Under Revisions

Local First Order Logic with Data: Toward Specification of Distributed Algorithms

Thèse de doctorat en informatique
École doctorale 386 – Sciences Mathématiques de Paris Centre

Par:
Olivier Stietel
Université Paris Cité, CNRS, IRIF, F-75013, Paris, France

Dirigé par:
Benedikt Bollig[1] **Arnaud Sangnier**[2]

Soutenue le 14 décembre 2023

Jury constitué de:

Nathalie BERTRAND[3]	Rapporteur
Radu IOSIF[4]	Rapporteur
Nathalie SZNAJDER[5]	Examinateur
Thomas ZEUME[6]	Examinateur
Benedikt BOLLIG[1]	Directeur de thèse
Arnaud SANGNIER[2]	Directeur de thèse

[1]Directeur de recherche, Université Paris-Saclay, CNRS, ENS Paris-Saclay, LMF, Paris, France
[2]Maître de conférences HDR, IRIF, Université de Paris, CNRS, Paris, France
[3]Directrice de recherche, Université Rennes, Inria & IRISA, France
[4]Directeur de recherche, Université Grenoble Alpes, CNRS, Grenoble INP, VERIMAG, 38000, France
[5]Maître de conférences, Sorbonne Universités, UPMC Université Paris 06, UMR 7606, LIP6, F-75005, Paris, France
[6]Professeur, Ruhr-Universität Bochum, Allemagne

Résumé en Français

Ce manuscrit s'inscrit dans le domaine de la vérification formelle, une discipline informatique visant à garantir l'absence d'erreurs dans les programmes. Dans ce contexte, la spécification en est une facette, et consiste à énoncer formellement les propriétés que le programme doit satisfaire. Notre focus se porte sur les algorithmes distribués, qui sont particulièrement difficile à conceptualiser. Pour aborder cette problématique, nous explorons les logiques avec données, adaptées à cet usage. Dans notre cadre, une donnée est un élément d'un ensemble dénombrable, les éléments des structures ont une ou plusieurs données et la logique peut seulement tester si deux données sont égales ou non. La manière dont nous analysons les logiques est au travers du problème de la satisfaisabilité. Cette étude débute par un état de l'art des résultats concernant différentes logiques sur diverses structures telle que les mots, les multi-ensembles, les mots avec données, les multi-ensembles avec données et les graphes. Nous comblons quelques lacunes identifiées dans la littérature, notamment des preuves manquantes et des cas non étudiés. Nous exposons également deux outils classiques pour aider à l'analyse du problème de la satisfaisabilité, à savoir les jeux d'Ehrenfeucht-Fraïssé et les problèmes de dominos. Ensuite, commence l'apport principale de ce travail qui réside dans la définition et l'étude d'une nouvelle logique avec données, que nous appelons le fragment local, introduisant à cet effet la modalité locale. Inspirée par la notion de localité et de graphe de Gaifman, notre approche se distingue par l'intégration de la localité dans la définition de notre logique, au niveau de la syntaxe. Nous définissons également un sous-fragment appelé le fragment existentiel. Nous démontrons que notre logique constitue effectivement un fragment de la logique du premier ordre, en cela que la modalité local peut s'exprimer dans la logique du premier ordre. Par la suite, nous analysons notre nouvelle logique sous l'angle du problème de la satisfaisabilité. Nous identifions des critères pour lesquels la logique est décidable et d'autres pour lesquels elle ne l'est pas. De plus, dans les cas où la logique est décidable, nous nous efforçons d'analyser la complexité du problème de la satisfaisabilité.

Mots-clés logique, vérification, spécification, donnée, premier-ordre, multi-ensemble, satisfaisabilité, distribué

Abstract in English

This manuscript is part of the field of formal verification, a computer science discipline aimed to ensure that programs are error-free. In this context, specification is one facet which consists of formally stating the properties the program has to satisfy. Our focus is on distributed algorithms which are particularly difficult to conceptualise. To tackle this problem, we explore data logic which is adapted to this purpose. In our framework, a data value is an element of a countable set, elements of structures have one or more data values and the logic can only test whether two data values are equal or not. The logics are analysed through the lens of the satisfiability problem. This work begins by exposing the state of the art concerning different logics on various structures such as words, multisets, data-words, data-multisets and graphs. Some identified gaps in the literature are filled, including missing proofs and unexplored cases. Additionally, two classical helpful tools in the analysis of the satisfiability problem are presented, namely Ehrenfeucht-Fraïssé games and domino problems. Afterwards, the main contribution of this work begins which is the definition and study of a new data logic, called the local fragment, introducing for this purpose the local modality. Inspired by the notion of locality and Gaifmans' graphs, our approach distinguishes itself by integrating locality in the definition of our logic, at the level of syntax. We show that our logic is indeed a fragment of first-order logic, as the local modality can be expressed in first-order logic. A sub-fragment called the existential fragment is also defined. Subsequently, our new logic from the point of view of the satisfiability problem is analysed; is identified criteria for which the logic is decidable and others for which it is not. Furthermore, in the cases where the logic is decidable, we try to analyse the complexity of the satisfiability problem, and succeeded in most of the cases.

Keywords logic, verification, specification, data, data values, first-order, multiset, data-multiset, satisfiability, distributed

Contents

Chapter 1

Introduction

The introduction is first in french, and then in english. The content of the two versions are exactly the same.

1.1 Introduction in French

1.1.1 Motivations Générales

Vérification L'informatique est une discipline à multiples facettes qui va au-delà du simple calcul sur ordinateur et s'appuie sur la mécanisation de la pensée grâce à un empilement de couches successives d'abstractions. Son but ultime étant d'obtenir une séquence de chaînes mécaniques et causales conduisant à des comportements hautement complexes. Cependant, cette réalisation ne se fait pas d'un seul geste; au contraire, elle nécessite l'accumulation progressive de différents niveaux d'abstraction, chacun contribuant à penser et maîtriser la complexité inhérente.

Le processus d'abstraction commence au niveau le plus fondamental, comprenant les composants électroniques jusqu'à atteindre le niveau du processeur. De là, il passe aux langages compilés avant d'atteindre son apogée au niveau du langage de programmation de haut niveau, culminant dans le développement d'une application logicielle concrète. Chacune de ces étapes de la progression recèle sa propre hiérarchie d'abstractions, chaque couche apportant une dimension supérieure de compréhension et de manipulation.

Le processus de développement se déroule typiquement par tâtonnement, ce qui implique des essais et des erreurs. Dans un premier temps, un prototype est créé puis testé et des erreurs sont identifiées. Par la suite, le prototype est amélioré et le cycle se répète. Finalement, la décision est prise que le prototype est prêt, mais l'élimination complète des bugs reste insaisissable. Les bugs ont tendance à persister, ce qui est particulièrement problématique pour les applications critiques. Cela soulève la question de trouver une meilleur méthode que des tests. La solution que nous proposons consiste à appliquer des principes mathématiques rigoureux pour atteindre un niveau de confiance inégalé dans nos systèmes informatiques.

Logique La logique, discipline fondamentale des mathématiques et de la philosophie, est une pierre angulaire de l'informatique. La logique a de nombreuses applications, de la conception de compilateurs aux bases de données, en passant par l'intelligence artificielle, la sémantique des langages de programmation et la sécurité informatique. Mais ici, nous nous concentrons uniquement sur la vérification. La logique permet de s'assurer qu'un programme fonctionne comme prévu. L'intérêt de la logique est double. Le premier est de pouvoir s'assurer, par une série de déductions, que notre programme ou algorithme fait ce que nous voulons qu'il fasse. Le second, plus subtil, est de pouvoir exprimer ce que nous attendons de notre algorithme. C'est ce qu'on appelle l'expressivité. Le langage naturel est mal adapté à cette tâche. Un exemple convaincant est toute la subtilité que les opérateurs modaux F et G de LTL nous permettent d'exprimer de façon claire, concise et sans ambiguïté.

Systèmes Distribués Les algorithmes usuels fonctionnent dans un environnement informatique centralisé, où une seule entité informatique exécute des instructions de manière séquentielle pour résoudre un problème. Ils présupposent l'accès à une mémoire globale et peuvent s'appuyer sur la puissance de calcul et les ressources d'un seul processeur. Les algorithmes distribués, en revanche, sont spécifiquement conçus pour les environnements informatiques décentralisés. Ils impliquent plusieurs entités autonomes (nœuds ou processus) qui interagissent sur un réseau pour résoudre collaborativement un problème. Ces algorithmes doivent tenir compte de l'absence de mémoire globale et de la nécessité d'une communication et d'une coordination entre les entités distribuées. Les principales problématiques des algorithmes distribués comprennent la gestion des questions de concomitance, de synchronisation, de transmission de messages, de tolérance aux pannes et de redimensionnement, qui ne sont habituellement pas rencontrées dans les algorithmes usuels.

Ce changement de paradigme que nous demande l'informatique distribuée est tout sauf mineur. Il rend la conception des algorithmes particulièrement compliquée, bien plus que dans les cas séquentiels, à tel point que même les experts commettent encore des erreurs, non seulement des erreurs d'implémentation facilement corrigibles, mais également des erreurs irréparables au stade de la conception. La vérification peut donc s'avérer utile pour les algorithmes distribués. En général, la vérification n'est pas une pratique répandue dans l'industrie parce qu'elle est coûteuse et prend du temps, ce qui rend le rapport bénéfice/coût trop peu intéressant. Elle est réservée à quelques applications de niche où des vies humaines sont en jeu (transport, nucléaire) ou lorsque d'énormes sommes d'argent sont en jeu (finance). En revanche, dans le domaine de l'informatique distribuée, elle permet déjà de trouver des erreurs qui n'auraient pas été détectées autrement. Un exemple en est l'utilisation de TLA+ dans l'industrie [53, 3], où elle a aidé à identifier des interblocage et d'autres défauts de conception.

1.1.2 Motivations Spécialisées

Les logiques avec données ont été introduites afin de raisonner sur des structures dont les éléments sont étiquetés par des données prisent dans un alphabet infini (e.g. documents

XML) [45]. Parmi les fragment expressifs et décidables, il y a notamment la logique à deux variable sur les mots et les arbres [7, 6]. La frontière de la décidabilité est cependant fragile; par exemple, les extensions avec deux données par éléments sont rapidement indécidables pour le problème de la satisfaisabilité. Néanmoins, du point de vue de la modélisation, ces extension jouent encore un rôle important. D'autres types de logiques permettent d'avoir deux données par éléments [31, 34], bien qu'elles ne supposent pas un univers linéairement ordonné ou arborescent. Là encore, la satisfaisabilité s'avère être décidable pour le fragment à deux variables de la logique du premier ordre. Lorsque l'on considère un nombre arbitraire de variables du premier ordre, ce que nous faisons dans ce travail, la frontière de la décidabilité est rapidement franchie sans contraintes supplémentaires. La restriction que nous considérons ici est la localité, un concept essentiel de la logique du premier ordre. Il est notoirement connu que la logique du premier ordre n'est capable d'exprimer que des propriétés locales : une formule du premier ordre peut toujours être écrite comme une combinaison de propriétés d'éléments qui ont une distance limitée, c'est-à-dire limitée par un rayon donné, par rapport à certains points de référence [27, 23]. En présence de (plusieurs) données, imposer une restriction de localité la logique peut contribuer à garantir la décidabilité de son problème de satisfaisabilité, comme nous le verrons ultérieurement.

À partir du chapitre 3, nous ne considérerons uniquement une extension naturelle de la logique du premier ordre sur des structures non ordonnées dont les éléments possèdent un nombre fixe de données provenant d'un domaine infini. Lorsqu'on spécifie le *comportement entrée-sortie* des algorithmes distribués [21, 38], les processus reçoivent une valeur d'entrée et produisent une valeur de sortie, ce qui nécessite deux données par processus. Dans les algorithmes d'élection d'un leader ou de renommage, par exemple, un processus reçoit son identifiant unique en entrée et doit finalement produire l'identifiant d'un leader commun (élection de leader) ou un identifiant unique provenant d'un espace de noms restreint (renommage). Il y a deux différences majeures entre la plupart des formalismes existants et notre langage. Alors que les autres logiques avec données sont généralement interprétées sur des mots ou des arbres, nous considérons des structures non ordonnées (ou multi-ensembles). Si chaque élément d'une telle structure représente un processus, nous ne présupposons alors pas d'architecture de communication particulière, nous considérons plutôt un nuage d'unités de calcul. En outre, les logiques avec données décidables sont généralement limitées à une donnée par élément, ce qui n'est pas suffisant pour modéliser une relation d'entrée-sortie. Par conséquent, nos modèles sont des structures algébriques composées d'un univers et de fonctions attribuant à chaque élément un nombre fixe d'entiers. il est à noter que pour de nombreux algorithmes distribués, les valeurs précises des données ne sont pas pertinentes; il en retourne que la plupart du temps, ce que compte est de savoir si certaines données sont égales ou non. Pareillement à [7, 6], nous ajoutons donc des relations binaires qui nous permettent de tester si deux valeurs de données sont identiques et, par exemple, si toutes les valeurs de sortie étaient déjà présentes dans la collection des valeurs d'entrée (comme requis pour l'élection d'un leader).

La première question fondamentale qui se pose est de savoir si une spécification donnée est cohérente; cela nous conduit au problème de la satisfaisabilité. Bien que la logique générale considérée ici s'avère être indécidable dans plusieurs contextes restrictifs, notre résultat principal montre que plusieurs fragments intéressants préservent la décidabilité. Ces fragments sont des logiques *local* dans le sens où les données ne peuvent être comparées que dans le voisinage direct d'un processus (quantifié) de référence.

Travaux Apparentés Les extensions orthogonales avec de multiples données incluent les *shuffle expressions for nested data* [4] et les logiques temporelles [30, 15]. D'autres généralisations des logiques avec données permettent d'ordonner les valeurs des données [39, 47]. L'application de méthodes formelles dans le contexte des algorithmes distribués est une approche plutôt récente mais prometteuse (cf. pour une vue d'ensemble [35]). Une branche particulière est le domaine des systèmes paramétrés, qui, plutôt que de se concentrer sur les données, se concentre sur le nombre (non borné) de processus comme paramètre [5, 18]. D'autres travaux connexes incluent [17], qui considère les logiques temporelles impliquant des quantifications sur les processus mais sans données, tandis que [1] introduit une variante (indécidable) de la logique dynamique propositionnelle qui permet de raisonner sur les identificateurs de processus totalement ordonnés dans les architectures en anneau. Les logiques du premier ordre pour *synthétiser* les algorithmes distribués ont été examinées dans [8, 25].

1.1.3 Disgression Mathématique

Dans cette sous-section, nous parlerons principalement de mathématiques. Cette sous-section se veut être compréhensible pour un mathématicien, ou au moins un logicien, qui n'a jamais étudié l'informatique.

Ce travail de logique est orienté vers la vérification. La plus grande différence avec la logique mathématique est que nous ne considérons ici que des structures finies. Dans cette sous-section uniquement, nous ferons référence à la théorie des modèles larges comme étant la théorie qui autorise des structures de toute cardinalité, et à la théorie des modèles finis comme étant la théorie qui n'autorise que des structures finis. Ainsi, la théorie des modèles larges est celle que l'on rencontre le plus souvent en mathématiques, tandis que la théorie des modèles finis prédomine en informatique. Loin d'être une distinction mineure, cela a de nombreuses implications, la plus importante étant que le théorème de compacité est faux dans le cas fini. Pour illustrer cela, nous pouvons considérer la suite de formules $(\lambda_n)_n$ qui stipule que le modèle a au moins n éléments. Toute sous-séquence finie est satisfaisable, mais la séquence entière ne l'est pas. Ceci a des implications positives et négatives qui sont, en somme, incommensurables.

Un effet positif notable est l'élimination de constructions étranges telles que les modèles non standard et les éléments non standard, cela a pour conséquence que nous n'avons pas à traiter les entiers qui ne sont pas un successeur itéré de 0. Du côté négatif, il y a le théorème de préservation de Łoś-Tarski, un résultat classique de la théorie des modèles larges, qui n'est plus vrai dans le cas fini. Ce théorème stipule qu'une formule préservée par l'extension des structures est équivalente à une formule existentielle. Une autre implication négative est qu'il n'existe pas de système de preuve complet, i.e. qui puisse refléter la vérité sémantique. C'est une conséquence directe de l'absence de théorème de complétude.

Dans un sens neutre, dans la théorie des modèles larges, le problème de validité est semi-décidable (il suffit d'énumérer tous les arbres de preuve et de voir si l'un d'entre eux prouve la formule). Dans la théorie des modèles finis, en revanche, c'est le problème de la satisfaisabilité qui est semi-décidable (il suffit d'énumérer toutes les structures et de voir si l'une d'entre elles satisfait la formule). Il est intéressant de noter que les problèmes de validité et de satisfaisabilité sont complémentaires l'un de l'autre, ce qui crée une symétrie entre les deux théories des modèles. De plus, combiné au fait que le problème de satisfaisabilité est indécidable en général pour la théorie des modèles finis, il fournit un autre argument contre l'existence d'un système de preuve complet.

1.1.4 Contributions et Organisation du Manuscrit

Dans le chapitre 2, nous commençons par la section 2.1 en donnant la définition générale de structures, logiques et problèmes de satisfaisabilité utilisées dans ce travail. Ensuite, dans la section 2.2, nous exposons les outils fréquemment utilisés dans l'étude des problèmes de satisfaisabilité. En particulier, pour les bornes supérieures, nous présentons la propriété du petit modèle et les jeux d'Ehrenfeucht-Fraïssé, tandis que pour les bornes inférieures, nous introduisons le problème du pavage. Dans la section 2.3, nous présentons un catalogue de résultats sur le problème de la satisfaisabilité sur les mots, les multi-ensembles, les mots avec données, les graphes et les multi-ensembles avec données. En outre, nous fournissons une preuve plus accessible du résultat déjà connu selon lequel FO^3 sur les graphes est indécidable (Théorème 2.3.20). Nous prouvons également le résultat connu mais avec une preuve manquante de l'indécidabilité de FO sur deux relations d'équivalence (Théorème 2.3.27). Nous faisons même plus, car notre preuve améliore encore le résultat, jusqu'à FO^3 (Théorème 2.3.31).

Dans le chapitre 3, nous explorons le concept de localité sur les multi-ensembles avec données. Dans la section 3.1, nous définissons notre fragment local. Puis, dans la section 3.2, nous exposons des résultats fondamentaux sur les fragments locaux, ce qui constitue une contribution originale à ce travail. Nous commençons par montrer que nous pouvons internaliser la distance dans la logique du premier ordre, pour ensuite utiliser ce fait afin d'établir des inclusions entre les différents fragments. Enfin, nous examinons le lien entre notre notion de graphe de données et la notion de graphe de Gaifman.

Dans le chapitre 4, nous étudions le fragment principal et décrivons la frontière entre la décidabilité et l'indécidabilité. Nous démontrons que le problème de satisfaisabilité de ce fragment est décidable lorsque l'on restreint les propriétés locales au rayon 1 (section 4.2, théorème 4.2.19). Cependant, pour tout rayon strictement supérieur à 1, le problème est indécidable (section 4.3, théorèmes 4.3.5 et 4.3.9). il est à noter que l'ajout d'une seule relation diagonale constitue toujours une extension de la logique du premier ordre à deux variables (décidable) avec deux relations d'équivalence [32, 31, 34] : les classes d'équivalence sont constituées des éléments ayant la même première valeur, respectivement, la même deuxième valeur. Pour aller plus loin, notre principale contribution technique est une réduction vers cette logique à deux variables. Cette réduction nécessite un réétiquetage minutieux des structures sous-jacentes afin de pouvoir exprimer la relation diagonale en termes de deux relations d'équivalence. En outre, la réduction tient compte du fait que notre logique ne restreint pas le nombre de variables. Nous pouvons en effet compter les éléments jusqu'à un certain seuil et exprimer, par exemple, qu'au plus cinq processus se bloquent (dans le contexte des algorithmes distribués). Ceci n'est a priori pas possible avec une logique à deux variables.

Dans le chapitre 5, nous étudions le fragment local orthogonal dans lequel la quantification globale est restreinte à être existentielle (tandis que la quantification à l'intérieur d'une modalité locale reste non restreinte). Dans sa première section 5.1, nous obtenons la décidabilité pour (i) le rayon 1 et un nombre arbitraire de données (théorème 5.1.9), et pour (ii) un rayon 2 et deux données (théorème 5.1.6). Dans tous les cas, nous fournissons des bornes supérieures et inférieures de complexité strictes. De plus, ces résultats marquent la frontière exacte de la décidabilité : la section 5.2 montre que la satisfaisabilité est indécidable dès que l'on considère le rayon 3 en présence de deux données (théorème 5.2.3), ou le rayon 2 avec trois données (théorème 5.2.5).

1.2 Introduction in English

1.2.1 General Motivations

Verification Computer science is a multifaceted discipline that goes beyond mere computation, instead relying on the mechanization of thought through the application of successive layers of abstractions. Its ultimate goal is to achieve a sequence of mechanical and causal chains leading to highly complex behaviours. However, this achievement does not occur in one fell swoop; rather, it requires the progressive accumulation of various levels of abstraction, each contributing to mastering and rationalizing inherent complexity.

The process commences at the most fundamental level, encompassing hardware components, and then ascends to the processor level. From there, it moves on to compiled languages before reaching its zenith at the high-level programming language, culminating in the development of a concrete software application. Each of these stages in the progression harbours its own hierarchy of abstractions, with each layer bringing a higher dimension of understanding and manipulation.

The typical development process involves trial and error. Initially, a prototype is created, tested, and errors are identified. Subsequently, the prototype is enhanced, and the cycle repeats. Eventually, a decision is made that the prototype is ready, but complete bug-proofing remains elusive. Bugs tend to persist, which is particularly problematic for critical applications. This raises the question of finding a superior approach to testing. Our proposed solution involves the application of rigorous mathematical principles to attain an unparalleled level of confidence in our computer systems.

Logic Logic, as a fundamental discipline within mathematics and philosophy, serves as the cornerstone of computer science. Logic has many applications, from compiler design to databases, artificial intelligence, semantics of programming languages, and computer security. But here, we focus only on verification. Logic helps ensure that a program works as intended. The interest in logic is twofold. The first is to be able to ensure through a series of deductions that our program or algorithm does what we want it to do. And the second, more subtle one, is to be able to express what we desire from our algorithm. This is called expressiveness. Natural language is poor for this task. A good example is all the subtlety that the modal operators F and G of LTL allow us to express clearly, concisely, and without ambiguity.

Distributed systems Regular algorithms operate in a centralized computing environment, where a single computing entity executes instructions sequentially to solve a problem. They assume access to a global memory and can rely on a single processor's computational power and resources. Distributed algorithms, on the other hand, are specifically tailored for decentralized computing environments. They involve multiple autonomous entities (nodes or processes) that interact over a network to collaboratively solve a problem. These algorithms must account for the lack of global memory and the need for communication and coordination between distributed entities. Key challenges in distributed algorithms include handling issues of concurrency, synchronization, message passing, fault tolerance, and scalability, which are not typically encountered in regular algorithms.

These changes towards distributed computing are far from minor. It makes the design of algorithms particularly complicated, much more so than in sequential cases, to the extent that even experts still make mistakes, not just easily correctable implementation errors, but rather irreparable mistakes at the design stage. So, verification has the potential to be useful for distributed algorithms. Generally, verification is not a widespread practice in industry because it is costly and time-consuming which makes the benefit/cost ratio not interesting enough. It is reserved for a few niche applications where human lives are at stake (transport, nuclear) or when huge sums of money are involved (finance). Whereas in distributed computing, it already helps to find errors which were unnoticeable otherwise. One example of this is the usage of TLA+ in industry [53, 3], where it helped to find deadlock and other design flaws.

1.2.2 Specialized Motivations

Data logics have been introduced to reason about structures whose elements are labeled with a value from an infinite alphabet (e.g., XML documents) [45]. Expressive decidable fragments include notably two-variable logics over words and trees [7, 6]. The decidability frontier is fragile, though. Extensions to two data values, for example, quickly lead to an undecidable satisfiability problem. From a modeling point of view, those extensions still play an important role. Other types of data logics allow *two* data values to be associated with an element [31, 34], though they do not assume a linearly ordered or tree-like universe. Again, satisfiability turned out to be decidable for the two-variable fragment of first-order logic. When considering an arbitrary number of first-order variables, which we do in this work, the decidability frontier is quickly crossed without further constraints. One of the restrictions we consider here is locality, an essential concept in first-order logic. It is well known that first-order logic is only able to express local properties: a first-order formula can always be written as a combination of properties of elements that have limited, i.e., bounded by a given radius, distance from some reference points [27, 23]. In the presence of (several) data values, imposing a corresponding locality restriction on a logic can help ensuring decidability of its satisfiability problem.

From Chapter 3, we only consider a natural extension of first-order logic over unordered structures whose elements carry a fixed number of data values from an infinite domain. When specifying the *input-output behavior* of distributed algorithms [21, 38], processes get an input value and produce an output value, which requires two data values per process. In leader election or renaming algorithms, for instance, a process gets its unique identifier as input, and it should eventually output the identifier of a common leader (leader election) or a unique identifier from a restricted name space (renaming). There are two major differences between most existing formalisms and our language. While previous data logics are usually interpreted over words or trees, we consider unordered structures (or multisets). When each element of such a structure represents a process, we therefore do not assume a particular processes architecture, but rather consider clouds of computing units. Moreover, decidable data logics are usually limited to one value per element, which would not be sufficient to model an input-output relation. Hence, our models are algebraic structures consisting of a universe and functions assigning to each element a fixed number of integers. We remark that, for many distributed algorithms, the precise data values are not relevant, as a matter of fact, often what really matters is whether or not some data are equal. Like [7, 6], we thus add binary relations that allow us to test if two data values are identical and, for example, whether all output values were already present in the collection of input values (as required for leader election).

The first fundamental question that arises is whether a given specification is consistent. This leads us to the satisfiability problem. While the general logic considered here turns out to be undecidable already in several restricted settings, our main result shows that some interesting fragments preserve decidability. The fragments are *local* logic in the sense that data values can only be compared within the direct neighborhood of a (quantified) reference process.

More Related Work Orthogonal extensions for multiple data values include shuffle expressions for nested data [4] and temporal logics [30, 15]. Other generalizations of data logics allow for an order on data values [39, 47]. The application of formal methods in the context of distributed algorithms is a rather recent but promising approach (cf. for a survey [35]). A particular branch is the area of parameterized systems, which, rather than on data, focuses on the (unbounded) number of processes as the parameter [5, 18]. Other related work includes [17], which considers temporal logics involving quantification over processes but without data, while [1] introduces an (undecidable) variant of propositional dynamic logic that allows one to reason about totally ordered process identifiers in ring architectures. First-order logics for *synthesizing* distributed algorithms were considered in [8, 25].

1.2.3 Mathematical Discussion

In this subsection, we will primarily discuss mathematics. This subsection should be understandable for a mathematician, or at least a logician, who has never studied computer science.

This work in logic is oriented towards verification. The biggest difference from mathematical logic is that we consider only finite structures here. In this subsection only, we will refer to large model theory as the theory that allows models of any cardinality, and finite model theory as the theory that only allows finite models. Therefore, large model theory is the one encountered most frequently in mathematics, while finite model theory predominates in computer science. Far from being a minor distinction, this has many implications, the most significant being that the compactness theorem is false in the finite case. To illustrate this, we can consider the sequence of formulas $(\lambda_n)_n$ which states that the model has at least n elements. Any finite subsequence is satisfiable, but the entire sequence is not. This has positive and negative implications, which are, in sum, immeasurable.

One positive effect is the elimination of strange things like non-standard models and non-standard elements, so for example we do not have to deal with integers which are not an iterated successor of 0. On the negative side, there is the Łoś–Tarski preservation theorem, a classic result in large model theory, which is no longer true in the finite case. This theorem states that a formula preserved under model expansion is equivalent to an existential formula. Another negative implication is that there is no complete proof system, i.e., one that can reflect semantic truth. This is a direct consequence of the absence of a completeness theorem.

In a neutral sense, in broad model theory, the validity problem is semi-decidable (you just need to enumerate all proof trees and see if any tree proves the formula). In finite model theory, on the other hand, it is the satisfiability problem that is semi-decidable (you just need to enumerate all structures and see if any satisfies the formula). It is interesting to note that the problems of validity and satisfiability are complementary to each other, creating a symmetry between the two model theories. Moreover, when combined with the fact that the satisfiability problem is undecidable in general for finite model theory, it provides another argument against the existence of a complete proof system.

1.2.4 Contributions and Organisation of the Thesis

In Chapter 2, we start with Section 2.1 by giving the general definition of structures, logics and satisfiability problems used in this work. Next in Section 2.2, we expose tools frequently used when studying satisfiability problems. Specifically, for upper bounding, we present the small model property and the Ehrenfeucht-Fraïssé games, whereas for lower bounds, we introduce the tiling problem. In Section 2.3 we present a catalogue of the results on the satisfiability problem over words, multisets, data-words, graphs and data-multisets. Additionally, we provide a more accessible proof of the already known result that FO^3 over graphs is undecidable (Theorem 2.3.20). We also prove the known result but with a missing proof of the undecidability of FO over two equivalence relations (Theorem 2.3.27). In fact, our proof improves the result further, down to FO^3 (Theorem 2.3.31).

In Chapter 3, we explore the concept of locality over data-multisets. In Section 3.1 define our local fragment. Then in Section 3.2, we expose fundamental results on the local fragments, which is an original contribution of this work. We start by showing that we can internalize the distance in first-order logic, and then use this to establish inclusions between different fragments. Finally we examine the connection between our notion of data graph with the notion of Gaifman graph.

In Chapter 4, we study the main fragments and depict the frontier between decidability and undecidability. We show that the fragment has a decidable satisfiability problem when restricting local properties to radius 1 (Section 4.2, Theorem 4.2.19). However, for any radius greater than 1 it is undecidable (Section 4.3, Theorems 4.3.5 and 4.3.9). Note that adding only one diagonal relation still constitutes an extension of the (decidable) two-variable first-order logic with two equivalence relations [32, 31, 34]: equivalence classes consist of those elements with the same first value, respectively, second value. In fact, our main technical contribution is a reduction to this two-variable logic. The reduction requires a careful relabelling of the underlying structures so as to be able to express the diagonal relation in terms of the two equivalence relations. In addition, the reduction takes care of the fact that our logic does not restrict the number of variables. We can actually count elements up to some threshold and express, for instance, that at most five processes crash (in the context of distributed algorithms). This is a priori not possible in two-variable logic.

In Chapter 5, we study orthogonal local fragments where global quantification is restricted to being existential (while quantification inside a local modality remains unrestricted). In its first Section 5.1, we obtain decidability for (i) radius 1 and an arbitrary number of data values (Theorem 5.1.9), and for (ii) radius 2 and two data values (Theorem 5.1.6). In all cases, we provide tight complexity upper and lower bounds. Moreover, these results mark the exact decidability frontier: Section 5.2 shows that satisfiability is undecidable as soon as we consider radius 3 in presence of two data values (Theorem 5.2.3), or radius 2 together with three data values (Theorem 5.2.5).

Chapter 2

Exploring Data Logics

The aim of this chapter is to explore established results on logic with data. The first section sets out the definitions needed to talk with ease about this subject. The second section then introduces techniques that often arise when analysing satisfiability problems. Lastly, the third section truly explores logic with data.

2.1 Gearing up: the Definitions

This section is devoted to the definitions. They are general, in order to cover all the cases seen in this work. We first define structures, then logics and finally the satisfiability problem.

2.1.1 Structures

By $\mathbb{N}$ we denote the set of natural numbers including 0 and we define $\overline{\mathbb{N}}$ as $\mathbb{N} \cup \{\infty\}$.

Let Σ be a finite set of unary predicates and Γ a finite set of binary predicates and $\kappa \in \mathbb{N}$ a number of data values. A *signature* Λ is a pair (Γ, κ). So that is, we exclude Σ from the signature, as we think that Γ and κ are more important. Furthermore, Γ and κ will be parameters to the later defined satisfiability problem, whereas we want that Σ is an input. The reason for this choice is that to classify structures, Γ and κ are more important than Σ. Indeed, relationships between pairs of elements are more important than the labelling of a single element as it is the most significant parameter for the decidability and complexity of the satisfiability problem. Another reason is to be able to refer to a class of structures without specifying a set of unary predicates. Yet it is required to specify Σ if we want to talk about an instance of a signature. A (Σ, Λ)-*structure* is a tuple

$$\mathfrak{A} = (A, (P_\sigma^{\mathfrak{A}})_{\sigma \in \Sigma}, (R_\gamma^{\mathfrak{A}})_{\gamma \in \Gamma}, f_1^{\mathfrak{A}}, \dots, f_\kappa^{\mathfrak{A}})$$

where A is a non-empty finite set, $P_\sigma^{\mathfrak{A}} \subseteq A$ for each σ, $R_\gamma^{\mathfrak{A}} \subseteq A \times A$ for each γ and $f_i^{\mathfrak{A}}$s are mappings $A \to \mathbb{N}$ called *data functions* assigning a *data value* to each element. We call a pair $(a, i) \in A \times \{1, \dots, \kappa\}$ a *data location* and it stores the data value $f_i^{\mathfrak{A}}(a)$. For $X \subseteq A$, we let $\mathit{Val}^{\mathfrak{A}}(X) = \{f_i^{\mathfrak{A}}(a) \mid a \in X, i \in \{1, \dots, \kappa\}\}$. When $\mathfrak{A}$ is clear from the context, we can omit it and simply write P_σ, R_γ and f_i instead of $P^{\mathfrak{A}}_\sigma$, $R^{\mathfrak{A}}_\gamma$ and $f^{\mathfrak{A}}_i$. For $\Lambda = (\Gamma, \kappa)$, we denote by $\mathrm{Str}(\Sigma; \Lambda)$ the set of (Σ, Λ)-structures (or structures over Σ and Γ with κ data values). Among Γ, we ask for the symbol $<$ to always be interpreted as a linear order over A.

2.1.2 Adding Interpretations

When defining structures, some information might be superfluous. For example, consider the two binary symbols $<$ and $+1$, and ask $+1$ to always be interpreted as the successor of $<$ (recall that $<$ is a linear order). The class of structures $\mathrm{Str}(\Sigma; (\{<\}, \kappa))$ and $\mathrm{Str}(\Sigma; (\{<, +1\}, \kappa))$ are equivalent in a very simple way: from any structure in $\mathrm{Str}(\Sigma; (\{<\}, \kappa))$, there is exactly one possibility to interpret $+1$ and from any structure in $\mathrm{Str}(\Sigma; (\{<, +1\}, \kappa))$ it suffices to forget the interpretation of $+1$. We thus make the choice to define structures in a minimal way: that is, we prefer $\mathrm{Str}(\Sigma; (\{<\}, \kappa))$ over $\mathrm{Str}(\Sigma; (\{<, +1\}, \kappa))$. Yet, we will sometimes want to have access to the relation $+1$ and it is not always definable in the logic (e.g. with two-variable first order logic defined later). Our solution is to be able to interpret the relation $+1$ when needed.

We introduce the symbols $+1$, $\tilde{+}1$, ${}_i{\sim}_j$ and $\sim$, for each of them we describe on which structures they can be interpreted and what is the interpretation then:

$+1$ is the *successor* of $<$. It can be interpreted on any structure $\mathfrak{A}$ interpreting $<$ (we recall that $<$ has to be a linear order on A) and then the interpretation of $+1$ is defined as $R^{\mathfrak{A}}_{+1} = \{(a, b) \in A \times A \mid a < b$ and there is no $c \in A$ such that $a < c$ and $c < b\}$. We will write $y = x + 1$ instead of $+1(x, y)$.

Example 2.1.1. If $\mathfrak{A} = (\{a, b, c, d\}, R_<) \in \mathrm{Str}(\emptyset; \{<\})$ such that $a < b < c < d$ on $\mathfrak{A}$, we have $R^{\mathfrak{A}}_{+1} = \{(a, b), (b, c), (c, d)\}$. ◇

$\tilde{+}1$ is the *class successor*. It can be interpreted on any structure $\mathfrak{A}$ interpreting $<$ and with $\kappa = 1$. Then the interpretation of $\tilde{+}1$ is defined as $R^{\mathfrak{A}}_{\tilde{+}1} = \{(a, b) \in A \times A \mid a < b$ and $f(a) = f(b)$ and there is no $c \in A$ such that $a < c$ and $c < b$ and $f(a) = f(c)\}$. We will write $y = x\tilde{+}1$ instead of $\tilde{+}1(x, y)$. See section 2.3.3 for more content on the class successor.

${}_i{\sim}_j$ to compare data values for $1 \leq i, j \leq \kappa$. It can be interpreted when $\kappa > 0$. The interpretation of ${}_i{\sim}_j$ is defined as $R^{\mathfrak{A}}_{{}_i{\sim}_j} = \{(a, b) \in A \times A \mid f_i(a) = f_j(b)\}$. We will write $x \;{}_i{\sim}_j\; y$ instead of ${}_i{\sim}_j(x, y)$. Theses symbols will be used later in the logic to test the equality of data values. If $\kappa = 1$ we may write $\sim$ for ${}_1{\sim}_1$. For convenience we define the sets $\mathcal{A}_\kappa, \mathcal{I}_\kappa$ and $\mathcal{S}_\kappa$ of symbols as

$$\mathcal{A}_\kappa = \{{}_i{\sim}_j \mid 1 \leq i, j \leq \kappa\}, \tag{2.1.2}$$

$$\mathcal{I}_\kappa = \{{}_i{\sim}_j \mid 1 \leq i \leq j \leq \kappa\}, \tag{2.1.3}$$

$$\mathcal{S}_\kappa = \{{}_i{\sim}_i \mid 1 \leq i \leq \kappa\}. \tag{2.1.4}$$

The set $\mathcal{A}_\kappa$ allows us to test the equality of any data values and the set $\mathcal{S}_\kappa$ allows us to test the equality of data values with the same position ($\mathcal{A}$ stands for All, $\mathcal{I}$ stands for Increasing and $\mathcal{S}$ stands for Same). They are the most employed binary symbols in this work.

2.1.3 Logics

Let $\mathcal{V} = \{x, y, \ldots\}$ be a countably infinite set of variables. Let $\mathcal{R}$ be a set of binary symbols taken among Γ, $\mathcal{A}_\kappa$ and the additional relations $+1$ and ∓ 1. That is, $\mathcal{R} \subseteq \Gamma \cup \mathcal{A}_\kappa \cup \{+1, \mp 1\}$. The set $\mathsf{FO}_\Lambda[\Sigma; \mathcal{R}]$ of *first order formulas* interpreted over (Σ, Λ)-structures is inductively given by the grammar

$$\varphi ::= \sigma(x) \mid \gamma(x, y) \mid x = y \mid \varphi \vee \varphi \mid \neg\varphi \mid \exists x.\varphi,$$

where x and y range over $\mathcal{V}$, σ ranges over Σ, and γ ranges over $\mathcal{R}$. We use standard abbreviations such as $\wedge$ for conjunction and $\rightarrow$ for implication. For certain symbols $\gamma \in \mathcal{R}$, we write $\gamma(x, y)$ in a more natural way: $x < y$ instead of $<(x, y)$ and $y = x + 1$ instead of $+1(x, y)$ and $y = x \mp 1$ instead of $\mp 1(x, y)$ and $x\,{}_i{\sim}_j\,y$ instead of ${}_i{\sim}_j(x, y)$. Given a formula φ, we define its set of *free variables* $FV(\varphi)$ inductively by

$$\begin{aligned} FV(\sigma(x)) &= \{x\} & FV(\varphi \vee \varphi') &= FV(\varphi) \cup FV(\varphi') \\ FV(\gamma(x, y)) &= \{x, y\} & FV(\neg\varphi) &= FV(\varphi) \\ FV(x = y) &= \{x, y\} & FV(\exists x.\varphi) &= FV(\varphi) \setminus \{x\} \end{aligned}$$

We write $\varphi(x_1, \ldots, x_k)$ to indicate that the free variables of φ are among $x_1, \ldots, x_k$. Moreover, we call φ a *sentence* if it does not contain free variables, i.e. $FV(\varphi) = \emptyset$.

For $\mathfrak{A} = (A, (P_\sigma)_{\sigma\in\Sigma}, (R_\gamma)_{\gamma\in\Gamma}, f_1, \ldots, f_\kappa) \in \mathrm{Str}(\Sigma; \Lambda)$ and a formula $\varphi \in \mathsf{FO}_\Lambda[\Sigma; \mathcal{R}]$, the *satisfaction relation* $\mathfrak{A} \models_I \varphi$ is defined with respect to an *interpretation function* $I : \mathcal{V} \to A$. The purpose of I is to assign an interpretation to every (free) variable of φ so that φ can be assigned a truth value. For $x \in \mathcal{V}$ and $a \in A$, the interpretation function $I[x/a]$ maps x to a and coincides with I on all other variables. We then define:

$$\begin{array}{ll} \mathfrak{A} \models_I \sigma(x) & \text{if } I(x) \in P_\sigma \\ \mathfrak{A} \models_I \gamma(x, y) & \text{if } (I(x), I(y)) \in R_\gamma \\ \mathfrak{A} \models_I x = y & \text{if } I(x) = I(y) \\ \mathfrak{A} \models_I \varphi_1 \vee \varphi_2 & \text{if } \mathfrak{A} \models_I \varphi_1 \text{ or } \mathfrak{A} \models_I \varphi_2 \\ \mathfrak{A} \models_I \neg\varphi & \text{if } \mathfrak{A} \not\models_I \varphi \\ \mathfrak{A} \models_I \exists x.\varphi & \text{if there is } a \in A \text{ s.t. } \mathfrak{A} \models_{I[x/a]} \varphi \end{array}$$

In particular, for a sentence φ (without free variables), we write $\mathfrak{A} \models \varphi$ if there exists an interpretation function I such that $\mathfrak{A} \models_I \varphi$.

For a structure $\mathfrak{A}$, a formula $\varphi(x_1, \ldots, x_k)$ and $a_1, \ldots, a_k \in A$, we write $\mathfrak{A} \models \varphi(a_1, \ldots a_k)$ if there exists an interpretation function I such that $\mathfrak{A} \models_{I[x_1/a_1]\ldots[x_k/a_k]} \varphi$. In the case of the atomic formulas $x\,{}_i{\sim}_j\,y$, we can write $a_1\,{}_i{\sim}_j^{\mathfrak{A}}\,a_2$ instead of $\mathfrak{A} \models a_1\,{}_i{\sim}_j\,a_2$

We recall the definition of $\overline{\mathbb{N}}$ as $\mathbb{N} \cup \{\infty\}$. Given $k \in \overline{\mathbb{N}}$, we define *k-variable first order* *logic* $\mathsf{FO}^k_\Lambda[\Sigma;\mathcal{R}]$ as the fragment consisting of $\varphi \in \mathsf{FO}_\Lambda[\Sigma;\mathcal{R}]$ such that in each sub formula φ' of φ, the number of free variable of φ' is at most k. Up to renaming variables, it is equivalent to ask for φ to be written with up to k different variables. Note that with $k = \infty$, we have $\mathsf{FO}^\infty_\Lambda[\Sigma;\mathcal{R}] = \mathsf{FO}_\Lambda[\Sigma;\mathcal{R}]$.

2.1.4 The Satisfiability Problem

We are now able to introduce the *satisfiability problem*, which is the central problem we will focus on. It is parametrised by a class of structures and a logic. Let $\mathcal{L}$ denote a generic class of first-order formulas, e.g. FO or FO^k. In particular, when $\mathcal{L} = \mathsf{FO}$, we have $\mathcal{L}_\Lambda[\Sigma;\mathcal{R}] = \mathsf{FO}_\Lambda[\Sigma;\mathcal{R}]$. We recall that $\mathcal{R}$ is a set of binary symbols. The problem Λ-Sat($\mathcal{L};\mathcal{R}$) for Λ, $\mathcal{L}$ and $\mathcal{R}$ is defined as follows :

Λ-Sat($\mathcal{L};\mathcal{R}$)	
Input:	Finite set Σ; sentence $\varphi \in \mathcal{L}_\Lambda[\Sigma;\mathcal{R}]$.
Question:	Is there $\mathfrak{A} \in \mathrm{Str}(\Sigma;\Lambda)$ such that $\mathfrak{A} \models \varphi$?

2.2 Physical Training: Recurring Techniques

In this section, we expose a proof schema which often arises when considering satisfiability problems. The schema splits in two, first how to show that a problem is feasible and then how to show that one cannot do better. So in the first subsection we expose the techniques for upper bounds: the small model property and Ehrenfeucht-Fraïssé games and in the second subsection expose the technique for lower bounds: the tilling problems. In both subsections we will illustrate this by analysing the satisfiability problem for multisets.

2.2.1 For Upper Bounds: Small Model Property and Ehrenfeucht-Fraïssé Games

This subsection will focus on directly proving that a logic is decidable. To this end, one idea is that given a formula, to enumerate all the structures while checking if one of them satisfies the formula. This procedure works if the formula is satisfiable, but will never terminate if the formula is not. To remedy this issue, the key idea is to find a way to restrain the search space and this is achieved thanks to the *small model property*. So first we introduce the small model property and show how it induces decidability. Then, we introduce Ehrenfeucht-Fraïssé games and show how it can help to establish a small model property. Finally we practice on multisets and in order to use what we have learned.

The Small Model Properties Put simply, the small model property is about the maximal size needed to satisfy a formula. Here is a formal formulation of it.

Definition 2.2.1 (Small model property)**.** Let $\mathcal{L}$ be a class of FO formulas, Λ a signature, $\mathcal{R}$ a set a binary symbols and $f : \mathbb{N} \to \mathbb{N}$. We say that the logic $\mathcal{L}_\Lambda[-;\mathcal{R}]$ has the *small* *model property* of size f when for all set Σ and sentence $\varphi \in \mathcal{L}_\Lambda[\Sigma; \mathcal{R}]$, if φ is satisfiable, then φ has a model of size at most $\mathcal{O}(f(|\varphi|))$.

We shall not see right now how one can obtain a small model property but rather its utility for decidability. We show that a small model property always entails a decidability result, as stated in the following Lemma:

Lemma 2.2.2. *If the logic $\mathcal{L}_\Lambda[-; \mathcal{R}]$ has a small model property of size f then the problem Λ-*Sat*$(\mathcal{L}; \mathcal{R})$ belongs to* NTime$(n \cdot (c \cdot f(n))^n)$ *for some constant c.*

Corollary 2.2.3. *Using notations of the lemma 2.2.2, if we have $f(n) = \mathcal{O}(n2^n)$ then the problem Λ-*Sat*$(\mathcal{L}; \mathcal{R})$ belongs to* NExp*.*

Before proving the lemma, we have to state a proposition on model checking.

Proposition 2.2.4 ([37, Proposition 6.6])**.** *Given Σ, a sentence $\varphi \in \mathcal{L}_\Lambda[\Sigma; \mathcal{R}]$ and a structure $\mathfrak{A} \in \mathrm{Str}(\Sigma; \Lambda)$, deciding if $\mathfrak{A} \models \varphi$ can be done in time $\mathcal{O}(|\varphi| \cdot |\mathfrak{A}|^{|\varphi|})$.*

We are now ready to prove Lemma 2.2.2.

Proof of Lemma 2.2.2: Let c be the constant hidden behind the big O of the assumption of the small model property. The procedure is simple and as follow: Given φ, we non-deterministically choose $\mathfrak{A} \in \mathrm{Str}(\Sigma; \Lambda)$ of size less than $c \cdot f(|\varphi|)$ and then accept φ if $\mathfrak{A} \models \varphi$. First, thanks to lemma 2.2.4, the procedure have the claimed complexity. Second, we have to show that the procedure is correct. It is the case since by assumption, we have a small model property, which implies that if φ is satisfiable, then φ has a model of size at most $c \cdot f(|\varphi|)$, so there is an accepting run. On the other hand, if φ is not satisfiable, then even more so φ doesn't have a model of size at most $c \cdot f(|\varphi|)$, so there is no accepting run. □

We have seen that small model property gives us decidability for satisfiability problems. We will now see how to obtain small model property. We want from a formula φ and a model $\mathfrak{A}$ of it, to exhibit a new structure $\mathfrak{B}$ of reasonable size which also satisfies φ. As the formula φ is general, we will in fact prove something more general. That the structures $\mathfrak{A}$ and $\mathfrak{B}$ are undistinguishable by a subset of formula to which φ belongs. And as $\mathfrak{A}$ is a model of φ, it therefore follows that $\mathfrak{B}$ is also a model of φ.

Ehrenfeucht-Fraïssé Games This presentation is highly inspired by [37]. We now define *Ehrenfeucht-Fraïssé games*. It takes three parameters, two structures on the same signature and alphabet $\mathfrak{A}$ and $\mathfrak{B}$ and an integer q which specify the number of rounds. We denote this game $\mathrm{EF}_q(\mathfrak{A}, \mathfrak{B})$. There are two players, namely the *Spoiler* and the *Duplicator*. The game consists of q rounds constituted of:

1. The Spoiler picks a structure ($\mathfrak{A}$ or $\mathfrak{B}$).
2. The Spoiler makes a move by picking an element of that structure: either $a \in A$ or $b \in B$.

3. The Duplicator responds by picking an element in the other structure.

At the end, we have two tuples $(a_1, \dots, a_q)$ and $(b_1, \dots, b_q)$. As we define a game, we need to define winning positions. In this situation, we say that the tuples $(a_1, \dots, a_q)$ and $(b_1, \dots, b_q)$ defines a *partial isomorphism* between $\mathfrak{A}$ and $\mathfrak{B}$ if the three following conditions hold:

- For any $1 \leq i, j \leq q$, we have $a_i = a_j$ iff $b_i = b_j$.
- For any $1 \leq i \leq q$ and $\sigma \in \Sigma$, we have $a_i \in P_\sigma^{\mathfrak{A}}$ iff $b_i \in P_\sigma^{\mathfrak{B}}$.
- For any $1 \leq i, j \leq q$ and $\gamma \in \mathcal{R}$, we have $(a_i, a_j) \in R_\gamma^{\mathfrak{A}}$ iff $(b_i, b_j) \in R_\gamma^{\mathfrak{B}}$.

In other words the substructures of $\mathfrak{A}$ generated by $\{a_1, \dots, a_q\}$ and $\mathfrak{B}$ generated by $\{b_1, \dots, b_q\}$ are isomorphic through the function sending a_i to b_i. We say that Duplicator has won the game $\mathrm{EF}_q(\mathfrak{A}, \mathfrak{B})$ if the tuples $(a_1, \dots, a_q)$ and $(b_1, \dots, b_q)$ is a partial isomorphism between $\mathfrak{A}$ and $\mathfrak{B}$. We say that the Duplicator has an q-round winning strategy in the Ehrenfeucht-Fraïssé game on $\mathfrak{A}$ and $\mathfrak{B}$ if the Duplicator can play in a way that guarantees a winning position after q-rounds, no matter how the Spoiler plays. If the Duplicator has a q-round winning strategy, we write $\mathfrak{A} \rightleftharpoons_q \mathfrak{B}$.

We now define an another binary relation on structures, denoted $\equiv_q$. If we see a formula φ as a tree, its *quantifier height* counts the maximum number of quantifier we see while going through a branch of φ. It is denoted $\mathrm{qh}(\varphi)$ and defined inductively on φ as follow:

$$\begin{aligned} \mathrm{qh}(\sigma(x)) &= 0 & \mathrm{qh}(\varphi \vee \varphi') &= \max(\mathrm{qh}(\varphi), \mathrm{qh}(\varphi')) \\ \mathrm{qh}(\gamma(x, y)) &= 0 & \mathrm{qh}(\neg\varphi) &= \mathrm{qh}(\varphi) \\ \mathrm{qh}(x = y) &= 0 & \mathrm{qh}(\exists x.\varphi) &= 1 + \mathrm{qh}(\varphi) \end{aligned}$$

Let $\mathfrak{A}, \mathfrak{B} \in \mathrm{Str}(\Sigma; \Lambda)$ be two structures. We say that $\mathfrak{A}$ and $\mathfrak{B}$ agree on formulas of quantifier height at most q if for any sentence $\varphi \in \mathsf{FO}_\Lambda[\Sigma; \mathcal{R}]$ with $\mathrm{qh}(\varphi) \leq q$ we have $\mathfrak{A} \models \varphi$ iff $\mathfrak{B} \models \varphi$. We write it $\mathfrak{A} \equiv_q \mathfrak{B}$.

We introduce now the Ehrenfeucht-Fraïssé Theorem which show how Ehrenfeucht-Fraïssé-games characterize precisely the expressiveness of first order logic. A proof can be found in [37].

Theorem 2.2.5 (*Ehrenfeucht-Fraïssé [37]*)

Let $\mathfrak{A}, \mathfrak{B} \in \mathrm{Str}(\Sigma; \Lambda)$ be two structures and q an integer. We have:

$$\mathfrak{A} \rightleftharpoons_q \mathfrak{B} \text{ iff } \mathfrak{A} \equiv_q \mathfrak{B}.$$

Practice on Multisets We now apply the theory developed earlier in this subsection to the multisets. It is the simplest case one can consider in our framework. We choose this application to showcase the technique. We introduce the signature $\mathbb{MS} = (\emptyset, 0)$ which represents the multisets. A multiset $\mathfrak{A} \in \mathrm{Str}(\Sigma; \mathbb{MS})$ is a tuple $(A, (P_\sigma)_{\sigma \in \Sigma})$ where A is a finite set and the P_σs are subsets of A. That is, a set of element with labelling and without any relations between the elements.

In order to show the decidability of FO over multisets, we aim at a small model property stated hereinafter:

Lemma 2.2.6. *Let $\varphi \in \mathsf{FO}_{\mathbb{MS}}[\Sigma;\emptyset]$. If φ is satisfiable, then φ has a model of size at most* $\mathrm{qh}(\varphi)\cdot 2^{|\Sigma|}$.

In order to prove the small model property, we want to find Ehrenfeucht-Fraïssé-games in which the Duplicator has a winning strategy. To this end, we introduce the relation $\cong_q$ on structures which will imply the relation $\rightleftharpoons_q$. For $U \subseteq \Sigma$ subset of unary predicates, we define the subset $E_U(\mathfrak{A})$ of A as $\{a \in A \mid \text{ for all } \sigma \in \Sigma, a \in P_\sigma \text{ iff } \sigma \in U\}$ and then we define $\#_U(\mathfrak{A})$ as the cardinal $|E_U(\mathfrak{A})|$. Let $\cong_q$ be the binary relation on $\mathrm{Str}(\Sigma;\mathbb{MS})$ defined as $\mathfrak{A} \cong_q \mathfrak{B}$ if for all $U \subseteq \Sigma$, either $\#_U(\mathfrak{A}) = \#_U(\mathfrak{B})$ or $\#_U(\mathfrak{A}) \geq q$ and $\#_U(\mathfrak{B}) \geq q$.

The next lemma describes a sufficient condition for the Duplicator to have a winning strategy:

Lemma 2.2.7. *For any structures $\mathfrak{A}, \mathfrak{B} \in \mathrm{Str}(\Sigma;\Lambda)$ and integer q, we have:*

$$\mathfrak{A} \cong_q \mathfrak{B} \textit{ implies } \mathfrak{A} \rightleftharpoons_q \mathfrak{B}$$

The Lemma is in fact an equivalence but for our purpose an implication suffices. For that matter, on other signature than $\mathbb{MS}$, it is unusual to be able to characterize $\rightleftharpoons_q$, i.e. it is impossible to pin down exactly when the Duplicator have a winning strategy.

Proof of Lemma 2.2.7: Assume that $\mathfrak{A} \cong_q \mathfrak{B}$, we want to show that the Duplicator has a winning strategy. Its strategy is simply to maintain a partial isomorphism at every turn. By symmetry on $\mathfrak{A}$ and $\mathfrak{B}$, the strategy is possible if for any $k < q$, $(a_1,\dots,a_k) \in A$ and $(b_1,\dots,b_k) \in B$ in partial isomorphism, and for any $a \in A$, there is a $b \in B$ such that $(a_1,\dots,a_k,a)$ and $(b_1,\dots,b_k,b)$ are in partial isomorphism. If there is a $i_0 \leq k$ such that $a_{i_0} = a$, Duplicator takes $b = b_{i_0}$ and it's easy to check that $(a_1,\dots,a_k,a)$ and $(b_1,\dots,b_k,b)$ are in partial isomorphism. So we now assume that $a \notin \{a_1,\dots,a_k\}$. Let $U \subseteq \Sigma$ defined with $U = \{\sigma \in \Sigma \mid a \in P_\sigma^{\mathfrak{A}}\}$. Let denote by α the cardinality of $E_U(\mathfrak{A}) \cap \{a_1,\dots,a_k\}$ and by β the cardinality of $E_U(\mathfrak{B}) \cap \{b_1,\dots,b_k\}$. By assumptions, we have that $\alpha = \beta \leq k$ and $\alpha < \#_U(\mathfrak{A})$ and we want $\#_U(\mathfrak{B}) > \beta$. Since $\mathfrak{A} \cong_q \mathfrak{B}$, we do a disjunction on whether $\#_U(\mathfrak{A}) < q$ or not. If $\#_U(\mathfrak{A}) < q$, we have $\#_U(\mathfrak{B}) = \#_U(\mathfrak{A}) > \alpha = \beta$. If $\#_U(\mathfrak{A}) \geq q$, we have $\#_U(\mathfrak{B}) \geq q > k \geq \beta$. In both cases, we have $\#_U(\mathfrak{B}) > \beta$ so we can pick for b any element of $E_U(\mathfrak{B}) \setminus \{b_1,\dots,b_k\}$ which is nonempty. □

Proof of Lemma 2.2.6: Let $\varphi \in \mathsf{FO}_{\mathbb{MS}}[\Sigma;\emptyset]$ satisfiable. Let $q = \mathrm{qh}(\varphi)$ and $\mathfrak{A} = (A, (P_\sigma)_{\sigma\in\Sigma})$ be a model of φ. We will construct $\mathfrak{B} = (B, (P'_\sigma)_{\sigma\in\Sigma}) \in \mathrm{Str}(\Sigma;\mathbb{MS})$ a model a φ with $|B| \leq q \cdot 2^{|\Sigma|}$. For all $U \subseteq \Sigma$, we define $B_U \subseteq E_U(\mathfrak{A})$ as:

$$B_U := \begin{cases} E_U(\mathfrak{A}) & \text{if } \#_U(\mathfrak{A}) < q \\ \text{A subset of } E_U(\mathfrak{A}) \text{ of cardinality } q & \text{if } \#_U(\mathfrak{A}) \geq q. \end{cases}$$

Then we define B as:

$$B := \bigcup_{U \subseteq \Sigma} B_U$$
$$P'_\sigma := P_\sigma \cap B$$

By construction of $\mathfrak{B}$, we have $\mathfrak{A} \cong_q \mathfrak{B}$. By Lemma 2.2.7 we have $\mathfrak{A} \rightleftharpoons_q \mathfrak{B}$, that is Duplicator has a winning strategy for the game $\mathrm{EF}_q(\mathfrak{A}, \mathfrak{B})$. And as $q = \mathrm{qh}(\varphi)$ by Ehrenfeucht-Fraïssé Theorem 2.2.5, we have that $\mathfrak{A}$ and $\mathfrak{B}$ are undistinguishable by φ, so $\mathfrak{B} \models \varphi$. Lastly the cardinality of B is indeed less than $q \cdot 2^{|\Sigma|}$. □

Theorem 2.2.8

The problems $\mathbb{MS}$-SAT$(\mathsf{FO}; \emptyset)$ *is in* NEXP.

Proof: We want to apply Lemma 2.2.2 with $f(n) = n2^n$. So we want to show that FO over multisets has the small model property of size f. Let $\varphi \in \mathsf{FO}_{\mathbb{MS}}[\Sigma; \emptyset]$ be a satisfiable formula. We can assume that $|\Sigma| \leq |\varphi|$ since there are at most $|\varphi|$ unary predicates appearing in φ. Thanks to Lemma 2.2.6, let $\mathfrak{A} \in \mathrm{Str}(\Sigma; \mathbb{MS})$ with $|\mathfrak{A}| \leq \mathrm{qh}(\varphi) \cdot 2^{|\Sigma|}$. We have that :

$$|\mathfrak{A}| \leq \mathrm{qh}(\varphi) \cdot 2^{|\Sigma|} \leq |\varphi| 2^{|\varphi|} = f(|\varphi|).$$

So we can apply Lemma 2.2.2 which gives us that $\mathbb{MS}$-SAT$(\mathsf{FO}; \emptyset)$ is in NEXP. □

2.2.2 For Lower Bounds: Domino Tilings

To analyse satisfiability problems, the tiling problem is a crucial tool for proving lower bounds. By lower bounds we are referring to nondeterministic-time or space complexity lower bounds, as well as undecidability results. The tiling problem is fundamental for three reasons. Firstly, it is closely related to the halting problem of Turing machines. The idea of the reduction from Turing machine to tiling is simple; although the proof is technical. Secondly, it is simple to state and needs no prerequisites. Indeed, the tiling problem is easily comprehensible even for a layperson. Lastly, it is a cornerstone to prove a large amount of lower bounds on satisfiability problems. For example, the original paper from Cook [13] on the Cook–Levin Theorem cites tiling problems. Furthermore, it factors usual proof and achieves to hide all the technicalities of encoding a Turing machine.

On the previous subsection 2.2.1 we described a way to obtain upper bounds through small model property. It may seems that those upper bounds on complexity obtained are sub-optimal, but this is not the case: a corresponding lower bound is very oftenly observed. This is due to the fact that a proof of an optimal small model property frequently yields a lower bound on the logic. This is done using tiling problems, which will be exposed in this subsection. But this implication is not automatic and requires a bit of ingenuity. However, in the next section 2.3, it is the case for all catalogued logics. To the best of my knowledge the only counterexample is LTL, where its satisfiability problem is PSPACE-complete whereas it satisfies only an exponential ultimately periodic model property (an equivalent of small model property for LTL) which only gives an NEXP upper bound. But LTL is out of scope of this work since its models are infinite.

The outline of this subsection is to first define the tiling problems, then as an example we use them to prove a NEXP lower bound and finally we expose some techniques which help to prove undecidability results.

Tiling problems A *domino system* $\mathcal{D}$ is a triple (D, H, V) where D is a finite set of dominoes and $H, V \subseteq D \times D$ are two binary relations[1]. For $m, n \in \mathbb{N}$, let $\mathfrak{G}_{m,n}$ denotes the standard grid on an $m \times n$ torus, i.e., $\mathfrak{G}_{m,n} = (G_{m,n}, H_{m,n}, V_{m,n})$ where $H_{m,n}$ and $V_{m,n}$ are two binary relations defined as follows:

$$\begin{aligned} G_{m,n} &= \mathbb{Z} \bmod m \times \mathbb{Z} \bmod n, \\ H_{m,n} &= \{((i,j),(i',j)) \mid i' - i \equiv 1 \mod m\}, \\ V_{m,n} &= \{((i,j),(i,j')) \mid j' - j \equiv 1 \mod n\}. \end{aligned}$$

When $m = n$ we can write $\mathfrak{G}_m = (G_m, H_m, V_m)$ instead of $\mathfrak{G}_{m,n} = (G_{m,n}, H_{m,n}, V_{m,n})$. Let H and V be two binary predicates and let $\mathbb{BB}$ be the signature

$$\mathbb{BB} = (\{\mathsf{H}, \mathsf{V}\}, 0).$$

Notice that both domino systems and the standard grids can be seen as $(\emptyset, \mathbb{BB})$-structures. We call elements of the collection $\mathrm{Str}(\emptyset; \mathbb{BB})$ *bi-binary structures*. For two bi-binary structures $\mathfrak{G} = (G, H, V)$ and $\mathfrak{G}' = (G', H', V')$, we say that $\mathfrak{G}$ is *homomorphically embeddable* into $\mathfrak{G}'$ if there is a morphism $\pi : \mathfrak{G} \to \mathfrak{G}'$, i.e., a mapping π such that, for all $a, a' \in G$, we have $(a, a') \in H \Rightarrow (\pi(a), \pi(a')) \in H'$ and $(a, a') \in V \Rightarrow (\pi(a), \pi(a')) \in V'$. For instance, $\mathfrak{G}_{k \cdot m}$ is homomorphically embeddable into $\mathfrak{G}_m$ through reduction mod m. For a domino system $\mathcal{D}$, a *periodic tiling* is a morphism $\pi : \mathfrak{G}_{m,n} \to \mathcal{D}$ for some m, n and we say that $\mathcal{D}$ *admits a periodic tiling* if there exists a periodic tiling for $\mathcal{D}$. We define the size of a tiling $\pi : \mathfrak{G}_{m,n} \to \mathcal{D}$ as the pair (m, n). Tilings of size (m, m) are said to be square.

From the notion of tiling, we define three decision problems, the first two depending on a function $f : \mathbb{N} \to \mathbb{N}$. The first one BOUNDED-TILING(f) asks if a domino system can tile a square grid of some given size, the second one CORRIDOR-TILING(f) asks if a domino system can tile a grid with one dimension fixed and the other one unbounded and the last one UNBOUNDED-TILING asks if a domino system admits a periodic tiling. The first one captures any non-deterministic time complexity classes, the second one any space complexity classes and the last one is undecidable.

BOUNDED-TILING(f)

Input:	A domino system $\mathcal{D}$ and an integer n encoded in unary.
Question:	Does $\mathcal{D}$ admit a periodic tiling of size $(f(n), f(n))$?

[1]This definition of a domino system does not match the intuitive definition with colours. For a proof of the equivalence, see the book [12].

We allow ourselfs to directly write the expression $f(n)$ instead of f. For instance we can write BOUNDED-TILING(n) instead of BOUNDED-TILING($n \mapsto n$).

Proposition 2.2.9 ([52, 12]). *The problem* BOUNDED-TILING(f) *is* NTIME(f)*-complete.*

For instance, the problem BOUNDED-TILING(n) is NP-complete, whereas the problem BOUNDED-TILING(2^n) is NEXP-complete, and the problem BOUNDED-TILING(2^{2^n}) is N2EXP-complete. Defining $G : \mathbb{N} \to \mathbb{N}$ as:

$$G(n) \longmapsto \begin{cases} 2 \text{ if } n = 0, \\ 2^{G(n-1)} \text{ else,} \end{cases}$$

the problem BOUNDED-TILING(G) is TOWER-complete.

CORRIDOR-TILING(f)	
Input:	A domino system $\mathcal{D}$ and an integer n encoded in unary.
Question:	Does $\mathcal{D}$ admit a periodic tiling of size $(f(n), m)$, for some m?

Proposition 2.2.10 ([52]). *The problem* CORRIDOR-TILING(f) *is* SPACE(f)*-complete.*

For instance, the problem CORRIDOR-TILING(n) is PSPACE-complete, whereas the problem CORRIDOR-TILING(2^n) is EXPSPACE-complete, and the problem BOUNDED-TILING(2^{2^n}) is 2EXPSPACE-complete.

UNBOUNDED-TILING	
Input:	A domino system $\mathcal{D}$.
Question:	Does $\mathcal{D}$ admit a square periodic tiling?

Proposition 2.2.11 ([52, 12]). *The problem* UNBOUNDED-TILING *is undecidable.*

Practice on Satisfiability on Multisets We now apply the theory developed earlier in this subsection to the multisets,as for the previous subsection 2.2.1 on upper bounds techniques. We recall that the signature of multiset is $\mathbb{MS} = (\emptyset, 0)$. We recall too that a multiset $\mathfrak{A} \in \mathrm{Str}(\Sigma; \mathbb{MS})$ is a tuple $(A, (P_\sigma)_{\sigma \in \Sigma})$ where A is a finite set and the P_σs are subsets of A.

We rely on the problem BOUNDED-TILING(2^n) in order to get our first lower bound result:

Theorem 2.2.12

The problem $\mathbb{MS}$-SAT(FO; $\emptyset$) *is* NEXP*-hard.*

Before proving Theorem 2.2.12, we now show that the small model property shown in Lemma 2.2.6 is optimal. While this step in not required, it is interesting for two reasons. First because parts of the proof will be reused for proving Theorem 2.2.12 and make the whole proof easier to understand. Second because it is a pattern that arises almost everywhere. The pattern is that a lower bound on the small model property implies a lower bound on the complexity of the satisfiability problem associated. It is more a heuristic rather than a formal proof which contrasts with the discussion on upper bounds in Subsection 2.2.1.

The following Proposition tells us that for FO over multisets, we cannot get better than an exponential small model property.

Proposition 2.2.13. *There are sequences $(\Sigma_n)_n$ of unary predicates and $(\varphi_n)_n$ of formulas in $\mathsf{FO}_{\mathbb{MS}}[\Sigma_n;\emptyset]$ such that the size of φ_n is growing polynomially while the smallest model of φ_n grows exponentially.*

Proof: Let $\Sigma_n = \{c_0, c_1, \dots, c_{n-1}\}$ be a set of n unary predicates and c denote the sequence of c_is. We will see the elements of Σ_n as bits and then any element of a structure interpreting them will represent an integer from 0 to $2^n - 1$. We use the convention that c_0 represents the bit of least weight. With this, we build a formula $\varphi^{c,n}_{zero}(x) \in \mathsf{FO}_{\mathbb{MS}}[\Sigma_n;\emptyset]$ with one free variable asking that the element represents 0 and a second formula $\varphi^{c,n}_{succ}(x,y) \in \mathsf{FO}_{\mathbb{MS}}[\Sigma_n;\emptyset]$ with two free variables asking that the integer represented by y is the successor of the integer represented by x modulo 2^n (successor of $2^n - 1$ being 0). Finally we define φ_n by asking for the existence of an element representing 0 and that every element has a successor.

$$\begin{aligned}
\varphi^{c,n}_{zero}(x) &= \bigwedge_{i=0}^{n-1} \neg c_i(x),\\
\varphi^{c,n}_{succ}(x,y) &= \bigvee_{i=0}^{n-1} \Big(\bigwedge_{j=0}^{i-1} \big(c_j(x) \wedge \neg c_j(y)\big)\\
&\qquad \wedge \neg c_i(x) \wedge c_i(y)\\
&\qquad \wedge \bigwedge_{j=i+1}^{n-1} \big(c_j(x) \leftrightarrow c_j(y)\big)\Big)\\
&\quad \vee \bigwedge_{j=0}^{i-1} \big(c_j(x) \wedge \neg c_j(y)\big)
\end{aligned}$$

Finally, we set φ_n as:

$$\varphi_n = \exists x.\varphi^{c,n}_{zero}(x) \wedge \forall x.\exists y.\varphi^{c,n}_{succ}(x,y).$$

It easy to see that if $\mathfrak{A} \in \mathrm{Str}(\Sigma_n;\emptyset)$ satisfies φ_n, then for any $0 \le k < 2^n$ there is an element in $\mathfrak{A}$ representing k, so $\mathfrak{A}$ has size at least 2^n. To conclude, it suffices to notice that φ_n is of quadratic size. $\square$

Proof of Theorem 2.2.12: We reuse the notation of the proof of Proposition 2.2.13. Let $\mathcal{D} = (D, H, V)$ be a domino system and let n be an integer. Let $\Sigma'_n = \{u_0, u_1, \dots, u_{n-1}, v_0, v_1, \dots, v_{n-1}\}$ and u, v representing respectively the sequences of u_is and v_is. We will build a formula $\varphi_{\mathcal{D},n} \in \mathsf{FO}_{\mathbb{MS}}[\Sigma'_n \cup D;\emptyset]$ and then show that $\mathcal{D}$ admits a tiling of size $(2^n, 2^n)$

iff the formula $\varphi_{\mathcal{D},n}$ is satisfiable. Let φ^{n}_{grid} be

$$\begin{aligned}\varphi^{n}_{grid} &= \exists x.\big(\varphi^{u,n}_{zero}(x) \;\wedge\; \varphi^{v,n}_{zero}(x)\big)\\ &\quad\wedge\; \forall x.\big(\exists y.\varphi^{u,n}_{succ}(x,y) \;\wedge\; \exists y.\varphi^{v,n}_{succ}(x,y)\big)\end{aligned}$$

$$\begin{aligned}\varphi^{\mathcal{D},n}_{cor} &= \forall x. \bigvee_{d\in D}\Bigg(d(x) \;\wedge\; \bigwedge_{\substack{d'\in D\\ d'\neq d}} \neg d'(x)\Bigg)\\ &\quad\wedge\; \forall x.\forall y.\Bigg(\varphi^{u,n}_{succ}(x,y) \to \bigvee_{(d,d')\in H} d(x) \;\wedge\; d'(y)\Bigg)\\ &\quad\wedge\; \forall x.\forall y.\Bigg(\varphi^{v,n}_{succ}(x,y) \to \bigvee_{(d,d')\in V} d(x) \;\wedge\; d'(y)\Bigg).\end{aligned}$$

finally we set $\varphi_{\mathcal{D},n}$ as :

$$\varphi_{\mathcal{D},n} = \varphi^{n}_{grid} \;\wedge\; \varphi^{\mathcal{D},n}_{cor}.$$

Let assume that $\mathcal{D}$ admits a tiling $\pi : \mathfrak{G}_{2^n} \to \mathcal{D}$ of size $(2^n, 2^n)$. We will define $P_\sigma \subseteq G_{2^n}$ for each $\sigma \in \Sigma'_n \cup D$ such that the structure $\mathfrak{A} := (G_{2^n}, (P_\sigma)_{\sigma\in\Sigma'_n\cup D})$ satisfies $\varphi_{\mathcal{D},n}$. We identify G_{2^n} with $\{0,\dots,2^n-1\} \times \{0,\dots,2^n-1\}$. We will see the element of u (resp. v) as bits and then use them to represent the first coordinate i (resp. second coordinate j). So for any $k \in \{0,\dots,2^n-1\}$, we set P_{u_k} such that $(i,j) \in P_{u_k}$ if the kth bit of binary expansion of i is 1. Similarly, we set P_{v_k} such that $(i,j) \in P_{u_k}$ if the kth bit of binary expansion of j is 1. For $d \in D$, we define P_d as :

$$P_d = \{a \in G_{2^n} \mid \pi(a) = d\}.$$

The structure $\mathfrak{A}$ indeed satisfies φ^{n}_{grid} because for any $(i,j) \in \{0,\dots,2^n-1\}\times\{0,\dots,2^n-1\}$ there is an element $a \in G_{2^n}$ such that the interpretations of the predicates us (resp. vs) on a represent i (resp. j), namely $a = (i,j)$. Because π is a morphism $\mathfrak{G}_{2^n} \to \mathcal{D}$, we have $\mathfrak{A} \models \varphi^{\mathcal{D},n}_{cor}$. Put together, we have $\mathfrak{A} \models \varphi_{\mathcal{D},n}$.

Conversely, let us assume that $\varphi_{\mathcal{D},n}$ is satisfiable. We want to build a tiling $\pi : \mathfrak{G}_{2^n} \to \mathcal{D}$ of size $(2^n, 2^n)$ of the domino system $\mathcal{D}$. Let $\mathfrak{A} = (A, (P_\sigma)_{\sigma\in\Sigma'_n\cup D}) \in \mathrm{Str}(\mathbb{MS}; \Sigma'_n \cup D)$ which satisfies $\varphi_{\mathcal{D},n}$. We define a map $g : A \to G_{2^n}$ which associates to any $a \in A$, the unique pair $(i,j) \in \{0,\dots,2^n-1\} \times \{0,\dots,2^n-1\}$ such that the number represented by the interpretation of the us (resp. vs) on a is i (resp j). Because $\mathfrak{A} \models \varphi^{n}_{grid}$, the map g is surjective (and note that g might not be injective). As $\mathfrak{A} \models \varphi^{\mathcal{D},n}_{cor}$, for any $b \in G_{2^n}$ we finally define $\pi(b)$ by first taking $a \in A$ such that $g(a) = b$ and setting $\pi(b)$ to be the only $d \in D$ such that d holds on a. Then π is indeed a morphism because $\mathfrak{A} \models \varphi^{\mathcal{D},n}_{cor}$. □

By carefully examining the proof of Theorem 2.2.12, one can observe that it proves also the the NExp-hardness of FO^2 over multisets.

Theorem 2.2.14

The problem $\mathbb{MS}$-Sat$(\mathsf{FO}^2;\emptyset)$ *is* NExp*-hard.*

Tools for Undecidability We now introduce tools to help us build reductions from the UNBOUNDED-TILING problem to satisfiability problems in order to prove that some of them are undecidable. It is highly inspired from [42]. We first use some specific bi-binary structures, which we call grid-like and which are easier to manipulate in our context to encode domino systems. A bi-binary structure $\mathfrak{G} = (A, H, V)$ is said to be *grid-like* if for some m, the grid $\mathfrak{G}_m$ is homomorphically embeddable into $\mathfrak{G}$. Consider the two following FO formulas over bi-binary structures:

$$\begin{aligned}\varphi_{progress} &= \forall x.(\exists y.\mathsf{H}xy \ \wedge\ \exists y.\mathsf{V}xy),\\ \varphi_{complete} &= \forall x.\forall y.\forall x'.\forall y'.((\mathsf{H}xy \ \wedge\ \mathsf{V}xx' \ \wedge\ \mathsf{V}yy') \rightarrow \mathsf{H}x'y').\end{aligned}$$

We illustrate the two formulas in Figure 2.1.

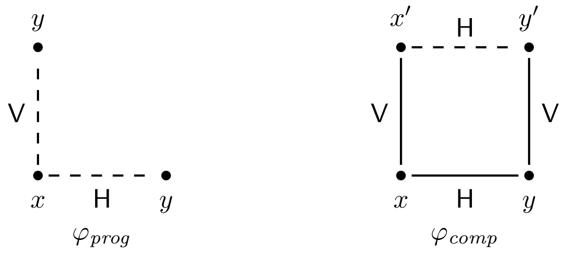


Figure 2.1: Illustration of the formulas $\varphi_{progress}$ and $\varphi_{complete}$. Plain lines mean universal quantification and dashed lines mean existential quantification.

The following lemma, stated and proved in [42], shows that these formulas suffice to characterize grid-like structures:

Lemma 2.2.15 ([42] Grid-like criterion)**.** *Let $\mathfrak{G} = (A, H, V)$ be a bi-binary structure. If $\mathfrak{G}$ satisfies $\varphi_{progress}$ and $\varphi_{complete}$, then $\mathfrak{G}$ is grid-like.*

Given a closed formula $\varphi \in \mathsf{FO}_\Lambda[\Sigma; \mathcal{R}]$, we define $\mathcal{M}(\varphi)$ the class of models of φ as:

$$\mathcal{M}(\varphi) = \{\mathfrak{A} \in \mathrm{Str}(\Sigma; \Lambda) \mid \mathfrak{A} \models \varphi\}.$$

Furthermore given $\mathfrak{A} = (A, (P_\sigma)_\sigma, (R_\gamma)_\gamma, (f_i)_i) \in \mathrm{Str}(\Sigma; \Lambda)$ and $\varphi_1(x, y) \in \mathsf{FO}_\Lambda[\Sigma; \mathcal{R}]$ with two free variables, we define the binary relation $[\![\varphi_1]\!]_{\mathfrak{A}}$ on A as:

$$[\![\varphi_1]\!]_{\mathfrak{A}} = \{(a, b) \in A \times A \mid \mathfrak{A} \models \varphi_1(a, b)\}.$$

Thus, given another formula $\varphi_2(x, y) \in \mathsf{FO}_\Lambda[\Sigma; \mathcal{R}]$ with two free variables, $(A, [\![\varphi_1]\!]_{\mathfrak{A}}, [\![\varphi_2]\!]_{\mathfrak{A}})$ is a bi-binary structure. We define the family of bi-binary structures $\mathcal{F}(\varphi, \varphi_1, \varphi_2)$ as:

$$\mathcal{F}(\varphi, \varphi_1, \varphi_2) = \{(A, [\![\varphi_1]\!]_{\mathfrak{A}}, [\![\varphi_2]\!]_{\mathfrak{A}}) \mid \mathfrak{A} \in \mathcal{M}(\varphi)\}.$$

Let $\mathcal{F}$ be a family of bi-binary structure, we say that $\mathcal{F}$ is a family of grid-like structures if all structures in $\mathcal{F}$ are grid-like. We say that $\mathcal{F}$ is a *rich family* if for any m, there is a k such that $\mathfrak{G}_{k \cdot m} \in \mathcal{F}$.

Lemma 2.2.16 ([42]). *Let $\mathcal{L}$ be a generic class of first-order formulas containing FO^2. Let $\varphi_H(x,y), \varphi_V(x,y), \varphi \in \mathcal{L}_\Lambda[\Sigma; \mathcal{R}]$ with φ a sentence. If $\mathcal{F}(\varphi, \varphi_H, \varphi_V)$ is a rich family of grid-like structures, then the problem Λ-*Sat*$(\mathcal{L}; \mathcal{R})$ is undecidable.*

Proof: We start by describing the outline of the proof. First, for any domino system $\mathcal{D} = (D, H, V)$, we build a sentence $\varphi_\mathcal{D} \in \mathcal{L}_\Lambda[\Sigma \cup D; \mathcal{R}]$. Then we show that $\mathcal{D}$ admits a square periodic tiling if and only if $\varphi_\mathcal{D}$ is satisfiable. Thus from the undecidability of the problem Unbounded-Tiling (Proposition 2.2.11), the lemma is proven.

Let $\mathcal{D} = (D, H, V)$ be a domino system. In order to encode the notion of tiling in logic, we see elements of D as unary predicates. Let us build the sentence $\varphi_\mathcal{D} \in \mathcal{L}_\Lambda[\Sigma \cup D; \mathcal{R}]$. We already have the formula φ which ensures that a structure is grid-like. From this, we now define $\varphi^\mathcal{D}_{cor}$ which ensures the placement of the tiles is correct:

$$\begin{aligned}\varphi^\mathcal{D}_{cor} =& \forall x. \bigvee_{d\in D} \Big(d(x) \wedge \bigwedge_{\substack{d'\in D\\ d'\neq d}} \neg d'(x) \Big) \\ &\wedge \forall x.\forall y. \Big(\varphi_H(x,y) \rightarrow \bigvee_{(d,d')\in H} d(x) \wedge d'(y) \Big) \\ &\wedge \forall x.\forall y. \Big(\varphi_V(x,y) \rightarrow \bigvee_{(d,d')\in V} d(x) \wedge d'(y) \Big).\end{aligned}$$

One can notice that the formula $\varphi^\mathcal{D}_{cor}$ is similar to the formula $\varphi^{\mathcal{D},n}_{cor}$ from the proof of Theorem 2.2.12. Finally, we define

$$\varphi_\mathcal{D} = \varphi \wedge \varphi^\mathcal{D}_{cor}.$$

Let $\mathcal{F}$ denotes the family $\mathcal{F}(\varphi, \varphi_1, \varphi_2)$.

Let assume that $\mathcal{D}$ admits periodic tiling $\pi : \mathfrak{G}_m \to \mathcal{D}$. By assumption, as the family $\mathcal{F}$ is rich, there is a k such that $\mathfrak{G}_{k\cdot m} \in \mathcal{F}$. By definition of $\mathcal{F}$, let $\mathfrak{A} \in \mathcal{M}(\varphi)$ be such that $(A, [\![\varphi_H]\!]_\mathfrak{A}, [\![\varphi_V]\!]_\mathfrak{A}) = \mathfrak{G}_{k\cdot m}$. We now define $\widehat{\mathfrak{A}} \in \mathrm{Str}(\Sigma \cup D; \Lambda)$ as an extension of $\mathfrak{A}$ with interpretation of unary symbols from D such that $\widehat{\mathfrak{A}} \models \varphi^\mathcal{D}_{cor}$. As the carrier set of $\mathfrak{A}$ is $G_{k\cdot m}$, we can for any $d \in D$ set $P^{\widehat{\mathfrak{A}}}_d = \{(i,j) \in \mathbb{Z} \bmod km \times \mathbb{Z} \bmod km \mid \pi((i \mod m, j \mod m)) = d\}$. We indeed have $\widehat{\mathfrak{A}} \models \varphi^\mathcal{D}_{cor}$ and as $\mathfrak{A} \in \mathcal{M}(\varphi)$, we have $\widehat{\mathfrak{A}} \models \varphi$. Put together, we have $\widehat{\mathfrak{A}} \models \varphi_\mathcal{D}$.

Conversely, let assume that $\varphi_\mathcal{D}$ is satisfiable. Let $\mathfrak{A}$ be a model of $\varphi_\mathcal{D}$, then by definition of $\varphi_\mathcal{D}$, the structure $\mathfrak{A}$ satisfies φ. By assumption, as the elements of $\mathcal{F}$ are grid-like, we have that $(A, [\![\varphi_H]\!]_\mathfrak{A}, [\![\varphi_V]\!]_\mathfrak{A})$ is grid like. It means that there is a morphism $\pi : \mathfrak{G}_m \to (A, [\![\varphi_H]\!]_\mathfrak{A}, [\![\varphi_V]\!]_\mathfrak{A})$. As $\mathfrak{A}$ satisfies $\varphi^\mathcal{D}_{cor}$, we can define a function $\rho : A \to D$ which takes $a \in A$ and associates with the unique $d \in D$ such that $a \in P_d$. Thanks to $\varphi^\mathcal{D}_{cor}$, we know that ρ is an morphism from $(A, [\![\varphi_H]\!]_\mathfrak{A}, [\![\varphi_V]\!]_\mathfrak{A})$ to $\mathcal{D}$. Composing the morphisms π and ρ into $\pi \circ \rho$, we get a tiling for $\mathcal{D}$. □

We now apply Lemma 2.2.16 freshly proven in order to obtain our first undecidability result: satisfiability of first order logic with two binary relations is undecidable. In this case, the proof is very simple. The reason is that all the tedious work is done in the proof of Proposition 2.2.11 where one must reduce from Turing machine in some way or another. The simplicity of the proof indicates that the both the tilings problem and Lemma 2.2.16 are well suited to prove undecidability of satisfiability problems.

Theorem 2.2.17

The problem $\mathbb{BB}$-$\textsc{Sat}(\mathsf{FO};\{\mathsf{H},\mathsf{V}\})$ *is undecidable.*

Proof: We apply Lemma 2.2.16. Take:

$$\varphi_H(x,y) = \mathsf{H}(x,y),$$
$$\varphi_V(x,y) = \mathsf{V}(x,y),$$
$$\varphi = \varphi_{progress} \wedge \varphi_{complete}.$$

We have to show that $\mathcal{F}(\varphi,\varphi_H,\varphi_V)$ is rich and that every structure in it is grid-like. We can notice that $\mathcal{F}(\varphi,\varphi_H,\varphi_V) = \mathcal{M}(\varphi)$ and as for any m, we have $\mathfrak{G}_m \models \varphi$, the family $\mathcal{F}$ is indeed rich. Then by definition, any $\mathfrak{A} \in \mathcal{F}(\varphi,\varphi_H,\varphi_V)$ satisfies φ thus by the grid-like criterion Lemma 2.2.15, $\mathfrak{A}$ is grid-like. As the assumptions of Lemma 2.2.16 are checked, the problem $\mathbb{BB}$-$\textsc{Sat}(\mathsf{FO};\{\mathsf{H},\mathsf{V}\})$ is undecidable. $\square$

2.3 The Safari Trip in Data Logics

In this section we will look at various logics on various structures. While some combinations have been studied for a long time and are well known, we will revisit each of them in the light of the satisfiability problems.

2.3.1 Words

The signature of words is $\mathbb{W} = (\{<\}, 0)$. If we instantiate the definition of structures of Section 2.1.1, a word $\mathfrak{A} \in \mathrm{Str}(\Sigma;\mathbb{W})$ is a tuple $(A, (P_\sigma)_{\sigma\in\Sigma}, R_<)$ where A is a finite set, $P_\sigma \subseteq A$ for each $\sigma \in \Sigma$ and $R_< \subseteq A \times A$ with the additional requirement that $R_<$ must be a linear order on A. In this setting, an element carries a set of unary predicates from Σ. While our definition of a word is similar to that of the field of model checking, it contrasts with the definition of automata theory where one often use a set of letters $\mathcal{A}$ and exactly one letter holds on each position. We can go to the automata's formalism by taking $\mathcal{A} = 2^\Sigma$, the powerset of Σ. This difference does not change what can be expressed in the logic nor whether a satisfiability problem is decidable or not but this difference can change the complexity up to an exponential factor in some situations.

$$p \quad q \quad \substack{p\\ q} \quad q \quad \emptyset \quad q$$

Figure 2.2: A word.

Example 2.3.1. Figure 2.2 depicts a word with $\Sigma = \{p,q\}$, $A = \{a_0,a_1,a_2,a_3,a_4,a_5\}$, $P_p^{\mathfrak{A}} = \{a_0,a_2\}$, $P_q^{\mathfrak{A}} = \{a_1,a_2,a_3,a_5\}$ and $<$ interpreted from the indices that is $(a_i,a_j) \in R_<$ iff $i < j$. The elements from left to right go from the smallest to the largest according to $<$ and for each element only the set of predicates holding on it is depicted. $\diamond$

On any word $\mathfrak{A} = (A, (P_\sigma)_{\sigma\in\Sigma}, R_<) \in \mathrm{Str}(\Sigma; \mathbb{W})$, the binary symbol $+1$ can be interpreted as the successor of $<$. Its interpretation R_{+1} is defined as:

$$R_{+1} = \{(a, b) \in A \times A \mid a < b \text{ and there is no } c \in A \text{ such that } a < c \text{ and } c < b\}.$$

We will write $y = x+1$ instead of $+1(x, y)$. The relation $+1$ can be represented in FO with $<$. More precisely, we can define a formula $\varphi_{+1}(x, y) \in \mathsf{FO}_{\mathbb{W}}[\Sigma; \{<\}]$ by

$$\varphi_{+1}(x, y) = x < y \ \wedge\ \neg\exists z. (x < z \ \wedge\ z < y).$$

Then our formula $\varphi_{+1}(x, y)$ satisfies that for any structure $\mathfrak{A}$ and two elements a, b in it, we have $\mathfrak{A} \models b = a + 1$ iff $\mathfrak{A} \models \varphi_{+1}(a, b)$. We observe that φ_{+1} uses only three variables. This implies that for any $k \in \overline{\mathbb{N}}$ with $k \geq 3$, the class of formulas $\mathsf{FO}^k_{\mathbb{W}}[\Sigma; \{<\}]$ is as expressive as $\mathsf{FO}^k_{\mathbb{W}}[\Sigma; \{<, +1\}]$, and the translation from the latter to the former is polynomial. Thus for each $k \geq 3$, the problems $\mathbb{W}$-SAT($\mathsf{FO}^k; \{<\}$) and $\mathbb{W}$-SAT($\mathsf{FO}^k; \{<, +1\}$) are of the same nature and complexity. However, things change in the case of FO^2, as exposed later in this subsection.

We now analyse the problem $\mathbb{W}$-SAT($\mathsf{FO}; \{<\}$). Finite state automata are strictly more expressive than FO on words, but under substructure closure (i.e. allowing some unary predicates to be forgotten), they are equally expressive. For an introduction to this topic, see [49]. This gives a decision procedure for $\mathbb{W}$-SAT($\mathsf{FO}; <$): given a formula φ, construct an automaton $\mathcal{A}$ equivalent to φ. Then φ is satifiable iff the language defined by $\mathcal{A}$ is nonempty. Since there is a potential blowup in the size of the automata, this gives us a TOWER upper bound for the problem $\mathbb{W}$-SAT($\mathsf{FO}; <$). It turns out that this bound is optimal: The first work showing the TOWER hardness is Stockmeyer's thesis [46], he first showed from scratch that checking the emptiness of a star-free regular expression is TOWER-hard and then constructed a reduction from the emptiness of a star-free regular expression to $\mathbb{W}$-SAT($\mathsf{FO}; <$). Then, in [43], Reinhardt gave a direct proof of TOWER-hardness by skipping the intermediate step and constructing from scratch a formula such that the smallest model is of size at least tower. We define the non-elementary function $G : \mathbb{N} \to \mathbb{N}$ as

$$G(n) \longmapsto \begin{cases} 2 \text{ if } n = 0, \\ 2^{G(n-1)} \text{ else.} \end{cases}$$

to help us talk about the class TOWER.

Theorem 2.3.2 (*[46, 43, 49]*)

The problem $\mathbb{W}$-SAT($\mathsf{FO}; <$) *is* TOWER*-complete.*

The usual proof that $\mathbb{W}$-SAT($\mathsf{FO}; <$) is in TOWER is based on a reduction to the emptiness of finite state automata and can be found in [49]. The reduction is inductive on the formula: it translates the negation of a formula into the complementation of the corresponding automaton and the existential quantification as projection. Then translating an alternation of quantifiers leads to an iteration of the powerset construction and leads to a tower blow-up. It turns out that in order to decide the satisfiability, this blow-up is unavoidable. We will devote some time proving hardness, as it is less often shown and may be compared to other proofs of hardness presented in this manuscript. Like many hardness proofs of satisfiability, a lower bound on the small model property is demonstrated using counters which is then used to reduce a tiling problem or the halting problem of Turing machine to it. This proof is noteworthy as it utilises counters in a recursive manner. Generally, in order to demonstrate NExp-hardness, it is necessary to be able to count up to 2^n, as seen in the proof of Theorem 2.3.9. To demonstrate the N2Exp-hardness, the ability to count up to 2^{2^n} is required, as demonstrated in the proof of Theorem 2.3.25. In contrast, in order to show Tower-hardness, counting up to $G(n) = 2^{2^{\cdot^{\cdot^{\cdot^{2^2}}}}}$ is necessary. Counting up to 2^n is not too hard, since it takes n bits to write any number smaller than 2^n in binary, Therefore we use n unary predicates to do it. Then counting up to 2^{2^n} is slightly harder as we have to use n unary predicate to count up to 2^n, and then we use those numbers as positions for digits to count up to 2^{2^n}. Finally, to count up to $2^{2^{\cdot^{\cdot^{\cdot^{2^2}}}}}$, we repeat this technique n times.

Proposition 2.3.3. *There exists a sequence of formulas in* FO *over words of size growing polynomially such that their smallest model grows non-elementary (i.e. is a* $\Omega(G)$*).*

Part of a proof of Proposition 2.3.3: In this proof, a word will be such that at any position, exactly one unary predicate holds. We first recursively build a sequence of set of words growing in Tower using the notion of counter. On the first step (and base case), we have two symbols 0 and 1: we can count from 0 to 1. Then on the second step, we use the two numbers of the first step to number digits which enable us to count in binary 00, 10, 01, 11 (least significant digit first). In order to be able to encode this in FO, we have to annotate the position of a digit with a number, so the sentence become

$$0^0 0^1, 1^0 0^1, 0^0 1^1, 1^0 1^1.$$

We can iterate one more time to get the third step: we use the four numbers to number digits again, so we can count the numbers $0000, 1000, 0100, \dots, 0111, 1111$, so $2^{2^2} = 16$ numbers in total. And again we annotate the position of a digit with a number of the previous step. So we will write

$$\begin{gathered}
0^{0^0 0^1} 0^{1^0 0^1} 0^{0^0 1^1} 0^{1^0 1^1}, \\
1^{0^0 0^1} 0^{1^0 0^1} 0^{0^0 1^1} 0^{1^0 1^1}, \\
0^{0^0 0^1} 1^{1^0 0^1} 0^{0^0 1^1} 0^{1^0 1^1}, \\
\dots \\
0^{0^0 0^1} 1^{1^0 0^1} 1^{0^0 1^1} 1^{1^0 1^1}, \\
1^{0^0 0^1} 1^{1^0 0^1} 1^{0^0 1^1} 1^{1^0 1^1}.
\end{gathered}$$

We now have an intuition on how the recursion in done to be able to count up to tower. Finally, in order order to encode this in words; we flatten the tree of exponents, we index each 0 or 1 with their height before the flattening and we add delimiters \$ at the beginning of each number which are indexed by the height of the number too. So the symbols 0 and 1 of the first step become 0_1 and 1_1 and we can count from $\$_1 0_1$ to $\$_1 1_1$. Next, on the second step we have the numbers:

$$\$_2 0_2 \$_1 0_1 0_2 \$_1 1_1,$$

$$\$_2 1_2 \$_1 0_1 0_2 \$_1 1_1,$$

$$\$_2 0_2 \$_1 0_1 1_2 \$_1 1_1,$$

$$\$_2 1_2 \$_1 0_1 1_2 \$_1 1_1.$$

On the third step, we get the numbers:

$$\$_3 0_3 \$_2 0_2 \$_1 0_1 0_2 \$_1 1_1 0_3 \$_2 1_2 \$_1 0_1 0_2 \$_1 1_1 0_3 \$_2 0_2 \$_1 0_1 1_2 \$_1 1_1 0_3 \$_2 1_2 \$_1 0_1 1_2 \$_1 1_1,$$
$$\$_3 1_3 \$_2 0_2 \$_1 0_1 0_2 \$_1 1_1 0_3 \$_2 1_2 \$_1 0_1 0_2 \$_1 1_1 0_3 \$_2 0_2 \$_1 0_1 1_2 \$_1 1_1 0_3 \$_2 1_2 \$_1 0_1 1_2 \$_1 1_1,$$
$$\$_3 0_3 \$_2 0_2 \$_1 0_1 0_2 \$_1 1_1 1_3 \$_2 1_2 \$_1 0_1 0_2 \$_1 1_1 0_3 \$_2 0_2 \$_1 0_1 1_2 \$_1 1_1 0_3 \$_2 1_2 \$_1 0_1 1_2 \$_1 1_1,$$
$$\ldots$$
$$\$_3 0_3 \$_2 0_2 \$_1 0_1 0_2 \$_1 1_1 1_3 \$_2 1_2 \$_1 0_1 0_2 \$_1 1_1 1_3 \$_2 0_2 \$_1 0_1 1_2 \$_1 1_1 1_3 \$_2 1_2 \$_1 0_1 1_2 \$_1 1_1,$$
$$\$_3 1_3 \$_2 0_2 \$_1 0_1 0_2 \$_1 1_1 1_3 \$_2 1_2 \$_1 0_1 0_2 \$_1 1_1 1_3 \$_2 0_2 \$_1 0_1 1_2 \$_1 1_1 1_3 \$_2 1_2 \$_1 0_1 1_2 \$_1 1_1.$$

We now define formally how to count up to any step. Take the alphabets

$$\Sigma_k = \{\$_k, 0_k, 1_k\}$$

for $1 \leq k \leq n$ and define

$$\Sigma_{<k} = \bigcup_{1 \leq i < k} \Sigma_i,$$

$$\Sigma_{>k} = \bigcup_{k < i \leq n} \Sigma_i.$$

We recall that the non-elementary function $G : \mathbb{N}^* \to \mathbb{N}$ is defined as

$$G(n) \longmapsto \begin{cases} 2 \text{ if } n = 0, \\ 2^{G(n-1)} \text{ else.} \end{cases}$$

For k and $1 \leq i < G(k)$ we define the word $c_{k,i} \in \Sigma_{\leq k}$ starting with

$$c_{1,0} = \$_1 0_1 \text{ and } c_{1,1} = \$_1 1_1,$$

and then

$$c_{2,0} = \$_2 0_2 c_{1,0} 0_2 c_{1,1} \text{ and } c_{2,1} = \$_2 1_2 c_{1,0} 0_2 c_{1,1},$$

$$c_{2,2} = \$_2 0_2 c_{1,0} 1_2 c_{1,1} \text{ and } c_{2,3} = \$_2 1_2 c_{1,0} 1_2 c_{1,1}.$$

We continue to define $c_{k,i}$ by induction on $k \in \{2, \ldots, n\}$. For any i with $0 \leq i < G(k)$, we write it in binary as $x_0 x_1 \ldots x_{G(k-1)-1}$ on the symbols 0_k and 1_k (x_0 being the least significant bit) and we can set $c_{k,i}$ with:

$$c_{k,i} = \$_k x_0 c_{k-1,0} x_1 c_{k-1,1} \dots x_{G(k-1)-1} c_{k-1,G(k-1)-1}.$$

In other words, $c_{k,i}$ is the concatenation of $\$_k$ and the sequence $(x_j c_{k-1,j})_{0 \leq j < G(k-1)}$. We claim that for each k the language

$$L_k = (\Sigma^*_{>k} c_{k,0} \Sigma^*_{>k} c_{k,1} \Sigma^*_{>k} \dots \Sigma^*_{>k} c_{k,G(k)-1} \Sigma^*_{>k})^+ \Sigma^*_{>k}$$

is definable with an $\mathsf{FO}_{\mathbb{W}}[\Sigma_{\leq k}; <]$ formula of size polynomial in n while the smallest word in that language being of size at least $G(k)$. The proof of the definability is in [43]. $\square$

We look now at FO with a restricted number of variable. We can ask the question of the expressiveness first. On signature $\mathbb{W}$, the logic FO^3 is as expressive as FO. It is proved in [28] using Ehrenfeucht-Fraïssé games and then in [22], extending the result in presence of interval preserving binary relations and the proof build an explicit translation from FO to FO^3.

Theorem 2.3.4 (*[28, 22]*)

$\mathsf{FO}^3_{\mathbb{W}}[\Sigma; <]$ is as expressive as $\mathsf{FO}_{\mathbb{W}}[\Sigma; <]$.

However expressiveness do not not tell anything directly. Upon examining the translation from star-free regular expressions to FO sentences over words from [46], it becomes apparent that the resulting formulas have at most three variables, which gives us:

Theorem 2.3.5 (*[46]*)

The problem $\mathbb{W}$-SAT($\mathsf{FO}^3; <$) is TOWER-Complete.

Proof: As FO over words is already in TOWER, we solely have to prove the hardness.

We use the fact that checking emptiness for star-free regular expression is TOWER-complete [46]. A star-free regular expression over Σ is defined by the grammar $E ::= a \mid E + E \mid E \cdot E \mid {}^c E$ where $a \in \Sigma$. The definition of the language represented by a regular expression is as usual, especially at any position of any word in them, there is exactly one unary predicate holding on it. From a star-free regular expression E we inductively associate a formula $\varphi_E(x, y)$ in $\mathsf{FO}^3_{\mathbb{W}}[\Sigma; <]$. The intended meaning of $\varphi_E(x, y)$ is that a word w will satisfy $\varphi_E(x, y)$ if the subword from x included to y excluded is in the language defined by E. The definition of $\varphi_E(x, y)$ is

$$\begin{aligned}
\varphi_a(x,y) &= a(x) \wedge y = x + 1 \text{ for } a \in \Sigma \\
\varphi_{E+E'}(x,y) &= \varphi_E(x,y) \vee \varphi_{E'}(x,y) \\
\varphi_{E \cdot E'}(x,y) &= \exists z. x \leq z \wedge z \leq y \wedge \varphi_E(x,z) \wedge \varphi_{E'}(z,y) \\
\varphi_{{}^c E}(x,y) &= \neg \varphi_E(x,y)
\end{aligned}$$

We can notice that as $\varphi_E(x, y)$ always has at most three free variables, it is indeed part of

FO^3. The final transformation associates E with:

$$\Big(\exists xy.\mathsf{FST}(x) \wedge \mathsf{LST}(y) \wedge \varphi_E(x,y)\Big) \wedge \forall x. \bigvee_{\sigma\in\Sigma} \Big(\sigma(x) \wedge \bigwedge_{\substack{\sigma'\in\Sigma\\ \sigma'\neq\sigma}} \neg\sigma'(x)\Big),$$

where $\mathsf{FST}(x) = \neg\exists z.z < x$ and $\mathsf{LST}(y) = \neg\exists z.y < z$. The regular expression E and its translation both define the same language and as checking emptiness for star-free regular expression is TOWER-hard, this implies that the satisfiability for FO^3 over words is TOWER-hard. □

Remark 2.3.6. With the transformation of the proof above, we can deduce that over words, FO^3 is as expressive as FO. Indeed, let $\varphi \in \mathsf{FO}_{\mathbb{W}}[\Sigma;<]$. It is known that there exists a star-free regular expression E equivalent to φ. Then by applying the transformation of the proof above to E, we obtain a formula in $\mathsf{FO}^3_{\mathbb{W}}[\Sigma;<]$ equivalent to φ. ◇

We look at the logic $\mathsf{FO}_{\mathbb{W}}[\Sigma;\{+1\}]$. The problem $\mathbb{W}$-SAT$(\mathsf{FO};\{+1\})$ is decidable since $\mathsf{FO}_{\mathbb{W}}[\Sigma;\{+1\}]$ is contained in $\mathsf{FO}_{\mathbb{W}}[\Sigma;\{<,+1\}]$, which we know to be decidable. But this does not enable us to pin down the complexity as the former logic is strictly less expressive than the latter. We solely know that the logic $\mathsf{FO}_{\mathbb{W}}[\Sigma;\{+1\}]$ defines "locally treshold testable languages" as described and proven in [48]. In this way, we know that the complexity of $\mathbb{W}$-SAT$(\mathsf{FO};\{+1\})$ is at most TOWER, but it seems that its precise complexity is unknown.

On the other hand, FO^2 is strictly less expressive on words than FO^3 and the complexity of the satisfiability problem of the former drops by a lot compared to the one of the latter. Although we have to be careful because $\mathsf{FO}^2_{\mathbb{W}}[\Sigma;<]$ is less expressive than $\mathsf{FO}^2_{\mathbb{W}}[\Sigma;<,+1]$.

Theorem 2.3.7 (*[19]*)

$\mathbb{W}$-SAT$(\mathsf{FO}^2;<)$ and $\mathbb{W}$-SAT$(\mathsf{FO}^2;<,+1)$ are NEXP-complete.

Theorem 2.3.7 is proven in [19] by establishing a correspondence between FO^2 over words and a fragment of LTL and then establishing a small small model property for this fragment of LTL. The hardness is entailed by the hardness of the satisfiability problem of FO^2 over multisets (Theorem 2.3.9).

2.3.2 Multisets

The multisets, designated with the signature $\mathbb{MS} = (\emptyset, 0)$, are the simplest structures we can consider in our setting. A multiset $\mathfrak{A} \in \mathrm{Str}(\Sigma;\mathbb{MS})$ is a tuple $(A,(P_\sigma)_{\sigma\in\Sigma})$ where A is a finite set and the P_σs are subsets of A. That is, a set of element with labelling and without any relations between the elements. Any two elements of A having the same set of unary predicates holding on them are then indistinguishable, justifying calling them multisets.

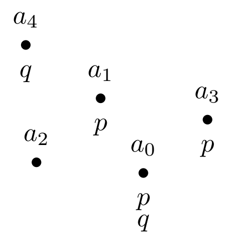


Figure 2.3: A multiset.

Example 2.3.8. Figure 2.3 depicts a multiset with $\Sigma = \{p,q\}$, $A = \{a_0,a_1,a_2,a_3,a_4\}$, $P_p = \{a_0,a_1,a_3\}$, $P_q = \{a_0,a_4\}$. We depict an element by a dot and we placed below it all the predicates holding on it. ◇

Yet the satisfiability problem for FO on $\mathbb{MS}$ is already NExp-complete. Restricting the logic is of no effect to tame the complexity as the satisfiability problem of FO^2 on multisets remains NExp-hard.

Theorem 2.3.9 (*[12, 19]*)

The problems $\mathbb{MS}$-SAT$(\mathsf{FO};\emptyset)$ *and* $\mathbb{MS}$-SAT$(\mathsf{FO}^2;\emptyset)$ *are* NExp*-complete.*

We already prove this theorem in the previous section as Theorems 2.2.8 and 2.2.14.

For external references, the proof of decidability can be found in [12] at Chapter 6.2.1. while the proof of hardness is described in [19]. Both their proofs are similar to ours, but ours are more detailed.

A Normal Form Furthermore than being satisfiable, FO on multisets actually has a useful normal form. Let $\varphi(x_1, \dots, x_n, y) \in \mathsf{FO}_{\mathbb{MS}}[\Sigma;\emptyset]$ and $k \geq 1$ be a natural number. We use $\exists^{\geq k} y.\varphi(x_1, \dots, x_n, y)$ as an abbreviation for:

$$\exists y_1 \dots \exists y_k. \bigwedge_{1 \leq i < j \leq k} \neg(y_i = y_j) \wedge \bigwedge_{1 \leq i \leq k} \varphi(x_1, \dots, x_n, y_i).$$

Thus, $\exists^{\geq k} y.\varphi$ says that there are at least k distinct elements y that verify φ. We call a formula of the form $\exists^{\geq k} y.\varphi$ a *threshold formula*. We also use $\exists^{= k} y.\varphi$ as an abbreviation for $\exists^{\geq k} y.\varphi \wedge \neg\exists^{\geq k+1} y.\varphi$.

When $\mathcal{R} = \emptyset$, the out-degree of every element is 0 so that, over this particular signature, we deal with structures of bounded degree. The following lemma will turn out to be useful. It is due to Hanf's locality theorem [27, 37] for structures of bounded degree (cf. [9, Theorem 2.4]).

Lemma 2.3.10. *Every formula φ from $\mathsf{FO}_{\mathbb{MS}}[\Sigma;\emptyset]$ is effectively equivalent to a Boolean combination of formulas of the form $\sigma(x)$ and $x = y$ with $\sigma \in \Sigma, x, y \in FV(\varphi)$ and threshold formulas of the form $\exists^{\geq k} y.\varphi_U(y)$ where $U \subseteq \Sigma$ and $\varphi_U(y) = \bigwedge_{\sigma \in U} \sigma(y) \wedge \bigwedge_{\sigma \in \Sigma \setminus U} \neg\sigma(y)$.*

An interesting follow-up question to the previous lemma is to find the exact maximum size of the translation. A quick analysis gives us a double exponential upper bound, but then nothing is known about more precise bounds.

2.3.3 Data-Words

The κ-data-words are represented with the signature $\kappa\mathbb{DW} = (\{<\}, \kappa)$ with $\kappa > 0$. When $\kappa = 1$, we simply call them data-words and use the signature $\mathbb{DW} = (\{<\}, 1)$. Unfolding the definition of structures of Section 2.1.1, a data-word $\mathfrak{A} \in \mathrm{Str}(\Sigma;\mathbb{DW})$ is a tuple $(A, (P_\sigma)_{\sigma\in\Sigma}, R_<, f)$ where A is a finite set, $P_\sigma \subseteq A$ for each $\sigma \in \Sigma$, $R_< \subseteq A \times A$ with the additional requirement that $R_<$ must be a linear order on A and f is a function $A \to \mathbb{N}$ representing data values.

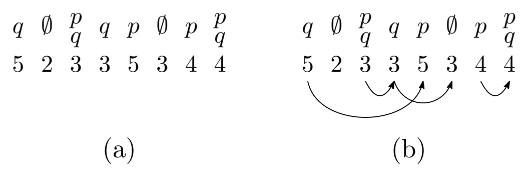


Figure 2.4: A data-word.

Example 2.3.11. Figure 2.4 (a) depicts a data word with $\Sigma = \{p, q\}$, $A = \{a_i \mid 0 \leq i < 8\}$, $P_p = \{a_2, a_4, a_6, a_7\}$, $P_q = \{a_0, a_2, a_3, a_7\}$ and $<$ is interpreted as in Example 2.3.1. Finally the value of f is depict below an element, for example $f(a_0) = 5, f(a_1) = 2$ and $f(a_2) = f(a_3) = 3$. ◇

We recall that on data-words, we can interpret the relations $\overset{\sim}{+}1$ called class successor allowing us to jump from a position to the next one with the same data value. Formally, we interpret it as:

$$R_{\overset{\sim}{+}1} = \{(a, b) \in A \times A \mid a < b \text{ and } f(a) = f(b) \text{ and}$$
$$\text{there is no } c \in A \text{ such that } a < c \text{ and } c < b \text{ and } f(a) = f(c)\}$$

Example 2.3.12. In Figure 2.4 (b) depicts the interpretation of the class successor of the word of Figure 2.4 (a). We have $R^{\mathfrak{A}}_{\overset{\sim}{+}1} = \{(a_0, a_4), (a_2, a_3), (a_3, a_5), (a_6, a_7)\}$. ◇

Remark 2.3.13. Similarly to $+1$, the binary relation $\overset{\sim}{+}1$ can be redefined from $<$ and $\sim$ with only three variables. That is, there is a formula $\varphi_{\overset{\sim}{+}1}(x, y) \in \mathsf{FO}^3_{\mathbb{DW}}[\emptyset; \{<, \sim\}]$ which simulate $\overset{\sim}{+}1$. The formula is given by:

$$\varphi_{\overset{\sim}{+}1}(x, y) = x < y \wedge x \sim y \wedge \neg\exists z.(x < z \wedge z < y \wedge x \sim z).$$

On the contrary, the binary relation $\overset{\sim}{+}1$ can not be defined with two variables, so $\mathsf{FO}^2_{\mathbb{DW}}[\Sigma; \{<, \sim\}]$ is strictly less expressive than $\mathsf{FO}^2_{\mathbb{DW}}[\Sigma; \{<, \sim, \overset{\sim}{+}1\}]$. ◇

In order to describe the complexity of the satisfiability of logics over data-words, we need to introduce the Ackermannnian complexity class, denoted Ack. Vaguely defined, the class Ack is the set of problems decidable in Ackermannnian time. For a more precise definition, see [44] where it corresponds to the class $\mathbb{F}_\omega$.

Let us start to tell the results on the decidability of FO over data-words. Even with one data value and three variable, it is undecidable.

Theorem 2.3.14 (*[7]*)

$\mathbb{DW}$-Sat$(\mathsf{FO}^3; \{\sim, <\})$ *is undecidable.*

With two data values, we start to have positive results, although the precise complexity change whether we add $+1$ and $\overset{\sim}{+}1$ or not. In [7], the authors found a N2Exp reduction from the problem $\mathbb{DW}$-Sat$(\mathsf{FO}^2; \{\sim, <, +1\})$ to the emptiness of multicounter automata, and a PTime one for the other way around. As Leroux in [36] shown that reachability for Petri nets is Ack-complete, it implies that $\mathbb{DW}$-Sat(FO^2; $\{\sim, <, +1\}$) is Ack-complete. Then they extended this method to $\mathbb{DW}$-Sat(FO^2; $\{\sim, +1, \mp 1\}$).

Theorem 2.3.15 (*[7]*)

$\mathbb{DW}$-Sat(FO^2; $\{\sim, <, +1\}$) *and* $\mathbb{DW}$-Sat(FO^2; $\{\sim, +1, \mp 1\}$) *are* Ack*-complete.*

Still from [7], it is possible to reduce $\mathbb{DW}$-Sat(FO^2; $\{\sim, <\}$) to $\mathbb{W}$-Sat(FO^2; $\{<\}$). As the latter problem is NExp (Theorem 2.3.7), this gives us:

Theorem 2.3.16 (*[7]*)

$\mathbb{DW}$-Sat(FO^2; $\{\sim, <\}$) *is* NExp*-complete.*

If one allows $+1$ but not $<$, we get:

Theorem 2.3.17 (*[6]*)

$\mathbb{DW}$-Sat(FO^2; $\{\sim, +1\}$) *is in* N2Exp*.*

Note that we do not have a matching lower bound for the last theorem.

To end on data-words, we look on what happens when we add more data values. Sadly, over 2-data-word and with two data values, the logic is already undecidable.

Theorem 2.3.18 (*[7]*)

$2\mathbb{DW}$-Sat(FO^2; $\{{}_1{\sim}_1, {}_2{\sim}_2, <\}$) *and* $2\mathbb{DW}$-Sat(FO^2; $\{{}_1{\sim}_1, {}_2{\sim}_2, +1, +2, +3\}$) *are undecidable.*

Yet, the author of [30] managed to defined a sub logic of FO^2 based on LTL (Linear-time Temporal Logic) in order to overcome the undecidability on data-words with multiple data values per element. We will not investigate more as LTL is outside the scope of this work and the definition in [30] of multiple data values differs from ours.

2.3.4 Graphs

In this subsection, we investigate the case of the graphs. Although graphs are not structures with data, FO on graphs is fundamental, as it is undecidable and many other undecidability results stem from it. Particularly, we will use it in the next subsection about κ-data-multisets.

The class of graphs are represented with the signature $\mathbb{G} = (\{E\}, 0)$, where E is a binary relation. In our setting, unfolding the definition, a graph $\mathcal{G} \in \mathrm{Str}(\Sigma; \mathbb{G})$ is a tuple $(V, (P_\sigma)_{\sigma\in\Sigma}, R_E)$ where V is a finite set, the P_σs are subsets of A and $R_E \subseteq V \times V$. Compared to the usual terminology in computer science, $\mathcal{G}$ would be called an oriented graph with self loop and labeled vertices. Furthermore the set V would be called the set of vertices of $\mathcal{G}$ and for $u, v \in V$, we would say that the vertex u is connected to v if $(u, v) \in R_E$.

We first present the result of plain FO over graphs, called Trakhtenbrot's Theorem in honor to the mathematician who first discovered this result. It is an old result and one of the first about finite satisfiability.

Theorem 2.3.19 (*Trakhtenbrot (1950), [50, 16, 37]*)

The problem $\mathbb{G}$-SAT(FO; $\{E\}$) is undecidable.

The Theorem still holds with the requirement that $\Sigma = \emptyset$. [50] proves the result by a reduction from recursive functions. The books [16, Theorem 7.2.1] and [37, Theorem 9.2] have a more modern presentation and they start from the halting problem for Turing machines but are incomplete. The proof of the book [16] is more extensive and contains the proof that for any Γ, the problem $(\Gamma, 0)$-SAT(FO; Γ) is reducible to $\mathbb{G}$-SAT(FO; $\{E\}$), which is the first step to prove Theorem 2.3.19.

When restricting the number of variables down to three, we get:

Theorem 2.3.20 (*[41]*)

The problem $\mathbb{G}$-SAT(FO^3; $\{E\}$) is undecidable.

The theorem still holds with the requirement that $\Sigma = \emptyset$ too and without using the equality symbol. The proof is found in [41] which is intricate and involves a lot of steps and technicalities. However, if we solely aim at proving that $\mathbb{G}$-SAT(FO^3; $\{E\}$) is undecidable, we have a shorter and easier proof thanks to our previous work on domino problems.

Proof of Theorem 2.3.20: The proof is done by applying Lemma 2.2.16 which is a tool to create a reduction from UNBOUNDED-TILING. Thus, we want to find a way to encode grids into graphs. To this end, we build a graph $\mathcal{G}_{2m}$ that corresponds to the grid $\mathfrak{G}_{2m}$. This graph is depicted locally in Figure 2.5.

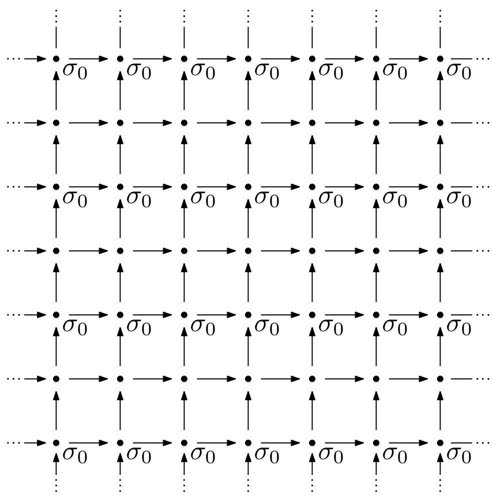


Figure 2.5: The local pattern of $\mathcal{G}_{2m}$.

To define $\mathcal{G}_{2m}$, we use one unary predicate σ_0, which tell us on which row we are modulo 2. We then define $\mathcal{G}_{2m} = (V^{\mathcal{G}_{2m}}, P_{\sigma_0}^{\mathcal{G}_{2m}}, R_E^{\mathcal{G}_{2m}}) \in \text{Str}(\{\sigma_0\}; \mathbb{G})$ as follows:

- $V^{\mathcal{G}_{2m}} = \mathbb{Z} \bmod 2m \times \mathbb{Z} \bmod 2m$,
- $P_{\sigma_0}^{\mathcal{G}_{2m}} = \{(i,j) \in V^{\mathcal{G}_{2m}} \mid j \equiv 0 \mod 2\}$,

- $R_E^{\mathcal{G}_{2m}} = \{((i,j),(i',j)) \in V^{\mathcal{G}_{2m}} \times V^{\mathcal{G}_{2m}} \mid i'-i \equiv 1 \mod 2m \text{ or } j'-j \equiv 1 \mod 2m\}$.

We define the quantifier free formulas $\varphi_H(x,y)$ and $\varphi_V(x,y)$ from the logic $\mathsf{FO}^3_{\mathbb{G}}[\{\sigma_0\};\{E\}]$ with two free variable as ($\oplus$ denotes exclusive or):

$$\varphi_H(x,y) := E(x,y) \wedge (\sigma_0(x) \leftrightarrow \sigma_0(y)),$$
$$\varphi_V(x,y) := E(x,y) \wedge (\sigma_0(x) \oplus \sigma_0(y)).$$

These formulas allow us to make the link between the graph $\mathcal{G}_{2m}$ and the grid $\mathfrak{G}_{2m}$, and we will use them later on to ensure that a graph has a shape 'similar' to a grid.

Remark 2.3.21. We correctly define $\mathcal{G}_{2m}, \varphi_H$ and φ_V as one can observe that the bi-binary structure $(V_{2m}, [\![\varphi_H]\!]_{\mathcal{G}_{2m}}, [\![\varphi_V]\!]_{\mathcal{G}_{2m}})$ is exactly $\mathfrak{G}_{2m}$. $\diamond$

We recall that if $\varphi \in \mathsf{FO}^3_{\mathbb{G}}[\{\sigma_0\};\{E\}]$ is a sentence, the family $\mathcal{F}(\varphi,\varphi_H,\varphi_V)$ of bi-binary structures defined as:

$$\mathcal{F}(\varphi,\varphi_H,\varphi_V) = \{(V^{\mathcal{G}}, [\![\varphi_1]\!]_{\mathfrak{A}}, [\![\varphi_2]\!]_{\mathfrak{A}}) \mid \mathcal{G} \in \mathrm{Str}(\{\sigma_0\};\mathbb{G}), \mathcal{G} \models \varphi\}.$$

Then to complete the proof, we aim to apply Lemma 2.2.16. To this end, we would like to find a sentence $\varphi \in \mathsf{FO}^3_{\mathbb{G}}[\{\sigma_0\};\{E\}]$ such that the family $\mathcal{F}(\varphi,\varphi_H,\varphi_V)$ is rich and that every structure in it is grid-like. One candidate for φ could be the conjunction of $\varphi^{graphs}_{progress}$ and $\varphi^{graphs}_{complete} \in \mathsf{FO}_{\mathbb{G}}[\{\sigma_0\};\{E\}]$ defined as:

$$\varphi^{graphs}_{progress} := \forall x.\big(\exists y.\varphi_H(x,y) \wedge \exists y.\varphi_V(x,y)\big),$$
$$\varphi^{graphs}_{complete} := \forall x.\forall y.\forall x'.\forall y'.\Big(\big(\varphi_H(x,y) \wedge \varphi_V(x,x') \wedge \varphi_V(y,y')\big) \to \varphi_H(x',y')\Big).$$

This would work excepts for the fact that the formula $\varphi^{graphs}_{complete}$ uses four variables. That's not an issue as we can substitute it with the equivalent formula $\varphi^{graphs}_{complete.bis} \in \mathsf{FO}^3_{\mathbb{G}}[\{\sigma_0\};\{E\}]$ defined as:

$$\varphi^{graphs}_{complete.bis} := \forall x.\forall y.\Big(\exists z.\Big(\varphi_V(z,x) \wedge \exists x.\big(\varphi_H(z,x) \wedge \varphi_V(x,y)\big)\Big) \to \varphi_H(x,y)\Big)$$

Then we define φ as:

$$\varphi := \varphi^{graphs}_{progress} \wedge \varphi^{graphs}_{complete.bis}.$$

The formula φ is indeed part of $\mathsf{FO}^3_{\mathbb{G}}[\{\sigma_0\};\{E\}]$.

We prove now that the family $\mathcal{F}(\varphi,\varphi_H,\varphi_V)$ is rich. let m be an integer. We have that $\mathcal{G}_{2m} \models \varphi$ and that the bi-binary structure $(V^{\mathcal{G}_{2m}}, [\![\varphi_H]\!]_{\mathcal{G}_{2m}}, [\![\varphi_V]\!]_{\mathcal{G}_{2m}})$ is the same as the grid $\mathfrak{G}_{2m}$. So the family is indeed rich. We prove now that that every structure in the family is grid-like. Let $\mathfrak{A}$ be a bi-binary structure in $\mathcal{F}(\varphi,\varphi_H,\varphi_V)$. Thanks to φ, we have that $\mathfrak{A}$ satisfies $\varphi_{progress}$ and $\varphi_{complete}$, so we can apply Lemma 2.2.15 which tells us that $\mathfrak{A}$ is grid-like. By application of Lemma 2.2.16, the problem $\mathbb{G}$-SAT(FO; $\{E\}$) is undecidable. $\square$

If we drop the number of variables to 2, we get a positive result. In fact we still have decidability for an arbitrary set of binary predicates Γ. It comes from the fact that the satisfiability problem of FO^2 on any relational structure is decidable. While [26] were not the first one to show the decidability, they were the first to pinpoint the exact complexity.

Theorem 2.3.22 (*[26]*)

The problem $\mathbb{G}$-$\textsc{Sat}(\mathsf{FO}^2; \{E\})$ is NExp*-complete. More generally, for any Γ, the problem $(\Gamma, 0)$-$\textsc{Sat}(\mathsf{FO}^2; \Gamma)$ is* NExp*-complete.*

2.3.5 Data-multisets

The κ-data-multisets are represented by the signature $\kappa\mathbb{DMS} = (\emptyset, \kappa)$ where $\kappa > 0$. A κ-data-multiset $\mathfrak{A} \in \mathrm{Str}(\Sigma; \kappa\mathbb{DMS})$ is a tuple $(A, (P_\sigma)_{\sigma\in\Sigma}, f_1, f_2, \ldots, f_\kappa)$ where A is a finite set, the P_σs are subsets of A and $f_1, f_2, \ldots, f_\kappa$ are functions $A \to \mathbb{N}$ representing data values. We will sometimes simply refer to a data-multiset as a *data structure*.

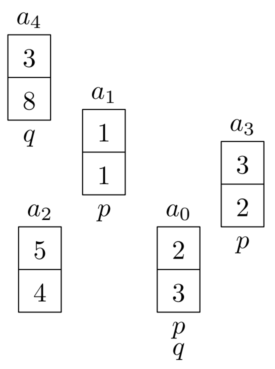


Figure 2.6: A 2-data-multiset.

Example 2.3.23. Figure 2.6 depicts a 2-data-set with $A = \{a_0, a_1, a_2, a_3, a_4\}$, $P_p = \{a_0, a_1, a_3\}$, $P_q = \{a_0, a_4\}$. Each element is represented by a domino with the first data value on top and the second at the bottom. For example, $f_1(a_0) = 2$ and $f_2(a_2) = 4$. ◇

On κ-data-multisets, we can interpret the relations ${}_i{\sim}_j$ allowing to compare data values and we have defined $\mathcal{A}_\kappa = \{{}_i{\sim}_j \mid 1 \leq i, j \leq \kappa\}$ and $\mathcal{S}_\kappa = \{{}_i{\sim}_i \mid 1 \leq i \leq \kappa\}$. We will write $x\,{}_i{\sim}_j\,y$ instead of ${}_i{\sim}_j(x, y)$.

Example 2.3.24. For the 2-data-multiset $\mathfrak{A}$ pictured in Figure 2.6, we have:

$$\begin{aligned}
R^{\mathfrak{A}}_{{}_1\sim_1} &= \{(a_3, a_4), (a_4, a_3)\},\\
R^{\mathfrak{A}}_{{}_2\sim_2} &= \emptyset,\\
R^{\mathfrak{A}}_{{}_1\sim_2} &= \{(a_0, a_3), (a_1, a_1), (a_3, a_0), (a_4, a_0)\},\\
R^{\mathfrak{A}}_{{}_2\sim_1} &= \{(a_0, a_3), (a_0, a_4), (a_1, a_1), (a_3, a_0)\}.
\end{aligned}$$

◇

One can notice that the relation ${}_i{\sim}_i$ are always interpreted as equivalence relation and that some converse holds. It implies that FO with set of binary symbols $\mathcal{S}_\kappa$ over $\kappa\mathbb{DMS}$ is the same as FO over sets equipped with κ equivalence relations.

We shall now begin to expose the results on κ-data-multisets. We start with one data value per element. The structures are the same as a set equipped with one equivalence relation. Plain FO is N2Exp-complete, as well as FO^3.

Theorem 2.3.25 (*[40]*)

The problems $1\mathbb{DMS}$-$\textsc{Sat}(\mathsf{FO}; \{\sim\})$ and $1\mathbb{DMS}$-$\textsc{Sat}(\mathsf{FO}^3; \{\sim\})$ are N2Exp*-complete.*

The proof in [40] is interesting and kind of iterate the technique of the proof of the NEXP-completeness of FO over multisets (Theorem 2.3.9), both for the lower and upper bound.

Proof: In [40], the authors studied the satisfiability problem for Hybrid logic over Kripke structures where the transition relation is an equivalence relation, and they showed that it is N2EXP-complete. Furthermore in [20], it is shown that Hybrid logic can be translated to first-order logic in polynomial time and this holds as well -for the converse translation. Since 1-data-multisets can be interpreted as Kripke structures with one equivalence relation, altogether this allows us to obtain the following preliminary result about the satisfiability of $1\mathbb{DMS}\text{-}\textsc{Sat}(\mathsf{FO};\{\sim\})$. Moreover, if we look more closely at the translation from Hybrid logic, we can notice that the resulting formulas use only three variables, which get us the result about the satisfiability of $1\mathbb{DMS}\text{-}\textsc{Sat}(\mathsf{FO}^3;\{\sim\})$. □

If we reduce the number of variable to two, we lost one level of exponential and get:

Theorem 2.3.26 (*[33]*)

The problem $1\mathbb{DMS}\text{-}\textsc{Sat}(\mathsf{FO}^2;\{\sim\})$ *is* NEXP-*complete.*

Note that the previous theorem is not an implication of Theorem 2.3.22 because the relation $\sim$ cannot be axiomatized in FO^2.

Next, we look at the case of two data values per element. The first result is that plain FO with two equivalence relations is undecidable.

Theorem 2.3.27 (*[29]*)

The problem $2\mathbb{DMS}\text{-}\textsc{Sat}(\mathsf{FO};\mathcal{S}_2)$ *is undecidable.*

Theorem 2.3.27 is frequently referenced in the literature but its proof is hard to locate. In [29], the theorem is claimed to be proven, but there are two issues. Firstly, it is proven for the version of the satisfiability problem that allows infinite models. Secondly, the article is written in an old formalism and the author does not give all the details which make it hard for a present reader to follow. That is why we provide a proof here. Another advantage in re-proving this theorem is that we are able to adapt the proof to FO^3 and give what seems to be a nonexistent result in the literature.

Proof of Theorem 2.3.27: The starting point is the problem $\mathbb{G}\text{-}\textsc{Sat}(\mathsf{FO};\{E\})$, the satisfiability of FO over graphs, known to be undecidable (cf Theorem 2.3.19).

Let us fix Σ and let s be a fresh unary predicate and $\Sigma' = \Sigma \uplus \{\mathsf{s}\}$. From a graph $\mathcal{G} = (V, (P_\sigma)_{\sigma\in\Sigma}, R_E) \in \mathrm{Str}(\Sigma;\mathbb{G})$, we associate a 2-data-multiset $T(\mathcal{G}) = (A, (P'_\sigma)_\sigma, f_1, f_2) \in \mathrm{Str}(\Sigma'; 2\mathbb{DMS})$. See Figure 2.7 for an example. Formally, the structure $T(\mathcal{G})$ is such that:

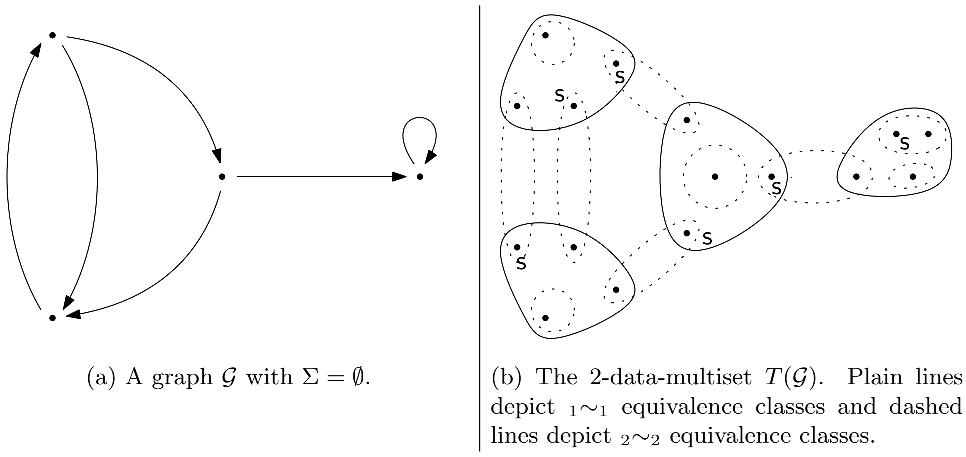
- $A = V \uplus (R_E \times \{0,1\})$,
- for $\sigma \in \Sigma$, $P'_\sigma = P_\sigma$,
- $P'_{\mathsf{s}} = \{((u,v),0) \mid (u,v) \in R_E\}$,

- let ξ be an injection $V \to \mathbb{N}$, we then require for $a \in A$:

$$f_1(a) = \begin{cases} \xi(u) \text{ if } a = u \in V \\ \xi(u) \text{ if } a = ((u,v),0) \in R_E \times \{0,1\} \\ \xi(v) \text{ if } a = ((u,v),1) \in R_E \times \{0,1\}, \end{cases}$$

- let ζ be an injection $R_E \to \mathbb{N} \setminus \xi(V)$, we then require for $a \in A$:

$$f_2(a) = \begin{cases} \xi(v) \text{ if } a = v \in V \\ \zeta((u,v)) \text{ if } a = ((u,v),\_) \in R_E \times \{0,1\}. \end{cases}$$



(a) A graph $\mathcal{G}$ with $\Sigma = \emptyset$.

(b) The 2-data-multiset $T(\mathcal{G})$. Plain lines depict ${}_1{\sim}_1$ equivalence classes and dashed lines depict ${}_2{\sim}_2$ equivalence classes.

Figure 2.7: An example of the transformation of a graph. Each edge from $\mathcal{G}$ is translated into a pair of elements, one for the start of the edge and the other for the end. The start is labeled with the predicate s and the pair form a ${}_2{\sim}_2$-class. The ${}_1{\sim}_1$-classes are constituted of a vertex and all the edges' extremities touching it.

We define two formulas $\varphi_{vertex}(x)$ and $\varphi_{edge}(x,y)$ of $\mathsf{FO}_{2\mathbb{DMS}}[\Sigma'; \mathcal{S}_2]$ with respectively one and two free variables. The formulas help us to retrieve the structure of $\mathcal{G}$ within $T(\mathcal{G})$. The formula $\varphi_{vertex}(x)$ tells us if the element at x corresponds to a vertex and the formula $\varphi_{edge}(x,y)$ tells us if the two vertices x and y are connected in the origin graph.

$$\varphi_{vertex}(x) := \neg\exists y. x \,{}_2{\sim}_2\, y \ \wedge\ x \neq y,$$
$$\varphi_{edge}(x,y) := \exists z. \exists t. x \,{}_1{\sim}_1\, z \ \wedge\ \mathsf{s}(z) \ \wedge\ z \,{}_2{\sim}_2\, t \ \wedge\ t \,{}_1{\sim}_1\, y.$$

Formally, they satisfy:

(i) For all $a \in A$, we have $a \in V$ iff $\mathfrak{A} \models \varphi_{vertex}(a)$,

(ii) For all $u, v \in V$, we have $(u,v) \in R_E$ iff $\mathfrak{A} \models \varphi_{edge}(u,v)$.

We now extend the transformation T on the logic in order to transform any formula of $\mathsf{FO}_{\mathbb{G}}[\Sigma; \{E\}]$ into a formula of $\mathsf{FO}_{2\mathbb{DMS}}[\Sigma'; \mathcal{S}_2]$ by induction:

$$\begin{aligned} T(\sigma(x)) &= \sigma(x), & T(\varphi \vee \varphi') &= T(\varphi) \vee T(\varphi'),\\ T(E(x,y)) &= \varphi_{edge}(x,y), & T(\neg\varphi) &= \neg T(\varphi),\\ T(x = y) &= x = y, & T(\exists x.\varphi) &= \exists x.(\varphi_{vertex}(x) \wedge \varphi). \end{aligned}$$

We correctly constructed T as it "pushes forward" the satisfiability:

Lemma 2.3.28. *Let $\mathcal{G} \in \mathrm{Str}(\Sigma; \mathbb{G})$ and $\varphi \in \mathsf{FO}_{\mathbb{G}}[\Sigma; \{E\}]$. If $\mathcal{G} \models \varphi$, then $T(\mathcal{G}) \models T(\varphi)$.*

The Lemma is in fact an equivalence, but for our purpose, an implication suffices.

Proof: The proof is done by fixing $\mathcal{G} = (V, (P_\sigma)_{\sigma\in\Sigma}, R_E) \in \mathrm{Str}(\Sigma; \mathbb{G})$ and then doing an induction on $\varphi(\vec{x})$ with the following induction hypothesis: "For any interpretation function I with values in V, we have $\mathcal{G} \models \varphi$ iff $T(\mathcal{G}) \models T(\varphi)$". □

Previous Lemma indicates that the satisfiability of φ implies the satisfiability of $T(\varphi)$. For the the converse, we have to logically constrain the 2-data-multisets. To this end we define $\varphi_1, \varphi_2, \varphi_3$ and φ_{cons} elements of $\mathsf{FO}_{2\mathbb{DMS}}[\Sigma'; \mathcal{S}_2]$. The formulas φ_1 and φ_2 constrain the ${}_1{\sim}_1$-classes so that for every ${}_1{\sim}_1$-class there is exactly one element satisfying φ_{vertex}. The formula φ_3 constrains the ${}_2{\sim}_2$-classes so that they have at most two elements, and if they have exactly two elements, then for exactly one of them the predicate s holds on it. The formulas are defined by (here $\oplus$ stands for exclusive or):

$$\begin{aligned} \varphi_1 &:= \forall x.\exists y. x\,{}_1{\sim}_1\,y \wedge \varphi_{vertex}(y),\\ \varphi_2 &:= \forall x.\forall y. \varphi_{vertex}(x) \wedge \varphi_{vertex}(y) \rightarrow \neg(x\,{}_1{\sim}_1\,y \wedge x \neq y),\\ \varphi_3 &:= \forall x.\forall y.(x \neq y \wedge x\,{}_2{\sim}_2\,y) \rightarrow \Big(\big(\forall z. x\,{}_2{\sim}_2\,z \rightarrow (z = x \vee z = y)\big) \wedge (\mathsf{s}(x) \oplus \mathsf{s}(y))\Big),\\ \varphi_{cons} &:= \varphi_1 \wedge \varphi_2 \wedge \varphi_3. \end{aligned}$$

The formula φ_{cons} fits our purpose for two reasons. The first one is that the transformation of any graph satisfies φ_{cons}:

Lemma 2.3.29. *For any $\mathcal{G} \in \mathrm{Str}(\Sigma; \mathbb{G})$, we have $T(\mathcal{G}) \models \varphi_{cons}$.*

The second reason is that if a 2-data-multiset satisfies $T(\varphi) \wedge \varphi_{cons}$, we can build a graph which satisfies φ:

Lemma 2.3.30. *For any $\mathfrak{A} \in \mathrm{Str}(\Sigma'; 2\mathbb{DMS})$ and $\varphi \in \mathsf{FO}_{\mathbb{G}}[\Sigma; \{E\}]$ such that $\mathfrak{A} \models T(\varphi) \wedge \varphi_{cons}$, there is a $\mathcal{G} \in \mathrm{Str}(\Sigma; \mathbb{G})$ such that $\mathcal{G} \models \varphi$.*

Proof: We will actually prove the stronger proposition: "For any $\mathfrak{A} \in \mathrm{Str}(\Sigma'; 2\mathbb{DMS})$, there is a $\mathcal{G} \in \mathrm{Str}(\Sigma; \mathbb{G})$ such that for any $\varphi \in \mathsf{FO}_{\mathbb{G}}[\Sigma; \{E\}]$ we have $\mathfrak{A} \models T(\varphi)$ iff $\mathcal{G} \models \varphi$".

Let us assume that we have $\mathfrak{A} = (A, (P_\sigma)_{\sigma\in\Sigma'}, f_1, f_2)$ such that $\mathfrak{A} \models \varphi_{cons}$. We build $\mathcal{G} = (V, (P'_\sigma)_{\sigma\in\Sigma}, R_E)$ such that:

- $V = \{a \in A \mid \mathfrak{A} \models \varphi_{vertex}(a)\}$,
- for any $\sigma \in \Sigma$, $P'_\sigma = P_\sigma \cap V$,
- $R_E = \{(u, v) \in V \times V \mid \mathfrak{A} \models \varphi_{edge}(u, v)\}$.

To end the proof of the lemma, we do an induction on $\varphi \in \mathsf{FO}_{\mathbb{G}}[\Sigma; \{E\}]$ with the following induction hypothesis: "For any interpretation function I with values in V, we have $\mathfrak{A} \models T(\varphi)$ iff $\mathcal{G} \models \varphi$". □

Finally, by putting everything together, we are able to construct the reduction from $\mathbb{G}$-SAT($\mathsf{FO}; \{E\}$) to $2\mathbb{DMS}$-SAT($\mathsf{FO}; \mathcal{S}_2$). Let $\varphi \in \mathsf{FO}_{\mathbb{G}}[\Sigma; \{E\}]$. We constructively build $T(\varphi)$ and we claim that φ is satisfiable iff $T(\varphi) \wedge \varphi_{cons}$ is satisfiable. The forward direction is a direct implication of Lemmas 2.3.28 and 2.3.29. The backward direction is equivalent to Lemma 2.3.30. □

If we restrict to three variables, the problem remains undecidable. The proof is an adaptation of the proof of Theorem 2.3.27.

Theorem 2.3.31

$2\mathbb{DMS}$-SAT($\mathsf{FO}^3; \mathcal{S}_2$) is undecidable.

Proof: The proof of Theorem 2.3.27 proves this theorem too, up to two adjustments. The first one is to start from the problem $\mathbb{G}$-SAT($\mathsf{FO}^3; \{E\}$) instead of $\mathbb{G}$-SAT($\mathsf{FO}; \{E\}$). We know thanks to Theorem 2.3.20 that the problem $\mathbb{G}$-SAT($\mathsf{FO}^3; \{E\}$) is undecidable. The second adjustment is about the formula $\varphi_{edge}(x, y)$. It uses four variables but we can reuse x in order to use only three variables. We call the new formula $\varphi'_{edge}(x, y) \in \mathsf{FO}^3_{2\mathbb{DMS}}[\{\mathsf{s}\}; \mathcal{S}_2]$ and it is defined as:

$$\varphi'_{edge}(x, y) := \exists z.\big(x\,{}_1{\sim}_1\,z \;\wedge\; \mathsf{s}(z) \;\wedge\; \exists x.(z\,{}_2{\sim}_2\,x \;\wedge\; x\,{}_1{\sim}_1\,y)\big).$$

Then all the remaining formulas use at most three variables. □

However, if we restrain to the two-variable, The satisfiability problem become decidable:

Theorem 2.3.32 (*[32]*)

The problem $2\mathbb{DMS}$-SAT($\mathsf{FO}^2; \mathcal{S}_2$) is N2EXP-complete.

Proof: In [32], it is shown that the satisfiability problem of two-variable logic over arbitrary structures with two equivalence relations is decidable. We now do a straightforward reduction to this problem. When $\mathcal{R} = \{(1, 1), (2, 2)\}$, we actually deal with arbitrary finite structures $\mathfrak{A}$ with a number of unary predicates and two equivalence relations, namely ${}_1{\sim}_1^{\mathfrak{A}}$ and ${}_2{\sim}_2^{\mathfrak{A}}$. According to [32], two-variable first-order logic over those structures is decidable. □

If we stay with two data values but allows diagonal relations ${}_1{\sim}_2$ and ${}_2{\sim}_1$, the nature of the satisfiability problem is unknown:

Open Problem 2.3.33. *The nature of the problem* $2\mathbb{DMS}\text{-}\textsc{Sat}(\mathsf{FO}^2; \mathcal{A}_2)$ *is unknown yet.*

Next, if we increase to three data values per element, the satisfiability problem become undecidable, even with only two variables.

Theorem 2.3.34 (*[33]*)

The problems $3\mathbb{DMS}\text{-}\textsc{Sat}(\mathsf{FO}^2; \mathcal{S}_3)$ *is undecidable.*

So far, we have reviewed already existing results on logics over data-multisets. In the next chapter, we shall introduce a new approach to circumvent the undecidability of FO over 2-data-multisets. The approach is based on a restriction of the allowed formulas, distinct from restraining the number of variables.

Chapter 3

The Local Fragments

From this chapter to the end of this document, we focus on first order logic on κ-data-multisets. Their signature is $\kappa\mathbb{DMS}$ and we presented them along with pre-existing results in section 2.3.5. We recall that a κ-data-multiset $\mathfrak{A} \in \mathrm{Str}(\Sigma; \kappa\mathbb{DMS})$ is a tuple $(A, (P_\sigma)_{\sigma\in\Sigma}, f_1, f_2, \ldots, f_\kappa)$ where A is a set, the P_σs are subsets of A and $f_1, f_2, \ldots, f_\kappa$ are functions $A \to \mathbb{N}$ representing data values. In particular, we will have $\mathcal{R} \subseteq \mathcal{A}_\kappa = \{{}_i{\sim}_j \mid 1 \leq i, j \leq \kappa\}$.

Motivations A computer science oriented application of data-multisets is to model the behavior of distributed algorithms. More precisely, we can see each element of a data multiset as a process or a computing unit. As there is no relationship between the elements, we therefore do not assume a particular processes architecture, but rather consider clouds of computing units. Then we can see the data values as data that the processes exchange, and in the case where $\kappa = 2$, we can see the first data value as an input and the second data value as an output. We now introduce the leader election problem. In leader election, a process gets its unique identifier as input, and it should eventually output the identifier of a common leader. We can define the formula $\varphi_{leader} \in \mathsf{FO}_{2\mathbb{DMS}}[\emptyset; \mathcal{A}_2]$ which states that there is an element x such that for any other element y, the output of y is the input of x:

$$\varphi_{leader} = \exists x. \forall y. x \,{}_1{\sim}_2\, y.$$

The first fundamental question that arises is whether a given specification is consistent. This leads us to the satisfiability problem. The following negative result, encountered in Chapter 2, calls for restrictions of the general logic (recalling that $\mathcal{S}_2 = \{{}_1{\sim}_1, {}_2{\sim}_2\}$):

Theorem 2.3.27 (*[29]*)

The problem $2\mathbb{DMS}$-$\textsc{Sat}(\mathsf{FO}; \mathcal{S}_2)$ *is undecidable.*

3.1 Definitions

First, the view of a node a includes all elements whose distance to a is bounded by a given radius. It is formalized using the notion of a Gaifman graph (for an introduction, see [37]). We use here a variant that is suitable for our setting and that we call *data graph*. Fix the set $\mathcal{R}$. Given a κ-data-multiset $\mathfrak{A} = (A, (P_\sigma)_\sigma, f_1, \dots, f_\kappa) \in \mathrm{Str}(\Sigma; \kappa\mathbb{DMS})$, its *data graph* $\mathcal{G}_\mathcal{R}(\mathfrak{A})$ is the graph with vertices being the data locations of $\mathfrak{A}$, and two data locations are connected if they either belong to the same element or they store the same data value (and are comparable with $\mathcal{R}$). Formally, we define $\mathcal{G}_\mathcal{R}(\mathfrak{A}) = (V_{\mathcal{G}_\mathcal{R}(\mathfrak{A})}, E_{\mathcal{G}_\mathcal{R}(\mathfrak{A})})$ with set of vertices

$$V_{\mathcal{G}_\mathcal{R}(\mathfrak{A})} = A \times \{1, \dots, \kappa\}$$

and set of edges

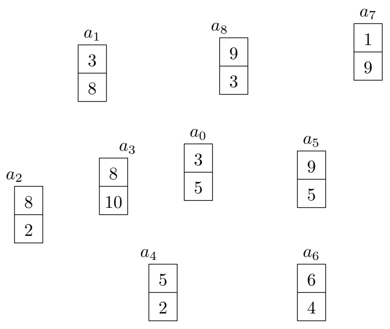


Figure 3.1: A 2-data-multiset (so $\kappa = 2$).

$$E_{\mathcal{G}_\mathcal{R}(\mathfrak{A})} = \{((a,i),(b,j)) \in V_{\mathcal{G}_\mathcal{R}(\mathfrak{A})} \times V_{\mathcal{G}_\mathcal{R}(\mathfrak{A})} \mid a = b, \text{ or } {}_i{\sim}_j \in \mathcal{R} \text{ and } a\,{}_i{\sim}_j\,b\}.$$

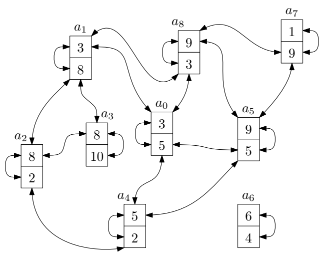
When $\mathcal{R} = \mathcal{A}_\kappa$ we can omit it and write $\mathcal{G}(\mathfrak{A})$ for $\mathcal{G}_\mathcal{R}(\mathfrak{A})$. Note that when $\mathcal{R} \neq \mathcal{A}_\kappa$ the data graph might be oriented. Examples of data graphs are depicted in Figure 3.2.

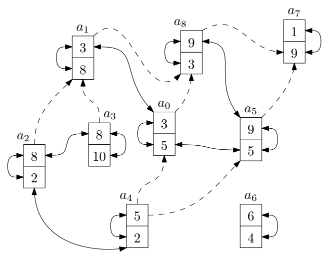
We define the distance $d^\mathfrak{A}_\mathcal{R}((a,i),(b,j)) \in \overline{\mathbb{N}}$ between two elements (a,i) and (b,j) from $A \times \{1, \dots, \kappa\}$ as the length of the shortest *directed* path from (a,i) to (b,j) in $\mathcal{G}_\mathcal{R}(\mathfrak{A})$. In fact, as the graph is directed, the distance function might not be symmetric. For $a \in A$ and $r \in \mathbb{N}$ representing a radius, the *r-ball around a* is the set

$$B^\mathfrak{A}_{r,\mathcal{R}}(a) = \{(b,j) \in V_{\mathcal{G}_\mathcal{R}(\mathfrak{A})} \mid d^\mathfrak{A}_\mathcal{R}((a,i),(b,j)) \leq r \text{ for some } i \in \{1, \dots, \kappa\}\}.$$

This ball contains the data locations of $\mathfrak{A}$ that can be reached from some data location of a through a path of length at most r.

Let $f_{\text{new}} : A \times \{1, \dots, \kappa\} \to \mathbb{N} \setminus \mathit{Val}^\mathfrak{A}(A)$ be an injective mapping. The *r-view of a in $\mathfrak{A}$*


(a) With $\mathcal{R} = \{{}_1{\sim}_1, {}_2{\sim}_2, {}_1{\sim}_2, {}_2{\sim}_1\}$.


(b) With $\mathcal{R} = \{{}_1{\sim}_1, {}_2{\sim}_2, {}_1{\sim}_2\}$.

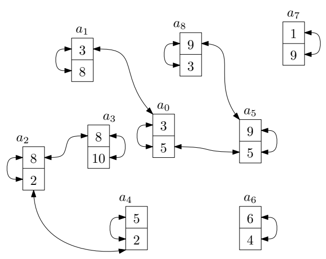

(c) With $\mathcal{R} = \{{}_1{\sim}_1, {}_2{\sim}_2\}$.

Figure 3.2: Different data graphs associated to the example in Figure 3.1, depending on the set of binary symbols $\mathcal{R}$ considered. Unidirectional edges are dashed.

with exterior is the structure $\mathfrak{A}|_{a,\mathcal{R}}^{r,\text{ext}} = (A, (P_\sigma)_\sigma, f'_1, \ldots, f'_\kappa) \in \text{Str}(\Sigma; \kappa\mathbb{DMS})$. Its universe is the same as the one of $\mathfrak{A}$, and the unary predicates stay the same and its data functions are

$$f'_i(b) = \begin{cases} f_i(b) & \text{if } (b,i) \in B^{\mathfrak{A}}_{r,\mathcal{R}}(a), \\ f_{\text{new}}((b,i)) & \text{otherwise.} \end{cases}$$

Example of views with exterior are depicted in Figure 3.3. Additionally, The *r-view of a*

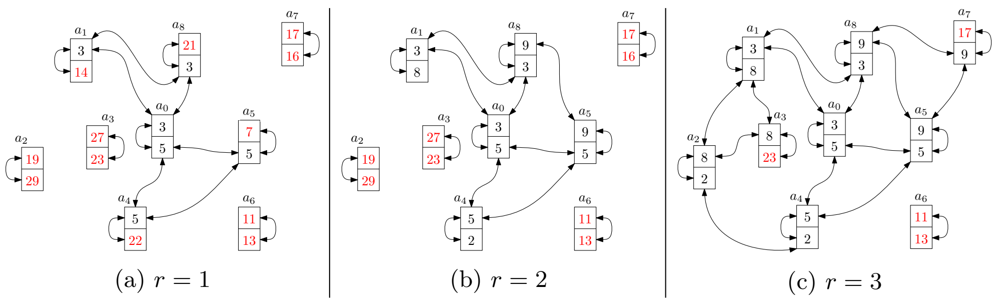


Figure 3.3: Some wiews with exterior around a_0 with $\mathcal{R} = \{{}_1{\sim}_1, {}_2{\sim}_2, {}_1{\sim}_2, {}_2{\sim}_1\}$ and for various radiuses r of the structure depicted in Figure 3.1. Modified data values are in red.

in $\mathfrak{A}$ *without exterior* is the structure $\mathfrak{A}|_{a,\mathcal{R}}^{r,\text{int}} = (A'', (P''_\sigma)_\sigma, f''_1, \ldots, f''_\kappa) \in \text{Str}(\Sigma; \kappa\mathbb{DMS})$. Its universe is

$$A'' = \{b \in A \mid (b,i) \in B^{\mathfrak{A}}_{r,\mathcal{R}}(a) \text{ for some } i \in \{1, \ldots, \kappa\}\}.$$

Moreover, P''_σ is the restriction of P_σ to A'' and f''_i is the restriction of f'_i to A''. Example of views without exterior are depicted in Figures 3.4 and 3.5.

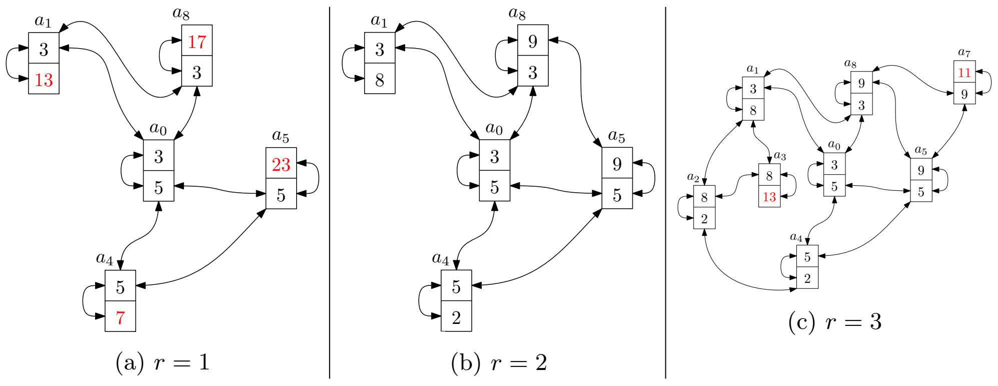


Figure 3.4: Some wiews without exterior around a_0 with $\mathcal{R} = \{{}_1{\sim}_1, {}_2{\sim}_2, {}_1{\sim}_2, {}_2{\sim}_1\}$ and for various radiuses r of the structure depicted in Figure 3.1. Modified data values are in red.

Remark 3.1.1. The choice of the function f_{new} is not unique. Although this choice does not change the interpretations of the data comparison relations ${}_i{\sim}_j$, which implies that for two different f_{new}, the resulting views are indiscernible with respect to $\mathsf{FO}_{\kappa\mathbb{DMS}}[\Sigma; \mathcal{R}]$. Furthermore, Propositions 3.2.5 and 3.2.8 show this too. $\diamond$

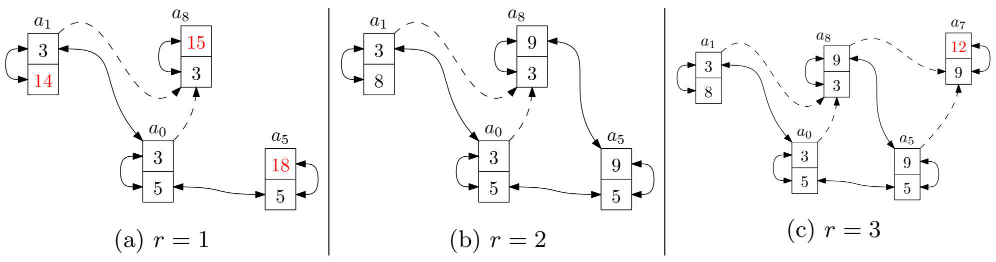


Figure 3.5: Some views without exterior around a_0 with $\mathcal{R} = \{{}_1\!\sim_1, {}_2\!\sim_2, {}_1\!\sim_2\}$ and for various radiuses r of the structure depicted in Figure 3.1. Unidirectional edges are dashed. Modified data values are in red.

Remark 3.1.2. Alternatively in the definition of the views, instead of using f_{new}, we could add a new value `None`. The semantic of `None` would be that if $f_i(a) = \texttt{None}$ then $\mathfrak{A} \not\models a\,{}_i\!\sim_j b$. Then at the data location outside the ball, we set its data value to `None`. This construction leads to the same semantic, but we prefer to not use it as it adds a distinctive new data value. $\diamond$

We are now ready to present the *local fragments* of first order logic over κ-data-multisets. The logic uses a new modality called the *local modality* $\langle\!\langle\psi\rangle\!\rangle_x^{r,\wp}$ which serves to constrain data values comparisons. The idea of this new modality is to allow ψ to only have access to the data locations at distance less or equal than r of x. The modality is interpreted using the views previously defined. As views are available in two sorts –with or without exterior–, the local fragments are available in two sorts too. We use the parameter $\wp$ which takes value in the set $\{\text{ext}, \text{int}\}$. The local fragment is denoted by $r\text{-}\mathsf{LF}_\kappa^\wp[\Sigma; \mathcal{R}]$ and is given by the grammar

$$\varphi \ ::= \ \langle\!\langle\psi\rangle\!\rangle_x^{r,\wp} \ \mid \ x = y \ \mid \ \exists x.\varphi \ \mid \ \varphi \vee \varphi \ \mid \ \neg\varphi$$

where ψ is a formula from $\mathsf{FO}_{\kappa\mathbb{DMS}}[\Sigma; \mathcal{R}]$ with (at most) one free variable x. For $\mathfrak{A} \in \text{Str}(\Sigma; \kappa\mathbb{DMS})$ and an interpretation function I, we set the truth value of the local modality as

$$\mathfrak{A} \models_I \langle\!\langle\psi\rangle\!\rangle_x^{r,\wp} \ \text{ iff } \ \mathfrak{A}|_{I(x),\mathcal{R}}^{r,\wp} \models_I \psi.$$

The semantic of the local modality depends on the value of $\mathcal{R}$ but we omit it because most of the time $\mathcal{R}$ is fixed and obvious. But if we want to specify it, we will put $\mathcal{R}$ as index. When $\wp = \text{ext}$, the notation $r\text{-}\mathsf{LF}_\kappa^{\text{ext}}[\Sigma; \mathcal{R}]$ denotes the local fragment with exterior and when $\wp = \text{int}$, the notation $r\text{-}\mathsf{LF}_\kappa^{\text{int}}[\Sigma; \mathcal{R}]$ denotes the local fragment without exterior. Note that in the notation of the local fragment $r\text{-}\mathsf{LF}_\kappa^\wp[\Sigma; \mathcal{R}]$, we write κ for the signature instead of $\kappa\mathbb{DMS}$. We allow ourself to do this because the local fragment only applies to κ-data-multisets and in order to lighten the notations.

Example 3.1.3 (Some local formulas). Let $\mathfrak{A}$ be the structure depicted in Figure 3.1. Let

$\psi_1(x)$ be the formula:

$$\psi_1(x) := \exists y.\exists z.\exists t.\ \Big(x\,{}_1{\sim}_1\,y \wedge y\,{}_2{\sim}_1\,z \wedge z\,{}_2{\sim}_2\,t \wedge t\,{}_1{\sim}_2\,x \\ \wedge x \neq y \wedge x \neq z \wedge x \neq t \wedge y \neq z \wedge y \neq t \wedge z \neq t\Big).$$

We have $\mathfrak{A} \models \exists x.\langle\!\langle \psi_1 \rangle\!\rangle^{3,\mathrm{int}}_{x,\mathcal{A}_2}$ while $\mathfrak{A} \not\models \exists x.\langle\!\langle \psi_1 \rangle\!\rangle^{2,\mathrm{int}}_{x,\mathcal{A}_2}$ and $\mathfrak{A} \not\models \exists x.\langle\!\langle \psi_1 \rangle\!\rangle^{3,\mathrm{int}}_{x,\mathcal{I}_2}$.

We now define $\psi_2(x) = \forall y. \bigvee_{i\sim_j \in \mathcal{R}} x\,{}_i{\sim}_j\,y$. For any 2-data-multiset $\mathfrak{B}$, we have $\mathfrak{B} \models \forall x.\langle\!\langle \psi_2 \rangle\!\rangle^{1,\mathrm{int}}_{x,\mathcal{R}}$ while we have $\mathfrak{A} \not\models \forall x.\langle\!\langle \psi_2 \rangle\!\rangle^{1,\mathrm{ext}}_{x,\mathcal{R}}$ ◇

We introduce the *existential fragment* of r-$\mathsf{LF}^{\wp}_{\kappa}[\Sigma;\mathcal{R}]$, which is denoted by $\exists$-r-$\mathsf{LF}^{\wp}_{\kappa}[\Sigma;\mathcal{R}]$ and given by the grammar

$$\varphi ::= \langle\!\langle \psi \rangle\!\rangle^{r,\wp}_{x} \mid x = y \mid \neg(x = y) \mid \exists x.\varphi \mid \varphi \vee \varphi \mid \varphi \wedge \varphi$$

where ψ is a formula from $\mathsf{FO}_{\kappa\mathbb{DMS}}[\Sigma;\mathcal{R}]$ with (at most) one free variable x. The quantifier free fragment qf-r-$\mathsf{LF}^{\wp}_{\kappa}[\Sigma;\mathcal{R}]$ is defined by the grammar

$$\varphi ::= \langle\!\langle \psi \rangle\!\rangle^{r,\wp}_{x} \mid x = y \mid \neg(x = y) \mid \varphi \vee \varphi \mid \varphi \wedge \varphi.$$

In Chapter 5, we will study the existential fragment and establish when it is decidable.

Remark 3.1.4. Note that for both these fragments, we do not impose any restrictions on the use of quantifiers in the formula ψ located inside the local modality $\langle\!\langle \psi \rangle\!\rangle^{r,\wp}_{x}$. ◇

3.2 Elementary facts about the local fragments

This sections exposes elementary facts on the local fragments. It is not mandatory to read this section in order to understand the proofs of decidability of the next sections, still it might be helpful for the reader in order to build an intuition of the local fragments. Proposition 3.2.5 and 3.2.8 are interesting because they show that we could define the local fragment in a purely syntactic way (even if it would be cumbersome).

In the first subsection 3.2.1 we show that the local fragment is indeed a fragment of first order logic and then how the different fragments are included in each other. In the second subsection 3.2.2 we study the relationship of the local fragments with the two variable fragment and the guarded fragment. In the third subsection 3.2.3 we investigate and formalize the link between the data graph and the Gaifman graph of a data-multiset.

3.2.1 Inclusions analysis

In the following subsection, we first justify the usage of the term fragment, that is that the local fragments are included in first order logic. Then we show that when r increases, the expressiveness increases too. Finally we study the relationship of the fragment with exterior with the one without exterior. In Figure 3.6 is depicted the inclusions we will show. For some inclusions, we will conjecture if it is strict or not although we can not show that an inclusion is strict. Indeed we lack tools in order to characterize the expressiveness of a local fragment yet. Filling this gap could lead to interesting future works for example by adapting the Ehrenfeucht-Fraïssé games to our context. As we do not have the tools yet to prove that something can not be expressed in the local fragment, it will only be conjectures.

But first, we have to show that the distance function can be expressed in first order logic.

$$\begin{array}{ccccccccc} 1\text{-}\mathsf{LF}^{\text{ext}}_{\kappa} & \subseteq & 2\text{-}\mathsf{LF}^{\text{ext}}_{\kappa} & \subseteq & 3\text{-}\mathsf{LF}^{\text{ext}}_{\kappa} & \subseteq & \cdots & \subseteq & \mathsf{FO}_{\kappa\mathbb{DMS}} \\ \cup\!\shortmid & & \cup\!\shortmid & & \cup\!\shortmid & & & & \\ 1\text{-}\mathsf{LF}^{\text{int}}_{\kappa} & \subseteq & 2\text{-}\mathsf{LF}^{\text{int}}_{\kappa} & \subseteq & 3\text{-}\mathsf{LF}^{\text{int}}_{\kappa} & \subseteq & \cdots & & \end{array}$$

Figure 3.6: The relationship between the different local fragments and first order logic. The sets Σ and $\mathcal{R}$ are fixed and we omit them for clarity.

Internalizing the distance

We show that the distance $d_{\mathcal{R}}((a,i),(b,j))$ can be internalized in first order logic, that is it is possible to express in first order logic that two data locations are at distance at most r. We define a family of formulas $\varphi^{\leq r}_{i,j,\mathcal{R}}(x,y) \in \mathsf{FO}_{\kappa\mathbb{DMS}}[\Sigma;\mathcal{R}]$ which reflects that the data locations (a,i) and (b,j) are at distance at most r. The formulas have two free variables and are parametrized with $i,j \in \{1,\dots,\kappa\}$ and $r \in \mathbb{N}$ and $\mathcal{R}$. The definition is inductive on r. Intuitively, for the definition of $\varphi^{\leq r+1}_{i,j,\mathcal{R}}$ we use the fact that two data locations (a,i) and (b,j) are at distance at most $r+1$ iff there is a third data location (c,k) such that it is at distance at most r from (a,i) and (b,j) is at distance at most 1 from (c,k). The formula $\perp$ denotes a formula which is never satisfied, i.e. is for any structure $\mathfrak{A}$, we have $\mathfrak{A} \not\models \perp$.

$$\varphi^{\leq 0}_{i,j,\mathcal{R}}(x,y) = \begin{cases} x = y & \text{if } i = j \\ \perp & \text{if } i \neq j \end{cases},$$

$$\varphi^{\leq 1}_{i,j,\mathcal{R}}(x,y) = \begin{cases} x = y \;\vee\; x \,{}_i\!\sim_j y & \text{if } {}_i\!\sim_j \in \mathcal{R} \\ x = y & \text{if } {}_i\!\sim_j \notin \mathcal{R} \end{cases},$$

$$\text{for } r \geq 1, \quad \varphi^{\leq r+1}_{i,j,\mathcal{R}}(x,y) = \exists z. \bigvee_{k=1}^{\kappa} \big(\varphi^{\leq r}_{i,k,\mathcal{R}}(x,z) \;\wedge\; \varphi^{\leq 1}_{k,j,\mathcal{R}}(z,x)\big).$$

The next lemma states that the formulas $\varphi^{\leq r}_{i,j,\mathcal{R}}(x,y)$ are correct.

Lemma 3.2.1. *For any structure $\mathfrak{A} \in \mathrm{Str}(\Sigma;\kappa\mathbb{DMS})$ and a, b elements of $\mathfrak{A}$, we have that*

$$d^{\mathfrak{A}}_{\mathcal{R}}((a,i),(b,j)) \leq r \;\; \textit{iff} \;\; \mathfrak{A} \models \varphi^{\leq r}_{i,j,\mathcal{R}}(a,b).$$

The proof is by induction on r. As a corollary, we can internalize the r-balls. Concretely, we can define a family of formulas $\varphi^{r}_{Ball,j,\mathcal{R}}(x,y) \in \mathsf{FO}_{\kappa\mathbb{DMS}}[\Sigma;\mathcal{R}]$ which reflect that the data location (y,j) is in the ball of radius r around x. The formulas have two free variables and is parametrized with $j \in \{1, \ldots, \kappa\}$ and $r \in \mathbb{N}$ and $\mathcal{R}$. The definition of $\varphi^{r}_{Ball,j,\mathcal{R}}$ use the formulas $\varphi^{\leq r}_{i,j,\mathcal{R}}$:

$$\varphi^{r}_{Ball,j,\mathcal{R}}(x,y) = \bigvee_{i=1}^{\kappa} \varphi^{\leq r}_{i,j,\mathcal{R}}(x,y)$$

Corollary 3.2.2. *For any structure $\mathfrak{A} \in \mathrm{Str}(\Sigma; \kappa\mathbb{DMS})$ and a, b elements of $\mathfrak{A}$, we have that*

$$(b,j) \in B^{\mathfrak{A}}_{r,\mathcal{R}}(a) \ \text{ iff } \ \mathfrak{A} \models \varphi^{r}_{Ball,j,\mathcal{R}}(a,b).$$

The corollary directly comes out of the definition of balls and lemma 3.2.1.

The local fragment is indeed a fragment

We build a translation from $r\text{-}\mathsf{LF}^{\wp}_{\kappa}[\Sigma;\mathcal{R}]$ to $\mathsf{FO}_{\kappa\mathbb{DMS}}[\Sigma;\mathcal{R}]$ preserving the satisfaction relation. We first tackle the case where $\wp = \mathrm{ext}$ and then the case where $\wp = \mathrm{int}$.

With exterior Given a formula $\varphi \in r\text{-}\mathsf{LF}^{\wp}_{\kappa}[\Sigma;\mathcal{R}]$, some of its subformulas will be local modalities of the form $\langle\!\langle \psi \rangle\!\rangle^{r,\mathrm{ext}}_{x}$. And within ψ they are data comparison atoms $y\,{}_j{\sim}_k\,z$. We transform such an atom into the formula $[\![y\,{}_j{\sim}_k\,z]\!]^{r}_{x,\mathcal{R}}$, and this formula has to emulate the data comparison under the view around x. So for the definition of $[\![y\,{}_j{\sim}_k\,z]\!]^{r}_{x,\mathcal{R}}$, we have to check that the data locations (y, j) and (z, k) are in the r-ball around x and that their data values are the same. Although when $j = k$, it is also sufficient that $y = z$ because in a view, a data location always has the same data as itself, even outside the ball. So we have to give two definitions whether $j = k$ or not. When $j \neq k$, the formula $[\![y\,{}_j{\sim}_k\,z]\!]^{r}_{x,\mathcal{R}}$ is defined by

$$[\![y\,{}_j{\sim}_k\,z]\!]^{r}_{x,\mathcal{R}} = y\,{}_j{\sim}_k\,z \ \wedge \ \varphi^{r}_{Ball,j,\mathcal{R}}(x,y) \ \wedge \ \varphi^{r}_{Ball,k,\mathcal{R}}(x,z),$$

and when $j = k$, the formula $[\![y\,{}_j{\sim}_j\,z]\!]^{r}_{x,\mathcal{R}}$ is defined by

$$[\![y\,{}_j{\sim}_j\,z]\!]^{r}_{x,\mathcal{R}} = \big(y\,{}_j{\sim}_j\,z \ \wedge \ \varphi^{r}_{Ball,j,\mathcal{R}}(x,y) \ \wedge \ \varphi^{r}_{Ball,j,\mathcal{R}}(x,z)\big) \ \vee \ y = z.$$

Next is the key lemma for the inclusion analysis, which states that we can simulate data comparisons within a view in first order logic.

Lemma 3.2.3. *For any structure $\mathfrak{A} \in \mathrm{Str}(\Sigma; \kappa\mathbb{DMS})$ and a an elements of $\mathfrak{A}$ and b, c elements of $\mathfrak{A}|^{r,\wp}_{a,\mathcal{R}}$, we have that*

$$\mathfrak{A}|^{r,\wp}_{a,\mathcal{R}} \models b\,{}_j{\sim}_k\,c \ \text{ iff } \ \mathfrak{A} \models [\![b\,{}_j{\sim}_k\,c]\!]^{r}_{a,\mathcal{R}}.$$

Proof: Let us express the component of $\mathfrak{A}$ and $\mathfrak{A}|^{r,\wp}_{a,\mathcal{R}}$ as $\mathfrak{A} = (A, (P_\sigma)_\sigma, f_1, \ldots, f_\kappa)$ and $\mathfrak{A}|^{r,\wp}_{a,\mathcal{R}} = (A', (P'_\sigma)_\sigma, f'_1, \ldots, f'_\kappa)$.

Let us assume that $\mathfrak{A}|^{r,\wp}_{a,\mathcal{R}} \models b\,{}_j{\sim}_k\,c$. We do a case analysis, the first case being $(b, j) \in B^{\mathfrak{A}}_{r,\mathcal{R}}(a)$. Then by definition of the view $\mathfrak{A}|^{r,\wp}_{a,\mathcal{R}}$, we have $f'_j(b) = f_j(b)$. By assumption we have $f'_j(b) = f'_k(c)$, which implies that $(c, k) \in B^{\mathfrak{A}}_{r,\mathcal{R}}(a)$, because otherwise by definition of the view, $f'_k(c) = f_{\text{new}}((c,k)) \neq f_k(b)$ by assumption on f_{new}. So we have that $f'_k(c) = f_k(c)$. Put together, we have $f_j(b) = f'_j(b) = f'_k(c) = f_k(c)$ which is equivalent to $\mathfrak{A} \models b\,{}_j{\sim}_k\, c$. And as $(b,j),(c,k) \in B^{\mathfrak{A}}_{r,\mathcal{R}}(a)$ by Corollary 3.2.2, we have that $\mathfrak{A} \models \varphi^r_{Ball,j,\mathcal{R}}(a,b)$ and $\mathfrak{A} \models \varphi^r_{Ball,k,\mathcal{R}}(a,c)$. So $\mathfrak{A} \models [\![b\,{}_j{\sim}_k\, c]\!]^r_{a,\mathcal{R}}$ as expected. The second case is when $(b,j) \notin B^{\mathfrak{A}}_{r,\mathcal{R}}(a)$. Then by definition of the view, we have $f'_j(b) = f_{\text{new}}(b)$ and by assumption we have $f'_j(b) = f'_k(c)$. Then by assumption on f_{new} and injectivity of f_{new}, we have $b = c$ and $j = k$. Thus $\mathfrak{A} \models [\![b\,{}_j{\sim}_k\, c]\!]^r_{a,\mathcal{R}}$ as expected.

For the converse, let us assume that $\mathfrak{A} \models [\![b\,{}_j{\sim}_k\, c]\!]^r_{a,\mathcal{R}}$. From the last assumptions, there are two possibilities, the first one being that $\mathfrak{A} \models b\,{}_j{\sim}_k\, c \wedge \left(\varphi^r_{Ball,j,\mathcal{R}}(a,b)\right) \wedge \left(\varphi^r_{Ball,k,\mathcal{R}}(a,c)\right)$. So we have first that $f_j(b) = f_k(c)$ and second by Corollary 3.2.2, it implies that both (b,j) and (c,k) are in $B^{\mathfrak{A}}_{r,\mathcal{R}}(a)$. Then by definition of the view, we have that $f'_j(b) = f_j(b)$ and $f'_k(c) = f_k(c)$. Put together, we have $f'_j(b) = f_j(b) = f_k(c) = f'_k(c)$ which is equivalent to our goal $\mathfrak{A}|^{r,\wp}_{a,\mathcal{R}} \models b\,{}_j{\sim}_k\, c$. The second possibility is that $j = k$ and $b = c$, which obviously implies that $f'_j(b) = f'_k(c)$ and concludes. $\square$

We go back on $\langle\!\langle\psi\rangle\!\rangle^{r,\wp}_x$. Within ψ, we transform each $y\,{}_j{\sim}_k\, z$ into $[\![y\,{}_j{\sim}_k\, z]\!]^r_{x,\mathcal{R}}$ and we call the resulting formula $\psi' \in \mathsf{FO}_{\kappa\mathbb{DMS}}[\Sigma;\mathcal{R}]$. The formula ψ' is equivalent to the local modality with exterior $\langle\!\langle\psi\rangle\!\rangle^{r,\text{ext}}_x$. We express it formally in the following lemma:

Lemma 3.2.4. *For any structure $\mathfrak{A} \in \mathrm{Str}(\Sigma;\kappa\mathbb{DMS})$ and any context $I : \mathcal{V} \to A$, we have:*

$$\mathfrak{A} \models_I \langle\!\langle\psi\rangle\!\rangle^{r,\text{ext}}_x \quad \textit{iff} \quad \mathfrak{A} \models_I \psi'.$$

The proof is done by induction on ψ and thanks to Lemma 3.2.3.

We are now able to describe the translation from the local fragment with exterior.

Proposition 3.2.5. *For any $\varphi \in r\text{-}\mathsf{LF}^{\text{ext}}_\kappa[\Sigma;\mathcal{R}]$, we can build in polynomial time an equivalent formula $\varphi' \in \mathsf{FO}_{\kappa\mathbb{DMS}}[\Sigma;\mathcal{R}]$. That is, for any structure $\mathfrak{A} \in \mathrm{Str}(\Sigma;\kappa\mathbb{DMS})$ we have $\mathfrak{A} \models \varphi$ iff $\mathfrak{A} \models \varphi'$.*

Proof: In φ, we replace each local modality $\langle\!\langle\psi\rangle\!\rangle^{r,\text{ext}}_x$ by ψ' as described above and it produces φ'. In order to show that the two formulas φ and φ' are equivalent, we do an induction on φ and use lemma 3.2.4. $\square$

Remark 3.2.6. Proposition 3.2.5 shows that the local fragment is indeed a fragment. One can wonder if the inclusion is strict in the following sense: for any $\varphi \in \mathsf{FO}_{\kappa\mathbb{DMS}}[\Sigma;\mathcal{R}]$, can we pick r and build $\varphi' \in r\text{-}\mathsf{LF}^{\text{ext}}_\kappa[\Sigma;\mathcal{R}]$ such that φ and φ' are equivalent. Gaifman's theorem (Originally presented in [23] or can be found in book [37] as Theorem 4.22) states that every formula of first order logic is equivalent to a local one. A direct application of the theorem shows that the inclusion is not strict. But the translation can lead to a non-elementary blow-up in size, see [14]. $\diamond$

Without exterior We now tackle the case when $\wp = \text{int}$. We will use our previous work on the case $\wp = \text{ext}$ as a starting point. As we restrain the universe in the view without exterior, we have one more step to do compared to the view with exterior: to constrain the quantification in ψ'. We start with ψ' defined right before Lemma 3.2.4 and then for any subformula $\exists y.\theta$ of it, we replace it with:

$$\exists y.\Big(\bigvee_{i,j=1}^{\kappa} \varphi^{\leq r}_{i,j,\mathcal{R}}(x,y)\Big) \wedge \theta$$

and call $\psi'' \in \mathsf{FO}_{\kappa\mathbb{DMS}}[\Sigma;\mathcal{R}]$ the result. The formula ψ'' is equivalent to the local modality without exterior $\langle\!\langle\psi\rangle\!\rangle_x^{r,\mathrm{int}}$. We express it formally in the following lemma:

Lemma 3.2.7. *For any structure $\mathfrak{A} \in \mathrm{Str}(\Sigma;\kappa\mathbb{DMS})$ and any context $I : \mathcal{V} \to A$, we have:*

$$\mathfrak{A} \models_I \langle\!\langle\psi\rangle\!\rangle_x^{r,\mathrm{int}} \quad \textit{iff} \quad \mathfrak{A} \models_I \psi''.$$

Proof: We only give the structure of the proof. Let us denote $a = I(x)$ and by A'' the universe of $\mathfrak{A}|^{r,\mathrm{int}}_{a,\mathcal{R}}$. First prove that the set A'' is equal to $\{b \in A \mid \mathfrak{A} \models \bigvee_{i,j=1}^{\kappa} \varphi^{\leq r}_{i,j,\mathcal{R}}(a,b)\}$. Then the lemma can be proved by an induction on ψ (with an arbitrary numbers of free variables), with the following induction hypothesis: For any context $I' : \mathcal{V} \to A''$, we have

$$\mathfrak{A}|^{r,\mathrm{int}}_{a,\mathcal{R}} \models_{I'} \psi \ \text{ iff } \ \mathfrak{A} \models_{I'} \psi''. \qquad \square$$

We are now able to describe the translation from the local fragment without exterior. From now, the proof is similar to the case $\wp = \mathrm{ext}$.

Proposition 3.2.8. *For any $\varphi \in r\text{-}\mathsf{LF}^{\mathrm{int}}_{\kappa}[\Sigma;\mathcal{R}]$, we can build in polynomial time an equivalent formula $\varphi' \in \mathsf{FO}_{\kappa\mathbb{DMS}}[\Sigma;\mathcal{R}]$. That is, for any structure $\mathfrak{A} \in \mathrm{Str}(\Sigma;\kappa\mathbb{DMS})$ we have $\mathfrak{A} \models \varphi$ iff $\mathfrak{A} \models \varphi'$.*

Proof: In φ, we replace each local modality $\langle\!\langle\psi\rangle\!\rangle_x^{r,\mathrm{int}}$ by ψ'' as described above and it produces φ'. In order to show that the two formulas φ and φ' are equivalent, we do an induction on φ and use lemma 3.2.7. $\square$

Now that we have shown that the local fragments are indeed fragments of first order logic, we can deduce from it the nature of the satisfiability problem with one data value, as stated in the next Corollary.

Corollary 3.2.9. *for any integer r, the two problems $1\mathbb{DMS}\text{-}\textsc{Sat}(r\text{-}\mathsf{LF}^{\mathrm{ext}};\{\sim\})$ and $1\mathbb{DMS}\text{-}\textsc{Sat}(r\text{-}\mathsf{LF}^{\mathrm{int}};\{\sim\})$ are decidable. More precisely, they are in* N2Exp.

Proof: Thanks to Propositions 3.2.5 and 3.2.8, we know that the problems can be reduced to $1\mathbb{DMS}\text{-}\textsc{Sat}(\mathsf{FO};\{\sim\})$. And Theorem 2.3.25 states that the latter is in N2Exp. $\square$

The monotonicity of the expressiveness

Here, we show that expressiveness increases when the radius r increases and when we go from $\wp = \mathrm{int}$ to $\wp = \mathrm{ext}$. That is, $r\text{-}\mathsf{LF}^{\wp}_{\kappa}[\Sigma;\mathcal{R}]$ is contained in $r'\text{-}\mathsf{LF}^{\wp}_{\kappa}[\Sigma;\mathcal{R}]$ for any $r < r'$ and $r\text{-}\mathsf{LF}^{\mathrm{int}}_{\kappa}[\Sigma;\mathcal{R}]$ is contained in $r\text{-}\mathsf{LF}^{\mathrm{ext}}_{\kappa}[\Sigma;\mathcal{R}]$. The techniques and proofs are similar to the one of Propositions 3.2.5 and 3.2.8.

We first state the monotonicity of expressiveness for the local fragment with exterior.

Proposition 3.2.10. *For any $r < r'$ and $\varphi \in r\text{-}\mathsf{LF}^{\text{ext}}_{\kappa}[\Sigma;\mathcal{R}]$, we can build in polynomial time an equivalent formula $\varphi' \in r'\text{-}\mathsf{LF}^{\text{ext}}_{\kappa}[\Sigma;\mathcal{R}]$. That is, for any structure $\mathfrak{A} \in \mathrm{Str}(\Sigma;\kappa\mathbb{DMS})$ we have $\mathfrak{A} \models \varphi$ iff $\mathfrak{A} \models \varphi'$.*

Proof: In φ, for each $\langle\!\langle \psi \rangle\!\rangle^{r,\text{ext}}_{x}$, construct ψ' as right before Lemma 3.2.4 and then replace the local modality with $\langle\!\langle \psi' \rangle\!\rangle^{r',\text{ext}}_{x}$ and this yields φ'. Then, the proof of the Proposition is similar to the one of Proposition 3.2.5, and the equivalent of the lemma 3.2.4 is that for any structure $\mathfrak{A} \in \mathrm{Str}(\Sigma;\kappa\mathbb{DMS})$ and any context $I : \mathcal{V} \to A$, we have:

$$\mathfrak{A} \models_I \langle\!\langle \psi \rangle\!\rangle^{r,\text{ext}}_{x} \text{ iff } \mathfrak{A} \models_I \langle\!\langle \psi' \rangle\!\rangle^{r',\text{ext}}_{x}. \qquad \square$$

Now, we state the monotonicity of expressiveness for the local fragment without exterior.

Proposition 3.2.11. *For any $r < r'$ and $\varphi \in r\text{-}\mathsf{LF}^{\text{int}}_{\kappa}[\Sigma;\mathcal{R}]$, we can build in polynomial time an equivalent formula $\varphi' \in r'\text{-}\mathsf{LF}^{\text{int}}_{\kappa}[\Sigma;\mathcal{R}]$. That is, for any structure $\mathfrak{A} \in \mathrm{Str}(\Sigma;\kappa\mathbb{DMS})$ we have $\mathfrak{A} \models \varphi$ iff $\mathfrak{A} \models \varphi'$.*

Proof: In φ, for each $\langle\!\langle \psi \rangle\!\rangle^{r,\text{int}}_{x}$, construct ψ'' as right before the Lemma 3.2.7 and then replace the local modality $\langle\!\langle \psi \rangle\!\rangle^{r,\text{int}}_{x}$ with $\langle\!\langle \psi'' \rangle\!\rangle^{r',\text{int}}_{x}$ and this yields φ'. Then, the proof of the Proposition is similar to the one of Proposition 3.2.8, and the equivalent of the lemma 3.2.7 is that for any structure $\mathfrak{A} \in \mathrm{Str}(\Sigma;\kappa\mathbb{DMS})$ and any context $I : \mathcal{V} \to A$, we have:

$$\mathfrak{A} \models_I \langle\!\langle \psi \rangle\!\rangle^{r,\text{int}}_{x} \text{ iff } \mathfrak{A} \models_I \langle\!\langle \psi'' \rangle\!\rangle^{r',\text{int}}_{x}. \qquad \square$$

Remark 3.2.12. Propositions 3.2.10 and 3.2.11 show that $r\text{-}\mathsf{LF}^{\wp}_{\kappa}[\Sigma;\mathcal{R}]$ is included in $r'\text{-}\mathsf{LF}^{\wp}_{\kappa}[\Sigma;\mathcal{R}]$ with respect to expressiveness when $r < r'$. We think that the inclusion is strict and the formula which states that for any a there is a b at distance between $r+1$ and r' is a witness of it. The formula is written as $\forall x.\langle\!\langle \exists y.\varphi^{\leq r'}_{1,1,\mathcal{R}}(x,y) \wedge \neg\varphi^{\leq r}_{1,1,\mathcal{R}}(x,y) \rangle\!\rangle^{r',\wp}_{x}$. We claim that this formula cannot be expressed in $r\text{-}\mathsf{LF}^{\wp}_{\kappa}[\Sigma;\mathcal{R}]$, even when extending the set of unary predicates Σ. Yet for the time being we lack the tools to prove such a claim. $\diamond$

And finally we state that the local fragment with exterior contains the local fragment without exterior.

Proposition 3.2.13. *For any $\varphi \in r\text{-}\mathsf{LF}^{\text{int}}_{\kappa}[\Sigma;\mathcal{R}]$, we can build in polynomial time an equivalent formula $\varphi' \in r\text{-}\mathsf{LF}^{\text{ext}}_{\kappa}[\Sigma;\mathcal{R}]$. That is, for any structure $\mathfrak{A} \in \mathrm{Str}(\Sigma;\kappa\mathbb{DMS})$ we have $\mathfrak{A} \models \varphi$ iff $\mathfrak{A} \models \varphi'$.*

Proof: In φ, for each $\langle\!\langle \psi \rangle\!\rangle^{r,\text{int}}_{x}$, construct ψ'' as above and then replace the local modality with $\langle\!\langle \psi'' \rangle\!\rangle^{r,\text{ext}}_{x}$ and this yields φ'. Then, the proof of the Proposition is similar to the one of Proposition 3.2.5, and the equivalent of the lemma 3.2.4 is that for any structure $\mathfrak{A} \in \mathrm{Str}(\Sigma;\kappa\mathbb{DMS})$ and any context $I : \mathcal{V} \to A$, we have:

$$\mathfrak{A} \models_I \langle\!\langle \psi \rangle\!\rangle^{r,\text{int}}_{x} \text{ iff } \mathfrak{A} \models_I \langle\!\langle \psi'' \rangle\!\rangle^{r,\text{ext}}_{x}. \qquad \square$$

3.2.2 Relationship with other logics

In this this subsection, we first show in Proposition 3.2.17 that the two-variable fragment $\mathsf{FO}^2_{\kappa\mathbb{DMS}}[\Sigma;\mathcal{R}]$ is a fragment of $1\text{-}\mathsf{LF}^{\text{ext}}_{\kappa}[\Sigma;\mathcal{R}]$, up the addition of new unary predicates. Then we define the guarded fragment $\mathsf{FO}^{g}_{\kappa\mathbb{DMS}}[\Sigma;\mathcal{R}]$ and show in Proposition 3.2.22 that it is a fragment of $1\text{-}\mathsf{LF}^{\text{int}}_{\kappa}[\Sigma;\mathcal{R}]$, up the addition of new unary predicates.

Relationship with the two-variables fragment

Here, we study the relationship of the local fragment with the two-variables fragment. We show that, up to the addition of new unary predicates, $\mathsf{FO}^2_{\kappa\mathbb{DMS}}[\Sigma;\mathcal{R}]$ is included in $1\text{-}\mathsf{LF}^{\text{ext}}_{\kappa}[\Sigma;\mathcal{R}]$ and as a corollary the satisfiability problem of the former is reducible to the one of the latter. To accomplish this, we use the Scott normal form which allows us to only consider formulas of quantification depth at most two. We say that a formula of $\mathsf{FO}^2_{\kappa\mathbb{DMS}}[\Sigma;\mathcal{R}]$ is in *Scott normal form* if is of the form:

$$\forall x.\forall y.\psi_0(x,y) \wedge \bigwedge_{p=1}^{m} \forall x.\exists y.\psi_p(x,y) \tag{3.2.14}$$

where the where $\psi_p \in \mathsf{FO}^2_{\kappa\mathbb{DMS}}[\Sigma;\mathcal{R}]$ for $p \in \{0,\dots,m\}$ are quantifier free (i.e. they are a boolean combination of atomic formulas) and have their free variables among $\{x,y\}$. The following Proposition states how every formula on two-variable is related to a formula in Scott normal form.

Proposition 3.2.15 (Scott)**.** *For any sentence $\varphi \in \mathsf{FO}^2_{\kappa\mathbb{DMS}}[\Sigma;\mathcal{R}]$, we can compute in polynomial time an extension of the unary predicates $\Sigma' \supseteq \Sigma$ and a formula $\varphi' \in \mathsf{FO}^2_{\kappa\mathbb{DMS}}[\Sigma';\mathcal{R}]$ in Scott normal form such that φ is satisfiable iff φ' is satisfiable.*

See [24] for a proof of Proposition 3.2.15 and more details on Scott normal form.

Remark 3.2.16. In Proposition 3.2.15, the models of φ and φ' are tightly related. If $\mathfrak{A} \in \mathrm{Str}(\Sigma;\kappa\mathbb{DMS})$ is a model of φ, there is a unique way to extend $\mathfrak{A}$ by adding the interpretation of every $\sigma \in \Sigma' \setminus \Sigma$ to get $\mathfrak{A}' \in \mathrm{Str}(\Sigma';\kappa\mathbb{DMS})$ a model of φ'. Conversely, if $\mathfrak{A}' \in \mathrm{Str}(\Sigma';\kappa\mathbb{DMS})$ a model of φ', then forgetting the interpretation of $\sigma \in \Sigma' \setminus \Sigma$ yields $\mathfrak{A} \in \mathrm{Str}(\Sigma;\kappa\mathbb{DMS})$, a model of φ. In fact, each predicate in $\Sigma' \setminus \Sigma$ corresponds to a subformula with one free variable of φ and tells where it holds on a structure. $\diamond$

We make the technical assumption that the set of binary predicate $\mathcal{R}$ is *symmetric*, that is for any $i,j \in \{1,\dots,\kappa\}$ if ${}_i{\sim}_j \in \mathcal{R}$ then ${}_j{\sim}_i \in \mathcal{R}$. Recalling that $\mathcal{A}_\kappa = \{{}_i{\sim}_j \mid 1 \le i,j \le \kappa\}$ and $\mathcal{I}_\kappa = \{{}_i{\sim}_j \mid 1 \le i \le j \le \kappa\}$, we make this assumption because $\mathsf{FO}^2_{\kappa\mathbb{DMS}}[\Sigma;\mathcal{I}_\kappa]$ is as expressive as $\mathsf{FO}^2_{\kappa\mathbb{DMS}}[\Sigma;\mathcal{A}_\kappa]$ while $r\text{-}\mathsf{LF}^{\wp}_{\kappa}[\Sigma;\mathcal{I}_\kappa]$ is strictly less expressive than $r\text{-}\mathsf{LF}^{\wp}_{\kappa}[\Sigma;\mathcal{A}_\kappa]$

Proposition 3.2.17. *If the set of binary predicate $\mathcal{R}$ is symmetric, for any sentence $\varphi \in \mathsf{FO}^2_{\kappa\mathbb{DMS}}[\Sigma;\mathcal{R}]$, we can compute in polynomial time an extension of the unary predicates $\Sigma' \supseteq \Sigma$ and a formula $\varphi' \in 1\text{-}\mathsf{LF}^{\text{ext}}_{\kappa\mathbb{DMS}}[\Sigma';\mathcal{R}]$ such that φ is satisfiable iff φ' is satisfiable.*

Proof: From φ, we apply Proposition 3.2.15 to get $\varphi'' \in \mathsf{FO}^2_{\kappa\mathbb{DMS}}[\Sigma'; \mathcal{R}]$ in Scott normal form. We therefore assume that φ'' has the shape described by equation (3.2.14). Then in φ'' we translate the subformula $\forall y.\psi_0(x, y)$ into $\langle\!\langle \forall y.\psi_0(x, y)\rangle\!\rangle^{1,\text{ext}}_x$ and for $p > 0$ we translate the subformula $\exists y.\psi_p(x, y)$ into $\langle\!\langle \exists y.\psi_p(x, y)\rangle\!\rangle^{1,\text{ext}}_x$ to get $\varphi' \in 1\text{-}\mathsf{LF}^{\text{ext}}_\kappa[\Sigma'; \mathcal{R}]$. Then to show that φ'' and φ' are equivalent, the core of the proof is to show that for any structure $\mathfrak{A} \in \text{Str}(\Sigma'; \kappa\mathbb{DMS})$ and a, b elements of it and ${}_i{\sim}_j \in \mathcal{R}$, we have

$$\mathfrak{A} \models a\,{}_i{\sim}_j\,b \;\text{ iff }\; \mathfrak{A}|^{1,\text{ext}}_{a,\mathcal{R}} \models a\,{}_i{\sim}_j\,b. \qquad \square$$

Remark 3.2.18. Remark 3.2.16 applies to Proposition 3.2.17 too. ◇

As a corollary of the previous Proposition, we have the reduction of the satisfiability problem of the two-variable fragment to the local fragment with exterior and radius one.

Theorem 3.2.19

There is a reduction from $\kappa\mathbb{DMS}\text{-}\textsc{Sat}(\mathsf{FO}^2; \mathcal{R})$ *to* $\kappa\mathbb{DMS}\text{-}\textsc{Sat}(1\text{-}\mathsf{LF}^{\text{ext}}; \mathcal{R})$. *Furthermore the reduction is in polynomial time .*

It turns out that the reciprocal of the previous Theorem is also true. The reciprocal being that we can reduce the satisfiability of the local fragment with radius one to the satisfiability of two variable first order logic. But this reduction is far from trivial and will be the whole subject of chapter 4.

The previous Theorem also allows us to shows that as soon as the number of data value is greater than 3 (i.e. $\kappa \geq 3$), the satisfiability problem for the local fragments are undecidable, as stated in the next Corollary. We recall that $\mathcal{S}_3 = \{{}_1{\sim}_1, {}_2{\sim}_2, {}_3{\sim}_3\}$.

Corollary 3.2.20. *The problem* $3\mathbb{DMS}\text{-}\textsc{Sat}(1\text{-}\mathsf{LF}^{\text{ext}}; \mathcal{S}_3)$ *is undecidable.*

Proof: Theorem 3.2.19 gives us a reduction from the problem $3\mathbb{DMS}\text{-}\textsc{Sat}(\mathsf{FO}^2; \mathcal{S}_3)$ to the problem $3\mathbb{DMS}\text{-}\textsc{Sat}(1\text{-}\mathsf{LF}^{\text{int}}; \mathcal{S}_3)$. And thanks to Theorem 2.3.34, we know that the former is undecidable. $\square$

Relationship with the guarded fragment

Here, we study the relationship of the local fragment with the two-variable fragment. The guarded fragment was originally introduced in [2]. We call $\mathsf{FO}^g_{\kappa\mathbb{DMS}}[\Sigma; \mathcal{R}]$ the *guarded fragment* and as in our context the predicates are of arity at most two, we see it as a fragment of two-variable first order logic $\mathsf{FO}^2_{\kappa\mathbb{DMS}}[\Sigma; \mathcal{R}]$. Its syntax is the following:

$$\varphi ::= \sigma(x) \;\mid\; \gamma(x, y) \;\mid\; x = y \;\mid\; \varphi \vee \varphi \;\mid\; \neg\varphi \;\mid\; \exists x.\big(\gamma(x, y) \;\wedge\; \varphi(x, y)\big) \;\mid\; \exists x.\varphi(x),$$

where x and y range over $\mathcal{V}$, σ ranges over Σ, γ ranges over $\mathcal{R}$ and recalling that the notation $\varphi(x, y)$ means that the free variables of φ are among $\{x, y\}$.

We show that in a sense, $\mathsf{FO}^g_{\kappa\mathbb{DMS}}[\Sigma; \mathcal{R}]$ is included in $1\text{-}\mathsf{LF}^{\text{int}}_\kappa[\Sigma; \mathcal{R}]$ and as a corollary the satisfiability problem of the former is reducible to the one of the latter. To accomplish this, we use the guarded Scott normal form which is a variation of the Scott normal form. We say that a formula of $\mathsf{FO}^g_{\kappa\mathbb{DMS}}[\Sigma; \mathcal{R}]$ is in *guarded Scott normal form* if it is of the form:

$$\forall x.\forall y.\left(\Big(\bigvee_{\gamma\in\mathcal{R}}\gamma(x,y)\Big)\rightarrow\psi_0\right)\wedge\bigwedge_{p=1}^{m}\forall x.\exists y.\left(\Big(\bigvee_{\gamma\in\mathcal{R}}\gamma(x,y)\Big)\wedge\psi_p\right)\tag{3.2.21}$$

where the $\psi_p\in\mathsf{FO}^2_{\kappa\mathbb{DMS}}[\Sigma;\mathcal{R}]$ for $p\in\{0,\dots,m\}$ are quantifier free (i.e. they are a boolean combination of atomic formulas) and have their free variables among $\{x,y\}$. By slightly adapting the proof of the existence of the Scott normal form of [24], one can prove that any formula of the guarded fragment is equivalent to a formula in guarded Scott normal form:

Proposition 3.2.22. *For any sentence $\varphi\in\mathsf{FO}^g_{\kappa\mathbb{DMS}}[\Sigma;\mathcal{R}]$, we can compute in polynomial time an extension of the unary predicates $\Sigma'\supseteq\Sigma$ and a formula $\varphi'\in\mathsf{FO}^g_{\kappa\mathbb{DMS}}[\Sigma';\mathcal{R}]$ in guarded Scott normal form such that φ is satisfiable iff φ' is satisfiable.*

Remark 3.2.23. Remark 3.2.16 applies to Proposition 3.2.22 too. ⋄

The next proposition and theorem state the reduction from the guarded fragment to the local fragment without exterior.

Proposition 3.2.24. *If the set of binary predicates $\mathcal{R}$ is symmetric, for any sentence $\varphi\in\mathsf{FO}^g_{\kappa\mathbb{DMS}}[\Sigma;\mathcal{R}]$, we can compute in polynomial time an extension of the unary predicates $\Sigma'\supseteq\Sigma$ and a formula $\varphi'\in 1\text{-}\mathsf{LF}^{\mathrm{int}}_{\kappa\mathbb{DMS}}[\Sigma;\mathcal{R}]$ such that φ is satisfiable iff φ' is satisfiable.*

Proof: From φ, we apply Proposition 3.2.22 to get $\varphi''\in\mathsf{FO}^2_{\kappa\mathbb{DMS}}[\Sigma';\mathcal{R}]$ in Scott normal form. We therefore assume that φ'' has the shape describe by equation (3.2.21). Then in φ'' we transform the subformula

$$\Big(\bigvee_{\gamma\in\mathcal{R}}\gamma(x,y)\Big)\rightarrow\psi_0$$

into

$$\langle\!\langle\Big(\bigvee_{\gamma\in\mathcal{R}}\gamma(x,y)\Big)\rightarrow\psi_0\rangle\!\rangle^{1,\mathrm{int}}_x$$

and for $p>0$ we transform the subformula

$$\exists y.\big(\gamma_i(x,y)\wedge\psi_i\big)$$

into

$$\langle\!\langle\exists y.\big(\gamma_i(x,y)\wedge\psi_i\big)\rangle\!\rangle^{1,\mathrm{int}}_x$$

to get $\varphi'\in 1\text{-}\mathsf{LF}^{\mathrm{int}}_{\kappa}[\Sigma';\mathcal{R}]$. Then to show that φ'' and φ' are equivalent, the two key step of the proof is first to notice that for any structure $\mathfrak{A}\in\mathrm{Str}(\Sigma';\kappa\mathbb{DMS})$ and if a,b are elements of $\mathfrak{A}$, then b is an element of $\mathfrak{A}|^{1,\mathrm{int}}_{a,\mathcal{R}}$ iff there is a $\gamma\in\mathcal{R}$ such that $\mathfrak{A}\models\gamma(a,b)$. The second step is to notice that for any element a of $\mathfrak{A}$ and b of $\mathfrak{A}|^{1,\mathrm{int}}_{a,\mathcal{R}}$ we have

$$\mathfrak{A}\models a\,{}_i{\sim}_j\,b\ \text{ iff }\ \mathfrak{A}|^{1,\mathrm{int}}_{a,\mathcal{R}}\models a\,{}_i{\sim}_j\,b.$$ □

Remark 3.2.25. The remark 3.2.16 applies to the previous Proposition too. ⋄

As a corollary of the previous Proposition, we have the reduction of the satisfiability problem of the guarded fragment to the local fragment without exterior and radius one.

Theorem 3.2.26

There is a reduction from $\kappa\mathbb{DMS}$-$\textsc{Sat}(\mathsf{FO}^g;\mathcal{R})$ to $\kappa\mathbb{DMS}$-$\textsc{Sat}(1\text{-}\mathsf{LF}^{\mathrm{int}};\mathcal{R})$. Furthermore the reduction is in polynomial time.

There is a usual extension of the guarded fragment, called equally the *loosely* guarded fragment or the *pairwise* guarded fragment. It is defined in [51]. To determine its relationship with the local fragments is a question less obvious than for the guarded fragment and remains open.

3.2.3 Links with Gaifman graph

The data graph $\mathcal{G}(\mathfrak{A})$ is closely related to the Gaifman graph of $\mathfrak{A}$. As previously said, the notion of data graph $\mathcal{G}(\mathfrak{A})$ is inspired from the notion of Gaifman graph. But their links are stronger; we prove in this subsection formal links between the two. Succinctly, the Gaifman graph of $\mathfrak{A}$ is the graph with set of vertices A and two vertex are connected if one of their data value are equal, that is set set of edges is $\cup_{i,j} R^{\mathfrak{A}}_{i\sim j}$, the union of the interpretations of the data comparison relations. We then have a morphism of graph from the data graph to the Gaifman graph which sends the data location (a,i) to its corresponding element a. Moreover, this morphism satisfies that if a and b are connected in the Gaifman graph, then there are i,j such that (a,i) and (b,j) are connected in the data graph. We made use of this connection to prove Proposition 3.2.27. We then unfold some consequences of this Proposition. Among them, Corollary 3.2.28 is crucial for the proof of decidability of the existential fragment in section 5.

Here for the sake of simplicity, we will assume that $\mathcal{R} = \mathcal{A}_\kappa$, that is we can compare the values of any data location. That is, we do note have to bother to check if a data comparison is in $\mathcal{R}$. One can easily generalise to any set $\mathcal{R}$ and it would not affect the aim of this subsection. The idea of the following proposition is to characterize the existence of a path in the data graph as the existence of a path in the Gaifman's graph.

Proposition 3.2.27 (Characterization of balls)**.** *Let $r \in \mathbb{N}^*$ and $\mathfrak{A} = (A, (P_\sigma)_\sigma, f_1, \ldots, f_\kappa) \in \mathrm{Str}(\Sigma;\kappa\mathbb{DMS})$ and $a, b \in A$ and $j \in \{1,\ldots,\kappa\}$. Let p be an integer such that $r = 2p-1$ if r is odd and $r = 2p$ otherwise.*

Then the data location (b,j) is in the ball $B^{\mathfrak{A}}_{r,\mathcal{R}}(a)$ iff there are $a_0 = a, a_1, \ldots, a_p = b \in A$ and $i_0, j_1, i_1, j_2, \ldots, i_{p-2}, j_{p-1}, i_{p-1}, j_p \in \{1,\ldots,\kappa\}$ such that

$$a_0 \;{}_{i_0}{\sim}^{\mathfrak{A}}_{j_1}\; a_1 \;{}_{i_1}{\sim}^{\mathfrak{A}}_{j_2}\; a_2 \;{}_{i_2}{\sim}^{\mathfrak{A}}_{j_3} \quad \cdots \quad {}_{i_{p-2}}{\sim}^{\mathfrak{A}}_{j_{p-1}}\; a_{p-1} \;{}_{i_{p-1}}{\sim}^{\mathfrak{A}}_{j_p}\; a_p.$$

If r is odd, we further ask for $j_p = j$.

Proof: We treat at the same time the case r odd and r even.

The backward direction is obvious: by induction we have that $(a_0, i_0) \in B^{\mathfrak{A}}_{0,\mathcal{R}}(a)$, then that $(a_1, j_1) \in B^{\mathfrak{A}}_{1,\mathcal{R}}(a)$ then that $(a_1, i_1) \in B^{\mathfrak{A}}_{2,\mathcal{R}}(a)$ and then $(a_2, j_2) \in B^{\mathfrak{A}}_{3,\mathcal{R}}(a)$ and so

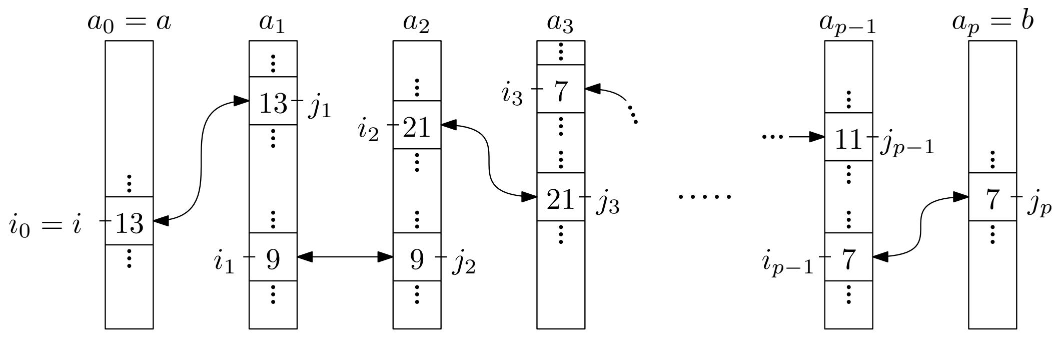


Figure 3.7: An illustration of the characterization of Proposition 3.2.27.

on. At the end we have $(a_p, j_p) \in B^{\mathfrak{A}}_{2p-1,\mathcal{R}}(a)$ and $(a_p, j) \in B^{\mathfrak{A}}_{2p,\mathcal{R}}(a)$ (note that whether r is even or odd does not affect the truth of the last statement). If r is odd, we then have $(a_p, j_p) = (b, j)$.

For the forward direction, let us assume that $(b, j) \in B^{\mathfrak{A}}_{r,\mathcal{R}}(a)$ then by definition of a ball, it means that there is a path of length at most r in the data graph $\mathcal{G}(\mathfrak{A})$ from a data location of a to the jth data location of b. Let l denotes the minimal length of such a path. So there are $(c_0, k_0), \dots, (c_l, k_l) \in A \times \{1, \dots, \kappa\}$ with $c_0 = a, c_l = b$ and $k_l = j$ such that for any $0 \leq n < l$, the data locations (c_n, k_n) and (c_{n+1}, k_{n+1}) are adjacent in $\mathcal{G}(\mathfrak{A})$. We show by induction on $0 \leq n < l$ that:

$$(*) \begin{cases} c_n \,{}_{k_n}\!\sim^{\mathfrak{A}}_{k_{n+1}} c_{n+1} & \text{if } n \text{ is even} \\ c_n = c_{n+1} & \text{if } n \text{ is odd} \end{cases}$$

For the base case $n = 0$, we have to show that $c_0 \neq c_1$. It is true because otherwise, we have $a = c_0 = c_1$ and then we have a path $(c_1, k_1) \cdots (c_l, k_l)$ of length $l - 1$ contradicting the minimality of l. We now treat the inductive case, so $n \geq 1$.

First, let assume that n is even. We want to show that $c_n \,{}_{k_n}\!\sim^{\mathfrak{A}}_{k_{n+1}} c_{n+1}$. We have that $n - 1$ is odd and by inductive hypothesis, we have $c_{n-1} = c_n$. Then $c_n \neq c_{n+1}$, because otherwise $c_{n-1} = c_n = c_{n+1}$ and then the data locations (c_{n-1}, k_{n-1}) and (c_{n+1}, k_{n+1}) are adjacent in $\mathcal{G}(\mathfrak{A})$ implying we can remove (c_n, k_n) from the path to get the new path $(c_0, k_0) \cdots (c_{n-1}, k_{n-1})(c_{n+1}, k_{n+1}) \cdots (c_l, k_l)$ of length $l - 1$ which contradicts the minimality of l. But (c_n, k_n) and (c_{n+1}, k_{n+1}) are adjacent in $\mathcal{G}(\mathfrak{A})$, thus $c_n \,{}_{k_n}\!\sim^{\mathfrak{A}}_{k_{n+1}} c_{n+1}$ as desired.

Second, let assume that n is odd. We want to show that $c_n = c_{n+1}$. We have that $n - 1$ is even and by inductive hypothesis, we have $c_{n-1} \,{}_{k_{n-1}}\!\sim^{\mathfrak{A}}_{k_n} c_n$. Then we don't have $c_n \,{}_{k_n}\!\sim^{\mathfrak{A}}_{k_{n+1}} c_{n+1}$, because otherwise it implies that $c_{n-1} \,{}_{k_{n-1}}\!\sim^{\mathfrak{A}}_{k_{n+1}} c_{n+1}$ and then the data locations (c_{n-1}, k_{n-1}) and (c_{n+1}, k_{n+1}) are adjacent in $\mathcal{G}(\mathfrak{A})$ implying we can remove (c_n, k_n) from the path to get the new path $(c_0, k_0) \cdots (c_{n-1}, k_{n-1})(c_{n+1}, k_{n+1}) \cdots (c_l, k_l)$ of length $l - 1$ which contradicts the minimality of l. But (c_n, k_n) and (c_{n+1}, k_{n+1}) are adjacent in $\mathcal{G}(\mathfrak{A})$, thus $c_n = c_{n+1}$ as desired.

We now have finished the inductive proof. We can pad the end the path with (b, j) to move from a path of length l to a path of length r and still have $(*)$ holding for it. We remind ourselves that we have p be such that $r = 2p - 1$ if r is odd and $r = 2p$ else. Now we define the sequence of a_ns, i_ns and j_ns by

$$\begin{cases} a_n = c_{2n} & \text{for } \ n \in \{0, \dots, p-1\}, \\ a_p = c_{2n-1} & \\ i_n = k_{2n} & \text{for } \ n \in \{0, \dots, p-1\}, \\ j_n = k_{2n-1} & \text{for } \ n \in \{1, \dots, p\}. \end{cases}$$

By the property $(*)$ we have that

$$a_0 \; {}_{i_0}{\sim}^{\mathfrak{A}}_{j_1} \; a_1 \; {}_{i_1}{\sim}^{\mathfrak{A}}_{j_2} \; a_2 \; {}_{i_2}{\sim}^{\mathfrak{A}}_{j_3} \quad \cdots \quad {}_{i_{p-2}}{\sim}^{\mathfrak{A}}_{j_{p-1}} \; a_{p-1} \; {}_{i_{p-1}}{\sim}^{\mathfrak{A}}_{j_p} \; a_p,$$

with $a_0 = a$ and $a_p = b$. Moreover when $r = 2p - 1$, we have $j_p = k_{2p-1} = k_r = j$. This concludes the proof. □

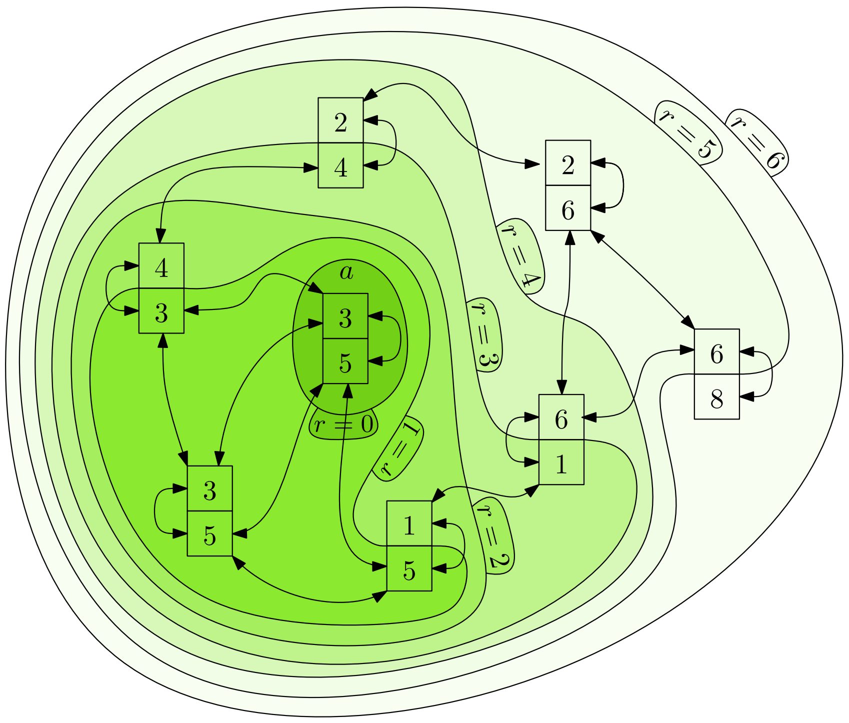


Figure 3.8: A 2-data-multiset and the balls around the element a for all the value of r.

Specifying Proposition 3.2.27 for $r \in \{1, 2, 3, 4\}$ we get the following Corollary.

Corollary 3.2.28. *Let $r \in \mathbb{N}^*$ and $\mathfrak{A} = (A, (P_\sigma)_\sigma, f_1, \dots, f_\kappa) \in \mathrm{Str}(\Sigma; \kappa\mathbb{DMS})$ and $a, b \in A$ and $j \in \{1, \dots, \kappa\}$.*

1. *$(b, j) \in B^{\mathfrak{A}}_{1,\mathcal{R}}(a)$ iff there is $i \in \{1, \dots, \kappa\}$ such that $a \, {}_i{\sim}^{\mathfrak{A}}_j \, b$.*

2. *$(b, j) \in B^{\mathfrak{A}}_{2,\mathcal{R}}(a)$ iff there exists $i, j' \in \{1, \dots, \kappa\}$ such that $a\,{}_i{\sim}^{\mathfrak{A}}_{j'}\, b$.*
3. *$(b, j) \in B^{\mathfrak{A}}_{3,\mathcal{R}}(a)$ iff there exists $i_0, j_1, i_1 \in \{1, \dots, \kappa\}$ and $c \in A$ such that*
$$a\ {}_{i_0}{\sim}^{\mathfrak{A}}_{j_1}\ c\ {}_{i_1}{\sim}^{\mathfrak{A}}_{j}\ b.$$
4. *$(b, j) \in B^{\mathfrak{A}}_{4,\mathcal{R}}(a)$ iff there exists $i_0, j_1, i_1, j_2 \in \{1, \dots, \kappa\}$ and $c \in A$ such that*
$$a\ {}_{i_0}{\sim}^{\mathfrak{A}}_{j_1}\ c\ {}_{i_1}{\sim}^{\mathfrak{A}}_{j_2}\ b.$$

The following remark establish the link between distances in the data graph and distances in the Gaifman graph.

Remark 3.2.29 (On distances). Let us consider the function from integers to integers $n \mapsto \lceil n/2 \rceil$, which for every $p \in \mathbb{N}$ sends both $2p-1$ and $2p$ to p. Let $\mathfrak{A} = (A, (P_\sigma)_\sigma, f_1, \dots, f_\kappa) \in \mathrm{Str}(\Sigma; \kappa\mathbb{DMS})$ and $a \in A$ and $r > 0$. A consequence of Proposition 3.2.27 is that the radius r ball in the Gaifman graph of $\mathfrak{A}$ is the same as the radius $2r$ ball in the data graph. ◇

Remark 3.2.30. Let $\mathfrak{A} = (A, (P_\sigma)_\sigma, f_1, \dots, f_\kappa) \in \mathrm{Str}(\Sigma; \kappa\mathbb{DMS})$ and $a, b \in A$. We look at the distances of the data locations of b from a. That is the set $\{\min_{i \in \{1,\dots,\kappa\}} d^{\mathfrak{A}}(((a, i), (b, j)) \mid j \in \{1, \dots, \kappa\}\}$. A direct consequence of the definition of the data graph is that the difference of two distances is at most one, and thus the set contains at most two values. Proposition 3.2.27 adds that the minimum distance is always odd and if the set is a singleton, the only distance is odd. Furthermore, if r is even then for any $j, j' \in \{1, \dots, \kappa\}$, we have that $(b, j) \in B^{\mathfrak{A}}_{r,\mathcal{R}}(a)$ iff $(b, j') \in B^{\mathfrak{A}}_{r,\mathcal{R}}(a)$. ◇

Chapter 4

Deciding the satisfiability of the local fragments

This chapter presents the work of [10]. It is dedicated to set the nature of the satisfiability problems of the local fragment. We only focus on the cases with two data values because if the number of data values is smaller or bigger, the satisfiability problems for the local fragment become trivial relative to the existing literature (see Corollaries 3.2.9 and 3.2.20). This chapter only talks about the local fragment without exterior as the article this chapter come from.

The first section establish a preliminary results that will be reused in the following section. Then the second section shows that $2\mathbb{DMS}\text{-}\textsc{Sat}(1\text{-}\mathsf{LF}^{\text{int}}; \{{}_1{\sim}_1, {}_2{\sim}_2, {}_1{\sim}_2\})$ is decidable. Finally, in the third section is dedicated to negative results, we show that the two problems $2\mathbb{DMS}\text{-}\textsc{Sat}(3\text{-}\mathsf{LF}^{\text{int}}; \{{}_1{\sim}_1, {}_2{\sim}_2\})$ and $2\mathbb{DMS}\text{-}\textsc{Sat}(2\text{-}\mathsf{LF}^{\text{int}}; \{{}_1{\sim}_1, {}_2{\sim}_2, {}_1{\sim}_2\})$ are undecidable.

From [10] there are two modifications; the first one is a slight change in the notation in order to fit the ones of this manuscript, and the second modification is that definitions and preliminary results have moved to Chapter 2.

4.1 Preliminaries: Extended Two-Variable Fragment

In this section, we will define the extended two-variable first-order logic and then show that the corresponding satisfiability problem is decidable. It extends two already existing results.

We recall that $\mathcal{S}_2 = \{{}_1{\sim}_1, {}_2{\sim}_2\}$. The first result is about FO^2. The two-variable fragment $\mathsf{FO}^2_{2\mathbb{DMS}}[\Sigma; \mathcal{S}_2]$ contains all $\mathsf{FO}[\Sigma; \mathcal{S}_2]$ formulas that use only two variables (usually x and y). In a two-variable formula, however, each of the two variables can be used arbitrarily often. We restate a theorem seen in Section 2.

Theorem 2.3.32 (*[32]*)

The problem $2\mathbb{DMS}\text{-}\textsc{Sat}(\mathsf{FO}^2; \mathcal{S}_2)$ *is* N2$\textsc{Exp}$*-complete.*

The second result is about first-order logic with only unary predicates.

Theorem 4.1.1

The problem $2\mathbb{DMS}\text{-}\textsc{Sat}(\mathsf{FO};\emptyset)$ *is* $\text{N}\textsc{Exp}$*-complete.*

Proof: As the set of binary symbols $\mathcal{R}$ is empty, that is $\mathcal{R}=\emptyset$, a formula $\varphi \in \mathsf{FO}_{2\mathbb{DMS}}[\Sigma;\emptyset]$ can be seen as formula about multisets without data values, that is $\varphi \in \mathsf{FO}_{\mathbb{MS}}[\Sigma;\emptyset]$. And Theorem 2.3.9 states that the problem $\mathbb{MS}\text{-}\textsc{Sat}(\mathsf{FO};\emptyset)$ is $\text{N}\textsc{Exp}$-complete. $\square$

Actually, those two results can be generalized to *extended* two-variable first-order logic. A formula belongs to ext-$\mathsf{FO}^2_{2\mathbb{DMS}}[\Sigma;\mathcal{R}]$ if it is of the form $\varphi \wedge \psi$ where $\varphi \in \mathsf{FO}_{2\mathbb{DMS}}[\Sigma;\emptyset]$ and $\psi \in \mathsf{FO}^2_{2\mathbb{DMS}}[\Sigma;\mathcal{R}]$. To obtain the next result, the idea consists in first translating the formula $\varphi \in \mathsf{FO}_{2\mathbb{DMS}}[\Sigma;\emptyset]$ into a two-variable formula thanks to new unary predicates.

Proposition 4.1.2. *The problem* $2\mathbb{DMS}\text{-}\textsc{Sat}(\text{ext-}\mathsf{FO}^2;\mathcal{S}_2)$ *is decidable.*

In order to prove Proposition 4.1.2, we first show that one can reduce the first-order part with $\mathcal{R}=\emptyset$ to a two-variable formula:

Proposition 4.1.3. *Let φ be an $\mathsf{FO}_{2\mathbb{DMS}}[\Sigma;\emptyset]$ sentence. Then, we can effectively construct $\varphi' \in \mathsf{FO}^2_{2\mathbb{DMS}}[\Sigma';\emptyset]$ with $\Sigma \subseteq \Sigma'$ such that φ is satisfiable iff φ' is satisfiable. Furthermore, if a structure $\mathfrak{A}$ satisfies φ, then we can add an interpretation of the predicates in $\Sigma' \setminus \Sigma$ to $\mathfrak{A}$ to get a model for φ'. Conversely, if a structure $\mathfrak{A}'$ satisfies φ', then forgetting the interpretation of the predicates in $\Sigma' \setminus \Sigma$ in $\mathfrak{A}'$ gives us a model for φ.*

Proof: We apply Lemma 2.3.10 to φ and then obtain φ''. As there is no free variable in φ, the formula φ'' is a boolean combination of formulas of the form $\exists^{\geq k} y.\varphi_U(y)$ where $U \subseteq \Sigma$. Let M be the maximal such k (if there is no threshold formula, φ'' is either *true* or *false*). We define a set of unary predicates $\Lambda_M = \{\eta_i \mid 1 \leq i \leq M\}$ and let $\Sigma' = \Sigma \cup \Lambda_M$. The following formulas will specify the meaning of the elements of Λ_M. First, let $\varphi_{same}(x,y) = \bigwedge_{\sigma\in\Sigma} \sigma(x) \leftrightarrow \sigma(y)$. With this, we define:

$$\begin{aligned}
\varphi^1_\eta &:= \forall x. \bigvee_{i\in[1,M]} \Big(\eta_i(x) \wedge \bigwedge_{j\in[1,M]\setminus\{i\}} \neg\eta_j(x)\Big) \\
\varphi^2_\eta &:= \forall x. \bigwedge_{i\in[1,M-1]} \big(\eta_i(x) \rightarrow \neg\exists y.(x \neq y \wedge \varphi_{same}(x,y) \wedge \eta_i(y))\big) \\
\varphi^3_\eta &:= \forall x. \bigwedge_{i\in[2,M]} \big(\eta_i(x) \rightarrow (\exists y.\varphi_{same}(x,y) \wedge \eta_{i-1}(y))\big)
\end{aligned}$$

We then denote $\varphi_\eta := \varphi^1_\eta \wedge \varphi^2_\eta \wedge \varphi^3_\eta \in \mathsf{FO}^2_{2\mathbb{DMS}}[\Sigma';\emptyset]$. Then, for a model $\mathfrak{A} \in \mathrm{Str}(\Sigma';2\mathbb{DMS})$ of φ_η with carrier set A, an element $a \in A$, and an integer $1 \leq i \leq M$, we have that $a \in P_{\eta_i}$ iff $|\{b \in A \mid \text{for all } \sigma \in \Sigma,\ a \in P_\sigma \text{ iff } b \in P_\sigma\}| \geq i$. Then in φ'', we replace all threshold formulas $\exists^{\geq k} y.\varphi_U(y)$ with $\exists y.\varphi_U(y) \wedge \eta_k(y)$ in order to obtain $\varphi''' \in \mathsf{FO}^2_{2\mathbb{DMS}}[\Sigma \cup \Lambda_M;\emptyset]$. Finally we take φ' as $\varphi''' \wedge \varphi_\eta$. $\square$

We are now ready to prove Proposition 4.1.2:

Proof of Proposition 4.1.2: Let $\varphi \wedge \psi$ be a sentence such that $\varphi \in \mathsf{FO}_{2\mathbb{DMS}}[\Sigma;\emptyset]$ and $\psi \in \mathsf{FO}^2_{2\mathbb{DMS}}[\Sigma;\mathcal{R}]$. We determine $\Sigma' \supseteq \Sigma$ and φ' in $\mathsf{FO}^2_{2\mathbb{DMS}}[\Sigma';\emptyset]$ according to Proposition 4.1.3. Then, by Theorem 2.3.32, it only remains to show that $\varphi \wedge \psi$ is satisfiable iff $\varphi' \wedge \psi$ is satisfiable.

Suppose there is $\mathfrak{A} \in \mathrm{Str}(\Sigma;2\mathbb{DMS})$ such that $\mathfrak{A} \models \varphi \wedge \psi$. By Proposition 4.1.3, we can add propositions from $\Sigma' \setminus \Sigma$ to $\mathfrak{A}$ to get a data structure $\mathfrak{A}'$ such that $\mathfrak{A}' \models \varphi'$. As ψ does not speak about propositions in $\Sigma' \setminus \Sigma$, we have $\mathfrak{A}' \models \psi$ and, therefore, $\mathfrak{A}' \models \varphi' \wedge \psi$.

Conversely, let $\mathfrak{A}' \in \mathrm{Str}(\Sigma';2\mathbb{DMS})$ such that $\mathfrak{A}' \models \varphi' \wedge \psi$. Then, again by Proposition 4.1.3, "forgetting" in $\mathfrak{A}'$ all labels in $\Sigma' \setminus \Sigma$ yields a structure $\mathfrak{A}$ such that $\mathfrak{A} \models \varphi$. As we still have $\mathfrak{A} \models \psi$, we conclude $\mathfrak{A} \models \varphi \wedge \psi$. □

4.2 Decidability With One Diagonal Relation

Recalling that $\mathcal{I}_2 = \{{}_1{\sim}_1, {}_2{\sim}_2, {}_1{\sim}_2\}$ and $\mathcal{S}_2 = \{{}_1{\sim}_1, {}_2{\sim}_2\}$, we will show in this section that $2\mathbb{DMS}$-SAT(1-$\mathsf{LF}^{\mathrm{int}}$; $\mathcal{I}_2$) is decidable. To this end, we will give a reduction to $2\mathbb{DMS}$-SAT(ext-FO^2; $\mathcal{S}_2$). The rest of this section is devoted to this reduction.

Henceforth, we fix a finite set Σ and we let Θ range over arbitrary finite sets such that $\Sigma \subseteq \Theta$ and $\Theta \cap \{\mathtt{eq}, \mathtt{ed}\} = \emptyset$, where $\mathtt{eq}$ and $\mathtt{ed}$ are special unary symbols that are introduced below.

We start with some crucial notion. Suppose $\mathcal{R}' \subseteq \mathcal{I}_2$ (which will later be instantiated by either $\mathcal{S}_2$ or $\mathcal{I}_2$). We define $\widetilde{\mathcal{R}'}$ as the set $\{(i,j) \mid {}_i{\sim}_j \in \mathcal{R}'\}$, then for example we have $\widetilde{\mathcal{I}_2} = \{(1,1),(2,2),(1,2)\}$. Consider a data structure $\mathfrak{A} = (A,(P_\sigma)_\sigma, f_1, f_2) \in \mathrm{Str}(\Theta;2\mathbb{DMS})$ with $\Sigma \subseteq \Theta$. Given $U \subseteq \Sigma$ and a nonempty set $R \subseteq \widetilde{\mathcal{R}'}$, the *environment* of $a \in A$ is defined as

$$\mathtt{Env}_{\mathfrak{A},\Sigma,\mathcal{R}'}(a,U,R) = \big\{b \in A \mid U = \{\sigma \in \Sigma \mid b \in P_\sigma\} \text{ and } R = \{(i,j) \in \widetilde{\mathcal{R}'} \mid a\,{}_i{\sim}_j^{\mathfrak{A}}\, b\}\big\}\,.$$

Thus, it contains the elements that carry exactly the labels from U (relative to Σ) and to which a is related precisely in terms of the relations in R (relative to $\mathcal{R}'$).

Example 4.2.1. Consider $\mathfrak{A} \in \mathrm{Str}(\Sigma;2\mathbb{DMS})$ from Figure 4.1(a) where $\Sigma = \emptyset$. Then, the set $\mathtt{Env}_{\mathfrak{A},\Sigma,\mathcal{I}_2}(a,\emptyset,\{(1,1),(1,2)\}) = \mathtt{Env}_{\mathfrak{A},\Sigma,\mathcal{S}_2}(a,\emptyset,\{(1,1)\})$ contains exactly the yellow elements (with data-value pairs $(1,1)$), and $\mathtt{Env}_{\mathfrak{A},\Sigma,\mathcal{I}_2}(a,\emptyset,\{(1,2)\})$ contains the two blue elements (with data-value pairs $(2,1)$ and $(3,1)$). ◇

Let us now go through the reduction step by step.

Step 1: Transform Binary into Unary Relations

In the first step, we get rid of the binary relations by representing them as unary ones. In fact, in a formula $\langle\!\langle\psi\rangle\!\rangle_x^{1,\mathrm{int}}$ from 1-$\mathsf{LF}_2^{\mathrm{int}}[\Sigma;\mathcal{I}_2]$, ψ only talks about elements that are directly related to $a = I(x)$ in terms of pairs from $\widetilde{\mathcal{I}_2}$. In fact, we can rewrite ψ into ψ' so that all comparisons are wrt. x, i.e., they are of the form $x\;{}_i{\sim}_j\;y$. Then, a pair $(i,j) \in \widetilde{\mathcal{I}_2}$ can be

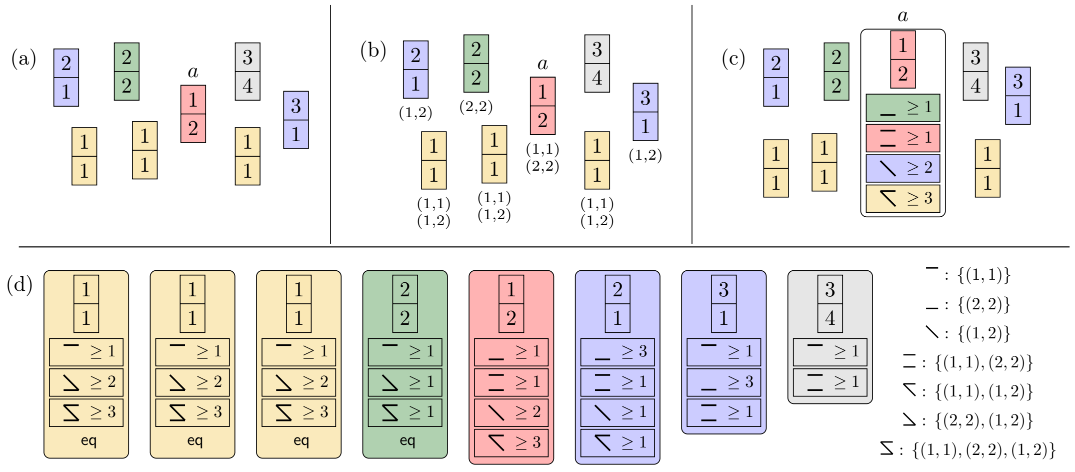


Figure 4.1: (a) A data structure over $\Sigma = \emptyset$. (b) Adding unary predicates for a given element a. (c) Adding counting constraints to a. (d) A well-typed data structure from $\mathrm{Str}(\{\texttt{eq}\} \cup \texttt{C}_3; 2\mathbb{DMS})$.

seen as a unary predicate that holds at b iff $a \ {}_i{\sim}_j\ b$. In this way, we eliminate the binary relations and replace ψ' with a first-order formula ψ'' over unary predicates.

Example 4.2.2. Adding unary relations to a data structure for a given element a is illustrated in Figure 4.1(b) (recall that $\Sigma = \emptyset$). ◇

Thanks to the unary predicates, we can now apply Lemma 2.3.10 (which was a consequence of locality of first-order logic over unary symbols only). That is, to know whether ψ'' holds when x is interpreted as a, it is enough to know how often every unary predicate is present in the environment of a, counted only up to some $M \geq 1$. However, we will then give up the information of whether the two data values at a coincide or not. Therefore, we introduce a unary predicate $\texttt{eq}$, which shall label those events whose two data values coincide. Accordingly, we say that $\mathfrak{A} = (A, (P_\sigma)_\sigma, f_1, f_2) \in \mathrm{Str}(\Theta \cup \{\texttt{eq}\}; 2\mathbb{DMS})$ is *eq-respecting* if, for all $a \in A$, we have $a \in P_{\texttt{eq}}$ iff $f_1(a) = f_2(a)$.

Once we add this information to a, it is enough to know the size of $\texttt{Env}_{\mathfrak{A},\Sigma,\mathcal{I}_2}(a, U, R)$ for every $U \subseteq \Sigma$ and nonempty $R \subseteq \widetilde{\mathcal{I}_2}$, measured up to M. To reason about theses sizes, we introduce a unary predicate $\wr U, R, m \int$ for all $U \subseteq \Sigma$, nonempty sets $R \subseteq \widetilde{\mathcal{I}_2}$, and $m \in \{1, \ldots, M\}$ (which is interpreted as "$\geq m$"). We also call such a predicate a *counting constraint* and denote the set of all counting constraints by $\texttt{C}_M$ (recall that we fixed Σ and $\mathcal{R} = \mathcal{I}_2$). For a finite set Θ with $\Sigma \subseteq \Theta$, we call $\mathfrak{A} = (A, (P_\sigma)_\sigma, f_1, f_2) \in \mathrm{Str}(\Theta \cup \texttt{C}_M; 2\mathbb{DMS})$ *cc-respecting* if, for all $a \in A$, we have $a \in P_{\wr U,R,m \int}$ iff $|\texttt{Env}_{\mathfrak{A},\Sigma,\mathcal{I}_2}(a, U, R)| \geq m$.

Finally, we call $\mathfrak{A} \in \mathrm{Str}(\Theta \cup \{\texttt{eq}\} \cup \texttt{C}_M; 2\mathbb{DMS})$ *well-typed* if it is eq-respecting and cc-respecting.

Example 4.2.3. In Figure 4.1(c), where we suppose $M = 3$ and $\Sigma = \emptyset$, the element a satisfies the counting constraints $\wr \emptyset, \{(2,2)\}, 1 \int$, $\wr \emptyset, \{(1,1),(2,2)\}, 1 \int$, $\wr \emptyset, \{(1,2)\}, 2 \int$, and $\wr \emptyset, \{(1,1),(1,2)\}, 3 \int$, as well as all inherited constraints for smaller constants (which we omitted). On this figure, we write $\wr \emptyset, R, m \int$ as $R \geq m$. In fact, pairs from R are represented as black bars in the obvious way (cf. Figure 4.1(d)); moreover, for each constraint, the corresponding elements have the same color. Finally, the data structure from Figure 4.1(d) is well-typed, i.e., eq- and cc-respecting. Again, we omit inherited constraints. $\diamond$

To summarize, we have the following reduction:

Lemma 4.2.4. *For each formula* $\varphi \in \text{1-}\mathsf{LF}_2^{\text{int}}[\Sigma;\mathcal{I}_2]$, *we can effectively compute* $M \in \mathbb{N}$ *and* $\chi \in \mathsf{FO}_{2\mathbb{DMS}}[\Sigma \cup \{\mathsf{eq}\} \cup \mathsf{C}_M;\emptyset]$ *such that* φ *is satisfiable iff* χ *has a well-typed model.*

Proof: Consider $\langle\!\langle \psi \rangle\!\rangle_x^{r,\text{int}}$ where ψ is a formula from $\mathsf{FO}_{2\mathbb{DMS}}[\Sigma;\mathcal{I}_2]$ with one free variable x. Wlog., we assume that x is not quantified in ψ. We replace, in ψ, every occurence of a formula $y\ {}_i{\sim}_j\ z$ with $y \neq x$ by

$$\bigvee_{\substack{k\in\{1,2\}\ \mid \\ (k,i),(k,j)\in\widetilde{\mathcal{I}_2}}} x\ {}_k{\sim}_i\ y \ \wedge\ x\ {}_k{\sim}_j\ z\,.$$

Call the resulting formula ψ'. Replace, in ψ', every formula $x\ {}_i{\sim}_j\ y$ by $(i,j)(y)$ to obtain an $\mathsf{FO}_{2\mathbb{DMS}}[\Sigma \cup \widetilde{\mathcal{I}_2};\emptyset]$ formula ψ''. Suppose $\mathfrak{A} = (A, (P_\sigma)_{\sigma\in\Sigma\cup\widetilde{\mathcal{I}_2}}, f_1, f_2) \in \text{Str}(\Sigma \cup \widetilde{\mathcal{I}_2};2\mathbb{DMS})$ and interpretation function I such that, for all $b \in A$ and $(i,j) \in \widetilde{\mathcal{I}_2}$, we have $b \in P_{(i,j)}$ iff $I(x)\ {}_i{\sim}_j\ b$. Then,

$$\mathfrak{A}|^{1,\text{int}}_{I(x),\mathcal{I}_2} \models_I \psi(x) \iff \mathfrak{A}|^{1,\text{int}}_{I(x),\mathcal{I}_2} \models_I \psi'(x) \iff \mathfrak{A}|^{1,\text{int}}_{I(x),\mathcal{I}_2} \models_I \psi''(x)\,.$$

Note that the first equivalence holds because in order to have access to the data value in the data location (y,j) and (z,k) in the 1-view around x, those data must be the same and comparable to one of x.

Then according to Lemma 2.3.10, we can effectively transform ψ'' into an equivalent $\mathsf{FO}_{2\mathbb{DMS}}[\Sigma \cup \widetilde{\mathcal{I}_2};\emptyset]$ formula $\hat{\psi}''$ that is a Boolean combination of formulas of the form $\sigma(x)$ with $\sigma \in \Sigma \cup \widetilde{\mathcal{I}_2}$ and threshold formulas of the form $\exists^{\geq k} y.\varphi_U(y)$ where $U \subseteq \Sigma \cup \widetilde{\mathcal{I}_2}$ and $\varphi_U(y) = \bigwedge_{\sigma\in U} \sigma(y) \wedge \bigwedge_{\sigma\in(\Sigma\cup\widetilde{\mathcal{I}_2})\setminus U} \neg\sigma(y)$. Let M be the maximal such k (or $M = 0$ if there is no threshold formula). Again, we assume that x is not quantified in $\hat{\psi}''$.

We obtain the $\mathsf{FO}_{2\mathbb{DMS}}[\Sigma \cup \{\mathsf{eq}\} \cup \mathsf{C}_M;\emptyset]$ formula χ from $\hat{\psi}''$ by replacing

- $(1,2)(x)$ by $\mathsf{eq}(x)$, and $(1,1)(x)$ and $(2,2)(x)$ by *true*,
- $\exists^{\geq k} y.\varphi_U(y)$ by $\begin{cases} \textit{false} & \text{if } U \cap \widetilde{\mathcal{I}_2} = \emptyset \\ \wr U \cap \Sigma, U \cap \widetilde{\mathcal{I}_2}, k \wr(x) & \text{if } U \cap \widetilde{\mathcal{I}_2} \neq \emptyset. \end{cases}$

When translating a threshold formula $\exists^{\geq k} y.\varphi_U(y)$, we first do a disjunction on the emptiness of $U \cap \widetilde{\mathcal{I}_2}$. We do so because if the intersection is empty, it means we are looking within the 1-view around x for elements y with no shared data values with x (w.r.t $\mathcal{I}_2$). But as we consider view without exterior, there is no such y.

We can then eliminate redundant *true* and *false*. Suppose a well-typed data structure $\mathfrak{A} = (A, (P_\sigma)_\sigma, f_1, f_2) \in \text{Str}(\Sigma \cup \widetilde{\mathcal{I}_2} \cup \{\mathsf{eq}\} \cup \mathsf{C}_M;2\mathbb{DMS})$ and interpretation function I such that, for all $b \in A$ and $(i,j) \in \widetilde{\mathcal{I}_2}$, we have $b \in P_{(i,j)}$ iff $I(x)\ {}_i{\sim}_j\ b$. Then,

$$\mathfrak{A}|^{1,\text{int}}_{I(x),\mathcal{I}_2} \models_I \hat{\psi}''(x) \iff \mathfrak{A}|^{1,\text{int}}_{I(x),\mathcal{I}_2} \models_I \chi(x)\,.$$

Moreover, for $U \subseteq \Sigma$, a nonempty set $R \subseteq \widetilde{\mathcal{I}_2}$, and $k \in \mathbb{N}$, we have

$$\mathfrak{A}|^{1,\mathrm{int}}_{I(x),\mathcal{I}_2} \models_I \wr U, R, k \wr(x) \iff \mathfrak{A} \models_I \wr U, R, k \wr(x)\,.$$

We deduce that, for all $\mathfrak{A} \in \mathrm{Str}(\Sigma; 2\mathbb{DMS})$ and interpretation functions I,

$$\mathfrak{A} \models_I \langle\!\langle \psi \rangle\!\rangle^{1,\mathrm{int}}_x \iff \mathfrak{A} \models_I \chi(x)\,.$$

This concludes the proof. □

Step 2: Well-Diagonalized Structures

In C_M, we still have the diagonal relation $(1,2) \in \widetilde{\mathcal{I}_2}$. Our goal is to get rid of it so that we only deal with the diagonal-free set $\widetilde{\mathcal{S}_2} = \{(1,1),(2,2)\}$. The idea is again to extend a given structure $\mathfrak{A}$, but now we add new elements, one for each value $n \in \mathit{Val}^{\mathfrak{A}}(A)$, which we tag with a unary symbol ed and whose two data values are n. Diagonal equality will be ensured through making a detour via these 'diagonal' elements (hence the name ed).

Formally, when we start from some $\mathfrak{A} = (A, (P_\sigma)_\sigma, f_1, f_2) \in \mathrm{Str}(\Theta \cup \{\mathsf{eq}\}; 2\mathbb{DMS})$, the data structure $\mathfrak{A} + \mathsf{ed} \in \mathrm{Str}(\Theta \cup \{\mathsf{eq}, \mathsf{ed}\}; 2\mathbb{DMS})$ is defined as $(A', (P'_\sigma), f'_1, f'_2)$ where $A' = A \uplus \mathit{Val}^{\mathfrak{A}}(A)$, $f'_i(a) = f_i(a)$ for all $a \in A$ and $i \in \{1,2\}$, $f'_1(a) = f'_2(a) = a$ for all $a \in \mathit{Val}^{\mathfrak{A}}(A)$, $P'_\sigma = P_\sigma$ for all $\sigma \in \Theta \setminus \{\mathsf{eq}\}$, $P'_{\mathsf{ed}} = \mathit{Val}^{\mathfrak{A}}(A)$, and $P'_{\mathsf{eq}} = P_{\mathsf{eq}} \cup \mathit{Val}^{\mathfrak{A}}(A)$.

Example 4.2.5. The structure $\mathfrak{A} + \mathsf{ed}$ is illustrated in Figure 4.2(a), with $\Theta = \emptyset$. ◇

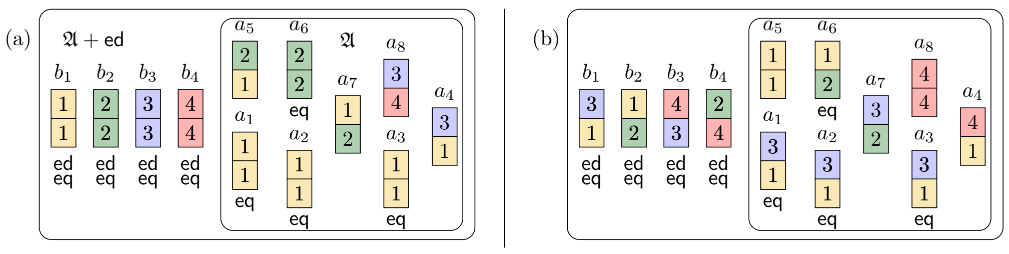


Figure 4.2: (a) Adding diagonal elements. (a)←(b) Making a data structure eq-respecting.

With this, we say that $\mathfrak{B} \in \mathrm{Str}(\Theta \cup \{\mathsf{eq}, \mathsf{ed}\}; 2\mathbb{DMS})$ is *well-diagonalized* if it is of the form $\mathfrak{A} + \mathsf{ed}$ for some *eq-respecting* $\mathfrak{A} \in \mathrm{Str}(\Theta \cup \{\mathsf{eq}\}; 2\mathbb{DMS})$. Note that then $\mathfrak{B}$ is eq-respecting, too.

Example 4.2.6. The data structure $\mathfrak{A} + \mathsf{ed}$ from Figure 4.2(a) is well-diagonalized. The one from Figure 4.2(b) is not well-diagonalized (in particular, it is not eq-respecting). ◇

We will need a way to ensure that the considered data structures are well-diagonalized. To this end, we introduce the following sentence from $\mathsf{FO}^2[\Theta \cup \{\mathsf{eq}, \mathsf{ed}\}; \mathcal{S}_2]$:

$$\begin{aligned}\xi^{\Theta}_{\mathsf{ed}} \;:=\; & \textstyle\bigwedge_{i \in \{1,2\}} \Big(\forall x. \exists y. (\mathsf{ed}(y) \wedge x \;{}_i{\sim}_i\; y) \wedge \big(\forall x. \forall y. (\mathsf{ed}(x) \wedge \mathsf{ed}(y) \wedge x \;{}_i{\sim}_i\; y) \rightarrow x = y\big)\Big) \\ & \wedge \big(\forall x. \mathsf{eq}(x) \leftrightarrow \exists y. (\mathsf{ed}(y) \wedge x \;{}_1{\sim}_1\; y \wedge x \;{}_2{\sim}_2\; y)\big) \\ & \wedge \big(\forall x. \mathsf{ed}(x) \rightarrow \textstyle\bigwedge_{\sigma \in \Theta} \neg\sigma(x)\big).\end{aligned}$$

Every structure that is well-diagonalized satisfies $\xi^{\Theta}_{\mathsf{ed}}$. The converse is not true in general. In particular, a model of $\xi^{\Theta}_{\mathsf{ed}}$ is not necessarily eq-respecting. However, if a structure satisfies a formula $\varphi \in \mathsf{FO}_{2\mathbb{DMS}}[\Theta \cup \{\mathsf{eq},\mathsf{ed}\};\mathcal{S}_2]$, then it is possible to perform a permutation on the first (or the second) values of its elements while preserving φ. This allows us to get:

Lemma 4.2.7. *Let $\mathfrak{B} \in \mathrm{Str}(\Theta \cup \{\mathsf{eq},\mathsf{ed}\}; 2\mathbb{DMS})$ and $\varphi \in \mathsf{FO}_{2\mathbb{DMS}}[\Theta \cup \{\mathsf{eq},\mathsf{ed}\};\mathcal{S}_2]$. If $\mathfrak{B} \models \varphi \wedge \xi^{\Theta}_{\mathsf{ed}}$, then there exists an eq-respecting $\mathfrak{A} \in \mathrm{Str}(\Theta \cup \{\mathsf{eq}\}; 2\mathbb{DMS})$ such that $\mathfrak{A}{+}\mathsf{ed} \models \varphi$.*

Example 4.2.8. Consider Figure 4.2 and let $\Theta = \emptyset$. The data structure from Figure 4.2(b) satisfies $\xi^{\Theta}_{\mathsf{ed}}$, though it is not well-diagonalized. Suppose it also satisfies some formula $\varphi \in \mathsf{FO}_{2\mathbb{DMS}}[\{\mathsf{eq},\mathsf{ed}\};\mathcal{S}_2]$. By permutation of the first data values, we obtain the well-diagonalized data structure in Figure 4.2(a). As φ does not talk about the diagonal relation, satisfaction of φ is preserved. ◇

Proof of Lemma 4.2.7: Let $\mathfrak{B} = (A, (P_\sigma), f_1, f_2)$ in $\mathrm{Str}(\Theta \cup \{\mathsf{eq},\mathsf{ed}\}; 2\mathbb{DMS})$ such that $\mathfrak{B} \models \varphi \wedge \xi_{\mathsf{ed}}$. We define the sets $D_1 = \{n \in \mathbb{N} \mid \exists b \in P_{\mathsf{ed}}.f_1(b) = n\}$ and $D_2 = \{n \in \mathbb{N} \mid \exists b \in P_{\mathsf{ed}}.f_2(b) = n\}$. Since

$$\mathfrak{B} \models \bigwedge_{i \in \{1,2\}} \Big(\big(\forall x.\exists y.\mathsf{ed}(y) \wedge x \;{}_i{\sim}_i\; y\big) \wedge \big(\forall x.\forall y.(\mathsf{ed}(x) \wedge \mathsf{ed}(y) \wedge x \;{}_i{\sim}_i\; y) \rightarrow x = y\big) \Big),$$

we deduce that $|D_1| = |P_{\mathsf{ed}}| = |D_2|$ and furthermore the mapping $\pi : D_1 \mapsto D_2$ defined by $\pi(n) = m$ iff there exists $b \in P_{\mathsf{ed}}$ such that $f_1(b) = n$ and $f_2(b) = m$ is a well-defined bijection. It is well defined because there is a single element b in P_{ed} such that $f_1(b) = n$ and it is a bijection because for all $m \in D_2$, there is a single $b \in P_{\mathsf{ed}}$ such that $f_2(b) = m$. We can consequently extend π to be a permutation from $\mathbb{N}$ to $\mathbb{N}$. We then take the model $\mathfrak{A}' = (A, (P_\sigma), \pi \circ f_1, f_2)$. Since $\varphi \wedge \xi_{\mathsf{ed}} \in \mathsf{FO}_{2\mathbb{DMS}}[\Theta \cup \{\mathsf{eq},\mathsf{ed}\};\mathcal{S}_2]$ with $\mathcal{S}_2 = \{(1,1),(2,2)\}$ and since $\mathfrak{B} \models \varphi \wedge \xi_{\mathsf{ed}}$, we deduce that $\mathfrak{A}' \models \varphi \wedge \xi_{\mathsf{ed}}$ because performing a permutation on the first data values of the elements of $\mathfrak{B}$ does not affect the satisfaction of $\varphi \wedge \xi_{\mathsf{ed}}$ (this is a consequence of the fact that there is no comparison between the first values and the second values of the elements). The satisfaction of ξ_{ed} by $\mathfrak{A}'$ allows us to deduce that $\mathfrak{A}'$ is well-diagonalized. We can in fact safely remove from $\mathfrak{A}'$ the elements of P_{ed} to obtain a structure $\mathfrak{A} \in \mathrm{Str}(\Theta \cup \{\mathsf{eq}\}; 2\mathbb{DMS})$ which is eq-respecting (this is due to the fact that $\mathfrak{A}' \models \big(\forall x.\mathsf{eq}(x) \leftrightarrow \exists y.(\mathsf{ed}(y) \wedge x \;{}_1{\sim}_1\; y \wedge x \;{}_2{\sim}_2\; y)\big) \wedge \big(\forall x.\mathsf{ed}(x) \rightarrow \bigwedge_{\sigma \in \Theta} \neg\sigma(x)\big)$) and such that $\mathfrak{A}' = \mathfrak{A} + \mathsf{ed}$. □

Finally, we can inductively translate $\varphi \in \mathsf{FO}_{2\mathbb{DMS}}[\Theta \cup \{\mathsf{eq}\};\emptyset]$ into a formula $[\![\varphi]\!]_{+\mathsf{ed}} \in \mathsf{FO}_{2\mathbb{DMS}}[\Theta \cup \{\mathsf{eq},\mathsf{ed}\};\emptyset]$ that does not take into account the extra ‘diagonal’ elements:

$$\begin{aligned} [\![\sigma(x)]\!]_{+\mathsf{ed}} &= \sigma(x) & [\![\neg\varphi]\!]_{+\mathsf{ed}} &= \neg[\![\varphi]\!]_{+\mathsf{ed}} \\ [\![x = y]\!]_{+\mathsf{ed}} &= (x = y) & [\![\exists x.\varphi]\!]_{+\mathsf{ed}} &= \exists x.(\neg\mathsf{ed}(x) \wedge [\![\varphi]\!]_{+\mathsf{ed}}) \\ [\![\varphi \vee \varphi']\!]_{+\mathsf{ed}} &= [\![\varphi]\!]_{+\mathsf{ed}} \vee [\![\varphi']\!]_{+\mathsf{ed}} \end{aligned}$$

We immediately obtain:

Lemma 4.2.9. *Let $\mathfrak{A} \in \mathrm{Str}(\Theta \cup \{\mathsf{eq}\}; 2\mathbb{DMS})$ and $\varphi \in \mathsf{FO}_{2\mathbb{DMS}}[\Theta \cup \{\mathsf{eq}\}; \emptyset]$ be a sentence. We have $\mathfrak{A} \models \varphi$ iff $\mathfrak{A} + \mathsf{ed} \models [\![\varphi]\!]_{+\mathsf{ed}}$.*

Step 3: Getting Rid Of the Diagonal Relation

We will now exploit well-diagonalized data structures to reason about environments relative to $\mathcal{I}_2$ $(= \{{}_1{\sim}_1, {}_2{\sim}_2, {}_1{\sim}_2\})$ in terms of environments relative to $\mathcal{S}_2 (= \{{}_1{\sim}_1, {}_2{\sim}_2\})$. Recall that Θ ranges over finite sets such that $\Sigma \subseteq \Theta$.

Lemma 4.2.10. *Let $\mathfrak{A} = (A, (P_\sigma)_\sigma, f_1, f_2) \in \mathrm{Str}(\Theta \cup \{\mathsf{eq}\}; 2\mathbb{DMS})$ be eq-respecting and $\mathfrak{B} = \mathfrak{A} + \mathsf{ed}$. Moreover, let $a \in A$, $U \subseteq \Sigma$, and $R \subseteq \widetilde{\mathcal{I}_2}$ be a nonempty set. We have* $\mathtt{Env}_{\mathfrak{A},\Sigma,\mathcal{I}_2}(a, U, R) =$

$$
\begin{cases}
\mathtt{Env}_{\mathfrak{B},\Sigma,\mathcal{S}_2}(a, U, \mathcal{S}_2) \setminus P_{\mathsf{ed}} & \text{if } a \in P_{\mathsf{eq}} \text{ and } R = \widetilde{\mathcal{I}_2} & (1)\\
\mathtt{Env}_{\mathfrak{B},\Sigma,\mathcal{S}_2}(a, U, \mathcal{S}_2) & \text{if } a \notin P_{\mathsf{eq}} \text{ and } R = \widetilde{\mathcal{S}_2} & (2)\\
\mathtt{Env}_{\mathfrak{B},\Sigma,\mathcal{S}_2}(a, U, \{(1,1)\}) \cap (P_{\mathsf{eq}} \setminus P_{\mathsf{ed}}) & \text{if } a \notin P_{\mathsf{eq}} \text{ and } R = \{(1,1),(1,2)\} & (3)\\
\mathtt{Env}_{\mathfrak{B},\Sigma,\mathcal{S}_2}(a, U, \{(2,2)\}) & \text{if } a \in P_{\mathsf{eq}} \text{ and } R = \{(2,2),(1,2)\} & (4)\\
\mathtt{Env}_{\mathfrak{B},\Sigma,\mathcal{S}_2}(a, U, \{(2,2)\}) \setminus P_{\mathsf{ed}} & \text{if } a \notin P_{\mathsf{eq}} \text{ and } R = \{(2,2)\} & (5)\\
\mathtt{Env}_{\mathfrak{B},\Sigma,\mathcal{S}_2}(a, U, \{(1,1)\}) \setminus P_{\mathsf{eq}} & \text{if } \qquad R = \{(1,1)\} & (6)\\
\mathtt{Env}_{\mathfrak{B},\Sigma,\mathcal{S}_2}(d, U, \{(2,2)\}) & \text{if } a \notin P_{\mathsf{eq}} \text{ and } R = \{(1,2)\} & (7)\\
\quad \text{for the unique } d \in P_{\mathsf{ed}} \text{ such that } d\,{}_1{\sim}_1^{\mathfrak{B}}\, a & & \\
\emptyset & \text{otherwise} & (8)
\end{cases}
$$

Example 4.2.11. Let us illustrate all the cases of Lemma 4.2.10 using Figure 4.2(a), and letting $\Sigma = \Theta = \emptyset$.

1. Let $a = a_1$ and $R = \widetilde{\mathcal{I}_2}$. Then, $\mathtt{Env}_{\mathfrak{A},\Sigma,\mathcal{I}_2}(a, \emptyset, R) = \{a_1, a_2, a_3\}$. We also have that $\mathtt{Env}_{\mathfrak{B},\Sigma,\mathcal{S}_2}(a, \emptyset, \mathcal{S}_2) = \{a_1, a_2, a_3, b_1\}$: These are the elements that coincide with a *exactly* on the first and the on the second data value when we dismiss the diagonal relation. Of course, as we consider $\mathfrak{B}$, this includes b_1, which we have to exclude. Thus, $\mathtt{Env}_{\mathfrak{A},\Sigma,\mathcal{I}_2}(a, \emptyset, R) = \mathtt{Env}_{\mathfrak{B},\Sigma,\mathcal{S}_2}(a, \emptyset, \mathcal{S}_2) \setminus P_{\mathsf{ed}}$.

2. Let $a = a_7$ and $R = \mathcal{S}_2$. Then, $\mathtt{Env}_{\mathfrak{A},\Sigma,\mathcal{I}_2}(a, \emptyset, R) = \mathtt{Env}_{\mathfrak{B},\Sigma,\mathcal{S}_2}(a, \emptyset, \mathcal{S}_2) = \{a_7\}$. Since $a \notin P_{\mathsf{eq}}$, it actually does not matter whether we include the diagonal relation or not.

3. Let $a = a_7$ and $R = \{(1,1),(1,2)\}$. Then, $\mathtt{Env}_{\mathfrak{A},\Sigma,\mathcal{I}_2}(a, \emptyset, R) = \{a_1, a_2, a_3\}$. So how do we get this set in $\mathfrak{B}$ without referring to the diagonal relation? The idea is to use only ${}_1{\sim}_1 \in \mathcal{S}_2$ and to ensure data equality by restricting to elements in P_{eq} (again excluding P_{ed}). Indeed, we have $\mathtt{Env}_{\mathfrak{B},\Sigma,\mathcal{S}_2}(a, \emptyset, \{(1,1)\}) \cap (P_{\mathsf{eq}} \setminus P_{\mathsf{ed}}) = \{a_1, a_2, a_3, b_1\} \cap (\{a_1, a_2, a_3, b_1\} \setminus \{b_1\}) = \{a_1, a_2, a_3\}$.

4. Let $a = a_1$ and $R = \{(2,2),(1,2)\}$. Then, $\mathtt{Env}_{\mathfrak{A},\Sigma,\mathcal{I}_2}(a, \emptyset, R) = \{a_4, a_5\}$. So we are looking for elements that have 1 as the second data value and a first data value different from 1, and this set is exactly $\mathtt{Env}_{\mathfrak{B},\Sigma,\mathcal{S}_2}(a, \emptyset, \{(2,2)\})$.

5. Let $a = a_5$ and $R = \{(2,2)\}$. Then, $\mathtt{Env}_{\mathfrak{A},\Sigma,\widetilde{\mathcal{I}_2}}(a, \emptyset, R) = \{a_1, a_2, a_3, a_4\}$, which is the set of elements that have 1 as the second data value and a first data value different from 2. Thus, this is exactly $\mathtt{Env}_{\mathfrak{B},\Sigma,\mathcal{S}_2}(a, \emptyset, \{(2,2)\}) \setminus P_{\mathsf{ed}}$ (i.e., after discarding $b_1 \in P_{\mathsf{ed}}$).
6. Let $a = a_4$ and $R = \{(1,1)\}$. We have $\mathtt{Env}_{\mathfrak{A},\Sigma,\widetilde{\mathcal{I}_2}}(a, \emptyset, R) = \{a_8\}$. Looking at $\mathfrak{B}$ and discarding the diagonal relation would also include b_3 and any element with data-value pair $(3,3)$. Discarding P_{eq}, we obtain $\mathtt{Env}_{\mathfrak{B},\Sigma,\mathcal{S}_2}(a, \emptyset, \{(1,1)\}) \setminus P_{\mathsf{eq}} = \{a_8, b_3\} \setminus \{b_3\} = \{a_8\}$.
7. Let $a = a_7$ and $R = \{(1,2)\}$. Then, $\mathtt{Env}_{\mathfrak{A},\Sigma,\widetilde{\mathcal{I}_2}}(a, \emptyset, R) = \{a_4, a_5\}$, which is the set of elements whose second data value is 1 and whose first data value is different from 1. The idea is now to change the reference point. Take the unique $d \in P_{\mathsf{ed}}$ such that $d\,{}_1\!\sim_1^{\mathfrak{B}} a$. Thus, $d = b_1$. The set $\mathtt{Env}_{\mathfrak{B},\Sigma,\mathcal{S}_2}(b_1, \emptyset, \{(2,2)\})$ gives us exactly the elements that have 1 as the second data value and a first value different from 1, as desired.

◇

Proof of Lemma 4.2.10: Let $\mathfrak{A} = (A, (P_\sigma)_\sigma, f_1, f_2) \in \mathrm{Str}(\Theta \cup \{\mathsf{eq}\}; 2\mathbb{DMS})$ be eq-respecting and $\mathfrak{B} = \mathfrak{A} + \mathsf{ed}$. We consider $a \in A$, $U \subseteq \Sigma$, and $R \subseteq \widetilde{\mathcal{I}_2}$ be a nonempty set. Note that by definition of $\mathtt{Env}$, we have $\mathtt{Env}_{\mathfrak{A},\Sigma,\widetilde{\mathcal{I}_2}}(a, U, R) = \mathtt{Env}_{\mathfrak{B},\Sigma,\widetilde{\mathcal{I}_2}}(a, U, R) \setminus P_{\mathsf{ed}}$ and when $R \neq \{(1,2)\}$, we have $\mathtt{Env}_{\mathfrak{B},\Sigma,\mathcal{S}_2}(a, U, R \setminus \{(1,2)\}) = \mathtt{Env}_{\mathfrak{B},\Sigma,\widetilde{\mathcal{I}_2}}(a, U, R \cup \{(1,2)\}) \cup \mathtt{Env}_{\mathfrak{B},\Sigma,\widetilde{\mathcal{I}_2}}(a, U, R \setminus \{(1,2)\})$. We will use these two equalities in the rest of the proof. We now perform a case analysis on R.

1. Assume $R = \widetilde{\mathcal{I}_2} = \{(1,1), (2,2), (1,2)\}$. First we suppose that $a \notin P_{\mathsf{eq}}$. Since $\mathfrak{A}$ is eq-respecting it implies that $f_1(a) \neq f_2(a)$. Now assume there exists $b \in \mathtt{Env}_{\mathfrak{A},\Sigma,\widetilde{\mathcal{I}_2}}(a, U, R)$, this means that $a\,{}_1\!\sim_1^{\mathfrak{A}} b$ and $a\,{}_2\!\sim_2^{\mathfrak{A}} b$ and $a\,{}_1\!\sim_2^{\mathfrak{A}} b$. Hence we have $f_2(a) = f_2(b)$ and $f_1(a) = f_2(b)$, which is a contradiction. Consequently $\mathtt{Env}_{\mathfrak{A},\Sigma,\widetilde{\mathcal{I}_2}}(a, U, R) = \emptyset$. We now suppose that $a \in P_{\mathsf{eq}}$. In that case, since $\mathfrak{A}$ is eq-respecting, we have $\mathtt{Env}_{\mathfrak{B},\Sigma,\widetilde{\mathcal{I}_2}}(a, U, R \setminus \{(1,2)\}) = \emptyset$. In fact if $a\,{}_2\!\sim_2^{\mathfrak{A}} b$ for some b then $a\,{}_1\!\sim_2^{\mathfrak{A}} b$ as $a \in P_{\mathsf{eq}}$. Hence we have $\mathtt{Env}_{\mathfrak{A},\Sigma,\widetilde{\mathcal{I}_2}}(a, U, R) = \mathtt{Env}_{\mathfrak{B},\Sigma,\widetilde{\mathcal{I}_2}}(a, U, R) \setminus P_{\mathsf{ed}} = \mathtt{Env}_{\mathfrak{B},\Sigma,\mathcal{S}_2}(a, U, \mathcal{S}_2) \setminus P_{\mathsf{ed}}$.
2. Assume $R = \mathcal{S}_2 = \{(1,1), (2,2)\}$. By a similar reasoning as the previous case, if $a \in P_{\mathsf{eq}}$, we have necessarily $\mathtt{Env}_{\mathfrak{A},\Sigma,\widetilde{\mathcal{I}_2}}(a, U, R) = \emptyset$. Now suppose $a \notin P_{\mathsf{eq}}$. Thanks to this hypothesis, we know that $P_{\mathsf{ed}} \cap \mathtt{Env}_{\mathfrak{B},\Sigma,\mathcal{S}_2}(a, U, \mathcal{S}_2) = \emptyset$ and that $\mathtt{Env}_{\mathfrak{B},\Sigma,\widetilde{\mathcal{I}_2}}(a, U, \{(1,1), (2,2), (1,2)\}) = \emptyset$. So we obtain directly $\mathtt{Env}_{\mathfrak{A},\Sigma,\widetilde{\mathcal{I}_2}}(a, U, \mathcal{S}_2) = \mathtt{Env}_{\mathfrak{B},\Sigma,\mathcal{S}_2}(a, U, \mathcal{S}_2)$.
3. Assume $R = \{(1,1), (1,2)\}$. Again it is obvious that if $a \in P_{\mathsf{eq}}$, we have $\mathtt{Env}_{\mathfrak{A},\Sigma,\widetilde{\mathcal{I}_2}}(a, U, R) = \emptyset$. We suppose that $a \notin P_{\mathsf{eq}}$. Note that we have that $\mathtt{Env}_{\mathfrak{B},\Sigma,\widetilde{\mathcal{I}_2}}(a, U, R) \subseteq P_{\mathsf{eq}}$ and $\mathtt{Env}_{\mathfrak{B},\Sigma,\widetilde{\mathcal{I}_2}}(a, U, \{1,1\}) \cap P_{\mathsf{eq}} = \emptyset$. Since $\mathtt{Env}_{\mathfrak{B},\Sigma,\mathcal{S}_2}(a, U, \{(1,1)\}) = \mathtt{Env}_{\mathfrak{B},\Sigma,\widetilde{\mathcal{I}_2}}(a, U, R) \cup \mathtt{Env}_{\mathfrak{B},\Sigma,\widetilde{\mathcal{I}_2}}(a, U, \{(1,1)\})$, we deduce that $\mathtt{Env}_{\mathfrak{B},\Sigma,\mathcal{S}_2}(a, U, \{(1,1)\}) \cap P_{\mathsf{eq}} = \mathtt{Env}_{\mathfrak{B},\Sigma,\widetilde{\mathcal{I}_2}}(a, U, R)$. From which we can conclude $\mathtt{Env}_{\mathfrak{A},\Sigma,\widetilde{\mathcal{I}_2}}(a, U, R) = \mathtt{Env}_{\mathfrak{B},\Sigma,\mathcal{S}_2}(a, U, \{(1,1)\}) \cap P_{\mathsf{eq}} \setminus P_{\mathsf{ed}}$.

4. Assume $R = \{(2,2),(1,2)\}$. Again, it is obvious that if $a \notin P_{\mathsf{eq}}$, we have $\mathtt{Env}_{\mathfrak{A},\Sigma,\mathcal{I}_2}(a,U,R) = \emptyset$. We now suppose that $a \in P_{\mathsf{eq}}$. In that case, we have that $\mathtt{Env}_{\mathfrak{B},\Sigma,\mathcal{I}_2}(a,U,R \setminus \{(1,2)\}) = \emptyset$ and furthermore $\mathtt{Env}_{\mathfrak{B},\Sigma,\mathcal{I}_2}(a,U,R) \cap P_{\mathsf{ed}} = \emptyset$. We can hence conclude that $\mathtt{Env}_{\mathfrak{A},\Sigma,\mathcal{I}_2}(a,U,R) = \mathtt{Env}_{\mathfrak{B},\Sigma,\mathcal{S}_2}(a,U,\{(2,2)\})$.

5. Assume $R = \{(2,2)\}$. As before if $a \in P_{\mathsf{eq}}$, we have $\mathtt{Env}_{\mathfrak{A},\Sigma,\mathcal{I}_2}(a,U,R) = \emptyset$. We now suppose that $a \notin P_{\mathsf{eq}}$. In that case, we have immediately $\mathtt{Env}_{\mathfrak{B},\Sigma,\mathcal{I}_2}(a,U,\{(2,2),(1,2)\}) = \emptyset$ and consequently

$$\mathtt{Env}_{\mathfrak{A},\Sigma,\mathcal{I}_2}(a,U,R) = \mathtt{Env}_{\mathfrak{B},\Sigma,\mathcal{I}_2}(a,U,\{(2,2)\}) \setminus P_{\mathsf{ed}} = \mathtt{Env}_{\mathfrak{B},\Sigma,\mathcal{S}_2}(a,U,\{(2,2)\}) \setminus P_{\mathsf{ed}}\,.$$

6. Assume $R = \{(1,1)\}$. Remember that we have:
$\mathtt{Env}_{\mathfrak{B},\Sigma,\mathcal{S}_2}(a,U,\{(1,1)\}) = \mathtt{Env}_{\mathfrak{B},\Sigma,\mathcal{I}_2}(a,U,\{(1,1),(1,2)\}) \cup \mathtt{Env}_{\mathfrak{B},\Sigma,\mathcal{I}_2}(a,U,\{(1,1)\})$.
But$\mathtt{Env}_{\mathfrak{B},\Sigma,\mathcal{I}_2}(a,U,\{(1,1),(1,2)\}) \subseteq P_{\mathsf{eq}}$ and $\mathtt{Env}_{\mathfrak{B},\Sigma,\mathcal{I}_2}(a,U,\{(1,1)\}) \cap P_{\mathsf{eq}} = \emptyset$. We hence deduce that $\mathtt{Env}_{\mathfrak{B},\Sigma,\mathcal{S}_2}(a,U,\{(1,1)\}) \setminus P_{\mathsf{eq}} = \mathtt{Env}_{\mathfrak{B},\Sigma,\mathcal{I}_2}(a,U,\{(1,1)\})$ and since $(\mathtt{Env}_{\mathfrak{B},\Sigma,\mathcal{S}_2}(a,U,\{(1,1)\}) \setminus P_{\mathsf{eq}}) \cap P_{\mathsf{ed}} = \emptyset$, we obtain $\mathtt{Env}_{\mathfrak{A},\Sigma,\mathcal{I}_2}(a,U,R) = \mathtt{Env}_{\mathfrak{B},\Sigma,\mathcal{S}_2}(a,U,\{(1,1)\}) \setminus P_{\mathsf{eq}}$.

7. Assume $R = \{(1,2)\}$. Again it is obvious that if $a \in P_{\mathsf{eq}}$, we have $\mathtt{Env}_{\mathfrak{A},\Sigma,\mathcal{I}_2}(a,U,R) = \emptyset$. We now suppose that $a \notin P_{\mathsf{eq}}$. By definition, since $\mathfrak{B} = \mathfrak{A} + \mathsf{ed}$, in $\mathfrak{B}$ there is a unique $d \in P_{\mathsf{ed}}$ such that $d \;{}_1{\sim}_1^{\mathfrak{B}}\; a$. We have then $\mathtt{Env}_{\mathfrak{B},\Sigma,\mathcal{I}_2}(a,U,R) = \mathtt{Env}_{\mathfrak{B},\Sigma,\mathcal{I}_2}(d,U,\{(1,2),(2,2)\})$. As for the case 4., we deduce that $\mathtt{Env}_{\mathfrak{B},\Sigma,\mathcal{I}_2}(d,U,\{(1,2),(2,2)\}) = \mathtt{Env}_{\mathfrak{B},\Sigma,\mathcal{S}_2}(d,U,\{(2,2)\})$.
Hence $\mathtt{Env}_{\mathfrak{B},\Sigma,\mathcal{I}_2}(a,U,R) = \mathtt{Env}_{\mathfrak{B},\Sigma,\mathcal{S}_2}(d,U,\{(2,2)\})$. □

Let us wrap up: By Lemmas 4.2.4 and 4.2.10, we end up with checking counting constraints in an extended data structure without using the diagonal relation.

Step 4: Counting in Two-Variable Logic

The next step is to express these constraints using two-variable formulas. Counting in two-variable logic is established using further unary predicates. These additional predicates allow us to define a partitioning of the universe of a structure into so-called *intersections*. Suppose $\mathfrak{A} = (A,(P_\sigma)_\sigma,f_1,f_2) \in \mathrm{Str}(\Theta \cup \{\mathsf{eq},\mathsf{ed}\};2\mathbb{DMS})$, where $\Sigma \subseteq \Theta$. Let $a \in A \setminus P_{\mathsf{ed}}$ and define $\ell_\Sigma(a) = \{\sigma \in \Sigma \mid a \in P_\sigma\}$. The *intersection* of a in $\mathfrak{A}$ is the set $\{b \in A \setminus P_{\mathsf{ed}} \mid a \;{}_1{\sim}_1\; b \wedge a \;{}_2{\sim}_2\; b \wedge \ell_\Sigma(a) = \ell_\Sigma(b)\}$. A set is called an *intersection* in $\mathfrak{A}$ if it is the intersection of some $a \in A \setminus P_{\mathsf{ed}}$.

Example 4.2.12. Consider Figure 4.3 and suppose $\Sigma = \{\mathsf{p}\}$. The intersections of the given data structure are gray-shaded. ◇

Let us introduce the various unary predicates, which will be assigned to *non-diagonal* elements. There are three types of them (for the first two types, also see Figure 4.3):

1. The unary predicates $\aleph_M^\gamma = \{\gamma_1,\ldots,\gamma_M\}$ have the following intended meaning: For all intersections I and $i \in \{1,\ldots,M\}$, we have $|I| \geq i$ iff there is $a \in I$ such that

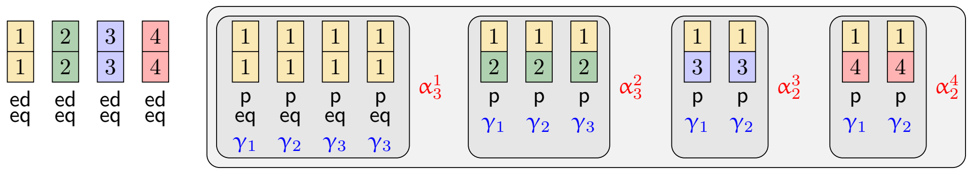


Figure 4.3: Counting intersections for $M = 3$ and elements with label p

$a \in P_{\gamma_i}$. In other words, the presence (or absence) of γ_i in an intersection I tells us whether $|I| \geq i$.

2. The predicates $\aleph_M^{\alpha} = \{\alpha_i^j \mid i \in \{1,\ldots,M\}$ and $j \in \{1,\ldots,M+2\}\}$ have the following meaning: If a is labeled with α_i^j, then (i) there are at least j intersections sharing the same first value and the same label set $\ell_\Sigma(a)$, and (ii) the intersection of a has i elements if $i \leq M-1$ and at least M elements if $i = M$. Hence, in α_i^j, index i counts the elements inside an intersection, and j labels up to $M+2$ different intersections. We need to go beyond M due to Lemma 4.2.10: When we remove certain elements (e.g., P_{eq}) from an environment, we must be sure to still have sufficiently many to be able to count until M.

3. Labels from $\aleph_M^{\beta} = \{\beta_i^j \mid i \in \{1,\ldots,M\}$ and $j \in \{1,\ldots,M+1\}\}$ will play a similar role as those in $\aleph_M^{\alpha}$ but consider the second values of the elements instead of the first ones.

Example 4.2.13. A suitable labeling for types γ and α is illustrated in Figure 4.3 for $M = 3$. ◇

Let $\aleph_M = \aleph_M^{\alpha} \cup \aleph_M^{\beta} \cup \aleph_M^{\gamma}$ denote the set of all these unary predicates. We now build sentences $\varphi_\alpha, \varphi_\beta, \varphi_\gamma \in \mathsf{FO}^2_{2\mathbb{DMS}}[\Theta \cup \{\mathsf{eq}, \mathsf{ed}\} \cup \aleph_M; \mathcal{S}_2]$ that guarantee the respective properties, and show that they indeed guarantee them. In particular, they make use of the formula $x \;{}_1{\sim}_1\; y \wedge x \;{}_2{\sim}_2\; y \wedge \bigwedge_{\sigma\in\Sigma} \sigma(x) \leftrightarrow \sigma(y)$ saying that two (non-diagonal) elements x and y are in the same intersection.

To deal with the predicates in $\aleph_M^{\gamma}$, we first define the formula $\varphi_{same}^{int} = x \;{}_1{\sim}_1\; y \wedge x \;{}_2{\sim}_2\; y \wedge \bigwedge_{\sigma\in\Sigma} \sigma(x) \leftrightarrow \sigma(y)$ and introduce the following formulas:

$$\varphi_\gamma^1(x) \;:=\; \bigvee_{i\in[1,M]} \Big(\gamma_i(x) \wedge \bigwedge_{j\in[1,M]\setminus\{i\}} \neg\gamma_j(x)\Big)$$

$$\varphi_\gamma^2(x) \;:=\; \bigwedge_{i\in[1,M-1]} \Big(\gamma_i(x) \rightarrow \neg\exists y.\big(x \neq y \wedge \varphi_{same}^{int}(x,y) \wedge \gamma_i(y)\big)\Big)$$

$$\varphi_\gamma^3(x) \;:=\; \bigwedge_{i\in[2,M]} \Big(\gamma_i(x) \rightarrow \big(\exists y.\varphi_{same}^{int}(x,y) \wedge \gamma_{i-1}(y)\big)\Big)$$

We then let $\varphi_\gamma := \forall x.\big(\neg\mathsf{ed}(x) \rightarrow (\varphi_\gamma^1(x) \wedge \varphi_\gamma^2(x) \wedge \varphi_\gamma^3(x))\big) \wedge \big(\mathsf{ed}(x) \rightarrow \bigwedge_{\gamma\in\aleph_M^{\gamma}} \neg\gamma(x)\big)$. Thus, a data structure satisfies φ_γ if no diagonal element is labelled with predicates in $\aleph_M^{\gamma}$

and (1) all its non-diagonal elements are labelled with exactly one predicate in $\aleph_M^{\gamma}$ (see φ_{γ}^1), (2) if $i \leq M-1$, then there are no two γ_i-labelled elements with the same labels of Σ and in the same intersection (see φ_{γ}^2), and (3) if $i \geq 2$, then for all γ_i-labelled elements, there exists an γ_{i-1}-labelled element with the same labels of Σ and in the same intersection (see φ_{γ}^3).

Lemma 4.2.14. *Let* $\mathfrak{A} = (A, (P_\sigma)_\sigma, f_1, f_2) \in \mathrm{Str}(\Sigma \cup \{\mathsf{eq}\} \cup \mathbb{C}_M \cup \aleph_M; 2\mathbb{DMS})$ *be eq-respecting and such that* $\mathfrak{A} + \mathsf{ed} \models \varphi_\gamma$. *We consider* $a \in A$ *and* $\gamma_i \in \aleph_M$ *and* $E_a = \{b \in A \mid a \ {}_1{\sim}_1\ b \wedge a \ {}_2{\sim}_2\ b \wedge \ell_\Sigma(a) = \ell_\Sigma(b)\}$. *Then,* $|E_a| \geq i$ *iff there exists* $b \in E_a$ *such that* $b \in P_{\gamma[i]}$.

Proof: For any $b \in E_a$, as $\mathfrak{A} + \mathsf{ed} \models \varphi_\gamma^1$ there is exactly one $j \in [1, M]$ such that $b \in P_{\gamma_j}$. This allow us to build the function $f : E_a \to [1, M]$ which associates to any $b \in E_a$ such a j. Let $J = \{f(b) | b \in E_a\}$ denotes the image of E_a under f. As $\mathfrak{A} + \mathsf{ed} \models \varphi_\gamma^3$, for any $j \in [2, M]$ if $j \in J$ then $j - 1 \in J$. And as $E_a \neq \emptyset$, there is $j_{\max} \in [1, M]$ such that $J = [1, j_{\max}]$. We can now rephrase our goal as $|E_a| \geq i$ iff $i \in J$. Assuming that $i \in J$, we have $i \leq j_{\max}$. As f is a function, we have $|E_a| \geq |J|$. As $|J| = j_{\max}$, we have that $|E_a| \geq i$. Conversely, assuming that $|E_a| \geq i$. Assume by contradiction that $i \notin J$, then $j_{\max} < i \leq M$. That is, for all $j \in J$, we have $j < M$. Since $\mathfrak{A} + \mathsf{ed} \models \varphi_\gamma^2$, all elements of J have exactly one preimage. So $|E_a| = |J| = j_{\max} < i$, which contradicts the assumption. □

It is then easy to see that, in an intersection, if there is an element a labelled by γ_i and no element labelled by γ_{i+1} for $i < M$, then the intersection has exactly i elements; moreover, if there is a node a labelled by γ_M then the intersection has at least M elements.

We now show how we use the predicates in $\aleph_M^{\alpha}$ and introduce the following formulas (where $\varphi_{same}^{int} = x \ {}_1{\sim}_1\ y \wedge x \ {}_2{\sim}_2\ y \wedge \bigwedge_{\sigma \in \Sigma} \sigma(x) \leftrightarrow \sigma(y)$ and $\varphi_{same} = \bigwedge_{\sigma \in \Sigma} \sigma(x) \leftrightarrow \sigma(y)$):

$$\varphi_{\alpha}^1(x) := \bigvee_{\substack{i \in [1,M] \\ j \in [1,M+2]}} \left(\alpha_i^j(x) \wedge \bigwedge_{\substack{k \in [1,M] \\ \ell \in [1,M+2] \\ (k,\ell) \neq (i,j)}} \neg \alpha_k^\ell(x) \right)$$

$$\varphi_{\alpha}^2(x) := \bigwedge_{\substack{i \in [1,M] \\ j \in [1,M+2]}} \left(\alpha_i^j(x) \to \forall y. \big((\neg \mathsf{ed}(y) \wedge \varphi_{same}^{int}(x,y)) \to \alpha_i^j(y) \big) \right)$$

$$\varphi_{\alpha}^3(x) := \bigwedge_{\substack{i \in [1,M-1] \\ j \in [1,M+2]}} \left(\alpha_i^j(x) \to \begin{pmatrix} \exists y. \big(\varphi_{same}^{int}(x,y) \wedge \gamma_i(y)\big) \\ \wedge \neg \exists y. \big(\varphi_{same}^{int}(x,y) \wedge \gamma_{i+1}(y)\big) \end{pmatrix} \right) \wedge$$
$$\bigwedge_{j \in [1,M+2]} \left(\alpha_M^j(x) \to \exists y. \big(\varphi_{same}^{int}(x,y) \wedge \gamma_M(y) \big) \right)$$

$$\varphi_{\alpha}^4(x) := \bigwedge_{\substack{i \in [1,M] \\ j \in [1,M+1]}} \left(\alpha_i^j(x) \to \forall y. \left(\begin{pmatrix} \neg \mathsf{ed}(y) \wedge \varphi_{same}(x,y) \\ \wedge\ x \ {}_1{\sim}_1\ y \wedge \neg (x \ {}_2{\sim}_2\ y) \end{pmatrix} \to \bigwedge_{k \in [1,M]} \neg \alpha_k^j(y) \right) \right)$$

$$\varphi^5_{\alpha}(x) := \bigwedge_{\substack{i\in[1,M]\\ j\in[2,M+2]}} \left(\alpha^j_i(x) \to \exists y.\big(\varphi_{same}(x,y) \wedge x \,{}_1{\sim}_1\, y \wedge \bigvee_{k\in[1,M]} \alpha^{j-1}_k(y)\big)\right)$$

We then define $\varphi_{\alpha} := \forall x.\big((\neg\mathsf{ed}(x)) \to (\varphi^1_{\alpha}(x) \wedge \varphi^2_{\alpha}(x) \wedge \varphi^3_{\alpha}(x) \wedge \varphi^4_{\alpha}(x) \wedge \varphi^5_{\alpha}(x))\big) \wedge (\mathsf{ed}(x) \to \bigwedge_{\alpha\in\aleph^{\alpha}_M} \neg\alpha(x))$. Note that φ_{α} is a two-variable formula in $\mathsf{FO}^2_{2\mathbb{DMS}}[\Theta \cup \{\mathsf{ed}\} \cup \aleph_M; \mathcal{S}_2]$. If a data structure satisfies φ_{α}, then no diagonal element is labelled with predicates in $\aleph^{\alpha}_M$ and all its non-diagonal elements are labelled with exactly one predicate in $\aleph^{\alpha}_M$ (see φ^1_{α}). Furthermore, all non-diagonal elements in a same intersection are labelled with the same α^j_i (see φ^2_{α}), and there are exactly i such elements in the intersection if $i \leq M-1$ and at least M otherwise (see φ^3_{α}). Finally, we want to identify up to $M+2$ different intersections sharing the same first value and we use the j in α^j_i for this matter. Formula φ^4_{α} tells us that no two non-diagonal elements with the same labels of Σ share the same index j (for $j \leq M+1$) if they do not belong to the same intersection and have the same first value. The formula φ^5_{α} specifies that, if an element a is labelled with α^j_i, then there are at least j different nonempty intersections with the same labels of Σ as a sharing the same first values. The next lemma formalizes the property of this labelling.

Lemma 4.2.15. *We consider $\mathfrak{A} = (A, (P_\sigma)_\sigma, f_1, f_2) \in \mathrm{Str}(\Sigma \cup \{\mathsf{eq}\} \cup \mathtt{C}_M \cup \aleph_M; 2\mathbb{DMS})$ eq-respecting and such that $\mathfrak{A} + \mathsf{ed} \models \varphi_{\gamma} \wedge \varphi_{\alpha}$ and $a \in A$. Let $S_{a,{}_1\sim_1} = \{b \in A \mid a \,{}_1{\sim}^{\mathfrak{A}}_1\, b \wedge \ell_\Sigma(a) = \ell_\Sigma(b)\}$ and $S^j_{a,{}_1\sim_1,i} = S_{a,{}_1\sim_1} \cap P_{\alpha^j_i}$ for all $i \in [1,M]$ and $j \in [1,M+2]$. The following properties hold:*

1. *We have $S_{a,{}_1\sim_1} = \bigcup_{i\in[1,M],j\in[1,M+2]} S^j_{a,{}_1\sim_1,i}$.*
2. *For all $j,\ell \in [1,M+2]$ and $i,k \in [1,M]$ such that $i \neq k$ or $j \neq l$, we have $S^j_{a,{}_1\sim_1,i} \cap S^{\ell}_{a,{}_1\sim_1,k} = \emptyset$.*
3. *For all $j \in [1,M+1]$ and $i \in [1,M]$ such that $b,c \in S^j_{a,{}_1\sim_1,i}$, we have $b \,{}_2{\sim}_2\, c$.*
4. *For all $b,c \in S_{a,{}_1\sim_1}$ such that $b \,{}_2{\sim}_2\, c$, there exist $j \in [1,M+2]$ and $i \in [1,M]$ such that $b,c \in S^j_{a,{}_1\sim_1,i}$.*
5. *For all $j \in [1,M+2]$ and $i \in [1,M]$ such that $b \in S^j_{a,{}_1\sim_1,i}$, we have*

$$\begin{cases} |\{c \in A \mid b \,{}_1{\sim}^{\mathfrak{A}}_1\, c \wedge b \,{}_2{\sim}^{\mathfrak{A}}_2\, c \wedge \ell_\Sigma(b) = \ell_\Sigma(c)\}| = i & \text{if } i \leq M-1 \\ |\{c \in A \mid b \,{}_1{\sim}^{\mathfrak{A}}_1\, c \wedge b \,{}_2{\sim}^{\mathfrak{A}}_2\, c \wedge \ell_\Sigma(b) = \ell_\Sigma(c)\}| \geq M & \text{otherwise.} \end{cases}$$

6. *For all $j \in [1,M+1]$, there exists at most one i such that $S^j_{a,{}_1\sim_1,i} \neq \emptyset$.*
7. *For all $j \in [2,M+2]$ and $i \in [1,M]$ such that $S^j_{a,{}_1\sim_1,i} \neq \emptyset$, there exists $k \in [1,M]$ such $S^{j-1}_{a,{}_1\sim_1,k} \neq \emptyset$.*

Proof: We prove the different statements:

1. Thanks to the formula $\varphi^1_{\alpha}(x)$ we have that $A = \bigcup_{i\in[1,M],j\in[1,M+2]} P_{\alpha^j_i}$. Since $S_{a,{}_1\sim_1} =$

$A \cap S_{a,{}_1\sim_1}$, we deduce that

$$S_{a,{}_1\sim_1} = \left(\bigcup_{i\in[1,M], j\in[1,M+2]} P_{\alpha_i^j} \right) \cap S_{a,{}_1\sim_1} = \bigcup_{i\in[1,M], j\in[1,M+2]} S^j_{a,{}_1\sim_1,i} .$$

2. This point can be directly deduced thanks to $\varphi^1_\alpha(x)$.

3. This point can be directly deduced thanks to $\varphi^4_\alpha(x)$.

4. Since $b \in S_{a,{}_1\sim_1}$, by 1. there exist $j \in [1, M+2]$ and $i \in [1, M]$ such that $b \in S^j_{a,{}_1\sim_1,i}$. Furthermore, since $c \in S_{a,{}_1\sim_1}$, using formula $\varphi^2_\alpha(x)$, we deduce that $c \in S^j_{a,{}_1\sim_1,i}$.

5. This point can be directly deduced thanks to formula $\varphi^3_\alpha(x)$ and to Lemma 4.2.14.

6. Assume there exist $i, i' \in [1, M]$ such that $i \neq i'$ and $S^j_{a,{}_1\sim_1,i} \neq \emptyset$ and $S^j_{a,{}_1\sim_1,i'} \neq \emptyset$. Let $b \in S^j_{a,{}_1\sim_1,i}$ and $c \in S^j_{a,{}_1\sim_1,i'} \neq \emptyset$. If $b \; {}_2\sim_2^{\mathfrak{A}} \; c$, then, by 5., we necessarily have $i = i'$. Hence we deduce that $b \; {}_2\sim_2^{\mathfrak{A}} \; c$ does not hold, and we can conclude thanks to formula $\varphi^4_\alpha(x)$.

7. This point can be directly deduced thanks to formula $\varphi^5_\alpha(x)$. □

While the predicates α_i^j deal with the relation ${}_1\sim_1$, we now define a similar formula $\varphi_\beta \in \mathsf{FO}^2[\Theta \cup \{\mathsf{ed}\} \cup \aleph_M; \mathcal{S}_2]$ for the predicates in $\aleph_M^\beta$ to count intersections connected by the binary relation ${}_2\sim_2$. We introduce hence the following formulas (where $\varphi^{int}_{same} = x \; {}_1\sim_1 \; y \wedge x \; {}_2\sim_2 \; y \wedge \bigwedge_{\sigma\in\Sigma} \sigma(x) \leftrightarrow \sigma(y)$ and $\varphi_{same} = \bigwedge_{\sigma\in\Sigma} \sigma(x) \leftrightarrow \sigma(y)$):

$$\varphi^1_\beta(x) := \bigvee_{\substack{i\in[1,M]\\ j\in[1,M+1]}} \left(\beta_i^j(x) \wedge \bigwedge_{\substack{k\in[1,M]\\ \ell\in[1,M+1]\\ (k,\ell)\neq(i,j)}} \neg\beta_k^\ell(x) \right)$$

$$\varphi^2_\beta(x) := \bigwedge_{\substack{i\in[1,M]\\ j\in[1,M+1]}} \left(\beta_i^j(x) \rightarrow \forall y. \big((\neg\mathsf{ed}(y) \wedge \varphi^{int}_{same}(x,y)) \rightarrow \beta_i^j(y) \big) \right)$$

$$\varphi^3_\beta(x) := \bigwedge_{\substack{i\in[1,M-1]\\ j\in[1,M+1]}} \left(\beta_i^j(x) \rightarrow \begin{pmatrix} \exists y. \big(\varphi^{int}_{same}(x,y) \wedge \gamma_i(y)\big) \\ \wedge \neg\exists y. \big(\varphi^{int}_{same}(x,y) \wedge \gamma_{i+1}(y)\big) \end{pmatrix} \right) \wedge$$
$$\bigwedge_{j\in[1,M+1]} \left(\beta_M^j(x) \rightarrow \exists y. \big(\varphi^{int}_{same}(x,y) \wedge \gamma_M(y)\big) \right)$$

$$\varphi^4_\beta(x) := \bigwedge_{\substack{i\in[1,M]\\ j\in[1,M]}} \left(\beta_i^j(x) \rightarrow \forall y. \left(\begin{pmatrix} \neg\mathsf{ed}(y) \wedge \varphi_{same}(x,y) \\ \wedge \neg(x \; {}_1\sim_1 \; y) \wedge x \; {}_2\sim_2 \; y \end{pmatrix} \rightarrow \bigwedge_{k\in[1,M]} \neg\beta_k^j(y) \right) \right)$$

$$\varphi^5_{\beta}(x) := \bigwedge_{\substack{i\in[1,M]\\ j\in[2,M+1]}} \left(\beta^j_i(x) \to \exists y.\left(\varphi_{same}(x,y) \wedge x \;{}_2{\sim}_2\; y \wedge \bigvee_{k\in[1,M+1]} \beta^{j-1}_k(y)\right)\right)$$

We then define $\varphi_\beta := \forall x.\big((\neg\mathsf{ed}(x)) \to (\varphi^1_\beta(x) \wedge \varphi^2_\beta(x) \wedge \varphi^3_\beta(x) \wedge \varphi^4_\beta(x))\big) \wedge (\mathsf{ed}(x) \to \bigwedge_{\beta\in\aleph^\beta_M} \neg\beta(x))$.

The following Lemma is the equivalent of the Lemma 4.2.15 for the relation ${}_2{\sim}_2$. Its proof is similar to the one of the Lemma 4.2.15.

Lemma 4.2.16. *We consider* $\mathfrak{A} = (A, (P_\sigma)_\sigma, f_1, f_2) \in \mathrm{Str}(\Sigma \cup \{\mathsf{eq}\} \cup \mathtt{C}_M \cup \aleph_M; 2\mathbb{DMS})$ *eq-respecting and such that* $\mathfrak{A} + \mathsf{ed} \models \varphi_\gamma \wedge \varphi_\beta$ *and* $a \in A$*. Let* $S_{a,{}_2{\sim}_2} = \{b \in A \mid a \;{}_2{\sim}_2^{\mathfrak{A}}\; b \wedge \ell_\Sigma(a) = \ell_\Sigma(b)\}$ *and* $S^j_{a,{}_2{\sim}_2,i} = S_{a,{}_2{\sim}_2} \cap P_{\beta^j_i}$ *for all* $i \in [1,M]$ *and* $j \in [1,M+1]$*. The following statements hold:*

1. *We have* $S_{a,{}_2{\sim}_2} = \bigcup_{i\in[1,M],j\in[1,M+1]} S^j_{a,{}_2{\sim}_2,i}$.
2. *For all* $j,\ell \in [1,M+1]$ *and* $i,k \in [1,M]$ *such that* $i \neq k$ *or* $j \neq l$*, we have* $S^j_{a,{}_2{\sim}_2,i} \cap S^\ell_{a,{}_2{\sim}_2,k} = \emptyset$.
3. *For all* $j \in [1,M+1]$ *and* $i \in [1,M]$ *such that* $b,c \in S^j_{a,{}_2{\sim}_2,i}$*, we have* $b \;{}_1{\sim}_1\; c$.
4. *For all* $b,c \in S_{a,{}_2{\sim}_2}$ *such that* $b \;{}_1{\sim}_1\; c$*, there exists* $j \in [1,M+1]$ *and* $i \in [1,M]$ *such that* $b,c \in S^j_{a,{}_2{\sim}_2,i}$.
5. *For all* $j \in [1,M+1]$ *and* $i \in [1,M]$ *such that* $b \in S^j_{a,{}_2{\sim}_2,i}$*, we have*

$$\begin{cases} |\{c \in A \mid b\,{}_1{\sim}_1^{\mathfrak{A}}\, c \wedge b\,{}_2{\sim}_2^{\mathfrak{A}}\, c \wedge \ell_\Sigma(b) = \ell_\Sigma(c)\}| = i & \text{if } i \leq M-1 \\ |\{c \in A \mid b\,{}_1{\sim}_1^{\mathfrak{A}}\, c \wedge b\,{}_2{\sim}_2^{\mathfrak{A}}\, c \wedge \ell_\Sigma(b) = \ell_\Sigma(c)\}| \geq M & \text{otherwise.} \end{cases}$$

6. *For all* $j \in [1,M]$*, there exists at most one* i *such that* $S^j_{a,{}_2{\sim}_2,i} \neq \emptyset$.
7. *For all* $j \in [2,M+1]$ *and* $i \in [1,M]$ *such that* $S^j_{a,{}_2{\sim}_2,i} \neq \emptyset$*, there exists* $k \in [1,M]$ *such* $S^{j-1}_{a,{}_2{\sim}_2,k} \neq \emptyset$.

Now that we can rely on a consistent labeling with predicates from $\aleph_M$, let us see how we can exploit it to express $\wr U, R, m \wr \in \mathtt{C}_M$, with additional help from Lemma 4.2.10, as a formula $\varphi_{U,R,m}(x) \in \mathsf{FO}^2_{2\mathbb{DMS}}[\Theta \cup \{\mathsf{eq},\mathsf{ed}\} \cup \aleph_M; \mathcal{S}_2]$ applied to *non-diagonal* elements (outside P_{ed}). Hereby, we will use, for $U \subseteq \Sigma$, the formula $\varphi_U(y) = \bigwedge_{\sigma\in U} \sigma(y) \wedge \bigwedge_{\sigma\in\Sigma\setminus U} \neg\sigma(y)$. We now provide the definition of the formulas $\varphi_{U,R,m}$ using a case analysis on the shape of R and the result of Lemma 4.2.10:

1. **Case** $R = \{(1,1),(2,2),(1,2)\}$**:** We need to say that (i) the element a under consideration is in P_{eq}, and (ii) there is an intersection of size at least m (i..e., it contains a γ_m-labeled element) whose elements b satisfy $a \;{}_1{\sim}_1\; b$, $a \;{}_2{\sim}_2\; b$, and $\ell_\Sigma(b) = U$:

$$\varphi_{U,R,m}(x) \;:=\; \mathsf{eq}(x) \wedge \exists y.\big(\varphi_U(y) \wedge x \;{}_1{\sim}_1\; y \wedge x \;{}_2{\sim}_2\; y \wedge \gamma_m(y)\big)$$

2. **Case $R = \{(1,1),(2,2)\}$:**

$$\varphi_{U,R,m}(x) \quad := \quad \neg\mathsf{eq}(x) \wedge \exists y.(\varphi_U(y) \wedge x \;{}_1{\sim}_1\; y \wedge x \;{}_2{\sim}_2\; y \wedge \gamma_m(y))$$

3. **Case $R = \{(1,1),(1,2)\}$:**

$$\varphi_{U,R,m}(x) \quad := \quad \neg\mathsf{eq}(x) \wedge \exists y.(\varphi_U(y) \wedge \mathsf{eq}(y) \wedge x \;{}_1{\sim}_1\; y \wedge \gamma_m(y))$$

4. **Case $R = \{(2,2),(1,2)\}$:** For this case, we first need an extra definition. For $m \in [1,M]$, we define $\mathcal{S}_{\beta,m}$ the set of subsets of $\aleph_M^{\alpha}$ as follows: $\mathcal{S}_{\beta,m} = \{\{\beta_{i_1}^{j_1},\ldots,\beta_{i_k}^{j_k}\} \mid i_1 + \ldots + i_k \geq m \text{ and } j_1 < j_2 < \ldots < j_k\}$. It corresponds to the sets of element β_i^j whose sum of i is greater than or equal to m. We then have:

$$\varphi_{U,R,m}(x) \quad := \quad \mathsf{eq}(x) \wedge \bigvee_{S\in\mathcal{S}_{\beta,m}} \bigwedge_{\beta\in S} \exists y.\big(\varphi_U(y) \wedge \neg\mathsf{eq}(y) \wedge \beta(y) \wedge x \;{}_2{\sim}_2\; y\big)$$

5. **Case $R = \{(2,2)\}$:** we use again the set $\mathcal{S}_{\beta,m}$ introduced previously.

$$\varphi_{U,R,m}(x) \quad := \quad \neg\mathsf{eq}(x) \wedge \bigvee_{S\in\mathcal{S}_{\beta,m}} \bigwedge_{\beta\in S} \exists y.\big(\varphi_U(y) \wedge \beta(y) \wedge \neg(x \;{}_1{\sim}_1\; y) \wedge x \;{}_2{\sim}_2\; y\big)$$

6. **Case $R = \{(1,1)\}$:** Similar to Case 4., we first need an extra definition. For $m \in \{1,\ldots,M\}$, we define the set $\mathcal{S}_{\alpha,m}$ of subsets of $\aleph_M^{\alpha}$ as follows: $\mathcal{S}_{\alpha,m} = \{\{\alpha_{i_1}^{j_1},\ldots,\alpha_{i_k}^{j_k}\} \mid i_1 + \ldots + i_k \geq m \text{ and } j_1 < j_2 < \ldots < j_k\}$. It corresponds to the sets of elements α_i^j whose sum of i is greater than or equal to m. We then have:

$$\varphi_{U,R,m}(x) := \bigvee_{S\in\mathcal{S}_{\alpha,m}} \bigwedge_{\alpha\in S} \exists y.\big(\varphi_U(y) \wedge \alpha(y) \wedge \neg\mathsf{eq}(y) \wedge x \;{}_1{\sim}_1\; y \wedge \neg(x \;{}_2{\sim}_2\; y)\big)$$

7. **Case $R = \{(1,2)\}$:** We use here again the set $\mathcal{S}_{\beta,m}$ introduced in Case 4.

$$\begin{aligned}\varphi_{U,R,m}(x) \quad := \quad & \neg\mathsf{eq}(x) \wedge \exists y.\Bigg(\mathsf{ed}(y) \wedge x \;{}_1{\sim}_1\; y \wedge \\ & \bigvee_{S\in\mathcal{S}_{\beta,m}} \bigwedge_{\sigma\in S} \exists x.\big(\varphi_U(x) \wedge \sigma(x) \wedge \neg(y \;{}_1{\sim}_1\; x) \wedge y \;{}_2{\sim}_2\; x\big)\Bigg)\end{aligned}$$

Finally, it remains to say that all elements are labeled with the suitable counting constraints. So we let $\varphi_{cc} = \forall x.\neg\mathsf{ed}(x) \rightarrow \bigwedge_{\wr U,R,m\int \in \mathtt{C}_M} \wr U,R,m\int(x) \leftrightarrow \varphi_{U,R,m}(x)$.

Lemma 4.2.17. *Let $\mathfrak{A} = (A,(P_\sigma)_\sigma,f_1,f_2) \in \mathrm{Str}(\Sigma \cup \{\mathsf{eq}\} \cup \mathtt{C}_M \cup \aleph_M; 2\mathbb{DMS})$ be eq-respecting. If $\mathfrak{A} + \mathsf{ed} \models \varphi_\alpha \wedge \varphi_\beta \wedge \varphi_\gamma \wedge \varphi_{cc}$, then $\mathfrak{A}$ is cc-respecting.*

Sketch of proof: Let $\mathfrak{A} = (A,(P_\sigma)_\sigma,f_1,f_2) \in \mathrm{Str}(\Sigma\cup\{\mathsf{eq}\}\cup\mathtt{C}_M\cup\aleph_M; 2\mathbb{DMS})$ be eq-respecting and such that $\mathfrak{A} + \mathsf{ed} \models \varphi_\alpha \wedge \varphi_\beta \wedge \varphi_\gamma \wedge \varphi_{cc}$. We need to show that for all $a \in A$ and all $\wr U,R,m\int \in \mathtt{C}_M$, we have $a \in P_{\wr U,R,m\int}$ iff $|\mathsf{Env}_{\mathfrak{A},\Sigma,\mathcal{I}_2}(a,U,R)| \geq m$. We consider $a \in A$.

Since $\mathfrak{A} + \mathsf{ed} \models \varphi_{cc}$, we deduce that $a \in P_{\wr U,R,m \wr}$ iff $\mathfrak{A} + \mathsf{ed} \models_{I[x/a]} \varphi_{U,R,m}(x)$. We need hence to show that $\mathfrak{A} + \mathsf{ed} \models_{I[x/a]} \varphi_{U,R,m}(x)$ iff $|\mathtt{Env}_{\mathfrak{A},\Sigma,\mathcal{I}_2}(a, U, R)| \geq m$. To prove this , we first use Lemma 4.2.10 to get a characterization of $\mathtt{Env}_{\mathfrak{A},\Sigma,\mathcal{I}_2}(a, U, R)$. This characterization is then directly translated into the formula $\varphi_{U,R,m}(x)$ which makes use of the label in $\aleph_M$ to count in the environment of a. The fact that this counting is performed correctly is guaranteed by Lemmas 4.2.14,4.2.15 and 4.2.16. Putting these arguments together, we can conclude that the lemma holds. □

Step 5: Putting it All Together

Let $\mathsf{All} = \Sigma \cup \{\mathsf{eq}, \mathsf{ed}\} \cup \mathtt{C}_M \cup \aleph_M$ denote the set of all the unary predicates that we have introduced so far. Recall that, after Step 1, we were left with $M \geq 1$ and a formula $\varphi \in \mathsf{FO}_{2\mathbb{DMS}}[\Sigma \cup \{\mathsf{eq}\} \cup \mathtt{C}_M; \emptyset]$. The question is now whether φ has a well-typed model (i.e., a model that is eq-respecting and cc-respecting). Altogether, we get the following reduction:

Proposition 4.2.18. *Let $\varphi \in \mathsf{FO}_{2\mathbb{DMS}}[\Sigma \cup \{\mathsf{eq}\} \cup \mathtt{C}_M; \emptyset]$. Then, φ has a well-typed model iff $\widehat{\varphi} := [\![\varphi]\!]_{+\mathsf{ed}} \wedge \xi_{\mathsf{ed}}^{\mathsf{All}\setminus\{\mathsf{eq},\mathsf{ed}\}} \wedge \varphi_\alpha \wedge \varphi_\beta \wedge \varphi_\gamma \wedge \varphi_{cc} \in \text{ext-}\mathsf{FO}^2_{2\mathbb{DMS}}[\mathsf{All}; \mathcal{S}_2]$ is satisfiable.*

Proof: Suppose $\widehat{\varphi}$ is satisfiable. Then, there is $\mathfrak{B} \in \mathrm{Str}(\mathsf{All}; 2\mathbb{DMS})$ such that $\mathfrak{B} \models \widehat{\varphi}$. By Lemma 4.2.7, there exists an eq-respecting data structure $\mathfrak{A} \in \mathrm{Str}(\Sigma \cup \{\mathsf{eq}\} \cup \mathtt{C}_M \cup \aleph_M; 2\mathbb{DMS})$ such that $\mathfrak{A} + \mathsf{ed} \models [\![\varphi]\!]_{+\mathsf{ed}} \wedge \varphi_\alpha \wedge \varphi_\beta \wedge \varphi_\gamma \wedge \varphi_{cc}$. Using Lemma 4.2.17, we deduce that $\mathfrak{A}$ is cc-respecting and, thus, well-typed. Furthermore, by Lemma 4.2.9, we have $\mathfrak{A} \models \varphi$. Note that $\mathfrak{A}$ belongs to $\mathrm{Str}(\Sigma \cup \{\mathsf{eq}\} \cup \mathtt{C}_M \cup \aleph_M; 2\mathbb{DMS})$. However, by removing the unary predicates in $\aleph_M$, we still have a model of φ from $\mathrm{Str}(\Sigma \cup \{\mathsf{eq}\} \cup \mathtt{C}_M; 2\mathbb{DMS})$ as required. Hence, φ has a well-typed model.

Assume now that there exists a well-typed data structure $\mathfrak{A} \in \mathrm{Str}(\Sigma \cup \{\mathsf{eq}\} \cup \mathtt{C}_M; 2\mathbb{DMS})$ such that $\mathfrak{A} \models \varphi$. Using Lemma 4.2.9, we have that $\mathfrak{A} + \mathsf{ed} \models [\![\varphi]\!]_{+\mathsf{ed}}$. Furthermore, using the fact that $\mathfrak{A}$ is well-typed, we can add the unary predicates from $\aleph_M$ to $\mathfrak{A} + \mathsf{ed}$ to obtain a data structure $\mathfrak{A}'$ in $\mathrm{Str}(\mathsf{All}; 2\mathbb{DMS})$ such that $\mathfrak{A}' \models \varphi_\alpha \wedge \varphi_\beta \wedge \varphi_\gamma \wedge \varphi_{cc}$. Note that $\mathfrak{A}'$ is well-diagonalized. We deduce that $\mathfrak{A}' \models \widehat{\varphi}$. □

Theorem 4.2.19

$2\mathbb{DMS}$-SAT$(1\text{-}\mathsf{LF}^{\text{int}}; \{(1,1),(2,2),(1,2)\})$ is decidable.

Proof: Let $\psi \in 1\text{-}\mathsf{LF}_2^{\text{int}}[\Sigma; (1,1),(2,2),(1,2)]$. Using Lemma 4.2.4, we can effectively compute $M \in \mathbb{N}$ and $\varphi \in \mathsf{FO}_{2\mathbb{DMS}}[\Sigma \cup \{\mathsf{eq}\} \cup \mathtt{C}_M; \emptyset]$ such that ψ is satisfiable iff φ has a well-typed model. By Proposition 4.2.18, φ has a well-typed model iff $\widehat{\varphi}$ is satisfiable. Since $\widehat{\varphi}$ belongs to ext-$\mathsf{FO}^2_{2\mathbb{DMS}}[\mathsf{All}; \mathcal{S}_2]$, we conclude using Proposition 4.1.2. □

4.3 Undecidability Results

We now show that extending the neighborhood radius yields undecidability. The starting point of our reduction is the domino problem. We already exposed the domino problem in

Section 2.2.2, although it was more general. We first recall some definition and notation so it is not mandatory to read Section 2.2.2 in order to understand the following section.

The Tiling Problem. A *domino system* $\mathcal{D}$ is a triple (D, H, V) where D is a finite set of dominoes and $H, V \subseteq D \times D$ are two binary relations. Let $\mathfrak{G}_m$ denote the standard grid on an $m \times m$ torus, i.e., $\mathfrak{G}_m = (G_m, H_m, V_m)$ where H_m and V_m are two binary relations defined as follows: $G_m = \mathbb{Z} \bmod m \times \mathbb{Z} \bmod m$, $H_m = \{((i,j),(i',j)) \mid i' - i \equiv 1 \mod m\}$, and $V_m = \{((i,j),(i,j')) \mid i' - i \equiv 1 \mod m\}$. In the sequel, we will suppose $\mathbb{Z} \bmod m = \{0, \ldots, m-1\}$ using the least positive member to represent residue classes.

A *bi-binary structure* is a triple (A, R_1, R_2) where A is a finite set and R_1, R_2 are subsets of $A \times A$. Domino systems and $\mathfrak{G}_m$ for any m are examples of bi-binary structures. For two bi-binary structures $\mathfrak{G} = (G, H, V)$ and $\mathfrak{G}' = (G', H', V')$, we say that $\mathfrak{G}$ is *homomorphically embeddable* into $\mathfrak{G}'$ if there is a morphism $\pi : \mathfrak{G} \to \mathfrak{G}'$, i.e., a mapping π such that, for all $a, a' \in G$, $(a, a') \in H \Rightarrow (\pi(a), \pi(a')) \in H'$ and $(a, a') \in V \Rightarrow (\pi(a), \pi(a')) \in V'$. For instance, $\mathfrak{G}_{k \cdot m}$ is homomorphically embeddable into $\mathfrak{G}_m$ through reduction mod m. For a domino system $\mathcal{D}$, a *periodic tiling* is a morphism $\tau : \mathfrak{G}_m \to \mathcal{D}$ for some m and we say that $\mathcal{D}$ *admits a periodic tiling* if there exists a periodic tiling of $\mathcal{D}$.

The problem UNBOUNDED-TILING (or *periodic tiling problem*), which is well known to be undecidable [12], is defined as follows: Given a domino system $\mathcal{D}$, does $\mathcal{D}$ admit a periodic tiling?

To use UNBOUNDED-TILING in our reductions, we first use some specific bi-binary structures, which we call grid-like and which are easier to manipulate in our context to encode domino systems. A bi-binary structure $\mathfrak{G} = (A, H, V)$ is said to be *grid-like* if some $\mathfrak{G}_m$ is homomorphically embeddable into $\mathfrak{G}$. The logic FO *over bi-binary structures* refers to the first-order logic on two binary relations H, V, and we write $\mathsf{H}xy$ to say that x and y are in relation for H. Consider the two following FO formulas over bi-binary structures: $\varphi_{complete} = \forall x.\forall y.\forall x'.\forall y'.((\mathsf{H}xy \wedge \mathsf{V}xx' \wedge \mathsf{V}yy') \to \mathsf{H}x'y')$ and $\varphi_{progress} = \forall x.(\exists y.\mathsf{H}xy \wedge \exists y.\mathsf{V}xy)$. The following lemma, first stated and proved in [42], shows that these formulas suffice to characterize grid-like structures:

Lemma 4.3.1 ([42]). *Let $\mathfrak{G} = (A, H, V)$ be a bi-binary structure. If $\mathfrak{G}$ satisfies $\varphi_{complete}$ and $\varphi_{progress}$, then $\mathfrak{G}$ is grid-like.*

Given $\mathfrak{A} = (A, (P_\sigma), f_1, f_2) \in \mathrm{Str}(\Sigma; 2\mathbb{DMS})$ and $\varphi(x,y) \in \mathsf{FO}_{2\mathbb{DMS}}[\Sigma; \mathcal{R}]$, we define the binary relation $[\![\varphi]\!]_{\mathfrak{A}} = \{(a,b) \in A \times A \mid \mathfrak{A} \models \varphi(a,b)\}$. Thus, given two $\mathsf{FO}_{2\mathbb{DMS}}[\Sigma; \mathcal{R}]$ formulas $\varphi_1(x,y), \varphi_2(x,y)$ with two free variables, $(A, [\![\varphi_1]\!]_{\mathfrak{A}}, [\![\varphi_2]\!]_{\mathfrak{A}})$ is a bi-binary structure.

As we want to reason on data structures, we build a data structure $\mathfrak{A}_{2m}$ that corresponds to the grid $\mathfrak{G}_{2m} = (G_{2m}, H_{2m}, V_{2m})$. This structure is depicted locally in Figure 4.4. To define $\mathfrak{A}_{2m}$, we use four unary predicates given by $\Sigma_{grid} = \{X_0, X_1, Y_0, Y_1\}$. They give us access to the coordinate modulo 2. We then define $\mathfrak{A}_{2m} = (G_{2m}, (P_\sigma), f_1, f_2) \in \mathrm{Str}(\Sigma_{grid}; 2\mathbb{DMS})$ as follows: For $k \in \{0,1\}$, we have $P_{X_k} = \{(i,j) \in G_{2m} \mid i \equiv k \mod 2\}$ and $P_{Y_k} = \{(i,j) \in G_{2m} \mid j \equiv k \mod 2\}$. For all $i, j \in \{0, \ldots, 2m-1\}$, we set $f_1(i,j) = ((i/2) \mod m) + m * ((j/2) \mod m)$ (where / stands for the Euclidian division). Finally, for all $i, j \in \{1, \ldots, 2m\}$, set $f_2(i \mod (2m), j \mod (2m)) = f_1(i-1, j-1)$.

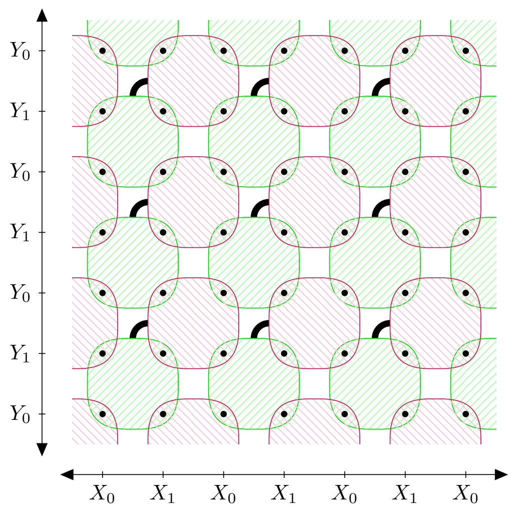


Figure 4.4: The local pattern of $\mathfrak{A}_{2m}$. Dots denote elements. Two dots are in the same ${}_1{\sim}_1$-equivalence class (resp. ${}_2{\sim}_2$) iff they are in the same green (resp. purple) area. The thick black lines represent the relation ${}_1{\sim}_2$ in the following way: if a ${}_1{\sim}_1$-equivalence class C_1 and a ${}_2{\sim}_2$-equivalence class C_2 are connected with a thick black line, then for any $a \in C_1$ and $b \in C_2$, we have $a \ {}_1{\sim}_2 \ b$.

We define the quantifier free formulas $\varphi_H(x, y)$ from the logic $\mathsf{FO}_{2\mathbb{DMS}}[\Sigma_{grid}; \mathcal{S}_2]$ with two free variable:

$$
\begin{aligned}
\varphi_H^{00} &= X_0(x) \wedge X_1(y) \wedge Y_0(x) \wedge Y_0(y) \wedge x \ {}_1{\sim}_1 \ y,\\
\varphi_H^{10} &= X_1(x) \wedge X_0(y) \wedge Y_0(x) \wedge Y_0(y) \wedge x \ {}_2{\sim}_2 \ y,\\
\varphi_H^{01} &= X_0(x) \wedge X_1(y) \wedge Y_1(x) \wedge Y_1(y) \wedge x \ {}_1{\sim}_1 \ y,\\
\varphi_H^{11} &= X_1(x) \wedge X_0(y) \wedge Y_1(x) \wedge Y_1(y) \wedge x \ {}_2{\sim}_2 \ y,\\
\varphi_H &= \varphi_H^{00} \vee \varphi_H^{10} \vee \varphi_H^{01} \vee \varphi_H^{11}.
\end{aligned}
$$

Similarly, we define $\varphi_V(x, y)$ from $\mathsf{FO}_{2\mathbb{DMS}}[\Sigma_{grid}; \mathcal{S}_2]$ with two free variable and quantifier free:

$$
\begin{aligned}
\varphi_V^{00} &= X_0(x) \wedge X_0(y) \wedge Y_0(x) \wedge Y_1(y) \wedge x \ {}_1{\sim}_1 \ y,\\
\varphi_V^{10} &= X_1(x) \wedge X_1(y) \wedge Y_0(x) \wedge Y_1(y) \wedge x \ {}_1{\sim}_1 \ y,\\
\varphi_V^{01} &= X_0(x) \wedge X_0(y) \wedge Y_1(x) \wedge Y_0(y) \wedge x \ {}_2{\sim}_2 \ y,\\
\varphi_V^{11} &= X_1(x) \wedge X_1(y) \wedge Y_1(x) \wedge Y_0(y) \wedge x \ {}_2{\sim}_2 \ y,\\
\varphi_V &= \varphi_V^{00} \vee \varphi_V^{10} \vee \varphi_V^{01} \vee \varphi_V^{11}.
\end{aligned}
$$

These formulas allow us to make the link between the data structure $\mathfrak{A}_{2m}$ and the grid $\mathfrak{G}_{2m}$, and we will use them later on to ensure that a data structure has a shape 'similar' to $\mathfrak{A}_{2m}$.

Remark 4.3.2. Note that, using the definitions of G_{2m} and of $\mathfrak{A}_{2m}$ we can show that, if $\mathfrak{G}$ is the bi-binary structure $(G_{2m}, [\![\varphi_H]\!]_{\mathfrak{A}_{2m}}, [\![\varphi_V]\!]_{\mathfrak{A}_{2m}})$, then $\mathfrak{G}_{2m} = \mathfrak{G}$. $\diamond$

Proof: We have hence to prove that $H_{2m} = \{(a, b) \mid \mathfrak{A}_{2m} \models \varphi_H(a, b)\}$ and $V_{2m} = \{(a, b) \mid \mathfrak{A}_{2m} \models \varphi_H(a, b)\}$. We first show that $H_{2m} \subseteq [\![\varphi_H]\!]_{\mathfrak{A}_{2m}}$. Let $((i, j), (i', j')) \in H_{2m}$. Hence we have $j = j'$ and $i' - i \equiv 1 \mod 2m$. We have then different cases according to the parity of

j, i and i'. Assume i, j are even. Then $(i,j),(i',j') \in P_{Y_0}$ and $(i,j) \in P_{X_0}$ and $(i',j) \in P_{X_1}$ and by definition of f_1, we have $f_1(i,j) = f_1(i',j)$, hence $((i,j),(i',j)) \in [\![\varphi_H^{00}]\!]_{\mathfrak{A}_{2m}}$ and $((i,j),(i',j)) \in [\![\varphi_H]\!]_{\mathfrak{A}_{2m}}$. The three other cases can be treated similarly.

We now prove that $H_{2m} \supseteq [\![\varphi_H]\!]_{\mathfrak{A}_{2m}}$. Let (a,b) be such that $\mathfrak{A}_{2m} \models \varphi_H(a,b)$. For φ_H to hold on (a,b), one of the φ_H^{ij} must hold. We treat the case $\mathfrak{A}_{2m} \models \varphi_H^{11}(a,b)$. Write (a_1,a_2) and (b_1,b_2) the coordinates of a and b respectively. As $a \in P_{X_0} \cap P_{Y_0}$ and $b \in P_{X_1} \cap P_{Y_0}$, we have that a_1, a_2, b_2 are even and b_1 is odd. As $a \;{}_1{\sim}_1\; b$, we have $((a_1/2) \mod m) + m * ((a_2/2) \mod m) = ((b_1/2) \mod m) + m * ((b_2/2) \mod m)$. This allows us to conclude that $a_2 = b_2$ and that $a_1 - b_2 \equiv 1 \mod m$. So we have $(a,b) \in H_{2m}$. The other cases can be treated in a similar way.

The proof that $V_{2m} = [\![\varphi_V]\!]_{\mathfrak{A}_{2m}}$ follows the exact same lines. $\square$

The Reduction to Radius 3. We first use the notions we introduced previously to show that $2\mathbb{DMS}$-SAT(3-$\mathsf{LF}_2^{\text{int}}$; $\{{}_1{\sim}_1, {}_2{\sim}_2\}$) is undecidable, hence we assume now that $\mathcal{R} = \{{}_1{\sim}_1, {}_2{\sim}_2\}$. The first step in our reduction from UNBOUNDED-TILING consists in defining $\varphi_{grid}^{\textit{3-loc}} \in$ 3-$\mathsf{LF}_2^{\text{int}}[\Sigma_{grid}; {}_1{\sim}_1, {}_2{\sim}_2]$ to check that a data structure corresponds to a grid ($\oplus$ stands for exclusive or):

$$\begin{aligned}
\varphi_{complete}^{\textit{3-loc}} &= \forall x. \langle\!\langle \forall y. \forall x'. \forall y'. \varphi_H(x,y) \wedge \varphi_V(x,x') \wedge \varphi_V(y,y') \rightarrow \varphi_H(x',y') \rangle\!\rangle_x^{3,\text{int}} \\
\varphi_{progress}^{\textit{3-loc}} &= \forall x. \langle\!\langle \exists y. \varphi_H(x,y) \wedge \exists y. \varphi_V(x,y) \rangle\!\rangle_x^{3,\text{int}} \\
\varphi_{grid}^{\textit{3-loc}} &= \varphi_{complete}^{\textit{3-loc}} \wedge \varphi_{progress}^{\textit{3-loc}} \wedge \forall x. \langle\!\langle (X_0(x) \oplus X_1(x)) \wedge (Y_0(x) \oplus Y_1(x)) \rangle\!\rangle_x^{3,\text{int}}
\end{aligned}$$

Lemma 4.3.3. *We have $\mathfrak{A}_{2m} \models \varphi_{grid}^{\textit{3-loc}}$. Moreover, for all structure $\mathfrak{A} = (A, (P_\sigma), f_1, f_2)$ in* $\text{Str}(\Sigma_{grid}; 2\mathbb{DMS})$*, if $\mathfrak{A} \models \varphi_{grid}^{\textit{3-loc}}$, then $(A, [\![\varphi_H]\!]_{\mathfrak{A}}, [\![\varphi_V]\!]_{\mathfrak{A}})$ is grid-like.*

Proof: We first show that $\mathfrak{A}_{2m} \models \varphi_{grid}^{\textit{3-loc}}$. In the proof, we assume that $m \geq 3$. The cases $m = 1$ or 2 are treated in the same way. Let us prove the first conjunct, that is $\mathfrak{A}_{2m} \models \varphi_{complete}^{\textit{3-loc}}$. Let $a \in G_{2m}$. We want to prove that

$$\mathfrak{A}_{2m}|_{a,\mathcal{S}_2}^{3,\text{int}} \models_{I[x/a]} \forall y. \forall x'. \forall y'. \varphi_H(x,y) \wedge \varphi_V(x,x') \wedge \varphi_V(y,y') \Rightarrow \varphi_H(x',y')$$

for some interpretation function I. We proceed by a case analysis on the values of $i,j \in \{0,1\}$ such that $a \in P_{X_i} \cap P_{Y_j}$. Assume that $(i,j) = (0,0)$. Then $\mathfrak{A}_{2m}|_{a,\mathcal{S}_2}^{3,\text{int}}$ is depicted in Figure 4.5a. Let b, a', b' such that

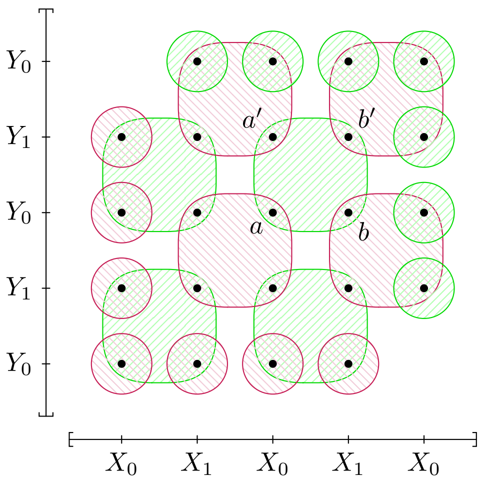
$$\mathfrak{A}_{2m}|_{a,\mathcal{S}_2}^{3,\text{int}} \models \varphi_H(a,b) \wedge \varphi_V(a,a') \wedge \varphi_V(b,b')\,.$$

We want to show

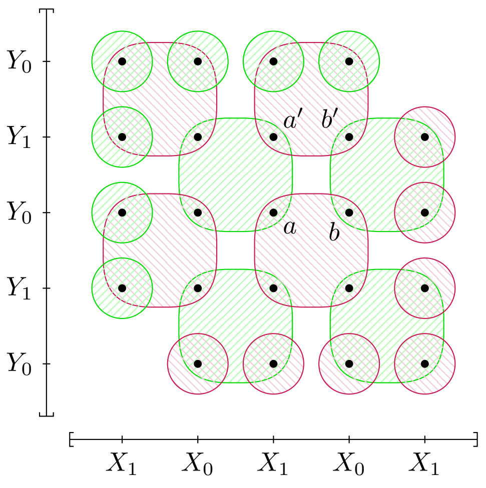
$$\mathfrak{A}_{2m}|_{a,\mathcal{S}_2}^{3,\text{int}} \models \varphi_H(a',b')\,.$$

By assumption on a and by looking at the definition of φ_H,

$$\mathfrak{A}_{2m}|_{a,\mathcal{S}_2}^{3,\text{int}} \models X_1(b) \wedge Y_0(b) \wedge a \;{}_1{\sim}_1\; b\,.$$



(a) $a \in P_{X_0} \cap P_{Y_0}$



(b) $a \in P_{X_1} \cap P_{Y_0}$

Figure 4.5: Some 3-local views of $\mathfrak{A}_{2m}$ for $\mathcal{R} = \{{}_1{\sim}_1, {}_2{\sim}_2\}$.

So by elimination we have that b is the element pointed by Figure 4.5a. In a similar way, a' and b' are indeed the elements pointed by Figure 4.5a. Hence, we deduce

$$\mathfrak{A}_{2m}|^{3,\mathrm{int}}_{a,\mathcal{S}_2} \models \varphi_H(a', b') .$$

The case $(i, j) = (1, 0)$ is depicted in Figure 4.5b and is proven in the same way just as the cases when $(i, j) = (1, 0)$ or $(i, j) = (1, 1)$.

Showing that $\mathfrak{A}_{2m} \models \varphi^{\textit{3-loc}}_{\textit{progress}}$ is done in the same way as showing that $\mathfrak{A}_{2m} \models \varphi^{\textit{3-loc}}_{\textit{complete}}$.

Finally, it is obvious that $\mathfrak{A}_{2m}$ satisfies the last conjunct of $\varphi^{\textit{3-loc}}_{\textit{grid}}$.

We now show that for all $\mathfrak{A} = (A, (P_\sigma), f_1, f_2)$ in $\mathrm{Str}(\Sigma_{grid}; 2\mathbb{DMS})$, if $\mathfrak{A} \models \varphi^{\textit{3-loc}}_{\textit{grid}}$ then $(A, [\![\varphi_H]\!]_{\mathfrak{A}}, [\![\varphi_V]\!]_{\mathfrak{A}})$ is grid-like. By Lemma 4.3.1, it suffices to prove that $(A, [\![\varphi_H]\!]_{\mathfrak{A}}, [\![\varphi_V]\!]_{\mathfrak{A}})$ satisfies $\varphi_{complete}$ and $\varphi_{progress}$. Let us prove that

$$(A, [\![\varphi_H]\!]_{\mathfrak{A}}, [\![\varphi_V]\!]_{\mathfrak{A}}) \models \forall x.\forall y.\forall x'.\forall y'.((\mathsf{H}xy \;\wedge\; \mathsf{V}xx' \;\wedge\; \mathsf{V}yy') \;\Rightarrow\; \mathsf{H}x'y') .$$

By definition of $(A, [\![\varphi_H]\!]_{\mathfrak{A}}, [\![\varphi_V]\!]_{\mathfrak{A}})$, this amounts to verifying that

$$\mathfrak{A} \models \forall x.\forall y.\forall x'.\forall y'.\varphi_H(x, y) \;\wedge\; \varphi_V(x, x') \;\wedge\; \varphi_V(y, y') \;\Rightarrow\; \varphi_H(x', y') .$$

Let $a, b, a', b' \in A$ such that $\mathfrak{A} \models \varphi_H(a, b) \wedge \varphi_V(a, a') \wedge \varphi_V(b, b')$. Let us show $\mathfrak{A} \models \varphi_H(a', b')$. We do a case analysis on $i, j \in \{0, 1\}$ such that $a \in P_{X_i} \cap P_{Y_j}$. We only perform the proof for the case $(i, j) = (1, 0)$, the other three case can be treated similarly. By looking at φ_H

and φ_V, we have

$$\begin{aligned}\mathfrak{A} &\models X_0(b) \wedge Y_0(b) \wedge a \,{}_2{\sim}_2\, b,\\ \mathfrak{A} &\models X_0(b') \wedge Y_1(b') \wedge b \,{}_1{\sim}_1\, b',\\ \mathfrak{A} &\models X_1(a') \wedge Y_1(a') \wedge a \,{}_1{\sim}_1\, a'.\end{aligned}$$

So b, a, b' are elements of $\mathfrak{A}|^{3,\mathrm{int}}_{a,\mathcal{S}_2}$ and

$$\mathfrak{A}|^{3,\mathrm{int}}_{a,\mathcal{S}_2} \models \varphi_H(a,b) \wedge \varphi_V(a,a') \wedge \varphi_V(b,b')\,.$$

Since by assumption $\mathfrak{A} \models \varphi^{\text{3-loc}}_{complete}$, we deduce that $\mathfrak{A}|^{3,\mathrm{int}}_{a,\mathcal{S}_2} \models \varphi_H(a',b')$. This allows us to conclude that $\mathfrak{A} \models \varphi_H(a',b')$.

We can prove in a similar way that $(A, [\![\varphi_H]\!]_{\mathfrak{A}}, [\![\varphi_V]\!]_{\mathfrak{A}}) \models \varphi_{progress}$ can be proved in a similar way. $\square$

Given a domino system $\mathcal{D} = (D, H_{\mathcal{D}}, V_{\mathcal{D}})$, we now provide a formula $\varphi_{\mathcal{D}}$ from the logic 3-$\mathsf{LF}_2^{\mathrm{int}}[D;\mathcal{S}_2]$ that guarantees that, if a data structure corresponding to a grid satisfies $\varphi_{\mathcal{D}}$, then it can be embedded into $\mathcal{D}$:

$$\begin{aligned}\varphi_{\mathcal{D}} \quad := \quad & \forall x.\langle\!\langle \bigvee_{d\in D}\Big(d(x) \wedge \bigwedge_{d\neq d'\in D} \neg(d(x) \wedge d'(x))\Big)\rangle\!\rangle^{3,\mathrm{int}}_x\\ & \wedge \forall x.\langle\!\langle \forall y.\varphi_H(x,y) \rightarrow \bigvee_{(d,d')\in H_{\mathcal{D}}} d(x) \wedge d'(y)\rangle\!\rangle^{3,\mathrm{int}}_x\\ & \wedge \forall x.\langle\!\langle \forall y.\varphi_V(x,y) \rightarrow \bigvee_{(d,d')\in V_{\mathcal{D}}} d(x) \wedge d'(y)\rangle\!\rangle^{3,\mathrm{int}}_x\end{aligned}$$

Proposition 4.3.4. *Given $\mathcal{D} = (D, H_{\mathcal{D}}, V_{\mathcal{D}})$ a domino system, $\mathcal{D}$ admits a periodic tiling iff the* 3-$\mathsf{LF}_2^{\mathrm{int}}[\Sigma_{grid} \uplus D;\mathcal{S}_2]$ *formula $\varphi^{\text{3-loc}}_{grid} \wedge \varphi_{\mathcal{D}}$ is satisfiable.*

Proof: First assume that $\mathcal{D}$ admits a periodic tiling and let $\tau : \mathfrak{G}_m \to D$ be one. As with Lemma 4.3.3 we already have that $\mathfrak{A}_{2m} \models \varphi^{\text{3-loc}}_{grid}$. From $\mathfrak{A}_{2m}$ we build another data structure $\mathfrak{A}'_{2m} \in \mathrm{Str}(\Sigma_{grid} \uplus D; 2\mathbb{DMS})$ by adding the predicates $(P_d)_{d\in D}$ as follow: for any $i,j \in \{0, 2m-1\}$ and $d \in \mathcal{D}$ we set $P_d((i,j))$ to hold iff $\tau((i \mod m, j \mod m)) = d$. We can then show that $\mathfrak{A}_{2m} \models \varphi_{\mathcal{D}}$.

Assume now that there exists $\mathfrak{A} = (A, (P_\sigma), f_1, f_2)$ in $\mathrm{Str}(\Sigma_{grid} \uplus D; 2\mathbb{DMS})$ such that $\mathfrak{A} \models \varphi^{\text{3-loc}}_{grid} \wedge \varphi_{\mathcal{D}}$. By Lemma 4.3.3, there exists $m > 0$ and a morphism $\pi : \mathfrak{G}_m \to (\mathfrak{A}, [\![\varphi_H]\!]_{\mathfrak{A}}, [\![\varphi_V]\!]_{\mathfrak{A}})$. It remains to show that there is a morphism $\tau : (\mathfrak{A}, [\![\varphi_H]\!]_{\mathfrak{A}}, [\![\varphi_V]\!]_{\mathfrak{A}}) \to \mathcal{D}$. For any $a \in A$, we set $\tau(a)$ to be a domino such that $P_{\tau(a)}(a)$ holds. Thanks to the first line of $\varphi_{\mathcal{D}}$, τ is well defined. Then thanks to the second and third line of $\varphi_{\mathcal{D}}$, we have that τ is a morphism. We deduce that $\tau \circ \pi$ is a periodic tiling of $\mathcal{D}$. $\square$

As a corollary of the proposition, we obtain the main result of this section.

Theorem 4.3.5

$2\mathbb{DMS}$-SAT(3-$\mathsf{LF}_2^{\mathrm{int}}$; $\{{}_1{\sim}_1, {}_2{\sim}_2\}$) *is undecidable.*

The Reduction to Radius 2. We also show that $2\mathbb{DMS}$-$\textsc{Sat}$(2-$\mathsf{LF}_2^{\text{int}}$; $\{_1{\sim}_1, {}_2{\sim}_2, {}_1{\sim}_2\}$) is undecidable. In that case, it is abit more subtle to build a formula similar to the formula $\varphi_{complete}$ as we have only neighborhood of radius 2, but we use the diagonal binary relation , $_1{\sim}_2$ to overcome this. We recall that $\mathcal{I}_2 = \{_1{\sim}_1, {}_2{\sim}_2, {}_1{\sim}_2\}$.

A *tri-binary structure* is a triple (A, H, V, W) where A is a set and H, V, W are three subsets of $A \times A$. Intuitively H, V will capture the horizontal and vertical adjacency relation whereas W will capture the diagonal adjacency. By an abuse of notation, $\mathfrak{G}_m$ will also refer to the tri-binary structure (G_m, H_m, V_m, W_m), were G_m, H_m and V_m are the same as before and:

$$W_m = \{((i,j),(i+1,j+1)) \mid i,j \in \mathbb{Z} \bmod m\}.$$

The logic FO *over tri-binary structure* is the same as FO over bi-binary structure with the addition of the binary symbol W. Let $\varphi'_{complete}$ be the following FO formula over tri-binary structure:

$$\varphi'_{complete} = \ \forall x.\forall y.\forall y'.(\mathsf{H}xy \wedge \mathsf{V}yy' \Rightarrow \mathsf{W}xy') \quad \wedge \quad \forall x.\forall x.\forall' y'.(\mathsf{W}xy' \wedge \mathsf{V}xx' \Rightarrow \mathsf{H}x'y')\,.$$

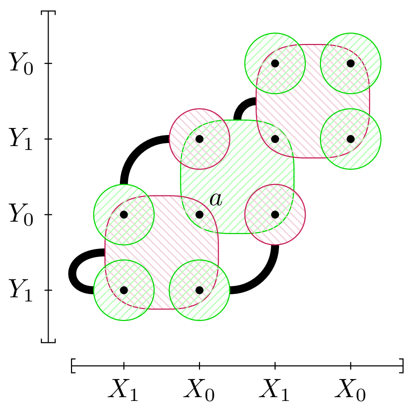
Lemma 4.3.6. *Let $\mathfrak{G} = (A, H, V, W)$ be a tri-binary structure. If $\mathfrak{G}$ satisfies $\varphi'_{complete}$ and $\varphi_{progress}$, then (A, H, V) is grid-like.*

Proof: It is sufficient to notice that formula $\varphi'_{complete}$ implies $\varphi_{complete}$ and then we apply Lemma 4.3.1. □

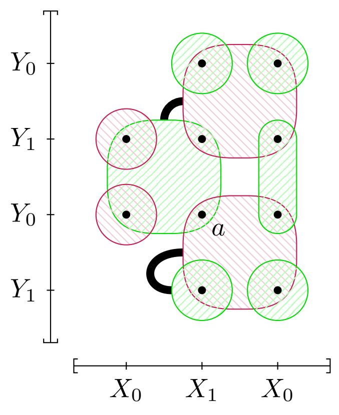
As in the previous subsection, we will consider data structures in $\mathrm{Str}(\Sigma_{grid}; 2\mathbb{DMS})$ to encode domino systems and we will use 2-$\mathsf{LF}_2^{\text{int}}[\Sigma_{grid}; \mathcal{I}_2]$ formulae in order to ensure that the data structures are grid-like and that an embedding of a domino system in it is feasible. In the previous section, to ensure that a data structure is a grid, we used completely the fact that we could look in our logical formulae to neighborhood of radius 3 (cf formula $\varphi_{grid}^{3\text{-}loc}$), but since here we want to look at neighborhoods of radius 2, we use the diagonal relation and rely on the result of the previous lemma. Consequently, we will need again the two quantifier free formulae $\varphi_H(x, y)$ and $\varphi_V(x, y)$ of $\mathsf{FO}_{2\mathbb{DMS}}[\Sigma_{grid}; \mathcal{S}_2]$ introduced in 4.3 and we define a new quantifier free formula $\varphi_W(x, y)$ in $\mathsf{FO}_{2\mathbb{DMS}}[\Sigma_{grid}; (1, 2)]$:

$$\begin{aligned}
\varphi_W^{00} &= X_0(x) \wedge X_1(y) \wedge Y_0(x) \wedge Y_1(y) \wedge x \;{}_1{\sim}_2\; y\\
\varphi_W^{10} &= X_1(x) \wedge X_0(y) \wedge Y_0(x) \wedge Y_1(y) \wedge x \;{}_1{\sim}_2\; y\\
\varphi_W^{01} &= X_0(x) \wedge X_1(y) \wedge Y_1(x) \wedge Y_0(y) \wedge x \;{}_1{\sim}_2\; y\\
\varphi_W^{11} &= X_1(x) \wedge X_0(y) \wedge Y_1(x) \wedge Y_0(y) \wedge x \;{}_1{\sim}_2\; y\\
\varphi_W &= \varphi_W^{00} \vee \varphi_W^{10} \vee \varphi_W^{01} \vee \varphi_W^{11}
\end{aligned}$$

We will now define a formula $\varphi_{grid}^{2\text{-}loc}$ in 2-$\mathsf{LF}_2^{\text{int}}[\Sigma_{grid}; {}_1{\sim}_1, {}_2{\sim}_2, {}_1{\sim}_2]$ which ensures that a



(a) If $X_0(a)$ and $Y_0(a)$ hold.



(b) If $X_1(a)$ and $Y_0(a)$ hold.

Figure 4.6: Some 2-local views of $\mathfrak{A}_{2m}$ for $\mathcal{R} = \{{}_1{\sim}_1, {}_2{\sim}_2, {}_1{\sim}_2\}$.

data structure corresponds to a grid. This formula is given by ($\oplus$ stands for exclusive or):

$$\begin{aligned}
\varphi^{\textit{2-loc}}_{\textit{complete}} &= \forall x.\langle\!\langle \forall y y'.\varphi_H(x,y) \wedge \varphi_V(y,y') \Rightarrow \varphi_W(x,y')\rangle\!\rangle^{2,\mathrm{int}}_x \\
&\quad \wedge \forall x.\langle\!\langle \forall y x' y'.\varphi_V(x,x') \wedge \varphi_W(x,y') \Rightarrow \varphi_H(x',y')\rangle\!\rangle^{2,\mathrm{int}}_x \\
\varphi^{\textit{2-loc}}_{\textit{progress}} &= \forall x.\langle\!\langle \exists y.\varphi_H(x,y) \wedge \exists y.\varphi_V(x,y)\rangle\!\rangle^{2,\mathrm{int}}_x \\
\varphi^{\textit{2-loc}}_{\textit{grid}} &= \varphi^{\textit{2-loc}}_{\textit{complete}} \wedge \varphi^{\textit{2-loc}}_{\textit{progress}} \wedge \forall x.\langle\!\langle (X_0(x) \oplus X_1(x)) \wedge (Y_0(x) \oplus Y_1(x))\rangle\!\rangle^{2,\mathrm{int}}_x
\end{aligned}$$

Lemma 4.3.7. *The following statements hold:*

1. $\mathfrak{A}_{2m} \models \varphi^{\textit{2-loc}}_{\textit{grid}}$*, and*

2. *for all* $\mathfrak{A} = (A, (P_\sigma), f_1, f_2) \in \mathrm{Str}(\Sigma_{grid}; 2\mathbb{DMS})$*, if* $\mathfrak{A} \models \varphi^{\textit{2-loc}}_{\textit{grid}}$*, then* $(A, [\![\varphi_H]\!]_{\mathfrak{A}}, [\![\varphi_V]\!]_{\mathfrak{A}})$ *is grid-like.*

Sketch of the proof: The proof is similar to the of Lemma 4.3.3. For the first point, Figure 4.6 provides some representation of $\mathfrak{A}_{2m}|^{2,\mathrm{int}}_{a,\mathcal{I}_2}$ for some elements $a \in G_{2m}$. For the second point, following the same reasonning as in Lemma 4.3.3, we first show that the tri-binary structure $(A, [\![\varphi_H]\!]_{\mathfrak{A}}, [\![\varphi_V]\!]_{\mathfrak{A}}, [\![\varphi_W]\!]_{\mathfrak{A}})$ satisfies $\varphi'_{complete}$ and $\varphi_{progress}$ and we use Lemma 4.3.6 to conclude. $\square$

As previously, we provide a formula $\varphi'_{\mathcal{D}}$ of 2-$\mathsf{LF}^{\mathrm{int}}_2[D; {}_1{\sim}_1, {}_2{\sim}_2, {}_1{\sim}_2]$ for any domino system $\mathcal{D} = (D, H_{\mathcal{D}}, V_{\mathcal{D}})$. This formalism is morally the same as the formula $\varphi_{\mathcal{D}}$, we only restrict the neighborhood, but in the fact this does not change anything:

$$\begin{aligned}
\varphi'_{\mathcal{D}} \quad := \quad & \forall x.\langle\!\langle \textstyle\bigvee_{d\in D} d(x) \wedge \bigwedge_{d\neq d'\in D} \neg(d(x) \wedge d'(x))\rangle\!\rangle^{2,\mathrm{int}}_x \\
& \wedge \forall x.\langle\!\langle \forall y.\varphi_H(x,y) \Rightarrow \textstyle\bigvee_{(d,d')\in H_{\mathcal{D}}} d(x) \wedge d'(y)\rangle\!\rangle^{2,\mathrm{int}}_x \\
& \wedge \forall x.\langle\!\langle \forall y.\varphi_V(x,y) \Rightarrow \textstyle\bigvee_{(d,d')\in V_{\mathcal{D}}} d(x) \wedge d'(y)\rangle\!\rangle^{2,\mathrm{int}}_x
\end{aligned}$$

We have the following proposition whose proof follows the same line as Proposition 4.3.4.

Proposition 4.3.8. *Given $\mathcal{D} = (D, H_\mathcal{D}, V_\mathcal{D})$ a domino system, we have that $\mathcal{D}$ admits a periodic tiling iff the* $2\text{-}\mathsf{LF}_2^{\mathrm{int}}[\Sigma_{grid} \uplus D; \{{}_1{\sim}_1, {}_2{\sim}_2, {}_1{\sim}_2\}]$ *formula* $\varphi_{grid}^{\textit{2-loc}} \wedge \varphi'_\mathcal{D}$ *is satisfiable.*

Finally, we obtain the desired undecidability result.

Theorem 4.3.9

$2\mathbb{DMS}$-$\textsc{Sat}(2\text{-}\mathsf{LF}_2^{\mathrm{int}}; \{{}_1{\sim}_1, {}_2{\sim}_2, {}_1{\sim}_2\})$ *is undecidable.*

Chapter 5

Deciding the satisfiability of the existential fragments

This chapter presents the work of [11]. It is dedicated to set the nature of the satisfiability problems of the existential fragment.

In this chapter, we will always have $\mathcal{R} = \mathcal{A}_\kappa = \{{}_i\!\sim_j \mid 1 \leq i,j \leq \kappa\}$, so we might sometimes omit to precise it. We will focus only on the fragment with exterior, that is we will always have $\wp = \mathrm{ext}$. With those choices for the parameters $\mathcal{R}$ and $\wp$, we will have the strongest decidability results.

From Chapter 3, we recall that the existential fragment is denoted $\exists\text{-}r\text{-}\mathsf{LF}_\kappa^{\mathrm{ext}}[\Sigma;\mathcal{A}_\kappa]$ and is given by the grammar:

$$\varphi \ ::= \ \langle\!\langle \psi \rangle\!\rangle_x^{r,\mathrm{ext}} \mid x = y \mid \neg(x = y) \mid \exists x.\varphi \mid \varphi \vee \varphi \mid \varphi \wedge \varphi$$

where ψ is a formula from $\mathsf{FO}_{\kappa\mathbb{DMS}}[\Sigma;\mathcal{A}_\kappa]$ with (at most) one free variable x. We also recall that the quantifier free fragment qf-r-$\mathsf{LF}_\kappa^{\mathrm{ext}}[\Sigma;\mathcal{A}_\kappa]$ is defined by the grammar

$$\varphi \ ::= \ \langle\!\langle \psi \rangle\!\rangle_x^{r,\mathrm{ext}} \mid x = y \mid \neg(x = y) \mid \varphi \vee \varphi \mid \varphi \wedge \varphi.$$

Note that for both these fragments, we do not impose any restrictions on the use of quantifiers in the formula ψ located inside the local modality $\langle\!\langle \psi \rangle\!\rangle_x^{r,\mathrm{ext}}$.

In the first section, which is dedicated to positive results, we show the decidability of $2\mathbb{DMS}$-$\textsc{Sat}(\exists\text{-}2\text{-}\mathsf{LF}^{\mathrm{ext}};\mathcal{A}_2)$ and for all $\kappa \geq 0$ the decidability of $\kappa\mathbb{DMS}$-$\textsc{Sat}(\exists\text{-}1\text{-}\mathsf{LF}^{\mathrm{ext}};\mathcal{A}_\kappa)$. In the second section, we show that as soon as we increase the parameters r or κ, the problems become undecidable, that is we show the undecidability of $2\mathbb{DMS}$-$\textsc{Sat}(\exists\text{-}3\text{-}\mathsf{LF}^{\mathrm{ext}};\mathcal{A}_2)$ and $3\mathbb{DMS}$-$\textsc{Sat}(\exists\text{-}2\text{-}\mathsf{LF}^{\mathrm{ext}};\mathcal{A}_3)$.

From [11] there are two modifications; the first one is a slight change in the notation in order to fit the ones of this manuscript, and the second modification is that definitions and preliminary results have moved to Chapter 2.

5.1 Decidability results

We show here decidability of $2\mathbb{DMS}$-$\textsc{Sat}(\exists\text{-}2\text{-}\mathsf{LF}^{\mathrm{ext}}; \mathcal{A}_2)$ and then, for all $\kappa \geq 0$, the decidability of $\kappa\mathbb{DMS}$-$\textsc{Sat}(\exists\text{-}1\text{-}\mathsf{LF}^{\mathrm{ext}}; \mathcal{A}_\kappa)$.

5.1.1 Two data values and balls of radius 2

In this section, we prove that the satisfiability problem for the existential fragment of local first-order logic with two data values and balls of radius two is decidable. To obtain this result we provide a reduction to the satisfiability problem for first-order logic over 1-data-multisets. Our reduction is based on the following intuition. Consider a 2-data-multiset $\mathfrak{A} = (A, (P_\sigma), f_1, f_2) \in \mathrm{Str}(\Sigma; 2\mathbb{DMS})$ and an element $a \in A$. If we take an element b in $B_2^{\mathfrak{A}}(a)$, the radius-2-ball around a, we know that either $f_1(b)$ or $f_2(b)$ is a common value with a. In fact, if b is at distance 1 of a, this holds by definition and if b is at distance 2 then b shares an element with c at distance 1 of a and this element has to be shared with a as well so b ends to be at distance 1 of a. The trick consists then in using extra-labels for elements sharing a value with a that can be forgotten and to keep only the value of b not present in a, this construction leading to a 1-data-multiset. It remains to show that we can ensure that a 1-data-multiset is the fruit of this construction in a formula of $\mathsf{FO}_{1\mathbb{DMS}}[\Sigma'; \{\sim\}]$ (where Σ' is obtained from Σ by adding extra predicates).

The first step for our reduction consists in providing a characterisation for the elements located in the radius-1-ball and the radius-2-ball around another element.

Lemma 5.1.1. *Let $\mathfrak{A} = (A, (P_\sigma), f_1, f_2) \in \mathrm{Str}(\Sigma; 2\mathbb{DMS})$ and $a, b \in A$ and $j \in \{1, 2\}$. We have:*

1. *$(b, j) \in B_1^{\mathfrak{A}}(a)$ iff there is $i \in \{1, 2\}$ such that $a\,{}_i{\sim}_j^{\mathfrak{A}}\, b$.*
2. *$(b, j) \in B_2^{\mathfrak{A}}(a)$ iff there exists $i, k \in \{1, 2\}$ such that $a\,{}_i{\sim}_k^{\mathfrak{A}}\, b$.*

Lemma 5.1.1 is a part of Corollary 3.2.28. But it is interesting to prove it more directly too.

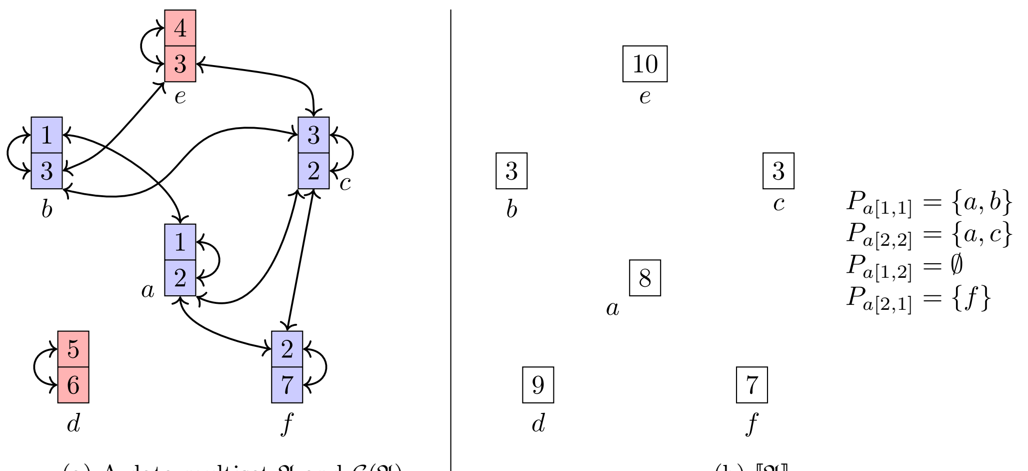
Proof: We show both statements:

1. Since $(b, j) \in B_1^{\mathfrak{A}}(a)$, by definition we have either $b = a$ and in that case $a\,{}_j{\sim}_j^{\mathfrak{A}}\, b$ holds, or $b \neq a$ and necessarily there exists $i \in \{1, 2\}$ such that $a\,{}_i{\sim}_j^{\mathfrak{A}}\, b$.
2. First, if there exists $i, k \in \{1, 2\}$ such that $a\,{}_i{\sim}_k^{\mathfrak{A}}\, b$, then $(b, k) \in B_1^{\mathfrak{A}}(a)$ and $(b, j) \in B_2^{\mathfrak{A}}(a)$ by definition. Assume now that $(b, j) \in B_2^{\mathfrak{A}}(a)$. Hence there exists $i \in \{1, 2\}$ such that $d^{\mathfrak{A}}((a, i), (b, j)) \leq 2$. We perform a case analysis on the value of $d^{\mathfrak{A}}((a, i), (b, j))$.
   - **Case** $d^{\mathfrak{A}}((a, i), (b, j)) = 0$. In that case $a = b$ and $i = j$ and we have $a\,{}_i{\sim}_i^{\mathfrak{A}}\, b$.
   - **Case** $d^{\mathfrak{A}}((a, i), (b, j)) = 1$. In that case, $((a, i), (b, j))$ is an edge in the data graph $\mathcal{G}(\mathfrak{A})$ of $\mathfrak{A}$ which means that $a\,{}_i{\sim}_j^{\mathfrak{A}}\, b$ holds.

- **Case** $d^{\mathfrak{A}}((a,i),(b,j)) = 2$. Note that we have by definition $a \neq b$. Furthermore, in that case, there is $(c,k) \in A \times \{1,2\}$ such that $((a,i),(c,k))$ and $((c,k),(b,j))$ are edges in $\mathcal{G}(\mathfrak{A})$. If $c \neq a$ and $c \neq b$, this implies that $a\,{}_i{\sim}_k^{\mathfrak{A}}\, c$ and $c\,{}_k{\sim}_j^{\mathfrak{A}}\, b$, so $a\,{}_i{\sim}_j^{\mathfrak{A}}\, b$ and $d^{\mathfrak{A}}((a,i),(b,j)) = 1$ which is a contradiction. If $c = a$ and $c \neq b$, this implies that $a\,{}_k{\sim}_j^{\mathfrak{A}}\, b$. If $c \neq a$ and $c = b$, this implies that $a\,{}_i{\sim}_k^{\mathfrak{A}}\, b$. □

We consider a formula $\varphi = \exists x_1 \ldots \exists x_n . \varphi_{qf}(x_1, \ldots, x_n)$ of $\exists$-2-$\mathsf{LF}_2^{\text{ext}}[\Sigma; \mathcal{A}_2]$ in prenex normal form, i.e., such that $\varphi_{qf}(x_1, \ldots, x_n) \in$ qf-2-$\mathsf{LF}_2^{\text{ext}}[\Sigma; \mathcal{A}_2]$. We know that there is a structure $\mathfrak{A} = (A, (P_\sigma)_{\sigma \in \Sigma}, f_1, f_2)$ in $\text{Str}(\Sigma; 2\mathbb{DMS})$ such that $\mathfrak{A} \models \varphi$ iff there are $a_1, \ldots, a_n \in A$ such that $\mathfrak{A} \models \varphi_{qf}(a_1, \ldots, a_n)$.

Let $\mathfrak{A} = (A, (P_\sigma)_{\sigma \in \Sigma}, f_1, f_2)$ be a structure in $\text{Str}(\Sigma; 2\mathbb{DMS})$ and $\vec{a} = (a_1, \ldots, a_n)$ a tuple of elements in A^n. We shall present the construction of a 1-data-multiset $[\![\mathfrak{A}]\!]_{\vec{a}}$ in $\text{Str}(\Sigma'; 1\mathbb{DMS})$ (with $\Sigma \subseteq \Sigma'$) with the same set of nodes as $\mathfrak{A}$, but where each node carries a single data value. In order to retrieve the data relations that hold in $\mathfrak{A}$ while reasoning over $[\![\mathfrak{A}]\!]_{\vec{a}}$, we introduce extra-predicates in Σ' to establish whether a node shares a common value with one of the nodes among $a_1, \ldots, a_n$ in $\mathfrak{A}$.



(a) A data-multiset $\mathfrak{A}$ and $\mathcal{G}(\mathfrak{A})$. (b) $[\![\mathfrak{A}]\!]_{(a)}$.

Figure 5.1

We now explain formally how we build $[\![\mathfrak{A}]\!]_{\vec{a}}$. Let $\Gamma_n = \{a_p[i,j] \mid p \in \{1, \ldots, n\}, i, j \in \{1,2\}\}$ be a set of new unary predicates and $\Sigma' = \Sigma \cup \Gamma_n$. For every element $b \in A$, the predicates in Γ_n are used to keep track of the relation between the data values of b and the one of $a_1, \ldots, a_n$ in $\mathfrak{A}$. Formally, we define $P_{a_p[i,j]} = \{b \in A \mid \mathfrak{A} \models a_p \;{}_i{\sim}_j\; b\}$. We now define a data function $f : A \to \mathbb{N}$. We recall for this matter that $\mathit{Val}^{\mathfrak{A}}(\vec{a}) = \{f_1(a_1), f_2(a_1), \ldots, f_1(a_n), f_2(a_n)\}$ and let $f_{\text{new}} : A \to \mathbb{N} \setminus \mathit{Val}^{\mathfrak{A}}(A)$ be an injection. For every $b \in A$, we set:

$$f(b) = \begin{cases} f_2(b) \text{ if } f_1(b) \in \mathit{Val}^{\mathfrak{A}}(\vec{a}) \text{ and } f_2(b) \notin \mathit{Val}^{\mathfrak{A}}(\vec{a}) \\ f_1(b) \text{ if } f_1(b) \notin \mathit{Val}^{\mathfrak{A}}(\vec{a}) \text{ and } f_2(b) \in \mathit{Val}^{\mathfrak{A}}(\vec{a}) \\ f_{\text{new}}(b) \text{ otherwise} \end{cases}$$

Hence depending if $f_1(b)$ or $f_2(b)$ is in $\mathit{Val}^{\mathfrak{A}}(\vec{a})$, it splits the elements of $\mathfrak{A}$ in four categories. If $f_1(b)$ and $f_2(b)$ are in $\mathit{Val}^{\mathfrak{A}}(\vec{a})$, the predicates in Γ_n allow us to retrieve all the data values of b. Given $j \in \{1,2\}$, if $f_j(b)$ is in $\mathit{Val}^{\mathfrak{A}}(\vec{a})$ but $f_{3-j}(b)$ is not, the new predicates will give us the j-th data value of b and we have to keep track of the $(3-j)$-th one, so we save it in $f(b)$. Lastly, if neither $f_1(b)$ nor $f_2(b)$ is in $\mathit{Val}^{\mathfrak{A}}(\vec{a})$, we will never be able to see the data values of b in φ_{q_f} (thanks to Lemma 5.1.1), so they do not matter to us. Finally, we have $[\![\mathfrak{A}]\!]_{\vec{a}} = (A, (P_\sigma)_{\sigma \in \Sigma'}, f)$. Figure 5.1b provides an example of $[\![\mathfrak{A}]\!]_{\vec{a}}$ for the data-multisets depicted on Figure 5.1a and $\vec{a} = (a)$.

The next lemma formalizes the connection existing between $\mathfrak{A}$ and $[\![\mathfrak{A}]\!]_{\vec{a}}$.

Lemma 5.1.2. *Let $b, c \in A$ and $j, k \in \{1,2\}$ and $p \in \{1, \dots, n\}$. The following statements then hold.*

1. *If $(b,j) \in B_1^{\mathfrak{A}}(a_p)$ and $(c,k) \in B_1^{\mathfrak{A}}(a_p)$ then $\mathfrak{A}|_{a_p}^{2,\text{ext}} \models b \;{}_j{\sim}_k\; c$ iff there is $i \in \{1,2\}$ s.t. $b \in P_{a_p[i,j]}$ and $c \in P_{a_p[i,k]}$.*

2. *If $(b,j) \in B_2^{\mathfrak{A}}(a_p) \setminus B_1^{\mathfrak{A}}(a_p)$ and $(c,k) \in B_1^{\mathfrak{A}}(a_p)$ then $\mathfrak{A}|_{a_p}^{2,\text{ext}} \not\models b \;{}_j{\sim}_k\; c$*

3. *If $(b,j), (c,k) \in B_2^{\mathfrak{A}}(a_p) \setminus B_1^{\mathfrak{A}}(a_p)$ then $\mathfrak{A}|_{a_p}^{2,\text{ext}} \models b \;{}_j{\sim}_k\; c$ iff either $b \;{}_1{\sim}_1^{[\![\mathfrak{A}]\!]_{\vec{a}}}\; c$ or there exists $p' \in \{1, \dots, n\}$ and $\ell \in \{1,2\}$ such that $b \in P_{a_{p'}[\ell,j]}$ and $c \in P_{a_{p'}[\ell,k]}$.*

4. *If $(b,j) \notin B_2^{\mathfrak{A}}(a_p)$ and $(c,k) \in B_2^{\mathfrak{A}}(a_p)$ then $\mathfrak{A}|_{a_p}^{2,\text{ext}} \not\models b \;{}_j{\sim}_k\; c$*

5. *If $(b,j) \notin B_2^{\mathfrak{A}}(a_p)$ and $(c,k) \notin B_2^{\mathfrak{A}}(a_p)$ then $\mathfrak{A}|_{a_p}^{2,\text{ext}} \models b \;{}_j{\sim}_k\; c$ iff $b = c$ and $j = k$.*

Proof: We suppose that $\mathfrak{A}|_{a_p}^{2,\text{ext}} = (A, (P_\sigma)_\sigma, f_1^p, f_2^p)$.

1. Assume that $(b,j) \in B_1^{\mathfrak{A}}(a_p)$ and $(c,k) \in B_1^{\mathfrak{A}}(a_p)$. It implies that $f_j^p(b) = f_j(b)$ and $f_k^p(c) = f_k(c)$. Then assume that $\mathfrak{A}|_{a_p}^{2,\text{ext}} \models b \;{}_j{\sim}_k\; c$. As $(b,j) \in B_1^{\mathfrak{A}}(a_p)$, thanks to Lemma 5.1.1.1 it means that there is a $i \in \{1,2\}$ such that $a_p \;{}_i{\sim}_j^{\mathfrak{A}}\; b$. So we have $f_k(c) = f_k^p(c) = f_j^p(b) = f_j(b) = f_i(a_p)$, that is $a_p \;{}_i{\sim}_k^{\mathfrak{A}}\; c$. Hence by definition, $b \in P_{a_p[i,j]}$ and $c \in P_{a_p[i,k]}$. Conversely, let $i \in \{1,2\}$ such that $b \in P_{a_p[i,j]}$ and $c \in P_{a_p[i,k]}$. This means that $a_p \;{}_i{\sim}_j^{\mathfrak{A}}\; b$ and $a_p \;{}_i{\sim}_k^{\mathfrak{A}}\; c$. So $f_j^p(b) = f_j(b) = f_i(a_p) = f_k(c) = f_k^p(c)$, that is $\mathfrak{A}|_{a_p}^{2,\text{ext}} \models b \;{}_j{\sim}_k\; c$.

2. Assume that $(b,j) \in B_2^{\mathfrak{A}}(a_p) \setminus B_1^{\mathfrak{A}}(a_p)$ and $(c,k) \in B_1^{\mathfrak{A}}(a_p)$. It implies that $f_j^p(b) = f_j(b)$ and $f_k^p(c) = f_k(c)$. Thanks to Lemma 5.1.1.1, $(c,k) \in B_1^{\mathfrak{A}}(a_p)$ implies that $f_k(c) \in \{f_1(a_p), f_2(a_p)\}$ and $(b,j) \notin B_1^{\mathfrak{A}}(a_p)$ implies that $f_j(b) \notin \{f_1(a_p), f_2(a_p)\}$. So $\mathfrak{A}|_{a_p}^{2,\text{ext}} \not\models b \;{}_j{\sim}_k\; c$.

3. Assume that $(b,j),(c,k) \in B_2^{\mathfrak{A}}(a_p) \setminus B_1^{\mathfrak{A}}(a_p)$. As previously, we have that $f_j(b) \notin \{f_1(a_p), f_2(a_p)\}$ and $f_k(c) \notin \{f_1(a_p), f_2(a_p)\}$, and thanks to Lemma 5.1.1.2, we have $f_{3-j}(b) \in \{f_1(a_p), f_2(a_p)\}$ and $f_{3-k}(b) \in \{f_1(a_p), f_2(a_p)\}$. There is then two cases:
   - Suppose there does not exists $p' \in \{1,\dots,n\}$ such that $f_j(b) \in \{f_1(a_{p'}), f_2(a_{p'})\}$. This allows us to deduce that $f_j^p(b) = f_j(b) = f(b)$ and $f_k^p(c) = f_k(c)$. If $\mathfrak{A}|_{a_p}^{2,\text{ext}} \models b\ {}_j{\sim}_k\ c$, then necessarily there does not exists $p' \in \{1,\dots,n\}$ such that $f_k(c) \in \{f_1(a_{p'}), f_2(a_{p'})\}$ so we have $f_k^p(c) = f_k(c) = f(c)$ and $f(b) = f(c)$, consequently $b\ {}_1{\sim}_1^{[\![\mathfrak{A}]\!]_{\vec{a}}}\ c$. Similarly assume that $b\ {}_1{\sim}_1^{[\![\mathfrak{A}]\!]_{\vec{a}}}\ c$, this means that $f(b) = f(c)$ and either $b = c$ and $k = j$ or $b \neq c$ and by injectivity of f,we have $f_j(b) = f(b) = f_k(c)$. This allows us to deduce that $\mathfrak{A}|_{a_p}^{2,\text{ext}} \models b\ {}_j{\sim}_k\ c$.
   - If there exists $p' \in \{1,\dots,n\}$ such that $f_j(b) = f_\ell(a_{p'})$ for some $\ell \in \{1,2\}$. Then we have $b \in P_{a_{p'}[\ell,j]}$. Consequently, we have $\mathfrak{A}|_{a_p}^{2,\text{ext}} \models b\ {}_j{\sim}_k\ c$ iff $c \in P_{a_{p'}[\ell,k]}$.

4. We prove the case 4 and 5 at the same time. Assume that $(b,j) \notin B_2^{\mathfrak{A}}(a_p)$. It means that in order to have $f_j^p(b) = f_k^p(c)$, we must have $(b,j) = (c,k)$. So if $(c,k) \in B_2^{\mathfrak{A}}(a_p)$, we can not have $\mathfrak{A}|_{a_p}^{2,\text{ext}} \models b\ {}_j{\sim}_k\ c$ which ends case 4. And if $(c,k) \notin B_2^{\mathfrak{A}}(a_p)$, we have that $\mathfrak{A}|_{a_p}^{2,\text{ext}} \models b\ {}_j{\sim}_k\ c$ iff $b = c$ and $j = k$. □

We see now how to translate the formula $\varphi_{qf}(x_1,\dots,x_n)$ into a formula $[\![\varphi_{qf}]\!](x_1,\dots,x_n)$ in $\mathsf{FO}_{1\mathbb{DMS}}[\Sigma';\{\sim\}]$ such that $\mathfrak{A}$ satisfies $\varphi_{qf}(a_1,\dots,a_n)$ iff, $[\![\mathfrak{A}]\!]_{\vec{a}}$ satisfies $[\![\varphi_{qf}]\!](a_1,\dots,a_n)$. Thanks to the previous lemma we know that if $\mathfrak{A}|_{a_p}^{2,\text{ext}} \models b\ {}_j{\sim}_k\ c$ then (b,j) and (c,k) must belong to the same set among $B_1^{\mathfrak{A}}(a_p)$, $B_2^{\mathfrak{A}}(a_p) \setminus B_1^{\mathfrak{A}}(a_p)$ and $A \setminus B_2^{\mathfrak{A}}(a_p)$ and we can test in $[\![\mathfrak{A}]\!]_{\vec{a}}$ whether (b,j) is a member of $B_1^{\mathfrak{A}}(a_p)$ or $B_2^{\mathfrak{A}}(a_p)$. Indeed, thanks to Lemmas 5.1.1.1 and 5.1.1.2, we have $(b,j) \in B_1^{\mathfrak{A}}(a_p)$ iff $b \in \bigcup_{i=1,2} P_{a_p[i,j]}$ and $(b,j) \in B_2^{\mathfrak{A}}(a_p)$ iff $b \in \bigcup_{i=1,2}^{j'=1,2} P_{a_p[i,j']}$. This reasoning leads to the following formulas in $\mathsf{FO}_{1\mathbb{DMS}}[\Sigma';\{\sim\}]$ with $p \in \{1,\dots,n\}$ and $j \in \{1,2\}$:

- $\varphi_{j,B_1(a_p)}(y) := a_p[1,j](y) \ \vee\ a_p[2,j](y)$ to test if the j-th field of an element belongs to $B_1^{\mathfrak{A}}(a_p)$
- $\varphi_{B_2(a_p)}(y) := \varphi_{1,B_1(a_p)}(y) \ \vee\ \varphi_{2,B_1(a_p)}(y)$ to test if a field of an element belongs to $B_2^{\mathfrak{A}}(a_p)$
- $\varphi_{j,B_2(a_p)\setminus B_1(a_p)}(y) := \varphi_{B_2(a_p)}(y) \ \wedge\ \neg\varphi_{j,B_1(a_p)}(y)$ to test that the j-th field of an element belongs to $B_2^{\mathfrak{A}}(a_p) \setminus B_1^{\mathfrak{A}}(a_p)$

We shall now present how we use these formulas to translate atomic formulas of the form $y\ {}_j{\sim}_k\ z$ under some $\langle\!\langle - \rangle\!\rangle_{x_p}^{2,\text{ext}}$. For this matter, we rely on the three following formulas of $\mathsf{FO}_{1\mathbb{DMS}}[\Sigma';\{\sim\}]$:

- The first formula asks for (y,j) and (z,k) to be in $B_1^1(a_p)$ (where here we abuse notations, using variables for the elements they represent) and for these two data values to coincide with one data value of a_p, it corresponds to Lemma 5.1.2.1:

$$\varphi_{j,k,a_p}^{r=1}(y,z) := \varphi_{j,B_1(a_p)}(y) \ \wedge\ \varphi_{k,B_1(a_p)}(z) \ \wedge\ \bigvee_{i=1,2} \big(a_p[i,j](y) \ \wedge\ a_p[i,k](z)\big)$$

- The second formula asks for (y,j) and (z,k) to be in $B_2^{\mathfrak{A}}(a_p) \setminus B_1^{\mathfrak{A}}(a_p)$ and checks either whether the data values of y and z in $[\![\mathfrak{A}]\!]_{\vec{a}}$ are equal or whether there exist p' and ℓ such that y belongs to $P_{a_{p'}[\ell,j]}$ and z belongs to $P_{a_{p'}[\ell,k]}$, it corresponds to Lemma 5.1.2.3:

$$\begin{aligned}\varphi^{r=2}_{j,k,a_p}(y,z) := {} & \varphi_{j,B_2(a_p)\setminus B_1(a_p)}(y) \wedge \varphi_{k,B_2(a_p)\setminus B_1(a_p)}(z) \\ & \wedge \left(y \sim z \vee \left(\bigvee_{p'=1}^{n} \bigvee_{\ell=1}^{2} a_{p'}[\ell,j](y) \wedge a_{p'}[\ell,k](z) \right)\right)\end{aligned}$$

- The third formula asks for (y,j) and (z,k) to not belong to $B_2^{\mathfrak{A}}(a_p)$ and for $y=z$, it corresponds to Lemma 5.1.2.5:

$$\varphi^{r>2}_{j,k,a_p}(y,z) := \begin{cases} \neg\varphi_{B_2(a_p)}(y) \wedge \neg\varphi_{B_2(a_p)}(z) \wedge y = z & \text{if } j = k \\ \bot & \text{otherwise} \end{cases}$$

Finally, here is the inductive definition of the translation $[\![-]\!]$ which uses sub transformations $[\![-]\!]_{x_p}$ in order to remember the centre of the ball and leads to the construction of $[\![\varphi_{qf}]\!](x_1,\ldots,x_n)$:

$$\begin{array}{rcl} [\![\varphi \vee \varphi']\!] & = & [\![\varphi]\!] \vee [\![\varphi']\!] \\ [\![x_p = x'_p]\!] & = & x_p = x'_p \\ [\![\neg\varphi]\!] & = & \neg[\![\varphi]\!] \\ [\![\langle\!\langle\psi\rangle\!\rangle^{2,\mathrm{ext}}_{x_p}]\!] & = & [\![\psi]\!]_{x_p} \\ [\![y \;{}_j\!\sim_k z]\!]_{x_p} & = & \varphi^{r=1}_{j,k,a_p}(y,z) \vee \varphi^{r=2}_{j,k,a_p}(y,z) \vee \varphi^{r>2}_{j,k,a_p}(y,z) \\ [\![\sigma(x)]\!]_{x_p} & = & \sigma(x) \\ [\![x = y]\!]_{x_p} & = & x = y \\ [\![\varphi \vee \varphi']\!]_{x_p} & = & [\![\varphi]\!]_{x_p} \vee [\![\varphi']\!]_{x_p} \\ [\![\neg\varphi]\!]_{x_p} & = & \neg[\![\varphi]\!]_{x_p} \\ [\![\exists x.\varphi]\!]_{x_p} & = & \exists x.[\![\varphi]\!]_{x_p} \end{array}$$

Lemma 5.1.3. *We have $\mathfrak{A} \models \varphi_{qf}(\vec{a})$ iff $[\![\mathfrak{A}]\!]_{\vec{a}} \models [\![\varphi_{qf}]\!](\vec{a})$.*

Proof: Because of the inductive definition of $[\![\varphi]\!]$ and that only the atomic formulas $y \;{}_j\!\sim_k z$ change, we only have to prove that given $b,c \in A$, we have $\mathfrak{A}|^{2,\mathrm{ext}}_{a_p} \models b \;{}_j\!\sim_k c$ iff $[\![\mathfrak{A}]\!]_{\vec{a}} \models [\![y \;{}_j\!\sim_k z]\!]_{x_p}(b,c)$.

We first suppose that $\mathfrak{A}|^{2,\mathrm{ext}}_{a_p} \models b \;{}_j\!\sim_k c$. Using Lemma 5.1.2, it implies that (b,j) and (c,k) belong to same set between $B_1^{\mathfrak{A}}(a_p)$, $B_2^{\mathfrak{A}}(a_p) \setminus B_1^{\mathfrak{A}}(a_p)$ and $A \setminus B_2^{\mathfrak{A}}(a_p)$. We proceed by a case analysis.

- If $(b,j),(c,k) \in B_1^{\mathfrak{A}}(a_p)$ then by lemma 5.1.2.1 we have that $[\![\mathfrak{A}]\!]_{\vec{a}} \models \varphi^{r=1}_{j,k,a_p}(b,c)$ and thus $[\![\mathfrak{A}]\!]_{\vec{a}} \models [\![y \;{}_j\!\sim_k z]\!]_{x_p}(b,c)$.
- If $(b,j),(c,k) \in B_2^{\mathfrak{A}}(a_p) \setminus B_1^{\mathfrak{A}}(a_p)$ then by lemma 5.1.2.3 we have that $[\![\mathfrak{A}]\!]_{\vec{a}} \models \varphi^{r=2}_{j,k,a_p}(b,c)$ and thus $[\![\mathfrak{A}]\!]_{\vec{a}} \models [\![y \;{}_j\!\sim_k z]\!]_{x_p}(b,c)$.

- If $(b,j),(c,k) \in A \setminus B_2^{\mathfrak{A}}(a_p)$ then by lemma 5.1.2.5 we have that $[\![\mathfrak{A}]\!]_{\vec{a}} \models \varphi^{r>2}_{j,k,a_p}(b,c)$ and thus $[\![\mathfrak{A}]\!]_{\vec{a}} \models [\![y \ {}_j{\sim}_k \ z]\!]_{x_p}(b,c)$.

We now suppose that $[\![\mathfrak{A}]\!]_{\vec{a}} \models [\![y \ {}_j{\sim}_k \ z]\!]_{x_p}(b,c)$. It means that $[\![\mathfrak{A}]\!]_{\vec{a}}$ satisfies at least $\varphi^{r=1}_{j,k,a_p}(b,c)$, $\varphi^{r=2}_{j,k,a_p}(b,c)$ or $\varphi^{r>2}_{j,k,a_p}(b,c)$. If $[\![\mathfrak{A}]\!]_{\vec{a}} \models \varphi^{r=1}_{j,k,a_p}(b,c)$, it implies that (b,j) and (c,k) are in $B_1^{\mathfrak{A}}(a_p)$, and we can then apply lemma 5.1.2.1 to deduce that $\mathfrak{A}|^{2,\text{ext}}_{a_p} \models b \ {}_j{\sim}_k \ c$. If $[\![\mathfrak{A}]\!]_{\vec{a}} \models \varphi^{r=2}_{j,k,a_p}(b,c)$, it implies that (b,j) and (c,k) are in $B_2^{\mathfrak{A}}(a_p) \setminus B_1^{\mathfrak{A}}(a_p)$, and we can then apply Lemma 5.1.2.3 to deduce that $\mathfrak{A}|^{2,\text{ext}}_{a_p} \models b \ {}_j{\sim}_k \ c$. If $[\![\mathfrak{A}]\!]_{\vec{a}} \models \varphi^{r>2}_{j,k,a_p}(b,c)$, it implies that (b,j) and (c,k) are in $A \setminus B_2^{\mathfrak{A}}(a_p)$, and we can then apply lemma 5.1.2.5 to deduce that $\mathfrak{A}|^{2,\text{ext}}_{a_p} \models b \ {}_j{\sim}_k \ c$. □

To provide a reduction from $2\mathbb{DMS}$-SAT($\exists$-2-LF$^{\text{ext}}$; $\mathcal{A}_2$) to $1\mathbb{DMS}$-SAT(FO; $\{\sim\}$), having the formula $[\![\varphi_{qf}]\!](x_1,\ldots,x_n)$ is not enough because to use the result of the previous Lemma, we need to ensure that there exists a model $\mathfrak{B}$ and a tuple of elements $(a_1,\ldots,a_n)$ such that $\mathfrak{B} \models [\![\varphi_{qf}]\!](a_1,\ldots,a_n)$ and as well that there exists $\mathfrak{A} \in \text{Str}(\Sigma; 2\mathbb{DMS})$ such that $\mathfrak{B} = [\![\mathfrak{A}]\!]_{\vec{a}}$. We explain now how we can ensure this last point.

Now, we want to characterize the structures of the form $[\![\mathfrak{A}]\!]_{\vec{a}}$. Given a structure $\mathfrak{B} = (A,(P_\sigma)_{\sigma\in\Sigma'},f) \in \text{Str}(\Sigma'; 1\mathbb{DMS})$ and $\vec{a} \in A$, we say that $(\mathfrak{B},\vec{a})$ is *well formed* iff there exists a structure $\mathfrak{A} \in \text{Str}(\Sigma; 2\mathbb{DMS})$ such that $\mathfrak{B} = [\![\mathfrak{A}]\!]_{\vec{a}}$. Hence $(\mathfrak{B},\vec{a})$ is *well formed* iff there exist two functions $f_1,f_2 : A \to \mathbb{N}$ such that $[\![\mathfrak{A}]\!]_{\vec{a}} = [\![(A,(P_\sigma)_{\sigma\in\Sigma},f_1,f_2)]\!]_{\vec{a}}$. We state three properties on $(\mathfrak{B},\vec{a})$, and we will show that they characterize being well formed.

1. (Transitivity) For all $b,c \in A$, $p,q \in \{1,\ldots,n\}$, $i,j,k,\ell \in \{1,2\}$ if $b \in P_{a_p[i,j]}$, $c \in P_{a_p[i,\ell]}$ and $b \in P_{a_q[k,j]}$ then $c \in P_{a_q[k,\ell]}$.
2. (Reflexivity) For all p and i, we have $a_p \in P_{a_p[i,i]}$,
3. (Uniqueness) For all $b \in A$, if $b \in \bigcap_{j=1,2}\bigcup^{i=1,2}_{p=1,\ldots,n} P_{a_p[i,j]}$ or $b \notin \bigcup_{j=1,2}\bigcup^{i=1,2}_{p=1,\ldots,n} P_{a_p[i,j]}$ then for any $c \in B$ such that $f(c) = f(b)$ we have $c = b$.

Each property can be expressed by a first order logic formula, which we respectively name φ_{tran}, φ_{refl} and φ_{uniq} and we denote by φ_{wf} their conjunction:

$$\begin{aligned}
\varphi_{tran} &= \forall y \forall z. \bigwedge_{p,q=1}^{n} \bigwedge_{i,j,k,\ell=1}^{2} \Big(a_p[i,j](y) \wedge a_p[i,\ell](z) \wedge a_q[k,j](y) \\
&\qquad\qquad \to a_q[k,\ell](z) \Big) \\
\varphi_{refl}(x_1,\ldots,x_n) &= \bigwedge_{p=1}^{n} \bigwedge_{i=1}^{2} a_p[i,i](x_p) \\
\varphi_{uniq} &= \forall y. \Big(\bigwedge_{j=1}^{2} \bigvee_{p=1}^{n} \bigvee_{i=1}^{2} a_p[i,j](y) \vee \bigwedge_{j=1}^{2} \bigwedge_{p=1}^{n} \bigwedge_{i=1}^{2} \neg a_p[i,j](y) \Big) \\
&\qquad \to (\forall z. y \sim z \to y = z) \\
\varphi_{wf}(x_1,\ldots,x_n) &= \varphi_{tran} \wedge \varphi_{refl}(x_1,\ldots,x_n) \wedge \varphi_{uniq}
\end{aligned}$$

The next lemma expresses that the formula φ_{wf} allows to characterise precisely the 1-data-multisets in $\mathrm{Str}(\Sigma'; 1\mathbb{DMS})$ which are well-formed.

Lemma 5.1.4. *Let $\mathfrak{B} \in \mathrm{Str}(\Sigma'; 1\mathbb{DMS})$ and $a_1, \ldots, a_n$ elements of $\mathfrak{B}$, then $(\mathfrak{B}, \vec{a})$ is well formed iff $\mathfrak{B} \models \varphi_{wf}(\vec{a})$.*

Proof: First, if $(\mathfrak{B}, \vec{a})$ is well formed, then there there exists $\mathfrak{A} \in \mathrm{Str}(\Sigma; 2\mathbb{DMS})$ such that $\mathfrak{B} = [\![\mathfrak{A}]\!]_{\vec{a}}$ and by construction we have $[\![\mathfrak{A}]\!]_{\vec{a}} \models \varphi_{wf}(\vec{a})$. We now suppose that $\mathfrak{B} = (A, (P_\sigma)_{\sigma\in\Sigma'}, f)$ and $\mathfrak{B} \models \varphi_{wf}(\vec{a})$. In order to define the functions $f_1, f_2 : A \to \mathbb{N}$, we need to introduce some objects.

We first define a function $g : \{1, \ldots, n\} \times \{1, 2\} \to \mathbb{N} \setminus Im(f)$ (where $Im(f)$ is the image of f in $\mathfrak{B}$) which verifies the following properly:

- for all $p, q \in \{1, \ldots, n\}$ and $i, j \in \{1, 2\}$, we have $a_q \in P_{a_p[i,j]}$ iff $g(p, i) = g(q, j)$.

We use this function to fix the two data values carried by the elements in $\{a_1, \ldots, a_m\}$.

We now explain why the function g is well defined, it is due to the fact that $\mathfrak{B} \models \varphi_{tran} \wedge \varphi_{refl}(a_1, \ldots, a_n)$. To prove it formally, let S be a binary relation on $\{1, \ldots, n\} \times \{1, 2\}$ defined by: $(p, i)S(q, j)$ if $a_q \in P_{a_p[i,j]}$. Then the function g is well defined iff S is an equivalence relation (i.e. if S is reflexive, symmetric and transitive). First S is reflexive iff for all (p, i), we have $(p, i)S(p, i)$. It is the case thanks to φ_{refl} which implies that $a_p \in P_{a_p[i,i]}$. Next we show that S is symmetric. So we assume that $(p, i)S(q, j)$ and we want to show that $(q, j)S(p, i)$. With $(p, i)S(q, j)$ we have that $a_q \in P_{a_p[i,j]}$ and thanks to φ_{refl} we have $a_p \in P_{a_p[i,i]}$ and $a_q \in P_{a_q[j,j]}$. So thanks to φ_{tran} we have $a_p \in P_{a_q[j,i]}$ as desired. Last we show that S is transitive, so we assume that $(p, i)S(p', i)$ and $(p', i')S(p'', i'')$ and we want to show that $(p, i)S(p'', i'')$. With $(p, i)S(p', i)$ and $(p', i')S(p'', i'')$ we have $a'_p \in P_{a_p[i,i']}$ and $a''_p \in P_{a_{p'}[i',i'']}$. Then thanks to φ_{refl} we have $a'_p \in P_{a_{p'}[i',i']}$ and then thanks to φ_{tran} we have $a''_p \in P_{a_p[i,i'']}$ as desired. We now are convinced that g is correctly defined.

We also need a natural d_{out} belonging to $\mathbb{N} \setminus (Im(g) \cup Im(f))$. For $j \in \{1, 2\}$, we define f_j as follows for all $b \in A$:

$$f_j(b) = \begin{cases} g(p, i) & \text{if for some } p, i \text{ we have } b \in P_{a_p[i,j]} \\ f(b) & \text{if for all } p, i \text{ we have } b \notin P_{a_p[i,j]} \text{ and for some } p, i \text{ we have } b \in P_{a_p[i,3-j]} \\ d_{out} & \text{if for all } p, i, j', \text{ we have } b \notin P_{a_p[i,j']} \end{cases}$$

Here again, we can show that since $\mathfrak{B} \models \varphi_{tran} \wedge \varphi_{refl}(a_1, \ldots, a_n)$, the functions f_1 and f_2 are well founded. Indeed, assume that $b \in P_{a_p[i,j]} \cap P_{a_q[k,j]}$, then we have necessarily that $g(p, i) = g(q, k)$. For this we need to show that $a_p \in a_q[k, i]$ and we use again the formula φ_{tran}. This can be obtained because we have $b \in P_{a_p[i,j]}$ and $a_p \in P_{a_p[i,i]}$ and $b \in P_{a_q[k,j]}$.

We then define $\mathfrak{A}$ as the 2-data-structures $(A, (P_\sigma)_{\sigma\in\Sigma}, f_1, f_2)$. It remains to prove that $\mathfrak{B} = [\![\mathfrak{A}]\!]_{\vec{a}}$.

First, note that for all $b \in A$, $p \in \{1, \ldots, n\}$ and $i, j \in \{1, 2\}$, we have $b \in P_{a_p[i,j]}$ iff $a_p \, {}_i{\sim}_j^{\mathfrak{A}} \, b$. Indeed, if we have $b \in P_{a_p[i,j]}$, we have that $f_j(b) = g(p, i)$ and since $a_p \in P_{a_p[i,i]}$ we have as well that $f_i(a_p) = g(p, i)$, as a consequence $a_p \, {}_i{\sim}_j^{\mathfrak{A}} \, b$. In the other direction, if

$a_p\,{}_i{\sim}_j^{\mathfrak{A}}\,b$, it means that $f_j(b) = f_i(a_p) = g(p,i)$ and thus $b \in P_{a_p[i,j]}$. Now to have $\mathfrak{B} = [\![\mathfrak{A}]\!]_{\vec{a}}$, one has only to be careful in the choice of function f_{new} while building $[\![\mathfrak{A}]\!]_{\vec{a}}$. We recall that this function is injective and is used to give a value to the elements $b \in A$ such that neither $f_1(b) \in Val^{\mathfrak{A}}(\vec{a})$ and $f_2(b) \notin Val^{\mathfrak{A}}(\vec{a})$ nor $f_1(b) \notin Val^{\mathfrak{A}}(\vec{a})$ and $f_2(b) \in Val^{\mathfrak{A}}(\vec{a})$. For these elements, we make f_{new} matches with the function f and the fact that we define an injection is guaranteed by the formula φ_{uniq}. ☐

Using the results of Lemma 5.1.3 and 5.1.4, we deduce that the starting formula $\varphi = \exists x_1 \ldots \exists x_n.\varphi_{qf}(x_1,\ldots,x_n)$ of $\exists$-2-$\mathsf{LF}_2^{\text{ext}}[\Sigma;\mathcal{A}_2]$ is satisfiable if and only if the final formula $\psi = \exists x_1 \ldots \exists x_n.[\![\varphi_{qf}]\!](x_1,\ldots,x_n)\wedge\varphi_{wf}(x_1,\ldots,x_n)$ is satisfiable. Note that ψ can be built in polynomial time from φ and that it belongs to $\mathsf{FO}_{1\mathbb{DMS}}[\Sigma';\{\sim\}]$. Hence, thanks to Theorem 2.3.25 which states that the problem $1\mathbb{DMS}$-$\textsc{Sat}(\mathsf{FO};\{\sim\})$ is in N2$\textsc{Exp}$, we then obtain that $2\mathbb{DMS}$-$\textsc{Sat}(\exists\text{-2-}\mathsf{LF}^{\text{ext}};\mathcal{A}_2)$ is in N2$\textsc{Exp}$.

We can as well obtain a matching lower bound thanks to a reduction from the problem $1\mathbb{DMS}$-$\textsc{Sat}(\mathsf{FO};\{\sim\})$. The bottleneck of this reduction is that in the formulas of $\exists$-2-$\mathsf{LF}_2^{\text{ext}}[\Sigma;\mathcal{A}_2]$, there is no restriction on the use of quantifiers for the formulas located under the scope of the $\langle\!\langle\cdot\rangle\!\rangle_x^{2,\text{ext}}$ modality and that we can extend a model $\mathsf{FO}_{1\mathbb{DMS}}[\Sigma;\{\sim\}]$ into a 2-data-multiset such that all elements and their values are located in the same radius-2-ball by adding everywhere a second data value equal to 0. More formally, let φ be a formula in $\mathsf{FO}_{1\mathbb{DMS}}[\Sigma;\{\sim\}]$ and consider the formula $\exists x.\langle\!\langle\varphi\rangle\!\rangle_x^{2,\text{ext}}$ where we interpret φ over 2-data structures (this formula simply never mentions the values located in the second fields). We have then the following lemma.

Lemma 5.1.5. *There exists $\mathfrak{A} \in \mathrm{Str}(\Sigma;1\mathbb{DMS})$ such that $\mathfrak{A} \models \varphi$ if and only if there exists $\mathfrak{B} \in \mathrm{Str}(\Sigma;2\mathbb{DMS})$ such that $\mathfrak{B} \models \exists x.\langle\!\langle\varphi\rangle\!\rangle_x^{2,\text{ext}}$.*

Proof: Assume that there exists $\mathfrak{A} = (A,(P_\sigma)_{\sigma\in\Sigma},f_1)$ in $\mathrm{Str}(\Sigma;1\mathbb{DMS})$ such that $\mathfrak{A} \models \varphi$. Consider the 2-data-multiset $\mathfrak{B} = (A,(P_\sigma)_{\sigma\in\Sigma},f_1,f_2)$ such that $f_2(a) = 0$ for all $a \in A$. Let $a \in A$. It is clear that we have $\mathfrak{B}|_a^{2,\text{ext}} = \mathfrak{B}$ and that $\mathfrak{B}|_a^{2,\text{ext}} \models \varphi$, because $\mathfrak{A} \models \varphi$ and φ never mentions the second values of the elements since it is a formula in $\mathsf{FO}_{1\mathbb{DMS}}[\Sigma;\{\sim\}]$. Consequently $\mathfrak{B} \models \exists x.\langle\!\langle\varphi\rangle\!\rangle_x^{2,\text{ext}}$.

Assume now that there exists $\mathfrak{B} = (A,(P_\sigma)_{\sigma\in\Sigma},f_1,f_2)$ in $\mathrm{Str}(\Sigma;2\mathbb{DMS})$ such that $\mathfrak{B} \models \exists x.\langle\!\langle\varphi\rangle\!\rangle_x^{2,\text{ext}}$. Hence there exists $a \in A$ such that $\mathfrak{B}|_a^{2,\text{ext}} \models \varphi$, but then by forgetting the second value in $\mathfrak{B}|_a^{2,\text{ext}}$ we obtain a model in $\mathrm{Str}(\Sigma;1\mathbb{DMS})$ which satisfies φ. ☐

Since $1\mathbb{DMS}$-$\textsc{Sat}(\mathsf{FO};\{\sim\})$ is N2$\textsc{Exp}$-hard (see Theorem 2.3.25), we obtain the desired lower bound.

Theorem 5.1.6

The problem $2\mathbb{DMS}$-$\textsc{Sat}(\exists\text{-2-}\mathsf{LF}^{\text{ext}};\mathcal{A}_2)$ is N2$\textsc{Exp}$-complete.

5.1.2 Balls of radius 1 and any number of data values

Let $\kappa \geq 1$. We first show that $\kappa\mathbb{DMS}$-$\textsc{Sat}(\exists\text{-1-}\mathsf{LF}^{\text{ext}};\mathcal{A}_\kappa)$ is in N$\textsc{Exp}$ by providing a reduction towards $\mathbb{MS}$-$\textsc{Sat}(\mathsf{FO};\emptyset)$. This reduction uses the characterisation of the radius-1-ball provided by Lemma 5.1.1 and is very similar to the reduction provided in the previous section. In fact, for an element b located in the radius-1-ball of another element a, we use extra unary predicates to explicit which are the values of b that are common with the values of a. We provide here the main step of this reduction whose proof follows the same line as the one of Theorem 5.1.6.

We consider a formula $\varphi = \exists x_1 \dots \exists x_n . \varphi_{qf}(x_1, \dots, x_n)$ of $\exists$-1-$\mathsf{LF}_\kappa^{\text{ext}}[\Sigma; \mathcal{A}_\kappa]$ in prenex normal form, i.e., such that $\varphi_{qf}(x_1, \dots, x_n) \in$ qf-1-$\mathsf{LF}_\kappa^{\text{ext}}[\Sigma; \mathcal{A}_\kappa]$. We know that there is a structure $\mathfrak{A} = (A, (P_\sigma)_{\sigma\in\Sigma}, f_1, f_2, \dots, f_\kappa)$ in $\text{Str}(\Sigma; \kappa\mathbb{DMS})$ such that $\mathfrak{A} \models \varphi$ if and only if there are $a_1, \dots, a_n \in A$ such that $\mathfrak{A} \models \varphi_{qf}(a_1, \dots, a_n)$. Let then $\mathfrak{A} = (A, (P_\sigma)_{\sigma\in\Sigma}, f_1, f_2, \dots, f_\kappa)$ in $\text{Str}(\Sigma; \kappa\mathbb{DMS})$ and a tuple $\vec{a} = (a_1, \dots, a_n)$ of elements in A^n. Let $\Omega_n = \{a_p[i,j] \mid p \in \{1, \dots, n\}, i, j \in \{1, \dots, \kappa\}\}$ be a set of new unary predicates and $\Sigma' = \Sigma \cup \Omega_n$. For every element $b \in A$, the predicates in Ω_n are used to keep track of the relation between the data values of b and the one of $a_1, \dots, a_n$ in $\mathfrak{A}$. Formally, we have $P_{a_p[i,j]} = \{b \in A \mid \mathfrak{A} \models a_p \ {}_i{\sim}_j \ b\}$. Finally, we build the 0-data-structure $[\![\mathfrak{A}]\!]'_{\vec{a}} = (A, (P_\sigma)_{\sigma\in\Sigma'})$. Similarly to Lemma 5.1.2, we have the following connection between $\mathfrak{A}$ and $[\![\mathfrak{A}]\!]'_{\vec{a}}$.

Lemma 5.1.7. *Let $b, c \in A$ and $j, k \in \{1, \dots, \kappa\}$ and $p \in \{1, \dots, n\}$. The following statements hold:*

1. *If $(b, j) \in B_1^{\mathfrak{A}}(a_p)$ and $(c, k) \in B_1^{\mathfrak{A}}(a_p)$ then $\mathfrak{A}|_{a_p}^{1,\text{ext}} \models b \ {}_j{\sim}_k \ c$ iff there is $i \in \{1, 2\}$ s.t. $b \in P_{a_p[i,j]}$ and $c \in P_{a_p[i,k]}$.*

2. *If $(b, j) \notin B_1^{\mathfrak{A}}(a_p)$ and $(c, k) \in B_1^{\mathfrak{A}}(a_p)$ then $\mathfrak{A}|_{a_p}^{1,\text{ext}} \not\models b \ {}_j{\sim}_k \ c$.*

3. *If $(b, j) \notin B_1^{\mathfrak{A}}(a_p)$ and $(c, k) \notin B_1^{\mathfrak{A}}(a_p)$ then $\mathfrak{A}|_{a_p}^{1,\text{ext}} \models b \ {}_j{\sim}_k \ c$ iff $b = c$ and $j = k$.*

We now see how we translate the formula $\varphi_{qf}(x_1, \dots, x_n)$ into a formula $[\![\varphi_{qf}]\!]'(x_1, \dots, x_n)$ in $\mathsf{FO}_{\mathbb{MS}}[\Sigma'; \emptyset]$ such that $\mathfrak{A}$ satisfies $\varphi_{qf}(a_1, \dots, a_n)$ iff $[\![\mathfrak{A}]\!]'_{\vec{a}}$ satisfies $[\![\varphi_{qf}]\!]'(a_1, \dots, a_n)$. As in the previous section, we introduce the following formula in $\mathsf{FO}_{\mathbb{MS}}[\Sigma'; \emptyset]$ with $p \in \{1, \dots, n\}$ and $j \in \{1, \dots, \kappa\}$ to test if the j-th field of an element belongs to $B_1^{\mathfrak{A}}(a_p)$:

$$\varphi_{j,B_1(a_p)}(y) := \bigvee_{i\in\{1,\dots,\kappa\}} a_p[i,j](y)$$

We now present how we translate atomic formulas of the form $y \ {}_j{\sim}_k \ z$ under some $\langle\!\langle - \rangle\!\rangle_{x_p}^{1,\text{ext}}$. For this matter, we rely on two formulas of $\mathsf{FO}_{\mathbb{MS}}[\Sigma'; \emptyset]$ which can be described as follows:

- The first formula asks for (y, j) and (z, k) to be in $B_1^{\mathfrak{A}}(a_p)$ (here we abuse notations, using variables for the elements they represent) and for these two data values to coincide with one data value of a_p, it corresponds to Lemma 5.1.7.1:

$$\psi_{j,k,a_p}^{r=1}(y, z) := \varphi_{j,B_1(a_p)}(y) \wedge \varphi_{k,B_1(a_p)}(z) \wedge \bigvee_{i=1}^{\kappa} \big(a_p[i,j](y) \wedge a_p[i,k](z)\big)$$

- The second formula asks for (y, j) and (z, k) to not belong to $B_1^{\mathfrak{A}}(a_p)$ and for $y = z$, it corresponds to Lemma 5.1.7.3:

$$\psi^{r>1}_{j,k,a_p}(y,z) := \begin{cases} \bigwedge_{i=1}^{\kappa}(\neg\varphi_{i,B_1(a_p)}(y) \wedge \neg\varphi_{i,B_1(a_p)}(z)) \,\wedge\, y = z & \text{if } j = k \\ \bot & \text{otherwise} \end{cases}$$

Finally, as before we provide an inductive definition of the translation $[\![-]\!]'$ which uses subtransformations $[\![-]\!]'_{x_p}$ in order to remember the centre of the ball and leads to the construction of $[\![\varphi_{qf}]\!]'(x_1,\ldots,x_n)$. We only detail the case

$$[\![y \ {}_j\!\sim_k\, z]\!]'_{x_p} = \psi^{r=1}_{j,k,a_p}(y,z) \,\vee\, \psi^{r>1}_{j,k,a_p}(y,z)$$

as the other cases are identical as for the translation $[\![-]\!]$ shown in the previous section. This leads to the following lemma (which is the pendant of Lemma 5.1.3).

Lemma 5.1.8. *We have $\mathfrak{A} \models \varphi_{qf}(\vec{a})$ iff $[\![\mathfrak{A}]\!]'_{\vec{a}} \models [\![\varphi_{qf}]\!]'(\vec{a})$.*

As we had to characterise the well-formed 1-data-multiset, a similar trick is necessary here. For this matter, we use the following formulas:

$$\psi_{tran} = \forall y \forall z. \bigwedge_{p,q=1}^{n} \bigwedge_{i,j,k,\ell=1}^{\kappa} \Big(a_p[i,j](y) \,\wedge\, a_p[i,\ell](z) \,\wedge\, a_q[k,j](y) \,\rightarrow\, a_q[k,\ell](z)\Big)$$

$$\psi_{refl}(x_1,\ldots,x_n) = \bigwedge_{p=1}^{n} \bigwedge_{i=1}^{\kappa} a_p[i,i](x_p)$$

$$\psi_{wf}(x_1,\ldots,x_n) = \psi_{tran} \,\wedge\, \psi_{refl}(x_1,\ldots,x_n)$$

Finally with the same reasoning as the one given in the previous section, we can show that the formula $\varphi = \exists x_1 \ldots \exists x_n.\varphi_{qf}(x_1,\ldots,x_n)$ of $\exists$-1-$\mathsf{LF}^{\text{ext}}_{\kappa}[\Sigma;\mathcal{A}_\kappa]$ is satisfiable iff the formula $\exists x_1 \ldots \exists x_n.[\![\varphi_{qf}]\!]'(x_1,\ldots,x_n) \wedge \psi_{wf}(x_1,\ldots,x_n)$ is satisfiable. Note that this latter formula can be built in polynomial time from φ and that it belongs to $\mathsf{FO}_{\mathbb{MS}}[\Sigma';\emptyset]$. Hence, thanks to Theorem 2.3.9 which states that the problem $\mathbb{MS}$-$\textsc{Sat}(\mathsf{FO};\emptyset)$ is in $\text{NE}\textsc{xp}$, we obtain that $\kappa\mathbb{DMS}$-$\textsc{Sat}(\exists\text{-1-}\mathsf{LF}^{\text{ext}};\mathcal{A}_\kappa)$ is in $\text{NE}\textsc{xp}$. The matching lower bound is as well obtained the same way by reducing $\mathbb{MS}$-$\textsc{Sat}(\mathsf{FO};\emptyset)$ to $\kappa\mathbb{DMS}$-$\textsc{Sat}(\exists\text{-1-}\mathsf{LF}^{\text{ext}};\mathcal{A}_\kappa)$ showing that a formula φ in $\mathsf{FO}_{\mathbb{MS}}[\Sigma;\emptyset]$ is satisfiable iff the formula $\exists x.\langle\!\langle\varphi\rangle\!\rangle^{1,\text{ext}}_{x}$ in $\exists$-1-$\mathsf{LF}^{\text{ext}}_{\kappa}[\Sigma;\mathcal{A}_\kappa]$ is satisfiable.

Theorem 5.1.9

For all $\kappa \geq 1$, the problem $\kappa\mathbb{DMS}$-$\textsc{Sat}(\exists\text{-1-}\mathsf{LF}^{\text{ext}};\mathcal{A}_\kappa)$ is $\text{NE}\textsc{xp}$-complete.

5.2 Undecidability results

We show here $2\mathbb{DMS}$-$\textsc{Sat}(\exists\text{-3-}\mathsf{LF}^{\text{ext}};\mathcal{A}_2)$ and $3\mathbb{DMS}$-$\textsc{Sat}(\exists\text{-2-}\mathsf{LF}^{\text{ext}};\mathcal{A}_3)$ are undecidable. To obtain this we provide reductions from $2\mathbb{DMS}$-$\textsc{Sat}(\mathsf{FO};\mathcal{A}_2)$ and we use the fact that any 2-data-multiset can be interpreted as a radius-3-ball of a 2-data-multiset or respectively as a radius-2-ball of a 3-data-multiset.

5.2.1 Radius 3 and two data values

In order to reduce $2\mathbb{DMS}\text{-}\textsc{Sat}(\mathsf{FO};\mathcal{A}_2)$ to $2\mathbb{DMS}\text{-}\textsc{Sat}(\exists\text{-}3\text{-}\mathsf{LF}^{\text{ext}};\mathcal{A}_2)$, we show that we can transform slightly any 2-data-multiset $\mathfrak{A}$ into an other 2-data-multiset $\mathfrak{A}_{\mathsf{ge}}$ such that $\mathfrak{A}_{\mathsf{ge}}$ corresponds to the radius-3-ball of any element of $\mathfrak{A}_{\mathsf{ge}}$ and this transformation has some kind of inverse. Furthermore, given a formula $\varphi \in \mathsf{FO}_{2\mathbb{DMS}}[\Sigma;\mathcal{A}_2]$, we transform it into a formula $T(\varphi)$ in $\exists\text{-}3\text{-}\mathsf{LF}_2^{\text{ext}}[\Sigma';\mathcal{A}_2]$ such that $\mathfrak{A}$ satisfies φ iff $\mathfrak{A}_{\mathsf{ge}}$ satisfies $T(\varphi)$. What follows is the formalisation of this reasoning.

Let $\mathfrak{A} = (A,(P_\sigma)_\sigma,f_1,f_2)$ be a 2-data structure in $\mathrm{Str}(\Sigma;2\mathbb{DMS})$ and ge be a fresh unary predicate not in Σ. From $\mathfrak{A}$ we build the following 2-data-multiset $\mathfrak{A}_{\mathsf{ge}} = (A',(P'_\sigma)_\sigma,f'_1,f'_2) \in \mathrm{Str}(\Sigma\cup\{\mathsf{ge}\};2\mathbb{DMS})$ such that:

- $A' = A \uplus \Big(\mathit{Val}^{\mathfrak{A}}(A) \times \mathit{Val}^{\mathfrak{A}}(A) \Big)$,
- for $i \in \{1,2\}$ and $a \in A$, $f'_i(a) = f_i(a)$ and for $(d_1,d_2) \in \mathit{Val}^{\mathfrak{A}}(A) \times \mathit{Val}^{\mathfrak{A}}(A)$, $f_i((d_1,d_2)) = d_i$,
- for $\sigma \in \Sigma$, $P'_\sigma = P_\sigma$,
- $P_{\mathsf{ge}} = \mathit{Val}^{\mathfrak{A}}(A) \times \mathit{Val}^{\mathfrak{A}}(A)$.

We have then the following property.

Lemma 5.2.1. $\mathfrak{A}_{\mathsf{ge}}|_a^{3,\text{ext}} = \mathfrak{A}_{\mathsf{ge}}$ *for all* $a \in A'$.

Proof: Let $b \in A'$ and $i,j \in \{1,2\}$. We show that $d^{\mathfrak{A}_{\mathsf{ge}}}((a,i),(b,j)) \leq 3$. i.e. that there is a path of length at most 3 from (a,i) to (b,j) in the data graph $\mathcal{G}(\mathfrak{A}_{\mathsf{ge}})$. By construction of $\mathfrak{A}_{\mathsf{ge}}$, there is an element $c \in A'$ such that $f_1(c) = f_i(a)$ and $f_2(c) = f_j(b)$. So we have the path $(a,i),(c,1),(c,2),(b,j)$ of length at most 3 from (a,i) to (b,j) in $\mathcal{G}(\mathfrak{A}_{\mathsf{ge}})$. □

Conversely, to $\mathfrak{A} = (A,(P_\sigma)_\sigma,f_1,f_2) \in \mathrm{Str}(\Sigma\cup\{\mathsf{ge}\};2\mathbb{DMS})$, we associate $\mathfrak{A}_{\backslash\mathsf{ge}} = (A',(P'_\sigma)_\sigma,f'_1,f'_2) \in \mathrm{Str}(\Sigma;2\mathbb{DMS})$ where:

- $A' = A \setminus P_{\mathsf{ge}}$,
- for $i \in \{1,2\}$ and $a \in A'$, $f'_i(a) = f_i(a)$,
- for $\sigma \in \Sigma$, $P'_\sigma = P'_\sigma \setminus P_{\mathsf{ge}}$.

Finally we inductively translate any formula $\varphi \in \mathsf{FO}_{2\mathbb{DMS}}[\Sigma;\mathcal{A}_2]$ into $T(\varphi) \in \mathsf{FO}_{2\mathbb{DMS}}[\Sigma\cup\{\mathsf{ge}\};\mathcal{A}_2]$ by making it quantify over elements not labeled with ge:

$$\begin{aligned}
T(\sigma(x)) &= \sigma(x),\\
T(x \;{}_i{\sim}_j\; y) &= x \;{}_i{\sim}_j\; y,\\
T(x = y) &= (x = y),\\
T(\exists x.\varphi) &= \exists x.\neg\mathsf{ge}(x) \wedge T(\varphi),\\
T(\varphi \vee \varphi') &= T(\varphi) \vee T(\varphi'),\\
T(\neg\varphi) &= \neg T(\varphi).
\end{aligned}$$

Lemma 5.2.2. *Let φ be a sentence in* $\mathsf{FO}_{2\mathbb{DMS}}[\Sigma;\mathcal{A}_2]$, $\mathfrak{A} \in \mathrm{Str}(\Sigma;2\mathbb{DMS})$ *and* $\mathfrak{B} \in \mathrm{Str}(\Sigma \cup \{\mathsf{ge}\};2\mathbb{DMS})$. *The two following properties hold:*

- $\mathfrak{A} \models \varphi$ *iff* $\mathfrak{A}_{\mathsf{ge}} \models T(\varphi)$
- $\mathfrak{B}_{\setminus\mathsf{ge}} \models \varphi$ *iff* $\mathfrak{B} \models T(\varphi)$.

Proof: As for any $\mathfrak{A} \in \mathrm{Str}(\Sigma;2\mathbb{DMS})$ we have $(\mathfrak{A}_{\mathsf{ge}})_{\setminus\mathsf{ge}} = \mathfrak{A}$, it is sufficient to prove the second point. We reason by induction on φ. Let $\mathfrak{A} = (A,(P_\sigma)_\sigma,f_1,f_2) \in \mathrm{Str}(\Sigma \cup \{\mathsf{ge}\};2\mathbb{DMS})$ and let $\mathfrak{A}_{\setminus\mathsf{ge}} = (A',(P'_\sigma)_\sigma,f'_1,f'_2) \in \mathrm{Str}(\Sigma;2\mathbb{DMS})$. The inductive hypothesis is that for any formula $\varphi \in \mathsf{FO}_{2\mathbb{DMS}}[\Sigma;\mathcal{A}_2]$ (closed or not) and any context interpretation function $I : \mathcal{V} \to A'$ we have $\mathfrak{A}_{\setminus\mathsf{ge}} \models_I \varphi$ iff $\mathfrak{A} \models_I T(\varphi)$. Note that the inductive hypothesis is well founded in the sense that the interpretation I always maps variables to elements of the structures.

We prove two cases: when φ is a unary predicate and when φ starts by an existential quantification, the other cases being similar. First, assume that $\varphi = \sigma(x)$ where $\sigma \in \Sigma$. $\mathfrak{A}_{\setminus\mathsf{ge}} \models_I \sigma(x)$ holds iff $I(x) \in P'_\sigma$. As $I(x) \in A \setminus P_{\mathsf{ge}}$, we have $I(x) \in P'_\sigma$ iff $I(x) \in P_\sigma$, which is equivalent to $\mathfrak{A} \models_I T(\sigma(x))$. Second assume $\varphi = \exists x.\varphi'$. Suppose that $\mathfrak{A}_{\setminus\mathsf{ge}} \models_I \exists x.\varphi'$. Thus, there is a $a \in A'$ such that $\mathfrak{A}_{\setminus\mathsf{ge}} \models_{I[x/a]} \varphi'$. By inductive hypothesis, we have $\mathfrak{A} \models_{I[x/a]} T(\varphi')$. As $a \in A' = A \setminus P_{\mathsf{ge}}$, we have $\mathfrak{A} \models_{I[x/a]} \neg\mathsf{ge}(x)$, so $\mathfrak{A} \models_I \exists x.\neg\mathsf{ge}(x) \wedge T(\varphi')$ as desired. Conversely, suppose that $\mathfrak{A} \models_I T(\exists x.\varphi')$. It means that there is a $a \in A$ such that $\mathfrak{A} \models_{I[x/a]} \neg\mathsf{ge}(x) \wedge T(\varphi')$. So we have that $a \in A' = A \setminus P_{\mathsf{ge}}$, which means that $I[x/a]$ takes values in A and we can apply the inductive hypothesis to get that $\mathfrak{A}_{\setminus\mathsf{ge}} \models_{I[x/a]} \varphi'$. So we have $\mathfrak{A}_{\setminus\mathsf{ge}} \models_I \exists x.\varphi'$. □

From Theorem 2.3.27, we know that $2\mathbb{DMS}$-SAT$(\mathsf{FO};\mathcal{A}_2)$ is undecidable. From a closed formula $\varphi \in \mathsf{FO}_{2\mathbb{DMS}}[\Sigma;\mathcal{A}_2]$, we build the formula $\exists x.\langle\langle T(\varphi)\rangle\rangle_x^{3,\mathrm{ext}} \in \exists\text{-}3\text{-}\mathsf{LF}_2^{\mathrm{ext}}[\Sigma \cup \{\mathsf{ge}\};\mathcal{A}_2]$. Now if φ is satisfiable, it means that there exists $\mathfrak{A} \in \mathrm{Str}(\Sigma;2\mathbb{DMS})$ such that $\mathfrak{A} \models \varphi$. By Lemma 5.2.2, $\mathfrak{A}_{\mathsf{ge}} \models T(\varphi)$. Let a be an element of $\mathfrak{A}$, then thanks to Lemma 5.2.1, we have $\mathfrak{A}_{\mathsf{ge}}|_a^{3,\mathrm{ext}} \models T(\varphi)$. Finally by definition of our logic, $\mathfrak{A}_{\mathsf{ge}} \models \exists x.\langle\langle T(\varphi)\rangle\rangle_x^{3,\mathrm{ext}}$. So $\exists x.\langle\langle T(\varphi)\rangle\rangle_x^{3,\mathrm{ext}}$ is satisfiable. Now assume that $\exists x.\langle\langle T(\varphi)\rangle\rangle_x^{3,\mathrm{ext}}$ is satisfiable. So there exist $\mathfrak{A} \in \mathrm{Str}(\Sigma \cup \{\mathsf{ge}\};2\mathbb{DMS})$ and an element a of $\mathfrak{A}$ such that $\mathfrak{A}|_a^{3,\mathrm{ext}} \models T(\varphi)$. Using Lemma 5.2.2, we obtain $(\mathfrak{A}|_a^{3,\mathrm{ext}})_{\setminus\mathsf{ge}} \models \varphi$. Hence φ is satisfiable. This shows that we can reduce $2\mathbb{DMS}$-SAT$(\mathsf{FO};\mathcal{A}_2)$ to $2\mathbb{DMS}$-SAT$(\exists\text{-}3\text{-}\mathsf{LF}^{\mathrm{ext}};\mathcal{A}_2)$.

Theorem 5.2.3

The problem $2\mathbb{DMS}$-SAT$(\exists\text{-}3\text{-}\mathsf{LF}^{\mathrm{ext}};\mathcal{A}_2)$ *is undecidable.*

5.2.2 Radius 2 and three data values

We provide here a reduction from $2\mathbb{DMS}$-SAT$(\mathsf{FO};\mathcal{A}_2)$ to $3\mathbb{DMS}$-SAT$(\exists\text{-}2\text{-}\mathsf{LF}^{\mathrm{ext}};\mathcal{A}_3)$. The idea is similar to the one used in the proof of Lemma 5.1.5 to show that the problem $2\mathbb{DMS}$-SAT$(\exists\text{-}2\text{-}\mathsf{LF}^{\mathrm{ext}};\mathcal{A}_2)$ is N2EXP-hard by reducing $1\mathbb{DMS}$-SAT$(\mathsf{FO};\{\sim\})$. Indeed we have the following Lemma.

Lemma 5.2.4. *Let φ be a formula in* $\mathsf{FO}_{2\mathbb{DMS}}[\Sigma;\mathcal{A}_2]$. *There exists* $\mathfrak{A} \in \mathrm{Str}(\Sigma; 2\mathbb{DMS})$ *such that* $\mathfrak{A} \models \varphi$ *if and only if there exists* $\mathfrak{B} \in \mathrm{Str}(\Sigma; 3\mathbb{DMS})$ *such that* $\mathfrak{B} \models \exists x.\langle\!\langle \varphi \rangle\!\rangle_x^{2,\mathrm{ext}}$.

Proof: Assume that there exists $\mathfrak{A} = (A, (P_\sigma)_{\sigma\in\Sigma}, f_1, f_2)$ in $\mathrm{Str}(\Sigma; 2\mathbb{DMS})$ such that $\mathfrak{A} \models \varphi$. Consider the 3-data-multiset $\mathfrak{B} = (A, (P_\sigma)_{\sigma\in\Sigma}, f_1, f_2, f_3)$ such that $f_3(a) = 0$ for all $a \in A$. Let $a \in A$. It is clear that we have $\mathfrak{B}|_a^{2,\mathrm{ext}} = \mathfrak{B}$ and that $\mathfrak{B}|_a^{2,\mathrm{ext}} \models \varphi$ (because $\mathfrak{A} \models \varphi$ and φ never mentions the third values of the elements since it is a formula in $\mathsf{FO}_{2\mathbb{DMS}}[\Sigma;\mathcal{A}_2]$). Consequently $\mathfrak{B} \models \exists x.\langle\!\langle \varphi \rangle\!\rangle_x^{2,\mathrm{ext}}$.

Assume now that there exists $\mathfrak{B} = (A, (P_\sigma)_{\sigma\in\Sigma}, f_1, f_2, f_3)$ in $\mathrm{Str}(\Sigma; 3\mathbb{DMS})$ such that $\mathfrak{B} \models \exists x.\langle\!\langle \varphi \rangle\!\rangle_x^{2,\mathrm{ext}}$. Hence there exists $a \in A$ such that $\mathfrak{B}|_a^{2,\mathrm{ext}} \models \varphi$, but then by forgetting the thir value in $\mathfrak{B}|_a^{2,\mathrm{ext}}$ we obtain a model in $\mathrm{Str}(\Sigma; 3\mathbb{DMS})$ which satisfies φ. $\square$

Using Theorem 2.3.27, we obtain the following result.

Theorem 5.2.5

The problem $3\mathbb{DMS}$-$\textsc{Sat}(\exists\text{-}2\text{-}\mathsf{LF}^{\mathrm{ext}};\mathcal{A}_3)$ *is undecidable.*

Conclusion

In this work, we have tried to pursue an exhaustive study in order to determine when the satisfiability for data logics and other related logics is decidable. We started by giving an overview of the already existing results. Then we focus solely on the local first order logic with data. The results we have obtained for the local fragments are sum up in Table 5.1. We observe that even if we consider the existential fragment, as soon as the view of the elements has a radius bigger than 3, the satisfiability problem is undecidable. This table allows us as well to see what are the missing elements in order to fully characterise the decidability status of this satisfiability problem. In particular, we do not know whether the decidability result of Theorem 4.2.19 still holds when considering two diagonal relations, but our proof technique does not seem to directly apply. It would be as well very interesting to establish the relative expressive power of these logics and to compare them with some other well-known fragments as for instance the guarded fragment. Finally, another research direction would be to see how our logic could be used to verify distributed algorithms with data (each element in our model representing a process).

Logic	**r**	κ	$\mathcal{R}$	**Decidability status**
$\mathsf{FO}_{\kappa\mathbb{DMS}}[\Sigma;\mathcal{R}]$	–	0	$\emptyset$	NExp-complete *(Thm 2.3.9[12, 19])*
	–	1	$\{{}_1{\sim}_1\}$	N2EXP-complete *(Thm 2.3.25[40, 20])*
	–	2	$\{{}_1{\sim}_1, {}_2{\sim}_2\}$	Undecidable *(Thm 2.3.27[29])*
r-$\mathsf{LF}^{\text{int}}_{\kappa}[\Sigma;\mathcal{R}]$	1	2	$\{{}_1{\sim}_1, {}_2{\sim}_2, {}_1{\sim}_2\}$	Decidable *(Thm 4.2.19)*
	1	2	$\{{}_1{\sim}_1, {}_2{\sim}_2, {}_2{\sim}_1\}$	Decidable *(Thm 4.2.19)*
	2	2	$\{{}_1{\sim}_1, {}_2{\sim}_2, {}_1{\sim}_2\}$	Undecidable *(Thm 4.3.9)*
	3	2	$\{{}_1{\sim}_1, {}_2{\sim}_2\}$	Undecidable *(Thm 4.3.5)*
$\exists$-r-$\mathsf{LF}^{\text{ext}}_{\kappa}[\Sigma;\mathcal{R}]$	1	≥ 1	$\{{}_1{\sim}_1\}$	NEXP-complete *(Thm 5.1.9)*
	2	2	$\{{}_1{\sim}_1, {}_2{\sim}_2, {}_1{\sim}_2, {}_2{\sim}_1\}$	N2EXP-complete *(Thm5.1.6)*
	3	2	$\{{}_1{\sim}_1, {}_2{\sim}_2, {}_1{\sim}_2, {}_2{\sim}_1\}$	Undecidable *(Thm 5.2.3)*
	2	3	$\mathcal{A}_3$	Undecidable *(Thm 5.2.5)*

Table 5.1: Summary of the results for the satisfiability problem

Index

Bibliography

[1] C. Aiswarya, B. Bollig, and P. Gastin. An automata-theoretic approach to the verification of distributed algorithms. *Inf. Comput.*, 259(Part 3):305–327, 2018.

[2] H. Andréka, I. Németi, and J. van Benthem. Modal languages and bounded fragments of predicate logic. *J. Philos. Log.*, 27(3):217–274, 1998. doi:10.1023/A:1004275029985.

[3] R. Beers. Pre-rtl formal verification: An intel experience. In *Proceedings of the 45th Annual Design Automation Conference*, DAC '08, page 806–811. Association for Computing Machinery, 2008. URL: https://doi.org/10.1145/1391469.1391675, doi:10.1145/1391469.1391675.

[4] H. Björklund and M. Bojanczyk. Shuffle expressions and words with nested data. In Ludek Kucera and Antonín Kucera, editors, *Mathematical Foundations of Computer Science 2007, 32nd International Symposium, MFCS 2007, Ceský Krumlov, Czech Republic, August 26-31, 2007, Proceedings*, volume 4708 of *Lecture Notes in Computer Science*, pages 750–761. Springer, 2007.

[5] R. Bloem, S. Jacobs, A. Khalimov, I. Konnov, S. Rubin, H. Veith, and J. Widder. *Decidability of Parameterized Verification.* Synthesis Lectures on Distributed Computing Theory. Morgan & Claypool Publishers, 2015.

[6] M. Bojanczyk, C. David, A. Muscholl, T. Schwentick, and L. Segoufin. Two-Variable Logic on Data Trees and XML Reasoning. In *twenty-fifth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database system*, pages 10–19, Chicago, United States, 2006. ACM Press. URL: https://hal.science/hal-00151833, doi:10.1145/1142351.1142354.

[7] M. Bojanczyk, C. David, A. Muscholl, T. Schwentick, and L. Segoufin. Two-variable logic on data words. *ACM Trans. Comput. Log.*, 12(4):27:1–27:26, 2011.

[8] B. Bollig, P. Bouyer, and F. Reiter. Identifiers in registers - describing network algorithms with logic. In *FOSSACS'19*, volume 11425 of *LNCS*, pages 115–132. Springer, 2019.

[9] B. Bollig and D. Kuske. An optimal construction of hanf sentences. *J. Appl. Log.*, 10(2):179–186, 2012. URL: https://doi.org/10.1016/j.jal.2012.01.002, doi:10.1016/j.jal.2012.01.002.

[10] B. Bollig, A. Sangnier, and O. Stietel. Local first-order logic with two data values. In *FSTTCS'21*, volume 213 of *LIPIcs*, pages 39:1–39:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021.

[11] B. Bollig, A. Sangnier, and O. Stietel. On the existential fragments of local first-order logics with data. In Pierre Ganty and Dario Della Monica, editors, *Proceedings of the 13th International Symposium on Games, Automata, Logics and Formal Verification, GandALF 2022, Madrid, Spain, September 21-23, 2022*, volume 370 of *EPTCS*, pages 1–16, 2022. URL: https://doi.org/10.4204/EPTCS.370.1, doi:10.4204/EPTCS.370.1.

[12] E. Börger, E. Grädel, and Y. Gurevich. *The Classical Decision Problem.* Perspectives in Mathematical Logic. Springer, 1997.

[13] S. A. Cook. The complexity of theorem-proving procedures. In *Proceedings of the Third Annual ACM Symposium on Theory of Computing*, STOC '71, page 151–158. Association for Computing Machinery, 1971. URL: https://doi.org/10.1145/800157.805047, doi:10.1145/800157.805047.

[14] A. Dawar, M. Grohe, S. Kreutzer, and N. Schweikardt. Model theory makes formulas large. In *Automata, Languages and Programming, 34th International Colloquium, ICALP 2007, Wroclaw, Poland, July 9-13, 2007, Proceedings*, pages 913–924, 2007. URL: https://doi.org/10.1007/978-3-540-73420-8_78, doi:10.1007/978-3-540-73420-8_78.

[15] N. Decker, P. Habermehl, M. Leucker, and D. Thoma. Ordered navigation on multi-attributed data words. In Paolo Baldan and Daniele Gorla, editors, *CONCUR 2014 - Concurrency Theory - 25th International Conference, CONCUR 2014, Rome, Italy, September 2-5, 2014. Proceedings*, volume 8704 of *Lecture Notes in Computer Science*, pages 497–511. Springer, 2014.

[16] H.-D. Ebbinghaus and J. Flum. *Finite model theory.* Perspectives in Mathematical Logic. Springer, 1995.

[17] E. A. Emerson and K. S. Namjoshi. On reasoning about rings. *Int. J. Found. Comput. Sci.*, 14(4):527–550, 2003.

[18] J. Esparza. Keeping a crowd safe: On the complexity of parameterized verification (invited talk). In *STACS'14)*, volume 25 of *LIPIcs*, pages 1–10. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2014.

[19] K. Etessami, M. Y. Vardi, and T. Wilke. First-order logic with two variables and unary temporal logic. In *Proceedings, 12th Annual IEEE Symposium on Logic in*

Computer Science, Warsaw, Poland, June 29 - July 2, 1997, pages 228–235. IEEE Computer Society, 1997. URL: https://doi.org/10.1109/LICS.1997.614950, doi:10.1109/LICS.1997.614950.

[20] M. Fitting. Torben braüner, hybrid logic and its proof-theory, applied logic series volume 37, springer, 2011, pp. XIII+231. ISBN: 978-94-007-0001-7. *Stud Logica*, 100(5):1051–1053, 2012.

[21] W. Fokkink. *Distributed Algorithms: An Intuitive Approach.* MIT Press, 2013.

[22] M. Fortin. FO $=$ FO^3 for linear orders with monotone binary relations. In C. Baier, I. Chatzigiannakis, P. Flocchini, and S. Leonardi, editors, *46th International Colloquium on Automata, Languages, and Programming, ICALP 2019, July 9-12, 2019, Patras, Greece*, volume 132 of *LIPIcs*, pages 116:1–116:13. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019. URL: https://doi.org/10.4230/LIPIcs.ICALP.2019.116, doi:10.4230/LIPIcs.ICALP.2019.116.

[23] H. Gaifman. On local and non-local properties. In J. Stern, editor, *Proceedings of the Herbrand Symposium*, volume 107 of *Studies in Logic and the Foundations of Mathematics*, pages 105–135. Elsevier, 1982. doi:https://doi.org/10.1016/S0049-237X(08)71879-2.

[24] E. Grädel and M. Otto. On logics with two variables. *Theor. Comput. Sci.*, 224(1-2):73–113, 1999. URL: https://doi.org/10.1016/S0304-3975(98)00308-9, doi:10.1016/S0304-3975(98)00308-9.

[25] S. Grumbach and Z. Wu. Logical locality entails frugal distributed computation over graphs (extended abstract). In *WG'09*, volume 5911 of *LNCS*, pages 154–165. Springer, 2009.

[26] E. Grädel, P. G. Kolaitis, and M. Y. Vardi. On the decision problem for two-variable first-order logic. *The Bulletin of Symbolic Logic*, 3(1):53–69, 1997. URL: http://www.jstor.org/stable/421196.

[27] W. Hanf. Model-theoretic methods in the study of elementary logic. In J.W. Addison, L. Henkin, and A. Tarski, editors, *The Theory of Models*, pages 132–145. North Holland, 1965.

[28] N. Immerman and D. Kozen. Definability with bounded number of bound variables. *Inf. Comput.*, 83(2):121–139, 1989. URL: https://doi.org/10.1016/0890-5401(89)90055-2, doi:10.1016/0890-5401(89)90055-2.

[29] A. Janiczak. Undecidability of some simple formalized theories. *Fundamenta Mathematicae*, 40:131–139, 1953.

[30] A. Kara, T. Schwentick, and T. Zeume. Temporal logics on words with multiple data values. In Kamal Lodaya and Meena Mahajan, editors, *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS*

2010, December 15-18, 2010, Chennai, India, volume 8 of *LIPIcs*, pages 481–492. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2010. URL: https://doi.org/10.4230/LIPIcs.FSTTCS.2010.481, doi:10.4230/LIPIcs.FSTTCS.2010.481.

[31] E. Kieronski. Results on the guarded fragment with equivalence or transitive relations. In C.-H. Luke Ong, editor, *CSL'05*, volume 3634 of *LNCS*, pages 309–324. Springer, 2005.

[32] E. Kieronski, J. Michaliszyn, I. Pratt-Hartmann, and L. Tendera. Two-variable first-order logic with equivalence closure. In *2012 27th Annual IEEE Symposium on Logic in Computer Science*, pages 431–440, 2012. doi:10.1109/LICS.2012.53.

[33] E. Kieronski and M. Otto. Small substructures and decidability issues for first-order logic with two variables. In *20th Annual IEEE Symposium on Logic in Computer Science (LICS' 05)*, pages 448–457, 2005. doi:10.1109/LICS.2005.49.

[34] E. Kieronski and L. Tendera. On finite satisfiability of two-variable first-order logic with equivalence relations. In *LICS'09*, pages 123–132. IEEE, 2009.

[35] I. V. Konnov, H. Veith, and J. Widder. What you always wanted to know about model checking of fault-tolerant distributed algorithms. In *PSI'15 in Memory of Helmut Veith*, volume 9609 of *LNCS*, pages 6–21. Springer, 2015.

[36] J. Leroux. The reachability problem for petri nets is not primitive recursive. *CoRR*, abs/2104.12695, 2021. URL: https://arxiv.org/abs/2104.12695, arXiv:2104.12695.

[37] L. Libkin. *Elements of Finite Model Theory*. Texts in Theoretical Computer Science. An EATCS Series. Springer, 2004. URL: http://www.cs.toronto.edu/%7Elibkin/fmt, doi:10.1007/978-3-662-07003-1.

[38] N. A. Lynch. *Distributed Algorithms*. Morgan Kaufmann Publishers Inc., 1996.

[39] A. Manuel and T. Zeume. Two-variable logic on 2-dimensional structures. In Simona Ronchi Della Rocca, editor, *Computer Science Logic 2013 (CSL 2013), CSL 2013, September 2-5, 2013, Torino, Italy*, volume 23 of *LIPIcs*, pages 484–499. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2013.

[40] M. Mundhenk and T. Schneider. The complexity of hybrid logics over equivalence relations. *J. Log. Lang. Inf.*, 18(4):493–514, 2009.

[41] Y. Nakamura. The undecidability of fo3 and the calculus of relations with just one binary relation. In Md. Aquil Khan and Amaldev Manuel, editors, *Logic and Its Applications*, pages 108–120, Berlin, Heidelberg, 2019. Springer Berlin Heidelberg.

[42] M. Otto. Two variable first-order logic over ordered domains. *Journal of Symbolic Logic*, 66(2):685–702, 2001. URL: https://doi.org/10.2307/2695037, doi:10.2307/2695037.

[43] K. Reinhardt. The complexity of translating logic to finite automata. In E. Grädel, W. Thomas, and T. Wilke, editors, *Automata, Logics, and Infinite Games: A Guide to Current Research [outcome of a Dagstuhl seminar, February 2001]*, volume 2500 of *Lecture Notes in Computer Science*, pages 231–238. Springer, 2001. URL: `https://doi.org/10.1007/3-540-36387-4_13`, `doi:10.1007/3-540-36387-4\_13`.

[44] S. Schmitz. Complexity hierarchies beyond elementary. *ACM Trans. Comput. Theory*, 8(1), feb 2016. URL: `https://doi.org/10.1145/2858784`, `doi:10.1145/2858784`.

[45] L. Segoufin. Automata and logics for words and trees over an infinite alphabet. In *CSL'06*, volume 4207 of *LNCS*, pages 41–57. Springer, 2006.

[46] L. J. Stockmeyer. *The complexity of decision problems in automata theory and logic.* PhD thesis, Massachusetts Institute of Technology. Dept. of Electrical Engineering, 1974.

[47] T. Tan. Extending two-variable logic on data trees with order on data values and its automata. *ACM Trans. Comput. Log.*, 15(1):8:1–8:39, 2014.

[48] W. Thomas. Classifying regular events in symbolic logic. *Journal of Computer and System Sciences*, 25(3):360–376, 1982. `doi:https://doi.org/10.1016/0022-0000(82)90016-2`.

[49] W. Thomas. Languages, automata, and logic. In Grzegorz Rozenberg and Arto Salomaa, editors, *Handbook of Formal Languages, Volume 3: Beyond Words*, pages 389–455. Springer, 1997. URL: `https://doi.org/10.1007/978-3-642-59126-6_7`, `doi:10.1007/978-3-642-59126-6\_7`.

[50] B. Trakhtenbrot. Impossibility of an algorithm for the decision problem in finite classes. *Doklady Akademii Nauk SSSR*, 70(4):569–572, 1950.

[51] J. van Benthem. Dynamic bits and pieces. 1997.

[52] P. van Emde Boas. *The Convenience of Tilings*, pages 331–363. CRC Press, 1997. URL: `https://doi.org/10.1201/9780429187490`, `doi:10.1201/9780429187490`.

[53] E. Verhulst, R.T. Boute, J.M. Sampaio Faria, B.H.C. Sputh, and V. Mezhuyev. *Formal Development of a Network-Centric RTOS*. Springer, 2011. URL: `https://doi.org/10.1007/978-1-4419-9736-4`, `doi:10.1007/978-1-4419-9736-4`.