Basic Research in Computer Science

Checking Consistency of Pedigree Information is NP-complete

(Preliminary Report)

Luca Aceto
Jens A. Hansen
Anna Ingólfsdóttir
Jacob Johnsen
John Knudsen

BRICS Report Series **RS-02-42**

ISSN 0909-0878 **October 2002**

See back inner page for a list of recent BRICS Report Series publications. Copies may be obtained by contacting:

BRICS
Department of Computer Science
University of Aarhus
Ny Munkegade, building 540
DK–8000 Aarhus C
Denmark
Telephone: +45 8942 3360
Telefax: +45 8942 3255
Internet: BRICS@brics.dk

BRICS publications are in general accessible through the World Wide Web and anonymous FTP through these URLs:

`http://www.brics.dk`
`ftp://ftp.brics.dk`
This document in subdirectory `RS/02/42/`

Checking Consistency of Pedigree Information is NP-complete (Preliminary Report)

Luca Aceto[1], Jens A. Hansen[1], Anna Ingólfsdóttir[1,2], Jacob Johnsen[1], and John Knudsen[1]

[1] **BRICS** (**B**asic **R**esearch in **C**omputer **S**cience), Centre of the Danish National Research Foundation, Department of Computer Science, Aalborg University, Fr. Bajersvej 7E, 9220 Aalborg Ø, Denmark, luca@cs.auc.dk, alsted@cs.auc.dk, annai@cs.auc.dk, johnsen@cs.auc.dk, johnk@cs.auc.dk

[2] deCODE Genetics, Sturlugata 8, 101 Reykjavik, Iceland, annai@decode.is

Abstract Consistency checking is a fundamental computational problem in genetics. Given a pedigree and information on the genotypes of some of the individuals in it, the aim of consistency checking is to determine whether these data are consistent with the classic Mendelian laws of inheritance. This problem arose originally from the geneticists' need to filter their input data from erroneous information, and is well motivated from both a biological and a sociological viewpoint. This paper shows that consistency checking is NP-complete, even in the presence of three alleles. Several other results on the computational complexity of problems from genetics that are related to consistency checking are also offered. In particular, it is shown that checking the consistency of pedigrees over two alleles can be done in polynomial time.

AMS SUBJECT CLASSIFICATION (1991): 68Q25, 92D10.
CR SUBJECT CLASSIFICATION (1991): F.2.2, J.3.
KEYWORDS AND PHRASES: Consistency checking, pedigrees, genotypes, NP-completeness, satisfiability, polynomial time complexity.

1 Introduction

Since the early days of development of the fields of (molecular) biology and genetics, mathematicians have thought that these fields needed an appropriate mathematical foundation, and that their development could benefit from a formal mathematical treatment. Despite these early calls for the "mathematization" of biology from some members of the mathematical community, it is only recently that the level of aspiration and the new challenges that have arisen in molecular biology and genetics have made these sciences increasingly dependent on mathematical modelling, mathematical analysis, and computation. In fact, modelling and algorithmic techniques from combinatorics and computer science are showing promise in helping the life sciences attain their goals. The reason for this interest in combinatorics and computer science in the life sciences is at least twofold. On the one hand, many parts and processes of the human body, and living organisms in general, work on a discrete basis, and it is on the discrete domain that

combinatorics and computer science excel. On the other, the amount of available data in the life sciences has increased, and will continue increasing, by orders of magnitude, and the analysis of these data requires automation and sophisticated algorithmic techniques.

A paradigmatic problem from the field of genetics in which the use of algorithmic techniques is by now widespread, and is embodied in software tools like Allegro [6], Genehunter [11], Merlin [1] and Pedcheck [12], is that of linkage analysis. *Linkage analysis* is a well established, statistical method used to relate genes in the human genome to some biological trait that an individual possesses. Example traits that may be investigated range from simple ones like blood type and eye colour to those that may predispose an individual for a disease. Genes causing major diseases (e.g., Parkinson's disease, obesity and anxiety) have already been discovered using this technique [2].

In order to track the inheritance of genetic traits, geneticists use computation structures called pedigrees. A *pedigree* describes the family relations amongst a collection of individuals, and (possibly partial) information on their genotypes—i.e., on the pairs of alleles at a locus in their genome. (An *allele* is one of the possible forms a gene may have.) Pedigrees are the subject of algorithmic analysis via methods like linkage analysis.

A computational problem that is closely related to that of linkage analysis is *consistency checking*. Given a pedigree and information on the genotypes of some of the individuals in it, the aim of consistency checking is to determine whether these data are consistent with the classic Mendelian laws of inheritance (see, e.g., the reference [10] and Sect. 2). If it turns out that the inheritance of the genotypes in the pedigree is in conflict with the Mendelian laws of inheritance, then the pedigree and the information on the genotypes are *inconsistent*. If no such conflict arises, then the data are *consistent*.

The problem of consistency checking arose originally from the geneticists' need to filter their input data from erroneous information, because inconsistent data are undesirable. According to [4], it is essential that all Mendelian inconsistencies be eliminated prior to linkage analysis as "a few inopportunely placed errors, if ignored, can tremendously affect evidence for linkage." Furthermore, as reported in [12], in many real-life cases the manual identification of inconsistencies can be very difficult, time consuming, and sometimes unsuccessful. It would therefore be most helpful to have automatic tool support for this task.

Another motivation for consistency checking is its applicability in determining family relationships. A DNA test is very useful in genealogical investigations, paternity issues and criminal investigations. For instance, the point of paternity issues has recently been brought up in the Danish press [3], where it is claimed that up to 10% of the Scandinavian population have wrong paternal information, and that the interest of such investigations is growing rapidly. This could be accelerated by the growing possibilities in society for performing DNA tests. According to [7], it is now possible to get a 10 marker fingerprint of a chromosome for approximately 300 euro by utilizing a DNA sampling kit at your own home. Thus it seems that genealogy studies based on genetic data have the possibility of becoming widely accessible in the near future. Of course, these studies must be based on consistent genealogical data to be meaningful. Note that it is not only with respect to humans that the verification of family relationships is rel-

evant. For instance, it is important that a horse breeder has the ability to document the family relationships of his/her horses, and that a botanist is certain of the family relationship of plants used in an experiment. Basically all living organisms are based on DNA, and can thus be subjected to consistency checking.

Hence, consistency checking is a well motivated problem from both a biological and a sociological viewpoint. Another issue is whether it is computationally feasible. The aim of this paper is to show that consistency checking is NP-complete, and thus that the existence of consistency checking algorithms that have polynomial worst case complexity is unlikely—cf., e.g., the claim by O'Connell and Weeks that their "new genotype-elimination algorithm is guaranteed to detect every Mendelian inconsistency efficiently and quickly" [13, pp. 1739–1740]. To the best of our knowledge, this is a new result in both computer science and genetics.

The first step in our technical developments is one of modelling and formalization, as is often the case in situations in which the computational problem at hand is presented using informal models and notations in the original literature. After discussing some of the biological background for our work (Sect. 2), we therefore propose a simple formal model for pedigrees and associated genotype information, argue that this model is in agreement with the informal one used in the genetics literature, and use it to formalize the consistency checking problem (Sect. 3). The consistency checking problem is shown to be NP-complete in Sect. 4, even in the presence of *three* alleles. Our proof of NP-hardness for this problem is based on a reduction from 3SAT (a classic NP-complete problem—see, e.g., [15]), and uses pedigrees with loops. As stated in [13], consistency checking of "pedigrees with loops…continues to pose daunting computational challenges." This is confirmed by the use of looping pedigrees in our NP-completeness proof, and by the fact that pedigrees without loops can be checked for consistency in polynomial time (Thm. 4). Moreover, since we wish to use our result to infer the hardness of consistency checking in genetically meaningful situations, we offer a detailed discussion of the reasonableness from a genetic viewpoint of our encoding of 3SAT in terms of pedigrees and genotype information. Sect. 5 presents results on the computational complexity of three problems from the genetics literature that are closely related to consistency checking. In particular, we show that checking consistency of pedigrees over *two* alleles is in P.

A full account of the work presented in this extended abstract may be found in [8], to which the reader is referred for more details, and information on further results and future work.

Related Work As previously mentioned, linkage analysis is a statistical method used to relate genes in the human genome to some biological trait that an individual possesses. Like this method, other pedigree analysis techniques involve calculations with probability distributions describing the likelihood of gene transmission from one generation to the next. The study [16] investigates the structural complexity of two important problems that are at the heart of many statistical pedigree analysis methods, viz. the calculation of the so-called *marginal probability*, and that of computing the so-called *maximum likelihood*. Both problems are shown NP-hard in *op. cit.* even for pedigrees without inbreeding loops, and with focus on a single gene. These results counter the common belief amongst practitioners that inbreeding loops are the main source of com-

putational difficulty in pedigree analysis. Our NP-completeness result for the problem of checking consistency of pedigree information offers another example of a fundamental problem in pedigree analysis that is complex even in the absence of inbreeding.

2 Biological Preliminaries

It is well understood that we inherit genetic material from our ancestors. The idea of inheriting traits was discovered by Gregor Mendel (1865). Arguably, Mendel's main contribution to modern genetics was the Mendelian laws of inheritance. Since these principles guide the development of the formal model presented in Sect. 3, we now present Mendel's laws, which specify the concept known as *Mendelian inheritance*, verbatim from [10], and discuss their impact on our work. For further background information on the concepts from genetics on which our work is based, we refer the reader to [8, Appendix A].

> **Unit factors in pairs:** *Genetic characters are controlled by unit factors that exist in pairs in individual organisms.*
> The unit factor, as Mendel describes it, is today known as a *gene*. Genes occur in pairs, a paternal and a maternal allele, which reside on each of the chromosomes constituting a chromosome pair. This law implies that the genotype of an individual should always be considered as a pair.
> **Dominance/Recessiveness:** *When two unlike unit factors responsible for a single character are present in a single individual, one unit factor is dominant over the other, which is said to be recessive.*
> Dominance and recessiveness refer to phenotype, and are not considered further in our biological model. We assume that it is always the individual's genotype, and not its phenotype, that is considered known. That is, it is always the specific alleles we use, and never an abstraction of the trait they code.
> **Segregation:** *During the formation of gametes, the paired unit factors separate and segregate randomly so that each gamete receives one or the other with equal likelihood.*
> The principle of segregation states that any combination of alleles, which form a genotype, should be considered equally likely to occur. This means that all possible combinations of the paternal and maternal alleles are possible.
> **Independent Assortment:** *During gamete formation, segregating pairs of unit factors assort independently of each other.*
> Independent assortment is central to the biological model. It states that each gene in a chromosome is inherited independently of all other genes. This makes it possible to consider only one gene at a time, as the gene under investigation is inherited independently from all of the other genes. This is the main motivation for *single gene analysis*, which is the type of analysis normally performed in genetics.

The Mendelian laws of inheritance have been chosen by many researchers as the starting point in their investigations (see, e.g., the references [5,13,14]). Furthermore, a large number of traits are today known to be caused by single gene disorders [9].

Pedigrees In order to track the inheritance of genetic traits, geneticists use structures called *pedigrees*. In our setting, we always assume that a pedigree has some (possibly incomplete) genotype information associated with it. A pedigree consists of individuals and their family relations. (See Fig. 1-1.) Each individual has an associated genotype, which we write just below the individual, that consists of two of the alleles for the gene under consideration.

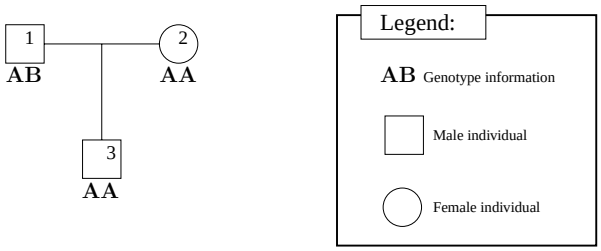

Figure 1-1: Example of a pedigree.

The genotype of each individual in a pedigree is either known through a genotyping process, or it is a set of genotypes which can be inferred from the Mendelian laws of inheritance. As the principle of segregation states, an individual inherits one allele from each parent. *Genotype phase* refers to the heredity of each allele of the genotype for an individual, that is, whether a given allele is inherited from the paternal or maternal side. In general, by observing a chromosome pair, it is not possible to say which part is inherited paternally or maternally. We have chosen to treat the genotype of each individual as phase unknown, irrespective of the knowledge of that of its ancestors, unless otherwise stated. When describing genotypes, we only write one of the equivalent genotypes ($\mathbf{AB}$ is equivalent to $\mathbf{BA}$).

Consistency Checking A pedigree with associated genotype information is *consistent* when all observed or inferred genotypes are possible according to the Mendelian laws of inheritance [14].

There can be several reasons for inconsistencies in a pedigree and its genotype information. For instance, a family relationship could be misspecified, or there could be errors in the genotyping process or mutation. Generally it is not possible to determine the source of error; it is simply established that the given genotype information is inconsistent with the pedigree under investigation. By way of example, consider the pedigree shown in Fig. 1-2. (In this pedigree, and in what follows, whenever an individual has no genotype information attached to it, then no genotype information is available for him/her.) This pedigree is inconsistent in two places. One of the inconsistencies is due to the fact that it is impossible that individual 5 could have inherited the $\mathbf{C}$ allele from either of her parents. To observe the other inconsistency, it is necessary to reason about more than two generations in the pedigree. The inconsistency appears because individ-

ual 7 cannot have inherited his **C** allele from individual 3, because individual 3 inherits her alleles from parents that do not have a **C** allele.

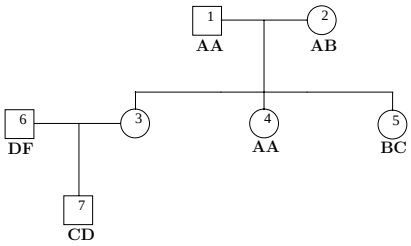

Figure 1-2: An inconsistent pedigree.

The example we have just presented does not capture the complexities that arise when dealing with large pedigrees. Each test is simple, but considering that multiple individuals can have different possible genotypes, and that the genotype of some individuals must be inferred by analyzing several generations, it should be clear that it can be a daunting task to analyze pedigrees by hand. As pointed out in Sect. 1, looping pedigrees present some special problems. A *loop* in a pedigree is a sequence of arcs that starts and ends in the same individual [13]. An example of a consistent, looping pedigree is depicted in Fig. 1-3.

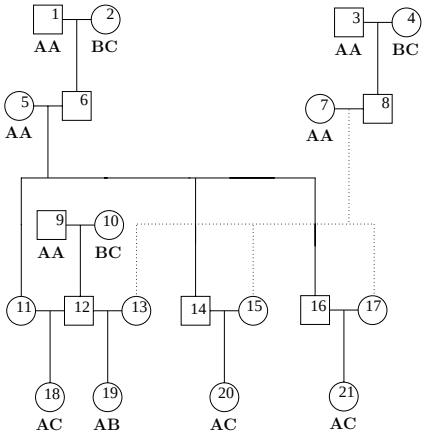

Figure 1-3: A consistent looping pedigree. The dotted line also represents a parents-child relationship

3 Formalizing Gene Inheritance and Mendelian Consistency

As already mentioned in Sect. 2, a pedigree is a fundamental computation structure used in genetics. In order to reason about pedigrees and the genotype information that they contain, we need a formal model for them. This we now proceed to present. We first offer models for pedigrees, and their associated genotype information. Then we use these models to formalize the consistency checking problem.

Definition 1 (Pedigree). *A* pedigree *consists of a 4-tuple* $P = \langle V, F, \mathbf{p}, \mathbf{m} \rangle$ *where:*

- V *is a finite, non-empty set of* members *of the pedigree, and* $F \subseteq V$ *is the set of* founders,
- $\mathbf{p}$,$\mathbf{m} : V \setminus F \longrightarrow V$ *are the* paternal *and* maternal functions, *respectively, where*

$$\mathbf{p}(V \setminus F) \cap \mathbf{m}(V \setminus F) = \emptyset$$

  (that is, nobody can be both a mother and a father), and
- $(n, n) \notin (\mathbf{p} \cup \mathbf{m})^+$, *for every* $n \in V$, *where* $(\mathbf{p} \cup \mathbf{m})^+$ *stands for the transitive closure of the binary relation obtained as the union of the graphs of* $\mathbf{p}$ *and* $\mathbf{m}$ *(that is, a member of the pedigree is never its own ancestor).*

The set $N = V \setminus F$ *is usually referred to as the set of* non-founders *of the pedigree.*

The described constraints on a pedigree are introduced to consistently follow the underlying biological model; observe, for instance, how the model makes it impossible for an individual to mate with itself, as no individual in a given pedigree can be both a father and mother. Note, furthermore, that, since the model specifies the sex of an individual only via the paternal and maternal functions, the sex of a "leaf" in a pedigree is not specified. In our examples and constructions, the sex of individuals in a pedigree without offspring will be chosen arbitrarily, as it is immaterial in consistency checking. Our pictorial representation of pedigrees (with associated genotype information) is borrowed from the genetics literature, and has already been introduced in, e.g., Fig. 1-1. That figure represents a pedigree whose founders are individuals 1 and 2, who are respectively the father and the mother of individual 3.

Consistency checking of a pedigree is based on its associated genotype information; intuitively, the pedigree defines the structure of the family relationships that are being modelled, and the genotype information is the data which must be consistent with the structure. We now present a formal genotype model. In what follows, it is always assumed that instances of this genotype model are in the context of a specific gene and pedigree. We also assume a fixed, finite set $\mathcal{A}$ of *alleles* ranged over by $\mathbf{A}$, $\mathbf{B}$ etc.

In the following definition $\mathrm{Two}(\mathcal{A})$ denotes the family of non-empty subsets of $\mathcal{A}$ that contain no more than two elements.

Definition 2 (Genotype Information). Genotype information *is a partial function* $\mathcal{G} : V \hookrightarrow \mathrm{Two}(\mathcal{A})$ *that associates a genotype to (some of) the members of the pedigree. The domain,* $\mathrm{dom}(\mathcal{G})$*, of the function is referred to as the* set of genotyped members *of the pedigree. The genotype information* $\mathcal{G}$ *is* complete *if* $\mathrm{dom}(\mathcal{G}) = V$.

Let $\mathcal{G}$ *and* $\mathcal{G}'$ *be two genotype information. We say that* $\mathcal{G}'$ extends $\mathcal{G}$ *if* $\mathrm{dom}(\mathcal{G})$ *is included in* $\mathrm{dom}(\mathcal{G}')$*, and* $\mathcal{G}$ *and* $\mathcal{G}'$ *coincide over* $\mathrm{dom}(\mathcal{G})$.

Remark 1. Note that in the definition above, a genotype information may be seen as assigning an *unordered* pair of elements to members of the pedigree. This indicates that the phase of the alleles is unknown. If a pedigree member is *homozygous* at a given locus in its genome, i.e., the two alleles at that locus coincide, the function $\mathcal{G}$ returns a singleton set.

In the standard literature on genetics, and in our pictorial representation of pedigrees, the genotype $\{\mathbf{A},\mathbf{B}\}$ is given as the string $\mathbf{AB}$ (or $\mathbf{BA}$). In particular the genotype $\{\mathbf{A}\}$ is given as $\mathbf{AA}$. In the remainder of this paper, we shall use these notations interchangeably without further explanations.

Considering consistency for a specific gene amounts to checking whether the pedigree and the genotype information are consistent according to the Mendelian law of segregation (see page 4). The law of segregation implicitly defines the following constraint on consistent genotype assignments:

Each individual must inherit precisely one allele from each of its parents.

Our order of business will now be to formalize this constraint, and what it means that a genotype information is consistent with respect to a pedigree.

Definition 3 (Consistent Genotype Information).

1. *A complete genotype information $\mathcal{G}$ for a pedigree P is* consistent *with P if, whenever $v \in N$:*
   (a) if $\mathcal{G}(v) = \{\mathbf{A},\mathbf{B}\}$, then either $\mathbf{A} \in \mathcal{G}(\mathbf{p}(v))$ and $\mathbf{B} \in \mathcal{G}(\mathbf{m}(v))$, or $\mathbf{B} \in \mathcal{G}(\mathbf{p}(v))$ and $\mathbf{A} \in \mathcal{G}(\mathbf{m}(v))$;
   (b) if $\mathcal{G}(v) = \{\mathbf{A}\}$, then $\mathbf{A}$ is contained in both $\mathcal{G}(\mathbf{p}(v))$ and $\mathcal{G}(\mathbf{m}(v))$.
2. *A genotype information is* consistent *with P if it can be extended to a complete, consistent genotype information for P.*

In what follows, CONS will denote the problem of checking whether a given genotype information is consistent with respect to a given pedigree.

4 Consistency Checking is NP-complete

Recall that CONS refers to the consistency checking problem with an arbitrary number of alleles. We shall use nCONS to refer to the consistency checking problem for a gene with n possible alleles. Our aim in the remainder of this section will be to show the following result:

Theorem 1. *nCONS ($n \geq 3$) and CONS are NP-complete.*

To prove this theorem, we shall first show that CONS, and thus nCONS ($n \geq 3$), is in NP. We then show that 3CONS is NP-hard.

It is not too hard to see that CONS is in NP. To this end, given any pedigree P with genotype information $\mathcal{G}$, it is sufficient to exhibit a certificate that is verifiable in polynomial time. The certificate for an instance of problem CONS is a complete and consistent genotype information $\mathcal{G}^c$ that extends $\mathcal{G}$ in the sense of Def. 2. To check

the consistency of $\mathcal{G}^c$ we only have to make sure that the conditions in Def. 3(1) are satisfied for each non-founder of the pedigree. This only takes constant time for each non-founder, and thus the whole consistency check takes linear time in the number of non-founders of the pedigree. Note that the complexity of this consistency check is independent of the number of possible alleles, which shows that nCONS is in NP for every n.

Our order of business will now be to show that 3CONS is NP-hard. Note that this is a strong indication that the structural complexity of consistency checking does *not* depend on the number of alleles for a gene, if that number is at least three. We shall stress the importance of this constant number of alleles from a genetic viewpoint later in this section. Our NP-hardness proof for 3CONS is by reduction from 3SAT. The central idea of the proof is that the structure of the pedigree along with the genotype information, generated from some 3SAT instance, mimic the variables and clauses of the input 3SAT instance as closely as possible. The constructed pedigree with genotype information is consistent if, and only if, the 3SAT instance it models is satisfiable.

We recall, for the sake of clarity, that 3SAT is the special case of the satisfiability problem for boolean formulae in which the input formulae are in *conjunctive normal form*, and all of their clauses (i.e., disjunctions of literals) have exactly three literals—where a literal is either a variable or a negated variable. Our aim, in the remainder of this section, is to offer a polynomial time reduction from 3SAT to 3CONS. In fact, it is not too hard to see that, without loss of generality, we can restrict ourselves to considering boolean formulae in conjunctive normal form whose clauses have the form $x \vee y$, $\overline{x} \vee \overline{y}$, $x \vee y \vee z$, or $\overline{x} \vee \overline{y} \vee \overline{z}$, for some distinct variables x, y, z. Indeed, any 3SAT instance can be brought into that form in the following four steps:

1. Remove all clauses containing complementary literals (as they evaluate to true).
2. Replace multiple occurrences of the same literal within a single clause with a single occurrence of the same literal (as $l \vee l = l$, for every literal l).
3. If a clause consists of a single literal, then
   (a) remove all clauses that contain this literal (as it must be assigned the value true) and
   (b) remove all occurrences of its negation in other clauses (as they have to be assigned the value false). If any clause reduces to the empty clause, then we know that there is no assignment that can satisfy the clause, and the formula is not satisfiable.
4. Finally, we put every clause in the formula into one of the forms $x \vee y$, $\overline{x} \vee \overline{y}$, $x \vee y \vee z$, or $\overline{x} \vee \overline{y} \vee \overline{z}$, for some distinct variables x, y, z. This can be done by introducing dummy variables. For instance, a clause of the form $\overline{x} \vee y \vee z$ is replaced with $(\overline{x} \vee \overline{p}) \wedge (y \vee z \vee p)$, for some fresh variable p. (We use a different variable p for each clause.)

It is clear that any instance of 3SAT can be rewritten to the form described above in polynomial time, and that the resulting formula is satisfiable if, and only if, so was the original one.

We are now ready to present our reduction from 3SAT to 3CONS. Let ϕ be an instance of 3SAT. In light of the above discussion, we may assume that ϕ is in conjunctive normal form, and that its clauses have one of the forms $x \vee y$, $\overline{x} \vee \overline{y}$, $x \vee y \vee z$,

or $\overline{x} \vee \overline{y} \vee \overline{z}$, for some distinct variables x, y, z. The construction of a pedigree with associated genotype information from ϕ proceeds in the following three steps:

1. Make variable gadgets for each of the variables in ϕ.
2. Make clause gadgets for each of the clauses in ϕ.
3. Combine the variable gadgets with the clause gadgets, and output the resulting pedigree.

We start by describing the construction of the variable gadgets. In our construction, we shall make use of three alleles, denoted by **A**, **F** and **T**. The alleles **F** and **T** are intended to play the role of "true" and "false" in the 3SAT problem. The third allele **A** is an auxiliary dummy allele used for controlling inheritance patterns.

For each variable x that occurs in ϕ we construct the pedigree P_x as shown in Fig. 1-4. The pedigree P_x consists of three genotyped members, and one ungenotyped individual n_x. The genotype of n_x can, however, be partly inferred by the Mendelian laws, and has the form $x\mathbf{A}$, where the "allelic variable" x takes either the value **F** or **T**. This is indicated by $x\mathbf{A}$ on the figure. Moreover, the allele x associated with the individual n_x is the only possible origin of a **T** or **F** allele that can be inherited further from the inheritance point of P_x. We shall refer to individual n_x in Fig. 1-4, as the *variable individual for* x. The illustration on the left of P_x in Fig. 1-4 shows how the variable gadgets are depicted in larger pedigrees.

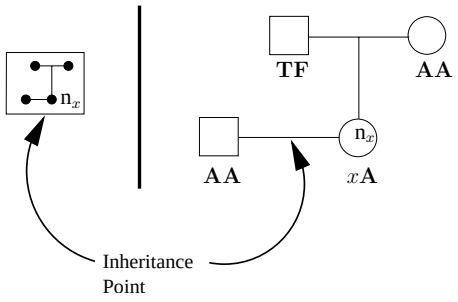

Figure 1-4: The variable gadget P_x that is used in the proof showing that 3CONS is NP-complete.

The next step in the reduction is to construct a clause gadget P_γ, for each clause γ in the formula ϕ. As we have already pointed out, there are only four different types of clauses we need to consider, and each leads to a different type of clause gadget. The four different types of clause gadget are depicted in Fig. 1-5.

In the following lemma we shall describe the connection between the consistency of the clause gadgets, and the satisfiability of the clauses they model. We assume that the only clauses that occur are of the form described above. Furthermore we assume the notion of substitution of truth values both for variables in the clauses, and for the "allelic variables" that occur in the gadgets. The lemma is proven by a simple case inspection, and is left to the reader.

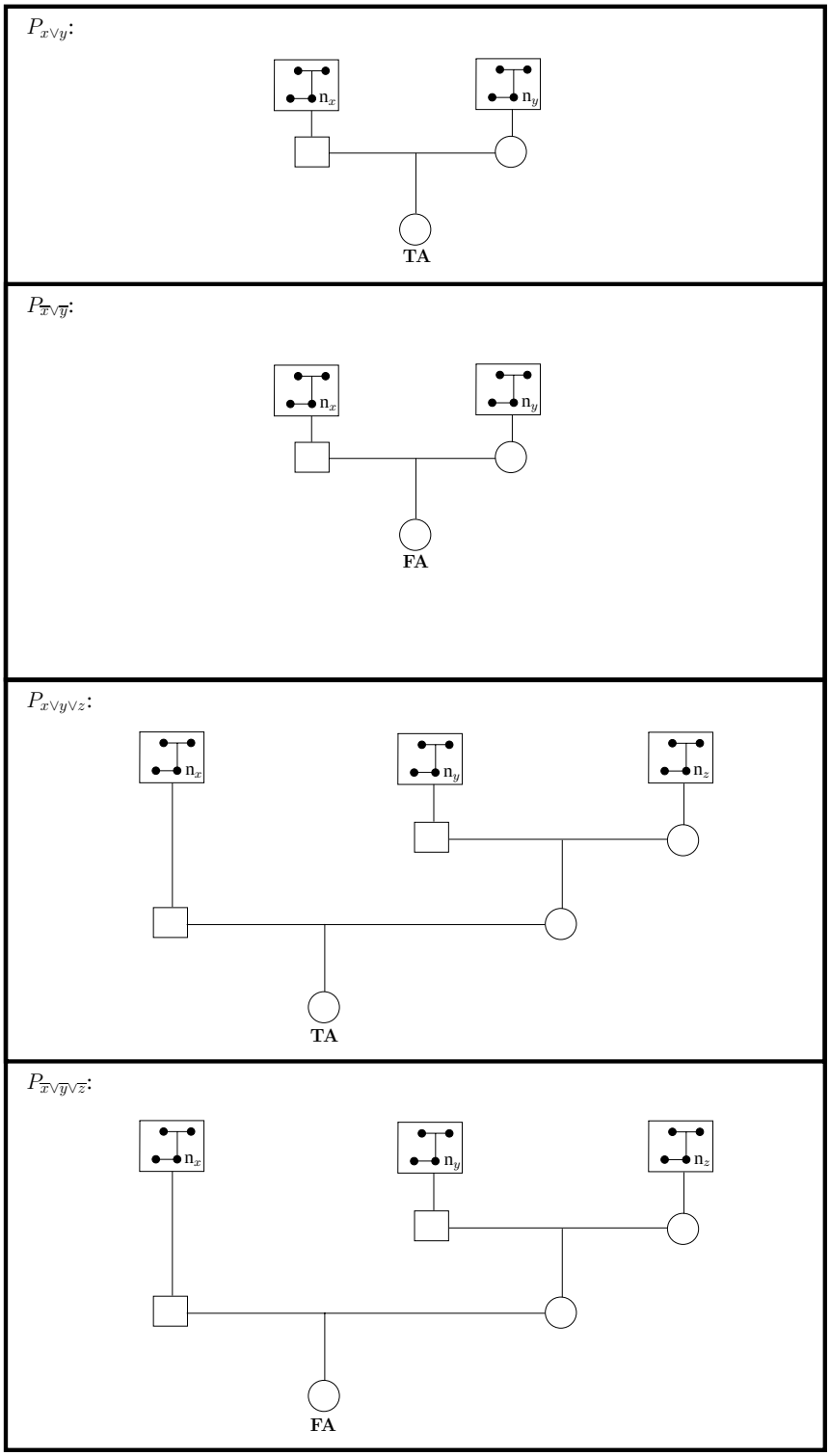

Figure 1-5: The pedigrees constructed for the four basic clause types along with their connections with the appropriate variable gadgets. Notice the symmetry between $P_{x \vee y}$ and $P_{\overline{x} \vee \overline{y}}$, and $P_{x \vee y \vee z}$ and $P_{\overline{x} \vee \overline{y} \vee \overline{z}}$, respectively.

Lemma 1. *Let γ be a clause, and let ρ be a variable assignment for this clause. Then $\gamma\rho$ evaluates to true iff $P_\gamma\rho$ is consistent.*

Having constructed a variable gadget for each variable and a clause gadget for each clause occurring in ϕ, we combine these gadgets as shown in Fig. 1-6, and output the resulting pedigree P_ϕ.

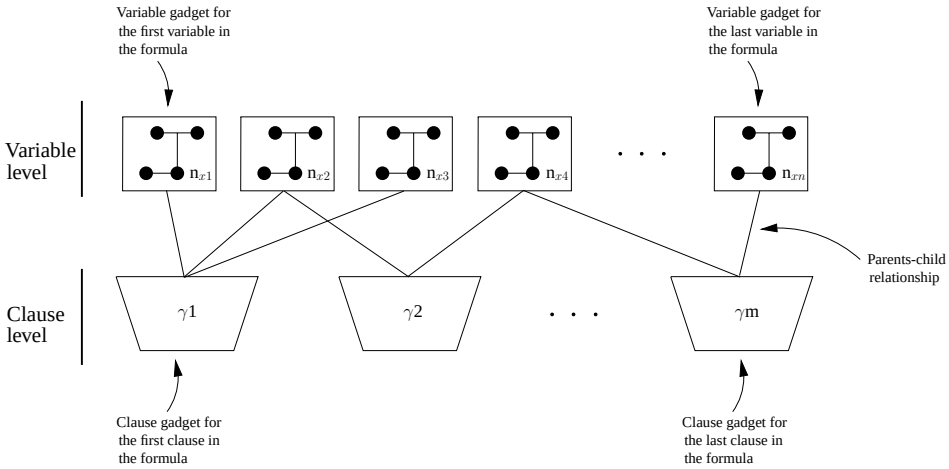

Figure 1-6: The general form of the pedigree constructed in the reduction from 3SAT. Notice that there exists a one to one correspondence between the number of variables and clauses in ϕ, and the number of variable gadgets and clause gadgets, respectively, in the constructed pedigree.

Example 1. Let ϕ be the formula

$$(x \vee u) \wedge (\overline{y} \vee \overline{u}) \wedge (x \vee y) \wedge (\overline{x} \vee \overline{y}) \ . \tag{1-1}$$

The pedigree produced from this formula by the construction described above is depicted in Fig. 1-7.

The following result states the correctness of our construction of P_ϕ from a 3SAT formula ϕ.

Lemma 2. *Let ϕ be a 3SAT formula, and let ρ be a variable assignment for this formula. Then $\phi\rho$ evaluates to true iff $P_\phi\rho$ is consistent. In particular, ϕ is satisfiable iff P_ϕ is consistent.*

Since the pedigree P_ϕ can be constructed in polynomial time from the formula ϕ, the lemma above allows us to conclude that 3CONS is NP-hard, and the proof of Thm. 1 is now complete.

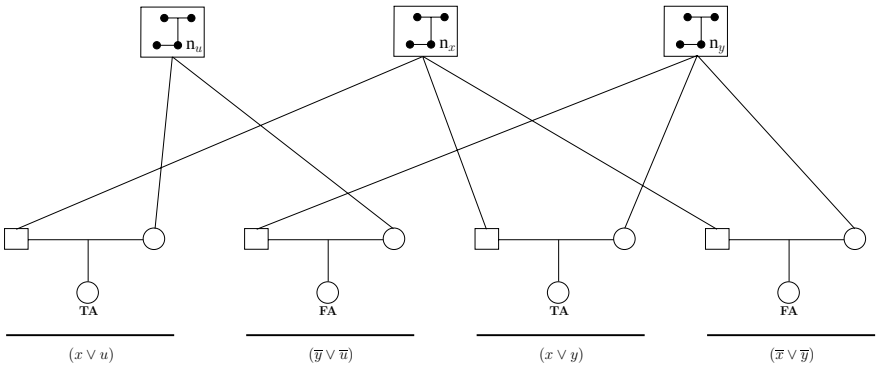

Figure 1-7: The pedigree for the formula (1-1). Note that formula (1-1) is satisfiable, and that the above pedigree is consistent.

The reference [8] also offers a reduction from 3SAT to CONS that uses a number of alleles that is linear in the number of variables in the input 3SAT instance. This reduction, albeit possibly conceptually simpler than the one presented here, is not reasonable from a genetic standpoint because the maximum number of alleles that can be expected for a gene is roughly 100. Furthermore, although inbreeding does occur often in the animal kingdom, it would still be critical from a genetic perspective if our modelling of 3SAT using pedigrees were based on severe inbreeding, and we have striven to avoid this problem in our constructions. (The loops that arise in the pedigrees resulting from our reductions are called *marriage loops* in the pedigree literature—see, e.g., [16]—, and are natural in real life pedigrees.) The question is whether our other modelling assumptions are fair in light of the biological knowledge on consistency checking, e.g., the amount and form of genotype information in the real world. We now discuss these aspects by analyzing the pedigree structure and the genotyped individuals produced by the reduction outlined above.

Number of Offspring The points were a large number of children from a single couple can occur are the inheritance points of the variable individuals. Every occurrence of the same variable implies that a child is constructed from the inheritance point. Theoretically, the reduction requires an arbitrary number of children from a single couple. Although it can be argued from a complexity theoretic perspective that it is equally complex to satisfy a formula in conjunctive normal form where at most three occurrences of a single variable are allowed (see, e.g., [15, Propn. 9.3]), it is also possible to argue strongly on the subject from a biological viewpoint. Two arguments can be brought forth, the first regarding an expansion of the structure, and the second regarding the gender of the variable individuals. First, we have argued in [8] that it is possible to model a variable gadget in such a way that no more than fifteen children are needed in the reduction. Second, it is theoretically possible for male individuals to have a large number of children, but with different women (the women still have an upper bound on the number of offspring). This naturally requires that the variable individuals be males.

Genotype History The aforementioned reduction from 3SAT to 3CONS requires genotype information five generations back. In the case of species with a lifespan of up to five years this does not seem an unreasonable assumption. But for humans and animals with a long lifespan, it can be doubtful whether such data exist. Although it can be argued that it is just a matter of time before such genotype history does exist for humans, we have shown in [8] that it is possible to perform a reduction where there is only genotype information for the individuals in the youngest generation of a pedigree, at the price of using a larger number of dummy alleles.

As already remarked in Sect. 1, our reduction from 3SAT to 3CONS employs looping pedigrees. The following result offers strong evidence that this is most likely necessary.

Theorem 2. *Checking the consistency of non-looping pedigrees can be performed in polynomial time.*

5 Further Results

In this section we discuss briefly three new problems related to CONS motivated by the underlying biology, and study their computational complexity.

Tolerance to Critical Genotypes Assume that it is revealed that some application of pedigrees with genotype information is tolerant to a specific number, say k, of critical genotypes in the genotype information. (A *critical genotype* is genotype information on an individual that, if removed, would make an inconsistent pedigree with genotype information consistent.) We denote the problem of deciding whether there are k critical genotypes in a CONS instance as kCRIT. Since it is possible to reduce any CONS instance to a kCRIT instance in polynomial time, we have that:

Theorem 3. *In the presence of at least three alleles, kCRIT is NP-complete for every $k \geq 0$.*

Consistency Checking with Two Alleles According to [17, p. 274], *single nucleotide polymorphisms* are utilized markers where two alleles exist. Consistency checking of such data amounts to the problem 2CONS. A relevant question is whether 2CONS is also NP-complete or whether it is polynomial time decidable. Three is often a magic number, when it comes to the structural complexity of a computational problem. For instance, 3COLORING and 3SAT are NP-complete, while 2COLORING and 2SAT are polynomial time decidable (see, e.g., [15, pp. 185 and 198]). The same holds for consistency checking in light of the following result:

Theorem 4. *2CONS is polynomial time decidable.*

Phase Known Consistency Checking In this paper, we have focused on consistency checking in a phase unknown setting. We now turn our focus to the task of consistency checking where the phase of the genotype information is known. The motivation for this type of investigation is that it is sometimes possible to infer the identity of the parent from whom some allele originated (and thereby also the origin of the other allele).

Definition 4. *A* phase known genotype information *is a partial function* $\mathcal{G}^p : V \hookrightarrow \mathcal{A} \times \mathcal{A}$*. The genotype information* $\mathcal{G}^p$ *is* complete *if* $\mathrm{dom}(\mathcal{G}^p) = V$.

A complete, phase known genotype information $\mathcal{G}^p$ *for a pedigree* P *is* consistent *with* P *if whenever* $v \in N$ *and* $\mathcal{G}^p(v) =$**(A,B)***, then* $\mathbf{A}$ *is one of the components of* $\mathcal{G}(\mathbf{p}(v))$*, and* $\mathbf{B}$ *is one of the components of* $\mathcal{G}(\mathbf{m}(v))$.

A phase known genotype information is consistent *with* P *if it can be extended to a complete and consistent phase known genotype information for* P.

Let PCONS be the problem of consistency checking a pedigree with phase known genotype information. We can argue that every CONS instance can be reduced in polynomial time to a PCONS instance that is consistent if, and only if, so was the original CONS instance. Since PCONS is easily seen to be in NP, we thus have that:

Theorem 5. *In the presence of at least three alleles, PCONS is NP-complete.*

Acknowledgments We thank Mogens Nielsen for the initial inspiration on a possible reduction showing that consistency checking is NP-complete, and Emmanuel Fleury for fruitful discussions on the topic of this paper.

References

1. G. R. ABECASIS, S. S. CHERNY, W. O. COOKSON, AND L. R. CARDON, *Merlin: Rapid analysis of dense genetic maps using sparse gene flow trees*, Nature Genetics, 30 (2002), pp. 97–101.
2. DECODE NEWS CENTER, `http://www.decode.com/news/releases/`, November 2001.
3. J. DOHRMANN, *Mænd vil vide, hvem de er far til*, Nordjyske, (2002). In Danish.
4. J. C. P. ERIC SOBEL AND K. LANGE, *Detection and integration of genotyping errors in statistical genetics*, American Journal of Human Genetics, 70 (2002), pp. 496–508.
5. D. F. GUDBJARTSSON, *Multipoint Linkage Analysis Based on Allele Sharing Models*, PhD thesis, Institute of Statistics and Decision Sciences, Duke University, 2000.
6. D. F. GUDBJARTSSON, K. JONASSON, AND C. A. KONG, *Fast multipoint linkage calculation with Allegro*, Nature Genetics, 20 (2000), pp. 12–13.
7. M. HAMER, *Back to your roots*, New Scientist, 2334 (2002), pp. 33–36.
8. J. A. HANSEN, J. JOHNSEN, AND J. KNUDSEN, *Computational complexity of consistency checking*, Master's thesis, Department of Computer Science, Aalborg University, June 2002. Available at `http://www.cs.auc.dk/~luca/PAPERS/hjk02.ps.gz`.
9. L. B. JORDE, J. C. CAREY, M. J. BAMSHAD, AND R. L. WHITE, *Medical Genetics*, Mosby, 1999.
10. W. S. KLUG AND M. R. CUMMINGS, *Concepts of Genetics*, Prentice Hall, 5th ed., 1997.
11. L. KRUGLYAK, M. J. DALY, M. P. REEVE-DALY, AND E. S. LANDER, *Parametric and nonparametric linkage analysis: A unified multipoint approach*, American Journal of Human Genetics, 58 (1996), pp. 1347–1363.
12. J. R. O'CONNELL AND D. E. WEEKS, *Pedcheck: A program for identification of genotype incompatibilities in linkage analysis*, American Journal of Human Genetics, 63 (1998), pp. 259–266.
13. ———, *An optimal algorithm for automatic genotype elimination*, American Journal of Human Genetics, 65 (1999), pp. 1733–1740.

14. J. OTT, *Analysis of Human Genetic Linkage*, The Johns Hopkins University Press, 3rd ed., 1999.
15. C. H. PAPADIMITRIOU, *Computational Complexity*, Addison Wesley, 1995.
16. A. PICCOLBONI AND D. GUSFIELD, *On the complexity of fundamental computational problems in pedigree analysis*, Tech. Rep. CSE-99-8, Computer Science Department, University of California, Davis, September 1999.
17. T. STRACHAN AND A. P. READ, *Human Molecular Genetics 2*, Wiley-Liss, 1999.

Recent BRICS Report Series Publications

RS-02-42 **Luca Aceto, Jens A. Hansen, Anna Ingólfsdóttir, Jacob Johnsen, and John Knudsen.** ***Checking Consistency of Pedigree Information is NP-complete (Preliminary Report).*** **October 2002. 16 pp.**

RS-02-41 **Stephen L. Bloom and Zoltán Ésik.** ***Axiomatizing Omega and Omega-op Powers of Words.*** **October 2002. 16 pp.**

RS-02-40 **Luca Aceto, Willem Jan Fokkink, and Anna Ingólfsdóttir.** ***A Note on an Expressiveness Hierarchy for Multi-exit Iteration.*** **September 2002. 8 pp.**

RS-02-39 **Stephen L. Bloom and Zoltán Ésik.** ***Some Remarks on Regular Words.*** **September 2002. 27 pp.**

RS-02-38 **Daniele Varacca.** ***The Powerdomain of Indexed Valuations.*** **September 2002. 54 pp. Short version appears in Plotkin, editor, *Seventeenth Annual IEEE Symposium on Logic in Computer Science*, LICS '02 Proceedings, 2002, pages 299–308.**

RS-02-37 **Mads Sig Ager, Olivier Danvy, and Mayer Goldberg.** ***A Symmetric Approach to Compilation and Decompilation.*** **August 2002. To appear in Neil Jones's Festschrift.**

RS-02-36 **Daniel Damian and Olivier Danvy.** ***CPS Transformation of Flow Information, Part II: Administrative Reductions.*** **August 2002. 9 pp. To appear in the *Journal of Functional Programming*. This report supersedes the earlier BRICS report RS-01-40.**

RS-02-35 **Patricia Bouyer.** ***Timed Automata May Cause Some Troubles.*** **August 2002. 44 pp.**

RS-02-34 **Morten Rhiger.** ***A Foundation for Embedded Languages.*** **August 2002. 29 pp.**

RS-02-33 **Vincent Balat and Olivier Danvy.** ***Memoization in Type-Directed Partial Evaluation.*** **July 2002. 18 pp. To appear in Batory and Consel, editors, *ACM SIGPLAN/SIGSOFT Conference on Generative Programming and Component Engineering*, GPCE '02 Proceedings, LNCS, 2002.**