Which Sets Can be Expressed as Disjoint Union and Subset Complement Without Möbius Cancellations?

Antoine Amarilli
https://a3nm.net/
a3nm@a3nm.net

Louis Jachiet
https://louis.jachiet.com/
louis@jachiet.com

Mikaël Monet
http://mikael-monet.net/
mikael.monet@imfd.cl

This note presents a combinatorial problem on the Boolean lattice asking about which sets of subsets (called *configurations*) can be achieved from the monotone subsets with only one minimal element (called *cones*) using the operations of disjoint union and subset complement without so-called Möbius cancellations, i.e., while ensuring that the Möbius functions of the configurations being combined do not lead to cancellations.

The problem is motivated by the so-called *intensional-extensional conjecture* on query evaluation on probabilistic tuple-independent databases, and whether unions of conjunctive queries that are tractable for this problem admit lineage representations in a tractable circuit class: see [Monet, 2020] for details. We believe that solving the problem presented in this note would imply a solution to this conjecture. However, this note only presents the combinatorial problem: it does not present the connection to this conjecture, and assumes no familiarity with database theory or circuit classes.

1 Preliminaries

For $k > 0$, we write $[k]$ the set $\{0, \ldots, k-1\}$. For a set S we write 2^S its powerset. For a set S and two functions $f, g : S \to \mathbb{Z}$, we write $f + g$ (resp., $f - g$) the function defined by $(f + g)(s) = f(s) + g(s)$ (resp., $(f - g)(s) = f(s) - g(s)$) for all $s \in S$. For a function $f : S \to \mathbb{Z}$ over a finite set S, we write $|f|$ to denote the L_1-norm of f, that is, $|f| \stackrel{\text{def}}{=} \sum_{s \in S} |f(s)|$.

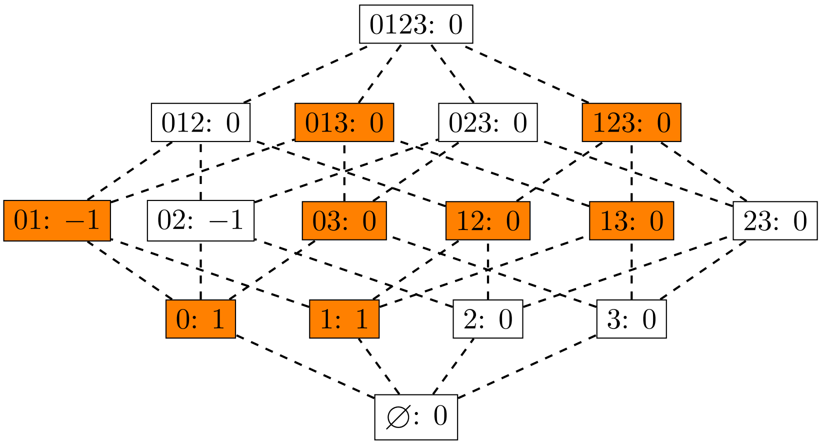

Figure 1: Visual representation of the configuration $\mathbf{s}$ from Example 2.1 and of its associated Möbius function $\mu_{\mathbf{s}}$. For simplicity, we write, e.g., "$01 : -1$" to mean that $\mu_{\mathbf{s}}(\{0, 1\}) = -1$. Here, $\mathbf{s}$ consists of all the colored nodes.

2 Problem Statement

In this section we fix $k \in \mathbb{N}$.

Configurations and Möbius functions. A *configuration* is a subset of $2^{[k]}$. Given a configuration $\mathbf{s} \subseteq 2^{[k]}$, its associated *Möbius function* $\mu_{\mathbf{s}} : 2^{[k]} \to \mathbb{Z}$ is defined by bottom-up induction as follows:

$$\mu_{\mathbf{s}}(n) \stackrel{\text{def}}{=} \begin{cases} 1 - \sum_{n' \subsetneq n} \mu_{\mathbf{s}}(n') \text{ if } n \in \mathbf{s} \\ - \sum_{n' \subsetneq n} \mu_{\mathbf{s}}(n') \text{ if } n \notin \mathbf{s} \end{cases} .$$

In particular, the value $\mu_{\mathbf{s}}(\varnothing)$ is 1 if $\varnothing \in \mathbf{s}$, and 0 if $\varnothing \notin \mathbf{s}$.

Example 2.1. *Let $k = 4$, and $\mathbf{s}$ be the configuration $\{\{0\}, \{1\}, \{0, 1\}, \{0, 3\}, \{1, 2\}, \{1, 3\}, \{0, 1, 3\}, \{1, 2, 3\}\}$. We have depicted in Figure 2 this configuration and its associated Möbius function.*

Remark 2.2. *Let $\mu : 2^{[k]} \to \mathbb{Z}$. Then μ is the Möbius function of some configuration if and only if the following holds: for all $n \in 2^{[k]}$, we have $\sum_{n' \subseteq n} \mu(n') \in \{0, 1\}$. The associated configuration is then simply $\{n \in 2^{[k]} \mid \sum_{n' \subseteq n} \mu(n') = 1\}$.*

Disjoint unions and differences. Let $\mathbf{s_1}, \mathbf{s_2}$ be two configurations. When $\mathbf{s_1} \cap \mathbf{s_2} = \varnothing$, we define the *disjoint union of* $\mathbf{s_1}$ *and* $\mathbf{s_2}$ by $\mathbf{s_1} \oplus \mathbf{s_2} \stackrel{\text{def}}{=} \mathbf{s_1} \cup \mathbf{s_2}$. When $\mathbf{s_2} \subseteq \mathbf{s_1}$, we define the *subset complement of* $\mathbf{s_1}$ *and* $\mathbf{s_2}$ by $\mathbf{s_1} \ominus \mathbf{s_2} \stackrel{\text{def}}{=} \mathbf{s_1} \backslash \mathbf{s_2}$. From now on, when we write $\mathbf{s_1} \oplus \mathbf{s_2}$ (resp., $\mathbf{s_1} \ominus \mathbf{s_2}$), we will always assume that $\mathbf{s_1} \cap \mathbf{s_2} = \varnothing$ (resp., that $\mathbf{s_2} \subseteq \mathbf{s_1}$), i.e., that the operation is well-defined.

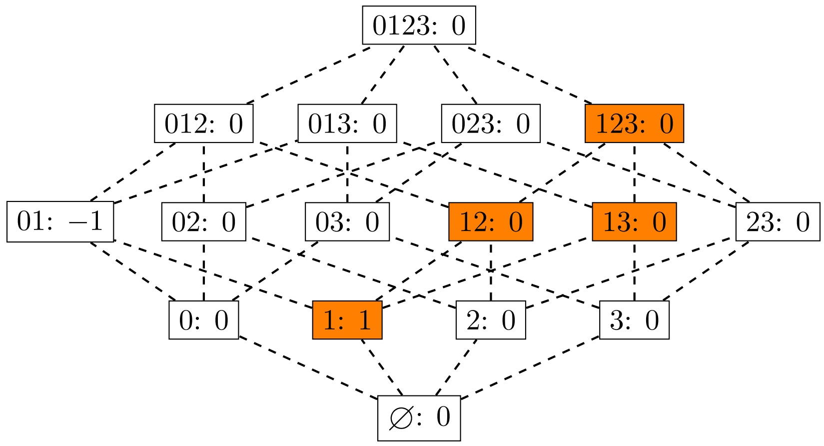

Figure 2: The configuration $\mathbf{s_1}$ from Example 2.4 and its associated Möbius function $\mu_{\mathbf{s_1}}$.

Lemma 2.3. *Let $\mathbf{s_1}, \mathbf{s_2}$ be two configurations such that $\mathbf{s_1} \oplus \mathbf{s_2}$ (resp., $\mathbf{s_1} \ominus \mathbf{s_2}$) is well-defined, and let $\mathbf{s}$ be the resulting configuration. Then we have $\mu_{\mathbf{s}} = \mu_{\mathbf{s_1}} + \mu_{\mathbf{s_2}}$ (resp., $\mu_{\mathbf{s}} = \mu_{\mathbf{s_1}} - \mu_{\mathbf{s_2}}$).*

Proof. It is routine to show by bottom-up induction that for all $n \in 2^{[k]}$ we indeed have $\mu_{\mathbf{s}}(n) = \mu_{\mathbf{s_1}}(n) + \mu_{\mathbf{s_2}}(n)$ (resp., $\mu_{\mathbf{s}}(n) = \mu_{\mathbf{s_1}}(n) - \mu_{\mathbf{s_2}}(n)$). □

Example 2.4. *Consider the configuration $\mathbf{s}$ from Example 2.1 and let $\mathbf{s_1}$ be the configuration $\{\{1\}, \{1, 2\}, \{1, 3\}, \{1, 2, 3\}\}$ and $\mathbf{s_2}$ be the configuration $\{\{0\}, \{0, 1\}, \{0, 3\}, \{0, 1, 3\}\}$ (depicted in Figures 2 and 3). Then we have $\mathbf{s} = \mathbf{s_1} \oplus \mathbf{s_2}$. Moreover, one can check that we indeed have $\mu_{\mathbf{s}} = \mu_{\mathbf{s_1}} + \mu_{\mathbf{s_2}}$.*

Cancellation-freeness. We now impose another restriction on how we can combine configurations using $\oplus$ and $\ominus$:

Definition 2.5. *Let $\mathbf{s_1}, \mathbf{s_2}$ be two configurations. We say that the operation $\mathbf{s_1} \oplus \mathbf{s_2}$ (resp., $\mathbf{s_1} \ominus \mathbf{s_2}$) is* cancellation-free *when the operation is well-defined (i.e., it is a disjoint union or subset complement) and when, letting $\mathbf{s}$ be the resulting configuration, we have $|\mu_{\mathbf{s}}| = |\mu_{\mathbf{s_1}}| + |\mu_{\mathbf{s_2}}|$.*

Note that Lemma 2.3 and the triangle inequality already implied that we have $|\mu_{\mathbf{s}}| \leqslant |\mu_{\mathbf{s_1}}| + |\mu_{\mathbf{s_2}}|$ when the operation is well-defined. Moreover, observe that this definition is equivalent to saying that, for all $n \in 2^{[k]}$, we have $|\mu_{\mathbf{s}}(n)| = |\mu_{\mathbf{s_1}}(n)| + |\mu_{\mathbf{s_2}}(n)|$, or equivalently that $\mu_{\mathbf{s_1}}(n)$ and $\mu_{\mathbf{s_2}}(n)$ have the same sign (or at least one of them is 0); hence the name *cancellation-free*.

Example 2.6. *Continuing Example 2.4, one can check that $\mathbf{s_1} \oplus \mathbf{s_2}$ is cancellation-free.*

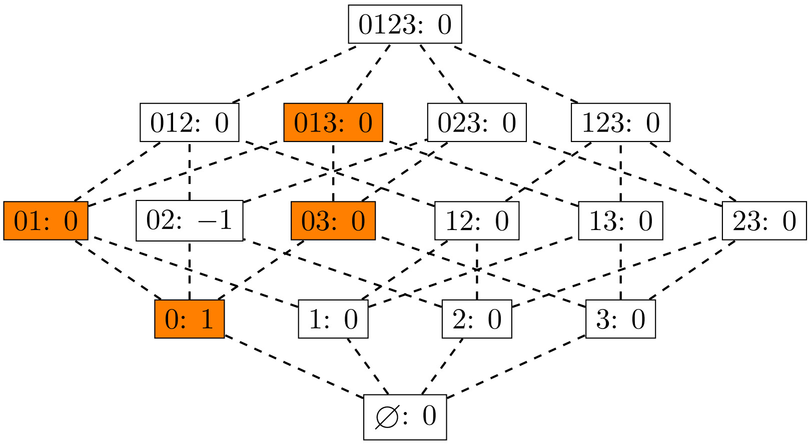

Figure 3: The configuration $\mathbf{s_2}$ from Example 2.4 and its associated Möbius function $\mu_{\mathbf{s_2}}$.

Cones and reachability. Our goal is to understand which configurations can be constructed using by cancellation-free $\oplus$ and $\ominus$ operations, from some basic configurations that we call *cones*. For each $n \in 2^{[k]}$, the *cone* C_n spanned by n is simply the configuration of all supersets of n, that is, $C_n \stackrel{\text{def}}{=} \{n' \in 2^{[k]} | n \subseteq n'\}$.

Remark 2.7. *Observe that the Möbius function of a cone C_n is simply $\mu_{C_n}(n) = 1$ and $\mu_{C_n}(n') = 0$ for all $n' \neq n$. So $|\mu_{C_n}| = 1$, and in fact cones are precisely the configurations whose Möbius function has norm equal to* 1*.*

The set $\mathcal{R}_k$ of *reachable configurations* is then defined as the set of all configurations that can be achieved by cancellation-free $\oplus$ and $\ominus$ operations from cones. Formally, it is the (unique) set of configurations which is minimal by inclusion, contains the empty configuration and all cones, and is closed under cancellation-free $\oplus$ and $\ominus$, i.e., if $\mathbf{s_1}$ and $\mathbf{s_2}$ are in $\mathcal{R}_k$ and if $\mathbf{s_1} \oplus \mathbf{s_2}$ (resp., $\mathbf{s_1} \ominus \mathbf{s_2}$) is cancellation-free then the resulting configuration is in $\mathcal{R}_k$.

Example 2.8. *Continuing Example 2.6, one can easily check that $\mathbf{s_1} = C_{\{2\}} \ominus C_{\{0,1\}}$ is cancellation-free, and that $\mathbf{s_1} = C_{\{0\}} \ominus C_{\{0,2\}}$ is also cancellation-free; hence $\mathbf{s_1}$ and $\mathbf{s_2}$ are reachable. Since $\mathbf{s_1} \oplus \mathbf{s_2}$ is also cancellation-free, we have that $\mathbf{s} = \mathbf{s_1} \oplus \mathbf{s_2}$ is also reachable.*

Problem statement. Our goal is to characterize which configurations are reachable in this sense. As we will see, this is in fact not the case of all configurations. Specifically, we want to show that all *monotone* configurations are reachable. A configuration $\mathbf{s}$ is *monotone* when for all $n, n' \in 2^{[k]}$, if $n \in \mathbf{s}$ and $n \subseteq n'$ then $n' \in \mathbf{s}$.

Conjecture 2.9. *All monotone configurations are reachable.*

Cone-reachability. A related problem, leading to a stronger conjecture on monotone functions, concerns *cone-reachability*. Intuitively, the set $\mathcal{R}'_k$ of *cone-reachable configurations* is the set of configurations that can be reached when imposing that the second operand to an operation is always a cone. Formally, it contains the empty configuration, contains all cones, and whenever $\mathbf{s} \in \mathcal{R}'_k$ then for any cone C_n, if $\mathbf{s} \oplus C_n$ is cancellation-free then it belongs to $\mathcal{R}'_k$, and the same holds for $\mathbf{s} \ominus C_n$.

A stronger conjecture than the above is:

Conjecture 2.10. *All monotone configurations are cone-reachable.*

We point out here that a variant this problem on arbitrary partial orders (instead of simply the Boolean powerset) has been presented in `https://cstheory.stackexchange.com/q/45679/38111`, where a generalization of Conjecture 2.10 has been stated for *join semi-lattices*.

3 Alternative Problem Statements

Let us consider the Boolean lattice over Boolean events $E_1, \ldots, E_n$, and let us consider an event $S = \bigcup_j C_j$, where each C_j is a *basic conjunction*, i.e., a conjunction of some of the events E_i.

An alternative way to phrase our problem is to ask whether S can be expressed as a function of basic conjunctions (not necessarily the ones used in the definition of S) using only the disjoint union operator ($T = T_1 \cup T_2$ where T_1 and T_2 are disjoint) and the subset complement operator ($T = T_1 \backslash T_2$ where $T_2 \subseteq T_1$).

In general, it is always possible to built S with disjoint union and subset complement by applying recursively the inclusion-exclusion principle on the definition of S. So the question that we ask is whether we can express S in the sense above but satisfying a *minimality requirement*: each basic conjunction C_j must be used exactly as specified by its coefficient in the Möbius function associated to the configuration corresponding to S, i.e., with the number of times indicated by the absolute value, and the polarity indicated by the sign of the Möbius function. This means in particular that conjunctions with Möbius coefficient zero cannot be used: these are the conjunctions that can be canceled out in the inclusion-exclusion formula.

An equivalent way to see this is to take the expression of S as a function of basic conjunctions given by the inclusion-exclusion formula, apply cancellations across terms so that the coefficient of each basic conjunction is its value in the Möbius function, and ask about whether these terms can be ordered such that additions correspond to disjoint unions and subtractions correspond to subset complements.

Alternatively, we ask whether S, seen as a monotone DNF Boolean function, can be expressed as a function of basic conjunctions (monotone conjunctions of variables) using disjoint union and negation, i.e., as a d-D circuit, as a function of basic conjunctions, each of which being used with the right polarity and cardinality. This connects back to the motivation of our problem, i.e., the intensional-extensional conjecture on probabilistic databases [Monet, 2020].

4 Current Results

4.1 Facts known by bruteforce

In this section, we discuss some preliminary results obtained experimentally by implementing several ways of solving the problem by bruteforce. We have put the necessary code publicly online `https://gitlab.com/Gruyere/mobius-unions-differences`. The code is very rough, and of course there could be errors affecting the results presented here. The code is not documented, but don't hesitate to ask if you have questions.

Values $k \leqslant 3$. For small values of k, we can reach all configurations, even with the stronger notion of cone-reachability:

Proposition 4.1. *For $k \in \{1, 2\}$, all configurations (in $2^{[k]}$) are cone-reachable.*

Proof. By bruteforce:

```
g++ -ocones1 -DN=1 -DONLY_CONES=1 -O2 bottom_up.cpp; ./cones1
g++ -ocones2 -DN=2 -DONLY_CONES=1 -O2 bottom_up.cpp; ./cones2
```

□

For $k = 3$, we start to see that some configurations are not cone-reachable, but all monotone configurations are still cone-reachable, and all configurations are reachable:

Proposition 4.2. *For $k = 3$, all configurations (in $2^{[3]}$) are reachable, and all configurations are cone-reachable except $\{\varnothing, \{0, 1, 2\}\}$ and its complement.*

Proof. By bruteforce:

```
g++ -oall3 -DN=3 -O2 bottom_up.cpp; ./all3
g++ -ocones3 -DN=3 -DONLY_CONES=1 -O2 bottom_up.cpp; ./cones3.
```

□

Value $k = 4$. For $k = 4$, we start to see that some configurations are not reachable, but all monotone configurations are still cone-reachable.

To present these results, we first introduce the notion of *equivalence*. We say that two configurations $\mathbf{s}$ and $\mathbf{s}'$ are *equivalent* if there exists a permutation σ of $[k]$ such that, writing $\sigma(n) = \{\sigma(i) \mid i \in n\}$ for all $n \in 2^{[k]}$, then $\{\sigma(n) \mid n \in \mathbf{s}\} = s'$. This is clearly an equivalence relation. As our problem is invariant under permutations of $[k]$, when considering sets of configurations, it always suffices to study them up to equivalence.

Second, we introduce the notion of *irreducibility*. We say that a configuration $\mathbf{s}$ is *reducible* if it can be expressed as a cancellation-free sum $\mathbf{s_1} \oplus \mathbf{s_2}$ or difference $\mathbf{s_1} \ominus \mathbf{s_2}$ for some configurations $\mathbf{s_1}, \mathbf{s_2}$ which are not the empty configuration. A configuration that is not reducible is said to be *irreducible*. We define *cone-irreducibility* in the expected way. Observe that the empty configuration and all cones are irreducible and in particular cone-irreducible. Moreover, all irreducible configurations that are not the empty configuration or a cone are not reachable. Intuitively, an irreducible configuration is a minimal kind of unreachable configuration, but as we will see there are some configurations that are reducible but not reachable.

We can now claim our result:

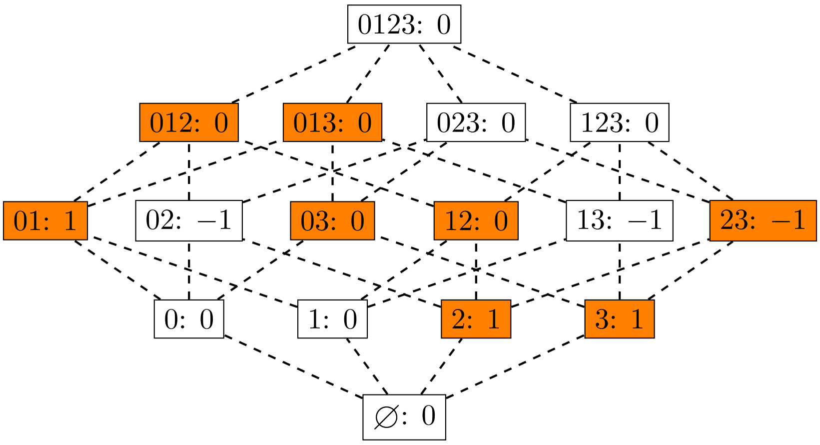

Figure 4: The configuration $\mathbf{s_3}$ from Proposition 4.3 and its associated Möbius function $\mu_{\mathbf{s_3}}$.

Proposition 4.3. *For $k = 4$, out of the $2^{2^4} = 65536$ configurations that exist, there are exactly 24 configurations that are not reachable, and none of them are monotone. There are 12 of these configurations that are irreducible, and they are all equivalent to the configuration $\mathbf{s_3}$ depicted in Figure 4.1. The other 12 are not irreducible and are all equivalent to the complement of $\mathbf{s_3}$.*

Moreover, there are exactly 2748 configurations that are not cone-reachable (248 up to equivalence), none of which are monotone, and 910 of which are irreducible (90 up to equivalence).

Proof. Bruteforce:

```
g++ -oall4 -DN=4 -O2 bottom_up.cpp; ./all4
g++ -ocones4 -DN=4 -DONLY_CONES=1 -O2 bottom_up.cpp; ./cones4
```

and variants with `TEST_IRRED` and `ONLY_MINIMAL` □

Value $k = 5$ and up. For $k = 5$, there are 2^{2^5} configurations, so around 4 billion, and we can no longer afford to test all of them. However, testing specifically the monotone configurations, we can show

Proposition 4.4. *For $k \in \{1, \ldots, 5\}$ all monotone configurations are cone-reachable.*

Proof. For $k \in \{1, \ldots, 5\}$ this follows from what precedes. Bruteforce for $k = 5$:

```
g++ -omonotone_cones5 -DN=5 -DALL=1 -DONLY_MONOTONE=1 -DEXPLAIN=0 \
  -DEXPLAIN=0 -DONLY_CONES=1 -O2 top_down.cpp && ./monotone_cones5
```

□

We nevertheless managed to test irreducibility for all these configurations:

Proposition 4.5. *For $k = 5$, there are exactly 38 irreducible configurations (up to equivalence).*

Proof. Bruteforce:

```
g++ -o all5reach -DN=5 -DALL=1 -DSTEP=1000000 -DONLY_MONOTONE=0 \
 -DONLY_MINIMAL=1 -DEXPLAIN=0 -O2 top_down.cpp && ./all5reach
```

The bruteforce considered 37333248 minimal functions, as per sequence 1405 of the Sloane handbook. ☐

Proposition 4.6. *For $k \in \{1, \ldots, 6\}$ all monotone configurations are reachable.*

Proof. For $k \in \{1, \ldots, 6\}$ this follows by what precedes. Bruteforce for $k = 6$ (this concludes extremely fast, with very little backtracking):

```
g++ -omonotone6 -DN=5 -DALL=1 -DONLY_MONOTONE=1 -DEXPLAIN=0 \
 -DEXPLAIN=0 -O2 top_down.cpp && ./monotone6
```

☐

We do not know if all monotone configurations are cone-reachable for $k = 6$, as the execution of our program is too slow to conclude in that case.

For standard reachability, for $k = 7$, there are several hundred million monotone Boolean functions even up to symmetry, so it has not been possible so far to test all of them, but all tested functions were reachable (and the computation was instantaneous).

For larger values of k, the program is sometimes too slow to conclude to the reachability or even reducibility of some monotone functions, but we do not know whether this because it is inefficient or whether it could be a counterexample to the conjecture.

4.2 Facts that hold in all generality

In this section, we present some miscellaneous facts that we obtained.

Connection with the notion of Euler characteristic. The following can be seen as an alternative way of defining the Möbius function of a configuration:

Proposition 4.7. *Let $\mathbf{s} \subseteq 2^{[k]}$ be a configuration, for some $k > 1$. Then, for all $n \in 2^{[k]}$, we have that*

$$\mu_{\mathbf{s}}(n) = \sum_{\substack{n' \subseteq n \\ n' \in \mathbf{s}}} (-1)^{|n \setminus n'|}.$$

Proof. Direct application of Möbius inversion formula on the poset $2^{[k]}$. ☐

It is related to the notion of *Euler characteristic* of a Boolean function. To establish the connection, observe that a configuration $\mathbf{s} \subseteq 2^{[k]}$ can be seen as a Boolean function over variables $[k]$. The *Euler characteristic* of $\mathbf{s}$ is $\mathsf{e}(\mathbf{s}) \overset{\text{def}}{=} \sum_{\substack{n \in 2^{[k]} \\ n \in \mathbf{s}}} (-1)^{|n|}$ [Roune and Sáenz-de Cabezón, 2013, Stanley, 2011]. Hence, $\mu_{\mathbf{s}}(n)$ is the Euler characteristic of the sub-configuration below n, multiplied by $(-1)^{|n|}$.

Constructing unreachable configurations from simpler unreachable configurations.

Definition 4.8. *Let $k > 1$, $\mathbf{s} \subseteq 2^{[k]}$ be a configuration in $2^{[k]}$ and $\mathbf{s}' \in 2^{[k+1]}$ be a configuration in $2^{[k+1]}$. We say that $\mathbf{s}'$* contains $\mathbf{s}$ as a lower-subconfiguration *when there exists $j \in [k+1]$ and a bijection $\sigma : [k] \to [k+1]\backslash\{j\}$ such that we have $\mathbf{s}' \cap 2^{[k+1]\backslash\{j\}} = \sigma(\mathbf{s})$.*

Here is a simple way to take an unreachable configuration for k and obtain unreachable configurations for $k+1$:

Lemma 4.9. *Let $k > 1$, $\mathbf{s} \subseteq 2^{[k]}$ be an unreachable configuration for k, and $\mathbf{s}' \subseteq 2^{[k+1]}$ be a configuration for $k+1$. Then, if $\mathbf{s}'$ contains $\mathbf{s}$ as a lower-subconfiguration then $\mathbf{s}'$ is unreachable as well (in $2^{[k+1]}$).*

Proof. Up to permutations, we can assume without loss of generality that j is k and that $\sigma : [k] \to [k]$ is the identity in Definition 4.8. Assume by way of contradiction that $\mathbf{s}'$ is reachable in $2^{[k+1]}$, and let T' be a parse tree of one of its decomposition; that is, a rooted ordered binary tree, whose leaves are labeled by cones and whose internal nodes are labeled by $\oplus$ or $\ominus$. For a node i of T let us write ${\mathbf{s}'}_i$ the corresponding configuration (so that ${\mathbf{s}'}_r$ is $\mathbf{s}'$ for r the root of T'). For a configuration $\mathbf{s}' \subseteq 2^{[k+1]}$, let us write $\breve{\mathbf{s}'} \subseteq 2^{[k]}$ the configuration of $2^{[k]}$ defined by $\breve{\mathbf{s}'} \stackrel{\text{def}}{=} \mathbf{s}' \cap 2^{[k]}$. Now, let T be the rooted ordered binary tree obtained from T' by replacing every cone C_n by $\breve{C_n}$, and considering that the operations are done in $2^{[k]}$. One can easily check that T is a valid parse tree (meaning that all the operations are well-defined and cancellation-free). Furthermore it is easy to check that, letting $\mathbf{s_i}$ be the configuration corresponding to node i of T, we have $\mathbf{s_i} = \breve{\mathbf{s}'_\mathbf{i}}$ for all nodes i. Hence $\mathbf{s} = \breve{\mathbf{s}'} = \breve{\mathbf{s}'_\mathbf{r}} = \mathbf{s_r}$ is reachable (in $2^{[k]}$), a contradiction. □

Degeneracy.

Definition 4.10. *Let $k > 1$, $\mathbf{s} \subseteq 2^{[k]}$ be a configuration and $j \in [k]$. We say that $\mathbf{s}$* does not depend on *j if for all $n \in 2^{[k]}$ the following holds: we have $n \in \mathbf{s}$ if and only if $n \cup \{j\} \in \mathbf{s}$. We call $\mathbf{s}$* degenerate *if there exists $j \in [k]$ such that $\mathbf{s}$ does not depend on j, and* nondegenerate *otherwise.*

Lemma 4.11. *Let $\mathbf{s} \subseteq 2^{[k]}$ be a configuration that does not depend on $j \in [k]$. Then, for every $n \in 2^{[k]}$ such that $j \in n$, we have that $\mu_\mathbf{s}(n) = 0$.*

Proof. Direct by bottom-up induction on $\{n \in 2^{[k]} \mid j \in n\}$. □

The following lemma tells us intuitively that an unreachable configuration $\mathbf{s}'$ of $2^{[k+1]}$ either contains an unreachable configuration of $2^{[k]}$ as a lower-subconfiguration (in which case Lemma 4.9 "explains" why $\mathbf{s}'$ is unreachable), or it must be nondegenerate:

Lemma 4.12. *Let $k > 1$, and let $\mathbf{s}' \subseteq 2^{[k+1]}$ be a configuration that is not reachable. Then one of the following it true (it could be both):*

1. *there exists a configuration* $\mathbf{s} \subseteq 2^{[k]}$ *such that* $\mathbf{s} \notin \mathcal{R}_k$ *and such that* $\mathbf{s}'$ *contains* $\mathbf{s}$ *as a lower-subconfiguration;*

2. $\mathbf{s}$ *is nondegenerate (that is,* $\mathbf{s}'$ *depends on all* $j \in [k+1]$*).*

Proof. Assume by way of contradiction that both 1 and 2 are false. Since $\mathbf{s}'$ is degenerate, and up to permutation of $[k+1]$, we can assume without loss of generality that $\mathbf{s}'$ does not depend on k. Let $\mathbf{s} \stackrel{\text{def}}{=} \mathbf{s}' \cap 2^{[k]}$. Clearly, $\mathbf{s}'$ contains $\mathbf{s}$ as a lower-subconfiguration. Because we assumed 2 to be false, $\mathbf{s}$ must then be reachable (in $2^{[k]}$). Let T be a parse tree of a decomposition of $\mathbf{s}$. For a configuration $\mathbf{s}'' \subseteq 2^{[k]}$, let us write $\widehat{\mathbf{s}''}$ the configuration of $2^{[k+1]}$ defined by $\widehat{\mathbf{s}''} \stackrel{\text{def}}{=} \mathbf{s}'' \cup \{n \cup \{k\} \mid n \in \widehat{\mathbf{s}''}\}$. Now, let T' be the parse tree obtained from T by replacing every cone $C_n \subseteq 2^{[k]}$ at a leaf of T by the cone $\widehat{C_n} \subseteq 2^{[k+1]}$, and considering that the operations are done in $2^{[k+1]}$. For a node i, we write $\mathbf{s_i}$ (resp., $\mathbf{s_i'}$) the configuration corresponding to node i in T (resp., in T'). Then, one can show by bottom-up induction on T that for all nodes i the following holds:

- the operation at that node in T' is well-defined and cancellation-free (this uses in particular Lemma 4.12);
- we have that $\mathbf{s_i'} = \widehat{\mathbf{s_i}}$.

But then this implies that $\mathbf{s}' = \widehat{\mathbf{s}}$ is reachable, a contradiction.

□

References

[Monet, 2020] Monet, M. (2020). Solving a special case of the intensional vs extensional conjecture in probabilistic databases. *arXiv preprint arXiv:1912.11864*.

[Roune and Sáenz-de Cabezón, 2013] Roune, B. H. and Sáenz-de Cabezón, E. (2013). Complexity and algorithms for Euler characteristic of simplicial complexes. *Journal of Symbolic Computation*, 50:170–196.

[Stanley, 2011] Stanley, R. P. (2011). *Enumerative Combinatorics: Volume 1*. 2nd edition.