Monte-Carlo algorithms in graph isomorphism testing

László Babai*

Eötvös University, Budapest, Hungary
Université de Montréal, Canada

Abstract

Abstract. We present an $O(V^4 \log V)$ coin flipping algorithm to test vertex-colored graphs with bounded color multiplicities for color-preserving isomorphism. We are also able to generate uniformly distributed random automorphisms of such graphs. A more general result finds generators for the intersection of cylindric subgroups of a direct product of groups in $O(n^{7/2} \log n)$ time, where n is the length of the input string. This result will be applied in another paper to find a polynomial time coin flipping algorithm to test isomorphism of graphs with bounded eigenvalue multiplicities. The most general result says that if $AutX$ is accessible by a chain $G_0 \geq \cdots \geq G_k = AutX$ of quickly recognizable groups such that the indices $|G_{i-1} : G_i|$ are small (but unknown), the order of $|G_0|$ is known and there is a fast way of generating uniformly distributed random members of G_0 then a set of generators of $AutX$ can be found by a fast algorithm. Applications of the main result improve the complexity of isomorphism testing for graphs with bounded valences to $\exp(n^{1/2+o(1)})$ and for distributive lattices to $O(n^{6 \log \log n})$.

*Apart from possible typos, this is a verbatim transcript of the author's 1979 technical report, **"Monte Carlo algorithms in graph isomorphism testing" (Université de Montréal, D.M.S. No. 79-10)**, with two footnotes added to correct a typo and a glaring omission in the original.

All algorithms depend on a series of independent coin flips and have a small probability of failure (reaching no decision), but, unlike for some classical Monte-Carlo algorithms, the correctness of the decision made can always be checked and we are not referred to the hope that events with small probability are practically impossible. We suggest the term "Las Vegas computation" for such strong Monte-Carlo procedures.

0 Monte Carlo or Las Vegas?

0.1

Fast Monte-Carlo algorithms to decide some interesting recognition problems have been around for a while now, the most notable among them being the Strassen-Solovay primality test [16].

One feature of this algorithm is that in case of a negative answer (the input number is decided to be prime) there is no check on the correctness of the answer. The situation is similar in the case when we wish to decide whether a multivariate polynomial vanishes identically, by substituting random numbers for the variables. (R. Zippel [18]). In some cases, however, we can check whether the decision reached was correct (as in Zippel's GCD algorithm [17]).

It may be worth distinguishing these two kinds of random algorithms by reserving the term "Monte-Carlo" for the Strassen-Solovay type algorithms. I propose the term "Las Vegas algorithm" for those stronger procedures where the correctness of the result can be checked. Adopting this terminology, a "coin-tossing" algorithm, computing a function $F(x)$ will be called a

Monte-Carlo algorithm if it has

INPUT: x (a string in a finite alphabet)

OUTPUT: y (believed to be equal to $F(x)$)

ERROR PROBABILITY: less than 1/3 (the error being $y \neq F(x)$)[1].

[1]Typo corrected: the original had 1/2 here. (Footnote added.)

We shall call the computation a *Las Vegas algorithm* if it has

INPUT: x

OUTPUT: either "?" or $F(x)$

PROBABILITY OF FAILURE: less than 1/2 (the failure meaning "?" output).

(Of course, repeatedly applying the algorithm t times, the probability of error/failure is reduced to less than 2^{-t}).

In particular, if our computation is to solve a recognition problem (i.e. $F(x) \in \{yes, no\}$) and its running time is a polynomial of the length of the input string[2], then F belongs to the class called RP by Adleman and Manders [1] if the algorithm is Monte-Carlo and it belongs to $\Delta^R = RP \cap coRP$ if the algorithm is Las Vegas.

Note that every Las Vegas computation is Monte Carlo, but not conversely.

0.2

In this paper we give (contrary to the title) *Las Vegas algorithms* to test isomorphism in certain classes of graphs. One of the applications of the main results of this paper (2.5, 5.2) will be a polynomial time Las Vegas isomorphism testing for graphs having bounded multiplicities of eigenvalues [6]. In particular, non-isomorphism is in NP for these classes of graphs (i.e. isomorphism is well characterized in the sense of Edmonds: the negative answer can also be verified in polynomial time) (since $\Delta^R \subseteq NP \cap coNP$). It is not known in general, whether non-isomorphism of two graphs on n vertices can be *proved* in $\exp(o(n))$ steps.

It is the author's dream that such a proof, though difficult, isn't out of reach anymore, and the present paper may be a contribution to that modest goal.

[2]We also need to assume that a "yes" answer is always correct. (Footnote added.)

0.3

To the author's knowledge, no coin-tossing algorithms have previously been known to solve truly combinatorial recognition problems, not solvable by any known polynomial time deterministic algorithm.

1 Introduction

1.1

Graph isomorphism testing by brute force takes $n!$ time. (n will always refer to the number of vertices.) Heuristic algorithms like the one treated in 4.4 are often used to classify the vertices and thereby reduce the number of trials needed (cf. [14]). If such a process splits the vertex set of a graph X into pieces of sizes $k_1, \ldots, k_r (\sum k_i = n)$ and the same happens to the graph Y then we are still left with $\prod_{i=1}^{r}(k_i!)$ bijections as candidates for an isomorphism $X \to Y$. It is frustrating that this number is exponentially large even if our vertex classification algorithm was as successful as to achieve $k_1 = \ldots = k_r = 3$, and we don't know of any deterministic algorithm that could test isomorphism of such pairs of graphs with so successfully classified vertices within subexponential time.

1.2

We are able, however, to give an $O(n^4 \log n)$ Las Vegas algorithm (for the case when all classes have bounded size). The precise statement of the problem to be solved is this. *Given two graphs with colored vertices, each color class having size $\leq k$, decide whether they admit a color preserving isomorphism.* The cost of our Las Vegas computation will be $O(k^{4k} n^4 \log n)$. (Theorems 3.1, 3.2). (We remark that for $k \leq 2$, there is a straightforward linear time deterministic algorithm.)

After the computation is done, we shall also be able to generate uniformly distributed random members of the automorphism group $AutX$ of the colored

graph X at $O(k!n^2)$ cost each. Moreover, the algorithm displays a set of generators of $AutX$.

1.3

The more general setting, covering both the vertex-colored graphs with small color multiplicities and the graphs with bounded eigenvalue multiplicities is the following.

Suppose $AutX$ is *polynomially accessible from a well-described group.* By this we mean that we have a chain of groups $G_0 \geq G_1 \geq \cdots \geq G_k = AutX (k < n^c)$ such that

(i) G_0 is *well-described*, i.e. we know its order $|G_0|$ and we are able to generate uniformly distributed random members of G_0 (cf. Def. 2.1);

(ii) the indices $|G_{i-1} : G_i|$ are small (uniformly bounded by a polynomial of n);

(iii) the groups G_i are *recognizable* (i.e., given a member of G_{i-1} we can decide in polynomial time whether it belongs to $G_i, i = 1, \ldots, k-1$).

Under these circumstances we proceed as follows (cf. Theorems 2.5 and 2.10).

We extend this chain by adding $G_k \geq G_{k+1} \geq \cdots \geq G_{k+n}$ where G_{k+j} is the stabilizer of the j^{th} vertex of X in $G_{k+j-1} (j = 1, \ldots, n)$. Clearly $|G_{k+n}| = 1$. By induction on i, we generate uniformly distributed random members of G_{i-1} in sufficient numbers such as to represent all left cosets of G_{i-1} mod G_i with high probability, and then select a complete set of left coset representatives. We use these coset representatives to generate uniformly distributed random members of G_i (once such elements of G_{i-1} are available) and continue. This way we compute the indices $r_i = |G_{i-1} : G_i$ with a possible error, but this will be eliminated by checking if $r_i = |Gi - 1 : G_i|$ but this can be eliminated by checking if $\prod_{i=1}^{k+n} r_i = |G_0|$. If not, we output "?"; if yes, we have all we wanted. $AutX$ is generated by the coset representatives below $G_k = AutX$.

1.4

We apply this procedure to obtain improved theoretical bounds on the complexity of isomorphism testing for distributive lattices ($n^{c \log\log n}$, Cor. 3.4), trivalent graphs ($\exp(cn^{1/2} \log n)$) and of graphs with bounded valence ($\exp(n^{1/2+o(1)})$, Theorem 4.1). The best previously known bound for these classes was $(1+c)^n$, c a positive constant (G.L. Miller [14]).

1.5

It is the author's hope that an $\exp((\log n)^c)$ Las Vegas isomorphism test for trivalent graphs may arrive in the not too distant future. Besides the Las Vegas algorithm presented here, results about the behavior of a simple canonical vertex classification algorithm (4.4) combined with a depth-first search may possibly possibly find further applications (Theorem 4.8, Algorithm 4.9).

1.6

Open problems, indicating the limits of (the author's) present knowledge are scattered throughout the paper. Some comments about the way how the presented ideas arose are included with the acknowledgments at the end of the paper.

1.7

Let me add here some comments on the *shortcomings* of the results. The fact that the algorithms are not deterministic is not a defect from either theoretical or practical point of view. Any attempt of practical implementation of the colored graph isomorphism test 3.1-3.2 should, however, be preceded by the use of all available heuristics, since the $O(n^4 \log n)$ running time is too long. It would be very interesting to see improvements on the exponent 4. The most essential deficiency of our algorithm is, however, that it *does not provide a canonical labeling* of the vertices. A canonical labeling of the class

K of graphs is an assignment of a labeled graph $C(X)$ to every graph $X \in K$ such that

(i) $X \cong C(X)$; (ii) $X \cong Y$ iff $C(X) = C(Y)$.

In all cases previously known to me, isomorphism testing algorithms actually yield canonical labeling (see e.g. [11], [2], [12]). R.E. Tarjan has kindly informed me that the celebrated fast planar graph isomorphism test [7], [8], [9] is no exception.

Problem 1.8. Find a (random) polynomial time *canonical labeling* algorithm for vertex-colored graphs (V, E, f) where every color class has size ≤ 3.

(Here, (V, E) is a graph and $f : V \to \{1, \ldots, n\}$ is an arbitrary map called "coloring," such that $|f^{-1}(i)| \leq 3 (i = 1, \ldots, n)$.)

Remark 1.9. The hierarchy of the results of this paper and ref. [6] is this:

$$\begin{array}{lllllll} 2.5 & \to 5.2 \to & 3.1 & \leftrightarrow & 3.2 \to & 3.3 \to & 3.4 \\ \downarrow & & \downarrow & & & & \\ 2.10 \to & [6] & 4.1 & & & & \\ \updownarrow & & & & & & \\ 2.11 & & & & & & \end{array}$$

In view of the technical nature of the proof of both $2.5 \to 5.2$, and $5.2 \to 3.1$, we present a direct proof of $2.5 \to 3.1$ in section 3.

2 The main result

We are going to handle large groups, far too large to be stored by their multiplication tables. The group elements will be thought of as $0-1$ strings, and we require that group operations be carried out by a fast algorithm. For our purposes it is not necessary to postulate that the set of those $0-1$ strings representing group elements be recognizable by a fast algorithm, although this will always be the case in the subsequent applications of the main result. The information we need about our large group is its order,

and an algorithm generating uniformly distributed random members of the group (using random numbers).

For notational convenience, we adopt the convention to identify non-negative integers M with the sets of their predecessors: $M = \{0, \dots, M-1\}$. A map $g : M \to G$ will be called *uniform* if $|g^{-1}(x)| = M/|G|$ for each $x \in G$.

Definition 2.1. We say that a group G is *well described* with respect to the time bounds (t, T) where $t \leq T$ if the following conditions are satisfied:

(a) the order N of G is given;

(b) we are given an algorithm executing group operations on G in less than t steps (in particular, the length of the $0-1$ strings representing the elements of G is less than t, hence $N = |G| < 2^t$);

(c) we are given a multiple M of N, where M is still less than 2^t, and an algorithm computing a uniform map $g : M \to G$ in less than T steps.

Remark 2.2. Condition (c) means that we can compute **uniformly distributed** random members of G, within T steps, employing uniformly distributed random integers from $M = \{0, \dots, M-1\}$.

Definition 2.3. The numbers N, M and the algorithms (b) and (c) constitute a **good description** of the group G w. r. to the time bounds t, T.

Definition 2.4. A chain of subgroups $G_0 \geq G_1 \geq \cdots \geq G_k$ is *recognizable* in time τ, if there is an algorithm with

INPUT: a pair (i, x) where x is known to belong to G_{i-1} $(1 \leq i \leq k)$;

OUTPUT: "yes" if $x \in G_i$, "no" otherwise;

RUNNING TIME: not exceeding τ.

(For $k = 1$, we shall say that G_1 is a subgroup of G_0, recognizable in time τ.)

Now we are able to formulate the main result of this paper.

Let G_0 be a group, well described w. r. to the time bounds t, T and let $G_0 \geq G_1 \geq \cdots \geq G_m = E$ $(|E| = 1)$ be a chain of subgroups, recognizable in time τ. Let $S_i \geq |G_{i-1} : G_i|$ be known upper bounds on the unknown indices of subsequent subgroups $(i = 1, \dots, m)$.

Theorem 2.5. There is a Las Vegas algorithm with

INPUT: a good description of G_0 w.r. to time bounds (t, T); an algorithm, recognizing the chain of subgroups $G_0 \geq G_1 \geq \cdots \geq G_m = E$ in τ steps; the integers $s_1, \ldots, s_m$ (where $s_i \geq |G_{i-1} : G_i|$).

OUTPUT: either "?" or else

(i) R_i, a complete set of left coset representatives of each G_{i-1} mod G_i $(i = 1, \ldots, m)$ $(|R_i| = |G_{i-1} : G_i|)$;

(ii) a good description of each G_i w. r. to the time bounds $(t, T + 2(s_1 + \ldots + s_i)\tau)$ $(i = 1, \ldots, m)$.

COST OF COMPUTATION: less than

$\sum_{i=1}^{m} s_i(\log s_i + \log(4m))(T + 2\tau \sum_{j=1}^{i} s_j) \leq ms(\log s + \log(4m))(T + (m+1)\tau s)$

binary operations and less than $4tms(\log s + \log m + 3)$ coin tosses, where $s = \max\{s_i : 1 \leq i \leq m\}$.

Remark 2.6. Output (i) yields, in particular, the orders of every G_i, and also a set of at most $\sum_{j=i+1}^{m} s_i$ generators of G_i. Output (ii) enables us to generate uniformly distributed random members of G_i.

Our procedure rests on the following two observations. First, we note that a recognizable subgroup H having small index in a well-described group G is itself well described, and all we need in order to construct a good description is a complete set of left coset representatives of G mod H. The second observation will be that we have a good chance of obtaining a complete set of representatives simply by guessing a sufficient number of random members of G and selecting a maximal subset of pairwise left incongruent elements among them (mod H).

Lemma 2.7. Let G be a group, well described w.r. to the time bounds (T, t) and H a subgroup of G, recognizable in time τ. Then H has a good description with time bounds $(t, T + 2\tau|G : H|)$. We can construct the good description once a good description of G and a complete set of left coset representatives of G mod H is given.

Proof. Let $\{a_1, \ldots, a_s\}$ be a complete set of left coset representatives of G mod H. We check Def. 2.1. (a), (b), (c) for H.

(a) The order of H is $|G|/s$. (b) is clearly inherited to H. (c) Let M denote the number occurring in the good description of G, and $g : M \to G$ the corresponding uniform function. We use the same number M for H. First we define a uniform map $h : G \to H$ by

$$h(x) = a_j^{-1}x \quad if \quad x \in a_jH.$$

(For any $x \in G$, $h(x)$ can be computed by computing $a_1^{-1}x, \ldots, a_s^{-1}x$ and determining which of them belongs to H.)

Now, let $\bar{g}(u) = hg(u)$ $(u \in M)$. Clearly, $\bar{g} : M \to H$ is uniform.

Let again G be a group, well described w.r. to the time bounds (t, T); and H a subgroup of G, recognizable in time τ. The index $|G : H|$ is not known to us, only an upper bound $s \geq |G : H|$.

Lemma 2.8. There exists a Monte Carlo algorithm with

INPUT: a good description of the group G w.r. to the time bounds (t, T);

an algorithm, recognizing the subgroup H in time τ;

an integer s which is known to be $\geq |G : H|$;

a positive number q (the parameter of the cost-security tradeoff).

OUTPUT: a positive integer k (believed to be equal to $|G : H|$);

a set $a_1, \ldots, a_k$ of elements of G, pairwise left incongruent mod H.

PROBABILITY OF ERROR: less than e^{-q}.

COST OF COMPUTATION: less than $s(\log s + q)(T + 2s\tau)$ elementary operations and $\lceil s(\log s + q) \rceil$ independent random numbers from the interval $M = \{0, \ldots, M - 1\}$.

($\lceil x \rceil$ denotes the smallest integer, not smaller than x.)

Note that this is a *Monte-Carlo* algorithm, *not* Las Vegas. The possible error is $k < |G : H|$.

Proof: The procedure is very simple. Guess $r = \lceil s(\log s + q) \rceil$ random members $b_1, \ldots, b_r$ of G. Of these, select a maximal subset $a_1, \ldots, a_k$ of pairwise left incongruent elements mod H. Output k and $a_1, \ldots, a_k$. End.

To select the a_i we have to perform at most $(r-1)s$ divisions of the form $b_i^{-1}b_j$ and test whether $b_i^{-1}b_j \in H$ each time.

We estimate the error probability. For a coset bH, the probability that $b_i \in bH$ is $1/|G:H| \geq 1/s$, hence the probability that none of the b_i belongs to bH is at most $(1-1/s)^r < e^{-r/s}$. Finally, the probability that there is a coset not represented by $b_1, \ldots, b_r$ (i.e., $k < |G:H|$) is less than $se^{-r/s} \leq e^{-q}$.

Now we are ready to prove Theorem 2.5. We construct subsets $R_1, R_2, \ldots$ ($R_i \subseteq G_{i-1}$) such that the members of R_i are pairwise left incongruent mod G_i. We pretend that R_i is a complete set of left coset representatives of G_{i-1} mod G_i (i.e. $|R_i| = |G_{i-1}:G_i|$) unless we obtain a proof of the contrary, in which case we output "?" and stop.

Suppose $R_1, \ldots, R_{i-1}$ have already been constructed. If our assumption, that $|R_j| = |G_{j-1}:G|$ holds for $j = 1, \ldots, i-1$, is correct then an $(i-1)$-tuple application of Lemma 2.7 defines a good description of G_{i-1} w.r. to the time bounds $(t, T + 2\tau(s_i + \ldots + si - 1))$. Hence we can apply Lemma 2.8 with $q = \log(4m)$ to obtain a set R_i which is a complete set of left coset representatives of G_{i-1} mod G_i with (conditional) probability exceeding $1 - e^{-q} = 1 - 1/4m$. We may, however, find it impossible to repeatedly apply Lemma 2.7, namely, if for some $j \leq i-1, |R_j| < |G_{j-1} : G_j|$, we may (randomly) generate an $x \in G_{j-1}$, not contained in any coset represented by R_j. If so, we output "?", else continue the procedure until $R_1, \ldots, R_m$ are constructed. Then we compute the product $|R_1| \cdots |R_m|$. If this number is less than $N = |G_0|$, we output "?" (we know there was an error in computing the indices $|G_{j-1} : G_j|$), else we conclude that there was no error, $|R_i| = |G_{i-1} : G_i|$ holds, indeed, for each $i = 1, \ldots, m$, and we output sets R_i and the good descriptions of the G_i.

The probability of failure ("?" output) is less than $m \cdot 1/(4m) = 1/4$, given the $Q = \sum_{i=1}^{m} \lceil s_i(\log s_i + \log(4m)) \rceil$ independent random numbers from M. We describe how to compute these numbers. Let $\ell = \lceil \log(M-1) \rceil$. By $4Q\ell$ coin tosses we define $4Q$ random numbers from the uniform distribution over 2^ℓ. If at least $3Q+1$ of these numbers are $\geq M$, we output "?" and end, else we select the first Q from those, not exceeding $M-1$. The probability of failure at this step is less than

$$\frac{1}{2^{4Q}} \sum_{j=0}^{Q-1} \left(\begin{array}{c} 4Q \\ j \end{array} \right) < \frac{1}{4}$$

.

Finally, the combined probability of failure either in the process of generating random numbers or in determining coset representatives is less than $1/4 + 1/4 = 1/2$.

As $\ell \leq t$, the number of coin tosses required is at most $4tm(s(\log s + \log(4m)) + 1)$. □

As a first application of the main result, we derive a sufficient condition for the existence of a polynomial time Las Vegas isomorphism testing algorithm for a class K of graphs.

In this context, n will always denote the number of vertices of the input graphs, and "polynomial" refers to "bounded by n^c" where the constant c depends on the class K only. A group is well described if it is so w.r. to polynomial time bounds.

Definition 2.9. A class of pairs (n, H) where n is a positive integer and H is a finite group, will be called *polynomially accessible from well described groups*, if there is an algorithm, which computes in n^c time:

(i) a good description of a group G_0;

(ii) a positive integer $k < n^c$;

(iii) recognizes a chain $G_0 \geq G_1 \geq \cdots \geq G_k = H$ of subgroups of G_0, where $|G_i : G_{i+1}| < n^c$ for every i. (The input is (n, H).

Theorem 2.10. Let K be a class of graphs such that the class $\{(n, AutX) : X \in K, |V(X)| = n\}$ of groups is polynomially accessible from well described groups. Then there is a polynomial time Las Vegas algorithm with:

INPUT: a graph $X \in K$;

OUTPUT: either "?", or a set of generators of $AutX$ (in particular, the orbit partition of $V(X)$); and a good description of $AutX$ in polynomial time.

Theorem 2.11. Let K be a class of graphs as in Theorem 2.10. Suppose

moreover that

(i) if $X \in K$ and Y is a connected component of X, then $Y \in K$;

(ii) If X and Y are connected graphs belonging of K then their vertex-disjoint union also belongs to K.

Then there is a polynomial time Las Vegas algorithm with

INPUT: a pair of graphs $X, Y \in K$;

OUTPUT: either "?" or an isomorphism between X and Y, or a proof that they are not isomorphic.

Proof 2.11 is a corollary to 2.10, since it suffices to test connected members X, Y of K for isomorphism. X and Y are isomorphic if one of the generators of $Aut(X \dot{\cup} Y)$ maps X onto Y.

In order to prove 2.10, let $X \in K$ and $G_0 \geq G_1 \geq \cdots \geq G_\ell = AutX$ be the chain of polynomially recognizable subgroups of a well described group G_0. Let $V = \{v_1, \ldots, v_n\}$ be the vertex set of X, and let $G_{\ell+i}$ denote the pointwise stabilizer of $\{v_1, \ldots, v_i\}$ in $AutX$. The chain $G_\ell \geq \cdots \geq G_{\ell+n} = E$ is clearly recognizable in linear time, hence, setting $m = n + \ell$, we may apply Theorem 2.5 to the chain $G_0 \geq \cdots \geq G_m = E$ to obtain the desired output within polynomial time, with $< 1/2$ probability of failure (but without any possibility of error). □

Remark 2.12. In Theorems 2.10 and 2.11, the group G_0 is not necessarily a subgroup of the symmetric group S_n. In one of the principal applications of these results [6] G_0 will be a direct product of groups of matrices, acting essentially on the eigen-subspaces of the adjacency matrix of X.

Remark 2.13. Note that, under the conditions of 2.10, we are able to generate in polynomial time uniformly distributed *random automorphisms* of X.

Remark 2.14. It may happen that we are unable to build the required tower of groups on top of $AutX$, but we can build such a tower on top of the stabilizer of some suitably chosen small subset of the vertex set. This approach will be used in Section 4.

3 Colored graphs with bounded multiplicities of colors, and applications to posets and distributive lattices.

We are going to apply the main result in a frequently occurring situation. We consider the isomorphism problem for vertex-colored digraphs, where the size of each color-class is bounded by a number k. (Isomorphisms preserve colors by definition.)

By *coloring* we mean an arbitrary function $f : V \to n$, not necessarily a good coloring in the sense of chromatic graph theory.

Theorem 3.1. There is a Las Vegas algorithm with:

INPUT: A vertex-colored digraph X;

OUTPUT: either a set of generators of $AutX$, or "?."

COST OF COMPUTATION: $O(k_{2k}n^4 \log n)$ operations and $O(k^k n^3 \log n)$ coin tosses,

where n is the number of vertices and k denotes the size of the largest color-class. (The O notation refers to absolute constraints, not depending on either k or n.)

Proof. Let p denote the number of color-classes and $V_1, \ldots, V_p$ the classes themselves. Let $W_i j$ denote the subgraph induced by $V_i \cup V_j (1 \leq i \leq j \leq p$. Let X_0 denote the empty graph on the colored vertex-set $V = V(X)$, the color-classes being $V_1, \ldots, V_p$. We define an increasing sequence of subgraphs of $X : X_0 \subseteq X_1 \subseteq \cdots \subseteq X_q = X$ where $q = \binom{p}{2}$, by setting $X_1 = X_0 \cup W_{12}$, $X_2 = X_1 \cup W_{13}, \ldots, X_{p-1} = X_{p-2} \cup W_{1p}$, $X_p = X_{p-1} \cup W_{23}, \ldots, X_q = X_{q-1} \cup W_{p-1,p}$. We view each of these digraphs as colored digraphs with the same coloration as in X. Further, we define the colored digraphs $X_{q+1}, \ldots, X_{q+p}$ by subsequent refinement of the color-classes to singletons. The edge set of $X_{q+1}, \ldots, X_{q+p}$ is the same as the edge set of $X = X_q$. X_{q+1} differs from X_{q+i-1} only in that the color-class V_i of X_{q+i-1} is split (by introducing new colors) into singletons $(i = 1, \ldots, p)$. Hence in X_{q+p} all vertices have different colors and so $Aut(X_{q+p}) = E(|E| = 1)$.

Set $m = 2q + p$ and $G_{2i} = AutX_i (i = 0, \ldots, q)$. For $X_i = X_{i-1} \cup W_{j\ell}$, let G_{2i-1} consist of those elements of G_{2i-2} whose restriction to $V - V_j$ coincides with the restriction of some member of G_{2i} to $V - V_j (i = 1, \ldots, q)$. Set $G_{2q+1} = AutX_{q+i}$ for $i = 0, \ldots, p$. Observe that $G_0 \geq G_1 \geq \cdots \geq G_m = E$.

Let $|V_i| = k_i (i = 1, \ldots, p)(k_i \leq k), \sum_{i=1}^{p} k_i = n$. The restriction of $AutX_0$ to V_j has order $k_j!$, hence

$$|G_{i-1} : G_i| \leq k! (i = 1, \ldots, 2q).$$

Further, clearly $|G_{2q+i-1} : G_{2q+i}| \leq k_i! (i = 1, \ldots, p)$, G_{2q+i} being the stabilizer of V_i in G_{2q+i-1}.

We conclude that $|G_{i-1} : G_i| \leq k! (i = 1, \ldots, m)$.

Suppose now that $x \sum G_{i-1}$. In order to decide whether $x \in G_i$ if $i \leq 2q$ we only have to check whether the restriction of x to a pair of color-classes is an automorphism of the subgraph, induced by these color-classes if i is even; and whether its restriction to one of these color-classes is the restriction of of an automorphism of such as induced subgraph on two classes if i is odd. To this end, we only have to compute the images of fewer than $4k^2$ edges. If $i > q$, all we have to check is whether V_{i-q} is pointwise fixed under x. Hence the chain $G_0 \geq G_1 \geq \cdots \geq G_m$ is recognizable in $\tau = O(k^2)$ time.

Finally, $G_0 = AutX_0 = S_{k_1} \times \ldots \times S_{k_p}$ (direct product of symmetric groups) is well described w.r. to the time bounds (t, T) where $t = \tau = O(n \log k$. Clearly, we may choose $M = N = |G_0| = \prod_{i=1}^{p} (k_i!)$ for the good description of G_0.

We have now all we need in order to apply Theorem 2.5. Our output will be either "?," or $R_{2q+1} \cup \ldots \cup R_m$, a generating set of $G_{2q} = AutX$. The probability of failure is less than $1/2$ by 2.5.

Let $s = k!$. Clearly, $m = 2q + p < n^2$.

The cost of computation is less than

$ms(\log s + \log (4m))(T + (m + 1)\tau s) =$

$= O(n^2 k! (k \log k + \log n)(n \log k + n^2 k^2 k!)) =$

$O(n^4 \log n (k + 2)!^2)$ operations and less than

$4tms(\log s + \log m + 3) = O(n \log k \cdot n^2 k! \cdot (k \log k + \log n)) =$

$= O(n^3 \log n(k+2)!)$ coin tosses. □

Corollary 3.2. There is a Las Vegas algorithm with

INPUT: vertex-colored digraphs X and Y;

OUTPUT: either "?," or an isomorphism between X and Y, or a proof that they are not isomorphic.

COST OF COMPUTATION: $O(k^{4k} n^4 \log n)$ operations and $O(k^{2k} n^3) \log n$ coin tosses,

where n is the number of vertices of X and k is the size of its largest color-class.

Proof. We may assume X and Y are connected and both have the same set of colors, each color occurring the same number of times in the two graphs. We apply Theorem 3.1. to the disjoint union $X \cup Y$. Clearly, X and Y are isomorphic if and only if at least one of the generators of $Aut(X \cup Y)$ interchanges X and Y, thus providing an isomorphism. The proof of non-isomorphism consists of displaying coset representatives for the subsequent subgroups in the chain, constructed in the proof of Theorem 3.1, and proving that each of them is a complete set of coset representatives (by multiplying their orders), and finally observing that none of the elements thus proven to generate $Aut(X \cup Y)$ interchange X and Y. □

We have an immediate application to partially ordered sets (posets). The *width* of a poset is the size of its largest antichain.

Corollary 3.3. There is a Las Vegas algorithm with

INPUT: posets X and Y;

OUTPUT and COST OF COMPUTATION: as in Corollary 3.2, with n being the number of elements of X, and k standing for the width of X.

Proof. The partial orders can be viewed as digraphs. Let us assign color i to a vertex x of X or Y if i is the length of the longest chain below x. Clearly, the color classes are antichains, hence their size does not exceed k. This coloration being invariant under isomorphisms, the result is immediate from 3.2. □

The isomorphism problem for lattices is isomorphism complete (i.e., polynomial time equivalent to graph isomorphism) (FRUCHT [4]). This is, however, very unlikely to be the case for distributive lattices, in view of the following result.

Corollary 3.4. There is a Las Vegas algorithm with

INPUT: distributive lattices X, Y (given by their operation tables);

OUTPUT: either "?," or an isomorphism between X and Y, or a decision that they are not isomorphic;

COST OF COMPUTATION: $O(n^{6 \log\log n})$ operations and coin tosses. ($n = |X| = |Y|$).

Proof. Every finite distributive lattice X is isomorphic to the lattice of ideals of the poset $J(X)$ of its join-irreducible elements. (Birkhoff, see [5, p. 61]). X and Y are isomorphic if and only if $J(X)$ and $J(Y)$ are isomorphic. Therefore we find the posets $J(X)$ and $J(Y)$ and apply Cor. 3.3. We have to estimate k, the width of $J(X)$. If $J(X)$ contains an antichain $A \subseteq J(X)$, then A generates a Boolean algebra on $2^{|A|}$ elements in X. Hence, $2^k \leq |X|$. Also, obviously $|J(X)| < |X|$, hence our estimate on the cost of the computation follows. □

Remark 3.5. There is an $n^{c \log\log n}$ isomorphism testing for projective planes, hence for modular lattices of length 3 (Gary L. Miller [12]). Can this be generalized to all modular lattices? Or, perhaps, modular lattices are isomorphism complete?

4 Trivalent graphs and graphs with bounded valences.

Gary L. Miller has pointed out to the author that isomorphism of trivalent graphs can be done in c^n time (as opposed to $n!$ for general graphs), simply by selecting an arbitrary cyclic order of the edges at each vertex, thus defining an orientable map, and testing the resulting maps for isomorphism. (The same argument works for graphs with bounded valences.)

We don't know any deterministic algorithm whose worst case behavior on trivalent graphs would be better than c^n for every $c > 1$. We are, however, able to replace the exponent n by essentially $\sqrt{n}$, using our Las Vegas algorithm.

Theorem 4.1. There is a Las Vegas algorithm with

INPUT: graphs X and Y with maximum valence 3;

OUTPUT: either "?," or an isomorphism of X and Y, or a decision that they are not isomorphic;

COST OF COMPUTATION: less than $\exp((4 + o(1))\sqrt{n}\log n)$ operations and coin tosses.

More generally, for maximum valence $d \geq 3$ the cost of our computation will be less than $\exp((4 + o(1))\sqrt{n}(\log n)^{1+\pi(d-1))/2)}$ where $\pi(x)$ denotes the number of primes not exceeding x. ($\pi(2) = 1, \pi(3) = \pi(4) = 2, \pi(5) = \pi(6) = 3$.)

Remark 4.2. For bounded d this is an $\exp(n^{1/2+o(1)})$ cost. For $d < (1 - 2c)\log n$ (c a positive constant), our cost is $\exp(n1 - c + o(1))$, still better than the previously known $\exp(n^{1+o(1)})$ bounds.

The next problem to be solved in this area is to extend the range of possible maximum degrees.

Problem 4.3. Find a positive constant c such that if X and Y are non-isomorphic graphs on n vertices with maximum valence less than n^c then their non-isomorphism has a proof, shorter than $\exp(n^{1-c})$.

Procedure 4.4. The proof of 4.1 will use a well-known naive *canonical vertex classification* which we have to describe here (cf. [15]).

Let $X = (V, E)$ be a graph with colored vertices, the function $f : v \to n$ being the coloration. We suppose that the set of colors actually occurring forms an initial segment of $n = \{0, 1, \ldots, n-1\}$. We refine the color classes as follows. For $v \in V$, let $k_i(v)$ denote the number of those neighbors of v having color i. Let us assign to v the vector $g(v) = (f(v), k_1(v), \ldots, k_n(v))$. Let us arrange these vectors lexicographically, say $w_1 < \cdots < w_r$ if there are r different ones among them. We define the refined coloring by $f'(v) = j$ if $g(v) = w_j$. Clearly $f(v_1) < f(v_2)$ implies $f'(v_1) < f'(v_2)$ but not conversely.

$f'(v_1) = f'(v_2)$ if $g(v_1) = g(v_2)$.

Let now $f_0 = f, f_1 = f', \ldots, f_{i+1} = f'_i$. Clearly, we have $f_{i_0} = f_{i_0+1} = \ldots = f_n = \ldots$ for some $i_0 \leq n$. (There are not more than $n-1$ proper refinements possible.) Let us call f_n the *stable refinement* of f and denote it by $\bar{f}$. Let us call a coloration f *stable* if $f = f'$ (hence $f = \bar{f}$). Clearly, the stable refinement of any coloration is stable.

By a *semiregular bipartite graph* we mean a bipartite graph with color partition $V = V_1 \cup V_2$ such that all vertices in one class have the same valence. For V_1, v_2 disjoint subsets of the vertex set of X, the bipartite subgraph induced by (V_1, V_2) means the bipartite graph on $V_1 \cup V_2$ whose edge set consists of those edges of X connecting a vertex of V_1 to a vertex of V_2. We denote this bipartite graph by $X(V_1, V_2)$. Also, the subgraph induced by V_1 will be denoted by $X(V_1)$. The following is straightforward.

Proposition 4.5. The coloration f of the graph X is stable iff all induced subgraphs $X(f^{-1}(i))$ are regular and all induced bipartite subgraphs $X(f^{-1}(i), f^{-1}(j))$ are semiregular $(i, j < n, i \neq j)$.

We shall need to change f by assigning a new color to a vertex v, unless its color-class was a singleton.

Definition 4.6. For a coloration f and $v \in V$ let the *v-pointed recoloration* f^0_v be defined by

$f^0_v(x) = f(x)$ if $x \neq v$,

$f^0_v(v) = min(n \ \{f(x) : x \neq v\})$.

Observe that $f^0_v = f \ iff \ f^{-1}(f(v)) = \{v\}$.

The f^0_v*-color-class* of v is necessarily a singleton. Let f_v denote the stable refinement of f^0_v. We call f_v the *stabilizer* of v in f. For $S = (v_1, \ldots, v_s)$ an ordered *s-tuple* of vertices, the stabilizer of S in f will be $f_S = f_{v_1 \ldots v_s}$, obtained by repeated application of the operation above. (The terms are borrowed from the theory of permutation groups. The classes of a stable coloring imitate some properties of the orbits of the automorphism group of a (colored) graph. We prove one of these analogies below, Lemma 4.8.).

The following is straightforward.

Proposition 4.7. Let f be a stable coloration of X such that the color-class of $x \in V$ is a singleton. Then for any vertices $y, z \in V$, $dist(x, y) \neq dist(x, z)$ implies $f(y) \neq f(z)$. □

Now we prove that if X is connected and a vertex of valence less than the maximum has a singleton for its color-class then all prime divisors of the lengths of the color-classes are less than the maximum valence.

For $m \geq 2$, let $pr(m)$ denote the *largest prime divisor* of m. Set $pr(1) = 1, pr(0) = 0$.

Now we give a bound on the prime divisors of the lengths of the color-classes of f_x for a connected graph X with bounded valences. Note that under the same conditions, the same bounds are easily seen to be valid for the prime divisors of the order of the stabilizer subgroup of the automorphism group and consequently for the orbit lengths of the stabilizer, cf. [3, Theorem 1].

Lemma 4.8. Suppose that all vertices of the connected graph X have valences $\leq d$. Let f be a stable coloration of X such that the color-class of a vertex x of valence $\leq d-1$ is a singleton (i.e. $f_x = f$). Then $pr(f-1(i)) \leq d-1$ for every $i < n$ (i.e. the lengths of the color-classes have no prime divisor exceeding $d - 1$).

Proof. Let $z \in V$. We prove by induction on the distance $dist(x, z)$ that the color-class of z has no prime divisors exceeding $d - 1$. This is true if $z = x$. Suppose it holds for all distances less than $dist(x, z) = k > 1$. Let $y \in V$ be a neighbor of z at distance $k - 1$ from x. Let A and B denote the color-classes of y and z, resp. ($A \neq B$ by 4.7.) $pr(|A|) \leq d - 1$ by the induction hypothesis. Let the vertices from A and B have valence a and b, resp., in the bipartite graph $X(A, B)$. Clearly, $a, b \leq d$. Moreover, $a \leq d-1$ since if $k \geq 2$ then y has a neighbor at distance $k - 2$ from x; if $k = 1$ then $y = x$ and $|B| \leq d - 1$ (since now B is a subset of the neighbors of x).

Clearly, $a|A| = b|B|$, $ab \neq 0$,

hence $pr(|B|) \leq \max(a, pr(|A|)) \leq d - 1$. □

Now we are able to describe our Las Vegas algorithm to test isomorphism of graphs with maximum valence d. We may supppose both X and Y are connected. (It suffices to test components.)

The idea is that we shall be able to reach a coloration with color-classes of moderate size by stabilizing an initial trivial coloration at a sequence of vertices, the sequence having moderate length. We do this in all possible ways and decide whether the obtained colored graphs are isomorphic using the Las Vegas algorithm of Cor. 3.2. "Moderate" here means $\sqrt{n}(\log n)^c, c$ constant. By "guess" we shall mean an arbitrary choice.

Algorithm 4.9. Choose a positive integer k. This will be our desired upper bound on the color multiplicities and we shall compute its optimum value at the end of the proof (before Remark 4.13). Guess an edge of X, and halve it by inserting a new vertex x_0 of valence 2. Denote the obtained graph on $n+1$ vertices by X'. Define a coloring g of X' by $g(x_0) = 0$ and $g(x) = 1$ for $x \in V$. Let f be the stable refinement of g. Clearly $f = f_{x_0}$. If there is a color-class of size exceeding k, let i be the smallest number such that the color-class $f^{-1}(i)$ has largest size. Guess a vertex x_1 from $f^{-1}(i)$. Compute the stable coloration $f_{x_0x_1}$ and guess a vertex x_2 from the first color-class of largest size if it is larger than k, etc., until we arrive at a sequence $S = (x_0, x_1, \ldots, x_s)$ of vertices such that the length of each color-class of f_s is at most k.

Let us execute the same operations on Y. There are $|E(Y)| \leq nd/2$ ways of guessing the edge to be halved by y_0, and we have at most n^s ways of guessing the sequence $y_1, \ldots, y_s$. Clearly, X and Y are isomorphic if there is a correct guess, i.e. for at least one of these sequences of arbitrary choice, the obtained colored graphs are isomorphic (colors are preserved under isomorphism by definition). Hence we may apply the Las Vegas algorithm of Cor. 3.2 to test whether the colored graph (X', f_s) is isomorphic to any of the at most $n_snd/2$ augmented and colored versions of Y. In each case when the Las Vegas algorithm fails (outputs "?"), we repeat it until decision is reached, but the total number of calls on the Las Vegas algorithm should not exceed $n^{s+1}d$. If this number of calls is insufficient (failure occurs in more than half of the cases), we output "?." Clearly, the probability that this happens is less than $1/2$.

The cost of the computation is $n^{s+1}d/2$ applications of the stable refinement procedure which costs (n^2d) each time, and an $n^{s+1}d$-fold application of the Las Vegas isomorphism test for colored graphs with at most k-fold colors. Hence the total number of operations required is

$$O(k^{4k}n^{s+5}d\log n) < O(k^{4k}n^{s+6}).$$

Clearly, there is a $k-s$ tradeoff here that we have to analyze.

Let

$$\delta(n,d) = \max_{2h\le n} |\{x : h \le x < 2h, prx < d\}|.$$

Clearly $\delta(n,3) = 1$. The following can be proved easily by induction on $d \ge 3$.

Proposition 4.10. $\delta(n,d) \le \prod \left(\frac{\log n}{\log p} + 1\right)$ where the product is taken over all primes $p, 3 \le p \le d-1$. □

Corollary 4.11. $\delta(n,d) \le \left(\frac{\log n \log 3}{+} 1\right)^{\pi(d-1)-1}$ where $\pi(x)$ denotes the number of primes, not exceeding x. □

Lemma 4.12. The number of points x_i we have to guess for x_i-pointed recoloration in order to achieve that each color-class has length $\le k$ in the above algorithm can be estimated by $s < 2\delta(n,d)n/k$.

Proof. Let A_0 be one of the color-classes under a stable coloration of X' with $f^{-1}(x_0) = \{x_0\}$.

Suppose $|A_0| > k$.

We wish to estimate how many of the x_i had to be selected from A_0 for x_i-pointed recoloration until we obtained a stable coloration such that each subclass of A_0 had size $\le |A_0|/2$. Let $z_1, \ldots, z_r$ be these x_i. Let A_i be the (unique) largest subclass of A_{i-1} after the stable refinement of the z_i-pointed recoloration ($i = 1, \ldots, r-1$). This class is unique since $|A_i| > |A_0|/2 \ge |i-1|/2$. It is a proper subset of A_{i-1} since $z_i \in A_{i-1}$ (z_i was always selected from a largest color-class) and $z_i \ni A_i$ (since the color-class of z_i became a singleton at this step). Hence $|A_0| > |A_1| > \cdots > |A_{r-1} > |A_0|/2$.

By Lemma 4.8, $pr(|A_i|) \le d-1$, hence $r \le \delta(n,d)$.

We conclude that it took less than $\delta(n,d)n/\ell$ pointed recolorations to reduce the size of the largest color-class from $\le 2\ell$ to $\le \ell$. (Namely, there are less than n/ℓ classes of size greater than ℓ.) Let us apply this argument to $\ell = k, 2k, 4k, \ldots, 2^m k$ where $2^{m-1}k \le n < 2^m k$. We obtain

that the total number of pointed recolorations occurring in the algorithm is $s < \delta(n,d)\left(\frac{n}{k} + \frac{n}{2k} + \ldots + \frac{n}{2^m k}\right) < \delta(n,d)\frac{2n}{k}$. □

End of the proof of Theorem 4.1.

Now we are kin the position to make our optimal choice of the parameter k. Let $k = \lceil(\delta(n,d)n)^{1/2}\rceil$. Then by 4.12 $s < 2(\delta(n,d)n)^{1/2}$ and the logarithm of the cost of the computation is estimated by

$$\log\left(O(k^{4k}n^{s+6})\right) = o(1) + 4k\log k + (s+6)\log n <$$

$$< \ (4\log n + 2\log\delta(n,d) + o(1))(\delta(n,d)n)^{1/2} <$$

$$< \ (4 + o(1))n^{1/2}(\log n)^{(1+\pi(d-1))1/2}$$

(using 4.11). □

Remark 4.13. The $o(1)$ notation refers to a quantity tending to 0 with $n \to \infty$ uniformly in d. Note that for $d > \log n$ our result doesn't make sense: even brute force is faster.

Remark 4.14. The guesses (for pointed recoloration) we made in the algorithm are an example of the *depth-first* search. The stable refinement procedure is a *breadth-first* search. The global approach of our Las Vegas algorithm using a tower of supergroups of $AutX$ could, however, be hardly classified by these terms.

Remark 4.16. It is the author's hope that a deeper understanding of the stabilizer of an edge in the automorphism group of trivalent graphs will result in a further great improvement on the complexity given in 4.1, and hopefully $\exp(c\log^2 n)$ could be reached. (For the class of arc-transitive trivalent graphs, this goal has been achieved by R. Lipton [11].) A nice intermediate problem, providing a good starting point for research, has recently been formulated by Maria Klawe.

Problem 4.17. (Maria Klawe [10]) By a marked graph we mean a triple $X = (V, E, R)$ where (V, E) is a graph and R is an equivalence relation. Estimate the complexity of isomorphism testing for marked binary trees. (Isomorphism preserves both relations E and R.)

It was this problem which led the author to Theorem 4.1. It is worth noting that the automorphism group of a binary tree (V, E) is the iterated

wreath product of cyclic groups of order two. It is well-described in the sense of Def. 2.1. It is a 2-group, hence there is a chain of subgroups

$$Aut(V,E) = G_0 \geq G_1 \geq \cdots \geq G_r = AutX$$

where $|G_i : G_{i+1}| \leq 2$. The problem is, however, to find such a chain, *recognizable* in polynomial (or $\exp(\log^2 n)$) time; then 2.10 could be applied.

In contrast to 4.17, Maria Klawe observes:

Proposition 4.18. (Maria Klawe [10]) Marked trees are isomorphism complete (i.e. isomorphism of marked trees is as hard as graph isomorphism).

Proof. Let $X = (V,E)$ be a graph on n vertices. Let (W,F) denote the tree defined by

$$W = \{w_0\} \cup V \cup \{(v,z) : (v,z) \in E\};$$

$$F = \{[w_0, v] : v \in V\} \cup \{[v,(v,z)] : v,z \in V, [v,z] \in E\}.$$

(The vertex w_0 is adjacent to n vertices. All points at distance two from w_0 have valence one. Two such points correspond to every edge of x.)

Define R on W by

$$R = \{(x,x) : x \in w\} \cup \{((v,z),(z,v)) : [v,z] \in E\}.$$

(Each equivalence class is either a singleton or a pair. The edges of X are in one-to-one correspondence with the 2-element classes of R.)

Let $T(X) = (W,F,R)$ denote the obtained marked tree. Clearly, $X \cong Y$ if $T(X) \cong T(Y)$, hence graph isomorphism reduces in linear time to marked tree isomorphism. □

4.19. I propose some terminology for often occurring complexity classes. The terms *exponential* and *subexponential* should be defined to be invariant under polynomial equivalence (substituting n^c for $n, c > 0$). Hence $f(n)$ grows *exponentially* if $\exp(n^c) < f(n) < \exp(n^d)(0 < c < d)$ for n large enough, and *subexponential* should mean $\exp(o(n))$. Important classes of subexponential functions are the *logonomial* functions of degree c, meaning $f(n) = \exp((\log n)^{c+o(1)})(c > 1)$. (This class is invariant under polynomial equivalence for every particular value of c. (Etymology: $\log f(n)$ is a polynomial of $\log n$.) $n^{\log\log n}$ is *sublogonomial* (i.e. $\exp((\log n)^{1+o(1)})$). It is worth in-

troducing such subclasses of the exponential functions that are invariant under linear substitutions only. $\exp(n^{2/3+o(1)})$ might be called 2/3-*exponential*. Theorem 4.1 says that isomorphism of graphs with bounded valence is at most *half-exponential* ($\exp(n^{1/2+o(1)})$), and the same holds for strongly regular graphs by [2]. Functions satisfying $n^c < f(n) < n^d, 0 < c < d < 1$ (for n large enough) might be called *frexponential* (the exponent of the exponent is a proper fraction). (R.L. Graham warns me; this term will never catch on.)

4.20. Using this terminology, the fundamental goals of the theory of isomorphism testing for the not too distant future are, in my opinion, the following.

PROBLEM A. Find a frexponential isomorphism testing algorithm for all graphs.

PROBLEM B. Find a logonomial isomorphism testing algorithm for trivalent graphs.

5 Intersection of cylindric subgroups of a direct product.

In this section we generalize the situation of Theorem 3.1. Theorem 5.2 is the result to be applied in [6] to find a polynomial time Las Vegas isomorphism testing algorithm for graphs whose adjacency matrix has bounded eigenvalue multiplicities.

Definition 5.1. Let $H_0, \ldots, H_{r-1}$ be groups and $J \subseteq r$ a subset of the index set. Let B be a subgroup of $\prod_{j\in J} H_j$. Let $D = \prod_{j<r} H_j$ be the direct product of all the H_j and $C = B \times \prod_{j\ni J} H_j$ a subgroup of D. We call C a *cylinder* with *base* B and *base index set* J.

We shall be interested in determining a set of generators of the intersection of a family cylinders given the factors H_j by their multiplications tables and the base groups by the set of their elements. (Group operations in B are defined by those in the H_j.)

Theorem 5.2. Let $H_0, \ldots, H_{r-1}$ be groups, $J_0, \ldots, J_{s-1}$ subsets of the index set r, and $B_i \leq \prod_{j\in J_i} H_j$ base subgroups for the cylinders $C_i =$

$B_i \times \prod j \ni J_i H_j \leq D$ where $D = \prod_{j<r} H_j$. Let $A = \cap_{i<s} C_i$.

There exists a Las Vegas algorithm with

INPUT: the integers r, s, the multiplication tables of the group $H_j (j < r)$, and the sets $B_i (i < s)$.

OUTPUT: Either "?," or a set of generators of A (the intersection of the cylinders) and a good description of A w.r. to the time bounds $O(s(\log s + \log h)), O(Ksh^2 \log h))$ where $h = \max\{|H_j|, |B_i| : j < s, i < r\}$ and $K = \sum_{i<r} |J_i|$.

COST OF COMPUTATION:

$$O(K^2 sh^3 \log h(\log h + \log K))$$

binary operations and

$$O(Ksh(\log^2 h + \log K(\log s + \log h)))$$

coin flips.

Remark 5.3. Compare these numbers with the $n = O((shn^2 + K) \log h)$ length of the input string.

We obtain that the algorithm runs within $O(jn^{7/2} \log n)$ steps. For bounded h, it is $O(n^3 \log n)$.

5.4. It is instructive to see how the situation of Theorem 3.1 fits in this scheme.

Let X be a vertex-colored digraph, the color-classes having size $k_j (j < r) (\sum_{j<r} k_j = n)$. Let H_j be the symmetric group S_{k_j} acting on the j th color-class. For $i < \ell < r$ let $J_{i\ell} = \{i, \ell\}$ and $B_{i\ell} = AutW_{i\ell}$ where $W_{i\ell}$ is the subgraph induced by the i^{th} and ℓ^{th} color-classes. Clearly, $AutX = \cap_{i<\ell<r} C_{i\ell}$ where $C_{i\ell}$ is the cylinder with base $B_{i\ell}$.

In what follows we generalize the procedure given in the proof of 3.1 to reduce 5.2 to the main result (2.5).

Proof of Theorem 5.2.

For $J \subseteq r$, set $H_J = \prod_{j \in J} H_j$.

For $I \subseteq J$ let $pr_J^I : H_J \to H_I$ denote the projection map.

For each $i < s$ choose a strictly descending chain of subsets

$J_i = J_i^0 \supset \cdots \supset J_i^{\ell i} = \emptyset$ where $\ell_i = |J_i|$

hence $J_i^P - J_i^{p+1} = \{j_i^p\}$ is a singleton for each $p < \ell_i$.

Set $B_i^p = pr_{J_i}^{Jp_i} B_i$ and let C_i^p be the cylinder with base B_i^p, i.e.,

$C_i^p = B_i^p \times H_{r-J_i^p}$.

Clearly, $C_i = C_i^0 \leq C_i^1 \leq \cdots \leq C_i^{\ell i} = D$,

and $|C_i^{p+1} : C_i^p| \leq |H_{j_i} p| \leq h$.

Moreover, $A = \cap\{C_i^j : i < s, j < \ell_i\}$.

Set $\sum_{i<q} \ell_i = m_q, q = 0, \ldots, s (m_0 = 0)$.

Define the chain $G_0 \geq G_1 \geq \cdots \geq G_{m_s} = A$ as follows. Let $G_0 = D$.

Suppose $m_q < t \leq m_{q+1}$ for some $q < s$ and G_{t-1} has already been defined. Let $t = m_q + (\ell_q - p)$ (clearly $0 \leq p < \ell_q$) and set

$G_t = G_{t-1} \cap C_q^p$.

Hence G_{m_s} is the intersection of all the cylinders C_q^p, implying $G_{m_s} = A$. It is easy to verify that $|G_{t-1} : G_t| \leq h$.

This follows from the fact that if A, B, C are arbitrary subgroups of a group and $B \leq A$ than $|A : B| \geq |A \cap C : B \cap C|$.

We want to estimate the complexity of recognizing the chain $G_0 \geq \cdots \geq G_{m_s}$. Given $x \in G_{t-1}$, we have to decide whether $x \in G_t$, i.e. whether $x \in C_q^p$. This is decided by searching through the list of elements of B_q^p which is obtained by deleting some columns of the array representing B_q and deciding whether $pr^{J_q^u} x$ occurs there, where $u = \ell_q - p$.

This takes at most $|B_q||J_q| \leq hs$ decisions of the type $y = y'$ for some $y, y' \in H_j$ for some $j < s$. Assuming the H_j are properly encoded, the cost of such a decision should be $O(\log |H_j|)$.

We conclude that the chain $G_0 \geq \cdots \geq G_{m_s}$ is recognizable in $\tau = O(sh \log h)$ steps.

In order to be able to apply 2.5, we have to continue our chain beyond A. Let E_j denote the unit subgroup of H_j and set

$K_d = \prod_{j<d} E_j \times \prod_{d \leq j<s} H_j \leq D$.

$(d = 0, \ldots, s)(K_0 = D, |K_d| = 1)$. Clearly $|K_d : K_{d+1}| = |H_d| \leq h$.

Set $G_{m_s+d} = G_{m_s} \cap K_d$, and $m_s + s = m$.

This way we get the chain $G_0 \geq G_1 \geq \cdots \geq G_m, |G_m| = 1$,

$|G_{i-1} : G_i| \leq h$ for $i = 1, \ldots, m$, and the chain is recognizable in $O(sh \log h)$ steps. We are almost ready to apply 2.5. We only have to check the time bounds of the good description of $G_0 = D$. The order of D is $\prod_{j<s} |H_j| \leq h^s$. With proper encoding, group operations in H_j are executed in $O(\log h)$ hence in D group operations cost $O(s \log(hs))$. A uniform map $|D| \to D$ is computable by subsequent divisions with remainder by the orders of the H_j. The cost of division of a number $\leq h^s$ by another not exceeding h is $O(s \log^2 h)$, hence the uniform map $|D| \to D$ will be computed in $O(s^2 \log^2 h)$ steps. All this amounts to a good description of D w.r. to the time bounds $(O(s \log(sh)), O(s^2 \log^2 h))$. (This corresponds to (t, T) in 2.5. Note that our h is denoted by s in 2.5.)

Now 2.5 says that the required output can be obtained at the cost of

$mh(\log h + \log(4m)(O(s^2 \log^2 h + (m+1)sh^2 \log h) = O(K^2 h^3 s \log h(\log h + \log K))$

elementary operations (since $m < 2m_s = K$), and

$O(mhs \log(sh)(\log h + \log m)) =$

$= O(Khs(\log^2 h + \log K(\log h + \log s)))$ coin flips.

The good description of A within the time bounds claimed follows similarly. □

Remark 5.5. The cylinder intersection problem for cosets is the following: for each $B_i \leq H_{J_i}$ select a $b_i \in H_{J_i}$ and take the coset $b_i B_i$ for the base of a cylinder $C_i' = b_i B_i \times \prod_{j \ni J_i H_j}$. Now the problem is to decide whether

$\cap_{i<s} C_i' = \emptyset$. This problem is easily seen to be equivalent to the cylinder intersection problem for groups, solved in polynomial Las Vegas time by 5.2. We remark that if all groups H_i are abelian, there is a deterministic polynomial time solution [6].

The *cylinder intersection problem* can also be formulated *for arbitrary sets* H_i rather than for groups. Let $H_i (i < r)$ be finite sets, $J_0, \ldots, J_{s-1} \subseteq r$ subsets of the index set and $B_i \subseteq \prod_{j \in J_i} H_j$ base subsets for the cylinders $C_i = B_i \times \prod_{j \notin J_i} H_j$. The problem is to decide whether $\cap_{i<s} C_i = \emptyset$.

The following observation of Gary L. Miller caused some headaches to the author.

Proposition 5.6. (Gary L. Miller [13]) The cylinder intersection problem is NP-complete, even in the particular case $|H_0| = \ldots = |H_{r-1}| = 3$ and $|J_0| = \ldots = |J_{s-1}| = 2$.

Proof. We reduce 3-colorability of graphs to the cylinder intersection problem.

Let $X = (r, E)$ be a graph on the vertex set $r = \{0, \ldots, r-1\}$. Let H_j be the 3-set $3 \times \{j\} (j < r)$. The elements of $D = \prod_{j<r} H_j$ are easily identified with the 3-colorations of the vertex set, regardless of adjacency. Let $E = \{e_0, \ldots, e_{s-1}\}$ and set $J_i = \{p, q\}$ if $e_i = [p, q] (i < s)$. Define the base set B_i by

$$B_i = \{((g, p), (h, q)) : g \neq h, g, h < 3\}.$$

So, $|B_i| = 6$.

Clearly the cylinder $C_i = B_i \times \prod_{j \neq p,q} H_j$

consists of those colorations of the vertex set respecting the adjacency of p and q, assigning different colors to them. This implies that the intersection $A = \cap_{i<s} C_i$ consists precisely of all good colorations of X, hence X is 3-colorable iff $A \neq \emptyset$. $\square$

6 Acknowledgements.

Announced erroneous results, mathematical discussions in various corners of the world, and three weeks of despair led to the results of this paper.

The first thoughts about the subject came to my mind after a half day's conversation with D. Yu. Grigoriev in Leningrad in November 1978 about a polynomial time isomorphism testing algorithm for graphs with bounded multiplicities of eigenvalues [6]. Shortly afterwards in Budapest I realized that such an algorithm would follow from a polynomial time solution of the cylinder intersection problem for groups, and I (mistakenly) believed I had found a deterministic algorithm for that. Still I don't know if such an algorithm exists.

At the 1979 SIGACT conference (early May in Atlanta) a polynomial time isomorphism testing for distributive lattices was announced, but the proof was incorrect and this problem is still open. At the end of May we discussed this problem with Gary L. Miller at MIT. He conjectured that $n^{c \log \log n}$ could be achieved without much difficulty. I replied that this follows from my cylinder intersection algorithm. The implication was correct (see 5.3, 3.1-3.4) but it turned out that I had no cylinder intersection algorithm. This was quite embarrassing since I had announced this non-existing result at many universities in three countries. Gary Miller gave me a prompt proof of the NP-completeness of the cylinder intersection problem for sets (5.6), considerably adding to my panic. Relief came in June at Université de Montréal with the discovery of the simple Las Vegas algorithm presented here (2.4).

My thanks are due to D. Yu. Grigoriev for starting this process, to Gary Miller for his kind hospitality and many stimulating conversations that influenced my entire view of complexity theory, and to Maria Klawe for asking Problem 4.17 a partial answer to which led to the entire material of Section 4.

It is my pleasure to list those institutions whose financial assistance directly contributed to the birth of this paper.

I am indebted to Eötvös University (Budapest) and to the scientific exchange program of Hungary and the U.S.S.R. for sponsoring my visit to

Soviet universities in Fall 1978; to Prof. B. Jónsson of Vanderbilt University (Nashville, Tennessee) for sponsoring my visit to the U.S.; to the Applied Mathematics Group at M.I.T.; to the Department of Mathematics at Université de Montréal; and to the Summer Research Workshop in Algebraic Combinatorics of the Canadian Mathematical Society at Simon Fraser University (Vancouver, July 1979) where Section 4 was conceived and this paper acquired its final form.

REFERENCES

[**1**] L. Adleman and K. Manders, Reducibility, randomness and intractability, Proc. 9^{th} Annual ACM Symp. on the Theory of Computing (1977), 151-163.

[**2**] L. Babai, On the complexity of canonical labeling of strongly regular graphs, SIAM J. Computing, to appear.

[**3**] L. Babai and L. Lovász, Permutation groups and almost regular graphs, Studia Sci. Math. Hung. 8 (1973), 141-150.

[**4**] R. Frucht, Lattices with given abstract group of automorphisms, Canad. J. Math. 2 (1950), 417-419.

[**5**] G. Grätzer, General Lattice Theory, Birkhäuser Verlag, Basel, 1978.

[**6**] D. Yu. Grigoriev and and L. Babai, Isomorphism testing for graphs with bounded eigenvalue multiplicities, in preparation.

[**7**] J. E. Hopcraft and R. E. Tarjan, Dividing a graph into triconnected components, SIAM J. Computing 2 (1973), 135-158.

[**8**] J. E. Hopcraft and R. E. Tarjan, A $V \log V$ algorithm for isomorphism of triconnected planar graphs, J. Comp. Syst. Sci. 7 (1973), 323-331.

[**9**] J. E. Hopcraft and J. K. Wong, Linear time algorithm for isomorphism of planar graphs, Proc. Sixth Annual ACM Sump. on the Theory of Computing (1974), 172-184.

[**10**] Maria Klawe, Marked trees are isomorphism complete, oral communication (1979).

[**11**] R. J. Lipton, The beacon set approach to graph isomorphism, SIAM J. Computing, to appear.

[**12**] Gary L. Miller, On the $n^{\log n}$ isomorphism technique, Proc. Tenth SIGACT Symp. on the Theory of Computing (1978), 51-58.

[**13**] Gary L. Miller, The cylinder intersection problem for 3-sets is NP-complete, oral communication (1979).

[**14**] Gary L. Miller, Trivalent graph isomorphism, oral communication (1979).

[**15**] R. C. Read and D. G. Corneil, The graph isomorphism disease, J. Graph Theory 1 (1977), 339-363.

[**16**] R. Solovay and V. Strassen, A fast Monte-Carlo test for Primality, SIAM J. Computing 6 (1977), 84-85.

[**17**] R. E. Zippel, Probabilistic Algorithms for Sparse Polynomials, Ph.D. Thesis, M.I.T. (1979).

[**18**] R. E. Zippel, Probabilistic algorithms for sparse polynomials, to appear.

Author's address:

László Babai
Computer Science Department
University of Chicago
1100 E. 58th St., Ry 152
Chicago, IL 60637