Alcom-FT Technical Report Series
ALCOMFT-TR-02-75

How to prove conditional randomness using the Principle of Deferred Decisions

Alexis C. Kaporis*‡ Lefteris M. Kirousis *‡ Yannis C. Stamatiou *†§

May 1, 2002

Abstract

We propose a methodology to determine the conditional that should be put on a random object so that its randomness is retained when we run a given algorithm on it. The methodology is based on the Principle of Deferred Decisions. The basic idea is to store the information about the object in "small pieces" in separate registers. The contents of the registers pertaining to the conditional are exposed, while the others are unexposed. Having separate registers for different types of information prevents exposing information that need not be exposed. We use this methodology to prove various randomness invariance results, one of which answers an open question.

1 Introduction

Let G be a uniformly random graph conditional on the number n of its vertices and the number m of its edges (i.e. all G with n vertices and m edges are equiprobable). It is intuitively obvious that if we delete from G a random vertex v and all edges incident on v, the new graph is uniformly random conditional on the new number of vertices $(n-1)$ and the new number of edges m' (m' is a random variable). In other words, given m', the new graph is equiprobable among all graphs with $n-1$ vertices and m' edges. Knuth [3, Lecture 3] has introduced a method, known as the Principle of Deferred Decisions, by which randomness claims such as the above can be verified. To illustrate how the method works in this case, consider $n+m$ cards facing down, or otherwise $n+m$ registers with unexposed content. The first n of them correspond to the vertices of the graph and the remaining m to its edges. The register of a vertex v contains pointers to the registers of the edges incident on v. The register of an edge e contains pointers to the registers of the two endpoints of e. That the registers are unexposed means that the pointers can be specified randomly in a uniform way. To delete a random vertex, point to a vertex-register; expose its contents; expose all edge-registers pointed

*University of Patras, Department of Computer Engineering and Informatics, GR-265 00, Patras, Greece. e-mail: {kaporis,kirousis,stamatiu}@ceid.upatras.gr

†Computer Technology Institute Kolokotroni 3, GR-262 21 Patras, Greece. e-mail: stamatiu@ceid.upatras.gr

‡Research supported by the University of Patras Research Committee (Project C. Carathéodory no. 2445).

§Research partially supported by the IST Programme of EU under contract no. IST-1999-14186 (ALCOM-FT).

to by this vertex-register; delete the exposed vertex-register and the exposed edge-registers; nullify pointers in other vertex-registers that point to deleted edge-registers (without exposing the later vertex-registers). The registers that have not been deleted remain unexposed and therefore they can be filled in randomly in a uniform way. The only conditional, i.e. the only exposed information about the graph, is the new number of the vertex-registers and the new number of the edge-registers.

The Principle of Deferred Decisions can be applied in more complicated situations. Consider a random graph conditional on (i) the number of its vertices, (ii) the number of its edges, (iii) the number of vertices of degree 1 and (iv) the number of vertices of degree 0 (isolated vertices). We claim that upon deleting a random vertex of degree 1 and its incident edge, the new graph is uniformly random conditional on the same type of information. This randomness claim is an immediate consequence of a more general theorem proved in [5] (see also [2]), where the degrees of the literals to be deleted are allowed to take values up to an arbitrary fixed integer k, assuming that the degree sequence of the graph is given up to k (the analogous is true if we allow the degrees to range up to $n-1$). The proof in [5] depends upon counting all possibilities. The result however can also be proved using the Principle of Deferred Decisions. Indeed, it suffices to augment the argument of the previous paragraph with the additional assumption that each vertex v is linked with an *exposed* degree-register that contains a three-valued parameter indicating whether the degree of that vertex is 0, 1, or ≥ 2. After any deletion step, these exposed registers are updated. It is helpful to assume, as Achlioptas [1] does when analyzing algorithms on random formulas, that there is an "intermediary", i.e. an agent independent of the deleting algorithm, who does the necessary updating of the unexposed registers. A similar proof, based on the Principle of Deferred Decisions, can be provided for any k, fixed or not.

However, with even more complicated situations, we may get onto a slippery slope. To explain why, consider the following example: Let a *B&W-graph* be a graph whose *edges* are either black or white. Call a vertex *all-white* if all the edges incident on it are white. Let the w-degree of a vertex v be the number of all-white vertices that v is connected with (notice that a black edge incident on v does not count towards the w-degree of v, while a white edge incident on v may or may not count towards the w-degree of v). Suppose we are given a random B&W-graph G conditional that the number of its vertices is given, and that for each vertex v, the w-degree of v as well as the number of black edges and the number of white edges incident on v are given. All other characteristics of G are assumed to be uniformly random. Formally, given a fixed integer n and a fixed array of integers $d_{w,i,j}$, where $w, i, j = 0, \ldots, n-1$, then G is chosen with equal probability among all B&W graphs such that for each w, i, j, G has $d_{w,i,j}$-many vertices v, such that the w-degree of v is w, and also i white edges and j black edges are incident on v; we assume that the values of the array are such that there is at least one such graph. Suppose now that we delete from G a vertex v that is chosen uniformly at random among all vertices with a specified w-degree, say 0. Suppose we also delete all edges (black and white) incident on v. Is the new graph random conditional on the same type of information?

Prima facie, one may think that the answer to the question of the previous paragraph is yes. Indeed, suppose that we have exposed registers that for each vertex give its w-degree, as well as the number of black edges and the number of white edges incident on it. All other information about the graph is assumed to be unexposed, i.e. random. After the deletion of a vertex v as previously described, and the subsequent deletion of all edges incident on v, we update again all exposed registers. Notice that since the same type of registers are exposed before and after the deletion, we may be tempted to conclude that the same type of information about the graph is known before and after the deletion. Therefore, the answer to the question seems to be affirmative. Unfortunately, this argument is erroneous. To see why, observe that if after the deletion of v, the exposed w-degree of another vertex u increases, then we can infer that in the new graph there exists at least one vertex v' that has just become all-white, that this transformation of v' must be the result of the deletion of a black edge joining v with v', and that additionally u is connected with at least one of these newly-all-white vertices. However the later type of information is not supplied by the exposed registers. The exposed registers only give the w-degree of u and the number of black and white edges incident on it. They do not specify a subset of the all-white vertices connected with u. (The same argument goes through if not only the w-degree of v, but also the number of white edges and the number of black edges incident on it are specified, as long as the later is non-zero.)

The essential reason for the error in the above argument is that the execution of an algorithmic operation on the graph, like the deletion of a vertex and the edges incident on it, may implicitly expose information that cannot be inferred from what is explicitly exposed. This is a common source of mistakes in randomness claims. We describe in Section 3 a methodology for proving conditional randomness by the Principle of Deferred Decisions that avoids such uncontrolled exposure of information. The basic idea is the following: assume that information about the random structure is stored in "small pieces" into registers, some of which are unexposed (i.e. the information they contain can be randomly specified, conditional on the type of information this register is assumed to contain) and some are exposed (i.e. the information they contain is not random, in other words it is part of the conditional). After an algorithmic operation, like the deletion of a vertex and its incident edges, is executed, all exposed registers are updated. Additionally, any unexposed registers that contain information that can be implicitly inferred are also exposed. If after the execution of the operation it turns out that the same type of registers are exposed as in the beginning, then this operation retains randomness. It pays to store "small pieces" of information in each register, so that exposures do not reveal information that need not be revealed. Of course one might claim that the safest way to prove conditional randomness claims is by counting arguments, and not through the Principle of Deferred Decisions. This might be true, but it is also true that in complicated situations, counting arguments are practically impossible.

The methodology outlined in the previous paragraph, as we will see through its specific applications, makes easy the specification of what the a priori exposed information should be, given the type of operations that the algorithm allows. Such considerations have lately attracted much attention in view of the increased interest on the probabilistic analysis of heuristics on random graphs and formulas that are studied in the context of the threshold problem for coloring or satisfiability (for an overview, see [4]).

Also, using this methodology, we answer a question posed by Molloy (in a private communication) as to what characteristics should be assumed to be conditional for a random CNF-formula comprised of 3- and 2-clauses, in order that its randomness be retained, given that we execute a Davis-Putnam heuristic where the literals to be satisfied are selected on the basis of the number of their appearances in each of the two types of clauses (Theorem 2 in Section 3).

2 Terminology and Notation

Our results can be applied in various contexts related to random graphs or formulas. However for concreteness, we first present them in the context of random formulas comprised only of 3- and 2-clauses. We introduce below the related terminology and notation.

Let V be a set of variables of cardinality n. Let L be the set of literals of V, i.e. elements of V or negations of them. A k-clause is a disjunction of exactly k literals from L. Let ϕ be a formula in Conjunctive Normal Form (CNF) comprised of only 3- and 2-clauses. Let the total number of clauses of the formula be m. Let also C_3 (respectively, C_2) denote the collection of 3-clauses (respectively, 2-clauses) of ϕ and let c_3, c_2 and l be the cardinalities of the sets C_3, C_2 and L, respectively. Obviously $c_3 + c_2 = m$ and $l = 2n$.

For $i = 0, 1, \ldots, m$, let D_i be the set of literals in L that have exactly i appearances in ϕ. The elements of D_i are said to have *degree* i. Literals whose negation is in D_0 are called *pure*. Notice that according to our terminology, a literal in L whose variable does not appear in the formula is pure.

Let D_1^3 (respectively, D_1^2) be the set of literals that have exactly one appearance in ϕ which occurs in a 3-clause (respectively, a 2-clause) of ϕ. Obviously, D_1 is the disjoint union of D_1^3 and D_1^2. Let also $D_{1\times 1}^2$ be the subset of D_1^2 comprised of literals that appear in a 2-clause whose second literal also belongs to D_1^2.

Let d_i, d_1^3, d_1^2 and $d_{1\times 1}^2$ be the cardinalities of D_i, D_1^3, D_1^2 and $D_{1\times 1}^2$, respectively. Obviously, $d_1 = d_1^3 + d_1^2$.

Consider the collection of formulas comprised of 3- and 2-clauses that have a given, fixed value for the parameters l, c_3, c_2, d_0, d_1^3, d_1^2 and $d_{1\times 1}^2$. Make this collection into a probability space by assigning to each one of its elements the same probability (we assume that the values of the parameters are such that this space is not empty). An element of this space is called a (uniformly) random $\{3,2\}$-CNF formula conditional on the values of the parameters l, c_3, c_2, d_0, d_1^3, d_1^2 and $d_{1\times 1}^2$. Similarly one can define random graphs conditional, e.g., on the number of edges and vertices, etc. We call such formulas or graphs, etc., conditionally random objects. Given an algorithm on a random object, the operations that it executes will be referred to as *permissible operations*. For random $\{3,2\}$-CNF formulas, we will examine, as a first example, the following three types of permissible operations on them:

- *Setting a pure literal.* Select uniformly at random (u.a.r) a pure literal, set it to TRUE and delete all clauses where it appears.
- *Setting a degree-one literal from 3-clauses.* Select u.a.r. a literal in D_1^3, set it to FALSE, delete it from the 3-clause it appears and delete all clauses where its negation appears.
- *Setting a degree-one literal from 2-clauses.* Select u.a.r. a literal in D_1^2, set it to FALSE, delete it from the 2-clause it appears and delete all clauses where its negation appears. Then as long as there are 1-clauses, choose u.a.r. a 1-clause, set its literal to TRUE, delete all clauses where it appears and delete its negation from any clause it appears. Ignore

(delete) any empty clauses that may occur during this procedure (this last provision is only a technicality in order to study the randomness of the formula independently from its satisfiability – of course, when such a permissible operation is used as a subroutine of an algorithm for satisfiability, the later should stop and report unsatisfiability when an empty clause appears).

3 Results

Theorem 1 *Let ϕ be a random $\{3,2\}$-CNF formula conditional on the values of the parameters l, c_3, c_2, d_0, d_1^3, d_1^2 and $d_{1\times 1}^2$. If any permissible operation as above is applied on ϕ, then the formula obtained is a random $\{3,2\}$-CNF formula conditional on the new values of the parameters l, c_3, c_2, d_0, d_1^3, d_1^2 and $d_{1\times 1}^2$.*

Proof Notice that the permissible operations do not differentiate between degree-one literals that appear in 2-clauses on the basis of the degree of the second literal of the corresponding 2-clause. Still, according to the statement of the theorem, randomness is preserved if it is conditional not only on d_1^2 but also on $d_{1\times 1}^2$. The reason for this will become clear later in this proof.

The proof is a non-trivial, we believe, application of the Principle of Deferred Decisions (see [3, Lecture 3]). We introduce a methodology to exploit this principle, which we call the methodology of *controlled exposure of information.*

An object like a formula or a graph can be modelled by a data structure. It is convenient to think of a data structure as a collection of registers that contain information about the object. For example, a data structure modelling a graph includes a register for each vertex which contains pointers to the registers of the edges incident on the vertex. It also includes a register for each edge which contains pointers to the registers of the endpoints of the edge. A data structure modelling a formula includes a register for each literal which contains pointers to the appearances of the literal in the clauses. It also includes a register for each appearance of each literal which contains a pointer to that literal (more information about the registers of a formula is supplied below). Registers are partitioned into groups; the elements of each group contain different type of information for the same part of the modelled object. For example, for each vertex of a graph, we may have several registers in one group: one with pointers to the edges incident to the vertex, another with the degree of this vertex, etc. Sometimes (but not always) we refer to the registers belonging to the same group as sub-registers of the group. We also assume that for each group, we have a head register with pointers to its sub-registers. When a sub-register of a group contains a pointer to another group, this pointer is assumed to be to the head register of that other group. Intuitively, the storing of different types of information into separate sub-registers is done in order to avoid exposing all information about a part of the modelled object when it is necessary to expose only a "small piece" of it.

A *data structure with unexposed information* is a data structure whose (sub-)registers are partitioned into two categories, named *unexposed* and *exposed* registers. The partitioning of the registers into these two categories is done according to rules given in the definition of the structure. These rules are based on the type of the contents of the registers. The head registers of the groups are always exposed. Intuitively, such a structure models an object whose characteristics stored in the unexposed registers are random, conditional on the information stored in the exposed registers. The same group may contain both exposed and unexposed sub-registers. For example, although the specific edges where a vertex appears may not be exposed, its degree may be exposed. Similarly, all positions in a formula where a literal appears may be exposed, but the negation of this literal may not be exposed (more on this below). We will now define the notion of *randomness preserving procedures* on such data structure. Each such procedure corresponds to *one* permissible operation on the random object.

DEFINITION 1 *A* randomness preserving *procedure on a data structure with unexposed information is a sequence of steps that belong to one of the following three types:*

- *Deletion of a register and all other registers of the same group (e.g., deletion of a vertex register, or a register of an appearance of a literal in a clause, together with all sub-registers of the same group). Upon Deletion, all registers of the group are considered as exposed.*
- *Update of the contents of an undeleted exposed register.*
- *Exposure of a hitherto unexposed register, i.e. the change of its category from unexposed to exposed (without deleting it).*

In addition, a randomness preserving procedure is subject to the following restrictions:

- *After the Deletion steps, the contents of all unexposed registers are updated. It may be convenient to think of the updates of unexposed vertices as being carried out not by the procedure, but by an independent agent called* intermediary *(also see [1]).*
- *After the updates of the unexposed registers by the intermediary, the procedure updates the exposed undeleted registers that need to be updated. Additionally, the procedure exposes any unexposed registers that contain information that can be implicitly inferred from the currently exposed registers (including the deleted ones) and from the knowledge of which were the exposed registers that were updated. As a remark, notice that the checking that the procedure satisfies this requirement can be carried out by the algorithm designer by going through all unexposed registers and examining if their contents (or any part of them) can be inferred from the currently exposed registers (the deleted ones included) and from the knowledge of which were the unexposed registers that were updated.*
- *After the execution of the procedure, the partition of the registers into unexposed and exposed in the new data structure should be subject to the same rules as in the original structure. Informally, the same type of information should be revealed both before the execution of the procedure and after it. This information may, of course, have changed as a result of the execution. As a remark, notice that the consequence of this requirement for the algorithm designer is that any type of register that should be exposed during the procedure, must be assumed to be a priori exposed. It is at this point where, we believe, the advantage of storing various types of information in separate registers in "small pieces" emerges.*

Observe that by the Principle of Deferred Decisions, in order to prove a randomness claim like the theorem under consideration, it suffices to find a data structure with unexposed information which models the conditionally random object referred to in the claim and then show that the permissible operations on the random object can be implemented as randomness preserving procedures on the structure with unexposed information. These two tasks although often trivial, sometimes, like in the present case, require a careful analysis.

We describe below a structure with unexposed information, say $\mathcal{S}$, that models a random $\{3,2\}$-CNF formula conditional on the parameters l, c_3, c_2, d_0, d_1^3, d_1^2 and $d_{1\times1}^2$. We also design randomness preserving procedures on $\mathcal{S}$ that implement the permissible operations on the formula.

- For each literal t in L, the structure $\mathcal{S}$ contains a group of three sub-registers collectively called literal registers. The first, called lit-appearance register, contains a list of pointers to the position registers where this literal appears (see below for what the position registers are); the second, called lit-degree register, contains a four-valued parameter that indicates whether t belongs to D_0 (i.e. t does not appear in the formula), or D_1^3 (i.e. t has degree 1 and appears in a 3-clause), or D_1^2 (i.e. t has degree 1 and appears in a 2-clause), or none of these (i.e. t has degree at least 2); the third, called lit-negation register, contains a pointer to the literal register of $\neg t$.

- For each appearance of a literal t in the formula (there are $3c_3 + 2c_2$ such appearances), the structure $\mathcal{S}$ contains a group of three sub-registers collectively called position registers. The first, called pos-label register, contains a pointer to the literal register of t; the second, called pos-clause-cardinality register, contains a Boolean variable indicating whether the clause of this particular appearance is a 3-clause or a 2-clause; the third, called pos-clause-tuple register, contains pointers to the other position registers of the same clause.

- Lit-degree registers are exposed; pos-clause-cardinality registers are exposed; pos-clause-tuple registers of 2-clauses are exposed, while pos-clause-tuple registers of 3-clauses are unexposed (as we will see, the exposure of pos-clause-tuple registers of 2-clauses is dictated by the requirement that the procedures retain randomness); lit-appearance registers of literals of degree one that appear in 2-clauses (i.e. have d_1^2 in their lit-degree register) are exposed (they contain a single pointer to a position register of a 2-clause), while all other lit-appearance registers (including those with the value d_1^3) are unexposed; pos-label registers of literal appearances in 2-clauses of degree 1 are exposed, while all other pos-label registers (including those in 3-clauses with degree 1) are unexposed (as we will see, the exposure of the lit-appearance and pos-label registers corresponding to literals with a unique appearance in 2-clauses is dictated by the requirement that the procedures retain randomness); finally all lit-negation registers are unexposed.

Remark It is important to notice that intuitively the fact that lit-negation literals are unexposed means that literals are considered as labels that do not reveal the pairing of literals with opposite logical sign. In other words, this pairing is random.

The description of $\mathcal{S}$ is complete. Now observe that based on the above rules of categorizing the registers into exposed and unexposed, we can conclude that the information that can be obtained from $\mathcal{S}$ consists of the values of the following parameters: l (the count of the literal registers), c_2 (the count of the position registers of 2-clauses, divided by 2), c_3 (the count of the position registers of 3-clauses, divided by 3), d_0 (the count of literal registers whose (exposed) lit-degree register contains d_0), d_1^3 (the count of literal registers whose (exposed) lit-degree register d_1^3), d_1^2 (the count of literal registers whose (exposed) lit-degree register contains d_1^2) and $d_{1\times 1}^2$ (double the count of pairs of (exposed) lit-appearance registers of degree one that point to position registers of the *same* 2-clause – recall that pos-clause-tuple registers of 2-clauses are exposed). All other information that can be extracted from $\mathcal{S}$ can be expressed in terms of the values of the parameters l, c_3, c_2, d_0, d_1^3, d_1^2 and $d_{1\times 1}^2$, only.

The only thing that remains to be explained to complete the proof of Theorem 1 is the design of the randomness preserving procedures on $\mathcal{S}$ that correspond to various permissible operations on the random $\{3,2\}$-CNF formula. The description of these procedures should be straightforward, we believe, from what has been explained so far. We only point out that when a permissible operation shrinks a clause from a 3-clause to a 2-clause, the corresponding randomness preserving procedure should expose the pos-clause-tuple registers of 2-clauses. This is because when moving a pair of literal appearances from 3-clauses to 2-clauses as a result of the shrinking, the pairing of these literal appearances in a 2-clause can be inferred from the knowledge that their respective pos-clause-cardinality registers were just updated (before the move to 2-clauses, the pair is determined by the deleted, and therefore exposed, pos-clause-tuple register of the literal set to FALSE). Also, when the procedure changes the value of a lit-degree register of a literal t from d_1^3 to d_1^2, it should expose the lit-appearance register of t (the lit-appearance register points to a single position register) and should also expose the pos-label register that points to t. This exposure is necessary because the updating from d_1^3 to d_1^2 of the lit-degree register of t reveals that t appears in the position just moved from a 3-clause to a 2-clause. This necessitates that all these registers are a priori exposed. Notice, as a remark, that the lit-negation registers were unexposed and remained so. This completes the proof of Theorem 1. ■

Remark Theorem 1 answers the question which parameters of the formula should be included in the conditional when we wish to delete pure literals and literals with a single appearance and, in addition, to differentiate between literals whose single appearance occurs in 3-clauses and 2-clauses, without destroying the randomness of the formula. It is expected that the cardinality of each of the classes of these literals should be included in the conditional, i.e. be non-random. However, perhaps it is not obvious that also the cardinality of 2-clauses whose both literals have a single appearance in the formula should be included in the conditional. We are lead to the conclusion that the parameter $d_{1\times 1}^2$ should be included in the conditional by the type of registers that the method of controlled exposure forces the algorithm designer to assume to be a priori exposed. In other words, this method enables the specification of what information should be conditional in order to retain randomness under a given set of permissible operations. The method generalizes to literals of arbitrary degree (see next Theorem), however then increasingly more complex information about the 2-clauses should be assumed to be non-random. On the contrary, it is easy to see, by applying this method, that in order to allow the deletion of a random literal with any specified number of appearances in a $\{3,2\}$-CNF formula, but without assuming any knowledge on how many of these appearances occur in 3-clauses or 2-clauses, it is sufficient to keep in the conditional only the degree sequence $d_i, i = 0, \ldots, m$ (d_i is the cardinality of the set of literals whose total number of appearances in the formula is i). The analogous is true if we restrict the deletions to literals of total number of appearances up to a fixed value k.

We now come to the generalization of the previous result to arbitrary degrees. Fix an even integer $2n$ (intended to be the number of literals of a formula), and an integer c_3 (intended to be the number of 3-clauses in a formula). A *structure for 2-clauses and degree-sets that is transparent with respect to negations* (structure, in short) is a set of unordered pairs $\underline{C}_2$ of integers from $\{1, \ldots, 2n\}$ (intended to represent the collection of 2-clauses of the formula) together with a collection of sets $D_i \subseteq \{1, \ldots, 2n\}, i = 0, \ldots, c_3$, such that $\sum_i i|D_i| = 3c_3$ (intended to represent the collection of sets of literals whose number of appearances in 3-clauses is i). Fix a structure G as described. A random formula ϕ conditional on G is constructed as follows: uniformly at random choose c_3 unordered triplets from $1, \ldots, 2n$ so that all integers in each D_i appear in exactly i such triplets; denote this set by $\underline{C}_3$; uniformly at random select a one-to-one and onto mapping $\mathrm{neg} : \{1, \ldots, n\} \to \{n+1, \ldots, 2n\}$; in the tuples of $\underline{C}_3$ and of $\underline{C}_2$, replace each $k = 1, \ldots, n$ with x_k and each $\mathrm{neg}(k), k = 1, \ldots, n$, with $\neg x_k$ and denote by C_3 and C_2, respectively, the sets of clauses thus obtained; let the formula ϕ be the one that has as 3-clauses and as 2-clauses the sets C_3 and C_2, respectively (we assumed that the variables of the formula are labelled as $x_1, \ldots, x_n$). Notice that although this is a severely restricted notion of a randomness, still a lot of information about the formula is uniformly random, namely the structure of 3-clauses and the pairing of literals of opposite logical sign. (Notice that since the negation function "neg" was random, a literal and its negation may appear in the same clause. If we wish to avoid this, "neg" should be a random one-to-one and onto mapping conditional that for no $i = 1, \ldots, n$ both i and $\mathrm{neg}(i)$ appear in the same tuple of either $\underline{C}_3$ or $\underline{C}_2$.) The method of controlled exposure of information leads now to the following result that answers a question posed by Molloy (in a private communication). We omit its proof.

Theorem 2 *Let ϕ be a random $\{3,2\}$-CNF formula conditional on a given structure G. If for arbitrary i and j, we choose uniformly at random a literal t with i appearances in 3-clauses and j appearances in 2-clauses, assign to it the value* TRUE *and perform the necessary deletions and shrinking of clauses accompanied by repeated setting to* TRUE *of literals in 1-clauses, as long as 1-clauses exist, then the new formula is random conditional on its structure G'.*

Finally, as a further application, let us see what information should be placed in the conditional if we wish to permit the deletion of vertices of a specified w-degree from a random B&W-graph (the notion of B&W-graphs is defined in the Introduction).

Given a B&W-graph, let W be the set of white edges with at least one endpoint incident on an all-white vertex. Call W the set of w-degree witnesses. We omit the details, but one can again define a notion of random B&W-graph conditional on the number of its vertices, the total number of its edges and the set of w-degree witnesses (arbitrarily place edges between vertices that are not all-white and then arbitrarily color them black or white, taking care to color black at least one edge for each non-all-white vertex).

Theorem 3 *If we delete a random vertex of a specified arbitrary w-degree from a B&W-graph that is random conditional on its number of vertices, its total number of edges and its set of w-degree witnesses, then the new graph is random conditional on the new number of vertices, the new total number of edges and the new set of w-degree witnesses.*

An analogous result holds if the deleted vertex has specified numbers of white and black edges incident on it (the conditional in the later case must be augmented to contain the sequence $d_{i,j}$ which gives the cardinality of vertices with i white and j black edges incident on them). We conclude this paper by the following

Informal Conjecture The information put in the conditionals of Theorems 1, 2 and 3 is the least possible, i.e. with weaker conditionals, randomness would not be retained.

Acknowledgment

The second author thanks D. Achlioptas and M. Molloy for several discussions about probabilistic arguments. Of course, the responsibility for any mistakes in the present work lies exclusively with its authors.

References

[1] D. Achlioptas, "Lower bounds for random 3-SAT via differential equations," *Theoretical Computer Science* 265 (1-2) (2001), pp. 159–185.

[2] A. Broder, A. Frieze and E. Upfal, On the satisfiability and maximum satisfiability of random 3-CNF formulas, in *41st Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)* (1993), pp. 322–330.

[3] D.E. Knuth, *Stable Marriage and its relation to other combinatorial problems: an introduction to the mathematical analysis of algorithms* (English Edition: CRM Proceedings & Lecture Notes 10, American Mathematical Society, 1997).

[4] M. Molloy, Thresholds for colourability and satisfiability in random graphs and Boolean formulae, in J. Hirschfeld, ed., *Surveys in Combinatorics, 2001* (LMS Lecture Note Series 288, Cambridge University Press, 2001), pp. 166–200.

[5] B. Pittel, J. Spencer and N. Wormald, "Sudden emergence of a giant k-core in a random graph," *Journal of Combinatorial Theory Series B* 67 (1996), pp. 11–151.