

# Learning a Boolean function

伊藤 碧己

2024 年 5 月 27 日

**Learning a Boolean function.** Sometimes we're given a "black box" that evaluates a Boolean function  $f(x_1, \dots, x_N)$ . We have no way to open the box, but we suspect that the function is actually quite simple. By plugging in various values for  $x = x_1 \dots x_N$ , we can observe the box's behavior and possibly learn the hidden rule that lies inside. For example, a secret function of  $N = 20$  Boolean variables might take on the values shown in Table 2, which lists 16 cases where  $f(x) = 1$  and 16 cases where  $f(x) = 0$ .

**ブール関数の学習.** 私たちは時々"ブラックボックス"を与えられる、ブール関数  $f(x_1, \dots, x_N)$  を評価する。私たちはその箱を開ける方法を持たない、しかし私たちは考える、その関数は実際には非常に単純なものであると。 $x = x_1 \dots x_N$  に様々な値を代入することで、私たちはその箱の振る舞いを観察することができ、その中にある隠れたルールを学ぶことができるかもしれない。例えば、 $N = 20$  のブール変数の秘密の関数は表 2 に示すような値をとるかもしれない、 $f(x) = 1$  である 16 のケースと  $f(x) = 0$  である 16 のケースが列挙されている。

Table 2  
VALUES TAKEN ON BY AN UNKNOWN FUNCTION

Cases where  $f(x) = 1$

Cases where  $f(x) = 0$

$x_1$	$x_2$	$x_3$	$x_4$	$x_5$	$x_6$	$x_7$	$x_8$	$x_9$	$\dots$	$x_{20}$
1	1	0	0	1	0	0	1	0	1	1
1	0	1	0	1	0	0	0	1	0	0
0	1	1	0	1	0	0	1	0	0	1
0	1	0	1	1	0	0	0	1	0	0
0	1	0	0	1	1	0	0	0	1	0
0	1	0	0	1	0	0	1	0	0	1
0	1	1	0	0	1	0	0	1	1	0
0	1	1	0	0	0	1	0	1	1	0
0	0	0	0	1	0	1	1	0	0	0
0	0	0	0	1	0	0	0	0	1	1
1	1	0	1	0	0	1	0	1	0	0
0	0	1	0	0	1	1	0	0	0	0
0	0	1	0	1	0	0	1	1	1	0
1	0	0	1	0	1	0	0	0	1	0
1	1	0	0	1	1	0	1	0	0	0
0	0	0	1	0	1	1	1	1	0	1
0	1	1	0	0	1	1	0	0	1	1
1	0	0	1	1	0	0	1	0	0	1
0	0	0	1	0	0	1	0	0	0	0
0	1	1	1	0	0	1	1	0	0	1
0	1	0	0	0	0	0	1	0	1	1

Suppose we assume that the function has a DNF (disjunctive normal form) with only a few terms. We'll see in a moment that it's easy to express such an assumption as a satisfiability problem. And when the author constructed clauses corresponding to Table 2 and presented them to a SAT solver, he did in fact learn

almost immediately that a very simple formula is consistent with all of the data:

$$f(x_1, \dots, x_{20}) = \bar{x}_2 \bar{x}_3 \bar{x}_{10} \vee \bar{x}_6 \bar{x}_{10} \bar{x}_{12} \vee x_8 \bar{x}_{13} \bar{x}_{15} \vee \bar{x}_8 x_{10} \bar{x}_{12}. \quad (27)$$

私たちはその関数が DNF (選言標準形) であると仮定するとしよう、いくつかの項のみを持つ。私たちはすぐに分かる、このような仮定を充足可能性問題として表現するのは簡単である。そして、筆者が表 2 に対応する節を構築し、SAT ソルバーに提示したところ、彼は実際に、ほとんど即座に知った、非常に単純な式がすべてのデータと一致すると：

$$f(x_1, \dots, x_{20}) = \bar{x}_2 \bar{x}_3 \bar{x}_{10} \vee \bar{x}_6 \bar{x}_{10} \bar{x}_{12} \vee x_8 \bar{x}_{13} \bar{x}_{15} \vee \bar{x}_8 x_{10} \bar{x}_{12} \quad (27)$$

This formula was discovered by constructing clauses in  $2MN$  variables  $p_{i,j}$  and  $q_{i,j}$  for  $1 \leq i \leq M$  and  $1 \leq j \leq N$ , where  $M$  is the maximum number of terms allowed in the DNF (here  $M = 4$ ) and where

$$p_{i,j} = [\text{term } i \text{ contains } x_j], \quad q_{i,j} = [\text{term } i \text{ contains } \bar{x}_j]. \quad (28)$$

この式は、 $2MN$  の変数  $p_{i,j}$  と  $q_{i,j}$  で節を構成することによって発見された、ここで  $1 \leq i \leq M$  かつ  $1 \leq j \leq N$  である、 $M$  は DNF で許される項の最大個数（ここでは  $M = 4$ ）である、また

$$p_{i,j} = [\text{項 } i \text{ は } x_j \text{ を含む}], \quad q_{i,j} = [\text{項 } i \text{ は } \bar{x}_j \text{ を含む}] \quad (28)$$

If the function is constrained to equal 1 at  $P$  specified points, we also use auxiliary variables  $z_{i,k}$  for  $1 \leq i \leq M$  and  $1 \leq k \leq P$ , one for each term at every such point.

Table 2 says that  $f(1, 1, 0, 0, \dots, 1) = 1$ , and we can capture this specification by constructing the clause

$$(z_{1,1} \vee z_{2,1} \vee \dots \vee z_{M,1}) \quad (29)$$

together with the clauses

$$(\bar{z}_{i,1} \vee \bar{q}_{i,1}) \wedge (\bar{z}_{i,1} \vee \bar{q}_{i,2}) \wedge (\bar{z}_{i,1} \vee \bar{p}_{i,3}) \wedge (\bar{z}_{i,1} \vee \bar{p}_{i,4}) \wedge \dots \wedge (\bar{z}_{i,1} \vee \bar{q}_{i,20}) \quad (30)$$

for  $1 \leq i \leq M$ . Translation: (29) says that at least one of the terms in the DNF must evaluate to true; and (30) says that, if term  $i$  is true at the point 1100...1, it cannot contain  $\bar{x}_1$  or  $\bar{x}_2$  or  $x_3$  or  $x_4$  or  $\dots$  or  $\bar{x}_{20}$ .

もし関数が  $P$  個の特定の点で 1 に等しくなるように制約されているなら, <sup>\*1</sup>加えて補助変数  $z_{i,k}$  を用いる,  $1 \leq i \leq M$ かつ  $1 \leq k \leq P$ , そのような全ての点について各項に 1 つずつ.

表 2 は,  $f(1, 1, 0, 0, \dots, 1) = 1$  であると述べており, 私たちはこの指定を捕らえることができる, 次の節を構成することによって

$$(z_{1,1} \vee z_{2,1} \vee \dots \vee z_{M,1}) \quad (29)$$

次の節たちとともに

$$(\bar{z}_{i,1} \vee \bar{q}_{i,1}) \wedge (\bar{z}_{i,1} \vee \bar{q}_{i,2}) \wedge (\bar{z}_{i,1} \vee \bar{p}_{i,3}) \wedge (\bar{z}_{i,1} \vee \bar{p}_{i,4}) \wedge \dots \wedge (\bar{z}_{i,1} \vee \bar{q}_{i,20}) \quad (30)$$

$1 \leq i \leq M$  に対して. 注釈: (29) は示す, DNF の項のうち少なくとも 1 つは真と評価されなければならない; そして (30) は示す, もし項  $i$  が点 1100...1 で真ならば, それは  $\bar{x}_1$  または  $\bar{x}_2$  または  $x_3$  または  $x_4$  または... または  $\bar{x}_{20}$  を含むことはできない.

---

<sup>\*1</sup>  $f(x_1, x_2, \dots, x_n) = 1$  となるブール変数の組み合わせが  $P$  個ある, という意味

Table 2 also tells us that  $f(1, 0, 1, 0, \dots, 1) = 0$ . This specification corresponds to the clauses

$$(q_{i,1} \vee p_{i,2} \vee q_{i,3} \vee p_{i,4} \vee \dots \vee q_{i,20}) \quad (31)$$

for  $1 \leq i \leq M$ . (Each term of the DNF must be zero at the given point; thus either  $\bar{x}_1$  or  $x_2$  or  $\bar{x}_3$  or  $x_4$  or  $\dots$  or  $\bar{x}_{20}$  must be present for each value of  $i$ .)

In general, every case where  $f(x) = 1$  yields one clause like (29) of length  $M$ , plus  $MN$  clauses like (30) of length 2. Every case where  $f(x) = 0$  yields  $M$  clauses like (31) of length  $N$ . We use  $q_{i,j}$  when  $x_j = 1$  at the point in question,

and  $p_{i,j}$  when  $x_j = 0$ , for both (30) and (31). This construction is due to A. P. Kamath, N. K. Karmarkar, K. G. Ramakrishnan, and M. G. C. Resende [Mathematical Programming 57 (1992), 215–238], who presented many examples. From Table 2, with  $M = 4$ ,  $N = 20$ , and  $P = 16$ , it generates 1360 clauses of total length 3904 in 224 variables; a SAT solver then finds a solution with  $p_{1,1} = q_{1,1} = p_{1,2} = 0$ ,  $q_{1,2} = 1, \dots$ , leading to (27).

表2はまた、 $f(1, 0, 1, 0, \dots, 1) = 0$ であることも示している。この指定は対応する、節

$$(q_{i,1} \vee p_{i,2} \vee q_{i,3} \vee p_{i,4} \vee \dots \vee q_{i,20}) \quad (31)$$

に。 $1 \leq i \leq M$ に対して。(DNFの各項は与えられた点で0でなければならない；それゆえ $\bar{x}_1$  or  $x_2$  or  $\bar{x}_3$  or  $x_4$  or  $\dots$  or  $\bar{x}_{20}$ のいずれかが存在しなければならない、 $i$ の各値に対して。

一般に、 $f(x) = 1$ となる全ての場合で生じる、(29)のような長さ  $M$ の節が1つ、加えて(30)のような長さ2の節が  $MN$ 。 $f(x) = 0$ となる全ての場合で生じる、(31)のような長さ  $N$ の節が  $M$ 。我々は  $x_j = 1$  の時  $q_{i,j}$  を用いる、この問題の中でその点において、かつ  $x_j = 0$  の時  $p_{i,j}$  を、(30)及び(31)の両方に。この構成は A. P. Kamath, N. K. Karmarkar, K. G. Ramakrishnan, and M. G. C. Resende (Mathematical Programming 57 (1992), 215-238)によるものである、(彼らは)多くの例を示している。表2より、 $M = 4$ ,  $N = 20$ ,  $P = 16$ で、それは 1360 の節を生み出す、合計の長さ 3904 で、224 の変数を持つ；SAT ソルバーはその時解を求める、 $p_{1,1} = q_{1,1} = p_{1,2} = 1, \dots, (27)$ を導く。

The simplicity of (27) makes it plausible that the SAT solver has indeed psyched out the true nature of the hidden function  $f(x)$ . The chance of agreeing with the correct value 32 times out of 32 is only 1 in  $2^{32}$ , so we seem to have overwhelming evidence in favor of that equation.

(27)の単純性はそれを有力にする、SAT ソルバーが確かに隠れた関数  $f(x)$  の本質を見

抜いたことを. 32 回中 32 回正しい値に一致する確率は  $2^{32}$  分の 1 なので, 我々は圧倒的な証拠を持っていると言えるだろう, この等式を支持する.

But no: Such reasoning is fallacious. The numbers in Table 2 actually arose in a completely different way, and Eq. (27) has essentially *no* credibility as a predictor of  $f(x)$  for any other values of  $x$ ! (See exercise 53.) The fallacy comes from the fact that short-DNF Boolean functions of 20 variables are not at all rare; there are many more than  $2^{32}$  of them.

しかしそうではない: そのような推論は誤りである. 表 2 の値は, 実際にはまったく異なる方法と式により生じたものであり, (27) は,  $f(x)$  の予測因子として本質的に何の信頼もない, 他のどのような  $x$  の値に対しても. (練習問題 53 を参照) この誤りは, 20 変数の short-DNF ブール関数が全く珍しいものではなく;  $2^{32}$  個より多く存在するという事実に起因する. 一方で, 隠れた関数  $f(x)$  が最大  $M$  個の項を持つ DNF を持つことが分

On the other hand, when we *do* know that the hidden function  $f(x)$  has a DNF with at most  $M$  terms (although we know nothing else about it), the clauses (29)–(31) give us a nice way to discover those terms, provided that we also have a sufficiently large and unbiased “training set” of observed values.

かっている場合 (しかしそれ以外には何も分かっていない), 節 (29)–(31) は, 我々にそれらの項を発見する良い方法を与えてくれる, 十分に大きく偏りのない, 値を観測する”訓練セット”を持っていれば.

For example, let's assume that (27) actually *is* the function in the box. If we examine  $f(x)$  at 32 random points  $x$ , we don't have enough data to make any deductions. But 100 random training points will almost always home in on the correct solution (27). This calculation typically involves 3942 clauses in 344 variables; yet it goes quickly, needing only about 100 million accesses to memory.

例えば, (27) が実際に箱の中の関数であると仮定しよう. もし我々が 32 のランダムな点  $x$  で  $f(x)$  を調べても, 我々は十分なデータを持っていない, どんな推論を行うにも. しかし 100 のランダムな学習点はほとんどいつも正しい解 (27) にたどり着くだろう. この計算には通常 344 の変数と, 3942 の節が関わっている; しかしそれは早く, 約 1 億回のメモリへのアクセスのみを必要とする.

One of the author's experiments with a 100-element training set yielded

$$\hat{f}(x_1, \dots, x_{20}) = \bar{x}_2 \bar{x}_3 \bar{x}_{10} \vee x_3 \bar{x}_6 \bar{x}_{10} \bar{x}_{12} \vee x_8 \bar{x}_{13} \bar{x}_{15} \vee \bar{x}_8 x_{10} \bar{x}_{12}, \quad (32)$$

which is close to the truth but not quite exact. (Exercise 59 proves that  $\hat{f}(x)$  is equal to  $f(x)$  more than 97% of the time.) Further study of this example showed that another nine training points were enough to deduce  $f(x)$  uniquely, thus obtaining 100% confidence (see exercise 61).

100 要素のトレーニングセットを用いた筆者の実験の 1 つは生み出した

$$\hat{f}(x_1, \dots, x_{20}) = \bar{x}_2 \bar{x}_3 \bar{x}_{10} \vee x_3 \bar{x}_6 \bar{x}_{10} \bar{x}_{12} \vee x_8 \bar{x}_{13} \bar{x}_{15} \vee \bar{x}_8 x_{10} \bar{x}_{12}, \quad (32)$$

これは真実に近いが完全には正確でない。 (練習問題 59 は  $\hat{f}(x)$  が 97% 以上の  $f(x)$  と等しいことを示す。) この例題に対するさらなる研究は示した、他の 9 個の訓練点は  $f(x)$  を一意に推論するのに十分であり、従って 100 % の信頼性が得られた (練習問題 61 参照)。

(本文のアルゴリズムを実際に使うプログラム:learning.py)

実行例:python learning.py input\_table2\_eq1.txt input\_table2\_eq0.txt 4

## Exercise 53

### 問題

► 53. [M20] The numbers in Table 2 are definitely nonrandom. Can you see why?

表 2 の数字は明確に非ランダムである。あなたは何故か分かるだろうか。

### knuth 先生の解答

53. On the left is the binary expansion of  $\pi$ , and on the right is the binary expansion of  $e$ , 20 bits at a time (see Appendix A).

One way to define  $f(x)$  for all 20-bit  $x$  is to write  $\pi/4 = \sum_{k=1}^{\infty} u_k / 2^{20k}$  and  $e/4 = \sum_{l=1}^{\infty} v_l / 2^{20l}$ , where each  $u_k$  and  $v_l$  is a 20-bit number. Let  $k$  and  $l$  be smallest such that  $x = u_k$  and  $x = v_l$ . Then  $f(x) = [k \leq l]$ .

Equation (27) has actually been contrived to *sustain* an illusion of magic: Many simple Boolean functions are consistent with the data in Table 2, even if we require four-term DNFs of three literals each. But only two of them, like (27), have the additional property that they actually agree with the definition of  $f(x)$  in the previous paragraph for ten more cases, using  $u_k$  up to  $k = 22$  and  $v_l$  up to  $l = 20$ ! One might almost begin to suspect that a SAT solver has discovered a deep new connection between  $\pi$  and  $e$ .

左側は  $\pi$  の 2 進展開, 右側は  $e$  の 2 進展開, 一度に 20 ビットずつ. (付録 A 参照).

すべての 20 ビットの  $x$  に対して  $f(x)$  を定義する 1 つの方法は,  $\pi/4 = \sum_{k=1}^{\infty} u_k / 2^{20k}$  と  $e/4 = \sum_{l=1}^{\infty} v_l / 2^{20l}$  を記述することである, ここで各  $u_k$  と  $v_l$  は 20 ビットの数である.  $x = u_k$  かつ  $x = v_l$  となるような  $k$  と  $l$  を最小にしてみよう. その時,  $f(x) = [k \leq l]$

式 (27) は、実際には魔法のような錯覚を持続させるために考案されたものである：多くの単純なブール関数は表 2 のデータと一致する, 3 リテラルずつの 4 項の DNF を必要とするとしても. しかし, そのうち 2 つだけが, (27) のように, 追加的な性質を持っている, それらが  $k = 22$  までの  $u_k$  と  $l = 20$  までの  $v_l$  を使った 10 以上のケースについて, 前の段落の  $f(x)$  の定義と実際に一致するという. 誰かは疑い始めてしまうかもしれない, SAT ソルバーが  $\pi$  と  $e$  の間に新しく深い関係を発見したのではないかと.

## APPENDIX A

# TABLES OF NUMERICAL QUANTITIES

**Table 1**

QUANTITIES THAT ARE FREQUENTLY USED IN STANDARD SUBROUTINES  
AND IN ANALYSIS OF COMPUTER PROGRAMS (40 DECIMAL PLACES)

---

$\sqrt{2} = 1.41421\ 35623\ 73095\ 04880\ 16887\ 24209\ 69807\ 85697-$
$\sqrt{3} = 1.73205\ 08075\ 68877\ 29352\ 74463\ 41505\ 87236\ 69428+$
$\sqrt{5} = 2.23606\ 79774\ 99789\ 69640\ 91736\ 68731\ 27623\ 54406+$
$\sqrt{10} = 3.16227\ 76601\ 68379\ 33199\ 88935\ 44432\ 71853\ 37196-$
$\sqrt[3]{2} = 1.25992\ 10498\ 94873\ 16476\ 72106\ 07278\ 22835\ 05703-$
$\sqrt[3]{3} = 1.44224\ 95703\ 07408\ 38232\ 16383\ 10780\ 10958\ 83919-$
$\sqrt[4]{2} = 1.18920\ 71150\ 02721\ 06671\ 74999\ 70560\ 47591\ 52930-$
$\ln 2 = 0.69314\ 71805\ 59945\ 30941\ 72321\ 21458\ 17656\ 80755+$
$\ln 3 = 1.09861\ 22886\ 68109\ 69139\ 52452\ 36922\ 52570\ 46475-$
$\ln 10 = 2.30258\ 50929\ 94045\ 68401\ 79914\ 54684\ 36420\ 76011+$
$1/\ln 2 = 1.44269\ 50408\ 88963\ 40735\ 99246\ 81001\ 89213\ 74266+$
$1/\ln 10 = 0.43429\ 44819\ 03251\ 82765\ 11289\ 18916\ 60508\ 22944-$
$\pi = 3.14159\ 26535\ 89793\ 23846\ 26433\ 83279\ 50288\ 41972-$
$1^\circ = \pi/180 = 0.01745\ 32925\ 19943\ 29576\ 92369\ 07684\ 88612\ 71344+$
$1/\pi = 0.31830\ 98861\ 83790\ 67153\ 77675\ 26745\ 02872\ 40689+$
$\pi^2 = 9.86960\ 44010\ 89358\ 61883\ 44909\ 99876\ 15113\ 53137-$
$\sqrt{\pi} = \Gamma(1/2) = 1.77245\ 38509\ 05516\ 02729\ 81674\ 83341\ 14518\ 27975+$
$\Gamma(1/3) = 2.67893\ 85347\ 07747\ 63365\ 56929\ 40974\ 67764\ 41287-$
$\Gamma(2/3) = 1.35411\ 79394\ 26400\ 41694\ 52880\ 28154\ 51378\ 55193+$
$e = 2.71828\ 18284\ 59045\ 23536\ 02874\ 71352\ 66249\ 77572+$
$1/e = 0.36787\ 94411\ 71442\ 32159\ 55237\ 70161\ 46086\ 74458+$
$e^2 = 7.38905\ 60989\ 30650\ 22723\ 04274\ 60575\ 00781\ 31803+$
$\gamma = 0.57721\ 56649\ 01532\ 86060\ 65120\ 90082\ 40243\ 10422-$
$\ln \pi = 1.14472\ 98858\ 49400\ 17414\ 34273\ 51353\ 05871\ 16473-$
$\phi = 1.61803\ 39887\ 49894\ 84820\ 45868\ 34365\ 63811\ 77203+$
$e^\gamma = 1.78107\ 24179\ 90197\ 98523\ 65041\ 03107\ 17954\ 91696+$
$e^{\pi/4} = 2.19328\ 00507\ 38015\ 45655\ 97696\ 59278\ 73822\ 34616+$
$\sin 1 = 0.84147\ 09848\ 07896\ 50665\ 25023\ 21630\ 29899\ 96226-$
$\cos 1 = 0.54030\ 23058\ 68139\ 71740\ 09366\ 07442\ 97660\ 37323+$
$-\zeta'(2) = 0.93754\ 82543\ 15843\ 75370\ 25740\ 94567\ 86497\ 78979-$
$\zeta(3) = 1.20205\ 69031\ 59594\ 28539\ 97381\ 61511\ 44999\ 07650-$
$\ln \phi = 0.48121\ 18250\ 59603\ 44749\ 77589\ 13424\ 36842\ 31352-$
$1/\ln \phi = 2.07808\ 69212\ 35027\ 53760\ 13226\ 06117\ 79576\ 77422-$
$-\ln \ln 2 = 0.36651\ 29205\ 81664\ 32701\ 24391\ 58232\ 66946\ 94543-$

---

## 自身の解答

1.  $\pi/4 = \sum_{k=1}^{\infty} u_k/2^{20k}$  は、 $\pi/4$  を 2 進数に変換し、上の桁から 20 ビットずつ切り出すことを表している。
  - 実際に計算するプログラム:ex53\_1.py
  - 実行例:python ex53\_1.py -pi 320
2.  $x = u_k$  かつ  $x = v_l$  となるような  $k$  と  $l$  を探索する
  - プログラム:ex53\_2.py
  - 実行例:python ex53\_2.py 32000
3.  $k = 17$  22 の  $u_k$  と  $l = 17$  20 の  $v_l$  において表 2 の  $f(x)$  の定義と一致する (=すなわち,  $u_k$  の 20 ビットを  $x$  とすると  $f(x)=1$  となり,  $v_l$  の 20 ビットを  $x$  とすると  $f(x)=0$  となる)
4. プログラム:
5. 実行例:

## Exercise 59

59. [M20] Compute the exact probability that  $\hat{f}(x)$  in (32) differs from  $f(x)$  in (27).

59.  $f(x) \oplus \hat{f}(x) = x_2\bar{x}_3\bar{x}_6\bar{x}_{10}\bar{x}_{12}(\bar{x}_8 \vee x_8(x_{13} \vee x_{15}))$  is a function of eight variables that has 7 solutions. Thus the probability is  $7/256 = .02734375$ .

## Exercise 61

61. [20] Explain how to test when a set of clauses generated from a training set via (29)–(31) is satisfiable only by the function  $f(x)$  in (27).

**61.** We can add 24 clauses  $(p_{a,1} \vee q_{a,1} \vee p_{a,2} \vee \bar{q}_{a,2} \vee p_{a,3} \vee \bar{q}_{a,3} \vee \dots \vee p_{b,1} \vee q_{b,1} \vee \dots \vee p_{c,1} \vee q_{c,1} \vee \dots \vee p_{d,1} \vee q_{d,1} \vee \dots \vee \bar{p}_{d,10} \vee q_{d,10} \vee \dots \vee p_{d,20} \vee q_{d,20})$ , one for each permutation  $abcd$  of  $\{1, 2, 3, 4\}$ ; the resulting clauses are satisfiable only by other functions  $f(x)$ .

But the situation is more complicated in larger examples, because a function can have many equivalent representations as a short DNF. A general scheme, to decide whether the function described by a particular setting  $p'_{i,j}$  and  $q'_{i,j}$  of the  $p$ s and  $q$ s is unique, would be to add more complicated clauses, which state that  $p_{i,j}$  and  $q_{i,j}$  give a different solution. Those clauses can be generated by the Tseytin encoding of

$$\bigvee_{i=1}^M \bigwedge_{j=1}^N ((\bar{p}_{i,j} \wedge \bar{x}_j) \vee (\bar{q}_{i,j} \wedge x_j)) \oplus \bigvee_{i=1}^M \bigwedge_{j=1}^N ((\bar{p}'_{i,j} \wedge \bar{x}_j) \vee (\bar{q}'_{i,j} \wedge x_j)).$$