-Complete. □



0		
	1	
		2

0	1	2		4	5		7	8
	4	5	3	7	8		1	2
	7	8		1	2	6	4	5
1	2	3	4	5	6	7	8	0
4	5	6	7	8	0	1	2	3
7	8	0	1	2	3	4	5	6
2	3	1	5	6	4	8	0	7
5	6	4	8	0	7	2	3	1
8	0	7	2	3	1	5	6	4

Fig. 1. Example of a mapping between instances of Latin Squares and Sudoku.

6 3-colorability

We have a reduction from 3SAT.

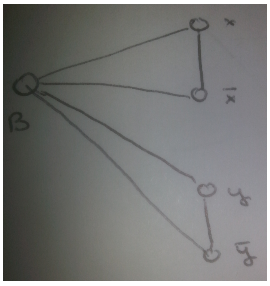


Fig. 2. variable gadget

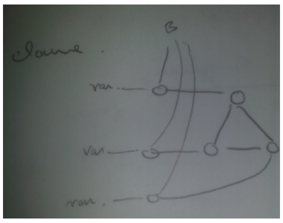


Fig. 3. clause gadget

Now we can do a similar thing as in triangle partition and make the F colors in the clause be the first ones written out. We also note that this throws away information, so it makes sense that it is still FCP-hard. Furthermore, if the formula is planar, then the graph can be turned planar, with the following transformation, where also we keep alternating up between $x, \overline{x}, x, \overline{x}, ...$ so that each appearence connects to one, and since these are connected in a path (also all to B), they will have the same color.

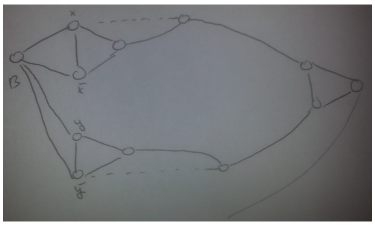


Fig. 4. making the reduction planar

7 Bag - Corral

If we assume that planar 3-coloring is FCP-hard, then Corral is FCP-hard. The clues here are lists of edges that are included in the graph. We can use the reduction from http://www2.stetson.edu/~efriedma/papers/corral/corral.html and note that since for each wire there are three possible fillings, the clues can be moved, and each implies the next, until we get to a turn (in particular, this corresponds to the edges that border the colors, but not true ones. Therefore, we can move the final solution to the values of the vertices. And so we can three-color the graph, since in a vertex, we know how to color them since we know which edge starts the signal.

8 Indepedent set

Trivial. Using reduction from https://www.cs.cmu.edu/~ckingsf/bioinfo-lectures/sat.pdf