Paper without a name

A. Auvolat A. Fromherz N. Jeannerod

June 01, 2014

Abstract

In this contribution, we investigate the problem of "SwitchBoxes". The goal is, given n wires, to generate all the permutations of these wires by using "boxes", that swap two wires. We tried to minimize the number of boxes that were necessary. We introduce a conjecture that puts our problem in relation with binary insertion sort. We successfully used a heavily optimized algorithm to prove this conjecture for small values of n.

1 Introduction

The SwitchBoxes problem is a combinatory problem. Given n wires, (coming from the top), we try to generate all the permutations of these wires by using boxes. A box takes two wires, and is allowed to swap these wires. We can concatenate these boxes to get a configuration of boxes. The configuration is said valid if, by swapping or not the wires on every box, we can obtain any permutation of the n wires.

Mathematically, we consider that the wires are numbers from 1 to n. A box is a permutation $\tau(\epsilon) = (i, j)^\epsilon$, where ϵ is equal to 0 or 1. A configuration of boxes C is the concatenation of k boxes $\tau_1, \ldots, \tau_k$, that means, $C(\epsilon_1, \ldots, \epsilon_k) = \tau_1^{\epsilon_1} \circ \ldots \circ \tau_k^{\epsilon_k}$. The configuration C is said valid if $C(\{0, 1\}^k) = \mathfrak{S}_n$.

Given this problem, we tried to determine the best lower bound of the number of boxes k.

We can easily determine trivial lower and upper bounds of the number of boxes :
Given the fact that C is an application, we need to have $Card(\{0; 1\}^k) \geq Card(\mathfrak{S}_n)$. We immediateley obtain that k has to be greater than $\log_2(n!)$.
We also have an upper bound : The traditionnal bubblesort is a valid configuration of the SwitchBoxes problem. We obtain that the optimal k is smaller than $\frac{(n-1)(n-2)}{2}$.

In this contribution, we first develop how the optimal k seems to be linked to the maximal number of comparisons for sorting n elements by binary insertion. We explain why we guess that those numbers are equal.

In a second step, we show how we proved this conjecture for n from 2 to 6, and how we demonstrated that the number of comparisons is an upper bound of the optimal number of boxes for n from 7 to 13 by optimizing the checking algorithm and by constructing a precise configuration.

In addition, we give some ideas we tried to develop in order to prove mathematically this conjecture.

In the rest of this paper, opt_n will be the optimal number of boxes for n wires.

2 Our conjecture

To understand the problem, we first tried manually to get opt_n for very small values of n.
For $n = 2$, the answer is quite simple : One box is enough and necessary to swap the two wires.

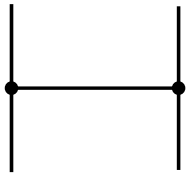

Figure 1: Trivial solution for 2 wires

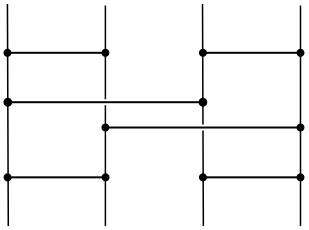

Figure 2: Simple non-optimal solution for 4 wires

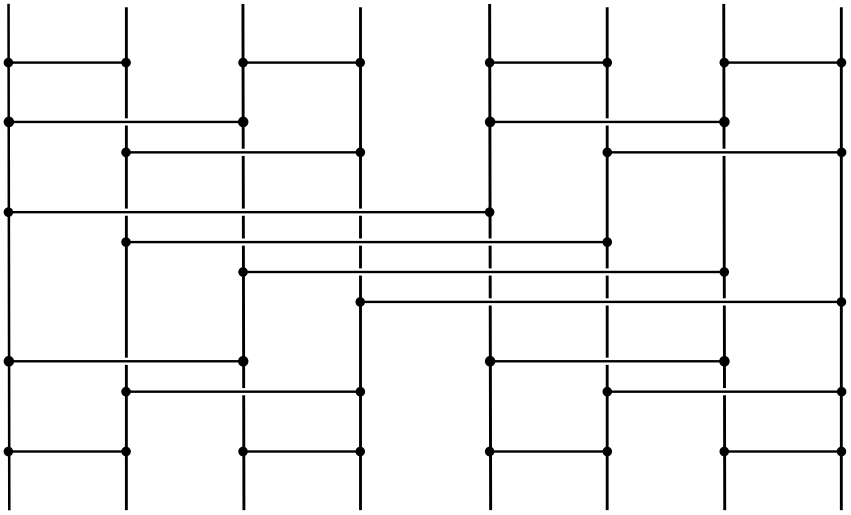

Figure 3: Simple non-optimal solution for 8 wires

For $n = 3$, two boxes aren't enough. Using a permutation, we can consider that the first box swap the wires 1 and 2, and that the second box swap the wires 2 and 3. Then the permutation $(1, 3)$ can't be generated with this configuration. Hence, we have $opt_3 = 3$.

On these representations, a box is a horizontal line that can swap the wires the points are on.

For greater values of n, we implemented a naive and brutal algorithm that for k given, creates all the possible configurations of k boxes, and checks if one of these configurations is valid. We first try it with $opt_{n-1} + 1$. Then, we increment k and try again until we get a valid configuration. With this algorithm, we easily get opt_n.
However, this only works for small values of n. For n greater than 6, we lack memory and time to achieve the computation.

Using this algorithm, we got the results in the table of figure 4.

According to the fact that this sequence is exactly the beginning of the sequence A001855 of

Number of wires n	opt_n
2	1
3	3
4	5
5	8
6	11

Figure 4: Minimum box count for small number of wires

the Encyclopedia of Integer Sequences[1], corresponding to the maximal number of comparisons for sorting n elements by binary insertion, we guessed that these sequences were equal.

Conjecture. *opt_n is equal to the maximal number of comparisons for sorting n elements by binary insertion.*

In order to prove this conjecture for greater values, we tried to optimize our checking algorithm. That will be described in the next part.

2.1 First examples

$n = 2 : S_2(e_1) = (1,2)^{e_1}$
$n = 3 : S_3(e_1, e_2, e_3) = (1,2)^{e_1} \circ (2,3)^{e_2} \circ (1,2)^{e_3}$
$n = 4$: fill in later

2.2 Notation

We abbreviate the previous notation in the following way :

$$S_2 = (1,2)$$

$$S_3 = (1,2)(2,3)(1,2)$$

$$S_4 = (1,2)(3,4)(1,3)(2,4)(1,2)$$

3 Hypothesis box system

We have found a box system that seems to generate all permutations for all n. We have managed to prove that this box system is valid for small values of n ($n \leq 13$), and have found that the box systems were still valid when some boxes were removed.

3.1 The basis box configuration

We first define the q-block at position i by :

$$B(q,i) = (q, q+2^i)(q+1, q+1+2^i)\cdots(q+2^i-1)(q+2*2^i-1)$$

For example:

$$B(0,i) = (i, i+1)$$

$$B(1,i) = (i, i+2)(i+1, i+3)$$

$$B(2,i) = (i, i+4)(i+1, i+5)(i+2, i+6)(i+3, i+7)$$

Then, for $n = 2^p$ we define for $q < p$ the (p,q)-line by :

$$L(p,q) = B(q,0)B(q,2^{q+1})\cdots B(q,2^p - 2^{q+1})$$

For example:

$$L(3,0) = (0,1)(2,3)(4,5)(6,7)$$

$$L(3,1) = (0,2)(1,3)(4,6)(5,7)$$

[1]The On-Line Encyclopedia of Integer Sequences, Sorting numbers: maximal number of comparisons for sorting n elements by binary insertion, `http://oeis.org/A001855`

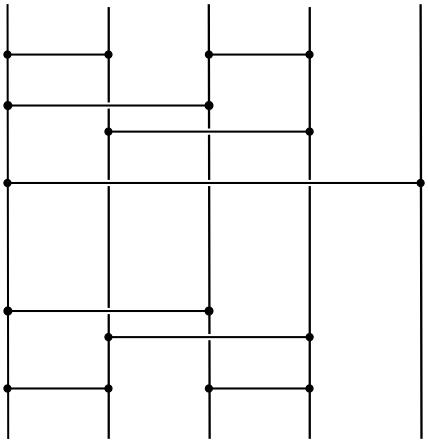

Figure 5: Solution for 5 wires

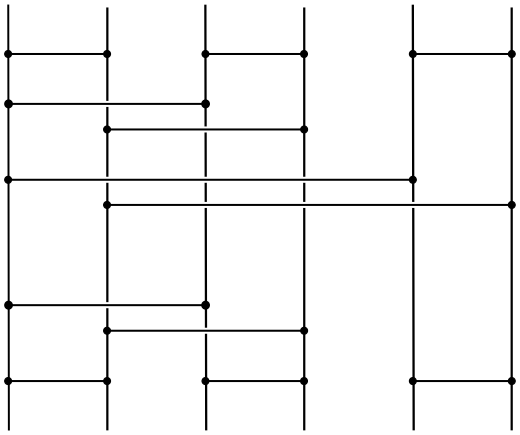

Figure 6: Solution for 6 wires

$$L(3,2) = (0,4)(1,5)(2,6)(3,7)$$

We now conjecture that the following configuration generates all permutations for $n = 2^p$:

$$S^0_{2^p} = L(p,0)L(p,1)L(p,2)\cdots L(p,p-1)L(p,p-2)\cdots L(p,1)L(p,0)$$

For example :

$$\begin{aligned} S^0_8 = \quad & L(3,0)L(3,1)L(3,2)L(3,1)L(3,0) \\ = \quad & (0,1)(2,3)(4,5)(6,7) \\ & (0,2)(1,3)(4,6)(5,7) \\ & (0,4)(1,5)(2,6)(3,7) \\ & (0,2)(1,3)(4,6)(5,7) \\ & (0,1)(2,3)(4,5)(6,7) \end{aligned}$$

3.2 Truncation

For n which is not a power of two, we construct S^0_n by taking $S^0_{2^{log_2(n)}}$ and keeping only boxes (i,j) having $i,j < n$.

For example, S^0_5 can be generated from S^0_8 and by removing quite a lot of the boxes. We obtain the following solution :

$$S^0_5 = (0,1)(2,3)(0,2)(1,3)(0,4)(0,2)(1,3)(0,1)(2,3)$$

3.3 Testing the box system

3.4 Useless box elimination