ランプ (Lamps)

解説：保坂

JOI **春合宿** Day 3　2019/03/22

問題

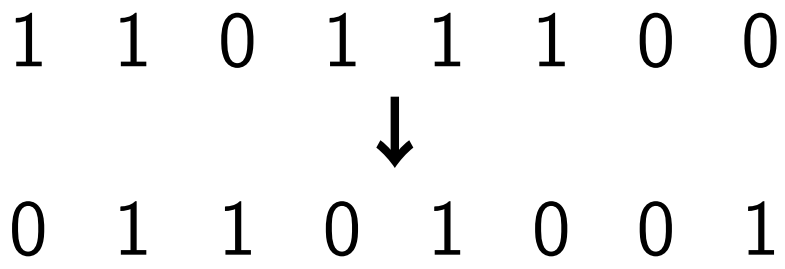
0, 1 からなる列に対する以下の 3 通りの操作で，
A から B にするための最小操作回数を求めよ

OFF 区間を選んで，全部 0 にする
ON 区間を選んで，全部 1 にする
TOGGLE 区間を選んで，0 を 1 に，1 を 0 にする

小課題

1. $N \leqq 18$
2. $N \leqq 2\,000$
3. $N \leqq 1\,000\,000$，A は全部 0
4. $N \leqq 1\,000\,000$

例 1



例 1

1 1 0 1 1 1 0 0

TOGGLE

0 0 1 0 1 1 0 0

ON

0 1 1 0 1 1 0 0

TOGGLE

0 1 1 0 1 0 1 1

OFF

0 1 1 0 1 0 0 1

単純な解法

全状態を探索する

- $0, 1$ **の状態は** 2^N **通り**
- **各状態からの遷移は** $O(N^2)$ **通り**

初期状態から各状態までの距離が BFS **で** $O(2^N N^2)$ **時間でわかる**

ビット演算

各操作を $O(1)$ 回の整数演算で行うために

区間 $[p, q)$ に対する操作

OFF `s & ~((1 << q) - (1 << p))`

ON `s | ((1 << q) - (1 << p))`

TOGGLE `s ^ ((1 << q) - (1 << p))`

特殊な場合

初期状態が全部 0 のとき

0	0	0	0	0	0	0	0	0	0	0	0
ON			ON						ON		
1	1	0	1	1	1	1	0	0	1	1	0

- **「連続する 1 の区間の個数」回の操作でできる (各区間で ON または TOGGLE をする)**
- **それ未満ではできない？**

下からの評価

“01” あるいは “10” と並んでいる場所は，1 回の操作で高々 2 箇所しか増えない

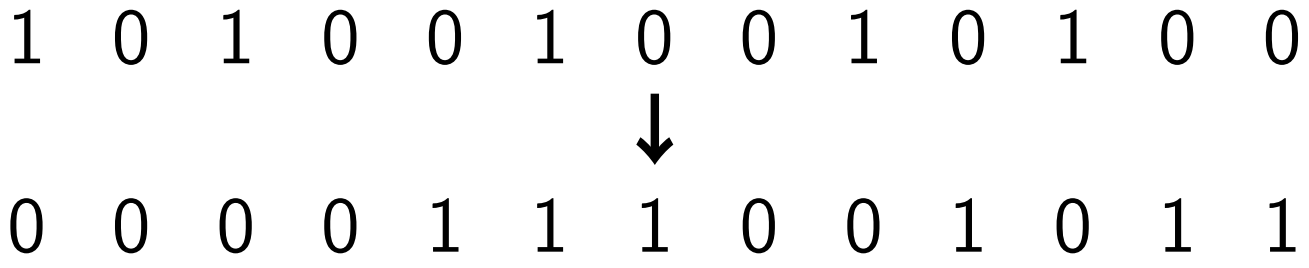
- 区間の左右の端の部分でしか増えないため
- 全体の左右の端には 0 が並んでいるとする

連続する 1 の区間を k 個作るには，“01” あるいは “10” と並ぶ場所を $2k$ 箇所作らないといけないので，k 回の操作は必要

満点解法に向けて

サンプルで考えてみよう

例 2



例 2

1 0 1 0 0 1 0 0 1 0 1 0 0

OFF

0 0 0 0 0 1 0 0 1 0 1 0 0

ON

0 0 0 0 1 1 1 0 1 0 1 0 0

TOGGLE

0 0 0 0 1 1 1 0 0 1 0 1 1

かんたん

例 3

0	0	1	1	0	0	0	1	0	0	1	0	0	0	0	1	1	0
									↓								
1	1	0	1	1	0	0	0	1	0	0	0	1	0	0	1	0	1

例 3

0 0 1 1 0 0 0 1 0 0 1 0 0 0 0 1 1 0

OFF

0 0 1 0 0 0 0 1 0 0 1 0 0 0 0 1 1 0

ON

0 0 1 0 0 1 1 1 0 0 1 0 0 0 0 1 1 0

ON

0 0 1 0 0 1 1 1 0 1 1 1 0 0 0 1 1 0

TOGGLE

1 1 0 1 1 0 0 0 1 0 0 0 1 0 0 1 1 0

TOGGLE

1 1 0 1 1 0 0 0 1 0 0 0 1 0 0 1 0 1

むずかしい

OFF, ON **による下準備と** TOGGLE **分けて考えるとよい？**

例 1 ふたたび

1 1 0 1 1 1 0 0

TOGGLE

0 0 1 0 1 1 0 0

ON

0 1 1 0 1 1 0 0

TOGGLE

0 1 1 0 1 0 1 1

OFF

0 1 1 0 1 0 0 1

これでもいいけど

例 1 ふたたび

1 1 0 1 1 1 0 0

OFF

1 0 0 1 1 1 0 0

TOGGLE

0 1 1 0 1 1 0 0

TOGGLE

0 1 1 0 1 0 1 1

OFF

0 1 1 0 1 0 0 1

こうしてもいいし

例 1 ふたたび

1 1 0 1 1 1 0 0

OFF

1 0 0 1 1 1 0 0

TOGGLE

0 1 1 0 1 1 0 0

ON

0 1 1 0 1 1 1 0

TOGGLE

0 1 1 0 1 0 0 1

こうもできるし

例 1 ふたたび

1 1 0 1 1 1 0 0

OFF

1 0 0 1 1 1 0 0

ON

1 0 0 1 1 1 1 0

TOGGLE

0 1 1 0 1 1 1 0

TOGGLE

0 1 1 0 1 0 0 1

こうしてもいい

操作の順番

OFF, ON を先にやるとしてよい

すべての OFF, ON **をすべての** TOGGLE **より先に行う最適解が存在する**

直感的な理由：OFF, ON **は操作前の列の情報を消してしまう感じ，**TOGGLE **は保つ感じだから**

操作の順番

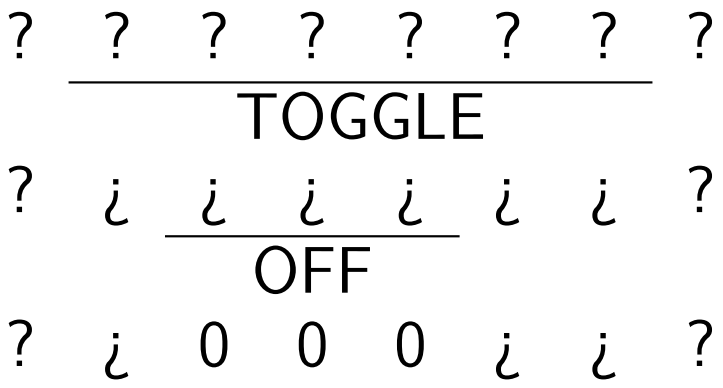
OFF, ON を先にやるとしてよい

すべての OFF, ON **をすべての** TOGGLE **より先に行う最適解が存在する**

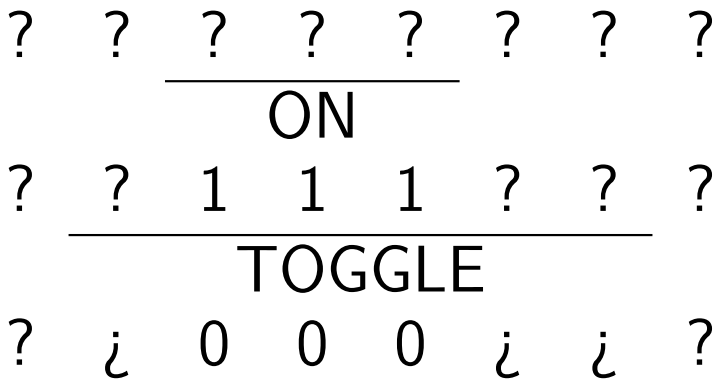
操作の結果を変えず回数を増やさずに，OFF, ON **を先に持ってこれることを示す**

操作の順番

TOGGLE **した部分の一部に** OFF [ON] **をする場合**

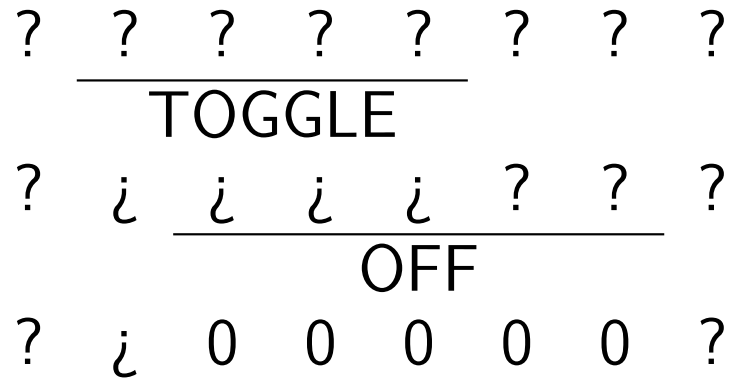


代わりに先に ON [OFF] **をする**

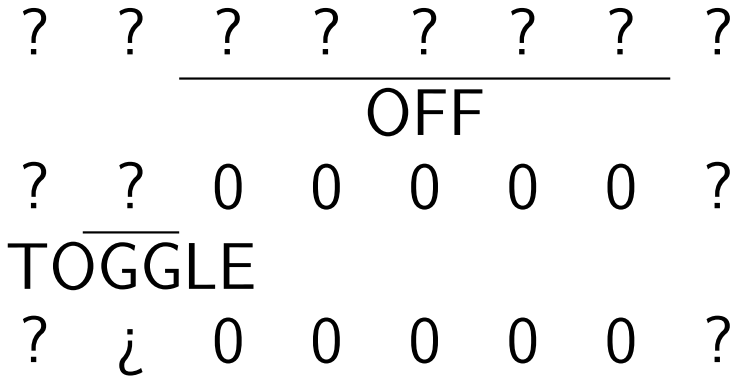


操作の順番

そうでない場合

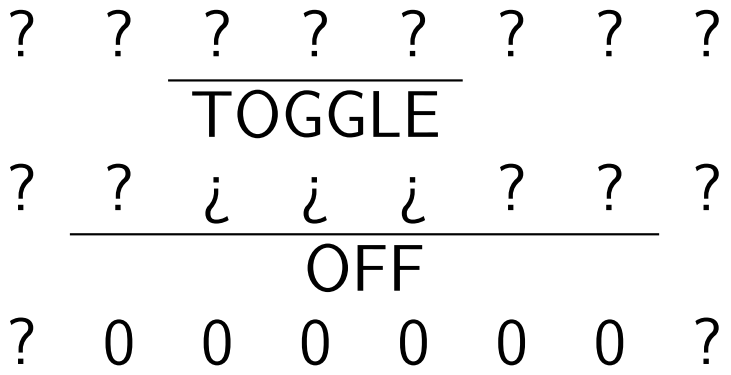


(TOGGLE する区間を縮めて) 入れ替える

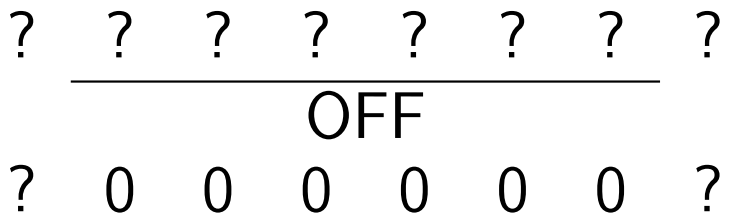


操作の順番

そうでない場合



(TOGGLE する区間を縮めて) 入れ替えるか消す



方針

操作の様子を以下の方針で考える

1. 操作 OFF, ON たちが終了した時点で，各ランプが「OFF にされた」「ON にされた」「最初のまま」のいずれになるか決める
2. 初期状態から 1. で決めた状態にするために必要な操作 OFF, ON の最小回数を求める
3. 1. で決めた状態から目標状態にするために必要な操作 TOGGLE の最小回数を求める

方針

操作の様子を以下の方針で考える

1. 操作 OFF, ON たちが終了した時点で，各ランプが「OFF にされた」「ON にされた」「最初のまま」のいずれになるか決める
2. 初期状態から 1. で決めた状態にするために必要な操作 OFF, ON の最小回数を求める
3. 1. で決めた状態から目標状態にするために必要な操作 TOGGLE の最小回数を求める

OFF, ON の回数

「OFF にされた」「ON にされた」「最初のまま」を 0, 1, ? で表すとする

?0111001111?00110111000?? みたいにするには？

? 0 1 1 1 0 0 1 1 1 1 ? 0 0 1 1 0 1 1 1 0 0 0 ? ?

OFF, ON の回数

? で区切って，0 の連続や 1 の連続はまとめて，

- 0：1 回
- 01：2 回
- 010：2 回
- 0101：3 回
- 01010：3 回
- 010101：4 回

OFF, ON の回数

長さ $k > 0$ **の** 0101... [1010...] **は最小** $\left\lceil \frac{k+1}{2} \right\rceil$ **回**

- 0101... **なら全体を** OFF **にしてから** ON **を** $\left\lfloor \frac{k}{2} \right\rfloor$ **回すればよい**
- **左右には** ? **が書いてあるとして，「隣り合った文字が異なる」ような箇所に注目**
  - 1 **回の操作で高々** 2 **箇所しか増えない**
  - $k+1$ **箇所作らないといけない**

方針

操作の様子を以下の方針で考える

1. 操作 OFF, ON たちが終了した時点で，各ランプが「OFF にされた」「ON にされた」「最初のまま」のいずれになるか決める
2. 初期状態から 1. で決めた状態にするために必要な操作 OFF, ON の最小回数を求める
3. 1. で決めた状態から目標状態にするために必要な操作 TOGGLE の最小回数を求める

TOGGLE の回数

1001010110 から 0100101111 に TOGGLE だけで変えるには？

0000000000 から 1101111001 に TOGGLE だけで変えるには？ と同じ (XOR をとった)
小課題 3 の解法で説明した通り

方針

操作の様子を以下の方針で考える

1. **操作 OFF, ON たちが終了した時点で，各ランプが「OFF にされた」「ON にされた」「最初のまま」のいずれになるか決める**
2. **初期状態から 1. で決めた状態にするために必要な操作 OFF, ON の最小回数を求める**
3. **1. で決めた状態から目標状態にするために必要な操作 TOGGLE の最小回数を求める**

動的計画法

0, 1, ? **の列をすべて試すと** 3^N **通りでダメ**

OFF, ON **の回数も** TOGGLE **の回数も，列を左から一度見ていくだけで求まる**
OFF, ON **の回数や** TOGGLE **の回数を求めるのに必要なものを状態として，左から** 0, 1, ? **の列を決めていく** DP **ができそう**

動的計画法

状態

- **今何文字目まで見たか**
- **直前は** 0, 1, ? **のどれであるか**
- (0 **の連続や** 1 **の連続をまとめたとき) 今作っている** 0101... [1010...] **の部分の長さ**

遷移

- **次を** 0, 1, ? **のどれにするか**

動的計画法

回数計算

- OFF, ON **の回数：**0101... **を閉じるときに足す**
- TOGGLE **の回数：目標との** XOR **が** 0 **から** 1 **になるときに足す**

$O(N^2)$ **時間・**$O(N^2)$ **メモリ (小課題** 2 **が解ける)**

効率の良い回数計算

長さ 1 or 2, 4, 6, . . . **になったときに** 1 **を足す**

- 0：1 回
- 01：2 回
- 010：2 回
- 0101：3 回
- 01010：3 回
- 010101：4 回

効率の良い回数計算

状態

- **今何文字目まで見たか**
- **直前は** 0, 1, ? **のどれであるか**
- (0 **の連続や** 1 **の連続をまとめたとき) 今作っているのは** 0101... **か** 1010... **か**

$5N$ **状態くらい**
$O(N)$ **時間・**$O(N)$ **メモリ (満点)**

操作の手順をもっと制限

実は……

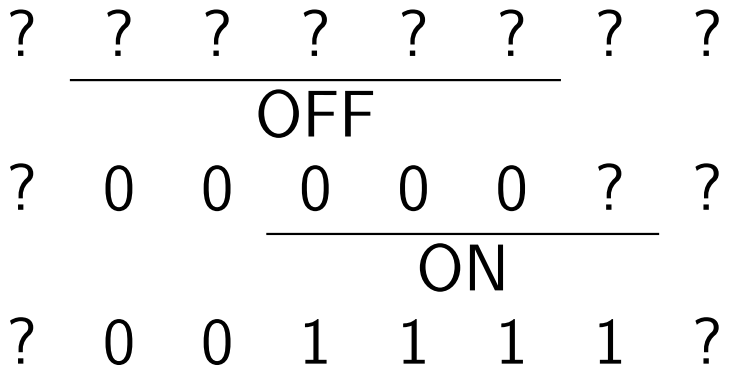
OFF, ON をする区間は重ならなくてよい

以下を満たす最適解が存在する

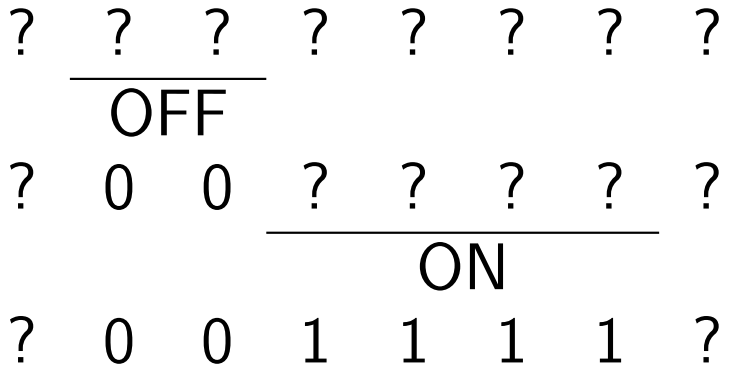
- **すべての** OFF, ON **をすべての** TOGGLE **より先に行う**
- OFF, ON **をする区間は互いに重ならない**
- OFF, ON **をする区間は隣接しない**

操作の手順をもっと制限

OFF や ON をする区間が重なっているのは……

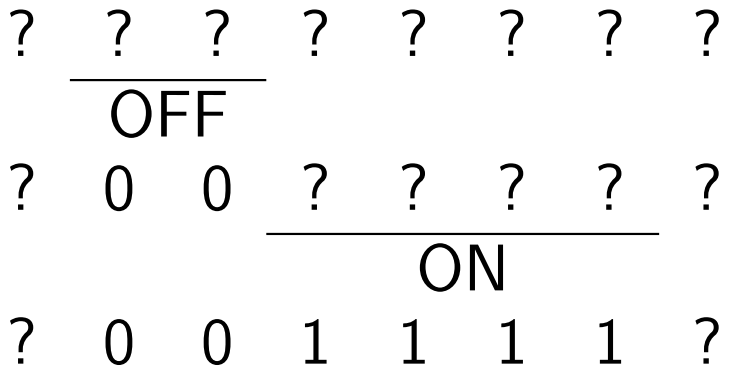


意味がない

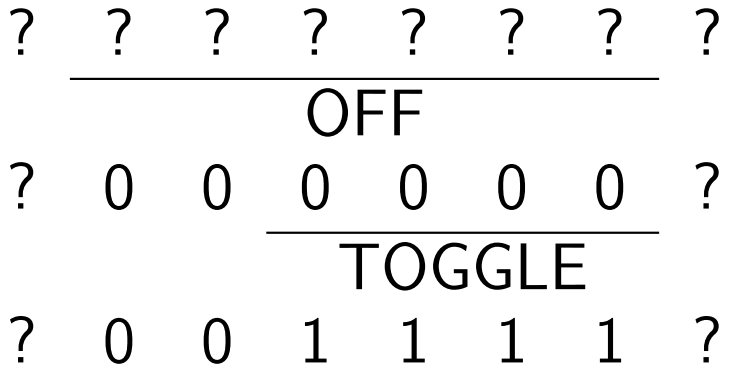


操作の手順をもっと制限

OFF や ON をする区間がくっついているのは……



TOGGLE を使うとくっついている部分を消せる



操作の手順をもっと制限

0, 1, ? **の列としては，**?...?0...0?...?0...0?...?1...1 **のように** 0 **と** 1 **が隣接しないものだけを考えればよい**

DP **の状態を** $3N$ **個くらいにできる**

得点分布

得点分布