The Duplicate Detection Problem

S. Kamal Abdali
Division of Computer-Communication Research
National Science Foundation
Arlington, VA 22230

December 19, 2003

1 Introduction

Here is a solution to a problem in Berlekamp and Buhler's *Puzzles Column* in MSRI's *Emissary* [1]. They state the problem as follows:

> 6. A read-only array of length n contains entries $a[0], a[1], \cdots, a[n-1]$ all taken from the set $\{1, ..., n-1\}$. By the pigeonhole principle there is a duplicated element (at least one).
>
> (a) Assume that there is exactly one duplicated entry. Find an algorithm (e.g., a program in your favorite programming language) that takes this array as input and then prints out the duplicated value. The program cannot modify the input array. It should run in linear time, and it should use a constant amount of storage words (each capable of storing integers up to n).
>
> (b) Write a program, subject to the same constraints, that prints a duplicated element, without making any assumptions on the number of duplicated entries. This program, too, is forbidden from modifying the input, should run in linear time, and should use constant extra space.
>
> Comment: Part (b) seems to be a lot harder than part (a), and it apparently requires entirely different techniques. We learned of problem (b) from Eric Roberts, who heard it at a SIGCSE meeting in Greece this summer. (Readers aware of earlier sources can share that information by sending email to berlek@math.berkeley.edu or jpb@reed.edu.)
>
> *Solutions will be posted in the near future at the Emissary web page, http://www.msri.org/publications/emissary/.*

Part (a) is easy. The given list consists of all integers in the range $1..n-1$ AND one more number k $(1 \leq k \leq n-1)$. Just compare the sum of the given numbers with the sum of all integers $\{1, 2, \cdots, n-1\}$ (i.e., with $(n-1)n/2$). The difference is the duplicated element k.

So let us move to Part (b).

2 Problem Statement

I changed the indexing to start with one instead of zero, and extended the range so the largest number is n instead of $n-1$. So the problem needs to be restated slightly: We are given a positive integer n, and an array of $n+1$ elements filled with integer values in the range $1..n$. Clearly some value must be duplicated at least once in the array. But there is no restriction on how many duplicates can occur in the array. The problem is to find any one of the duplicated values.

3 Observations

Let the given array be $a[1], a[2], \cdots, a[n{+}1]$ in which the elements $a[i]$ have values in the range $\{1, 2, \cdots, n\}$.

We define an *indirection list* to be a sequences of the form $i, a[i], a[a[i]], a[a[a[i]]], \cdots$, for some $1 \leq i \leq n+1$.

To write such sequences more concisely, we define $a^0[i] = i, a^1[i] = a[i]$, and $a^{j+1}[i] = a[a^j[i]]$ for $j > 1$. That is, $a^j[i]$ means $a[a[\cdots[a[a[i]]\cdots]]$ with j levels of subscripting. Now we can express the indirection list above as $i, a[i], a^2[i], a^3[i], \cdots$.

Since the values occurring in an indirection list range in $1..n$ only, some values repeat themselves. Moreover, from the way indirection lists are constructed, the repetitions occur in cycles of fixed lengths, not exceeding n.

If in the indirection list $i, a[i], a^2[i], a^3[i], \cdots$, the first element i occurs again, then we define the *period* of that indirection list to be the number of elements between two successive occurrences of i. If i does not occur again in the indirection list, then we say that its period is infinite. Elements do repeat in indirection lists of infinite period also, but no cycle starts at the first position.

If some terms are deleted from the beginning of an indirection list, then the remaining subsequence is also, of course, an indirection list. Starting with an indirection list of infinite period, if we keep deleting terms from the beginning, then at some point the first element is bound to repeat, making the truncated indirection list one of finite period.

As an example, suppose that $n = 9$, and the given array is:

i	1	2	3	4	5	6	7	8	9	10
$a[i]$	4	6	6	7	8	9	1	2	5	3

The indirection lists for this array are shown in the table below, which contain many examples of periodic values and of indirection lists with finite and inifinite periods.

i	indirection list starting from i	period
1	1, 4, 7, 1, $\cdots$	3
2	2, 6, 9, 5, 8, 2, $\cdots$	5
3	3, 6, 9, 5, 8, 2, 6, 9, 5, 8, 2, $\cdots$	infinite
4	4, 7, 1, 4, $\cdots$	3
5	5, 8, 2, 6, 9, 5, $\cdots$	5
6	6, 9, 5, 8, 2, 6, $\cdots$	5
7	7, 1, 4, 7, $\cdots$	3
8	8, 2, 6, 9, 5, 8, $\cdots$	5
9	9, 5, 8, 2, 6, 9, $\cdots$	5
10	10, 3, 6, 9, 5, 8, 2, 6, 9, 5, 8, $\cdots$	infinite

Indirection lists of infinite period arise because of duplicated values. Some duplicated value d succeeds two different list elements k and l corresponding to the two different positions that this value occupies in the array a (i.e. $a[k] = a[l] = d$). Moreover, the value d follows k first then follows l repeatedly, and k itself never shows up again in the iteration list.

For example, the indirection list starting with 3 has an infinite period, with the duplicated value 6 starting a cycle when it follows 3 and then follows 2 repeatedly. While this indirection list 3, 6, 9, 5, 8, 2, 6, 9, 5, $\cdots$ is of infinite period, the truncated indirection list 6, 9, 5, 8, 2, 6, 9, 5, $\cdots$ is of period 5. The first element, 6, of this truncated indirect list is a duplicated value. Indirection lists of infinite period thus serve as the key to finding duplicated values, and lead us to the following two-phase solution to our problem:

1. Construct an indirection list of infinite period.
2. Keep truncating that indirection list until it becomes one of finite period. The first element in that list is one of the duplicate values.

4 Solution

First, let us look at an inefficient way to solve the problem:

Phase 1: For $i = 1, 2, \cdots$ we generate the indirection list starting with i and find its period until a list of infinite period is found. We do this by testing, for each i, the successive values of $a[i], a^2[i], a^3[i], \cdots$ for being equal to i. If $a^j[i] = i$ for some $1 < j \leq n + 1$, then j is, of course, the (finite) period of this indirection list. If such a j is found, then we go to the next value of i and try again. If for any i, none of $a^j[i] = i$ for $1 \leq j \leq n + 1$ then i cannot occur again in the sequence any later, and the period of this indirection list is infinite.

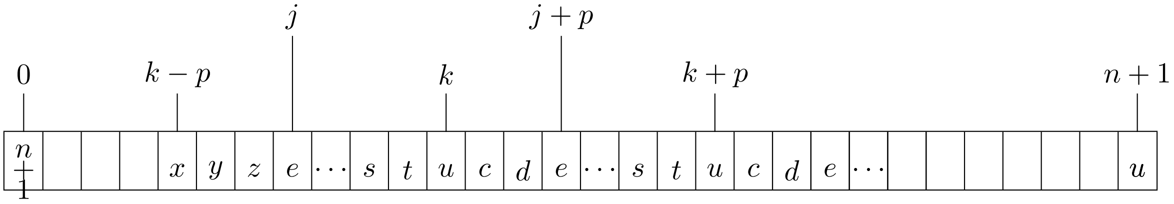
Since the array has at least one duplicated value, a indirection list of infinite period exists, and is going to be found this way.

Phase 2: Once an indirection list of infinite period has been found, say, the one starting at i, then we generate successive values $a[i], a^2[i], \cdots$ representing the starting element of successive truncated indirection lists. For each such starting element, say k, we test at most $n+1$ values $a[k], a^2[k], a^3[k], \cdots$ to see if any of these equals k. For some k this would be so, and that k would be a duplicated value.

The above method uses $O(n^2)$ steps for each of the two phases, so we need to look for a better method.

Phase 1: Phase 1 really doesn't require any time because an indirection list of infinite period is available without doing any search. The indirection list starting with $n+1$ is such a list since $n+1$ cannot occur again as an element. This indirect list consists of $n+1, a[n+1], a^2[n+1], a^3[n+1], \cdots$.

Phase 2: The purpose of Phase 2 is to find the first element (after the starting element $n+1$) that repeats itself in the indirection list. Let us look at the following diagram in which the cells represent elements of the indirection list starting with $n+1$ and the vertical lines indicate positions in that list.



Suppose that the element we are looking for is e which first occurs in position j in the indirection list, then reoccurs after every pth position. That is, $e = a^j[n+1] = a^{j+p}[n+1] = \cdots$. Now every element that repeats itself in the indirection list is going to occur in some position between j and $j+p$. An instance of such an can be found by going sufficiently far in the indirection list. In fact, $a^{n+1}[n+1]$, call it u, is such an element. Hence we use the following steps to carry out Phase 2:

1. Find $u = a^{n+1}[n+1]$.
2. Find the position k of the first occurrence of u in the list, i.e., find the first k among successive integers starting with $k = 1$ such that $a^k[n+1] = u$.
3. Find the period of that element, i.e., by looking successively at $a^{k+1}[n+1], a^{k+2}[n+1], \cdots$, find p such that $a^{k+p}[n+1] = u$.
4. Find the first j among successive integers starting with $j = k-p+1$ such that $a^j[n+1] = a^{j+p}[n+1]$. $e = a^j[n+1]$ is a duplicated value in the array a.

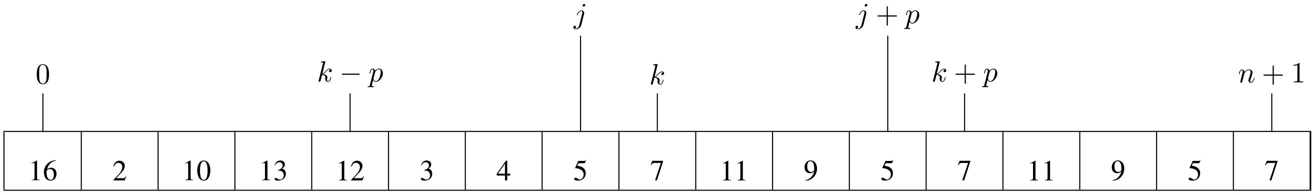
Step 1, Steps 2 and 3 together, and Step 4 require at most $O(n)$ operations each.

5 Example

As an example, suppose that $n = 15$, and the given array is:

i	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
$a[i]$	14	10	4	5	7	5	11	1	5	13	9	3	12	15	8	2

The indirection list starting with $n + 1 = 16$ is diagrammed below together with various position markers to determine the duplicated value.



The element in position $n + 1 = 16$, i.e. $a^{16}[16]$ is 7.

Searching for it from the beginning we find that $a^8[16] = 7$, so $k = 8$.

To determine the period of this element, we check $a^9[16], a^{10}[16], \cdots$, finding that $a^{12}[16] = 7$. So $p = 12 - 8 = 4$.

$k - p = 4$. So now we compare $a^5[16]$ with $a^9[16]$, $a^6[16]$ with $a^{10}[16]$, $\cdots$. It turns out that $a^7[16] = a^{11}[16] = 5$. So 5 is a duplicated value.

References

[1] Elwyn R. Berlekamp and Joe P. Buhler, “Puzzles Column,” *Emissary: Mathematical Sciences Research Institute*, Problem 6, p. 10, Spring/Fall 2003.