EGOI 2024 Editorial - Garden Decorations

Problem author: Massimo Cairo

The problem

There is a sequence of N bits. You need to apply the given permutation to this sequence, but you cannot just read and write random elements of the sequence. Instead, first you go from left to right, and must write a new value for each element after reading its current value. Then, you forget everything you saw (your program is restarted), and you go from right to left, and must write a new value for each element after reading its current value. Then, you forget everything you saw again, and go from left to right, and so on. Your goal is to apply the given permutation correctly in at most W steps, and the lower W is, the better. To get any points, you need $W \leq 60$, and for a full score, you need $W \leq 3$.

Test group 1: N = 2

In this test group there are only two possible permutations. If the permutation is the identity permutation we just output $W = 0$ and terminate your program as there is nothing to do.

Else the permutation is a swap of two elements. We can solve this in 3 runs. In the first run, we keep the first element as is ($b_{1,0} = b_{0,0}$) and set $b_{1,1} = b_{0,0} \oplus b_{0,1}$. In the second run, we go through the elements in reverse order. We keep the second element as is ($b_{2,1} = b_{1,1}$) and set

$$b_{2,0} = b_{1,0} \oplus b_{1,1} = b_{0,0} \oplus b_{0,0} \oplus b_{0,1} = b_{0,1}.$$

In the third run we again process the elements from the front. We keep $b_{3,0} = b_{2,0} = b_{0,1}$ and set $b_{3,1} = b_{2,0} \oplus b_{2,1} = b_{0,1} \oplus b_{0,0} \oplus b_{0,1}$.

Test group 2: $N \leq 15$

In this test group we will see how we can use the swap that we established in Test group 1 to process a full permutation. Note that we can represent any permutation as a sequence of at most $N - 1$ swaps. For every swap, we need 3 moves, and one more move to get back to start with a forward move. This solves the problem in $4 \cdot (N - 1) \leq 56$ moves.

Going back is actually not necessary, and we can instead start the swap in the back pass. This gives $3 \cdot (N - 1) \leq 42$ moves in total. You can find ideas on reducing W even further in the solutions for test group 5.

Test group 3: Reversal

In this test group the given permutation is the reverse permutation, and therefore it consists of many independent swaps.

Note that we do not have to do swaps one by one. Instead, we can apply exactly the same computation for several independent swaps in the same three passes. Since in this test group all swaps are independent, we can solve it this way with $W = 3$.

Test group 4: Shift to the right

In this test group the permutation is a cyclic shift to the right. We can generalize the approach from test group 1 as follows.

In the first run, we set each element to the sum of this element and all previous elements:

- $b_{1,0} = b_{0,0}$
- $b_{1,1} = b_{0,0} \oplus b_{0,1}$
- $b_{1,2} = b_{0,0} \oplus b_{0,1} \oplus b_{0,2}$
- ...
- $b_{1,N-1} = b_{0,0} \oplus b_{0,1} \oplus \cdots \oplus b_{0,N-1}$

,

In the second backward run, we set each element to the sum of this element and the next element. Most of the original terms cancel out, and we get:

- $b_{2,N-1} = b_{1,N-1} = b_{0,0} \oplus b_{0,1} \oplus \cdots \oplus b_{0,N-1}$
- $b_{2,N-2} = b_{1,N-1} \oplus b_{1,N-2} = b_{0,N-1}$
- $b_{2,N-3} = b_{1,N-2} \oplus b_{1,N-3} = b_{0,N-2}$
- ...
- $b_{2,1} = b_{0,2}$
- $b_{2,0} = b_{0,1}$

Almost all elements are already correct, we just need to add all other elements to the last element in the third run:

- $b_{3,0} = b_{2,0} = b_{0,1}$
- $b_{3,1} = b_{2,1} = b_{0,2}$
- ...
- $b_{3,N-2} = b_{0,N-1}$
- $b_{3,N-1} = b_{2,0} \oplus b_{2,1} \oplus \cdots \oplus b_{2,N-1} = b_{0,0}$

,

Test group 5: Shift to the left

In this test group the permutation is a cyclic shift to the left. We can generalize the approach from test group 1 in a slightly different manner.

In the first run, we keep the first element unchanged, and set each subsequent element to the sum of this element and previous element:

- $b_{1,0} = b_{0,0}$
- $b_{1,1} = b_{0,0} \oplus b_{0,1}$
- $b_{1,2} = b_{0,1} \oplus b_{0,2}$
- ...
- $b_{1,N-1} = b_{0,N-2} \oplus b_{0,N-1}$

,

In the second backward run, we set each element to the sum of this element and all subsequent elements. Most of the original terms cancel out, and we get:

- $b_{2,N-1} = b_{1,N-1} = b_{0,N-2} \oplus b_{0,N-1}$
- $b_{2,N-2} = b_{1,N-1} \oplus b_{1,N-2} = b_{0,N-2} \oplus b_{0,N-1} \oplus b_{0,N-3} \oplus b_{0,N-2} = b_{0,N-1} \oplus b_{0,N-3}$
- $b_{2,N-3} = b_{1,N-1} \oplus b_{1,N-2} \oplus b_{1,N-3} = b_{0,N-1} \oplus b_{0,N-4}$
- ...
- $b_{2,1} = b_{0,N-1} \oplus b_{0,0}$
- $b_{2,0} = b_{0,N-1}$

And finally, in the third forward run, we just add the first element to all other elements to cancel it out:

- $b_{3,0} = b_{2,0} = b_{0,N-1}$
- $b_{3,1} = b_{2,1} \oplus b_{2,0} = b_{0,0}$
- $b_{3,2} = b_{2,2} \oplus b_{2,0} = b_{0,1}$
- ...
- $b_{3,N-1} = b_{0,N-2}$

$W = 6$: 90 points

In test group 3, we have managed to solve the reverse permutation with $W = 3$ thanks to the fact that it consists of independent swaps. There are many more such permutations which are also called *permutations of order 2*.

Moreover, it turns out that every permutation can be represented as a product of two permutations of order 2! Which means that we can just apply those two permutations in order, each using 3 passes, to obtain a solution with $W = 6$.

To prove that this is the case, it suffices to show how to represent a cycle in this manner, since this construction can then be applied to all cycles of a permutation independently.

First, consider a cycle of even length. We start with an identity permutation $(0, 1, \ldots, 2k-1)$. Let us swap pairs of adjacent elements first, we get: $(1, 0, 3, 2, \ldots, 2k-1, 2k-2)$. Now let us swap pairs of adjacent elements, but starting with position 1, we get: $(1, 3, 0, 5, 2, \ldots, 2k-1, 2k-4, 2k-2)$. This permutation is one long cycle: it goes from 0 to 1, then over all odd numbers to $2k-1$, then to $2k-2$, and then back over all even numbers to 0.

Since we obtained some long cycle by applying two permutations of order 2 sequentially, we can then obtain the cycle we need just by appropriately renumbering the elements. One can also check that for a cycle of odd length exactly the same construction works correctly.

$W = 5$: 95 points

In the previous solution, we need to apply two permutations of order 2, and we know how to apply each using 3 passes. It turns out that we can apply the last pass of the first permutation simultaneously with the first pass of the second permutation. Indeed, if we build the 3 passes for each permutation as forward-backward-forward, then the last pass of the first permutation and the first pass of the second permutation are both forward passes. So we can apply both of them at the same time by substituting the expressions needed for the first permutation into the expressions needed for the second permutation.

This improves this solution to $W = 5$.

$W = 3$: Full solution

The full solution relies on the definition of a few parameters. For each element $x \in [n]$ we write $S(x)$ and $T(x)$ for the source and the target in the permutation respectively. We denote by $T^i(x)$ the ith successor of x, to make our arguments easier, we include $T^0(x) = x$ in this definition. Formally

$$T^i(x) = \underbrace{T(T(...T}_{i \text{ times}}(x))).$$

We define $S^i(x)$ accordingly.

We then define $Y(x)$ to be the largest element k such that the path from x to $Y(x)$ never goes to an element smaller than x.

$$Y(x) := T^k(x) \text{ where } k = \max_j \left\{ T^i(x) > x \quad \forall i \in [0..j] \right\}.$$

Note that if the out edge of x goes to the left (i.e. we have $T(x) < x$)), then we set $Y(x) = x$.

Similarly, we define $A(x)$ to be the

$$A(x) := S^k(x) \text{ where } k = \max_j \left\{ S^i(x) > x \quad \forall i \in [0..j] \right\}.$$

Again, if the in edge of x comes from the left (i.e. we have $S(x) < x$)), then we set $A(x) = x$.

Additionally, for every cycle in the permutation the smallest element in the cycle will play a special role. We call this smallest element the zero of the cycle.

Algorithm

First forward Phase For the zeros in the cycles we set $b_{1,x} = b_{0,x}$. For every x that is not the zero in a cycle we set $b_{1,x} = b_{0,x} \oplus b_{0,S(A(x))}$. Note that by the definition of $A(x)$, we have that $S(A(x))$ is guaranteed to be smaller than x and is thus available when we compute this.

Backward Phase We define two sequences of vertices

- $P(x) = p_0, p_1, , ..., p_\ell$ with $p_0 = T(x)$ and $p_{i+1} = T(Y(p_i))$ while $p_i > x$. If $T(x) = x$ then $P = \emptyset$. In particular that means that for p_ℓ we have $T(Y(p_\ell)) \leq x$.
- $Q(x) = q_0, q_1, , ..., q_\ell$ with $q_0 = A(x)$ and $q_{i+1} = T(Y(q_i))$ while $p_i > x$. If $A(x) = x$ then $Q = \emptyset$. In particular that means that for q_ℓ we have $T(Y(q_\ell)) \leq x$.

Note that both definitions guarantee all of the visited elements to be strictly larger than x.

1. (for all x) We set $b_{temp} = b_{1,x} \oplus b_{2,p_0} \oplus \ldots \oplus b_{2,p_\ell}$,
2. (if x is the zero in a cycle) We set $b_{2,x} = b_{temp}$,
3. (if x is not the zero in a cycle) We set $b_{2,x} = b_{temp} \oplus b_{2,p_0} \oplus \ldots \oplus b_{2,p_\ell}$.

Second forward Phase For the zeros in the cycles we set $b_{3,x} = b_{2,x}$. For every x that is not the zero in a cycle we set $b_{3,x} = b_{2,x} \oplus b_{3,T(Y(x))}$. Note that by the definition of $Y(x)$, we have that $T(Y(x))$ is guaranteed to be smaller than x and is thus available when we compute this.

Correctness

We define two parameters to keep track of the state of our algorithm. $F(x)$ and $B(x)$. These two parameters will denote which two bits we are currently storing in the position x.

In the beginning we set $F(x) = \emptyset$ and $B(x) = x$ for every element. We will do the proof for the different phases separately.

Claim 1. *After the end of the first forward phase we have $F(x) = S(A(x))$ and $B(x) = x$ for all $x \neq 0$.*

By construction of the first phase this is exactly what we do. By definition of $A(x)$ we have that $S(A(x)) < x$ for all $x \neq 1$ so we can always xor it into position x.

Claim 2. *After the end of the backward phase we the following two conditions hold for every x that is not a zero.*

(C1) $B(x) = Y(x)$,

(C2) $F(x) = S(x)$.

Proof. We prove this claim by induction over the elements of the permutation, going from the highest element in a cycle backwards to the lowest that is not a zero. We first show the base case, that is that if x is the largest element in a cycle, then it is already in the desired state.

After the first forward phase we had $B(x) = x$. Note that as the last element does not have any elements to the right, we clearly have that the out edge of this element goes to the left and thus $Y(x) = x$ which proves our statement. By the same argument, we have $A(x) = x$ and thus $F(x) = A(S(x)) = S(x)$.

For the inductive step, fix some x and assume that $B(x') = Y(x')$ and $F(x') = S(x')$ holds for all $x' > x$. We show that this implies that the statement is also true for x in two steps. First, we look at $B(x)$, secondly, we consider $F(x)$.

C1: $B(x) = Y(x)$

We look at the elements we xor into x in the first part of the backwards phase.

If $P = \emptyset$ then $T(x) < x$. In this case we have $Y(x) = x$ and thus $B(x) = x$ with no modification satisfies condition.

Else we modify $B(x)$. At the end of this phase we have

$$\begin{aligned} B(x) &= x \oplus p_0 \oplus p_1 \oplus \ldots \oplus p_\ell \\ &= x \oplus \underbrace{S(A(T(x)))}_{F(p_0))} \oplus \underbrace{Y(p_0)}_{B(p_0)} \oplus \underbrace{S(T(Y(p_0)))}_{F(p_1))} \oplus Y(p_1) \oplus \ldots \oplus \underbrace{S(T(Y(p_0)))}_{F(p_1))} \oplus Y(p_\ell) \end{aligned}$$

One can observe that, by the way we decided to include the elements in our path, we have $F(p_i) = B(p_{i-1})$ and $F(p_0) = x$ as $T(x) > x$ implies $A(T(x)) = T(x)$. This cancels out all but the $B(p_\ell) = Y(p_\ell)$ part of the chain.

We now show that $Y(p_\ell) = Y(x)$. By definition of the path, p_ℓ is a successor of x such that all elements on the path between x and p_ℓ are larger than x. Note that as p_ℓ is the last vertex on the path we know that $T(Y(p_\ell)) \leq x$. This gives that $Y(p_\ell)$ is the last element that can be reached by following the permutation without crossing x, which by the definition of $Y(x)$ implies that $Y(p_\ell) = Y(x)$ as desired.

C2: $F(x) = S(x)$

The proof for $F(x)$ is very similar to the proof for $B(x)$. We look at the elements we xor into x in the second part of the backwards phase. At the end of this phase we have

$$\begin{aligned} F(x) &= S(A(x)) \oplus q_0 \oplus q_1 \oplus \ldots \oplus q_\ell \\ &= S(A(x)) \oplus \underbrace{S(A(x))}_{F(q_0))} \oplus \underbrace{Y(q_0)}_{B(q_0)} \oplus \underbrace{S(T(Y(q_0)))}_{F(q_1))} \oplus Y(q_1) \oplus \ldots \oplus \underbrace{S(T(Y(q_0)))}_{F(q_1))} \oplus Y(q_\ell) \\ &= S(A(x)) \oplus F(p_0) \oplus B(p_\ell) \end{aligned}$$

First we will show $F(p_0) = S(A(A(x)) = S(A(x))$. This follows from the fact that $A((A(x)) = A(x)$ as the source of the element needs to be larger than $A(x)$. We now show that $Y(q_\ell) = S(x)$. By definition of the path, q_ℓ is a successor of $A(x)$ such that all elements on the path between $A(x)$ and q_ℓ lie on the path between $A(X)$ and x, which is larger than x by definition of $A(x)$. Note that as q_ℓ is the last vertex on the path we know that $T(Y(q_\ell)) \leq x$. As q_ℓ lies on the path between $A(x)$ and x, which lies to the right of x, the only way for an element to become smaller than x is to pass through x. This implies $T(Y(q_\ell))) = x$ and we are done as this immediately gives $Y(q_\ell) = S(x)$.

□

Claim 3. *After the end of the backwards phase we have $F(x) = S(x)$ and $B(x) = \emptyset$ for all zeros in the permutation.*

Proof. Note that for the zeros the backwards phase is slightly different. Again, we get

$$\begin{aligned} B(x) &= x \oplus p_0 \oplus p_1 \oplus \ldots \oplus p_\ell \\ &= 0 \oplus \underbrace{S(T(0))}_{F(p_0))} \oplus \underbrace{Y(p_0)}_{B(p_0)} \oplus \underbrace{S(T(Y(p_0)))}_{F(p_1))} \oplus Y(p_1) \oplus \ldots \oplus \underbrace{S(T(Y(p_0)))}_{F(p_1))} \oplus Y(p_\ell) \\ &= Y(p_\ell) \end{aligned}$$

So all we need to show is $Y(p_\ell) = S(x)$.

By definition, $Y(p_\ell)$ has an out edge going to the left that ends in a position $i \leq x$. This means that $Y(p_\ell)$ is the last vertex in the path of successors of x. In particular, this implies $T(Y(p_\ell)) = x$ as x is the minimum element of the permutation appearing in this cycle and thus $Y(p_\ell) = S(x)$ as desired. □

Claim 4. *After the second forward phase we have $F(x) = S(x)$ and $B(x) = \emptyset$ for all x.*

Proof. We again prove this statement by induction over the elements, this time starting from the front. For the zeros we already have the desired state from Claim 3 so there is nothing to prove.

For x that are not zeros in a cycle, assume that we have processed the elements up to x until now. For all $x' < x$ we get by the induction hypothesis $F(x') = S(x')$ and $B(x') = \emptyset$. That means that when we xor $T(Y(x))$ into x it has the value $S(T(Y(x))) = Y(x)$. This cancels out with $B(x)$ and we get $F(x) = S(x)$ and $B(x) = \emptyset$ for all x. □