A

Belief merging by examples

PAOLO LIBERATORE, Sapienza University of Rome

A common assumption in belief revision is that the reliability of the information sources is either given, derived from temporal information, or the same for all. This article does not describe a new semantics for integration but studies the problem of obtaining the reliability of the sources given the result of a previous merging. As an example, corrections performed manually on the result of merging some databases may indicate that the relative reliability of their sources is different from what previously assumed, helping subsequent data mergings.

CCS Concepts: •**Computing methodologies** → **Nonmonotonic, default reasoning and belief revision;**

Additional Key Words and Phrases: Belief merging, reliability estimation

1. INTRODUCTION

When integrating information coming from different sources, a distinction is made between revision [Gärdenfors 1988; Darwiche and Pearl 1997; Jin and Thielscher 2007; Peppas 2008; Delgrande 2012] (new information more reliable than old) and merging [Liberatore and Schaerf 1998; Chopra et al. 2006; Konieczny and Pérez 2011] (same reliability). More generally, priorities or weights are assigned to the sources to indicate their reliability [Nebel 1992; 1998; Rott 1993; Delgrande et al. 2006]. *Measures* and *aggregation functions* allow for fine-grained policies of integration [Konieczny et al. 2004; Everaere et al. 2010; Konieczny and Pérez 2011]. Families of operators are then defined, all depending in a way or another from the relative reliability of the sources. The two basic cases of non-iterated revision and merging result from giving priority to the new information or the same to all pieces of information to be incorporated, respectively. The strength of information sources has been studied in the field of cognitive psychology, where it was determined to depend on the order in which the information is given [Wang et al. 2000], on the size of the group generating it [Mannes 2009] and other social factors [See et al. 2011].

The first time merging is done, the relative reliability of the pieces of information to be integrated cannot come other than from sources external to the merging process. However, subsequent mergings may then take advantage of the previous results. The following example shows such a situation.

Example 1.1. Three different databases related to the same domain are to be merged. In lack of information about their relative reliability, this operation is done assuming them equally reliable. The result is then checked and found out to be inconsistent. The programmers go over the database and fix the problems using direct knowledge of the domain. The resulting database is later found out to be equal to that obtained by merging under the assumption that the second database is more reliable

---

Author's address: DIAG, Sapienza University of Rome, Via Ariosto 25, 00185 Rome, Italy. Email: paolo@liberatore.org

 1529-3785/YYYY/01-ARTA $15.00
DOI: http://dx.doi.org/10.1145/0000000.0000000

than the first and the third. When other data comes from the same sources, this information is used to merge it.

The lack of meta-information regarding reliability shows up in other contexts. For example, the reliability of some sensors may depend on where they are used (e.g., some are reliable indoor but mostly useless outdoor); an ordering among them may be obtained when other data confirms or contradicts the result of merging their output.

The problem considered in this article is to estimate the reliability of formulae $K_1, \ldots, K_m$ so that their integration produces a given other formula R. Contrary to most work in belief revision, no new semantics for merging are introduced, and this is because the point is not how to obtain R from $K_1, \ldots, K_m$, but how to reckon the reliability of $K_1, \ldots, K_m$ from R. This formula R is given, not the outcome of the process: it is the corrected merged database in the first example and the information compared to the result of merging data coming from the sensors in the second.

— two sources provide a and $\neg a \wedge b$; lacking information about their reliability, the result is the disjunction $a \vee (\neg a \wedge b) = \mathsf{true}$;
— the actual state of the world is detected to be $\neg a \wedge b$;
— this formula $\neg a \wedge b$ is the result of merging a and $\neg a \wedge b$ when the source of the second is assumed more reliable;
— other two formulae $a \wedge c$ and $b \wedge \neg c$ arrive from the same sources; since the second is more reliable, merging produces $b \wedge \neg c$.

The procedure looks straightforward because it involves only two very simple formulae under a trivial semantics of merging by taking either one of them or their disjunction, depending on their relative reliability. If none of these possible outcomes coincide with the given formula then one may (more details are in Section 5):

(1) assume that R is not equal to the expected result of merging but a "more precise" formula, or that it represents incomplete information;
(2) take into account that some sources produce reliable information on some aspects of the domain and unreliable in others, so they may be split for example on the variables;
(3) check whether the result can be obtained using a different method of integration.

The present article analyzes the problem for two existing merging semantics: minimal sum of distances [Konieczny and Pérez 2011; Konieczny et al. 2002; 2004] and prioritized base merging [Nebel 1992; 1998; Rott 1993], also called discrimin merging [Delgrande et al. 2006]. However, any other of the several existing merging semantics can be used [Konieczny and Pérez 2011; Delgrande et al. 2006]. These two has been chosen not only because they are of interest by themselves, but also because they are at the extreme opposite of the range of merging semantics: the first one is based on a numeric evaluation of the distance of models from the knowledge bases; the second hinges on formulae rather than models and employs a notion of priority that is qualitative rather than quantitative.

For merging based on sums of distances [Konieczny and Pérez 2011; Konieczny et al. 2002; 2004], a necessary and sufficient condition for R to be the result of merging K_1 and K_2 with some weights is given. This result allows to easily derive upper bounds on the complexity of obtainability, which is in Π^p_{i+1} whenever checking distance is in Π^p_i or in Σ^p_i. This implies that the problem is in coNP for the drastic distance and in Π^p_2 for the Hamming distance. Hardness for these cases is proved. A tractable case for the Hamming distance is determined. Using the same necessary and sufficient condition, a local search algorithm for determining the weights is shown.

Prioritized base merging [Nebel 1992; 1998; Rott 1993] depends on the *maxsets* of the formulae $K_1, \ldots, K_m$, which are the maximally consistent subsets of them. The properties proved for this semantics are: some formulae R cannot be obtained from $K_1, \ldots, K_m$; this may happen even if R is the disjunction of some of their maxsets; however, this is impossible if $m \leq 3$; some formulae R can be obtained as the result of merging only if the priority ordering has at least n classes, and this holds for ever n; if the maxsets form a Berge-acyclic graph and R is the disjunction of some of them, then R is always obtainable; an algorithm for producing the priority ordering in this case is given.

If all maximally consistent subsets have size two or less the problem becomes a problem on graphs, where weights are to be assigned to nodes in such a way some edges are selected and some other are excluded. This belief revision problem requires a (quite long) argument on graph transformations to obtain a simple necessary and sufficient condition: non-obtainability is the same as the presence of *alternating* cycles of edges.

Surprisingly, complexity turns out not to be higher than that of computing the result of merging [Eiter and Gottlob 1992; 1996; Liberatore 1997a; 1997b; Nebel 1998; Liberatore and Schaerf 2001; Delgrande et al. 2013], at least in some cases. For example, given a consistent R and $K_1, \ldots, K_m$ with constant m or with maximally consistent subsets of size two or less, checking whether R is obtainable is only coNP-complete, thus solvable within a reasonable size of formulae by modern SAT-solvers.

The article is organized as follows: a section introduces the basic settings, the following the definitions and results using the sums of distances and prioritized base merging, respectively, including an algorithm each. Then, the question of what to do if a given formula is not obtainable is considered. A final section draws some conclusions.

2. PRELIMINARIES

The knowledge bases to be merged are denoted by $K_1, \ldots, K_m$ throughout this article. They are assumed to be consistent propositional formulae. The same for the expected result R, unless explicitly indicated otherwise.

Two merging semantics are considered in this article, the first based on the weighted sum of distances, the second on a priority ordering. Formula R is *obtainable* from $K_1, \ldots, K_m$ if it is the result of merging these formula with some weights or priorities. Using the first semantics, this amounts to checking the existence of weights such that R is the result of merging $K_1, \ldots, K_m$ with these weights. For the second semantics, the definition is the same with a priority ordering instead of the weights.

Obtainability means that R is the result of merging $K_1, \ldots, K_m$ with some relative reliability among these knowledge bases. Determining this reliability ordering is the aim of two algorithms, one for each of the considered merging semantics. What to do if R is not obtainable is considered in Section 5.

3. WEIGHTED SUM

Model-based merging operators [Konieczny and Pérez 2011; Konieczny et al. 2002; 2004] work from a measure of the distance between models, selecting only the ones that are at minimal total distance from the knowledge bases. Different semantics result from different distances measures and different methods for combining them. Two measures of interest are [Konieczny et al. 2002; Revesz 1997; Lin and Mendelzon 1999]:

Drastic distance: $d(I, I) = 0$, $d(I, J) = 1$ if $J \neq I$;
Hamming distance: $d(I, J)$ is the number of variables evaluated differently by I and J.

Distance measures extend to knowledge bases: $d(I, K)$ is the minimal value of $d(I, J)$ for $J \models K$. The drastic distance from a model to a knowledge base is therefore 0 if the model satisfies the base and 1 otherwise. The Hamming distance is the minimal number of variables that are assigned different values by the model and by a model of the knowledge base.

Distances can be further extended from one to more knowledge bases in various ways. One is to define $d(I, K_1, \ldots, K_m)$ to be the sum of the distances $d(I, K_i)$; other methods exist [Konieczny and Pérez 2011]. If the sources of the knowledge base differ in reliability, a weighted sum can be used in place of the sum [Konieczny et al. 2002; 2004]. Let $\{w_1, \ldots, w_m\}$ be the weights, which are assumed positive integers (null, negative or real values can also be of interest, but are not considered in this article). The weighted distance from I to $\{K_1, \ldots, K_m\}$ is:

$$d(I, K_1, \ldots, K_m) = \sum_{1 \leq i \leq m} w_i \times d(I, K_i)$$

Alternatively, the *distance vector* of I is the array $(d(I, K_1), \ldots, d(I, K_m))$ and the weighted distance is obtained by multiplying it with the weight vector $(w_1, \ldots, w_m)$. Either way, merging selects the models of minimal weighted distance from the knowledge bases [Revesz 1997; Lin and Mendelzon 1999; Konieczny et al. 2002; 2004].

The problem of obtainability is that of finding positive integers $w_1, \ldots, w_m$ such that the result of merging $K_1, \ldots, K_m$ is a given formula R. As usual, the complexity analysis is done on the decision version of this problem, that of checking the existence of such weights. The algorithm in Section 3.2 searches for actual values. Some considerations on what to do if they do not exist are in Section 5.

The following restriction is considered in this section: two knowledge bases only. In other words, $m = 2$. The knowledge bases are K_1 and K_2 only. This restriction simplifies the definition of d to:

$$d(I, K_1, K_2) = w_1 \times d(I, K_1) + w_2 \times d(I, K_2)$$

For every model I, its distance vector from $\{K_1, K_2\}$ is $(d(I, K_1), d(I, K_2))$.

Obtainability is the existence of weights that produce the given result R. Weights $(1, 2)$ produce the same results as $(2, 4)$: the weighted distance of the first pair is double that of the second for every model; therefore, the minimal models are the same. Instead of a pair of weights w_1 and w_2 it suffices to search for the value of their ratio $\frac{w_1}{w_2}$. This is a simpler problem because such a value can be obtained by simple algebraic manipulation from two models of R in most cases. Otherwise, some constraints on its value derive from models of $\neg R$.

The following expression is useful for relating models, as it often coincides with $\frac{w_1}{w_2}$ if I and J both satisfy R and gives a bound to this fraction if I does and J does not.

$$p(I, J; K_1, K_2) = \frac{d(J, K_2) - d(I, K_2)}{d(I, K_1) - d(J, K_1)}$$

Since the two knowledge bases are always K_1 and K_2 in this section, $p(I, J; K_1, K_2)$ is shortened to $p(I, J)$. The value of $p(I, J)$ can be used to determine whether some models can be in the result of merging and some others cannot. The formal conditions are shown in a following theorem, but an example may in the meantime help clarifying how it is useful.

Let R be a formula having models I, J and L but not M, and let the distance from each of the models I, J, L and M to the two knowledge bases be as in the following table:

	K_1	K_2
I	1	5
J	2	3
L	3	1
M	4	2

By definition, p takes the following values:

$$\begin{aligned} p(I,J) &= \frac{3-5}{1-2} = \frac{-2}{-1} = 2 \\ p(I,L) &= \frac{1-5}{1-3} = \frac{-4}{-2} = 2 \\ p(I,M) &= \frac{2-5}{1-4} = \frac{-3}{-3} = 1 \end{aligned}$$

A necessary condition for R being obtainable is that p has the same value for all pairs of models of R, like $p(I,J) = p(I,L)$. Another necessary condition is that p is different if a model of a pair does not satisfy R, like $p(I,M) \neq p(I,J)$. These are only necessary conditions, the full sufficient and necessary condition is in Theorem 3.1.

PROPERTY 1. *Two models I and J have the same distance from $\{K_1, K_2\}$ weighted by w_1 and w_2 if and only if either $d(I,K_1) = d(J,K_1)$ and $d(I,K_2) = d(J,K_2)$ or $d(I,K_1) \neq d(J,K_1)$ and $\frac{w_1}{w_2} = p(I,J)$.*

Proof. The distance from I and J to K_1 and K_2 weighted by w_1 and w_2 is:

$$\begin{aligned} d(I,K_1,K_2) &= w_1 \times d(I,K_1) + w_2 \times d(I,K_2) \\ d(J,K_1,K_2) &= w_1 \times d(J,K_1) + w_2 \times d(J,K_2) \end{aligned}$$

If these amounts coincide, then:

$$\begin{aligned} w_1 \times d(I,K_1) + w_2 \times d(I,K_2) &= w_1 \times d(J,K_1) + w_2 \times d(J,K_2) \\ w_1 \times (d(I,K_1) - d(J,K_1)) &= w_2 \times (d(J,K_2) - d(I,K_2)) \end{aligned}$$

This equation is true if $d(I,K_1) = d(J,K_1)$ and $d(I,K_2) = d(J,K_2)$. Otherwise, both sides can be divided by $d(I,K_1) - d(J,K_1)$ and by w_2, which by assumption is larger than zero, obtaining:

$$\frac{w_1}{w_2} = \frac{d(J,K_2) - d(I,K_2)}{d(I,K_1) - d(J,K_1)}$$

The right-hand side of this equation is $p(I,J)$. □

This property expresses a condition for I and J to have the same weighted distance from the knowledge bases. If R is the result of merging with weights w_1 and w_2, this condition holds for every two models I and J of R. In particular, I, J and L satisfy the result of merging only if $p(I,J)$ and $p(I,L)$ both coincide with $\frac{w_1}{w_2}$, which implies $p(I,J) = p(I,L)$. This could be checked by first calculating $p(I,J) = \frac{w_1}{w_2}$ and then checking whether every other $p(I,L)$ coincides with this value.

PROPERTY 2. *Model I is closer than model M to $\{K_1, K_2\}$ with weights w_1 and w_2 if and only if either:*

— *$d(I,K_1) = d(M,K_1)$ and $d(I,K_2) < d(M,K_2)$; or*

— $d(I, K_1) - d(M, K_1) > 0$ *and* $\frac{w_1}{w_2} < p(I, M)$*; or*
— $d(I, K_1) - d(M, K_1) < 0$ *and* $\frac{w_1}{w_2} > p(I, M)$.

Proof. The distance is $w_1 \times d(I, K_1) + w_2 \times d(I, K_2)$ for I and $w_1 \times d(M, K_1) + w_2 \times d(M, K_2)$ for M. Therefore, I is closer than M to $\{K_1, K_2\}$ if:

$$\begin{aligned} w_1 \times d(I, K_1) + w_2 \times d(I, K_2) &< w_1 \times d(M, K_1) + w_2 \times d(M, K_2) \\ w_1 \times (d(I, K_1) - d(M, K_1)) &< w_2 \times (d(M, K_2) - d(I, K_2)) \end{aligned}$$

By assumption, w_2 is strictly positive. Therefore, both sides of this inequation can be divided by it. Instead, $d(I, K_1) - d(M, K_1)$ may be positive, negative or zero. In latter case, $d(I, K_1) = d(M, K_1)$, which implies that I is closer than M to the bases if and only if $d(I, K_2) < d(M, K_2)$, regardless of the weights.

If $d(I, K_1) - d(M, K_1)$ is positive, both sides of the inequation can be divided by it:

$$\frac{w_1}{w_2} < \frac{d(M, K_2) - d(I, K_2)}{d(I, K_1) - d(M, K_1)} \text{ if } d(I, K_1) - d(M, K_1) > 0$$

The inequation is $\frac{w_1}{w_2} < p(I, M)$. In the other case, dividing both sides by the negative number $d(I, K_1) - d(M, K_1)$ changes $<$ into $>$:

$$\frac{w_1}{w_2} > \frac{d(M, K_2) - d(I, K_2)}{d(I, K_1) - d(M, K_1)} \text{ if } d(I, K_1) - d(M, K_1) < 0$$

The inequation is $\frac{w_1}{w_2} > p(I, M)$. $\square$

These properties show that most pairs of models constrain the value of $\frac{w_1}{w_2}$. In particular, two models of R are enough to uniquely fix it, unless they are at the same distance from K_1. Models that do not satisfy R only generate inequations. If there are at least two models of R at different distances from K_1 this is not a problem, as they determine $\frac{w_1}{w_2}$ and what is left to do is check the inequations.

Otherwise, more complex constraints among models not satisfying R may result. As an example, if all models of R are at distance $(4, 4)$ and two models not of R at distance $(1, 8)$ and $(8, 1)$, then R is obtainable with $w_1 = w_2 = 1$. Two other models not in R at distance $(1, 5)$ and $(5, 1)$ make R unobtainable.

If I, J, L are all models of R, then both $p(I, J)$ and $p(I, L)$ coincide with $\frac{w_1}{w_2}$, and therefore coincide with each other: $p(I, J) = p(I, L)$. For the same reason, if $I \models R$, $J \models R$ and $L \not\models R$, then $p(I, J) < p(I, L)$ or $p(I, J) > p(I, L)$, depending on the sign of $d(I, K_1) - d(L, K_1)$.

These constraints are enough if R has at least two models with differing distance from K_1. Otherwise, R does not set a value for $\frac{w_1}{w_2}$, which can therefore be varied to exclude models not satisfying R. In particular, two inequations of opposite comparison can be combined: if $I \models R$, $M \not\models R$, $N \not\models R$, $d(I, K_1) - d(N, K_1) > 0$ and $d(I, K_1) - d(M, K_1) < 0$, then $\frac{w_1}{w_2} < p(I, N)$ and $\frac{w_1}{w_2} > p(I, M)$, leading to $p(I, M) < p(I, N)$.

THEOREM 3.1. *A satisfiable formula R is obtainable from $\{K_1, K_2\}$ if and only if for all models I, J, L that satisfy R and all M, N that do not, the following conditions hold:*

(1) *if* $d(I, K_1) \geq d(J, K_1)$ *then* $d(I, K_2) \leq d(J, K_2)$
(2) *if* $d(I, K_1) \geq d(M, K_1)$ *then* $d(I, K_2) < d(M, K_2)$
(3) $p(I, J) = p(I, L)$
if $d(I, K_1) - d(J, K_1) \neq 0$ *and* $d(I, K_1) - d(L, K_1) \neq 0$

(*4*) $p(I, J) < p(I, M)$
if $d(I, K_1) - d(J, K_1) \neq 0$ *and* $d(I, K_1) - d(M, K_1) > 0$
(*5*) $p(I, J) > p(I, M)$
if $d(I, K_1) - d(J, K_1) \neq 0$ *and* $d(I, K_1) - d(M, K_1) < 0$
(*6*) $p(I, N) < p(I, M)$
if $d(I, K_1) - d(M, K_1) > 0$ *and* $d(I, K_1) - d(N, K_1) < 0$

Proof. Assuming the conditions true, we derive values of w_1 and w_2 that make the result of merging being exactly R. Two cases are possible: in the first, all models of R have the same distance to K_1 and the same distance to K_2; in the second, at least two models of R have different distances.

If all models of R are at the same distance from K_1 and from K_2, then every pair of weights makes them having the same weighted distance. Therefore, the problem is only with models not satisfying R, which must be at a greater distance. Let I, M and N be:

— I is a model of R;
— M is one of the models not satisfying R with a minimal value of $p(I, M)$ among the ones with $d(I, K_1) - d(M, K_1) > 0$, if any;
— N is one of the models not satisfying R with a maximal value of $p(I, N)$ among the ones with $d(I, K_1) - d(N, K_1) < 0$, if any.

By the sixth condition of the lemma, under these conditions $p(I, N) < p(I, M)$. If $\frac{w_1}{w_2}$ is between $p(I, N)$ and $p(I, M)$, then $\frac{w_1}{w_2}$ is smaller than $p(I, M')$ for every $M' \not\models R$ with $d(I, K_1) - d(M', K_1) > 0$, thanks to the minimality of M. By Property 2, this implies that M' is further from the bases than I. The same applies to models N' with $d(I, K_1) - d(N', K_1) < 0$, thanks to the maximality of N. For the models L such that $d(I, K_1) - d(L, K_1) = 0$, the second condition of the lemma implies that $d(I, K_2) < d(L, K_2)$, proving that they are further from the bases than I regardless of the weights.

If no such M or no such N exist, the corresponding constraint is void. This can be formalized by replacing $p(I, N)$ with 0 and $p(I, M)$ with the number of the variables.

If $p(I, N)$ is negative, $\frac{w_1}{w_2}$ is determined as follows. Since M is such that $d(I, K_1) - d(M, K_1) > 0$, it holds $d(I, K_1) > d(M, K_1)$. By the second condition of the lemma $d(I, K_2) < d(M, K_2)$, which ensures that $p(I, M)$ is strictly positive. By definition of this expression, its minimal positive value is $\frac{1}{n}$, obtained by taking the minimal value of the numerator (1 or -1) and the maximal value of the denominator (n or $-n$). Since $p(I, N)$ is negative, a value between it and $\frac{1}{n}$ is $\frac{1}{n+1}$.

If $p(I, N)$ is positive, this value may not work, but the average between it and $p(I, M)$ is positive, and can therefore be used as $\frac{w_1}{w_2}$. Let $p(I, M) = \frac{a}{b}$ and $p(I, N) = \frac{c}{d}$.

$$
\begin{aligned}
\frac{p(I, N) + p(I, M)}{2} &= \frac{\frac{a}{b} + \frac{c}{d}}{2} \\
&= \frac{a}{2b} + \frac{c}{2d} \\
&= \frac{ad}{2bd} + \frac{cb}{2bd} \\
&= \frac{ad + cb}{2bd}
\end{aligned}
$$

Since this is the average between two positive values, it is positive. The numerator and the denominator may both be negative, but their absolute values produce the same fraction. Since this is $\frac{w_1}{w_2}$, the weights can be taken to be:

$$\begin{aligned} w_1 &= |(d(N,K_2) - d(I,K_2)) \times (d(I,K_1) - d(M,K_1)) + \\ &\quad (d(M,K_2) - d(I,K_2)) \times (d(I,K_1) - d(N,K_1))| \\ w_2 &= |2 \times (d(I,K_1) - d(M,K_1)) \times (d(I,K_1) - d(N,K_1))| \end{aligned}$$

Using such weights, every model not satisfying R is further from the bases than all models satisfying R, which proves that if all models of R have the same distances from K_1 and K_2, then R is obtainable if the conditions in the statement of the lemma are true.

If there exists I and J such that $d(I,K_1) \neq d(J,K_1)$, then $\frac{w_1}{w_2}$ is uniquely determined by Property 1 to be $p(I,J)$:

$$\frac{w_1}{w_2} = \frac{d(J,K_2) - d(I,K_2)}{d(I,K_1) - d(J,K_1)}$$

Two values producing this fraction are:

$$\begin{aligned} w_1 &= |d(J,K_2) - d(I,K_2)| \\ w_2 &= |d(I,K_1) - d(J,K_1)| \end{aligned}$$

By the first assumption of the lemma, if $d(I,K_1) - d(J,K_1)$ is negative then $d(I,K_2) - d(J,K_2)$ is positive, and vice versa. As a result, $\frac{w_1}{w_2}$ is $\frac{d(J,K_2)-d(I,K_2)}{d(I,K_1)-d(J,K_1)}$ despite the absolute values.

Let L be another model of R. If $d(I,K_1) = d(L,K_1)$, by the first condition of the lemma $d(I,K_2) = d(L,K_2)$, which implies that I and L are at the same weighted distance from the bases regardless of the weights. Otherwise, $d(I,K_1) \neq d(L,K_1)$, and Property 1 applies: if $\frac{w_1}{w_2} = p(I,L)$ then I and L are at the same distance from the bases. But $\frac{w_1}{w_2}$ has been proved to be equal to $p(I,J)$, and by the third assumption of the lemma $p(I,J) = p(I,L)$.

Let $M \not\models R$. By the assumptions of the lemma, $p(I,J) < p(I,M)$ if $d(I,K_1) - d(M,K_1) > 0$ and $p(I,J) > p(I,M)$ if $d(I,K_1) - d(M,K_1) < 0$. By Property 2, the distance from M to $\{K_1,K_2\}$ is greater than that of I. That concludes the proof that the conditions of the lemma imply that R is obtainable.

If some of the conditions of the lemma are falsified, then R is not obtainable from $\{K_1,K_2\}$ with any weights. This is proved for each condition at time.

The first condition is false if $d(I,K_1) \geq d(J,K_1)$ but $d(I,K_2) > d(J,K_2)$. In such conditions the weighted distance of I is less than that of J regardless of the weights, implying that J is not in the result of the merging in spite of $J \models R$.

The second condition is false if $d(I,K_1) \geq d(M,K_1)$ and $d(I,K_2) \geq d(M,K_2)$, which imply that the weighted distance of I is greater than or equal to that of M regardless of the weights, implying that either M is in the result of merging or I is not, while $I \models R$ and $M \not\models R$.

The third condition is false if $p(I,J) \neq p(I,L)$ for some I,J,L that are models of R with $d(I,K_1) \neq d(J,K_1)$ and $d(I,K_1) \neq d(L,K_1)$. By Property 1, I and J are at the same distance only if $\frac{w_1}{w_2}$ is $p(I,J)$; I and L are at the same distance only if it is $p(I,L)$.

These are different, showing that no pair of weights makes I, J and L to be at the same weighted distance from the bases.

The fourth condition is false if $d(I, K_1) \neq d(J, K_1)$, $d(I, K_1) > d(M, K_1)$ and $p(I, J) \geq p(I, M)$. The first implies $\frac{w_1}{w_2} = p(I, J)$ by Property 1 and $I \models R$, $J \models R$, the second that $\frac{w_1}{w_2} < p(I, M)$ by Property 2 and $I \models R$ and $M \not\models R$. Therefore, $p(I, J) < p(I, M)$, contradicting $p(I, J) \geq p(I, M)$.

The fifth condition is similar, with $d(I, K_1) < d(M, K_1)$ implying $\frac{w_1}{w_2} > p(I, M)$, which together with $\frac{w_1}{w_2} = p(I, J)$ contradicts $p(I, J) \leq p(I, M)$.

The sixth condition is false if $d(I, K_1) - d(N, K_1) > 0$, $d(I, K_1) - d(M, K_1) < 0$ and $p(I, M) \geq p(I, N)$. Since $I \models R$, $M \not\models R$ and $N \not\models R$, Property 2 applies: $p(I, M) < \frac{w_1}{w_2} < p(I, N)$, contradicting $p(I, M) \geq p(I, N)$. □

In the particular case $K_1 = K_2$, Condition 1 implies $d(I, K_1) = d(J, K_1)$, which nullifies Conditions 3–5. In a similar way, Condition 2 implies $d(I, K_1) < d(M, K_1)$, which nullifies Condition 6. As a result, obtainability simplifies to all models of R being at the same distance from K_1 and all other models being at a greater distance.

Another particular case of interest is when R is the result of merging K_1 and K_2 with $w_1 = w_2 = 1$. In this case, $p(I, J)$ is 1 for all pairs of models I and J of R, which means $d(I, K_1) - d(J, K_1) = d(J, K_2) - d(I, K_2)$. A model M that does not satisfy R may be closer to K_1 than I or not, leading to Condition 4 or to Condition 5, respectively.

In the example after the definition of $p(I, J)$, the value $p(I, J) = 2$ implies that R can only be obtained by setting weights such that $\frac{w_1}{w_2} = 2$. This implies that $p(I, L) = 2$ was also necessary to obtainability. The value $p(I, M) = 1$ does not alone support the obtainability of R; by Theorem 3.1, since M does not satisfy R the value of $p(I, M)$ has not only to be different from $p(I, J)$, but also less or greater than it depending on the sign of $d(I, K_1) - d(M, K_1)$. In this particular case this difference is $1 - 4$; since it is negative, by Condition 4 it should be $p(I, K_1) > p(I, M)$ which is indeed the case.

3.1. Complexity

Theorem 3.1 expresses obtainability in terms of a universally quantified condition containing $d(I, K_i)$. If determining such a value is polynomial, the problem is in coNP. Two cases where this happens are:

— d is the drastic distance;
— d is the Hamming distance and both K_1 and K_2 are conjunctions of literals.

If determining the value of $d(I, K_i)$ is in some complexity class harder than polynomial-time, the complexity of obtainability is in an higher level of the polynomial hierarchy than coNP. This is the case for example for the Hamming distance in general, since in this case $d(I, K_i)$ is the minimal number of literals that differ from I and a model of K_i. Obtainability can be rewritten as:

$$\forall I J \ldots \ \forall d_I^1 d_I^2 d_J^1 \ldots \ \begin{pmatrix} (\exists I' \models K_1 \ . \ d(I, I') \leq d_I^1) \wedge \\ (\forall I'' \models K_1 \ . \ d(I, I'') \geq d_I^1) \end{pmatrix} \wedge \cdots \rightarrow \text{(conditions in Theorem 3.1)}$$

Since the quantifiers $\exists I'$ and $\forall I''$ are inside the premise of an implication, they are negated. However, they are still two independent quantifiers. Therefore, this is a $\forall\exists QBF$, which hints that obtainability is in Π_2^p. The same happens if checking $d(I, K_i) \leq x$ is in NP or in coNP. More generally, the complexity of obtainability is one level over the complexity of calculating the distance between a model and a knowledge base.

There is however a limit case to keep into account: if $d(I, K_i)$ is a number so large that exponential space is required to represent it, then $d(I, K_i) \leq x$ is true (and hence trivial to check) for every value of x of size comparable to I and K_i. In such cases, $d(I, K_i) \leq x$ may take polynomial time in the total size of I, K_i and x, but only because the enormous size of x dwarfs the computation on I and K_i, which may be superpolynomial in the size of I and K_i only. While this can be considered a limit case, it is still to be taken into account.

THEOREM 3.2. *If determining $d(I, K) \leq x$ is in the complexity class Π_i^p or Σ_i^p and $d(I, K)$ is representable in space polynomial in that of I and K, then obtainability of a satisfiable formula from two formulae with a weighted sum of distances is in Π_{i+1}^p.*

Proof. By Theorem 3.1, obtainability can be expressed as formula with some universal quantifiers in the front $\forall I, J, L, M, N$ and a formula F containing $d(I, K_1)$, $d(I, K_2)$, $d(J, K_1)$, etc. Equivalently:

$$
\begin{aligned}
&\forall I, J, L, M, N \ \forall d_I^1, d_I^2, d_J^1, d_J^2, \ldots \\
&\quad (d(I, K_1) \leq d_I^1) \wedge \\
&\quad \neg(d(I, K_1) \leq d_I^1 - 1) \wedge \\
&\quad (d(J, K_1) \leq d_J^1) \wedge \\
&\quad \neg(d(J, K_1) \leq d_J^1 - 1) \wedge \\
&\quad \vdots \\
&\quad \rightarrow F[d(I, K_1)/d_I^1, d(I, K_2)/d_I^2, d(J, K_1)/d_J^1, d(J, K_2)/d_J^2, \ldots]
\end{aligned}
$$

Quantification over d_I^1, d_I^2, etc. can be done because by assumption $d(I, K_1)$, $d(I, K_2)$ etc. are bounded in size by a polynomial in the size of the models and of the formulae. In other words, these values can be be represented with a polynomial amount of bits.

If d can be calculated in polynomial time, the whole problem is in coNP. Otherwise, subformulae $d(I, K_1) \leq d_I^1$ occur in the premise of an implication, so they are in fact negated. However, if each is in Π_i^p or in Σ_i^p, they can be expressed as an alternation of i quantifiers. The whole problem, with the universal quantifier in the front, is therefore in Π_{i+1}^p. $\square$

This theorem implies the three ad-hoc complexity results obtained above: that obtainability is in coNP for the drastic distance and for the Hamming distance when the knowledge bases are conjunctions of literals, and is in Π_2^p in the general case for the Hamming distance. A general hardness result can be given from some assumptions about the distance function.

A pseudodistance is a function such that $d(I, J) = d(J, I)$, $d(I, I) = 0$ and $d(I, J) > 0$ for every $J \neq I$. Its extension to a distance from a knowledge base obeys: $d(I, K) = 0$ if $I \models K$ and $d(I, K) > 0$ otherwise. If K_1 and K_2 have some common models, these have weighted distance 0 regardless of the weights. Since merging selects minimal models, in this case the result comprises exactly the common models. In particular, if K_1 and K_2 coincide, merge produces a formula equivalent to them. This holds for every pseudodistance, and can be used to prove that obtainability is coNP-hard for every pseudodistance.

THEOREM 3.3. *Obtainability of a consistent formula from two knowledge bases is* coNP*-hard for every pseudodistance.*

Proof. The claim is proved by reduction from propositional unsatisfiability. Let F be a propositional formula. The corresponding obtainability problem is defined by $K_1 = K_2 = y$ and $R = y \vee F$, where y is a variable not in F. Since K_1 and K_2 coincide, the result of merging is y. If F is satisfied by a model I then R has a model $I \cup \{\neg y\}$ that does not satisfy y. Vice versa, if F is unsatisfiable then R coincides with y. $\Box$

Since obtainability for drastic distance and Hamming distance from conjunctions of literals is in coNP, and these are pseudodistances, obtainability using them is coNP complete. The unrestricted problem with the Hamming distance is Π_2^p-hard. This is proved by reduction from the problem of establishing the validity of a formula $\forall X \exists Y.F$. The translation is based on two main ideas:

(1) separate models having different evaluations of X by a large distance;
(2) for each evaluation of X, K_1 and R contain the subformula $Y^\neg \wedge Y'^\neg$ that sets all variables in Y and a copy of it Y' to false; K_2 instead contains $F \wedge (Y \not\equiv Y')$.

The second property makes the model of K_1 being at distance n from K_2, but only if R is satisfiable, and such models are in the result of merging with $w_1 \ll w_2$. Formal proof follows.

THEOREM 3.4. *Obtainability with the weighted sum of Hamming distance from two knowledge bases is Π_2^p-complete.*

Proof. Membership follows from Theorem 3.2, since checking $d(I, K) \leq x$ is in NP for the Hamming distance. Indeed, $d(I, K) \leq x$ holds if there exists $J \models K$ such that $d(I, J) \leq x$, and the distance between two models can be determined in polynomial time.

Hardness is proved by reduction from the problem $\forall\exists QBF$.

First, the problem of checking the validity of $\forall X \exists Y.F$ remains hard even if F is known to be satisfiable. This is proved by reduction from the problem without the restriction: $\forall X \exists Y.G$ is valid if and only if $\forall z \forall X \exists Y.G \vee z$ is valid, where z is a new variable: indeed, this formula is equivalent to $(\forall X \exists Y.G \vee \top) \wedge (\forall X \exists Y.G \vee \bot)$; the first part of this conjunction is tautological, the second is equivalent to the original QBF.

Second, the problem of checking the validity of $\forall X \exists Y.F$ with F satisfiable is reduced to obtainability. Let $n = |X| = |Y|$ and $Y', X_1, \ldots, X_{2n}$ be each a set of n new variables.

$$
\begin{aligned}
K_1 &= (X \equiv X_1 \equiv \cdots \equiv X_{2n}) \wedge Y^\neg \wedge Y'^\neg \\
K_2 &= (X \equiv X_1 \equiv \cdots \equiv X_{2n}) \wedge (Y \not\equiv Y') \wedge F \\
R &= K_1
\end{aligned}
$$

That the reduction works is proved by first proving three preliminary claims:

(1) the distance between models of K_1 or K_2 differing on the evaluation of X is $2n$ or more;
(2) no model has distance vector $(0, k)$ with $k < n$;
(3) the models that have distance vector $(0, n)$ are exactly the models of K_1 that satisfy F by changing the values of Y in some way;

Since both K_1 and K_2 contain $X \equiv X_1 \equiv \cdots \equiv X_{2n}$, if two of their models differ even on a single variable in X they also differ on all its $2n$ copies. Therefore, models of K_1 and K_2 with different evaluations of X are at least $2n$ apart.

To prove that no model is at distance $(0, k)$ with $k < n$, it suffices to consider the models of K_1, since these are the only ones with 0 in the first position of the distance vector. Let I be a model of K_1. By the previous property, models of K_2 with a different evaluation of X are at distance $2n$ or more. The models of K_2 with the same evaluation of X differ only on the values of Y and Y'. However, since K_2 contains $Y \not\equiv Y'$, all models of K_2 have exactly n positive literals in $Y \cup Y'$. Since K_1 contains $Y^\neg$ and $Y'^\neg$, its models assign false to all $Y \cup Y'$. As a result, the distance between these models is n, leading to a distance vector $(0, n)$.

A model has distance vector $(0, n)$ only if it is a model of K_1. Let I be model of K_1, and I^X its restriction to the variables X only. By the above point, the distance between I and a model of K_2 with a differing evaluation of X is $(0, 2n)$ or more. Therefore, these models of K_2 can be excluded from consideration: it suffices to consider models of K_2 with the same evaluation I^X on the variables X. Since K_2 implies F, such models exist if and only if F can be satisfied by adding a suitable evaluation of the variables Y to I^X. If this is the case, a model of K_2 has the same values of I on X and all its copies X_i, while it assigns exactly n among Y and Y' to true. Since I has the same values of X but sets false all variables Y and Y', its distance from that model is exactly $(0, n)$.

Let F be a satisfiable formula over variables X and Y.

If $\forall X \exists Y.F$ is true, then for every evaluation I^X over X some evaluation over variables Y makes F true. This means that every model I of K_1 has distance vector $(0, n)$. Formula K_1 is obtained by merging K_1 and K_2 with weights $w_1 = n + 1$ and $w_2 = 1$: models with distance vector $(0, n)$ have weighted distance n, all other models have distance vector (k, k') with $k > 0$ and weighted distance $n + 1$ or more.

If $\forall X \exists Y.F$ is false, then an evaluation I^X over X makes F true for no evaluation of Y. Let I be the model of K_1 with I^X as its evaluation of X; the values of all X_i are by definition the same and Y, Y' all false. By what proved above, since I^X cannot be extended to satisfy F, the distance vector of I is $(0, k)$ with $k > n$. Since some models have distance vector $(0, n)$, this model I is in the result of merging for no choice of w_1 and w_2. □

3.2. Local search algorithm

An algorithm using local search is shown. It employs two elements of the proof of Theorem 3.1 to obtain $\frac{w_1}{w_2}$ or some bounds on its value. No assumption is made over $d(I, K)$ other than the availability of a procedure to determine it; for the drastic distance this is straightforward, as it amounts to check whether $I \models K$; for the Hamming distance, since the problem is NP-complete, an approximate method can be used instead. Once $\frac{w_1}{w_2}$ is determined, the knowledge bases are merged and the result checked for equivalence to R. This final check is necessary because the value of $\frac{w_1}{w_2}$ is not determined with certainty: not all models of R and of $\neg R$ are checked.

Property 1 ensures that if two models of R are such that the denominator of $p(I, J)$ is not null, then $\frac{w_1}{w_2} = p(I, J)$. Two such models can be looked upon using local search. During the run of the procedure, models that do not satisfy R are used to establish or refine bounds on the value of $\frac{w_1}{w_2}$. This is useful because, as Property 2 shows, even if for all pairs of models of R the denominator of $p(I, J)$ is zero, the models that do not satisfy R still constrain $\frac{w_1}{w_2}$.

Summing up, the algorithm does two things at the same time:

(1) looks for two models I and J of R such that $p(I, J)$ has a non-zero denominator;
(2) if a model I of R has been found, for every model M of $\neg R$ found during the search $p(I, M)$ is used to refine two bounds.

In the following algorithm, conditions involving I are to be considered false if I is unassigned, for example when the algorithm starts. The result is $\frac{w_1}{w_2}$ or the special value "unobtainable"; the first is assumed to be returned as a pair of integers, rather than a (possibly truncated) rational value. The maximal distance between two models is denoted by n; this is 1 for the drastic distance and the number of variables for the Hamming distance. This is also the maximal value of $p(I, J)$ and the reason why a is initialized to $n+1$. The algorithm depends on a parameter that is common to local search algorithms, the maximal number of iterations before giving up, here named $maxiter$.

ALGORITHM 1.

(1) $a = n + 1$; $b = -n - 1$
(2) $iter = 0$
(3) *if* $iter \bmod restart = 0$ *set O=random model*
(4) *change* O *by local search for a model of* R *(see below)*
(5) *if* $O \models R$ *and* I *is unassigned set* $I = O$
(6) *if* $O \models R$ *and* $p(I, O)$ *has a non-zero denominator, then:*
— *if* $p(I, O)$ *is positive and between* a *and* b *then return* $p(I, O)$
— *otherwise return unobtainable*
(7) *if* $O \not\models R$ *and* $d(I, K_1) - d(O, K_1) > 0$ *then* $a = min(a, p(I, O))$
(8) *if* $O \not\models R$ *and* $d(I, K_1) - d(O, K_1) < 0$ *then* $b = max(b, p(I, O))$
(9) *if* $a < 0$ *or* $a \leq b$ *return unobtainable*
(10) $iter = iter + 1$
(11) *if* $iter < maxiter$ *go to Step 3*
(12) *return* $\frac{a+b}{2}$

Point 4 is a step of a local search for a model of R: for example, if F is in CNF it may change the value of a single variable in such a way the number of clauses that are satisfied by the current interpretation is increased as much as possible. More refined methods can be employed, such as making random moves with a certain probability, which may remain constant or decrease with the number of iterations.

This algorithm returns $\frac{w_1}{w_2}$ as a pair of integer numbers, which can be used as the weights w_1 and w_2. If merging with these weights produces R, then they are the searched weights. Otherwise, if the value is returned from Step 6 then R is not obtainable. If it is returned from Step 12, then one may attempt some other value between a and b, or keep searching some more.

Several variants may be considered.

(1) Step 4 looks for a model of R, but after a number of iterations without finding one that makes the denominator of $p(I, O)$ different than zero, it makes sense to aim at minimizing a and maximizing b instead;
(2) models with a distance vector strictly greater than others cannot be in the result of merge; therefore, if they satisfy R then R is not obtainable; if they are not in R they can be neglected;
(3) instead of returning immediately after determining $p(I, J)$ in Step 6, one may proceed with local search and check whether some other models of R and of $\neg R$ satisfy the conditions of Theorem 3.1.

The main idea of the algorithm can also be carried to some other methods for propositional satisfiability. Other algorithms can indeed find two models of R to determine $\frac{w_1}{w_2}$ and some interpretations not satisfying R to set bounds on this fraction. All that is needed is the possibility of continuing after finding the first model, and the ability to identify interpretations not satisfying the formula during the search. DPLL [Nieuwenhuis et al. 2006] and propositional tableau [d'Agostino 1999] can be used in place of local search.

3.3. Tractable case

This section shows a tractable case of obtainability: the measure is the Hamming distance, the knowledge bases are conjunctions of literals and the expected result of merging is a Horn or Krom formula.

THEOREM 3.5. *If K_1 and K_2 are conjunctions of literals, determining whether a Horn or Krom formula R is obtainable by the weighted sum of the Hamming distances is in* P.

Proof. For a model I and a variable x, let $I \cdot x$ denote a model that is identical to I except that x is assigned the value true. $I \cdot \neg x$ is the same with value false. The first step of the proof is a property of $d(I, K_i)$ when K_i entails a literal or does not mention a variable.

— if K_i entails x then $d(J \cdot x, K_i) < d(J \cdot \neg x, K_i)$; since K_i entails x, all its models set x to true; this hold in particular for every model J that is one of the closest to I; since $I \cdot \neg x$ and $I \cdot x$ have the same differing literals from J except for x, which is positive in J, then $d(I \cdot x, K_u) < d(I \cdot \neg x, K_i)$; the same property holds when K_i entails $\neg x$;
— if K_i does not contain x then $d(I \cdot x, K_i) = d(I \cdot \neg x, K_i)$; since K_i does not mention x, it it is satisfied by $J \cdot x$ if and only if it is satisfied by $J \cdot \neg x$ for every interpretation J; therefore, if J is a model at a minimal distance from I then $J \cdot x$ is at minimal distance from $I \cdot x$; the same holds for $\neg x$; therefore, $d(I \cdot x, K_i) = d(I \cdot \neg x, K_i)$.

The second step of the proof relates merge result to the weighted distance of $I \cdot x$ and $I \cdot \neg x$. Both are based on merge being defined from the set of models of minimal weighted distance.

(1) if every model $I \cdot \neg x$ has greater weighted distance from $\{K_1, K_2\}$ than $I \cdot x$ then the merge result implies x, and the same for $\neg x$; indeed, since every model where x is false is further than the same one where x is true, minimal models all have x true;
(2) if every model I is at the same weighted distance from $\{K_1, K_2\}$ as $I \cdot x$ and $I \cdot \neg x$ then the merge result does not mention x; indeed, if this is true then minimal models are symmetric with respect to x and $\neg x$; the value of x is therefore irrelevant to the satisfaction of the merge result.

The claim can now be proved. Variables are divided in the three groups: those mentioned neither in K_1 nor in K_2; those occurring in a base but not with the opposite sign in the other; those occurring with opposite signs.

If neither K_1 nor K_2 mention x then for every I it holds $d(I \cdot x, K_1) = d(I \cdot \neg x, K_1)$ and $d(I \cdot x, K_2) = d(I \cdot \neg x, K_2)$, which imply that $I \cdot x$ and $I \cdot \neg x$ have the same weighted distance regardless of the weights. This implies that the merge result does not mention x.

If x is in K_1 and is not mentioned in K_2, then $d(I \cdot x, K_1) < d(I \cdot \neg x, K_1)$ and $d(I \cdot x, K_2) = d(I \cdot \neg x, K_2)$, which imply that $I \cdot x$ has lower weighted distance than $I \cdot \neg x$. If x is also in K_2 then $d(I \cdot x, K_2) < d(I \cdot \neg x, K_2)$, and the result is the same. In both cases, the result of the merge entails x.

If $K_1 \models x$ and $K_2 \models \neg x$, then $d(I \cdot x, K_1) < d(I \cdot \neg x, K_1)$ and $d(I \cdot x, K_2) > d(I \cdot \neg x, K_2)$. The result of merge depends on the weights. If $w_1 > w_2$ then $I \cdot x$ has lower weighted distance than $I \cdot \neg x$, proving that the merge result entails x. The same holds for all other literals that are in K_1. In other words, if $w_1 > w_2$ then the result of merge contains all literals in K_1 that occur with the opposite sign in K_2. The same holds in reverse if $w_1 < w_2$: the result of merge contains all literals of K_2. If $w_1 = w_2$ then $I \cdot x$

and $I \cdot \neg x$ have the same weighted distance, proving that the result of merge does not mention x.

As a result, if $w_1 > w_2$ then the result of merge contains not only the literals that are in K_1 and do not occur negated in K_2, but also the ones that occur negated in K_2. The contrary happens if $w_1 < w_2$. If $w_1 = w_2$ then the result of merge does not contain the variables with opposite sign in K_1 and K_2. Each of these three possible results can be checked for equivalence with R in polynomial time because of the Horn or Krom restriction. □

4. PRIORITY BASE MERGING

Priority base merging [Nebel 1992; 1998; Rott 1993; Delgrande et al. 2006] is a semantics that selects groups of formulae based on a priority ordering over them. Such an ordering over the knowledge bases $K_1, \ldots, K_m$ can be defined as an ordered partition P of them (this representation is similar to the one used by Rott [1993] for orderings over formulae); the classes of the partition are denoted $P(1), P(2), P(3), \ldots$ and are not empty. The lower the class K_i belongs to, the higher its reliability is. Such a partition allows comparing two sets of formulae: $L \equiv N$ if and only if L and N are equal; $L < N$ if and only if $P(1) \cap L = P(1) \cap N, \ldots P(i-1) \cap L = P(i-1) \cap N$ and $P(i) \cap L \supset P(i) \cap N$ for some number i, possibly 1.

The *maxsets* of a set of formulae $K_1, \ldots, K_m$ are its maximally consistent subsets. Formally, M is a maxset of $K_1, \ldots, K_m$ if M is consistent, $M \subseteq \{K_1, \ldots, K_m\}$ and $M \cup \{K_i\}$ is inconsistent for every $K_i \in \{K_1, \ldots, K_m\} \backslash M$. Maxsets can be recast in terms of base remainder sets [Alchourrón et al. 1985; Booth et al. 2011].

Merging $K_1, \ldots, K_m$ according to a priority ordering is disjoining the maxsets that are minimal according to the ordering [Nebel 1992; 1998; Rott 1993; Delgrande et al. 2006]. This is equivalent to disjoining the minimal consistent subsets, including the ones that are not maximally consistent.

By definition, the result of merging is always an or-of-maxsets. However, not all possible or-of-maxsets are produced by merging: some are not generated by any priority partition. Given an or-of-maxsets of $K_1, \ldots, K_m$, the maxsets it contains are called *selected*, the others *excluded*. The aim is to find an ordering, if any, that makes the selected maxsets minimal and the other ones non-minimal.

A formula R is *obtainable* from $K_1, \ldots, K_m$ if it can be obtained by merging these formulae. For the merging based on priority orderings, this amount to checking the existence of an ordering that makes the result of merging $K_1, \ldots, K_m$ equal to R. This condition is equivalent to the existence of an ordering such that the minimal maxsets are exactly the selected ones. The difference between "selected" and "minimal" is that the first one is a requirement (the maxset is in the expected result R) while the second is a condition over a specific ordering (it makes the maxset minimal). Not all formulae are obtainable, and this will be formally proved.

Given formulae R and $K_1, \ldots, K_m$, the problem of obtainability is that of finding (search problem) or deciding the existence of (decision problem) a priority ordering such that R is the result of merging $K_1, \ldots, K_m$ with that ordering.

As usual, the complexity analysis is carried over the decision version of the problem, but the algorithm in Section 4.4 is aimed at finding the actual priority ordering, if one exists. Otherwise, Section 5 describes some possible courses of actions in case of unobtainability.

A number of properties related to obtainability are shown. The first ones are about maxsets in general, the following about the specific problem of obtaining a formula as the result of merging with an appropriate priority ordering.

4.1. Properties of maxsets

A general property of maxsets is that they are pairwise inconsistent. This is quite a folklore result, and is proved here only for the sake of completeness.

LEMMA 4.1. *Two different maxsets of the same set of formulae are mutually inconsistent.*

Proof. To the contrary, assume that M and N are two differing maxsets such that $M \cup N$ is consistent. Since M and N differ, either $M \backslash N$ or $N \backslash M$ is not empty. In the first case, since $M \cup N = N \cup (M \backslash N)$, then N is consistent with other formulae not in N. This contradicts the assumption that N is a maxset: no formula can be be consistently added to N. A similar line proves the impossibility of the other case. $\square$

LEMMA 4.2. *If M is a maxset of $K_1, \ldots, K_m$ and I one of its models, then $M = \{K_i \mid I \models K_i\}$.*

Proof. I is a model of M if it is a model of all formulae of M, that is, the formulae of M are a subsets of those satisfied by I. This proves that $M \subseteq \{K_i \mid I \models K_i\}$. If such a containment were strict, the formulae K_i that are not in M would be consistent with M because they are satisfied by I, contradicting the assumption that M is a maxset. $\square$

When checking minimality using a priority ordering, considering all consistent subsets or the maxsets obly does not make difference, as the following lemma shows.

LEMMA 4.3. *If $N \subset M$ then M is less than N according to every priority ordering, where N and M are two sets of formulae.*

Proof. If $N \subset M$ then $N \cap P(i) \subseteq M \cap P(i)$ for every i. Since the containment is strict, $M \backslash N$ is not empty. Let K_i be an element of it, and j its class. Containment $N \cap P(i) \subseteq M \cap P(i)$ holds for all i's, including $i = j$. For this index, however, $K_i \notin N \cap P(j)$ while $K_i \in M \cap P(j)$, proving that M is strictly less than N according to the ordering. $\square$

A consequence of this lemma is that the consistent subsets that are minimal according to an arbitrary ordering are also maxsets. Also, a maxset is minimal if and only if is not less than another consistent subset.

LEMMA 4.4. *A maxset is not minimal if and only if some other consistent subset of the same set of formulae is less than it according to the priority ordering.*

Proof. Let M be a maxset and N a consistent subset that compares less than it according to the ordering. If N is also maximally consistent, the claim holds. Otherwise, some formulae can be added to it to obtain a maxset. Adding formulae only makes N lesser according to the ordering P, by Lemma 4.3. This proves that if some consistent subset of formulae is less than M, then M is not a minimal maxset.

Vice versa, if M is not minimal then another maxset N is less than it according to the ordering. Since N is by definition of maxset a consistent subset of formulae, the claim holds. $\square$

The following lemma helps in identifying the minimal maxsets.

LEMMA 4.5. *For every maxset M that is minimal according to priority P it holds $M \cap P(1) \neq \emptyset$.*

Proof. To the contrary, assume that $M \cap P(1) = \emptyset$. By definition of priorities, $P(1)$ is not empty. Let K be a formula of it. By the assumption that all formulae are consistent, $\{K\}$ is consistent. Moreover, $P(1) \cap M \subset P(1) \cap \{K\}$, which by definition implies $\{K\} < M$, contradicting the assumption that M is minimal. $\square$

In words, minimal maxsets have at least a formula in the first class of the priority partition. This result depends on all formulae being consistent and no priority class being empty, both of which are assumed in this article.

The next lemma is useful for producing maxsets with some given property. It tells how to build formulae that have some given maxsets. In particular, the maxsets are specified on letters $A, B, C, D, \ldots$, which are just arbitrary symbols. Given some sets of them, such that $\{A, B\}$, $\{B, C, D\}$, etc., one can build a formula for A, a formula for B, etc., in such a way the maxsets of these formulae are exactly the given sets $\{A, B\}$, $\{B, C, D\}$, etc. The only requirements is that none of these sets is contained in another: for example, if $\{A, B\}$ is given then $\{A, B, C\}$ cannot.

For example, given the sets of letters $\{A, B\}$, $\{A, C\}$ and $\{B, C\}$, the next lemma shows which formulae to use in place of the letters: x for A, y for B and $x \not\equiv y$ for C. Their maxsets are indeed $\{x, y\}$, $\{x, x \not\equiv y\}$ and $\{y, x \not\equiv y\}$.

LEMMA 4.6. *Given some sets of letters, none of these sets contained in another, there exists a formula for each letter so that the maxsets of these formulae correspond to the given sets of letters.*

Proof. For n sets, $\lceil \log n \rceil$ propositional variables are required. Each set of letters is associated a unique propositional interpretation; this is possible because by construction there are at least n propositional interpretations over these variables.

For each such interpretation, one can build a formula that is satisfied only by it. For example, if the interpretation makes x and y false and z true, the formula is $\neg x \wedge \neg y \wedge z$. Since each set of letters is associated a propositional interpretation, is also associated to the corresponding formula.

If letter L is in the sets $S_1, S_2, \ldots$, and these sets corresponds to formulae $F_1, F_2, \ldots$, the formula of L is their disjunction $F_1 \vee F_2 \vee \cdots$. As a result, the formula corresponding to the letter L is satisfied exactly by the interpretations of the sets $S_1, S_2, \ldots$.

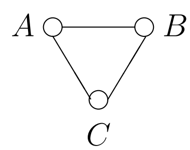
By construction, if a set of letters is associated to the interpretation I, the formulae corresponding to the letters in the set are satisfied by I. This proves that each set of letters corresponds to a consistent set of formulae. This set is also maximally consistent because: a. no other formula is satisfied by that interpretation; and b. if all formulae of the set plus some others are satisfied by another interpretation, then the set corresponding to that interpretation strictly contains the considered one, contradicting the assumption that none of the sets strictly contains another.

To conclude the proof, the formulae do not have other maxsets. This is because the formulae are only satisfied by some of the interpretations corresponding to the sets of letters, and each of them is the only model of a maxset. □

Intuitively, this lemma proves that letters can be used in place of formulae, and sets of letters for their maxsets. Provided that no set is contained in another, it is always possible to build a set of formulae to use in place of the letters, and the sets of letters will be their maxsets. This method can be used for example to show that maxsets may form a sort of "cycles". The first step is to define the sets of letters:

(1) $\{A, B\}$
(2) $\{A, C\}$
(3) $\{B, C\}$

Binary sets can be drawn as edges of a graph, a graphical representation that will be used also in the rest of this article:



Instead of showing formulae with maxsets having the given property, the maxsets are expressed as sets of letters, each representing a formula. Lemma 4.6 tells that such formulae exist, its proof how to build them. In this case, three sets require two variables, like x and y. The interpretations associated to the sets can be chosen arbitrarily, for example:

— $\{A, B\} \Rightarrow \{x, y\}$
— $\{A, C\} \Rightarrow \{x, \neg y\}$
— $\{B, C\} \Rightarrow \{\neg x, y\}$

Since A is in $\{A, B\}$ and in $\{A, C\}$, its formula is one satisfied by the models of these two sets: $\{x, y\}$ and $\{x, \neg y\}$. For example, A is $(x \wedge y) \vee (x \wedge \neg y)$, which simplifies to x. In the same way, $B = y$ and $C = (x \not\equiv y)$.

These formulae $x, y, x \not\equiv y$ have the required maxsets, each composed of exactly two formulae over three. From now on, this explicit construction of formulae from sets of letters representing their maxsets is generally not done, with Lemma 4.6 referenced as evidence that it is possible. This is first done in the proof of Lemma 4.10, showing that a formula that is an or of some maxsets may not be obtainable with any ordering.

The next two lemmas show that some results are easy to obtain: selecting all maxsets or just a single one.

LEMMA 4.7. *The priority ordering that gives maximal priority to all formulae makes all maxsets minimal.*

Proof. A maxset M could be non-minimal only if there exist another maxset N such that $N < M$. Since all formulae are in $P(1)$, the definition of ordering of maxsets simplifies to: $N < M$ if $M \subset N$. This contradicts the assumption that M is maximally consistent. □

LEMMA 4.8. *The priority ordering that gives maximal priority to exactly the formulae of a maxset makes it the only minimal one.*

Proof. By contradiction, if M is not minimal then $N < M$ for some other maxset N. This implies either $P(1) \cap M \subseteq P(1) \cap N$ or $P(1) \cap M \subset P(1) \cap N$. The latter contradicts $P(1) = M$. The former implies $M \subseteq P(1) \cap N$, which is only possible if $M = N$ or $M \subset N$, and a maxset is never contained in another. □

4.2. Properties of obtainability

The following lemma expresses equivalent conditions for a maxset to be a disjunct of the result of merging.

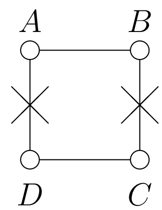
LEMMA 4.9. *If R is obtainable by priority base merging from some formulae and M is a maxset of them, the following conditions are equivalent:*

— *M is consistent with R;*
— *$M \models R$;*
— *M is selected in all orderings that generate R.*

Proof. Since the maxsets are mutually inconsistent by Lemma 4.1, each model of R is contained in exactly a maxset M. Therefore, M is one of the disjuncts that form R if and only if it is consistent with R, and this holds in every ordering that generate R. □

By definition, merging produces a disjunction of some of the maxsets, the minimal ones according to the priority ordering. A first question is whether all disjunctions of maxsets are obtainable with an appropriate ordering. The following lemma shows that the answer is no.

The counterexample uses four maxsets, of which two are selected and two excluded. "Selected" and "excluded" indicates whether a maxset is in the disjunction that is the expected result of merging. In other words, the required ordering would have the selected maxsets as the minimal ones. If maxsets are binary, they can be depicted as a graph, where a crossed edge represents an excluded maxset:



LEMMA 4.10. *No priority ordering selects $\{A, B\}$ and $\{C, D\}$ while excluding $\{B, C\}$ and $\{D, A\}$.*

Proof. By Lemma 4.6, letters and sets of letters can be used in place of formulae and their maxsets, respectively. The following maxsets are proved not be obtained by any ordering:

(1) $\{A, B\}$ selected
(2) $\{B, C\}$ excluded
(3) $\{C, D\}$ selected
(4) $\{D, A\}$ excluded

In words, no priority ordering makes the first and third maxsets minimal out of these four.

To the contrary, assume that such an ordering exists. By Lemma 4.5, since $\{A, B\}$ is selected, either A or B is in the first class of the priority partition. For the same reason, either C or D is.

The first class cannot include both A and D, as otherwise $\{A, D\}$ would be minimal. For the same reason, it cannot include both B and C, since $\{B, C\}$ is excluded. The only remaining cases are A and C in the first class, or B and D. The second case is omitted by symmetry: it is the same as the first swapping A with B and C with D.

In the first case, B and D are not in the first class of the priority partition. Since both $\{A, B\}$ and $\{C, D\}$ are selected, if one of them is not in the second class either, so is the other. Since classes cannot be empty, B and D are in the second class:

A	C
B	D

This ordering selects $\{A, B\}$ and $\{C, D\}$ as required, but also $\{B, C\}$. This contradicts the assumption that $\{B, C\}$ is excluded. □

By Lemma 4.6, letters A, B, C, D can be replaced by formulae in such a way the four sets in the lemma represent their maxsets. The impossibility of selecting the first and third while excluding the second and fourth proves that the disjunction of the first and third maxsets is not obtainable.

COROLLARY 4.11. *There exists R and $K_1, \ldots, K_m$ such that R is the disjunction of some of the maxsets of $K_1, \ldots, K_m$ but is not obtainable by priority base merging.*

An application of Lemma 4.6 allows finding the actual formulae to use in place of A, B, C, D. The unobtainable result is then $(A \wedge B) \vee (C \wedge D)$. Formulae like these are later used as the basis of a hardness result.

The maxsets of this lemma form a cycle in which selected and excluded maxsets alternate. This condition is shown to be necessary and sufficient in the case of maxsets comprising two formulae or less.

The counterexample involves four formulae and four maxsets. This is the minimal condition for unobtainability: a result that is an or-of-maxsets is always obtainable if the formulae to be merged are three or less.

THEOREM 4.12. *Every consistent or-of-maxsets is obtainable by priority base merging if the maxsets are less than four.*

Proof. If a set of formulae has a single maxset, the only possible result of merge is the maxset itself, which is therefore always obtainable. With two maxsets, only two cases are possible: select one of them, or both. Lemma 4.8 and Lemma 4.7 cover both cases.

With three maxsets, these lemmas proves that selecting one or all of them is always possible. The only remaining case is that of two selected maxsets out of three. Let them be M, N, and L, where the first two are selected. Being maxsets, M has a formula not in L, and the same for N:

— $M \backslash L \neq \emptyset$
— $N \backslash L \neq \emptyset$

If $M \backslash L$ and $N \backslash L$ intersect, place this intersection in $P(1)$ and all other formulae in $P(2)$. This way, M and N have the same formulae in $P(1)$ while L has none, proving that M and N are selected while L is not.

If $M \backslash L$ and $N \backslash L$ do not intersect, place their union in $P(1)$ and all other formulae in $P(2)$. This ordering guarantees that both M and N have formulae in $P(1)$ while L has none, and that $P(1) \cap M$ and $P(1) \cap N$ are not contained one in the other. □

Since three formulae have at most three maxsets, this theorems proves that every consistent or-of-maxsets of three formulae is obtainable with an appropriate priority ordering.

Lemma 4.10 uses four formulae, indeed: $\{A, B\}$, $\{B, C\}$, $\{C, D\}$, $\{D, A\}$. The disjunction of the first three of these maxsets is also unobtainable: this can be proved in the same line as Lemma 4.10, and shows a case where all maxsets but one are unobtainable. In contrast, Lemma 4.7 and Lemma 4.8 state that a single maxset and all maxsets are always obtainable.

The four maxsets form a cycle, when seen as a graph: $\{A, B\}, \{B, C\}, \{C, D\}, \{D, A\}$. When considering maxsets comprising more than two elements, the notion of Berge–acyclicity [Fagin 1983] for hypergraphs ensure obtainability, as the next theorem shows. An hypergraph is Berge-acyclic if its incidence graph is acyclic. The incidence graph has a node for every edge and one for every node of the hypergraph; two nodes are linked by an edge if they correspond one to an hyperedge and the other a node of the hyperedge.

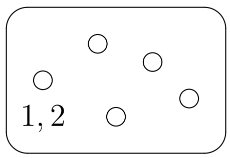
THEOREM 4.13. *Every disjunction of a nonempty subset of a set of maxsets that is Berge–acyclic is obtainable by priority base merging.*

Proof. A set of sets that is Berge-acyclic can be seen as a tree of sets, where each set shares a single node with its parent and one with each of its children. A priority ordering can be build starting from a maxset, labeling its formulae and then moving to its children.

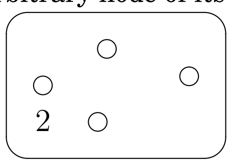
At each step, a set having a single labeled node is considered, and the labeling is extended to its other nodes. A label is either a single number n greater than one or a pair $1, n$ with n greater than one. The meaning of $1, n$ will be clarified later, but it roughly means that the node is part of a selected maxset whose other nodes are labeled n.

The procedure includes some choices, such as the root and a node in each set. It is however not nondeterministic, as it works for any of these choices; in other words, every choice can be resolved by taking arbitrary choices.

The procedure starts from the root. If this maxset is selected, an arbitrary node of its is labeled $1, 2$:

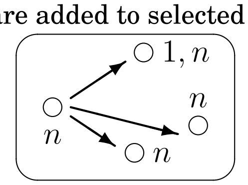


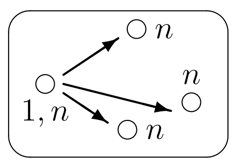
If it is excluded, an arbitrary node of its is labeled 2:



The algorithm descends the tree. When moving from the parent to a child, the former is all labeled and the latter shares a single labeled node with it and its other nodes are unlabeled. These are labeled and the procedure moves to the children.

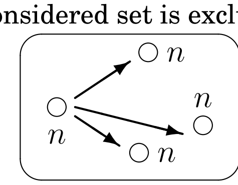
Labels are added to selected edges are follows:

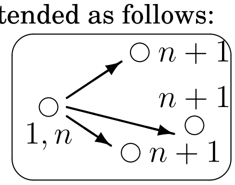




In words, if the only label is n, an arbitrary other node is labeled $1, n$ and the remaining (if any) are labeled n. If the only label is $1, n$, the other nodes are labeled n.

If the considered set is excluded, labels are extended as follows:





In words, if the only label is n, the others are n. If it is $1, n$, the others are $n + 1$.

This labeling is iterated until all nodes are labeled. Labels then tell the class each formula goes into: $1, n$ means class one, n means class n. If the maxsets form a forest, which for example happens if there are isolated maxsets, the procedure is iterated on all its trees.

The procedure of labelling ensures that the following conditions hold:

(1) every maxset contains at most a label $1, n$;
(2) if it does, the others are all n if selected or $n + 1$ if excluded;

(3) otherwise, the maxset is excluded and its labels are equal to a value greater than one;
(4) every label $1, n$ is in at least a selected maxset, and every selected maxset contains at least a label $1, n$.

In other words, every selected maxset contains a label $1, n$ and the remaining labels are n; every excluded maxset has either equal labels greater than one or a label $1, n$ and all others $n + 1$; every $1, n$ label is in at least a selected maxset.

This way, selected maxsets are minimal because they contain a node in class one, the rest in class n, and all other maxsets containing the same node in class one have the others in class n. Excluded maxsets are not minimal because they either contain no formula in class one, or otherwise they contain a formula labeled $1, n$, the others are in class $n + 1$, and the node labeled $1, n$ is is in another maxset having formulae in class n.

In order to complete the proof, we show that the four conditions are ensured when the procedure start, and that none of its step makes them false.

If the first maxset is selected, its first label is $1, 2$ and the others are 2. If it is excluded, all its labels are 2. The conditions therefore hold up to this point.

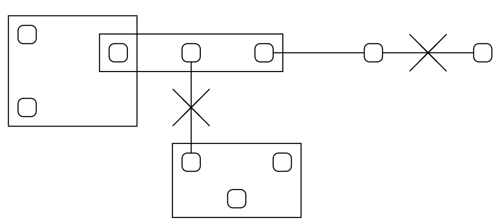
At each iteration:

— if the maxset is selected, either it has the initial node $1, n$ and is added n to the others, or it has n in the first node and is added $1, n$ to one of the others and n to the remaining ones; this ensures that it contains at least a label $1, n$ and the others are all n;
— if the maxset is excluded, it ends up with all labels $n > 1$, or with a single label $1, n$ and the others $n + 1$.

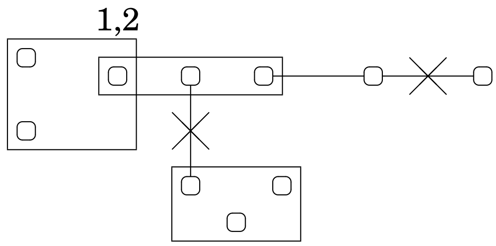
Either way, a set may contain a label $1, n$ only if it is the initial label, and then no other $1, m$ is ever added, or it is added in a single node of a selected set that has n has the initial label.

Finally, a label $1, n$ is added only in a single case: a selected set, if the initial node is labeled n. As a result, every $1, n$ is in a selected set that contains n has the other labels. $\square$

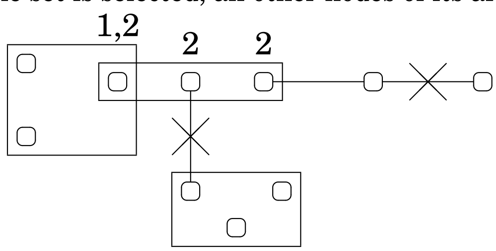
As an example, the algorithm is shown on the following set of sets, where lines denotes binary sets and boxes ternary sets; a cross indicates an excluded set, the other sets are included.



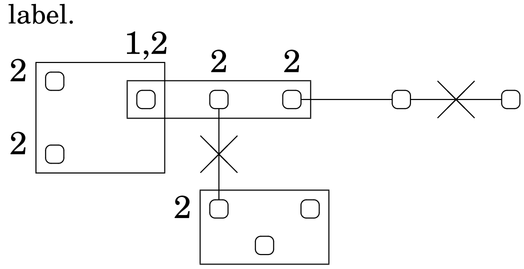
The central three-node set can be taken as the root of a tree-like structure of sets. Since this set is selected, an arbitrary node of it is labeled $1, 2$.



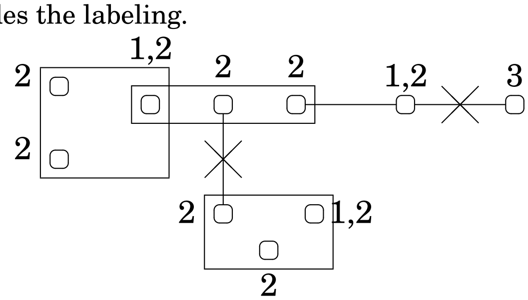
The algorithm now proceed with its iterative step. Since a node of the central set is labeled $1, 2$ and the set is selected, all other nodes of its are labeled 2.



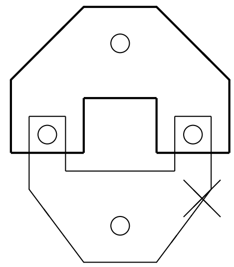
The three-node set on the left is labeled in the same way. The vertical segment representing a binary set is instead excluded. Since one of its nodes is labeled 2, the other takes the same label.



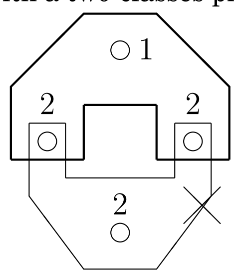
The two horizontal segments are labeled in different ways. The first is selected and one of its nodes is labeled 2; therefore, the other takes $1, 2$. The second is excluded and one of its nodes is labeled $1, 2$; therefore, the other takes 3. The set at the bottom has only the label 2 and is selected; therefore, one of its other nodes is labeled $1, 2$, the other 2. This concludes the labeling.



While Berge-acyclic hypergraphs are obtainable, the converse is not always the case: some Berge-cyclic hypergraphs are obtainable. Contrasting Theorem 4.20 (at the end of the next Section 4.3), which proves that alternating cycles imply unobtainability for binary maxsets, in the general case alternating cycles may be obtainable:



The maxset on the top is selected, the other excluded. This hypergraph is Berge-cyclic, yet is obtained with a two-classes priority ordering:



4.3. Binary maxsets

A particular case of the problem of obtainability by priority base merging is when maxsets comprise at most two formulae. This may be guaranteed to hold in a specific domains, but the main reasons for studying this case are: first, it provides proofs of existence of some specific cases, such as one requiring n classes of priority for obtainability; second, it is a subcase where a necessary and sufficient condition for obtainability can be given, that of alternating cycles of maxsets; third, it provides guiding principles for a future study of the general case, where no such necessary and sufficient condition has been found.

When all maxsets comprise at most two formulae, they can be seen as a graph:

— nodes are formulae;
— isolated nodes are singleton maxsets;
— edges are maxsets of two formulae.

Lemma 4.6 ensures that the contrary also holds: every graph corresponds to the maxsets of some formulae. As a result, properties of graphs carry to sets of maxsets.

The analysis of the subcase of binary maxsets is long and requires a number of intermediate steps. The formal proofs are in the electronic appendix, this section only summarizes the main results.

Definition 4.14. A cycle is a path ending in the same node where it started.

This is different from the definition of simple cycles, which are not allowed to cross an edge more than once.

When all maxsets contain at most two formulae, the singletons can be excluded from consideration because of Lemma 4.1: $\{A\}$ cannot be contained in any other maxset; therefore, inclusion or exclusion do not affect the other maxsets. What remains is a set binary maxsets, which can be seen as a graph where nodes are formulae and edges are maxsets. Some edges correspond to selected maxsets, the others to excluded maxsets.

Definition 4.15. A *selected-excluded graph* (abbreviated: *se graph*) is a graph whose edges are partitioned in two sets: selected and excluded.

Since edges are maxsets, the distinction indicates which are required to be in the result of merging and which are not. Most of the proofs regarding binary maxsets employ assignments of some formulae to priority class.

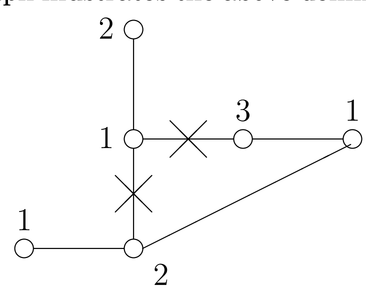
Definition 4.16. A *partially assigned se graph* has some nodes assigned positive integer values. If all nodes are assigned the graph is *totally assigned*.

In a totally assigned se graph, all formulae are assigned a class. Therefore, one may determine the minimal edges (i.e., the edges that correspond to minimal maxsets) and check whether they are exactly the selected ones.

Definition 4.17. A totally assigned graph is obtainable if the minimal edges according to the priority ordering obtained from the numbers assigned to the nodes are exactly the selected ones.

This definition may look tautological, but is rather close to the opposite. In a se graph, the selected edges are the maxsets that are required to be in the result of merging: if $\{A, B\} \models R$, the edge (A, B) is selected and vice versa. The values assigned to nodes may or may not make such a maxset minimal. If it is not, the edge is *incorrectly excluded*. Similarly, an excluded edge that is minimal according to the values is incorrectly selected. If no edge is incorrectly selected or excluded the ordering produces the required result.

The following graph illustrates the above definitions:



This is a se graph since some edges are marked as excluded (the crossed ones); the others are selected. It is also totally assigned since each node is assigned a number (its class in the priority ordering). It would be obtainable if every selected edge were minimal and every excluded edge were not. This is not the case, as the vertical excluded edge is minimal. Indeed, it is less than the edge of values 1 and 3 and incomparable to the others; for example, the top edge has in common with it the node of value 1, but the two nodes of value 2 are different.

This graph would be obtainable if the value of the bottom node were 3 instead of 2; the bottom edge would remain minimal as it shares its node of value 1 with no other edge. Graphically, obtainability means that the non-crossed edges are exactly the minimal ones according to the numbers.

Definition 4.18. A partially or totally assigned se graph G extends another one H if they have the same nodes and edges and all nodes assigned in H are also assigned in G to the same values.

A se graph is therefore obtainable if and only if it can be extended to a totally assigned se graph that is obtainable. On totally assigned se graphs obtainability can be checked by determining the minimal maxsets according to the ordering given by the values.

Se graphs can be simplified without affecting obtainability: the resulting graph is obtainable if and only if the original one is. In particular:

— Disconnection: certain edges can be removed or replaced by edges between one of their original nodes and an isolated copy of the other;
— Merging: certain edges can be merged by identifying their nodes pairwise; certain nodes can be merged;
— Full disconnection: a node that is only touched by excluded edges can be replaced by a node for each of these edges;
— Tail removal: a chain of edges that do not participate in any cycle can be removed;
— Zigzag folding: a path of selected edges is turned into a single edge by merging all nodes in even positions and all in odd positions;

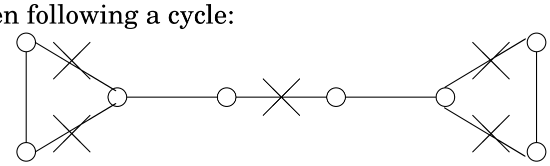
The following consequences can be drawn:

— in any obtainable total assigned se graph containing a triangle of selected edges, the nodes of the triangle have value one;
— in any obtainable total assigned se graph containing a chain of alternating excluded-selected edges with the first node assigned one, the values of the other even nodes are one and the values of the odd nodes are strictly increasing;
— for every n there exists a graph that is only obtained by assignments with at least n different values;
— a graph containing a cycle of alternating (single excluded edge)–(chain of odd selected edges) is unobtainable;
— a graph is obtainable if and only if the result of applying full disconnection, removal of tails and zigzag folding as far as possible is an empty graph;

The third result correspond to the following theorem, when carried back to maxsets and formulae.

THEOREM 4.19. *For any n, there exists R and $K_1, \ldots, K_m$ such that R is obtainable by priority base merging from $K_1, \ldots, K_m$ only with priority partitions having n classes or more.*

Turning the second-last of these results into a necessary and sufficient condition requires keeping into account that some edges can be crossed twice in the opposite directions when following a cycle:



This graph is unobtainable, but its only alternating cycle crosses the chain of three edges in the middle twice, once left-to-right and once right-to-left. The main result regarding binary maxsets is: a graph is unobtainable if and only if it contains a cycle of alternating (single excluded edge)-(chain of odd selected edges) that crosses the same edge at most twice.

Expressed in terms of maxsets and formulae, this fact is the following theorem.

THEOREM 4.20. *Formula R is unobtainable from a set $K_1, \ldots, K_m$ having no maxset of size greater than two if and only if a cycle of (single maxset not in R)-(chain of odd maxsets in R) that crosses the same maxset at most twice exists.*

4.4. Algorithm

Theorem 4.13 ensures that every or-of-maxsets is obtainable if the maxsets form a Berge-acyclic hypergraph. The following algorithm combines the method for iteratively labeling formulae with the search for maxsets. It is guaranteed to work if the maxsets form a Berge–acyclic hypergraph, but may produce a correct result even if they do not.

ALGORITHM 2.

(1) *for each pair of formulae* K_i, K_j*, determine their mutual consistency*
(2) *set* $L = \emptyset$
(3) $M = \{K_i, K_j\}$*, where* $\{K_i, K_j\}$ *is consistent,* $K_i \in L$ *and* $K_j \notin L$*; if such a pair does not exists (e.g.,* $L = \emptyset$*) then* $M = \{K_i\}$ *with* $K_i \notin L$*; if* L *contains all* K_i*'s, stop*
(4) *choose* K_j *such that* $\{K_i, K_j\}$ *is consistent for every* $K_i \in M$*; if no such* K_j *exists, go to Step 7*
(5) *if* $M \cup \{K_j\}$ *is inconsistent, go Step 4 and choose another* K_j
(6) $M = M \cup \{K_j\}$ *and go to Step 4*
(7) $L = L \cup M$
(8) *if* $M \models R$*, then:*
  (a) *if no formula of* M *is labeled, then label one with* $1, 2$ *and the others with* 2*;*
  (b) *if a formula is labeled* $1, n$ *and the others are unlabeled, label the others* n
  (c) *if a formula is labeled* n *and the others are unlabeled, label one of the others* $1, n$ *and the others* n
  (d) *otherwise, the set of maxsets is not acyclic: terminate with error*
(9) *if* $M \not\models R$
  (a) *if no formula of* M *is labeled, label all of them* 2
  (b) *if a formula is labeled* $1, n$ *and the others are unlabeled, label the others* $n + 1$
  (c) *if a formula is labeled* n *and the others and unlabeled, label the others* n
  (d) *otherwise, the set of maxsets is not acyclic: terminate with error*
(10) *go to Step 3*

If a formula is labeled $1, n$ *its priority class is one; if it is labeled* n*, it is* n*. If the result of merging with this priority ordering is* R*, then* R *is obtainable.*

The final check is necessary unless R is guaranteed to be an or-of-maxsets. The algorithm includes some choices (e.g., "choose K_j", "label one node with 1,2") but is not nondeterministic: arbitrary choices can be taken.

Entailment $M \models R$ can be replaced with the consistency of $M \cup \{R\}$. The algorithm can be improved by caching the inconsistent sets $M \cup \{K_j\}$ detected in Step 5, especially the small ones. This information can be useful when later checking another $M' \cup \{K_j\}$: if $M \cup \{K_j\} \subseteq M' \cup \{K_j\}$, unsatisfiability is established at no additional cost.

THEOREM 4.21. *If the maxsets of* $K_1, \ldots, K_m$ *are Berge-acyclic, Algorithm 2 establishes the obtainability of* R *from them and outputs a priority ordering that generates* R *if one exists.*

Proof. The algorithm works by iteratively generating a new maxset M from a labeled formula, and then labeling its other formulae according to the rules of Theorem 4.13.

In particular, during the algorithm the following conditions hold:

— all formulae of the maxsets found so far are labeled;
— L is the union of the maxsets found so far;
— M is a subset of a maxset not (yet) in L.

At the beginning these conditions are vacuously true, as no maxset has been found and no formula is labeled. No step violates them: Step 3 guarantees that every generated M is a new maxset, as it is built upon at least a formula that is not in the previous ones; Step 7 is reached only when M is a maxset, ensuring the validity of the first of three conditions; the two following steps label the formulae of this newly found maxset.

Since labeling is performed as in Theorem 4.13, if the set of maxsets is acyclic and R is an or-of-maxsets, the result is a priority ordering generating R. $\square$

If the maxsets are not Berge-acyclic, the algorithm stops when it reaches a maxset that already contains two or more labels. In some cases, there is no way it could continue. For example, there is no way to extend labels $1, n$ and $1, m$ with $n \neq m$ to the rest of a selected maxset. In the other cases, such as two labels greater than one, the algorithm may still continue and obtain a correct ordering.

4.5. Complexity

A necessary condition to obtainability is that the formula to obtain is the disjunction of some maxsets of the formulae to be merged. An obvious way to check this is to consider all possible sets of subsets of formulae, checking that each of them is maximally consistent, and that their disjunction is equivalent to the result to obtain. However, the problem can be reformulated in a much simper way using some properties of maxsets.

LEMMA 4.22. *Formula R is an or-of-maxsets of $K_1, \ldots, K_m$ if and only if, for every $I \models R$, it holds $M \models R$ and $M \cup \{K_i\} \models \bot$ for every $K_i \notin M$, where $M = \{K_i \mid I \models K_i\}$.*

Proof. By Lemma 4.1, maxsets do not share models. Therefore, if R is an or-of-maxsets then each of its models is in exactly one maxset. In particular, Lemma 4.2 tells that $M = \{K_i \mid I \models K_i\}$ is the maxset containing I, if any. The additional conditions ensure that M is actually a maxset (no other formula is consistent with it) and that the disjunction of such M's do not include models not in R. $\square$

As a consequence of this property, checking whether R is an or-of-maxsets is not harder than propositional entailment.

THEOREM 4.23. *Checking whether R is an or-of-maxsets of $K_1, \ldots, K_m$ is in* coNP.

Proof. Let X be the set of variables. By Lemma 4.22, the property can be checked by considering each model I over X, building $M = \{K_i \mid I \models K_i\}$ and verifying a number of independent entailments: $M \models R$ and $M \cup \{K_i\} \models \bot$ for every $K_i \notin M$. Since M can be built in polynomial time from I, the subproblem is equivalent to a single validity check, and can therefore be expressed in terms of a QBF in the form $\forall Y.F$. Since the whole problem is to check this for every model I over X, it is equivalent to $\forall X \forall Y.F$, and is therefore in coNP. $\square$

Hardness holds even with only two formulae to be merged.

THEOREM 4.24. *Checking whether R is an or-of-maxsets of a set of two formulae is* coNP-*hard.*

Proof. The claim is proved by reduction from the problem of establishing the unsatisfiability of a formula F. Reduction is as follows: formula F is inconsistent if and only if $R = \neg c$ is an or-of-maxsets of $A = \neg c$ and $B = c \vee (d \wedge F)$, where c and d are two new variables, not occurring in F.

Regardless of the consistency of F, resolution turns $A \wedge B$ into $\neg c \wedge d \wedge F$. As result, if F is inconsistent so is $A \wedge B$. Therefore, the maxsets are $\{A\}$ and $\{B\}$. Since R is the same as A, it can be seen as the disjunction of the single element $\{A\}$.

If F is consistent, so is $A \wedge B$. Therefore, the only maxset is $\{A, B\}$, which is equivalent to $A \wedge B = \neg c \wedge d \wedge F$. Model $\{c = \mathsf{false}, d = \mathsf{false}\}$ falsifies this formula while satisfying R. Therefore, R is not an or-of-maxsets. $\square$

These results do not require R to be consistent. If it is not, R is still an or-of-maxsets, as $\bigvee \emptyset = \bot$. However, this case is not allowed as a result of merging: an inconsistent formula is never obtainable.

By Lemma 4.12, if the formulae are three or less then every consistent or-of-maxset is obtainable. By definition, obtainable formulae are or-of-maxsets. Therefore, the last theorem also proves the complexity of obtainability in this case.

COROLLARY 4.25. *Checking whether a consistent formula is obtainable by priority base merging from two formulae is* coNP*-hard.*

Unfortunately, Theorem 4.23 does not extend to obtainability. Indeed, while verifying whether a formula is an or-of-maxsets can be done “locally”, by checking each model I and its maxset M at time, obtainability is a global conditions over the maxsets: for a given ordering, a maxset may be minimal or not depending on the others. This makes the problem harder than checking whether a formula is an or-of-maxsets.

THEOREM 4.26. *Checking whether a formula is obtainable by priority base merging is in* Σ_3^p.

Proof. By Lemma 4.2, for every model I of a maxset M it holds $M = \{K_i \mid I \models K_i\}$. This provides a way for expressing the problem of obtainability of R from $K_1, \ldots, K_m$: there exists a priority ordering P such that every model of R corresponds to a minimal maxset and every model of $\neg R$ corresponds to a subset that is either non-minimal or not a maxset at all.

For every model I of R the set $M = \{K_i \mid I \models K_i\}$ has to be a minimal maxset. By Lemma 4.4, this is equivalent to M being not greater than another consistent subset N. In other words, for every $N \subseteq \{K_1, \ldots, K_m\}$ either N is inconsistent or it is not less than M according to P. Comparing according to P can be done in polynomial time, as it amounts to checking which formulae of M and N are in $P(1)$, $P(2)$, etc. The quantifiers in these conditions are all universal; therefore, the subproblem can be expressed as a $\forall QBF$.

For every model I that does not satisfy R, the set $M = \{K_i \mid I \models K_i\}$ should not be a minimal maxset according to P. This means that M can be either not a maxset or not a minimal one according to P.

(1) Since all $K_i \in M$ satisfy I, the set M is consistent. As a result, it is not a maxset only if it can be added some formulae without violating consistency. This is equivalent to the consistency of $M \cup \{K_j\}$ for some $K_j \notin M$. This case can therefore be expressed as a $\exists QBF$.
(2) In the other case, M is a maxset but is not minimal. By Lemma 4.4, this is equivalent to the existence of a consistent subset $N \subseteq \{K_i\}$ that is less than M according to the ordering. This subset N needs not to be a maxset. Therefore, what is to be checked is only the existence of a subset N that is consistent and that is less than M according to the ordering. This case can therefore be expressed as a $\exists QBF$.

Both conditions can therefore be expressed as a $\exists QBF$. They have to hold for every model I that does not satisfy R. This corresponds to adding a universal quantifier (over I) to the front of the QBF, which therefore becomes a $\forall\exists QBF$. The first subproblem was expressed as $\exists QBF$; therefore, it can also be expressed as $\forall\exists QBF$. This is the check to be done for a single priority ordering P. Since the problem is to establish the existence of an ordering satisfying these conditions, the whole problem is expressed as a $\exists\forall\exists QBF$, and is therefore in Σ_3^p. □

The following result shows that even with four formulae (the smallest case of unobtainable consistent or-of-maxsets) obtainability is coNP-hard even if the formula is assumed to be a consistent or-of-maxsets.

THEOREM 4.27. *Checking whether R is obtainable by priority base merging from four formulae is* coNP*-hard, and this result holds even assuming that R is a consistent or-of-maxsets.*

Proof. The claim is proved by reduction from propositional unsatisfiability. By Lemma 4.10, $R = (A \wedge B) \vee (C \wedge D)$ is not obtainable from A, B, C, D if the maxsets are $\{A, B\}$, $\{B, C\}$, $\{C, D\}$ and $\{D, A\}$. Lemma 4.6 gives the following formulae:

— $A = (x \wedge y) \vee (\neg x \wedge \neg y)$
— $B = (x \wedge y) \vee (x \wedge \neg y)$
— $C = (x \wedge \neg y) \vee (\neg x \wedge y)$
— $D = (\neg x \wedge y) \vee (\neg x \wedge \neg y)$

The maxset $\{D, A\}$ is equivalent to $\neg x \wedge \neg y$. A formula F can be added to it by changing D and A:

— $A' = (x \wedge y) \vee (\neg x \wedge \neg y \wedge F)$
— $B = (x \wedge y) \vee (x \wedge \neg y)$
— $C = (x \wedge \neg y) \vee (\neg x \wedge y)$
— $D' = (\neg x \wedge y) \vee (\neg x \wedge \neg y \wedge F)$

This provides the required reduction from propositional unsatisfiability to obtainability. Indeed, if x and y are two new variables, not occurring in F, then F is unsatisfiable if and only if $R = (A' \wedge B) \vee (C \wedge D')$ is obtainable from A', B, C, D'.

The maxsets of the four formulae are $\{A', B\} \equiv x \wedge y$, $\{B, C\} \equiv x \wedge \neg y$, $\{C, D'\} \equiv \neg x \wedge y$ and, if F is consistent, $\{D', A'\} \equiv \neg x \wedge \neg y \wedge F$. As a result, if F is consistent then maxsets are as in Lemma 4.10, and R is therefore unobtainable. Otherwise, there are only three maxsets, and R is the disjunction of two of them. Lemma 4.12 ensures that every or-of-maxsets is obtainable in this case. □

Obtainability depends on the existence of orderings over the maxsets, which may be exponentially many. This number reduces to quadratic if the maxsets comprise at most two formulae.

THEOREM 4.28. *Checking whether a consistent or-of-maxsets is obtainable by priority base merging is in* coNP *if all maxsets comprise at most two formulae.*

Proof. The result is unobtainable if the graph of maxsets is unobtainable, which by Theorem A.21 is equivalent to the presence of an alternating cycle. Since the nodes are formulae, this condition can be reformulated as: there exists a sequence of formulae $A_1, B_1, A_2, B_2, \ldots$, each appearing at most twice, such that:

(1) every pair of consecutive formulae is consistent: $A_i \wedge B_i \not\models \bot$, $B_i \wedge A_{i+1} \not\models \bot$, $\ldots$; checking that such pairs are also maximally consistent is unnecessary by the assumption that no maxset contains more than two formulae;
(2) $A_i \wedge B_i \wedge R \not\models \bot$: by Lemma 4.9, this is equivalent to $\{A_i, B_i\}$ being selected;
(3) either $B_i \wedge A_{i+1} \not\models R$ or $A_{i+1} \wedge B_{i+1} \wedge R \not\models \bot$; still by Lemma 4.9, this condition is equivalent to: if $\{B_i, A_{i+1}\}$ is selected, so is $\{A_{i+1}, B_{i+1}\}$.

Selection can be expressed both as $M \models R$ and $M \wedge R \not\models \bot$. Using the first condition when the requirement is negated and the second when it is positive allows expressing unobtainability in terms of non-entailment only. In particular, it is reformulated as the existence of such a cycle that satisfies a number of conditions based on non-entailment. Therefore, unobtainability is in NP, and obtainability in coNP. □

This allows for a precise characterization of complexity for the case of binary maxsets.

COROLLARY 4.29. *Checking whether a consistent or-of-maxsets is obtainable by priority base merging is* coNP *complete if all maxsets comprise at most two formulae.*

4.6. Constant number of formulae

For technical reasons, obtainability is extended to pairs (S, E) where both S and E are sets of sets of formulae. Such a pair is obtainable if there exists a priority ordering such that the sets in S are exactly the minimal ones among $S \cup E$. In other words, (S, E) is obtainable if there exists an ordering that makes the sets in S to be the minimal ones among $S \cup E$.

Obtainability can be defined from this concept: R is obtainable if $R \equiv \bigvee S$, (S, E) is obtainable and $S \cup E$ is the set of all maxsets of $K_1, \ldots, K_m$. Obtainability of pairs can therefore be considered an extension of the usual concept of obtainability where the condition that $S \cup E$ is the set of maxsets is lifted. In a way, (S, E) is obtainable if $\bigvee S$ is obtainable from a set having $S \cup E$ as its sets of maxsets.

The following lemma concerns the obtainability of a pair (S, E), where S and E are sets of sets of formulae, not necessarily maxsets and not necessarily all of them. Unobtainability is monotonic with respect to the excluded sets: adding new ones and enlarging the existing ones does not change unobtainability.

LEMMA 4.30. *If S and E are sets of sets such that none is contained in another and (S, E) is not obtainable so is (S, E'), where E' is the result of adding some sets of formulae to E and some formulae to some sets of E.*

Proof. Given the assumption of no mutual containment, every pair $(S, \emptyset)$ is obtainable by placing all formulae of S in class one. Therefore, unobtainability is due to the presence of E: every partition that makes all sets in S minimal also makes minimal some $N \in E$. This means that for every $M \in S$, the set M is not less than N according to the ordering. The two sets N and M coincide up to class $i - 1$ but $N \cap P(i) \not\subseteq M \cap P(i)$ for some class i, possibly $i = 1$. Adding formulae to N or new sets to E do not change this condition. □

Obtainability can be defined as follows: there exists a set S such that the result is equivalent to $\bigvee S$, S is a subset of maxsets and (S, E) is obtainable, where E are the maxsets not in S. In the case of a constant number of formulae, their sets and therefore maxsets are in constant number as well. Quantifying over them does not therefore increase the complexity of the problem.

However, the remaining quantifiers are not all of the same kind. For example, the condition that R is an or-of-maxsets is:

$$
\begin{gathered}
R \text{ is an or-of-maxsets of } \{K_1, \ldots, K_m\} \\
\Updownarrow \\
\exists S \subseteq 2^{\{K_1, \ldots, K_m\}} \text{ such that } \begin{array}{l} R \equiv \bigvee S \\ \forall M \in S \,.\, M \not\models \bot \text{ and } \forall K \not\in M \,.\, M \cup \{K\} \models \bot \end{array}
\end{gathered}
$$

The quantifiers over S, M and K are not a problem because the choices are on sets of constant cardinality. Instead, $M \not\models \bot$ is an existential quantification (there exists a model satisfying all formulae of M) and all others are universal (e.g. all models satisfying M also satisfy $\bigvee S$).

These quantifiers can be removed by relaxing the condition over M, that is, accepting some sets other than M. This is the technique used by Nebel [1998] for the generalized closed-world assumption (GCWA) and the WIDTIO revision: instead of considering only the sets specified by the definition, include some others others that do not affect the final result.

Omitting details, $GCWA(T)$ is T with a certain set of literals F added; what made determining the exact complexity of the problem $GCWA(T) \models A$ difficult was that checking membership of a single literal in F is already Π^p_2-hard, thus requiring a polynomial number of calls to a Π^p_2 oracle for verifying $T \cup F \models A$. Nebel [1998] overcame this difficulty by switching from F to its supersets: $T \cup F \not\models A$ if and only if $T \cup S \not\models A$ for some $S \supseteq F$. In spite of the seeming increase of complexity, the problem is simplified because checking whether $S \supseteq F$ is in Σ^p_2. Therefore, the whole non-entailment problem is in Σ^p_2, as it amounts to guess a set S satisfying a condition in Σ^p_2 and a model that satisfies $T \cup S$ but not A. In a nutshell, the core of the method is:

> instead of the specific set F use a group of sets that includes it, provided that the other sets do not affect the final result.

In the present case, the key point is that inconsistent sets in S are irrelevant: if S contains an inconsistent set M, then $\bigvee S = \{M\} \vee \bigvee(S\backslash\{M\}) = \bot \vee \bigvee(S\backslash\{M\}) = \bigvee(S\backslash\{M\})$; inconsistent sets do not contribute to the disjunction. As a result, the condition can be relaxed by allowing such sets M: requiring that M is a maxset is changed into just $M \cup \{K\} \models \bot$ for every $K \not\in M$. The M's satisfying this condition are either maxsets or inconsistent sets of formulae, but the latter do not affect $\bigvee S$.

$$
\begin{array}{c}
R \text{ is an or-of-maxsets of } \{K_1, \ldots, K_m\} \\
\Updownarrow \\
\exists S \subseteq 2^{\{K_1,\ldots,K_m\}} \text{ such that} \\
\begin{array}{l}
R \equiv \bigvee S \\
\forall M \in S \; \forall K \not\in M \; . \; M \cup \{K\} \models \bot
\end{array}
\end{array}
$$

This condition contains only universal quantifiers: $R \equiv \bigvee S$ is equivalent to "every model satisfying R also satisfies $\bigvee S$ and vice versa"; $M \cup \{K\} \models \bot$ is "every model falsifies $M \cup \{K\}$". The quantifiers over S, M and K are choices over sets of constant cardinality, so they do not affect complexity. They can be replaced by conjunctions and disjunctions.

As a result, checking whether R is an or-of-maxsets is in coNP for a constant number of formulae. This fact is subsumed by Theorem 4.23, which states the same for any number of formulae. However, with some changes the condition extends to obtainability, for which no similar result hold in the general case. Lemma 4.30 ensures the correctness of relaxing.

LEMMA 4.31. *R is obtainable by priority base merging from $K_1, \ldots, K_m$ if and only if there exists a nonempty $S \subseteq 2^{\{K_1,\ldots,K_m\}}$ such that:*

(*1*) *$R \equiv \bigvee S$;*
(*2*) *$\forall M \in S$, $\forall K \not\in M$, $M \cup \{K\} \models \bot$;*
(*3*) *$\forall E \subseteq 2^{\{K_1,\ldots,K_M\}}$, either $\exists M \in E$ such that $M \models \bot$ or $\exists M \in E$ such that $M \subseteq M'$ for some $M' \in S$ or (S, E) is obtainable.*

Proof. The first two points are equivalent to R being an or-of-maxsets. The third resembles the definition of obtainability, but E is not the set of maxsets not in S. Rather, if the condition is false is an arbitrary set of consistent subsets such that (S, E) is not obtainable.

Lemma 4.30 however ensures that if such a set E is enlarged by adding arbitrary new sets and arbitrary new formulae to existing sets, the pair (S, E) remains unobtainable. As a result, if there exists E such that (S, E) is unobtainable, E can be added formulae and sets to make it the set of maxsets not in S.

— R obtainable. The three conditions above hold for S equal to the set of selected maxsets. This choice makes the first and second points true. If the third point were false, then (S, E) would be unobtainable for some set of consistent sets E such that none of its element is contained in one of S. Since an $N \in E$ is not contained in a selected maxset, it can be enlarged to make it a maxset, and that would be an excluded one. Adding the other excluded maxsets, E is turned into the set of excluded maxsets E'. By Lemma 4.30, since (S, E) is unobtainable so is (S, E'), contradicting the assumption that R is obtainable.

— R unobtainable. If R is not an or-of-maxsets, then for no S points 1 and 2 hold. Otherwise, R is an or-of-maxsets S but (S, E) is not obtainable, where E is the set of the other maxsets. For such E the third point of the condition is violated.

□

The conditions in this lemma only contain universal quantifier, apart the ones on sets of constant size. The complexity of the problem is the obvious consequence of this.

COROLLARY 4.32. *Checking obtainability by priority base merging from a constant number of formulae is in* coNP.

Once obtainability is established, the problem is to find the ordering generating the result. This problem can be recast as that of checking whether a partial assignment of formulae to classes can be extended to form an ordering generating the required result of merging.

THEOREM 4.33. *Checking whether a priority ordering can be extended to generate R as the result of merging a constant number of formulae $K_1, \ldots, K_m$ is* coNP *complete.*

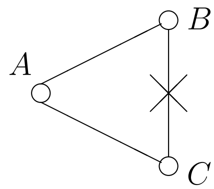
Proof. The problem is hard with an empty ordering, as it is equivalent to obtainability. It is also in coNP: it is the same as obtainability by adding the condition that the ordering extends the given one. In the statement of Lemma 4.31, the only point where the ordering matters is when (S, E) is checked to be obtainable. Therefore, the problem can be expressed by simply changing the subcondition "(S, E) is obtainable" into "(S, E) is obtained by an ordering extending the given partial one" in the statement of Lemma 4.31. Since the additional check has cost linear in the number of the formulae, complexity remains the same. □

A related question is whether a priority ordering can be uniquely extended to generate the required result. This amounts to finding such an ordering, if any, and then checking that no other priority ordering would do the same.

THEOREM 4.34. *Checking whether a priority ordering not extending a given one generates R as the result of merging a constant number of formulae $K_1, \ldots, K_m$ is* coNP-*complete.*

Proof. Lemma 4.31 expresses this problem by changing the condition that (S, E) is obtainable to its obtainability with an ordering not extending the given one. This proves that the problem is in coNP.

Hardness is proved using three formulae with maxsets $\{A, B\}$, $\{A, C\}$, and $\{B, C\}$, where the latter is excluded and only exists if a formula F is satisfiable.



If the third maxset exists, the only ordering excluding it while selecting the other two is the one containing A in class one and B and C in class two. Indeed, if both B and C are in class one, by Lemma 4.7 $\{B, C\}$ would be selected. If A and B are in class one and C is not, $\{A, C\}$ would be excluded. Since either A or B is in class one by Lemma 4.8, the only remaining case is A in class one. The other two formulae B and C cannot be in different classes, as otherwise one between $\{A, B\}$ and $\{A, C\}$ would be excluded. Therefore, the only ordering obtaining the required result has A in class one and B and C in class two.

The same ordering selects the same two maxsets even if the third maxset does not exists. Since the result is the disjunction of all maxsets, Lemma 4.7 applies: it is also obtained by placing all three formulae in class one. Therefore, a second ordering selects $\{A, B\}$ and $\{A, C\}$ in this case.

The problem is therefore that of generating formulae such that $\{B, C\}$ is consistent if and only if a formula F is. Lemma 4.6, with F added to $\{B, C\}$, gives:

$$\begin{aligned} A &= (x \wedge \neg y) \vee (\neg x \wedge \neg y) \\ B &= (x \wedge \neg y) \vee (x \wedge y \wedge F) \\ C &= (\neg x \wedge \neg y) \vee (x \wedge y \wedge F) \end{aligned}$$

The set of all three formulae is inconsistent, as A is only satisfied by partial models $\{x = \mathsf{true}, y = \mathsf{false}\}$ and $\{x = \mathsf{false}, y = \mathsf{false}\}$, while C is falsified by the first and B by the second. Pairs of formulae are all consistent:

$$\begin{aligned} \{A, B\} &= x \wedge \neg y \\ \{A, C\} &= \neg x \wedge \neg y \\ \{B, C\} &= x \wedge y \wedge F \end{aligned}$$

The third is consistent if and only if F is consistent. As a result, the maxsets $\{A, B\}$ and $\{A, C\}$ always exist, and are selected when the required result is $R = \neg y$ because they are consistent with it. The third maxset $\{B, C\}$ only exists if F is consistent, and if this is the case is excluded because it is inconsistent with R.

As shown before, R is uniquely obtainable if and only if $\{B, C\}$ is not a maxset, which is equivalent to the inconsistency of F. As a result, unique obtainability is coNP-hard. □

5. WHAT TO DO IN CASE OF UNOBTAINABILITY

After establishing obtainability, the next step is to determine the weights or the priority ordering. The algorithms in Section 3.2 and Section 4.4 searches for them, but of course cannot find anything in case of unobtainability. A question therefore remains: what to do in this case?

Various possibilities exist. One is to relax the condition that R is exactly the outcome of merging, still maintaining that R is a formula that is known to be true. Lifting equivalence and only requiring consistency is coherent with this principle: R does not discriminate among its models, so each could be the actual state of the world. The result of merging may therefore only be required to have one such model. In other words, it is expected that merging produces a formula consistent with R rather than equivalent to it.

LEMMA 5.1. *There exists a priority partition such that merging $K_1, \ldots, K_m$ is consistent with R if and only if R is consistent with one of the maxsets of $K_1, \ldots, K_m$.*

Proof. If one of the maxsets is consistent with R, the ordering of Lemma 4.8 allows selecting it only. The result of merging is equal to this maxset, which is by assumption consistent with R.

In the other way around, if R is consistent with the result of merging $K_1, \ldots, K_m$ with some ordering, since this result is the disjunction of some of the maxsets, then R is consistent with at least a maxset. □

Even when merging is not supposed to be a process of search of a single propositional model, a similar idea can be applied. Assuming that the situation is characterized by a set of models, both the result of merging and R result from bounding it as close as possible. The difference is that R is known to be correct, so it contains all these models, while merging only aims at doing the same. Under this assumption, the problem is to find an ordering such that the set of models of R is strictly contained in the result of revision. Since what is known about this set is only that R contains it, the result of merging should be implied by R. Unfortunately, this condition does not constraint the ordering at all.

LEMMA 5.2. *Merging $K_1, \ldots, K_m$ with some priority ordering is entailed by R if and only if R entails the disjunction of all maxsets.*

Proof. If R entails the disjunction of all maxsets, such a disjunction can be obtained as the result of the revision by the ordering in Lemma 4.7. Vice versa, if R entails the result of merging $K_1, \ldots, K_m$ with some ordering, since this result is the disjunction of some maxsets, then R also entails the disjunction of all maxsets. □

Requiring that R is entailed by the result of merging or consistent with it gives no information about the relative reliability of the sources. To obtain such an information some additional constraint is needed, such as R being as close as possible to the result of merging, possibly also implying or being consistent with it. In other words, the aim moves from obtaining R with the appropriate priorities to approximating it as much as possible.

If a result is unobtainable, another possible line of action is to consider whether the given pieces of knowledge produce it using a different merging mechanism. In other words, instead of using merging by priorities, one of the many other systems [Konieczny and Pérez 2011; Peppas 2008; Konieczny et al. 2004; Everaere et al. 2010; Jin and Thielscher 2007; Liberatore and Schaerf 1998] may be employed instead.

Another possible solution is to split the sources on their variables. If a renowed computer scientist tells some property of computational classes and that the fastest way to go a certain restaurant is to turn left at the next turn, the first information should be assigned higher priority than the second, as there is no a priori reason why an expert in computing should know the roads better than anyone else. According to this principle, when a result is not obtainable some source K_i may be split into $\{K_i^1, \ldots, K_i^r\}$, for example using a partition of the variables to decide which part of K_i goes into K_i^1, which in K_i^2, etc.

A totally different direction is to lift the assumption that R is a formula known with certainty. Instead, it could be just a formula coming from a source of high reliability.

Table I. Main theorems on obtainability

— necessary and sufficient condition for obtainability from two knowledge bases using the weighted sum of distances: Theorem 3.1
— every consistent or-of-maxsets is obtainable by priority base merging if the maxsets are less than four: Theorem 4.12
— every disjunction of a nonempty subset of a set of maxsets that is Berge-acyclic is obtainable by priority base merge: Theorem 4.13
— for every n there exists a formula that requires a priority of n levels or more to be obtained from a set of binary maxsets: Theorem 4.19
— necessary and sufficient condition for obtainability from formulae having all binary maxsets: Theorem 4.20

Obtainability then generalizes to the case where no such source may be available [Liberatore 2014].

A solution suggested by one of the anonymous referees is to extend the set of knowledge bases $\{K_1, \ldots, K_m\}$ by some other formulae $K_{m+1}, \ldots, K_r$. Such an addition is motivated if some information has been neglected, for example because it is not expressed explicitly or because it has wrongly been considered irrelevant. Of course, some constraints on the new formulae has to be specified to avoid trivializing the problem.

Even with all these alternatives, it is still possible that the known information R cannot be obtained from the knowledge bases. For example, no semantics allows obtaining $R = x$ from $K_1 = \neg x$ and $K_2 = \neg x$. This is however a rational outcome: if the knowledge bases totally agree, merging should produce them as the result, no matter by which weights, priorities or other relative reliability measures; this intuition is formalized by postulate IC2 of merging [Konieczny and Pérez 2011]. If x is true, then two knowledge bases equal to $\neg x$ are just useless. Unobtainability provides significant information even in this case: the sources are unreliable, and can therefore be ignored from this point on.

6. CONCLUSIONS

In this article, the problem of establishing the relative reliability of knowledge bases given the result of their merge is studied. This is in a way the reverse of the usual problem of merging them, like abduction [Douven 2011] reverses implication: from some information one attempts to derive what has generated it.

Two semantics for merging are considered for this inversion: sums of distances [Konieczny and Pérez 2011; Konieczny et al. 2002; 2004] and priority base merging [Nebel 1992; 1998; Rott 1993; Delgrande et al. 2006]. In a way, these can be considered at the extreme opposite of the spectrum of the many possible semantics for merging [Konieczny and Pérez 2011; Delgrande et al. 2006]: the first is numeric, model-based and majority-obeying; the second is qualitative (priority-based), syntax-dependent and not majority-obeying. The idea of obtaining reliability information, in whichever form they are expressed, can be however applied to other semantics for merging. How this idea can be extended to problems encompassing merging, contraction [Peppas et al. 2012] and update [Herzig et al. 2013] is also left as an open problem.

The main result proved for the semantics based on the sum of distances is an equivalent formulation for the condition of K_1 and K_2 generating R with some weights. From this, complexity upper bounds follow, as well as the core of a local search algorithm for determining weights. In particular, whenever the distance measure used is in Π_i^p or in Σ_i^p, obtainability is in Π_{i+1}^p. Two relevant measures are the drastic and the Hamming distances, for which the problem is proved coNP and Π_2^p-complete, respectively. A tractable subcase is proved.

Table II. Complexity of obtainability

	membership	hardness
arbitrary pseudodistance	-	coNP-hard
pseudodistance in Π_i^p or Σ_i^p	in Π_{i+1}^p	coNP-hard
Hamming distance	in Π_2^p	Π_2^p-hard
drastic distance	in coNP	coNP-hard
K_1 and K_2 conjunctions of literals, R Horn or Krom	in P	

Complexity of obtaining R from K_1 and K_2 using distance-based merging.

	membership	hardness
priority base merging	in Σ_3^p	coNP-hard
constant m	in coNP	coNP-hard
Berge-acyclic maxsets	in P	
binary maxsets, R is a consistent or-of-maxsets	in coNP	coNP-hard

Complexity of obtaining R from $K_1, \ldots, K_m$ using priority base merging.

The complexity analysis of priority base merging shows that obtainability is not harder than computing the result of merging with a fixed priority ordering for the considered subcases. Given that obtainability is the existence of a priority ordering generating a given result, at a first looks it may seem harder. Most of the problems in belief revision are at the second level of the polynomial hierarchy [Eiter and Gottlob 1992; 1996; Liberatore 1997a; Nebel 1998; Liberatore and Schaerf 2001], even in some simple restrictions like two formulae to be integrated. In contrast, obtainability proved coNP complete with a constant number of formulae or with maxsets of two or less formulae. The problem of obtainability in general is however still open, so it may prove harder. If Theorem A.21 extends in some form from graphs to hypergraphs, obtainability may be still in coNP in the general case.

What to do if the result is not obtainable? Various alternatives are outlined: relax the condition that R is exactly the result of merging, use another semantics of merging (for example, if R is unobtainable with priority merging one may try the weighted sum of Hamming distances), split the sources (for example, by the variables), lift the assumption that R is known with certainty, use some other information. However, in some cases a result should not be obtainable, like when all sources agree on x and the result is $\neg x$; in such cases, unobtainability still provide the useful warning that the sources are unreliable.

While the present article concentrates on obtainability, a sensible question is whether a given result is uniquely obtainable or not; another question is whether it can be obtained not with arbitrary weights or priorities, but with weights or priorities obeying some constraints, such as the weight of a base being greater than that of another. The problem of obtaining plausibility information from iterated revisions instead of merging has been considered in other articles [Booth and Nittka 2008; Liberatore 2015]; an open question is whether the two ideas can be put together, given that merging with integrity constraints generalize both merging and revision [Konieczny and Pérez 2011]. Some technical questions are also left open by this article, such as extending the necessary and sufficient condition for obtainability for weighted sum of distances from two to an arbitrary number of knowledge bases, and that for priority base merging from binary maxsets to arbitrary maxsets. Also, the gap between the hardness and membership results for priority base merging is also very large in the general case, as the problem is in Σ_3^p but only coNP-hard.

ELECTRONIC APPENDIX

The electronic appendix for this article can be accessed in the ACM Digital Library.

REFERENCES

C.E. Alchourrón, P. Gärdenfors, and D. Makinson. 1985. On the Logic of Theory Change: Partial Meet contraction and revision Functions. *Journal of Symbolic Logic* 50 (1985), 510–530.

R. Booth and A. Nittka. 2008. Reconstructing an Agent's Epistemic State from Observations about its Beliefs and Non-beliefs. *Journal of Logic and Computation* 18 (2008), 755–782. Issue 5. DOI: http://dx.doi.org/10.1093/logcom/exm091

R. Booth, Meyer T.A., I.J. Varzinczak, and Wassermann R. 2011. On the Link between Partial Meet, Kernel, and Infra Contraction and its Application to Horn Logic. *Journal of Artificial Intelligence Research* 42 (2011), 31–53.

S. Chopra, A. Ghose, and T. Meyer. 2006. Social choice theory, belief merging, and strategy-proofness. *Information Fusion* 7, 1 (2006), 61–79.

M. d'Agostino. 1999. Tableau methods for classical propositional logic. In *Handbook of tableau methods*. Springer, 45–123.

A. Darwiche and J. Pearl. 1997. On the Logic of Iterated Belief Revision. *Artificial Intelligence Journal* 89, 1–2 (1997), 1–29.

J.P. Delgrande. 2012. Revising beliefs on the basis of evidence. *International Journal of Approximate Reasoning* 53, 3 (2012), 396–412.

J.P. Delgrande, D. Dubois, and J. Lang. 2006. Iterated Revision as Prioritized Merging. In *Proceedings, Tenth International Conference on Principles of Knowledge Representation and Reasoning, KR-2006*. 210–220.

J.P. Delgrande, T. Schaub, H. Tompits, and S. Woltran. 2013. A Model-Theoretic Approach to Belief Change in Answer Set Programming. *ACM Transactions on Computational Logic* 14, 2 (2013), 14. DOI: http://dx.doi.org/10.1145/2480759.2480766

I. Douven. 2011. Abduction. (2011). Stanford Encyclopedia of Philosophy.

T. Eiter and G. Gottlob. 1992. On the Complexity of Propositional Knowledge Base Revision, Updates and Counterfactuals. *Artificial Intelligence Journal* 57 (1992), 227–270.

T. Eiter and G. Gottlob. 1996. The Complexity of Nested Counterfactuals and Iterated Knowledge Base Revisions. *J. Comput. System Sci.* 53, 3 (1996), 497–512.

P. Everaere, S. Konieczny, and P. Marquis. 2010. Disjunctive merging: Quota and Gmin merging operators. *Artificial Intelligence Journal* 174, 12–13 (2010), 824–849.

R. Fagin. 1983. Degrees of acyclicity for hypergraphs and relational database schemes. *Journal of the ACM* 30 (1983), 514–550. Issue 3.

P. Gärdenfors. 1988. *Knowledge in Flux: Modeling the Dynamics of Epistemic States*. Bradford Books, MIT Press, Cambridge, MA.

A. Herzig, J. Lang, and P. Marquis. 2013. Propositional Update Operators Based on Formula/Literal Dependence. *ACM Transactions on Computational Logic* 14, 3 (2013), 24. DOI: http://dx.doi.org/10.1145/2499937.2499945

Y. Jin and M. Thielscher. 2007. Iterated belief revision, revised. *Artificial Intelligence Journal* 171, 1 (2007), 1–18.

S. Konieczny, J. Lang, and P. Marquis. 2002. Distance-based merging: a general framework and some complexity results. In *Proceedings of the Eighth International Conference on Principles of Knowledge Representation and Reasoning (KR 2002)*. 97–108.

S. Konieczny, J. Lang, and P. Marquis. 2004. DA^2 merging operators. *Artificial Intelligence* 157, 1-2 (2004), 49–79.

S. Konieczny and R.P. Pérez. 2011. Logic based merging. *Journal of Philosophical Logic* 40, 2 (2011), 239–270.

P. Liberatore. 1997a. The complexity of belief update. In *Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence (IJCAI'97)*. 68–73.

P. Liberatore. 1997b. The complexity of iterated belief revision. In *Proceedings of the Sixth International Conference on Database Theory (ICDT'97)*. 276–290.

P. Liberatore. 2014. Belief revision by reliability assessment. (2014). Manuscript.

P. Liberatore. 2015. Revision by history. *Journal of Artificial Intelligence Research* 52 (2015), 287–329. DOI: http://dx.doi.org/10.1613/jair.4608

P. Liberatore and M. Schaerf. 1998. Arbitration (or how to merge knowledge bases). *IEEE Transactions on Knowledge and Data Engineering* 10, 1 (1998), 76–90.

P. Liberatore and M. Schaerf. 2001. Belief Revision and Update: Complexity of Model Checking. *J. Comput. System Sci.* 62, 1 (2001), 43–72.

J. Lin and A.O. Mendelzon. 1999. *Knowledge base merging by majority*. Springer, 195–218.

A.E. Mannes. 2009. Are We Wise About the Wisdom of Crowds? The Use of Group Judgments in Belief Revision. *Management Science* 55, 8 (2009), 1267–1279.

B. Nebel. 1992. *Syntax-Based Approaches to Belief Revision*. Cambridge University Press, 52–88.

B. Nebel. 1998. How hard is it to revise a knowledge base? In *Belief Change*, D. Dubois and H. Prade (Eds.). Handbook of Defeasible Reasoning and Uncertainty Management Systems, Vol. 3. Springer, 77–145.

R. Nieuwenhuis, A. Oliveras, and C. Tinelli. 2006. Solving SAT and SAT Modulo Theories: From an Abstract Davis–Putnam–Logemann–Loveland Procedure to DPLL(T). *Journal of the ACM* 53, 6 (2006), 937–977. DOI:http://dx.doi.org/10.1145/1217856.1217859

P. Peppas. 2008. Belief revision. In *Handbook of Knowledge Representation*. Elsevier, 317–359.

P. Peppas, C.D. Koutras, and M.A. Williams. 2012. Maps in Multiple Belief Change. *ACM Transactions on Computational Logic* 13, 4 (2012), 30. DOI:http://dx.doi.org/10.1145/2362355.2362358

P. Revesz. 1997. On the Semantics of Arbitration. *International Journal of Algebra and Computation* 7 (1997), 133–160. Issue 2.

H. Rott. 1993. Belief contraction in the context for the general theory of rational choice. *Journal of Symbolic Logic* 58, 4 (1993), 1426–1450.

K. See, W. Morrison, N. Rothman, and J. Soll. 2011. The detrimental effects of power on confidence, advice taking, and accuracy. *Organizational Behavior and Human Decision Processes* 116, 2 (2011), 272–285.

H. Wang, J. Zhang, and T. R. Johnson. 2000. Human belief revision and order effect. In *Proceedings of the 22th Annual Conference of the Cognitive Science Society*.

Online Appendix to: Belief merging by examples

PAOLO LIBERATORE, Sapienza University of Rome

This appendix contains the formal proofs concerning the case of binary maxsets. When all maxsets comprise at most two formulae, they can be seen as a graph:

— nodes are formulae;
— isolated nodes are singleton maxsets;
— edges are maxsets of two formulae.

Lemma 4.6 ensures that the contrary also holds: every graph corresponds to the set of maxsets of some formulae. As a result, properties of graphs carry to sets of maxsets.

This section is organized as follows:

(1) definitions and basic properties;
(2) transformations on graphs;
(3) properties of some specific graphs or subgraphs;
(4) proof that a graph is unobtainable if and only if it contains a cycle of alternating single excluded edge–odd sequence of selected edges.

Definition A.1 (4.14 in the article). A cycle is path ending in the same node where it started.

This is different from the definition of simple cycles, which are not allowed to cross an edge more than once.

A.1. Definitions

When all maxsets contain at most two formulae, the singletons can be excluded from consideration because of Lemma 4.1: $\{A\}$ cannot be contained in any other maxset; therefore, inclusion or exclusion do not affect the other maxsets. What remains is a set of binary maxsets, which can be seen as a graph where nodes are formulae and edges are maxsets. Some edges correspond to selected maxsets, the others to excluded maxsets.

Definition A.2 (4.15 in the article). A *selected-excluded graph* (abbreviated: *se graph*) is a graph whose edges are partitioned in two sets: selected and excluded.

Since edges are maxsets, the distinction indicates which are required to be in the result of merging and which are not. Most of the proofs regarding binary maxsets employ assignments of some formulae to priority class.

Definition A.3 (4.16 in the article). A *partially assigned se graph* has some nodes assigned positive integer values. If all nodes are assigned the graph is totally assigned.

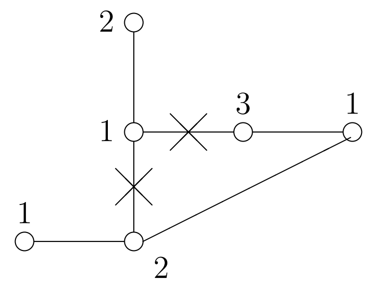
In a totally assigned se graph, all formulae are assigned a class. Therefore, one may determine the minimal edges (i.e., the edges that correspond to minimal maxsets) and check whether they are exactly the selected ones.

Definition A.4 (4.17 in the article). A totally assigned graph is obtainable if the minimal edges according to the priority ordering obtained from the numbers assigned to the nodes are exactly the selected ones.

DOI: http://dx.doi.org/10.1145/0000000.0000000

This definition may look tautological, but is rather close to the opposite. In a se graph, the selected edges are the maxsets that are required to be in the result of merging: if $\{A, B\} \models R$, the edge (A, B) is selected and vice versa. The values assigned to nodes may or may not make such a maxset minimal. If it is not, the edge is *incorrectly excluded*. Similarly, an excluded edge that is minimal according to the values is incorrectly selected. If no edge is incorrectly selected or excluded the ordering produces the required result.

The following graph illustrates the above definitions:



This is a se graph since some edges are marked as excluded (the crossed ones); the others are selected. It is also totally assigned since each node is assigned a value (its class in the priority ordering). It would be obtainable if every selected edge were minimal and every excluded edge were not. This is not the case, as the vertical excluded edge is minimal. Indeed, it is greater than the edge of values 1 and 3 and incomparable to the others; for example, the top edge has in common with it the node of value 1, but the two nodes of value 2 are different.

This graph would be obtainable if the value of the bottom node were 3 instead of 2; the bottom edge would remain minimal as it shares its node of value 1 with no other edge. Graphically, obtainability means that the crosses indicate the edges that are minimal according to the numbers.

Definition A.5 (4.18 in the article). A partially or totally assigned se graph G extends another one H if they have the same nodes and edges and all nodes assigned in H are also assigned in G to the same values.

A se graph is therefore obtainable if and only if it can be extended to a totally assigned se graph that is obtainable. On totally assigned se graphs obtainability can be checked by determining the minimal maxsets according to the ordering given by the values.

A.2. Influence

On totally assigned se graphs, one can check selection or exclusion of every edge by determining its minimality according the values. The following lemma shows which values affect the minimality of a particular edge.

LEMMA A.6. *In a totally assigned se graph, minimality of an edge (a, b) depends only on:*

(*1*) *the values of a and b, and*
(*2*) *the values of the nodes linked to a, if the value of a is one and the value of b is not;*
(*3*) *the values of the nodes linked to b, if the value of b is one and the value of a is not.*

Proof. If the values of a and b are both one, the edge is minimal no matter of what the other values are. If a and b are both greater than one, the edge is not minimal.

Of the remaining case, suffices to consider a assigned to one and b to a larger value: the other possibility is specular. If all nodes linked to a are greater or equal than b, then (a, b) is minimal. If one of them is lesser, it is not. In both cases, no other value of the graph affects the result. □

This lemma can be refined: of a node of value one, the only information that counts is the minimal values of nodes linked to it.

A.3. Value-depending transformations

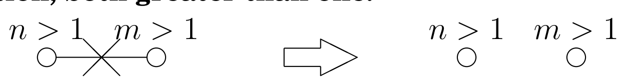
Se graphs can be simplified without affecting obtainability: the resulting graph is obtainable if and only if the original one is. Correctness is proved by a detour to the totally assigned graphs extending the original and resulting graphs. In particular:

- a partially assigned se graph is obtainable if and only if it can be extended to a totally assigned graph that is also obtainable;
- obtainability on totally assigned se graphs is verified by checking that the minimal edges are exactly the selected ones;
- the transformations do not turn a minimal edge into a non-minimal one in the totally assigned se graphs, and vice versa;
- in most cases, the transformations remove or add only edges that are correctly selected or excluded in the totally assigned se graph; otherwise, they replace correctly/incorrectly selected or excluded edges with edges that are equally correct or incorrect.

All this proves that the transformations are correct: they map a partially assigned se graph into another whose extensions to totally assigned se graphs correspond to the ones of the original graph, and this correspondence maps obtainable graphs into obtainable graphs and vice versa. As a result, the original graph is obtainable if and only if the resulting graph is. In most cases, obtainability is maintained simply because edge minimality is unaffected by the transformation.

The first simplification is disconnection, which is done in three different ways depending on the values.

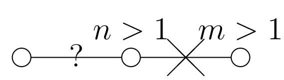
Disconnection, both greater than one.



An edge between two nodes of value greater than one can be removed.

In every extension to a totally assigned se graph, the edge is correctly excluded. Therefore, obtainability in both the graph before and after the change depends only on the minimality of the other edges.

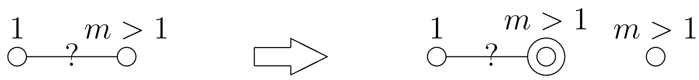
If an edge does not touch the disconnected one, by Lemma A.6 its minimality is unaffected by the change. But the lemma implies the same for edges touching the deleted one:



In this and the following figures, a question mark indicates that the edge may be selected or excluded, and the following reasoning holds in both cases.

Since $n > 1$, minimality of the other edge depends on n only, and not on nodes linked to the one of value n. The presence of the removed edge is therefore irrelevant.

Disconnection, one assigned one.



The double circle is a new node connected to none else. In this transformation, an edge between a node of value one and a node of value greater than one becomes an edge between the first and an isolated copy of the second.

In the totally assigned se graph extending the original one the edge may be minimal or not, but either way its status is not changed by the transformation, as its nodes maintain their value and its node of value one is connected to the same nodes as before. As a result, selection is either correct in both graphs or incorrect in both.

Regarding the other edges, minimality is not changed by the disconnection. If one such edge does not touch the disconnected one, or touches the node greater than one, Lemma A.6 tells that its minimality is not affected. But the same also holds for edges touching the node of value one, since this is connected to the same nodes as before, except that instead of the old node of value m is connected to a new node of value m.

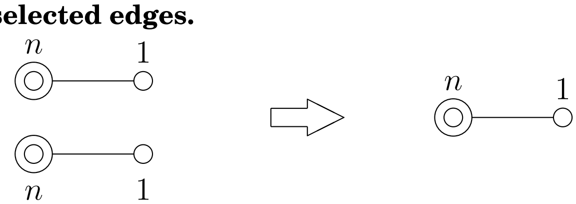
Disconnection, both assigned one.

The double circles are new nodes, connected to none else. An edge between two nodes of value one is split in two, each linking one of the original nodes to an isolated copy of the second.

The original edge is correctly selected in the original graph, and the two new ones are correctly selected in the resulting one. Therefore, obtainability depends only on the minimality of the other edges, which will be proved to be unchanged by the transformation.

By symmetry and Lemma A.6, the only relevant case is about edges touching the first node of the original edge. After the change, the node is still connected to the same other nodes and to a node of value one, as before. Therefore, minimality of the other edge is unaffected.

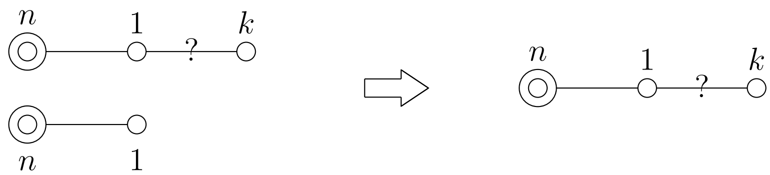
Merging of selected edges.



The double circles indicate nodes connected to none else. The two original nodes of value one may be touched by other edges, which are connected to the merged node of value one after the transformation.

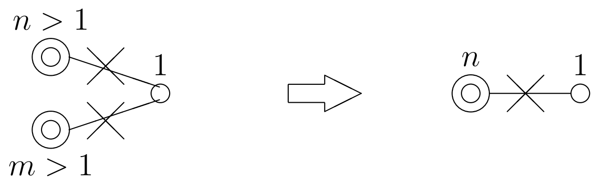
If any of the two nodes of value one is linked to one of value less than n, the same happens in the resulting edge, and vice versa. As a result, if any of the original edges is incorrectly selected so is the resulting edge, and vice versa. Therefore, it remains to show that obtainability is unaffected by the change only if the two original nodes assigned one are not linked to a node of value less than n.

Selection of edges not touching the nodes assigned one is not changed because of Lemma A.6. Regarding the edges touching one of these, let k be the value of the other node:



In the original totally assigned se graph, all other nodes linked to the ones assigned one have values greater than n. As a result, the minimality of this edge depends only on whether k is equal to n or greater. The same happens in the resulting graph.

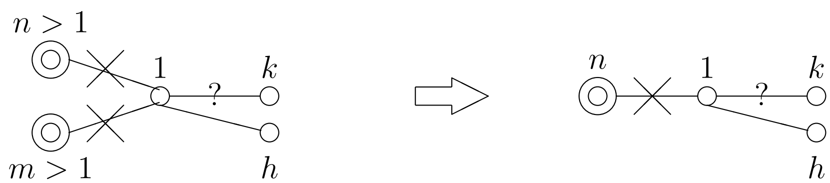
Merging of excluded edges.



In this figure, $n \leq m$. Double circles indicates nodes connected to none else.

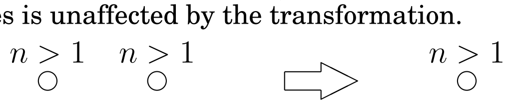
If the node of value one is only connected to nodes of value greater or equal than n, then the original totally assigned se graph is unobtainable, and so is the graph resulting from the transformation. Therefore, the only situation where obtainability could be altered is when the node of value 1 is connected to at least a node of value less than n.

An edge that does not touch the node of value one is unaffected by the change by Lemma A.6. Let k be the value of the other node of an edge touching it, and h the minimal values of nodes connected to the same node:

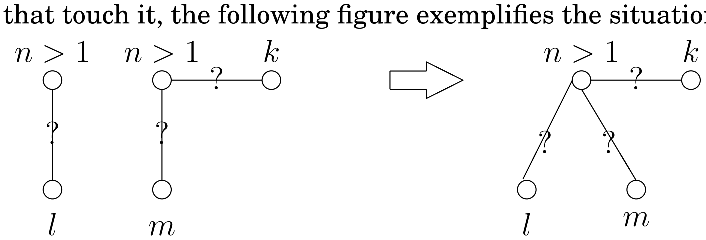


Since by assumption h if the minimal value of nodes connected to the node of value 1, minimality of the edge of values $1, k$ only depends on whether $k = h$ or not, in both the original and modified graph. This condition is not altered by the transformation.

Merging of nodes of equal values, greater than one. In this case, no edge is added or removed. The point is therefore only to prove that selection of an edge touching one of the two nodes is unaffected by the transformation.



Edges not touching any of the two nodes are unaffected by Lemma A.6. Regarding the ones that touch it, the following figure exemplifies the situation.



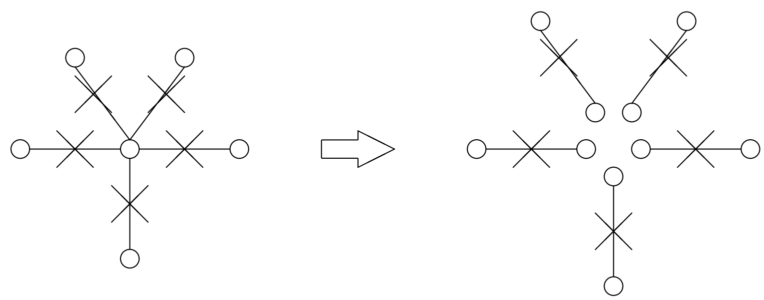
By Lemma A.6, the edge from nodes of values n and k is minimal or not depending on the value of n, but not on the other nodes linked to the one of value n. Therefore, the new link to the node of value l does not influence to the minimality of the edge.

In the following, two transformations are shown that, contrary to the ones above, do not require any condition on the value of the nodes. They can be therefore applied to se graphs that are totally unassigned.

A.4. Unassigned graphs transformations

The simplifications in the previous section assume knowledge of the values of nodes in the part of the graph to be changed. Some transformations that can be applied to unassigned se graphs are now presented. Contrary to the ones in the previous sections, these apply to nodes that are not assigned yet. They are valid regardless of the values these nodes may take: they map obtainable graphs into obtainable graphs, and unobtainable graphs into unobtainable graphs.

Definition A.7. The *full disconnection* of a node that is only touched by excluded edges is the replacement of the node with one for each of these edges.



LEMMA A.8. *Full disconnection maps obtainable graphs into obtainable graphs, and vice versa.*

Proof. The claim is proved by showing how to map values of the original node to values of its copies in the disconnected version of the graph: the single value is assigned to the copies; vice versa, if the copies have different values, set the original to their maximum.

As a preliminary observation, if the value of the node of a non-minimal edge is increased, the edge remains non-minimal.

If the original graph is obtainable, there exists at least an extension of it to a totally assigned se graph that is obtainable. If the central node has value one, it is changed to two; the graph remains obtainable. The nodes of the graph that results from the transformation are assigned as follows: the copies of the node that is broken get the same value of the original node; all other values are left unchanged. By Lemma A.6, these edges remain non-minimal, as they are still connected to a node of the same value greater than one. The edges connected to them are not changed either: even if the other node is assigned one, it is still connected to a node of the same value.

If the resulting graph is obtainable, it has at least an extension to an obtainable totally assigned se graph. The nodes that result from the disconnection may have the same value or not, and these values may even be all one. In the latter case, these values are all changed to two. Otherwise, they are all changed to the maximum of these values. This way, all these nodes are set to the same value. The original graph is then assigned values as follows: the node that was broken is assigned to the value of the resulting nodes; all others are the same. By Lemma A.6, all edges touching the broken node remain non-minimal because they are still connected to a node of the same value greater than one; the other edges remains minimal or not for the same reason of the previous case. □

The second transformation is about the removal of edges that do not participate in any cycle. Such edges form chains that may be isolated to the rest of the graph, or connected by one node only.

Definition A.9. The *removal of a tail* is the deletion of a chain of edges that do not participate in any cycle.

Removing all such edges leads to a graph where every edge is part of some cycle.

LEMMA A.10. *Removing tails does not alter obtainability.*

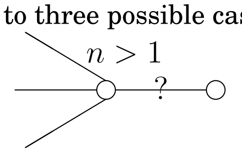
Proof. The claim is proved for tails comprising a single edge. Longer tails can be dealt with by removing edges one at time, from the end to the beginning. That tails end is a consequence of the finiteness of the graphs and the lack of cycles containing them.

Removing an edge releases a constraint: the edge is no longer required to be minimal if selected and non-minimal if excluded. As a result, if the original graph is obtainable, so is the one resulting from the transformation. Remains to prove the other direction: if the graph resulting from the removal is obtainable, the edge can be added back without violating obtainability.

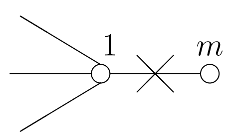
If the graph after removal is obtainable, an obtainable totally assigned se graph extending it exists. Recovering the removed edge introduces either one or two nodes. It is shown that these can be assigned values so that obtainability is maintained.

The case of two nodes added back is only possible if the edge is connected to none else. In this case, the values can be set to both one for a selected edge or two for an excluded one.

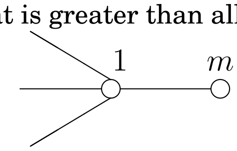
In the other case, one of the nodes is also in the graph after removal, so it has a values. This could be equal to one or greater. In the first case, the edge could be selected or excluded. This leads to three possible cases, the first being:



A value is to be chosen for the reintroduced right node so that the totally assigned se graph remains obtainable. By Lemma A.6, minimality of the other edges is not affected by the value of the right node, which can be therefore set to 1 if the edge is selected and 2 if excluded.



In this second case, the left node is one and the edge is excluded: the other node is assigned to a value that is greater than all other nodes connected to the left one.



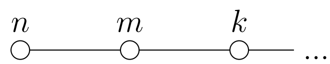
This is the third case. If the node of value 1 is connected via another selected edge to a node of value n, set $m = n$. If it is only touched by excluded edges, set $m = 1$. □

Another transformation is the zigzag folding, where a chain of selected edges is reduced to a single one by merging the first, third, fifth, etc. node of the chain and the second, fourth, etc.

Correctness is proved in two steps: first, a sequence of selected edges has alternating values ($n - m - n - m - \cdots$) in every obtainable totally assigned se graph; second, by a sequence of transformations, this result is used to prove that the sequence can be folded into a single selected edge.

LEMMA A.11. *The nodes of a chain of selected edges in a totally assigned obtainable graph has alternating values, that is,* $n - m - n - m - n - m - \cdots$.

Proof. Let n, m and k be the values of three consecutive nodes of the chain. The claim follows from $k = n$ for every possible values of n and m.



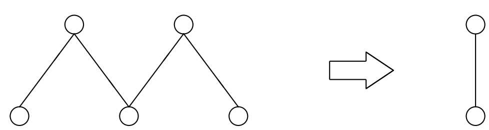
Various cases are possible:

— $n > 1$: by Lemma 4.5, in every selected edge at least one node has value 1; therefore, $m = 1$; if $k < n$, then $(1, k)$ is preferred over $(1, n)$; if $l > n$, the converse happens; since both edges are selected, $k = n$;
— $n = 1$, $m = 1$: if k greater than 1, then $(1, 1)$ is preferred over $(1, k)$; therefore, $k = 1$;
— $n = 1$, $m > 1$: the edge values (m, k) is selected; by Lemma 4.5, one between m and k is 1; since $m > 1$, if follows that $k = 1$, which is the same as n.

Since the alternation holds for every triple of consecutive nodes, it holds for the whole chain. □

This property implies that, regardless of the values of the other nodes of the graph, the only way to produce a correct assignment is by setting the nodes of the chain to values that alternate between two values.

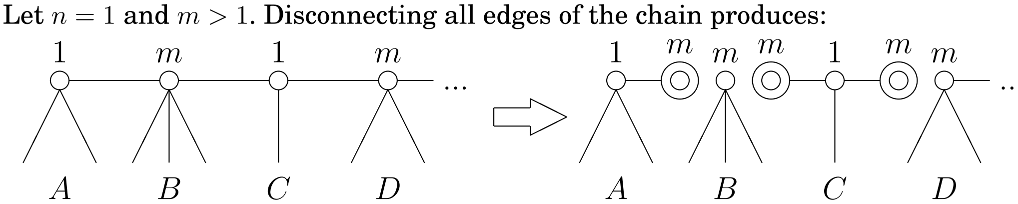
Definition A.12. Given a se graph, a *zigzag folding* of a chain of selected edges is the merging of all nodes of odd position and nodes of even position.



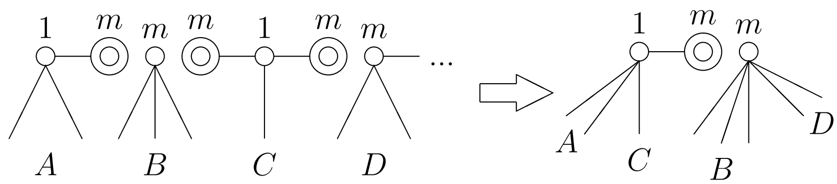
LEMMA A.13. *The zigzag folding maps obtainable graphs into obtainable graphs and vice versa.*

Proof. In every totally assigned se graph extending the given one, the nodes of the chain have alternating values $n - m - n - m - \cdots$ by Lemma A.11. By Lemma 4.5, one between n and m is one. The other may be one or greater.

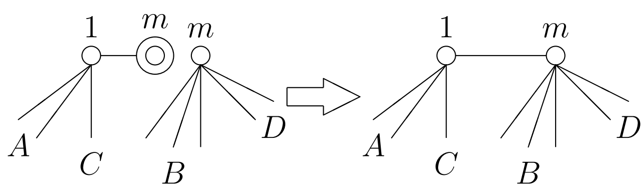
Let $n = 1$ and $m > 1$. Disconnecting all edges of the chain produces:



In this figure, A indicates the connections of the first node of the chain, B to the second, etc. Merging of selected edges and nodes of value greater than one collapse the nodes into two ones:



The two nodes of value m can be then merged back by applying disconnection in reverse:



The same can be done if $n = m = 1$, or $m = 1$ and $n > 1$. This proves that, regardless of the two values of the nodes of the chain, obtainability is the same if the chain is folded in a zigzag manner. In other words:

(1) for every se graph, every obtainable totally assigned se graph extending it has alternating values for the nodes of the chain;
(2) no matter what these values are, obtainability is not altered by folding the chain.

Therefore, folding turns an obtainable graph into an obtainable graph. If the original graph is instead unobtainable, still has extensions to totally assigned se graph with alternating values for the chain; however, these extensions incorrectly select or exclude some edge. This condition is not changed by folding, either. □

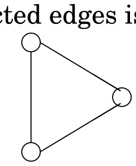
This lemma proves that every chain of selected edges can be turned into a single edge. The same can be done iteratively until the graph is left with no such a chain, so that no selected edge touches another one. Excluded edges may still form chains of arbitrary length, though.

A.5. Forced values

Some graphs require values to obey some simple conditions for obtaining the expected result.

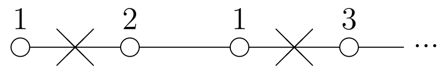
LEMMA A.14. *In any obtainable total assigned se graph containing a triangle of selected edges, the nodes of the triangle have value one.*

Proof. A triangle of selected edges is also a chain:



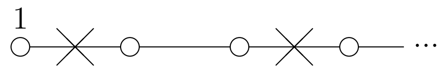
Let n, m, and k be the values of these nodes. By Lemma A.11, $n = k$. But also $m = n$, as the sequence is $n - m - k - n$. Since either n or m is equal to one by Lemma 4.5, it follows $n = m = 1$ and also $k = n = 1$. □

The following lemma shows that values are forced to increase in a chain of edges that are alternatively excluded and selected. In this configuration, if the first node is assigned 1 the values are $1 - n - 1 - m - k - \cdots$ with $1 < n < m < k < \ldots$. At a minimum, these values are 2, 3, 4, etc.

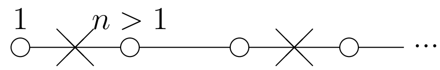


LEMMA A.15. *In any obtainable total assigned se graph containing a chain of alternating excluded-selected edges with the first node assigned one, the values of the other even nodes are one and of the even nodes are strictly increasing.*

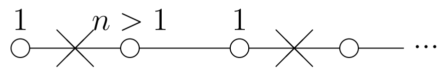
Proof. The chain begins with value 1 and an excluded edge:



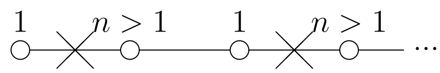
The next node cannot be one, as otherwise the edge would have values 1 and 1, so it would be minimal. Let $n > 1$ be the value of this node:



The second edge is selected: by Lemma 4.5, it has at least a node assigned one. Since $n > 1$, this cannot be other than the third node:



The values of the second edge are $n > 1$ and 1. The third edge also has the node assigned 1. In order to be non-minimal, the other value has to be greater than n:



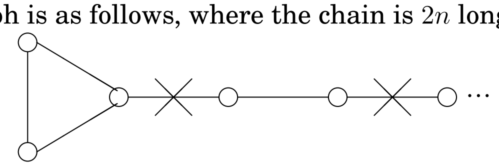
The proof can be iterated indefinitely, showing that each node of odd position has value one, and each node of even position has a value that is greater than the node two positions on the left of it. □

A.6. Graphs requiring n values to be obtainable

Several results are affected by whether values are equal to one or greater. This may suggest that what really matters about a value is whether it is one or not. In some cases, for example, a priority ordering that produces the expected result can be obtained by placing a formula for each maxset in class one, and all remaining ones in class two. This is however not always the case, as the next theorem shows: some graphs can be obtained only with n priority classes.

LEMMA A.16. *For every n there exists a graph that is only obtained by assignments with at least n different values.*

Proof. The graph is as follows, where the chain is $2n$ long:



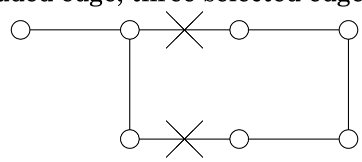
By Lemma A.14, the nodes of the triangle have value one in all totally assigned se graph extending this one. This also holds for the starting node of the chain, making Lemma A.11 applicable. The values of the chain are therefore 1, $n > 1$, 1, $m > n$, 1, $k > m, \ldots$ Since the chain is $2n$ long, it contains n strictly increasing values. □

When this lemma on graphs is recast in terms of formulae, it shows a sort of counterexample to the converse of Property 4.5: a priority ordering cannot always obtained by choosing one formula for each maxset to place in class one. To the contrary, some results can be obtained only with a large number of classes.

THEOREM A.17 (4.19 in the article). *For any n, there exists R and $K_1, \ldots, K_m$ such that R is obtainable by priority base merging from $K_1, \ldots, K_m$ only with priority partitions having n classes or more.*

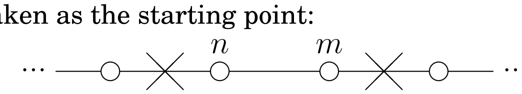
A.7. Unobtainable graphs

The following graph contains a cycle of edges in a form that the following lemma shows making it unobtainable. The edge on the left is irrelevant to obtainability, what counts is the cycle: an excluded edge, three selected edges, an excluded edge, a selected edge.



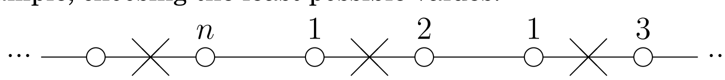
LEMMA A.18. *A graph containing a cycle of alternating (single excluded edge)–(chain of odd selected edges) is unobtainable.*

Proof. By Lemma A.13, chains of odd selected edges can be folded into a single edge where the first and last nodes are the same. After this transformation, the cycle becomes a sequence of alternating excluded and selected edges. An arbitrary selected edge can be taken as the starting point:

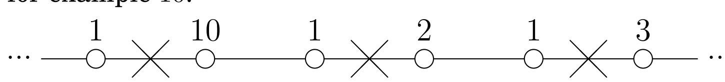


Lemma 4.5 tells that one among n and m is equal to one for the edge to be selected. It can be assumed $m = 1$, the other case is symmetric proceeding right-to-left.

By Lemma A.11, the next values alternates between one and an increasing value. As an example, choosing the least possible values:



The values at the end of excluded edges are increasing. Following the cycle, n gets its value, for example 10:



The first edge has values 1 and 10, the next one has the same node of value one and another of value 2. Therefore, the second is minimal and the first is not, opposite to the requirement. □

The following lemma shows a necessary and sufficient condition to obtainability.

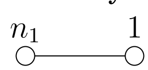
LEMMA A.19. *A graph is obtainable if and only if the result of applying full disconnection, removal of tails and zigzag folding as far as possible is an empty graph.*

Proof. These operations do not change obtainability. An empty graph is obtainable, as it does not contain edges on which selection can be incorrect; therefore, if the transformations lead to an empty graph, the original one is obtainable.

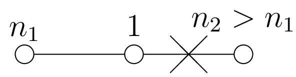
In the other way around, if the resulting graph is not empty:

(1) every node is touched by at least two edges, as otherwise the single edge would have been deleted by removal of tails;
(2) every node is touched by exactly one selected edge and one or more excluded edges; otherwise, two selected edges would have been folded, and excluded edges only separated by full disconnection.

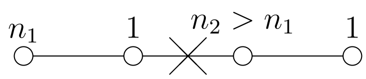
As a result of the second point, if the graph is not empty it contains at least a selected edge. For the graph to be obtainable, either one or its two nodes has to be assigned one by Lemma 4.5. The other node may be one or a greater value.



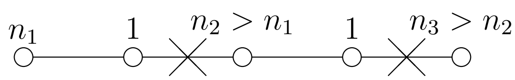
The case in which the values are reversed is identical.
By the first property of this graph, the node of value 1 is touched by at last another edge, which is excluded because of the second property.



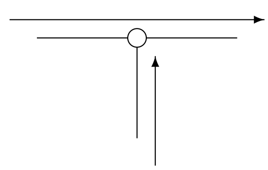
By Lemma A.15, n_2 is greater than n_1, as otherwise the first edge would not be selected and the second not excluded. By the two properties of the graph, the node of value n_2 is connected to at least a selected edge:



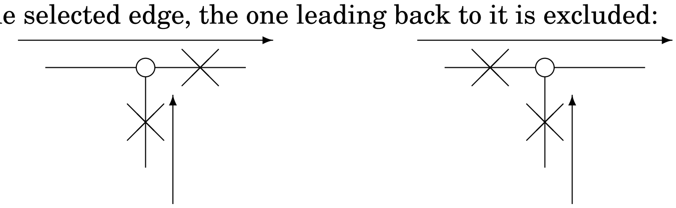
The last node is in turn connected to an excluded edge:



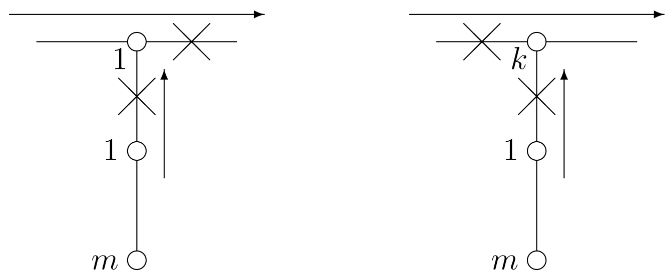
Again, $n_3 > n_2$ by Lemma A.15. The sequence proceeds alternating selected and excluded edges. By Lemma A.15, the nodes at the end of a selected edge have value 1, the others have increasing values. Since every node is touched by at least two edges in this graph, the sequence can be extended indefinitely, until it reaches a node that it already crossed.



Since the path is alternating, one of the two horizontal edges is selected and the other is excluded, leading to two possible cases. Since no node is touched by more than one selected edge, the one leading back to it is excluded:



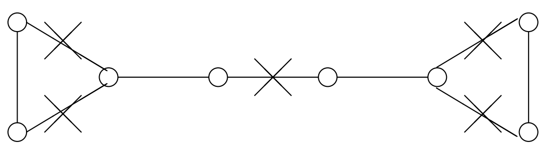
All values on the path obey the rules of Lemma A.15: one at the end of a selected edge, increasing the others.



In the first case, the vertical excluded edge is incorrectly selected. In the second case, by Lemma A.15 m is greater than k because it is later in the sequence; as a result, the vertical selected edge is incorrectly excluded.

This proves that assigning the first selected edge values n_1 and 1 leads to unobtainability. But the same happens, by symmetry, if these values are reversed. □

Lemma A.18 shows that a graph is unobtainable if it contains an alternating cycle. A proof similar to the one of the last lemma allows reversing this result, if cycles are allowed to follow an edge twice in opposite directions. An example where this is necessary is:



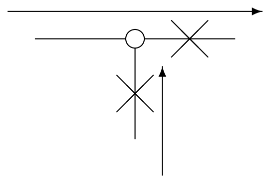
None of the three transformations can be applied, as the graph contains no tails, no chain of selected edges, and no node connected to excluded edges only. The graph is therefore unobtainable. However, the only alternating cycles cross the chain of three edges in the middle twice, once left-to-right and once right-to-left.

LEMMA A.20. *If a graph is unobtainable, it contains an alternating (single excluded edge)–(chain of odd selected edges) cycle that contains the same edge at most twice.*

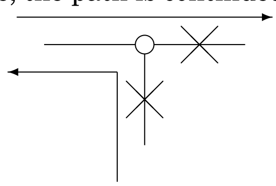
Proof. The claim is proved in two parts: first, the transformations do not add or remove alternating cycles; second, if the resulting graph is not empty, it contains an alternating cycle. By Lemma A.18, if the graph is unobtainable then the resulting graph is not empty; therefore, the original graph also has an alternating chain.

— full disconnection does not open alternating cycles, as every node in them is touched by a selected edge (no consecutive excluded edges); it does not create a new one either, as it only disconnect edges;
— tail removal only remove edges, so it never creates a new cycle; it does not touch existing cycles, alternating or otherwise;
— zigzag foldings do change cycles; however, it turns every path of odd selected edges into another path of selected edges of length one, and one is still an odd number; in the same way, paths of even edges are turned into paths of zero length; as a result, a cycle exists after the change if and only if it existed beforehand, and it is alternating if it was.

The second part of the proof shows that a non-empty graph resulting from applying the three transformations contains an alternating cycle. In particular, one alternating between single excluded edges and single selected edges. This is shown with a proof similar to the one of the previous theorem, with a difference. If the path reaches one of its previous nodes, this is not the end of the cycle if this would lead to two consecutive excluded edges:



If the edge after the node is selected, the cycle could be closed as an alternating one. This being not the case, the path is continued on the left:



The sequence can continue indefinitely. Since there are only a finite number of edges and only two directions for each edge, at some point the sequence comes back to an edge in the same direction it followed it before. The cycle is closed at that point. □

Since the existence of an alternating cycle implies unobtainability but is also implied by it, it is a characterization of this property.

LEMMA A.21. *A graph is unobtainable if and only if it contains a cycle of alternating (single excluded edge)-(chain of odd selected edges) that crosses the same edge at most twice.*

Expressed in terms of maxsets, it leads to the following theorem.

THEOREM A.22 (4.20 in the article). *Formula R is unobtainable from a set $K_1, \ldots, K_m$ having no maxset of size greater than two if and only if a cycle of (single maxset not in R)-(chain of odd maxsets in R) that crosses the same maxset at most twice exists.*