

Average-Case Analysis of Some Plurality Algorithms

LAURENT ALONSO

INRIA-Lorraine

and

EDWARD M. REINGOLD

Illinois Institute of Technology

Given a set of n elements, each of which is colored one of c colors, we must determine an element of the plurality (most frequently occurring) color by pairwise equal/unequal color comparisons of elements. We focus on the expected number of color comparisons when the c^n colorings are equally probable. We analyze an obvious algorithm, showing that its expected performance is $\frac{c^2+c-2}{2c}n - O(c^2)$, with variance $\Theta(c^2n)$. We present and analyze an algorithm for the case $c = 3$ colors whose average complexity on the 3^n equally probable inputs is $\frac{7083}{5425}n + O(\sqrt{n}) = 1.3056 \cdots n + O(\sqrt{n})$, substantially better than the expected complexity $\frac{5}{3}n + O(1) = 1.6666 \cdots n + O(1)$ of the obvious algorithm. We describe a similar algorithm for $c = 4$ colors whose average complexity on the 4^n equally probable inputs is $\frac{761311}{402850}n + O(\log n) = 1.8898 \cdots n + O(\log n)$, substantially better than the expected complexity $\frac{9}{4}n + O(1) = 2.25n + O(1)$ of the obvious algorithm.

Categories and Subject Descriptors: F.2.2 [Nonnumerical Algorithms and Problems]: Sorting and Searching; G.2.1 [Combinatorics]: Combinatorial Algorithms

General Terms: Algorithms, Theory

Additional Key Words and Phrases: Algorithm analysis, majority problem, plurality problem

1. INTRODUCTION

The *plurality problem*, first posed in [Alonso et al. 1993], is to determine an element of the most frequently occurring color in a set of n elements, using only pairwise equal/unequal color comparisons of elements; each element is colored with one of c colors. The first results for $c > 2$ were for three colors: Aigner et al. [2005] proved that $3\lfloor n/2 \rfloor - 2$ color comparisons are necessary and that $\lfloor 5n/3 \rfloor - 2$ suffice; they proved in general that $cn/40$ color comparisons are necessary for c colors, a lower bound improved to $2cn/27 - o(n)$ in [Král' et al. 2008]. Chung et al. [2007] showed that $(c-1-1/c)n-2$ comparisons suffice, and that for nonadaptive strategies with $c \geq 3$, $n^2/6 - 3n/2$ comparisons are necessary to determine plurality.

The case $c = 3$ was also studied in [Dvořák et al. 2007], where $3n/2 - O(\sqrt{n \log n})$ comparisons are proved necessary and $3n/2 + O(1)$ are proved sufficient when randomized algorithms are allowed. For c colors, Král' et al. [2008] proved a lower bound of $\Omega(cn)$ for randomized algorithms. Dvořák et al. [2007] also studied the *partition problem* in which the three color classes of the elements must be completely

Authors' addresses: Laurent Alonso, INRIA-Lorraine and LORIA, Université Henri Poincaré-Nancy I, BP 239, 54506 Vandoeuvre-lès-Nancy, France. Email: Laurent.Alonso@loria.fr
Edward M. Reingold, Department of Computer Science, Illinois Institute of Technology, 10 West 31st Street, Chicago, Illinois 60616-2987 USA. Email: reingold@iit.edu

determined; they proved that $2n - 3$ comparisons are necessary and sufficient in the ordinary case and $(5n - 8)/3 + o(1)$ in the randomized case; for the ordinary case of the partition problem with c colors and $n \geq c$, they proved that $(c - 1)n - \binom{c}{2}$ comparisons are necessary and sufficient; Král' et al. [2008] showed that $(c - 1)(n - c)/4$ are necessary for randomized algorithms. With an unknown number of colors, $\binom{n-1}{2}$ comparisons are necessary and sufficient to determine a color that occurs at least as often as any other color (the “non-strict” plurality problem) [Srivastava and Taylor 2005].

The strongest worst-case results for determining plurality were proved in [Alonso and Reingold 2008b]. They proved that $(c - 1)(n - c)/2$ color comparisons are necessary to determine the plurality color, and that $(0.775c + 5.9)n + O(c^2)$ color comparisons are sufficient.

Our paper [Alonso and Reingold 2008a] establishes the strongest known lower bounds in the average case for $c \geq 3$, and for randomized algorithms for $c \geq 4$. That paper includes a general lower bound of $\frac{c}{3}n - O(\sqrt{n})$ for $c \geq 2$, as well as stronger particular bounds of $\frac{7}{6}n - O(\sqrt{n})$ for $c = 3$, $\frac{54}{35}n - O(\sqrt{n})$ for $c = 4$, $\frac{607}{315}n - O(\sqrt{n})$ for $c = 5$, $\frac{1592}{693}n - O(\sqrt{n})$ for $c = 6$, $\frac{7985}{3003}n - O(\sqrt{n})$ for $c = 7$, and $\frac{19402}{6435}n - O(\sqrt{n})$ for $c = 8$.

The present paper is concerned only with designing algorithms and analyzing their average-case behavior. We begin, in the next section, by describing the most obvious algorithm to determine plurality, proving that the expected number of color comparisons for c colors is $\frac{c^2+c-2}{2c}n - O(c^2)$, with variance $\Theta(c^2n)$, assuming that all c^n possible colorings are equally probable. For $c = 3$ this gives $\frac{5}{3}n + O(1) = 1.6666 \dots n + O(1)$ with variance $\frac{2}{9}n + O(1)$ and for $c = 4$ it gives $\frac{9}{4}n + O(1) = 2.25n + O(1)$ with variance $\frac{11}{16}n + O(1)$, establishing a point of comparison for the more sophisticated algorithms we then develop.

In Section 3 we describe a simple algorithm for $c = 3$ colors and analyze its average behavior on n elements where the 3^n possible colorings are equally probable. Our analysis shows that the expected number of color comparisons is $\frac{29}{21}n + O(\log n)$. Then, in the next section, we give a similar but much more complex algorithm for which the constant factor is improved from $\frac{29}{21} \approx 1.38$ to $\frac{229}{175} \approx 1.309$. Following that, we introduce a new subtlety that yields an algorithm with average complexity $\frac{7083}{5425}n + O(\sqrt{n}) = 1.3056 \dots n + O(\sqrt{n})$ and suggest how to make further improvements. Finally, we describe an algorithm for $c = 4$ colors whose average complexity on the 4^n equally probable inputs is $\frac{5413}{2814}n + O(\log n) = 1.923596 \dots n + O(\log n)$ and show how to improve it to $\frac{761311}{402850}n + O(\log n) = 1.8898 \dots n + O(\log n)$. We conclude with some open problems.

2. THE OBVIOUS ALGORITHM

The Obvious Algorithm to determine plurality is to process the n elements sequentially, comparing each element to the color groups found among the previously processed elements. The first element requires no color comparisons and is established as a color group; subsequent elements are compared to the existing color groups and either merged into one they match or established as a new color group. An element that fails to match $c - 1$ color classes must belong to the c th color class.

THEOREM 2.1. *If all c^n possible colorings are equally probable, the expected num-*

ber of color comparisons for the Obvious Algorithm is

$$\frac{c^2 + c - 2}{2c}n - \frac{c^2}{4} - \frac{3c}{4} + H_c + o(1),$$

with variance

$$V(n) = \frac{(c-1)(c-2)(c^2+3c-6)}{12c^2}n - \frac{c(c+7)(2c-11)}{72} - 2H_c + H_c^{(2)} + o(1),$$

where H_c are the harmonic numbers, $H_c = \sum_{i=1}^c \frac{1}{i} = \ln c + O(1)$ and $H_c^{(2)} = \sum_{i=1}^c \frac{1}{i^2} = O(1)$.

PROOF. The Obvious Algorithm is similar to the distribution into piles by the radix sort [Knuth 1998, Section 5.2.5] and we rely in part on the analysis given there, especially [Knuth 1998, solution to Exercise 5.2.5.-6]. If $a < c$ colors have appeared among the first $k > 0$ elements processed, the average cost to process the $(k+1)$ st element is

$$\frac{1}{c}(1+2+\cdots+a) + \frac{c-a}{c}a = \frac{a(2c+1-a)}{2c} \quad (1)$$

color comparisons: if the $(k+1)$ st element is in the i th color class, $\max(i, a)$ color comparisons are used. If all c colors have appeared among the first k elements, the average cost of processing the $(k+1)$ st element is

$$\frac{1}{c}(1+2+\cdots+(c-1)) + \frac{1}{c}(c-1) = \frac{c^2+c-2}{2c} \quad (2)$$

color comparisons because if the $(k+1)$ st element is in the i th color class, $\max(i, c-1)$ color comparisons are used.

Let p_{ka} be the probability that exactly a different colors are found among the first k elements; from [Knuth 1997b, Equation 3.3.2D-(5)],

$$p_{ka} = \frac{c^a}{c^k} \left\{ \begin{matrix} k \\ a \end{matrix} \right\}, \quad (3)$$

where $c^a = c(c-1)\cdots(c-a+1)$ is the “falling factorial” [Knuth 1997a, Section 1.2.5] and $\left\{ \begin{matrix} k \\ a \end{matrix} \right\}$ is a Stirling number of the second kind [Knuth 1997a, Section 1.2.6]; of course, $\sum_{a=1}^c p_{ka} = 1$. Combining (1), (2), and (3), the expected number of color

comparisons to process the $(k + 1)$ st element is

$$\begin{aligned}
& \sum_{a=1}^{c-1} p_{ka} \frac{a(2c+1-a)}{2c} + p_{kc} \frac{c^2+c-2}{2c} \\
&= \frac{1}{2c} \sum_{a=1}^c p_{ka} a(2c+1-a) - \frac{p_{kc}}{c} \\
&= \frac{1}{2c} \sum_{a=1}^c p_{ka} [c - (c-a)][(c-a) + (c+1)] - \frac{p_{kc}}{c} \\
&= \frac{c^2+c-2}{2c} - \frac{1}{2c} \sum_{a=1}^c (c-a)p_{ka} - \frac{1}{2c} \sum_{a=1}^c (c-a)^2 p_{ka} \\
&\quad + \frac{1-p_{kc}}{c}.
\end{aligned}$$

From [Knuth 1998, solution to Exercise 5.2.5.-6],

$$\sum_{a=1}^c (c-a)p_{ka} = c \left(1 - \frac{1}{c}\right)^k$$

and

$$\sum_{a=1}^c (c-a)^2 p_{ka} = c(c-1) \left(1 - \frac{2}{c}\right)^k + c \left(1 - \frac{1}{c}\right)^k,$$

so the expected cost to process the $(k + 1)$ st element is

$$\frac{c^2+c-2}{2c} - \frac{c \left(1 - \frac{1}{c}\right)^k}{2c} - \frac{c(c-1) \left(1 - \frac{2}{c}\right)^k + c \left(1 - \frac{1}{c}\right)^k}{2c} + \frac{1-p_{kc}}{c}.$$

Simplifying, this is

$$\frac{c^2+c-2}{2c} - \frac{(c-1)}{2} \left(1 - \frac{2}{c}\right)^k - \left(1 - \frac{1}{c}\right)^k + \frac{1-p_{kc}}{c}.$$

Summing this for $k = 1$ to $n - 1$ gives the expected number of color comparisons used in inserting the second, third, \dots , n th elements (the first element is inserted at no cost). Hence the expected number of color comparisons used by the Obvious Algorithm is

$$\frac{c^2+c-2}{2c}n - \frac{c^2}{4} - \frac{3c}{4} + O(1) + \frac{1}{c} \sum_{k=1}^{n-1} (1-p_{kc}). \quad (4)$$

Because the summation in the latter term depends on n , we must bound it; we will prove that term is $O(\log c)$. Note that $1 - p_{kc}$ is the probability that all c colors are *not* found among the first k elements; this is the same as the probability that one color is found, or two colors are found, \dots , or $c - 1$ colors are found. Note also

that $\left\{ \begin{smallmatrix} k \\ a \end{smallmatrix} \right\} = 0$ and $p_{ka} = 0$ for $k < a$. Thus, the summation is

$$\begin{aligned}
\sum_{k=1}^{n-1} (1 - p_{kc}) &= \sum_{k=1}^{n-1} \sum_{a=1}^{c-1} p_{ka} \\
&= \sum_{a=1}^{c-1} \sum_{k=1}^{n-1} p_{ka} \\
&\leq \sum_{a=1}^{c-1} c^a \sum_{k=1}^{\infty} \left\{ \begin{smallmatrix} k \\ a \end{smallmatrix} \right\} c^{-k} \\
&= \sum_{a=1}^{c-1} \frac{c^a}{c^a \left(1 - \frac{1}{c}\right) \left(1 - \frac{2}{c}\right) \cdots \left(1 - \frac{a}{c}\right)}. \tag{5}
\end{aligned}$$

By setting $z = c^{-1}$ in the generating function of the Stirling numbers of the second kind [Knuth 1997a, Equation 1.2.9-(28)], we get

$$\sum_{k=0}^{\infty} \left\{ \begin{smallmatrix} k \\ a \end{smallmatrix} \right\} z^k = \frac{z^a}{(1-z)(1-2z)\cdots(1-az)}.$$

The a th term of (5) is at most $c/(c-a)$ and hence,

$$\begin{aligned}
\sum_{k=1}^{n-1} (1 - p_{kc}) &\leq \sum_{a=1}^{c-1} \frac{c}{c-a} \\
&= cH_{c-1},
\end{aligned}$$

giving a final term in (4) of $O(\log c)$. We defer the analysis of the variance to the appendix; we will see there that the last two terms in (4), $O(1) + \frac{1}{c} \sum_{k=1}^{n-1} (1 - p_{kc})$, are in fact $H_c + o(1)$. \square

COROLLARY 2.2. *If all 3^n possible colorings are equally probable, the expected number of color comparisons for the Obvious Algorithm is $\frac{5}{3}n - O(1)$, with variance $\frac{2}{9}n + O(1)$. \square*

COROLLARY 2.3. *If all 4^n possible colorings are equally probable, the expected number of color comparisons for the Obvious Algorithm is $\frac{9}{4}n - O(1)$, with variance $\frac{11}{16}n + O(1)$. \square*

3. A SIMPLE ALGORITHM

Our Simple Algorithm for $c = 3$ aggregates elements into limited types of triples of subsets of elements, a triple consisting of subsets of each of the three colors. We describe these triples by a shorthand notation for the relative size of the constituent subsets. For example, if two subsets are empty and the third is not empty, we call that type 1. If the lone non-empty subset has m elements, we call that type $m \times 1$ and we represent it schematically as

$$\textcircled{m}$$

We use “type 1” to refer generically to triples of type $m \times 1$ for arbitrary m ; m is called the *multiplier* of the triple. If one of the subsets of a triple is empty and

the other two each contain equal numbers of elements, we call that type 11, read “one-one”; if each of those non-empty subsets contains m elements, we call that $m \times 11$ and we represent it schematically as



The solid line joining the two sets indicates that their colors do not match. We use “type 11” to refer generically to triples of type $m \times 11$ for arbitrary m . We will extend this notation for triples further in subsequent sections—that is, we have not (yet) allowed for all three subsets in a triple to be non-empty; if that were to happen, we could discard equal numbers of elements of each color from the sets without affecting overall plurality. In this section we only need to consider cases in which the non-empty sets of a triple have equal numbers of elements.

In Stage I of the Simple Algorithm we consider the n elements to be n triples of type 1×1 , and repeatedly apply the following two steps as long as possible:

- A. Take two triples of type $m \times 1$ and compare an element of each (see Figure 1). If the colors are equal (probability $1/3$), we obtain a triple of type $2m \times 1$; otherwise (probability $2/3$) we obtain a triple of type $m \times 11$.
- B. Take two triples of type $m \times 11$ and compare an element of each (see Figure 2).
 1. If the colors are equal (probability $1/3$), we compare an element from each of the *other* two nonempty members of the triple; if the colors are equal (probability $1/2$), we obtain a triple of type $2m \times 1$; otherwise (probability $1/2$) we obtain a triple of type $m \times 1$ because we can discard m elements of each of the three colors, leaving just m elements of one of the colors.
 2. If the colors are unequal (probability $2/3$), we compare an element from one of the uncomparing subsets with the previously compared subset from the other triple.
 - i. If they are equal (probability $1/2$), we compare the other two subsets of the triples; with probability $1/2$ they are equal and the two triples of type $m \times 11$ yield a triple of type $2m \times 11$. Otherwise we can discard m of each color to leave a triple of type $m \times 1$.
 - ii. If the colors are unequal (probability $1/2$), we can discard m of each color to leave a triple of type $m \times 1$.

Steps A and B are illustrated as tree diagrams in Figures 1 and 2, respectively. In these and all figures, a solid line joining the two sets indicates that their colors do not match, while a dashed line shows which two sets have their colors compared in a node. The upper child of that node results if the colors match; the lower child results if the colors do not match. In later figures (like Figure 3) we use \emptyset as a leaf when no triple is produced.

Because of the discarding of elements that can occur in Step B (Figure 2), it is not true that a triple of type $m \times 1$ corresponds to m elements. All one can say is that a triple of type $m \times 1$ corresponds to *at least* m elements. Similarly, a triple of type $m \times 11$ corresponds to *at least* $2m$ elements.

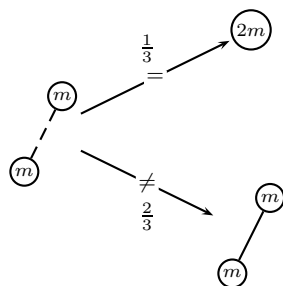


Fig. 1. Simple Algorithm, Step A; characteristic equation $2c_1 = 1 + \frac{1}{3}c_1 + \frac{2}{3}c_{11}$. In this and all other figures, a circle indicates a set of elements and the size of the set is given in the circle. A solid line joining the two sets indicates that their colors do not match; a dashed line shows which two sets have their colors compared.

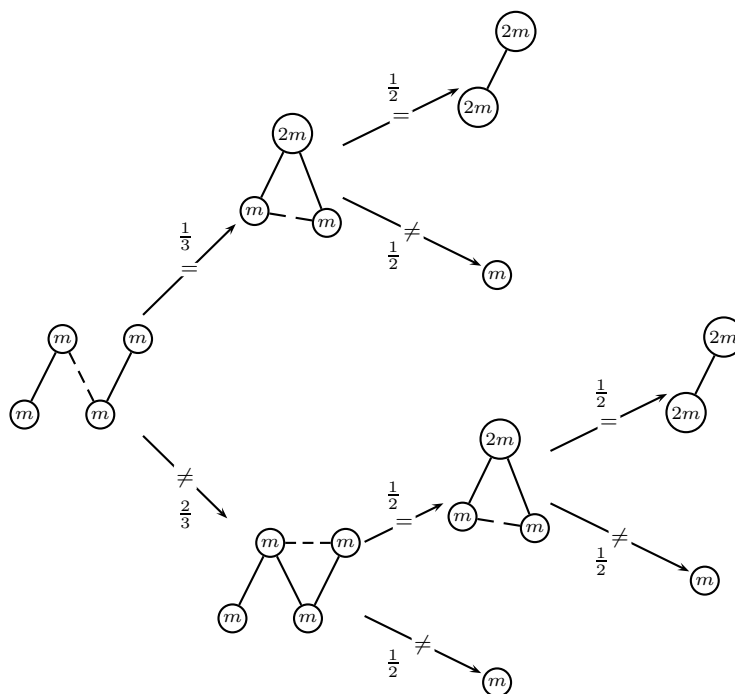


Fig. 2. Simple Algorithm, Step B; characteristic equation $2c_{11} = \frac{7}{3} + \frac{2}{3}c_1 + \frac{1}{3}c_{11}$.

When neither Step A or B can be applied, if no triples remain, there is no plurality. Otherwise, there will be at most one triple of type $2^k \times 1$ and one of type $2^k \times 11$ for all $k \leq \lg n$; that is, there will be at most $O(\log n)$ remaining triples. At this point, Stage I ends and Stage II of the algorithm uses $O(\log n)$ color comparisons (at most two color comparisons per triple of type 1 and at most three per triple of type 11) to count the numbers of elements of each color in the $O(\log n)$ remaining triples, keeping a representative element of each color. If

the counts are equal, there is no plurality; otherwise, the plurality color among the $O(\log n)$ remaining triples, and a representative element of that color, is then evident. Thus we have,

LEMMA 3.1. *The expected number of comparisons in Stage II of the Simple Algorithm is $O(\log n)$. \square*

The correctness of the Simple Algorithm is obvious: only equal numbers of elements of each of the three colors are discarded in Stage I, so the plurality color among the remaining elements, determined in Stage II, must be the plurality color of the original set.

We must analyze the expected number of color comparisons used in Stage I as Steps A and B are applied. Consider the state of affairs at an arbitrary point in the algorithm as Steps A and B are being applied. Let \mathcal{S} be the set of all triples at that point, partitioned into \mathcal{S}_1 and \mathcal{S}_{11} , the triples of types 1 and 11, respectively. Let e_1 and e_{11} be the expected values of $|\mathcal{S}_1|$ and $|\mathcal{S}_{11}|$, respectively, at the end Stage I; at the beginning of Stage I, both of these are $O(\log n)$ by the discussion leading to Lemma 3.1.

LEMMA 3.2. *At any point in Stage I of the Simple Algorithm the expected number of color comparisons in the remainder of Stage I is*

$$\frac{29}{21}(|\mathcal{S}_1| - e_1) + \frac{41}{21}(|\mathcal{S}_{11}| - e_{11}).$$

PROOF. By induction on the maximum number of steps remaining in Stage I. If no steps remain, clearly $|\mathcal{S}_1| = e_1$ and $|\mathcal{S}_{11}| = e_{11}$. Therefore, $\frac{29}{21}(|\mathcal{S}_1| - e_1) + \frac{41}{21}(|\mathcal{S}_{11}| - e_{11}) = 0$.

Suppose k steps remain. Let \mathcal{S}'_1 and \mathcal{S}'_{11} be the sets after the next step and let e'_1 and e'_{11} be the expected values of $|\mathcal{S}'_1|$ and $|\mathcal{S}'_{11}|$, respectively, at the end Stage I. The hypothesis holds by induction because at most $k - 1$ steps remain, so that

$$\frac{29}{21}(|\mathcal{S}'_1| - e'_1) + \frac{41}{21}(|\mathcal{S}'_{11}| - e'_{11})$$

is the expected number of color comparisons of the remainder of Stage I.

If that step is a Step A, the expected values of $|\mathcal{S}'_1|$ and $|\mathcal{S}'_{11}|$ are

$$|\mathcal{S}'_1| = \frac{1}{3}(|\mathcal{S}_1| - 1) + \frac{2}{3}(|\mathcal{S}_1| - 2),$$

and

$$|\mathcal{S}'_{11}| = \frac{1}{3}|\mathcal{S}_{11}| + \frac{2}{3}(|\mathcal{S}_{11}| + 1).$$

Thus the expected number of color comparisons after that step is

$$\begin{aligned} & 1 + \frac{29}{21}(|\mathcal{S}'_1| - e'_1) + \frac{41}{21}(|\mathcal{S}'_{11}| - e'_{11}) & (6) \\ & = 1 + \frac{29}{21} \left(\frac{1}{3}(|\mathcal{S}_1| - 1) + \frac{2}{3}(|\mathcal{S}_1| - 2) - e_1 \right) + \frac{41}{21} \left(\frac{1}{3}|\mathcal{S}_{11}| + \frac{2}{3}(|\mathcal{S}_{11}| + 1) - e_{11} \right) \\ & = \frac{29}{21}(|\mathcal{S}_1| - e_1) + \frac{41}{21}(|\mathcal{S}_{11}| - e_{11}), \end{aligned}$$

because $e'_1 = e_1$ and $e'_{11} = e_{11}$ by definition.

Similarly, if that step is a Step B, the expected values of $|\mathcal{S}'_1|$ and $|\mathcal{S}'_{11}|$ are

$$|\mathcal{S}'_1| = \frac{2}{3}(|\mathcal{S}_1| + 1) + \frac{1}{3}|\mathcal{S}_1|,$$

and

$$|\mathcal{S}'_{11}| = \frac{1}{3}(|\mathcal{S}_{11}| - 1) + \frac{2}{3}(|\mathcal{S}_{11}| - 2).$$

Thus the expected number of color comparisons after that step is

$$\begin{aligned} & \frac{7}{3} + \frac{29}{21}(|\mathcal{S}'_1| - e'_1) + \frac{41}{21}(|\mathcal{S}'_{11}| - e'_{11}) \\ &= \frac{7}{3} + \frac{29}{21} \left(\frac{2}{3}(|\mathcal{S}_1| + 1) + \frac{1}{3}|\mathcal{S}_1| - e_1 \right) + \frac{41}{21} \left(\frac{1}{3}(|\mathcal{S}_{11}| - 1) + \frac{2}{3}(|\mathcal{S}_{11}| - 2) - e_{11} \right) \\ &= \frac{29}{21}(|\mathcal{S}_1| - e_1) + \frac{41}{21}(|\mathcal{S}_{11}| - e_{11}), \end{aligned} \quad (7)$$

and the induction follows. \square

COROLLARY 3.3. *Stage I of the Simple Algorithm uses at most $\frac{29}{21}n$ expected color comparisons.*

PROOF. When Stage I begins, $|\mathcal{S}_1| = n$ and $|\mathcal{S}_{11}| = 0$, so the corollary follows from Lemma 3.2 noting that $e_1 \geq 0$ and $e_{11} \geq 0$. \square

THEOREM 3.4. *If all 3^n possible colorings are equally probable, the expected number of color comparisons for the Simple Algorithm is $\frac{29}{21}n + O(\log n)$.*

PROOF. By Corollary 3.3, Stage I uses at most $\frac{29}{21}n$ expected color comparisons. By Lemma 3.1, Stage II uses at most $O(\log n)$ expected color comparisons. This gives us an upper bound on the expected number of color comparisons of $\frac{29}{21}n + O(\log n)$. To get a lower bound, we use Lemma 3.2 with $|\mathcal{S}_1| = n$ and $|\mathcal{S}_{11}| = 0$, recalling that both e_1 and e_{11} are $O(\log n)$, giving a lower bound on the expected number of comparisons of Stage I of $\frac{29}{21}n - O(\log n)$, which is thus a lower bound on the expected number of comparisons for the Simple Algorithm. \square

The coefficients $\frac{29}{21}$ and $\frac{41}{21}$ in Lemma 3.2 are determined so as to make the induction work: that is, we choose constants c_1 and c_{11} for the upper bound

$$c_1(|\mathcal{S}_1| - e_1) + c_{11}(|\mathcal{S}_{11}| - e_{11})$$

so that for Step A from (6)

$$1 + c_1 \left(\frac{1}{3}(-1) + \frac{2}{3}(-2) \right) + c_{11} \frac{2}{3} = 0,$$

or,

$$2c_1 = 1 + \frac{1}{3}c_1 + \frac{2}{3}c_{11}. \quad (8)$$

Similarly for Step B from (7),

$$2c_{11} = \frac{7}{3} + \frac{2}{3}c_1 + \frac{1}{3}c_{11}. \quad (9)$$

We can understand c_s to be the expected number of color comparisons used over the course of Stage I attributable to a single triple of type s . For example, using Step A (Figure 1), the expected number of color comparisons attributable to two triples of type 1 merged at the root is, on the one hand, $2c_s$ by definition, and on the other hand must be $1 + c_1/3 + 2c_{11}/3$ from the tree structure; equating these two expressions gives (8). A similar examination of Step B (Figure 2) gives (9).

Thus we can read these *characteristic equations* (8) and (9) directly from the roots and the leaves of Figures 1 and 2, respectively, by ignoring the sizes of subsets in all cases. Solving the simultaneous equations (8) and (9) gives us

$$\begin{aligned} c_1 &= \frac{29}{21} \\ c_{11} &= \frac{41}{21}. \end{aligned}$$

Stage I begins with n triples of type 1 and none of type 11, so the value of c_1 tells us the coefficient of n in the expected performance of Stage I of the algorithm. In the remainder of the paper we will simply solve the appropriate characteristic equations to get the coefficient of n in the expected behavior.

4. A REFINED ALGORITHM

We need to extend our notation for triples of subsets to the case in which one of the subsets is empty, another contains some elements, and the third contains twice as many elements; we call this type 21, read “two-one”; if the two non-empty subsets contain m and $2m$ elements, respectively, we call it type $m \times 21$. We use “type 21” to refer generically to triples of type $m \times 21$ for arbitrary m . As with triples of types $m \times 1$ and $m \times 11$, the discarding that can occur means that a triple of type $m \times 21$ corresponds to *at least* $3m$ elements.

The Refined Algorithm consists of an initial stage, Stage I, in which triples of types $m \times 1$, $m \times 11$, and $m \times 21$ are aggregated into larger and larger triples. Stage I is followed by a clean-up phase, Stage II that we describe below. In Stage I, we use Step A of our Simple Algorithm from the previous section, together with a new Step B shown in Figure 3 and Step C shown in Figure 4:

- (1) Find the smallest multiplier m for which Step C can be applied, and apply it.
- (2) If there is no multiplier m for which Step C can be applied, find the smallest multiplier m for which Step B can be applied, and apply it.
- (3) If there is no multiplier m for which Steps B or C can be applied, find the smallest multiplier m for which Step A can be applied, and apply it.

When there is no multiplier m for which any of Steps A, B, or C can be applied, Stage I ends and then Stage II of the algorithm uses at most three color comparisons per triple to count the number of elements of each color among the remaining triples, keeping a representative element of each color. The plurality color, and a representative element of that color, is then evident. As with the Simple Algorithm, the correctness of the Refined Algorithm is immediate.

Extending our notation from the Simple Algorithm, let e_{21} be the expected value of $|\mathcal{S}_{21}|$ at the end Stage I. We will see in Lemma 4.2 that e_1 and e_{21} are both

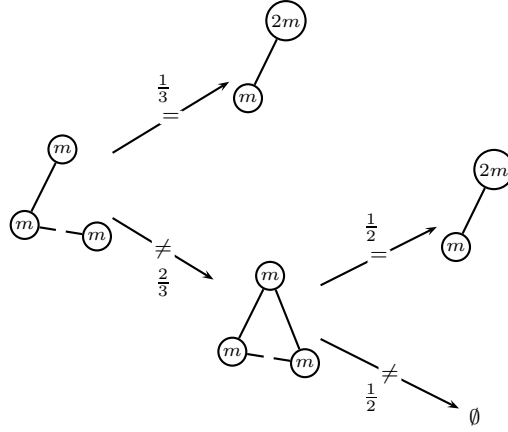


Fig. 3. Refined Algorithm, Step B; characteristic equation $c_{11} + c_1 = \frac{5}{3} + \frac{2}{3}c_{21}$. We use \emptyset to show a leaf when no triple is produced.

$O(\log^2 n)$ at the beginning of Stage I of the Refined Algorithm; similarly, as from Lemma 4.3, e_{11} is $O(\sqrt{n})$.

LEMMA 4.1. *The expected number of color comparisons used in Stage I of the Refined Algorithm is $\frac{229}{175}(n - e_1) - \frac{62}{35}e_{11} - \frac{53}{25}e_{21}$.*

PROOF. We follow the model of the previous section and solve the set of simultaneous characteristic equations for Steps A, B, and C for the Refined Algorithm, as given in the captions of Figures 1, 3, and 4, respectively:

$$\begin{aligned} 2c_1 &= 1 + \frac{1}{3}c_1 + \frac{2}{3}c_{11} \\ c_{11} + c_1 &= \frac{5}{3} + \frac{2}{3}c_{21} \\ 2c_{21} &= \frac{8}{3} + \frac{1}{6}c_1 + \frac{1}{6}c_{11} + \frac{1}{2}c_{21}, \end{aligned}$$

obtaining

$$\begin{aligned} c_1 &= \frac{229}{175} \\ c_{11} &= \frac{62}{35} \\ c_{21} &= \frac{53}{25}. \end{aligned}$$

The lemma follows. \square

To analyze Stage II of the Refined Algorithm is more complicated than in the Simple Algorithm because the set of triples that remains at the end of Stage I is much more complex. We use two lemmas to prove that the expected number of color comparisons in Stage II is $O(\sqrt{n})$.

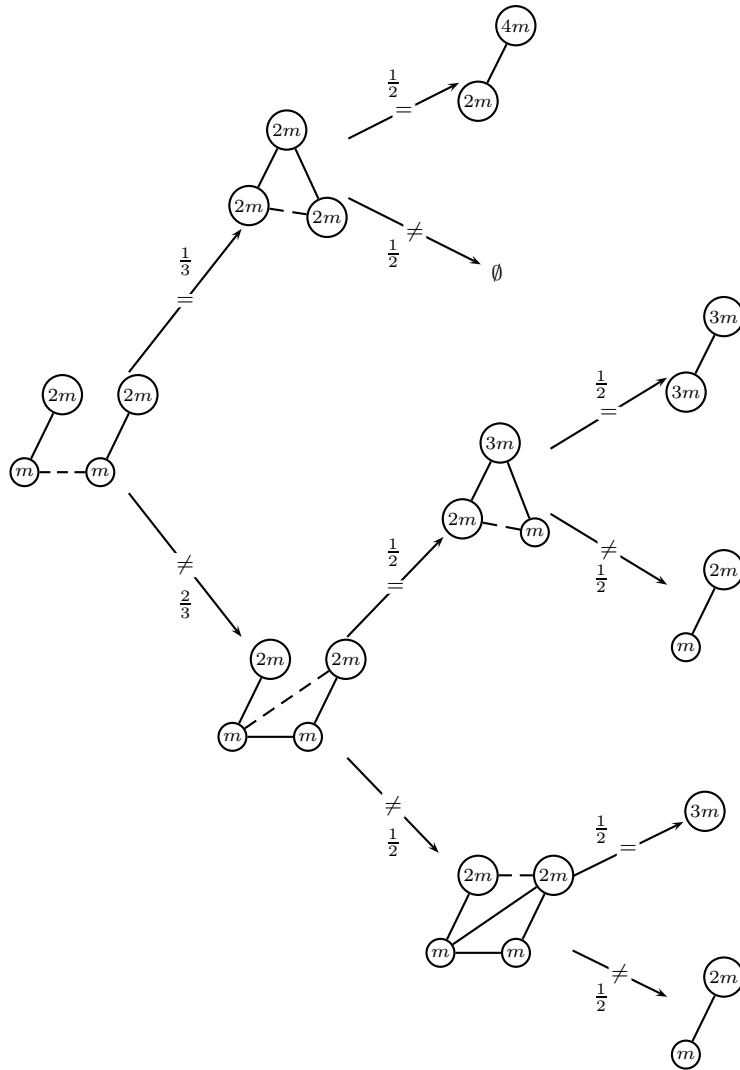


Fig. 4. Refined Algorithm, Step C; characteristic equation $2c_{21} = \frac{8}{3} + \frac{1}{6}c_1 + \frac{1}{6}c_{11} + \frac{1}{2}c_{21}$.

LEMMA 4.2. *The number of color comparisons used in Stage II of the Refined Algorithm to merge triples of types 1, $2^k \times 11$, and 21 is $O(\log^2 n)$.*

PROOF. By the nature of Steps A, B, and C, all triples have multipliers of $m \leq n$ and $m = 2^k 3^l$, $k \geq 0$, $l \geq 0$. Such m are the “Pratt increments” for Shell sort [Knuth 1998, exercise 5.2.1-30]: there are $\frac{1}{2}(\log_2 n)(\log_3 n) + O(\log n)$ of them. Because Stage I continues until none of the steps can be applied, there can be at most one triple of type $m \times 1$ or type $m \times 21$; these require two or three color comparisons, respectively, to merge in Stage II. Thus Stage II needs $O(\log^2 n)$ color comparisons to merge triples of types 1 and 21.

For triples of type $2^k \times 11$, we note that there can be at most one for each value of $k \leq \log_2 n$: consider the moment a second such triple could be constructed—it could only come from Step A combining two triples of type $2^k \times 1$, but if such a triple exists, it must be combined with the existing triple of type $2^k \times 11$ using Step B because Step B has priority over Step A in the Refined Algorithm. Triples of type 11 need three color comparisons to be merged in Stage II, so $O(\log n)$ color comparisons to merge triples of type $2^k \times 11$. \square

The situation is more intricate for triples of type $2^k 3^l \times 11$, $k \geq 0$, $l > 0$, because there can be multiple triples for each pair of k and l values. On the average, however, there will not be too many.

LEMMA 4.3. *The expected number of all triples of type $m \times 11$, $m = 2^k 3^l$, $k \geq 0$, $l > 0$, in Stage II of the Refined Algorithm is at most $O(\sqrt{n})$.*

PROOF. We first show that the expected number of triples of type $m \times 11$, for any fixed $m = 2^k 3^l$, $k \geq 0$, $l > 0$, in Stage II of the Refined Algorithm is at most $\sqrt{\frac{n}{2\pi m}} + O(1)$.

Triples of type $m \times 11$ are created only by Steps A and C, but as in the latter part of the proof of Lemma 4.2, the priority of Step B over Step A means that only one such triple in Stage II can arise from Step A. Thus all triples in Stage II of type $m \times 11$, $m = 2^k 3^l$, $k \geq 0$, $l > 0$, except one, arise from Step C. Applying Step C requires two triples of type 21. Let $\mathcal{S}_{m \times 21}$ be the set of triples of type $m \times 21$ formed by Step B from a triple of type $m \times 11$ and a triple of type $m \times 1$, or formed by Step C from two triples of type $\frac{m}{2} \times 21$; these $|\mathcal{S}_{m \times 21}|$ triples are disjoint, and each contains at least $3m$ elements, so that $|\mathcal{S}_{m \times 21}| \leq \frac{n}{3m}$. Step C can either reduce the number of type $m \times 21$ triples by one, or can reduce it by two while leaving a triple of type $2m \times 21$, $3m \times 11$, $3m \times 1$, or the empty triple, none which can ever lead to a triple of type $m \times 11$. Thus Step C can be applied at most $|\mathcal{S}_{m \times 21}| \leq \frac{n}{3m}$ times.

Define

$$p_{ij} = \begin{cases} \text{probability that Step C was applied } i \text{ times to types} \\ \frac{m}{3} \times 21 \text{ and gave either a type } m \times 1 \text{ or type } m \times 11 \\ \text{exactly } j \text{ times (that is, gave a type 21 or nothing} \\ \text{exactly } i - j \text{ times)} \end{cases}$$

Clearly, $\sum p_{ij} = 1$ and $p_{ij} = 0$ for $i < j$. Also, from the the discussion in the previous paragraph, $p_{ij} = 0$ for $i > \frac{n}{3(m/3)} = n/m$. The conditional probability that Step C yields a type $m \times 11$, given that it yields either a type $m \times 1$ or type $m \times 11$, is $1/2$, so the conditional probability that Step C created t triples of type $m \times 1$ and $j - t$ triples of $m \times 11$ is $\binom{j}{t}/2^j$. In such a case, Step B would be applied to those j triples at least $\min(t, j - t)$ times, leaving no triples of type $m \times 11$ if $t > j - t$ and at most $j - 2t$ triples of type $m \times 11$ otherwise. In other words, at most $\max(0, j - 2t)$ triples of type $m \times 11$ would remain untouched by Step B. Thus the expected number of triples of type $m \times 11$ that remain at the end of Stage I of the Refined Algorithm is at most

$$\sum_{0 \leq j \leq i \leq n/m} p_{ij} \sum_{t=0}^j \max(0, j - 2t) \binom{j}{t} / 2^j \leq \sum_{0 \leq j \leq i \leq n/m} p_{ij} \sum_{t=0}^{j/2} (j - 2t) \binom{j}{t} / 2^j \quad (10)$$

The inner sum can be evaluated by elementary combinatorics together with Stirling's formula—this inner sum was needed in [Alonso et al. 1997, equation (11)] and was the subject of problem A-4 in the 1974 Putnam competition [Hillman 1975] (see also [Alexanderson et al. 1985, pages 22 and 87]); it is,

$$\sum_{t=0}^{j/2} (j-2t) \binom{j}{t} / 2^j = \sqrt{\frac{j}{2\pi}} + O(1).$$

Thus (10) is bounded above by

$$\begin{aligned} \sum_{0 \leq j \leq i \leq n/m} p_{ij} \left(\sqrt{\frac{j}{2\pi}} + O(1) \right) &\leq \sum_{0 \leq j \leq i \leq n/m} p_{ij} \left(\sqrt{\frac{n/m}{2\pi}} + O(1) \right) \\ &\leq \sqrt{\frac{n}{2\pi m}} + O(1). \end{aligned}$$

Thus the expected number of all triples of type $2^k 3^l \times 11$, $k \geq 0$, $l > 0$ is at most

$$\begin{aligned} \sum_{\substack{0 \leq k \leq \log_2 n \\ 0 < l \leq \log_3 n}} \left[\sqrt{\frac{n}{2^{k+1} 3^l \pi}} + O(1) \right] &= \sqrt{\frac{n}{\pi}} \sum_{\substack{0 < k \leq \log_2 n \\ 0 < l \leq \log_3 n}} \frac{1}{\sqrt{2^k \sqrt{3^l}}} + O(\log^2 n) \\ &\leq \sqrt{\frac{n}{\pi}} \left(\sum_{k \geq 0} \frac{1}{\sqrt{2^k}} \right) \left(\sum_{l \geq 0} \frac{1}{\sqrt{3^l}} \right) + O(\log^2 n) \\ &= O(\sqrt{n}), \end{aligned}$$

as claimed. \square

THEOREM 4.4. *If all 3^n possible colorings are equally probable, the expected number of color comparisons for the Refined Algorithm is $\frac{229}{175}n + O(\sqrt{n})$.*

PROOF. From Lemma 4.1 we know that Stage I uses at most $\frac{229}{175}n$ color comparisons. By the nature of Steps A, B, and C, only triples of type 1, types $2^k 3^l \times 11$, $k \geq 0$, $l \geq 0$, and type 21 remain after Stage I and we know from Lemma 4.2 that merging types 1, $2^k \times 11$, and 21 use only $O(\log^2 n)$ color comparisons. From Lemma 4.3, the expected number of triples of type $2^k 3^l \times 11$ is $O(\sqrt{n})$. Merging these triples in Stage II uses three color comparisons per triple, for an expected number of $O(\sqrt{n})$ color comparisons. This implies that $\frac{229}{175}n + O(\sqrt{n})$ is an upper bound on the expected number of color comparisons.

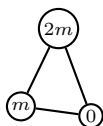
Using $e_1 = O(\log^2 n)$, $e_{11} = O(\sqrt{n})$, and $e_{21} = O(\log^2 n)$, Lemma 4.1 proves that $\frac{229}{175}n - O(\sqrt{n})$ is a lower bound. \square

5. A MORE SUBTLE ALGORITHM

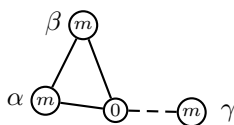
Our Refined Algorithm ignores some potentially useful information in Step C (Figure 4): To collapse the two triples of type $m \times 21$ into a single triple of type $m \times 21$ (the lowest leaf in Figure 4, for example), the algorithm discards m elements of each color. As we observed, such discarding does not affect the plurality, but we are left with a triple of type $m \times 21$ for which we have an element of the third color, in

contrast to triples of type $m \times 21$ that result from Step B of the Refined Algorithm (top two leaves in Figure 3) in which no element of the third color is available; by discriminating between the two cases we can get a better algorithm.

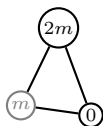
We depict a triple of type $m \times 21$ for which we know an element of the third color as



and allow the use of the element from $\textcircled{0}$ in color comparisons. However, such color comparisons can yield incomplete information. Consider the color comparison

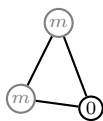


If the colors are equal, we have m elements of each of the three colors, so the $3m$ elements can be discarded and no triples remain (\emptyset). But, if the colors are unequal, we know only that the color of γ is the same as one of α or β , but not which one. We could, of course, use an extra color comparison to resolve the issue, but we do not bother (except for a strange case in Section 6 when $c = 4$). Instead, we just maintain a triple of type $m \times 21$ for which we do not have an element from the set corresponding to the 1 (we have an element from $\textcircled{0}$ and an element from γ corresponds to the 2). We write such a triple as $m \times 2\hat{1}$ and show it as



with the set corresponding to $\hat{1}$ in faint gray to indicate that, although we know it exists, it is a phantom—ghostlike, impalpable. A triple of type $2\hat{1}$ thus means that, of the three colors, the triple contains twice as many of a color₁ as of a color₂ and we have sample elements of color₁ and color₃, but no sample of color₂. For simplicity, we do not distinguish between triples $2\hat{1}$ and triples 21 (for which we have a sample of each of the colors)—we call them all type $2\hat{1}$; in other words, a type $2\hat{1}$ may have elements known for each of the three colors.

Similarly, we now allow a triple $m \times \hat{1}\hat{1}$, shown as



when we have an element from $\textcircled{0}$, but no specific element from either 1 set. Again, for simplicity, we do not distinguish between triples $\hat{1}\hat{1}$ and triples 11 for which we know an element of the third color—we call them all type $\hat{1}\hat{1}$; in other words, a type $\hat{1}\hat{1}$ may have elements known for each of the three colors.

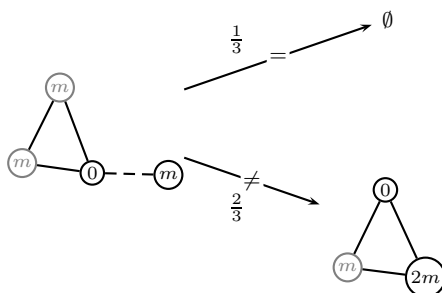


Fig. 5. More Subtle Algorithm, Step B'; characteristic equation $c_{\hat{1}\hat{1}} + c_1 = 1 + \frac{2}{3}c_{2\hat{1}}$. This is a variant of Step B from Figure 3 applied to type $m \times \hat{1}\hat{1}$ instead of type $m \times 11$. We show a set corresponding to $\hat{1}$ in faint gray to indicate that, although we know it exists, it is a phantom—impalpable in the sense that we may not have a specific element from it that we can use in color comparisons. The \emptyset indicates an empty set of the third color; though the set is empty, we do have an element of that color available for color comparisons.

We must reexamine Steps A, B, and C from the Refined Algorithm to adapt them to triples of the more general types $\hat{1}\hat{1}$ and $2\hat{1}$. For Step A (Figure 1), which combines two triples of type 1, no adaptation is needed. For Step B (Figure 3), which combines a triple of type 1 with a triple of type 11, we add Step B' shown in Figure 5 to combine a triple of type 1 with a triple of the more general type $\hat{1}\hat{1}$. For Step C (Figure 4), we modify it slightly, as shown in Figure 6, to show the altered leaves of the more general type $2\hat{1}$, and we must add Step C', a version of Step C that combines two type $2\hat{1}$ triples, as shown in Figure 7. Thus we obtain Stage I of the More Subtle Algorithm:

- (1) Find the smallest multiplier m for which either Step C or C' can be applied, and apply it.
- (2) If there is no multiplier m for which Step C or C' can be applied, find the smallest multiplier m for which Step B or B' can be applied, and apply it.
- (3) If there is no multiplier m for which Steps B, B', C, or C' can be applied, find the smallest multiplier m for which Step A can be applied, and apply it.

When there is no multiplier m for which any of Steps A, B, B', C, or C' can be applied, Stage I ends and then Stage II of the algorithm counts the number elements of each color in the remaining triples; the plurality color is then evident.

As with the Refined Algorithm, the correctness of the More Subtle Algorithm is immediate. But how, in Stage II, do we count elements among triples of types $\hat{1}\hat{1}$ or $2\hat{1}$ which have phantom sets? The type $\hat{1}\hat{1}$ triple is created only in Step C', and it is, in fact, a type 11 triple for which we happen to have an element of the remaining color; thus it can be handled as in Stage II of the Refined Algorithm. For type $2\hat{1}$ triples we know (at least) an element of two of the three constituent sets, so we can count these sets appropriately. The remaining set must be of the remaining color, so it too can be properly counted. At the end of Stage II, then, the counts tell us the plurality color, and give us a representative element *unless* we have the exceptional case in which the plurality color comes *only* from the $\hat{1}$ sets of triples of type $2\hat{1}$. However, such $\hat{1}$ sets account for just one third of the elements

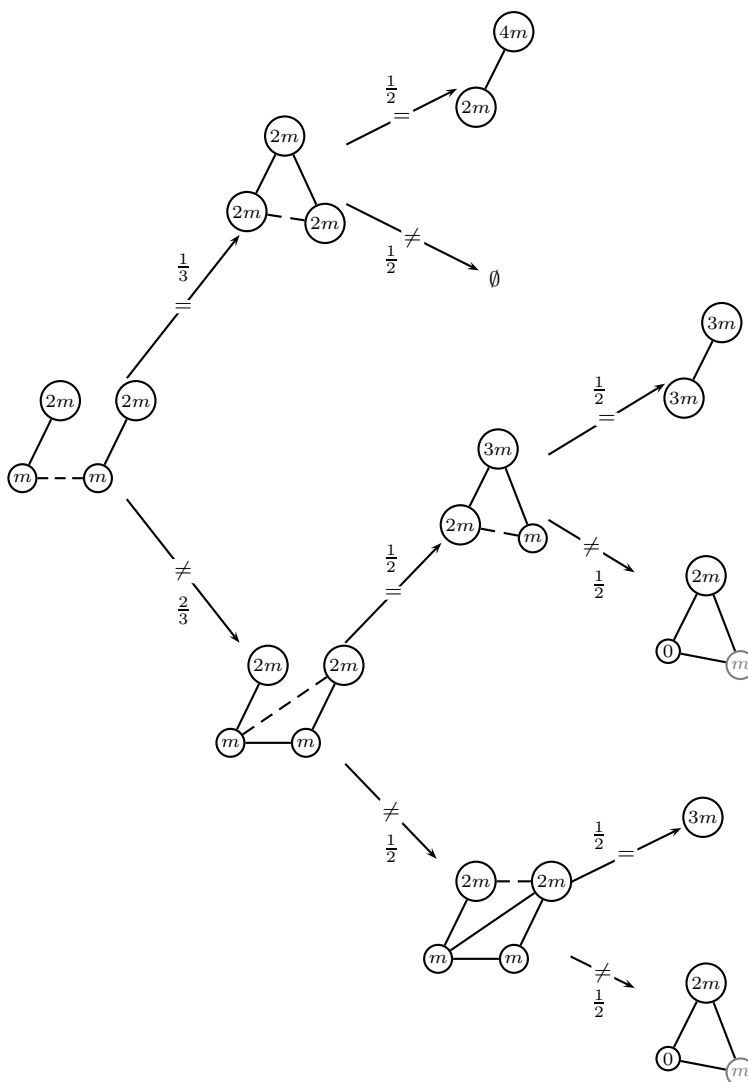


Fig. 6. More Subtle Algorithm, Step C; characteristic equation $2c_{21} = \frac{8}{3} + \frac{1}{6}c_1 + \frac{1}{6}c_{11} + \frac{1}{3}c_{21} + \frac{1}{6}c_{21}$. This is Step C from the Refined Algorithm (Figure 4), modified to show the more general outcomes of type $2\hat{1}$. Note that these $2\hat{1}$ triples are, in fact, type 21 triples in which we have an element of the third color, rather than type $2\hat{1}$ triples in which we do not have elements of the two 1-sets.

in type $2\hat{1}$ triples, so those elements can be no more than one third of all elements, and hence cannot be a plurality—in other words, in that exceptional case there is no plurality and the algorithm can so report.

THEOREM 5.1. *If all 3^n possible colorings are equally probable, the expected number of color comparisons for the More Subtle Algorithm is $\frac{7083}{5425}n + O(\sqrt{n})$.*

PROOF. The set of simultaneous characteristic equations for Steps A, B, B', C,

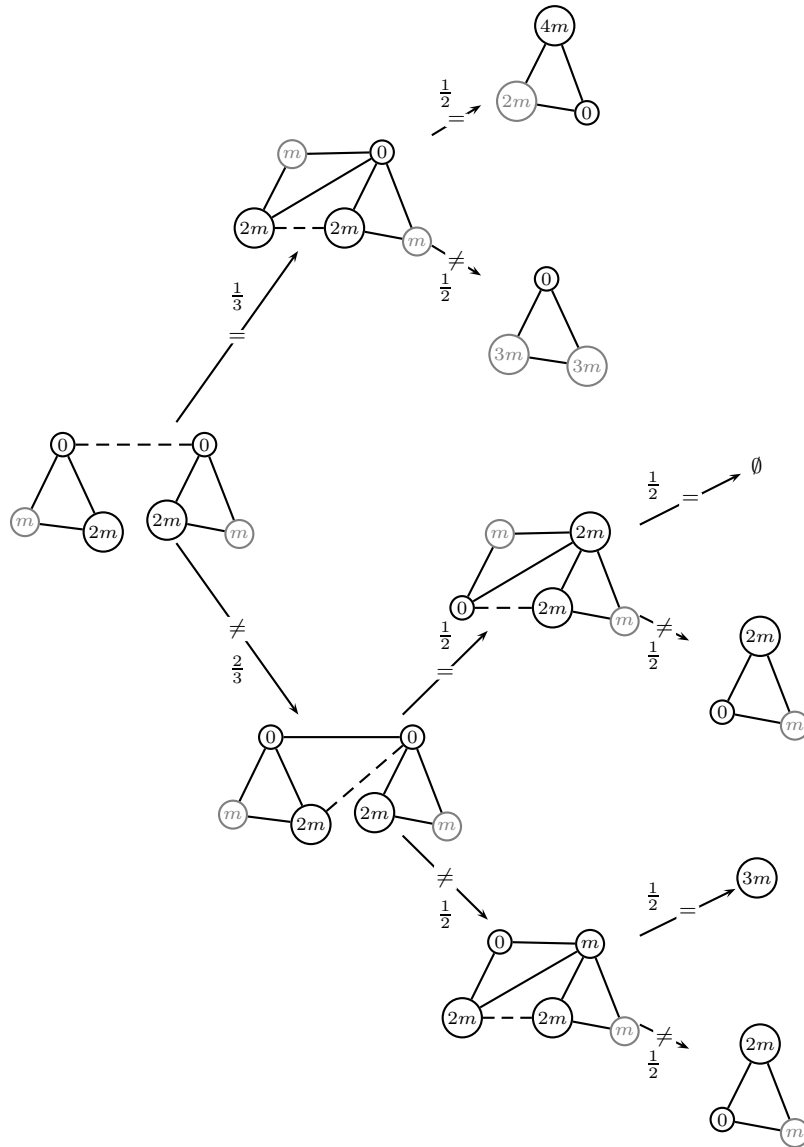


Fig. 7. More Subtle Algorithm, Step C'; characteristic equation $2c_{2\hat{1}} = \frac{8}{3} + \frac{1}{6}c_1 + \frac{1}{6}c_{1\hat{1}} + \frac{1}{2}c_{2\hat{1}}$. This is a variant of Step C from Figure 6 applied to type $m \times 2\hat{1}$ instead of type $m \times 21$. This step can be applied to triples of type 21, if we have an element of the third color in both cases.

and C' for the More Subtle Algorithm, as given in the captions of Figures 1, 3, 5, 6, and 7, respectively, are

$$\begin{aligned} 2c_1 &= 1 + \frac{1}{3}c_1 + \frac{2}{3}c_{11} \\ c_{11} + c_1 &= \frac{5}{3} + \frac{2}{3}c_{21} \\ c_{\hat{1}\hat{1}} + c_1 &= 1 + \frac{2}{3}c_{2\hat{1}} \\ 2c_{21} &= \frac{8}{3} + \frac{1}{6}c_1 + \frac{1}{6}c_{11} + \frac{1}{3}c_{2\hat{1}} + \frac{1}{6}c_{21} \\ 2c_{2\hat{1}} &= \frac{8}{3} + \frac{1}{6}c_1 + \frac{1}{6}c_{\hat{1}\hat{1}} + \frac{1}{2}c_{2\hat{1}}, \end{aligned}$$

which has solution

$$\begin{aligned} c_1 &= \frac{7083}{5425} \\ c_{11} &= \frac{1914}{1085} \\ c_{\hat{1}\hat{1}} &= \frac{1144}{1085} \\ c_{21} &= \frac{1631}{775} \\ c_{2\hat{1}} &= \frac{51}{25}. \end{aligned}$$

Thus the expected number of color comparisons used in Stage I of the More Subtle Algorithm is at most $\frac{7083}{5425}n$. Because the structure of the trees in Figures 3 and 5 is identical (only the relative probabilities of the two types of resulting triples varies) and because the structure of the trees in Figures 6 and 7 is identical if we do not distinguish type 11 from $\hat{1}\hat{1}$ and type 21 from $2\hat{1}$, Lemmas 4.2 and 4.3 hold for combined types 11 and $\hat{1}\hat{1}$ and combined types 21 and $2\hat{1}$. Thus as in Theorem 4.4, Stage II uses at most an expected number of $O(\sqrt{n})$ color comparisons.

The lower bound follows as in Theorems 3.4 and 5.1. \square

We can obtain a slightly deeper analysis as well: we can compute the average numbers of times each type of step is applied and the average number of triples created of each type. Let n_A be the average number of time Step A is applied, n_B the average number of time Step B is applied, and so forth. Stage I begins with n triples of type 1, and (from the characteristic equations) the average number of triples of type 1 at the end of Stage I is

$$n - 2n_A + \frac{1}{3}n_A - n_B - n_{B'} + \frac{1}{6}n_C + \frac{1}{6}n_{C'}.$$

But we also know from Lemma 4.2 (as altered to apply to the More Subtle Algorithm) that this is a $O(\log^2 n)$. Similar arguments for the other types of triples

yields the simultaneous equations

$$\begin{aligned} n - \frac{5}{3}n_A - n_B - n_{B'} + \frac{1}{6}n_C + \frac{1}{6}n_{C'} &= O(\log^2 n) \\ \frac{2}{3}n_A - n_B + \frac{1}{6}n_C &= O(\sqrt{n}) \\ -n_{B'} + \frac{1}{6}n_{C'} &= O(\sqrt{n}) \\ \frac{2}{3}n_B - \frac{11}{6}n_C &= O(\log^2 n) \\ \frac{2}{3}n_{B'} + \frac{1}{3}n_C - \frac{3}{2}n_{C'} &= O(\log^2 n), \end{aligned}$$

with solution

$$\begin{aligned} n_A &= \frac{3}{7}n + O(\sqrt{n}) \\ n_B &= \frac{66}{217}n + O(\sqrt{n}) \\ n_{B'} &= \frac{24}{5425}n + O(\sqrt{n}) \\ n_C &= \frac{24}{217}n + O(\sqrt{n}) \\ n_{C'} &= \frac{144}{5425}n + O(\sqrt{n}). \end{aligned}$$

These values allow us to compute the expected number of triples of each type created in Stage I. For example, there are $n_A/3 + n_C/6 + n_{C'}/6$ triples of type 1. We thus obtain

$$\begin{aligned} n_1 &= \frac{29}{175}n + O(\sqrt{n}) \\ n_{11} &= \frac{66}{217}n + O(\sqrt{n}) \\ n_{\hat{1}\hat{1}} &= \frac{24}{5425}n + O(\sqrt{n}) \\ n_{21} &= \frac{48}{217}n + O(\sqrt{n}) \\ n_{2\hat{1}} &= \frac{272}{5425}n + O(\sqrt{n}), \end{aligned}$$

where n_1 is the expected number of triples of type 1 created in Stage I, and so on.

Moreover, the average number of color comparisons used in Stage I can be expressed as

$$n_A + \frac{5}{3}n_B + n_{B'} + \frac{8}{3}n_C + \frac{8}{3}n_{C'} = \frac{7083}{5425}n + O(\sqrt{n})$$

This differs from the complexity of the Refined Algorithm only in the separation of Step B from B' and the separation of Step C from C'. But the coefficients of n_C and $n_{C'}$ for Steps C and C' are equal, so the only difference is the coefficient of $n_{B'}$ which is 1 instead of 5/3, the coefficient of n_B . We conclude that it is the presence of triples of type $\hat{1}\hat{1}$ that give us the improvement of the More Subtle Algorithm

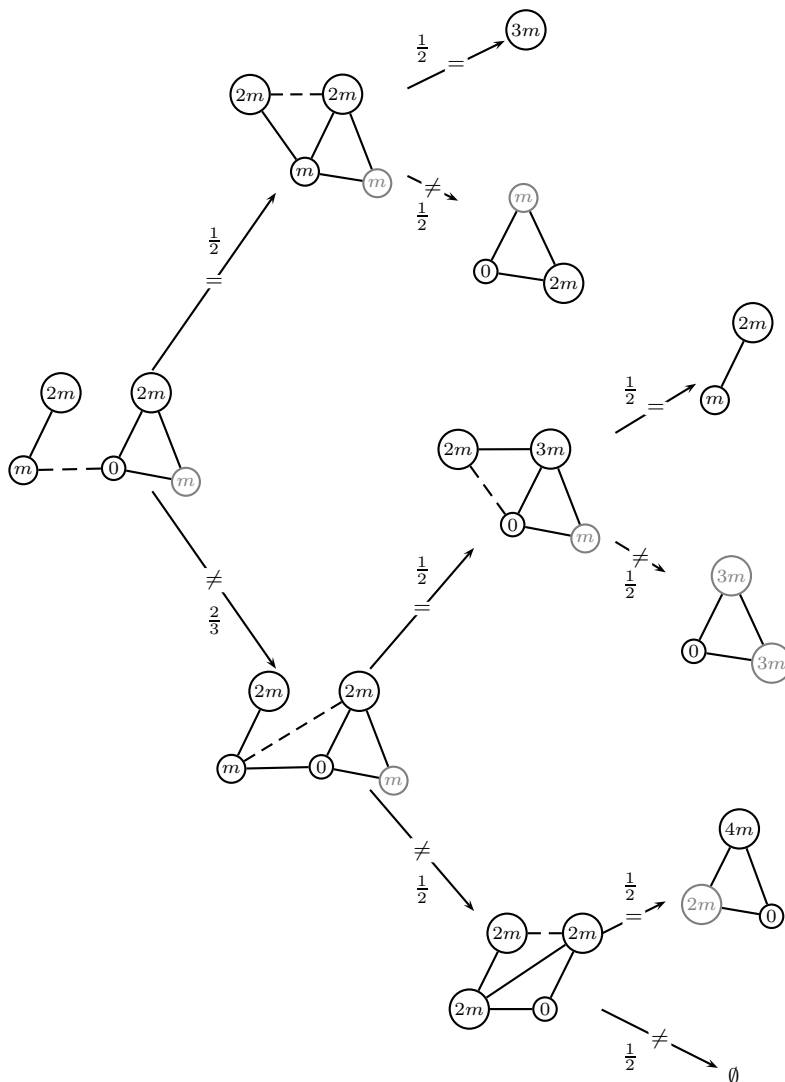


Fig. 8. Possible improvement of the More Subtle Algorithm, Step C'' ; characteristic equation $c_{21} + c_{2\hat{1}} = \frac{8}{3} + \frac{1}{6}c_1 + \frac{1}{6}c_{1\hat{1}} + \frac{1}{6}c_{21} + \frac{1}{3}c_{2\hat{1}}$. This is a variant of Step C from Figure 6 applied to a type $m \times 2\hat{1}$ and a type $m \times 21$; it is intended to create more triples of type $\hat{1}\hat{1}$.

over the Refined Algorithm; if we can form even more triples of this type and fewer of type 11 , we expect a lower complexity. It might be useful, therefore, to add a new, high-priority rule, Step C'' in Figure 8, to merge type 21 with type $2\hat{1}$; given two type 21 triples and two type $2\hat{1}$ triples, the algorithm would then apply Step C'' twice and create at least one triple of type $\hat{1}\hat{1}$ with probability $11/36$ and two such triples with probability $1/36$, while applying Steps C and C' would create such a triple with probability $1/6$ and could never create two of them.

Hence we could modify Stage I to be:

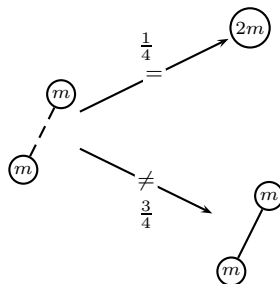


Fig. 9. Four Color Algorithm, Step A; characteristic equation $2c_1 = 1 + \frac{1}{4}c_1 + \frac{3}{4}c_{11}$.

- (1) Choose the smallest multiplier m such that one of Steps A, B, B', C, C', or C'' can be applied.
- (2) For this multiplier m , apply the step with the top priority, where Step B' has the highest priority, Step B is next highest, Step A is next, then Steps C'', C, and C'.

Unfortunately, the characteristic equations are then over-determined and our methods do not suffice to analyze this algorithm.

6. THE CASE OF FOUR COLORS

We can devise an algorithm for $c = 4$ colors similar to those for $c = 3$ colors in the previous sections, but with considerably more intricacy. Naturally, we must deal with quadruples, not triples. To eliminate some of the intricacy, our Stage I does not try to do anything subtle with certain quadruples. Because of the complexity in interpreting some quadruple types when four colors are involved, in the complicated figures in this section we label the root quadruples and the leaf quadruples with their types. Stage I of our Four Color Algorithm is:

- (1) For any multiplier m such that one of Steps A, B, or C from Figures 9, 10, or 11, respectively, can be applied, apply that step.
- (2) If none of these steps can be applied, merge all quadruples of types 31 and 332 using the Obvious Algorithm of Section 2.

When there is no multiplier m for which any of Steps A, B, or C can be applied and no quadruples of types 31 and 332, Stage I ends and then Stage II of the algorithm merges all remaining type 1, 11, and 211 quadruples, using at most six color comparisons per quadruple, to count the number of elements of each color among the remaining quadruples; a representative element of each color is maintained during this process. The plurality color, and a representative element of that color, is then evident. The correctness of the Refined Algorithm is immediate.

THEOREM 6.1. *If all 4^n possible colorings are equally probable, the expected number of color comparisons for the Four-Color Algorithm is $\frac{5413}{2814}n + O(\log n)$.*

PROOF. We cannot simply solve the characteristic equations given in Figures 9, 10, and 11 because there are five unknowns and only three equations. However, the Four Color Algorithm handles quadruples of types 31 and 332 with the Obvious

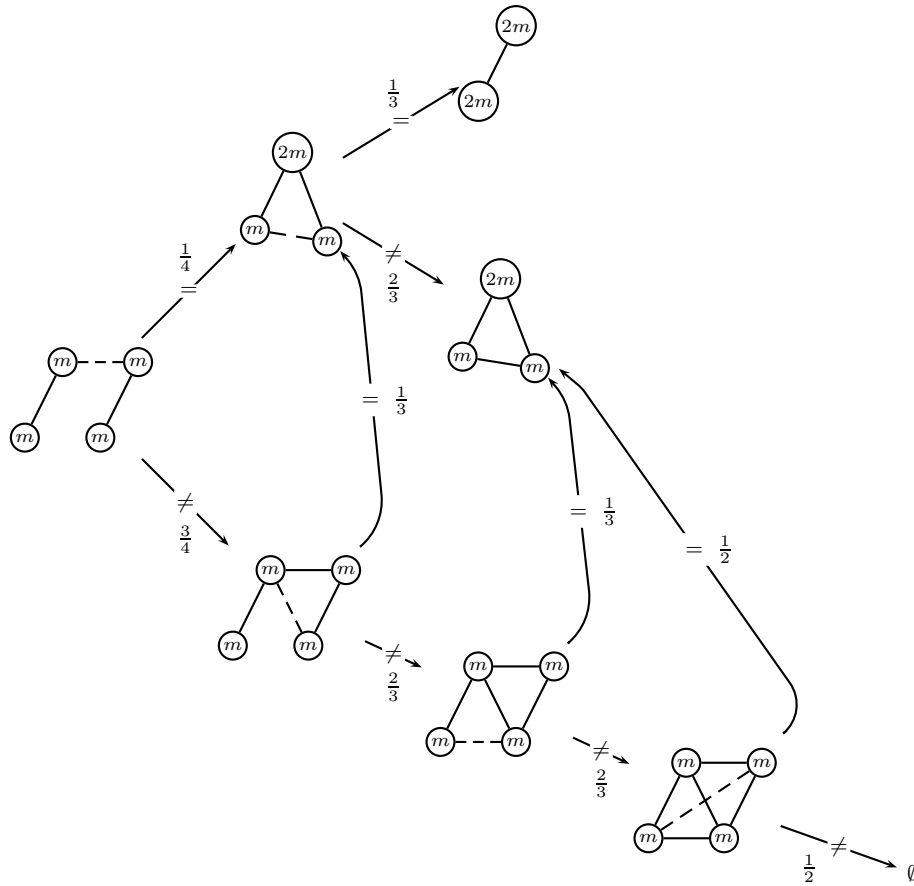


Fig. 10. Four Color Algorithm, Step B; characteristic equation $2c_{11} = \frac{37}{12} + \frac{2}{3}c_{211} + \frac{1}{6}c_{111}$.

Algorithm, so we know from equations (1) and (2) that inserting the first element of a quadruple will have an expected cost of at most

$$1 + \frac{3}{4} \left(1 + \frac{2}{3} \right) = \frac{9}{4},$$

color comparisons, the second at most $1 + 2/3 = 5/3$ color comparisons, and the third at most 1 color comparison. Thus we know

$$c_{31} \leq \frac{9}{4} + \frac{5}{3} = \frac{47}{12}$$

and

$$c_{332} \leq \frac{9}{4} + \frac{5}{3} + 1 = \frac{59}{12}. \tag{11}$$

However, for quadruples of type 332, after we have used the Obvious Algorithm to insert the first such quadruple, we will have at least three representative colors, so the remaining type 332 quadruples will be inserted with exactly $\frac{59}{12}$ expected com-

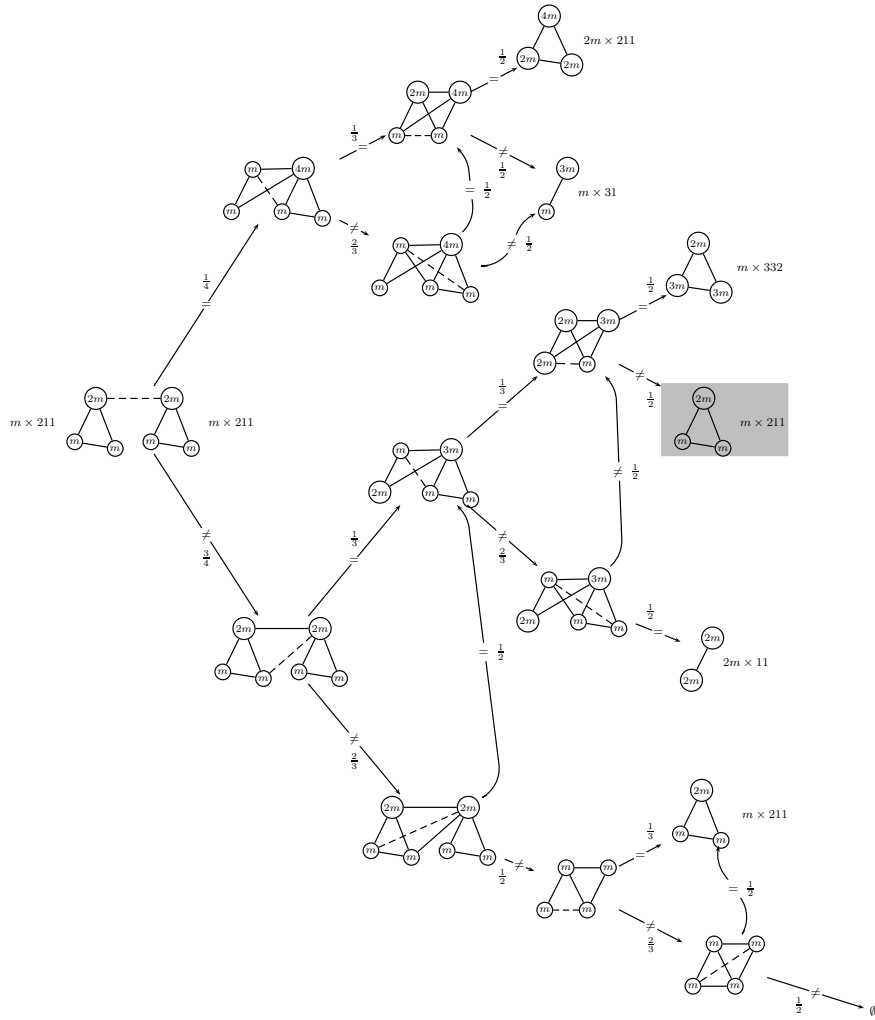


Fig. 11. Four Color Algorithm, Step C; characteristic equation $2c_{211} = \frac{53}{12} + \frac{5}{12}c_{211} + \frac{1}{6}c_{11} + \frac{1}{6}c_{31} + \frac{1}{6}c_{332}$. Because of the complexity in interpreting the quadruple types in this and subsequent figures, we label the root quadruples and the leaf quadruples with their types. In the gray-shaded leaf, of type 211, we have an element of the fourth color and if we do not need elements of the two sets of size m , we could consider this leaf to be of type $2\hat{1}\hat{1}$, as we do for the Refined Four Color Algorithm. In that case, the characteristic equation is $2c_{211} = \frac{53}{12} + \frac{1}{4}c_{211} + \frac{1}{6}c_{2\hat{1}\hat{1}} + \frac{1}{6}c_{11} + \frac{1}{6}c_{31} + \frac{1}{6}c_{332}$.

parisons. Thus the expected number of comparisons to insert all type 332 quadruples is at least $\frac{59}{12}(|S_{332}|-1)$ (“at least” because there may already be representatives of three or more colors when the first such quadruple is inserted). Similarly, because the probability is 6^{-k+1} that any k quadruples of type 31 contain among them only two different colors, the expected number of comparisons for inserting the type 31 quadruples is at least $\frac{47}{12}(|S_{31}|-1 - \sum_{k=0}^{\infty} 6^{-k}) = \frac{47}{12}|S_{31}| - O(1)$. So we must take

$$c_{31} = \frac{47}{12} \text{ and } c_{332} = \frac{59}{12}.$$

Combining these values for c_{31} and c_{332} with the characteristic equations for Steps A, B, and C,

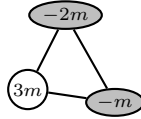
$$\begin{aligned} 2c_1 &= 1 + \frac{1}{4}c_1 + \frac{3}{4}c_{11} \\ 2c_{11} &= \frac{37}{12} + \frac{2}{3}c_{211} + \frac{1}{6}c_{11} \\ 2c_{211} &= \frac{53}{12} + \frac{5}{12}c_{211} + \frac{1}{6}c_{11} + \frac{1}{6}c_{31} + \frac{1}{6}c_{332}, \end{aligned}$$

we find $c_1 = \frac{5413}{2814}$, so that Stage I of the Four-Color Algorithm uses an average of $\frac{5413}{2814}n$ color comparisons. Stage I can leave only one quadruple each of types $m \times 1$, $m \times 11$, and $m \times 211$ for each multiplier m of the form $2^k < n$, so at most $O(\log n)$ color comparisons are used in Stage II. Thus, the expected number of color comparisons for the Four-Color Algorithm is $\frac{5413}{2814}n + O(\log n)$. \square

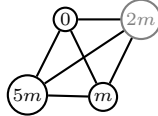
To refine the Four Color Algorithm we apply the idea used to derive the More Subtle Algorithm from the Refined Algorithm for three colors: the use of impalpable sets and $\textcircled{0}$ in quadruples. However the situation is more complex with four colors, so will simplify the diagrams by using a gray oval



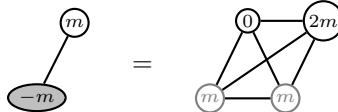
to represent a color for which there is a *deficit* of m elements in a quadruple compared to the other sets in the quadruple. Just as in a $\textcircled{0}$, we have an element of the deficit color available for color comparisons. For example, when we then write



it means that we have a quadruple in which one color has $3m + 2m = 5m$ elements, another has $-2m + 2m = 0$ elements, and a third has $-m + 2m = m$ elements; all of these colors have an element of that color available for color comparisons; but the implicit fourth color, of which we there $0 + 2m = 2m$ elements is not available for color comparisons. Thus this diagram is equivalent to $5\hat{2}1$, or



Similarly,



is a type $m \times 2\hat{1}\hat{1}$ quadruple. The gray oval notation for deficit sets could have been used in Figures 5–8, but the resulting simplifications would be minor because there

we deal only with three colors.

Stage I of our Refined Four Color Algorithm is then:

- (1) For any multiplier m such that one of Steps A, B, C, C', D, or D' from Figures 9–14, respectively, can be applied (using the alternative version of Figure 11 with the type $2\hat{1}\hat{1}$ leaf in place of the gray-shaded type 211 leaf), apply that step. However, the very first time a phantom quadruple occurs, we use an extra color comparison (which we never did in the case $c = 3$) to determine a sample element of each of color class, reserve those four elements, and set aside the quadruple until Stage II.
- (2) If none of these steps can be applied, merge the quadruple set aside in Step I and all quadruples of types 321, $3\hat{2}\hat{1}$, $5\hat{2}\hat{1}$, $5\hat{2}\hat{1}$ and 543 using the Obvious Algorithm of Section 2. If there are any phantom sets in quadruples, use the sample elements of the four colors reserved in Stage I to compare the known sets of the quadruple and thus accumulate totals of each color. If there are no phantom sets, the totals of each color can be accumulated directly.¹

When Stage I ends, Stage II merges any remaining quadruples of types 1, 11, 211, $2\hat{1}\hat{1}$, 31, $3\hat{1}$, 332, and $33\hat{2}$. After the counts of each color are completed in Stage II, the plurality color and a representative element of that color are then evident. The correctness of the Refined Four Color Algorithm is immediate.

THEOREM 6.2. *If all 4^n possible colorings are equally probable, the expected number of color comparisons for the Refined Four Color Algorithm is $\frac{761311}{402850}n + O(\sqrt{n})$.*

PROOF. We cannot simply solve the characteristic equations for Steps A, B, C, C', D, and D' (Figures 9–14) because there are thirteen unknowns and only six equations:

$$\begin{aligned} 2c_1 &= 1 + \frac{1}{4}c_1 + \frac{3}{4}c_{11} \\ 2c_{11} &= \frac{37}{12} + \frac{2}{3}c_{211} + \frac{1}{6}c_{11} \\ 2c_{211} &= \frac{53}{12} + \frac{1}{4}c_{211} + \frac{1}{6}c_{2\hat{1}\hat{1}} + \frac{1}{6}c_{11} + \frac{1}{6}c_{31} + \frac{1}{6}c_{332} \\ 2c_{2\hat{1}\hat{1}} &= \frac{37}{12} + \frac{5}{12}c_{2\hat{1}\hat{1}} + \frac{1}{6}c_{11} + \frac{1}{6}c_{3\hat{1}} + \frac{1}{6}c_{33\hat{2}} \\ c_{31} + c_{332} &= \frac{11}{3} + \frac{1}{6}c_{211} + \frac{1}{4}c_{2\hat{1}\hat{1}} + \frac{1}{6}c_{321} + \frac{1}{6}c_{5\hat{2}\hat{1}} + \frac{1}{6}c_{543} \\ c_{3\hat{1}} + c_{33\hat{2}} &= \frac{53}{12} + \frac{1}{12}c_{211} + \frac{1}{3}c_{2\hat{1}\hat{1}} + \frac{1}{6}c_{3\hat{2}\hat{1}} + \frac{1}{6}c_{5\hat{2}\hat{1}} + \frac{1}{6}c_{543}, \end{aligned}$$

However, notice that by lucky coincidence the pair of variables c_{31} and c_{332} is not independent; neither is the pair $c_{3\hat{1}}$ and $c_{33\hat{2}}$: the last two equations can be multiplied by $1/6$ and subtracted from the middle two equations, respectively, to leave four equations in nine unknowns. Also, as we did in the proof of Theorem 6.1, we can use the expected cost $59/12$ from (11), which applies to quadruples of all

¹We could also have used this method of maintaining a set of sample elements of each color to handle Stage II of the More Subtle Algorithm for $c = 3$ colors of Section 5.

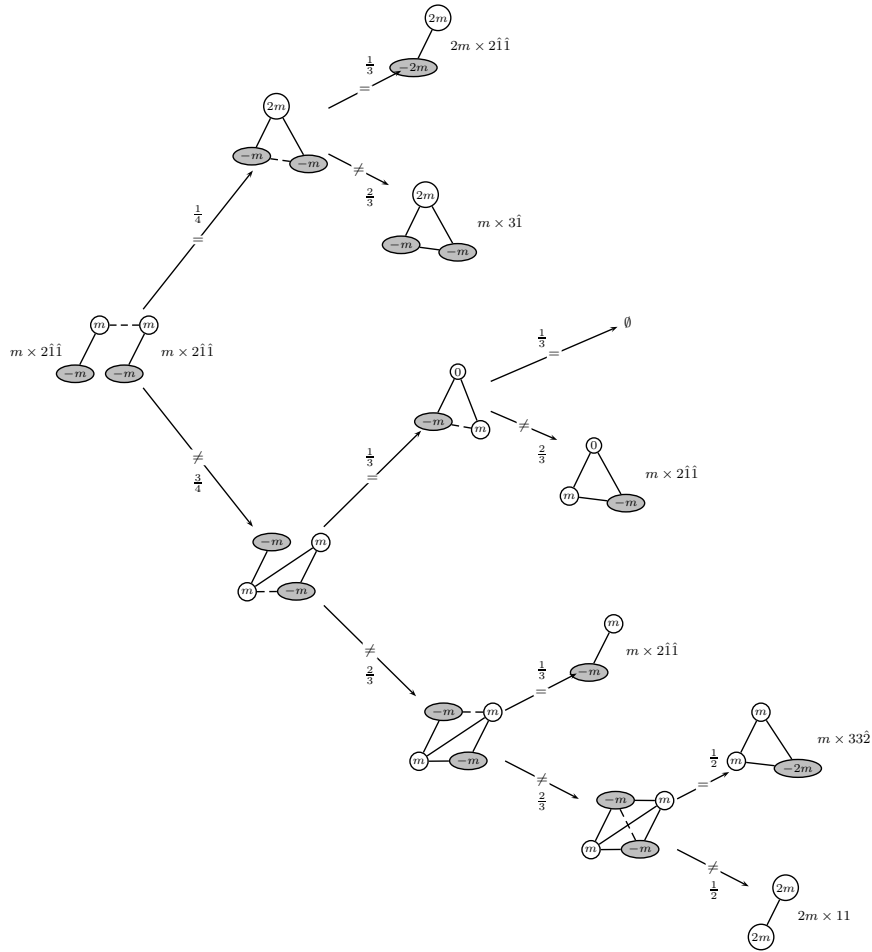


Fig. 12. Refined Four Color Algorithm, Step C'; characteristic equation $2c_{2\hat{1}\hat{1}} = \frac{37}{12} + \frac{5}{12}c_{2\hat{1}\hat{1}} + \frac{1}{6}c_{11} + \frac{1}{6}c_{3\hat{1}} + \frac{1}{6}c_{33\hat{2}}$.

types, to give us expected costs for the five types 321 , $3\hat{2}1$, $5\hat{2}\hat{1}$, $52\hat{1}$, and 543 . We thus find $c_1 = \frac{761311}{402850}$, so that Stage I of the Refined Four-Color Algorithm uses an average of $\frac{761311}{402850}n$ color comparisons.

The analysis of Stage II parallels the analysis of Stage II of the Refined Algorithm for $c = 3$ colors: there are at most $O(\log^2 n)$ quadruples of types 111 , 211 , and $2\hat{1}\hat{1}$, while the average number of quadruples of type 31 and type 332 (also type $3\hat{1}$ and type $33\hat{2}$ quadruples, respectively) is $O(\sqrt{n})$ by an analysis similar to that of the average number of remaining type 11 and $\hat{1}\hat{1}$ triples for the More Subtle Algorithm for $c = 3$. So, on the average, at most $O(\sqrt{n})$ color comparisons are needed in Stage II. This implies the expected number of color comparisons for the Refined Four-Color Algorithm is $\frac{761311}{402850}n + O(\sqrt{n})$. \square

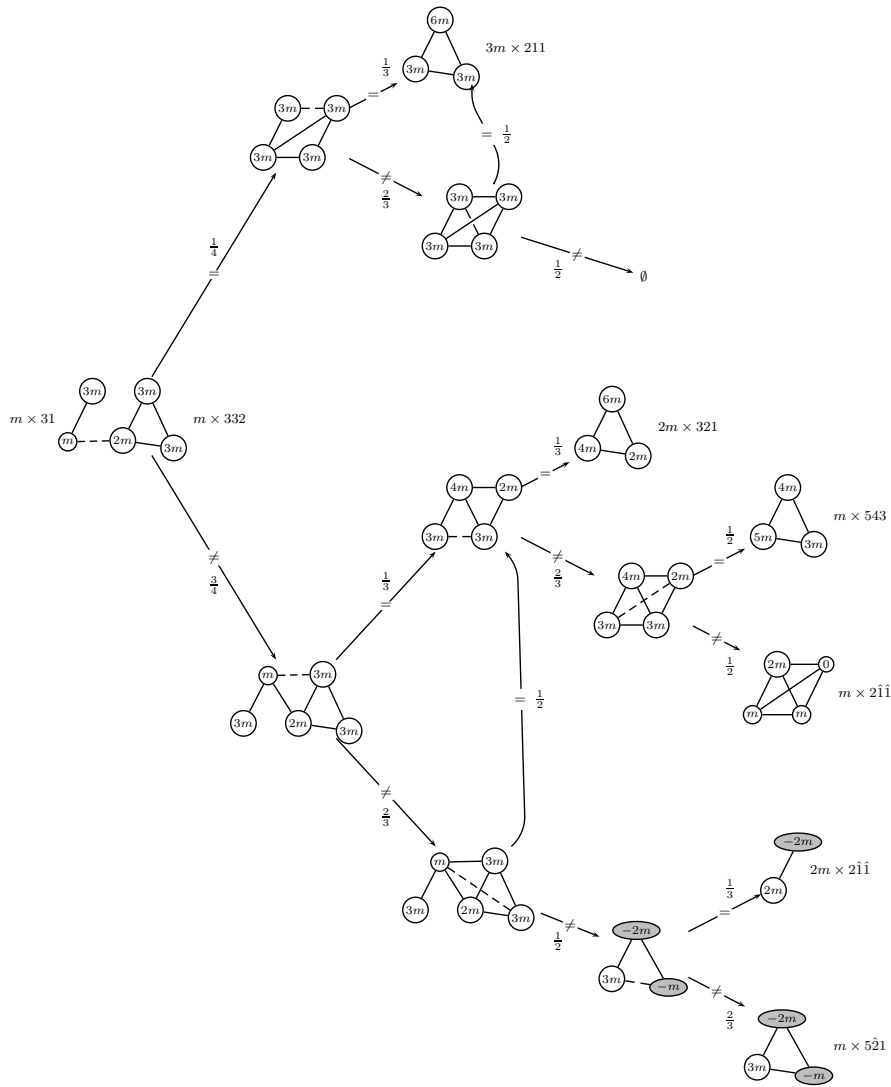


Fig. 13. Refined Four Color Algorithm, Step D; characteristic equation $c_{31} + c_{332} = \frac{11}{3} + \frac{1}{6}c_{211} + \frac{1}{4}c_{211i} + \frac{1}{6}c_{321} + \frac{1}{6}c_{521} + \frac{1}{6}c_{543}$

7. OPEN PROBLEMS

Most glaring is the open problem of narrowing the gap between our algorithms and the lower bounds of [Alonso and Reingold 2008a] for the expected number of color comparisons needed by a plurality algorithm for $c \geq 3$. Judging by the subtlety of the argument in [Alonso et al. 1997] where an exact lower bound for $c = 2$ (the majority problem) is proved, exact lower bounds for $c \geq 3$ will be hard to establish.

The Obvious Algorithm of Section 2 can be improved by discarding equal numbers of elements of each color class whenever possible—this will decrease the number

of non-empty color classes as the elements are processed, thus saving color comparisons. We conjecture that the savings is only $O(\sqrt{n})$ and hence would not change the main term of the performance compared to the Obvious Algorithm.

Except for the Obvious Algorithm, we are unable to determine the variances of the number of color comparisons used by our algorithms. What are they?

We are unable to analyze the complexity of the improvement to the More Subtle Algorithm of Section 5. What is its performance? We conjecture that it uses an average of $\frac{1201316741}{921984000}n + O(\sqrt{n}) \approx 1.302969 \dots n + O(\sqrt{n})$ color comparisons (this value comes from conjecturing that, except for triples of types 1×211 , 2×211 , 4×211 , 8×211 , and 3×211 , there are more type $m \times 2\hat{1}1$ triples than type $m \times 211$ triples and that Step C is only applied with very small probability; then, using some crude bounds on the number of triples types $m \times 211$ and $m \times 2\hat{1}1$ for multipliers $m = 1, 2, 4, 6, 8$, we bound the number of times Step C is applied). Can it be further improved or is it optimal for $c = 3$? We do not believe it to be optimal; we conjecture that the optimal algorithm for $c = 3$ uses an average of $\frac{5}{4}n$ color comparisons.

Algorithms of great complexity could be explored by exhaustive search. Using a computer should make it possible to find better algorithms for $c = 4$ (and maybe for $c = 3$).

The case $c \geq 5$ is beyond our patience, given our current methods—a more general formalization is needed. What gains can we expect compared to the Obvious Algorithm? The Obvious Algorithm appears to get more difficult beat as c increases; is it optimal at some point?

A. APPENDIX: VARIANCE OF THE OBVIOUS ALGORITHM

To illustrate the method used to calculate the variance of the Obvious Algorithm, we first use that method to obtain a tighter bound on the average. Let $D_a(n)$ be the expected number of color comparisons to insert n elements after a colors have already appeared. $D_0(n)$ is the average value we computed in the proof of Theorem 2.1; examining this generalization is the key to a more precise result. Let $\alpha = \frac{c^2+c-2}{2c}$. Then, as per our discussion in the proof of Theorem 2.1,

$$D_a(n) = \begin{cases} 0 & \text{if } n = 0 \text{ and } a \neq c, \\ \alpha + D_c(n-1) = \alpha n & \text{if } a = c, \\ \frac{a}{c}(D_a(n-1) + \frac{a+1}{2}) + \frac{c-a}{c}(D_{a+1}(n-1) + a) & \text{if } a < c. \end{cases}$$

Defining

$$\alpha_a = H_{c-a} - \frac{(c-a)(c-a+3)}{4},$$

we have

$$(c-a)\alpha_a + c\alpha = (c-a)\alpha_{a+1} + \frac{a(a+1)}{2} + a(c-a). \quad (12)$$

As we might guess from our analysis in Theorem 2.1, the dominant part of $D_a(n)$ turns out to be $\alpha n + \alpha_a$, so we define the difference $E_a(n) = D_a(n) - \alpha n - \alpha_a$, and analyze this error term. Using equation (12), the recurrence for $D_a(n)$ translates

to

$$E_a(n) = \begin{cases} -\alpha_a & \text{if } n = 0 \text{ and } a \neq c, \\ 0 & \text{if } a = c, \\ \frac{a}{c}E_a(n-1) + \frac{c-a}{c}E_{a+1}(n-1) & \text{if } a < c. \end{cases}$$

Except for the initial condition, this recurrence is identical to the recurrence for the probability that, when we insert n elements after a colors have already appeared, we still have fewer than c colors: Let that probability be $q_a(n)$. We have,

$$q_a(n) = \begin{cases} 1 & \text{if } n = 0 \text{ and } a \neq c, \\ 0 & \text{if } a = c, \\ \frac{a}{c}q_a(n-1) + \frac{c-a}{c}q_{a+1}(n-1) & \text{if } a < c, \end{cases}$$

so that

$$-\frac{c(c+3)}{4}q_a(0) \leq E_a(0) \leq 0 \quad (13)$$

and

$$-\frac{c(c+3)}{4}q_c(n) \leq E_c(n) \leq 0. \quad (14)$$

We can estimate $q_a(n)$ directly as

$$\begin{aligned} 0 \leq q_a(n) &\leq \frac{(c-a)(c-1)^n}{c^n} \\ &= (c-a) \left(1 - \frac{1}{c}\right)^n, \end{aligned}$$

with the upper bound coming from overcounting: select a color not among the a colors already present and color all elements with any of the $c-1$ colors other than that selected color. Using this estimate with (13) and (14) we have,

$$-\frac{c(c+3)}{4}q_a(n) \leq E_a(n) \leq 0,$$

for all a and n . Thus,

$$E_a(n) = O\left(c^3 \left(1 - \frac{1}{c}\right)^n\right) = o(1),$$

which gives

$$D_a(n) = \alpha n - \frac{(c-a)(c-a+3)}{4} + H_{c-a} + o(1). \quad (15)$$

Setting $a = 0$, we get the value for the expected number of color comparisons for the Obvious Algorithm stated in Theorem 2.1.

The technique of estimating the error can be repeated by taking

$$\hat{E}_a(n) = E_a(n) + (c-a) \left(1 - \frac{1}{c}\right)^n + \frac{(c-a)(c-a-1)}{4} \left(1 - \frac{2}{c}\right)^n,$$

showing that it is defined by the same recurrence as $E_a(n)$ but with initial condition $\hat{E}_a(0) = H_{c-a}$, and proving by induction that $0 \leq \hat{E}_a(n) \leq (c-a)(1-1/c)^n$ holds

for all a and n . We then obtain a slightly better low order term,

$$D_a(n) = \alpha n - \frac{(c-a)(c-a+3)}{4} + H_{c-a} + O\left(c\left(1 - \frac{1}{c}\right)^n\right).$$

Incidentally, we can relate the values of $q_0(n)$ and $\hat{E}_0(n)$ to the probabilities p_{ka} from (3). We extend the p_{ka} notation by defining p_{kab} as the probability that we obtain b colors when we insert k elements after a colors are already known. Then, $p_{na} = p_{n0a}$ and we obtain

$$q_a(n) = \sum_{b=0}^{c-1} p_{nab}$$

and

$$\hat{E}_a(n) = \sum_{b=0}^{c-1} H_{c-b} p_{nab}.$$

Now taking $a = 0$,

$$q_0(n) = \sum_{b=0}^{c-1} p_{nb}$$

and

$$\hat{E}_0(n) = \sum_{b=0}^{c-1} H_{c-b} p_{nb}.$$

We are now ready to derive the variance of the number of color comparisons for the Obvious Algorithm. Let \mathcal{C}_n be the set of the c^n colorings with n elements; for a given coloring, let x_i be the number of operations needed to insert the i th element. The variance is defined as

$$\frac{1}{c^n} \sum_{\mathcal{C}_n} \left(\sum_{i=1}^n x_i \right)^2 - \left(\frac{1}{c^n} \sum_{\mathcal{C}_n} \sum_{i=1}^n x_i \right)^2 = U(n) + V(n) - D_0(n)^2, \quad (16)$$

where

$$U(n) = \frac{1}{c^n} \sum_{\mathcal{C}_n} \left(\sum_{i=1}^n x_i \right)^2,$$

and

$$V(n) = \frac{2}{c^n} \sum_{\mathcal{C}_n} \sum_{i=1}^n x_i \sum_{j=i+1}^n x_j.$$

From (15),

$$D_0(n)^2 = \left(\alpha n - \frac{c(c+3)}{4} + H_c \right)^2 + o(1), \quad (17)$$

so we just need to evaluate $U(n)$ and $V(n)$.

We use the error estimation method illustrated above to compute $U(n)$. As we did for $D(n)$ above, generalize $U(n)$ to $U_a(n)$ when a colors have already appeared and define

$$\beta = \frac{(c-1)(2c^2 + 5c - 6)}{6c}.$$

We have, $U_a(0) = 0$,

$$\begin{aligned} U_c(n) &= \frac{1}{c} \left(\sum_{i=1}^{c-1} i^2 + (c-1)^2 \right) + U_c(n-1) \\ &= \beta + U_c(n-1), \end{aligned}$$

so that $U_c(n) = \beta n$; when $a \neq c$,

$$U_a(n) = \frac{a}{c} \left(\frac{(a+1)(2a+1)}{6} + U_a(n-1) \right) + \frac{c-a}{c} (a^2 + U_{a+1}(n-1))$$

because when elements are inserted with a colors already known, the cost squared is

$$\frac{1}{c} \left(\sum_{i=1}^a i^2 \right) = \frac{a(a+1)(2a+1)}{6c}$$

for an element of a known color, and a^2 for a newly discovered color.

Defining

$$\begin{aligned} \beta_a &= \frac{-8a^3 + 3(2c+7)a^2 + (12c^2 + 12c - 7)a - c(10c^2 + 33c - 7)}{36} \\ &\quad + (2c-1)H_{c-a}, \end{aligned}$$

we have $\beta_c = 0$ and

$$c\beta + (c-a)\beta_a = \frac{a(a+1)(2a+1)}{6} + (c-a)a^2 + (c-a)\beta_{a+1}.$$

Then we can define the error $\tilde{E}_a(n) = U_a(n) - \beta n - \beta_a$, obtaining

$$\tilde{E}_a(n) = \begin{cases} -\beta_a & \text{if } n = 0 \text{ and } a \neq c, \\ 0 & \text{if } a = c, \\ \frac{a}{c}\tilde{E}_a(n-1) + \frac{c-a}{c}\tilde{E}_{a+1}(n-1) & \text{if } a < c. \end{cases}$$

Noting that $\beta_a = O(c^3)$, we find $\tilde{E}_a(n) = O(c^4(1-1/c)^n) = o(1)$, and hence

$$U(n) = \beta n + \beta_0 + o(1) = \beta n - \frac{c(2c+7)(5c-1)}{36} + (2c-1)H_c + o(1). \quad (18)$$

Now we must compute $V(n)$. For any boolean expression Q , define $[Q] = 1$ if Q is true and $[Q] = 0$, if Q is false. Recall from (3) that p_{ka} is the probability that exactly a different colors are found among the first k elements. Using the definition of $V(n)$ and summing, for each k , the probability of the number of known colors at the k th insertion times the resulting cost, we have,

$$V(n) = 2 \sum_{k=1}^{n-1} \sum_{a=0}^c p_{(k-1)a} \left(\frac{a^2 + a - 2[a=c]}{2c} D_a(n-k) + \frac{c-a}{c} a D_{a+1}(n-k) \right)$$

(if the k th element is one of the a colors, the average cost for insertion is $\frac{a+1}{2}$, except when $a = c$ when it is $\frac{a+1}{2} - \frac{1}{c}$; if it is a new color, the average cost for insertion is a). We want to approximate $V(n)$ by replacing $D_a(n)$ by $\alpha n + \alpha_a$ in the above formula and must analyze the resulting error. The error turns out to be $o(1)$, as we now show.

First, suppose that $k \leq \sqrt{n}$, then $D_a(n-k) - \alpha n - \alpha_a = O(c(1-1/c)^{n-\sqrt{n}})$. Therefore the error caused by replacing $D_a(n)$ with $\alpha n + \alpha_a$ for $k \leq \sqrt{n}$ is

$$O\left(\sum_{k=1}^{\sqrt{n}} \sum_{a=0}^c p_{(k-1)a} c(1-1/c)^{n-\sqrt{n}}\right) = o(1).$$

Now, consider $k \geq \sqrt{n}$. When $a = c$, $D_c(n) = \alpha n + \alpha_a$ and there is no error; otherwise, the error is $O(c^3)$. For each value of $k \geq \sqrt{n}$,

$$\sum_{a=0}^c p_{(k-1)a} \leq c(1-1/c)^{\sqrt{n}},$$

so summing over $k \geq \sqrt{n}$ gives the total error as $O(nc(1-1/c)^{\sqrt{n}-1}c^3) = o(1)$.

Now define $V_a(n)$ by $V_a(1) = 0$, $V_c(n) = \alpha^2(n-1)n$, and for $a \neq c$,

$$\begin{aligned} V(n) &= \frac{a(2c-a+1)}{c}\alpha(n-1) + \frac{a(a+1)\alpha_a + 2a(c-a)\alpha_{a+1}}{c} \\ &\quad + \frac{a}{c}V_a(n-1) + \frac{c-a}{c}V_{a+1}(n-1). \end{aligned}$$

Proceeding as before, we find

$$\begin{aligned} V_{c-a}(n) &= \alpha^2 n(n-1) + \frac{\alpha}{2}n(-a^2 - 3a + 4H_a) \\ &\quad + \frac{a^4}{16} + \frac{a^3}{8} + \frac{(8c-8H_a-1)a^2}{16} + \frac{12c+7-12H_a}{8}a \\ &\quad - (2c+1)H_a + H_a^2 + H_a^{(2)} + o(1). \end{aligned}$$

Finally we can calculate the variance. We know that $V(n) = V_0(n) + o(1)$, so from (16), (17), and (18) we find the variance to be

$$\begin{aligned} U(n) + V_0(n) + o(1) - D_0(n)^2 &= \\ &= \frac{(c-1)(c-2)(c^2+3c-6)}{12c^2}n - \frac{c(c+7)(2c-11)}{72} - 2H_c + H_c^{(2)} + o(1), \end{aligned}$$

as claimed.

REFERENCES

- AIGNER, M., MARCO, G. D., AND MONTANGERO, M. 2005. The plurality problem with three colors and more. *Theoret. Comput. Sci.* 337, 1–3 (June), 319–330.
- ALEXANDERSON, G. L., KLOSINSKI, L. F., AND LARSON, L. C. 1985. *The William Lowell Putnam Mathematical Competition, Problems and Solutions: 1965–1984*. The Mathematical Association of America, Washington, D.C.
- ALONSO, L. AND REINGOLD, E. M. 2008a. Average-case lower bounds for the plurality problem. *ACM Trans. Algorithms* 4, 3 (Jun), 27–1–27–17.
- ALONSO, L. AND REINGOLD, E. M. 2008b. Determining plurality. *ACM Trans. Algorithms* 4, 3 (Jun), 26–1–26–19.

- ALONSO, L., REINGOLD, E. M., AND SCHOTT, R. 1993. Determining the majority. *Inf. Process. Lett.* 47, 253–255.
- ALONSO, L., REINGOLD, E. M., AND SCHOTT, R. 1997. The average-case complexity of determining the majority. *SIAM J. Comput.* 26, 1, 1–14.
- CHUNG, F., GRAHAM, R., MAO, J., AND YAO, A. 2007. Oblivious and adaptive strategies for the majority and plurality problems. *Algorithmica* 48, 2, 147–157. An earlier version appeared in *Computing and Combinatorics 11th Annual International Conference, COCOON 2005*.
- DVOŘÁK, Z., JELÍNEK, V., KRÁL', D., KYNČL, J., AND SAKS, M. 2007. Probabilistic strategies for the partition and plurality problems. *Random Struct. Algorithms* 30, 1-2, 63–77. An earlier version appeared in *22nd Annual Symposium on Theoretical Aspects of Computer Science, STACS 2005*.
- HILLMAN, A. P. 1975. The William Lowell Putnam mathematical competition. *Amer. Math. Monthly* 82, 9 (Nov.), 905–912.
- KNUTH, D. E. 1997a. *The Art of Computer Programming, Volume 1: Fundamental Algorithms*, 3rd ed. Addison Wesley Longman Publishing Co., Inc., Redwood City, CA, USA.
- KNUTH, D. E. 1997b. *The Art of Computer Programming, Volume 2: Seminumerical Algorithms*, 3rd ed. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.
- KNUTH, D. E. 1998. *The Art of Computer Programming, Volume 3: Sorting and Searching*, 2nd ed. Addison Wesley Longman Publishing Co., Inc., Redwood City, CA, USA.
- KRÁL', D., SGALL, J., AND TICHÝ, T. 2008. Randomized strategies for the plurality problem. *Discr. Appl. Math.* 0000, 00 (XXX), 000–000. An earlier version appeared as Report KAM-DIMATIA Series 2005-722 and ITI Series 2005-238, Charles University, Prague, 2005.
- SRIVASTAVA, N. AND TAYLOR, A. D. 2005. Tight bounds on plurality. *Inf. Process. Lett.* 96, 3, 93–95.

Received Month, Year; revised Month, Year; accepted Month, Year