On the complexity of Client-Waiter and Waiter-Client games

Valentin Gledel, Nacim Oijid, Sébastien Tavenas, Stéphan Thomassé

On the complexity of Client-Waiter and Waiter-Client games *

Valentin Gledel[1], Nacim Oijid[2], Sébastien Tavenas[1], and Stéphan Thomassé[3]

[1]Université Savoie Mont Blanc, CNRS UMR5127, LAMA, Chambéry, F-73000, France
[2]Univ Lyon, CNRS, INSA Lyon, UCBL, Centrale Lyon, Univ Lyon 2, LIRIS, UMR5205, F-69622 Villeurbanne, France.
[3]Univ Lyon, EnsL, UCBL, CNRS, LIP, F-69342, LYON Cedex 07, France

Abstract

Positional games were introduced by Hales and Jewett in 1963, and their study became more popular after Erdős and Selfridge's first result on their connection to Ramsey theory and hypergraph coloring in 1973. Several conventions of these games exist, and the most popular one, Maker-Breaker was proved to be PSPACE-complete by Schaefer in 1978. The study of their complexity then stopped for decades, until 2017 when Bonnet, Jamain, and Saffidine proved that Maker-Breaker is $W[1]$-complete when parameterized by the number of moves. The study was then intensified when Rahman and Watson improved Schaefer's result in 2021 by proving that the PSPACE-hardness holds for 6-uniform hypergraphs. More recently, Galliot, Gravier, and Sivignon proved that computing the winner on rank 3 hypergraphs is in P.

We focus here on the Client-Waiter and the Waiter-Client conventions. Both were proved to be NP-hard by Csernenszky, Martin, and Pluhár in 2011, but neither completeness nor positive results were known for these conventions. In this paper, we complete the study of these conventions by proving that the former is PSPACE-complete, even restricted to 6-uniform hypergraphs, and by providing an FPT-algorithm for the latter, parameterized by the size of its largest edge. In particular, the winner of Waiter-Client can be computed in polynomial time in k-uniform hypergraphs for any fixed integer k. Finally, in search of finding the exact bound between the polynomial result and the hardness result, we focused on the complexity of rank 3 hypergraphs in the Client-Waiter convention. We provide an algorithm that runs in polynomial time with an oracle in NP.

*This research was partly supported by the ANR project P-GASE (ANR-21-CE48-0001-01)

1 Introduction

Positional games were introduced by Hales and Jewett [HJ63] as a generalization of the Tic-Tac-Toe game. These games are played on a hypergraph $H = (V, E)$, on which the players alternately claim an unclaimed vertex of V. The Tic-Tac-Toe corresponds to the convention Maker-Maker: the first player who claims all vertices of an edge of E wins. However, since the second player cannot win in Maker-Maker games, and since Maker-Maker games are not hereditary, for the outcome, the study quickly switched to another convention: Maker-Breaker.

In the most studied convention, Maker-Breaker, Maker (one player) tries to claim all the vertices of an edge, while Breaker (the other player) aims to prevent her from doing so. If not explicitly specified, we assume here that Maker plays the first move. The study became more popular in 1973 when Erdős and Selfridge [ES73] obtained the following criterion.

Theorem 1 (Erdős-Selfridge criterion)**.** *Let $H = (V, E)$ be a hypergraph. If*

$$\sum_{e \in E} 2^{-|e|} < \frac{1}{2},$$

then the position is a win for Breaker.

For the biased versions of Maker-Breaker, for example in the $(1 : b)$ one (at each turn Maker selects one vertex then Breaker selects b of them), one can ask for different families of hypergraphs, what is the threshold for b which turn the position for Breaker from loser to winner. Surprisingly, for a large number of hypergraphs families (but not all), it has been found that the threshold for the random game (where both players play randomly) is essentially the same as the threshold for the optimal play. This phenomenon, known as the "probabilistic intuition" was first demonstrated by Chvatal and Erdős [CE78], and then further developed in numerous papers particularly by Beck (see for example the books [Bec02] and [HKSS14]).

Schaefer proved in 1978 that it is PSPACE-complete to determine the winner of a Maker-Breaker game [Sch78] (the problem appears there under the name G_{pos}(POS CNF)). More precisely, he shows the PSPACE-hardness even for hypergraphs of rank at most 11 (i.e., such that the size of the edges is bounded by 11). A simplified proof can be found in [Bys04]. The result was improved in 2021 by Rahman and Watson [RW21]: the problem is PSPACE-complete for k-uniform hypergraphs with $k \geq 6$ (i.e., hypergraphs where all edges have size exactly k). Byskoz [Bys04] also notices that deciding the winner in a Maker-Breaker can be reduced to the same problem in the Maker-Maker convention (up-to increasing by 1 the maximal size of its edges). In particular, deciding who wins in a Maker-Maker game is PSPACE-complete for k-uniform hypergraphs as soon as $k \geq 7$.

On the positive side, Kutz [Kut05] proved in 2005 that the problem is tractable for 2-uniform hypergraphs and 3-uniform linear ones[1]. It was improved

[1] Hypergraphs whose intersection of each pair of edges is of size at most one.

recently by Galliot *et al.* [Gal23, GGS22]: deciding the winner is tractable for rank 3 hypergraphs.

Since the introduction of positional games, the studies have focused on the Maker-Breaker convention. In order to have a better understanding of this convention, Beck [Bec02] introduced the Client-Waiter and Waiter-Client conventions in 2002 under the names Chooser-Picker and Picker-Chooser. Their current names were suggested by Hefetz, Krivelevich, and Tan [HKT16] as they are less ambiguous. In both conventions, Waiter selects two vertices of the hypergraph and offers them to Client. Client then chooses one to claim, and the second one is given to Waiter. If the number of vertices is odd, the last vertex goes to Client. In the Client-Waiter convention, Client wins if he claims all vertices of an edge, otherwise Waiter wins. In the Waiter-Client convention, Waiter wins if she claims all vertices of an edge, otherwise Client wins.

Notice that these conventions can also be seen as variations of Avoider-Enforcer. In particular, the definition we gives for Waiter-Client is the one which appears originally in Beck [Bec02]. But since [HKT16], Waiter-Client is often defined following the Avoider-Enforcer convention: Waiter wins if she can forces Client to claim a whole edge (and if the number of vertices is odd, the last vertex goes to Waiter). One can notice that both definitions correspond in fact to the same game. Unlike the fact that Maker-Breaker and Avoider-Enforcer are very different games, in this "I-cut-you'll-choose way" paradigm, since Client picks a vertex from only two possibilities, it is symmetric to associate the chosen vertex to Client or to Waiter.

The study of Client-Waiter and Waiter-Client games were first motivated by its similarities to Maker-Breaker games. For example, Bednarska-Bzdęga (improving previous results [Bec02, CMP09]) proved that Theorem 1 also holds in Waiter-Client convention [BB13]. Moreover, the "probabilistic intuition" continues to work [Bec02] in these conventions. More recently, this probabilistic method has been stated in a more general case for the biased Waiter-Client H-game by Bednarska-Bzdęga, Hefetz and Łuczak in 2016 [BBHL16]. This conjecture has just been proved recently by Nenadov in 2023 [Nen23]. These similarities led Beck [Bec02] and Csernenszky, Mándity, and Pluhár [CMP09] to conjecture that if Maker wins in a Maker-Breaker game on some hypergraph H going second, then Waiter wins in Waiter-Client. Notice that up to considering the transversal of H, the conjecture also implies that a win for Breaker as a second player implies a win for Waiter in the Client-Waiter game. But this conjecture was disproved by Knox [Kno12] in 2012. Today however, Client-Waiter and Waiter-Client games are studied independently of Maker-Breaker games: Csernenszky [Cse10] proved in 2010 that 7-in-a-row Waiter-Client is a Client win on the infinite grid, while this problem is still open in the Maker-Breaker convention, and Hefetz *et al.* [HKT16] studied several classical games in Waiter-Client convention.

In terms of complexity, only few results were known about Waiter-Client and Client-Waiter games. In contrast to Maker-Breaker, Maker-Maker, Avoider-Avoider and Avoider-Enforcer, which are known to be PSPACE-complete [Sch78, Bys04, BH19, RW21, GO23], the asymmetry between the players moves in

Waiter-Client and Client-Waiter conventions makes it more difficult to obtain reductions. In fact, Waiter has more choices than Client in her moves, which makes most reduction techniques fail. Both problems have been conjectured PSPACE-complete in [CMP09]. In [CMP11], the authors show that both problems are NP-hard. However, in the Waiter-Client convention, the hypergraph has an exponential number of edges but is given in a succint way: it is the transversal of a given hypergraph. This leaves open the question of whether the problem of deciding who wins a Waiter-Client game is still NP-hard when the hypergraph is given via the list of its edges.

The study of the parameterized complexity of combinatorial games emerged roughly together with the study of parameterized problems, and some strong results about complexity theory are due to this study. For instance, Abrahamson, Rodney, Downey, and Fellows[ADF93] proved that $AW[3] = AW[*]$, through the game Geography. Only few results are known on positional games with the parameterized complexity paradigm, and only related to three conventions: Maker-Breaker, Maker-Maker and Avoider-Enforcer. Their study started by Downey and Fellows, conjecturing that Short Generalized Hex was FPT, but it was disproved (unless FPT= W[1]) by Bonnet, Jamain and Saffidine in 2016 [BJS16], proving its W[1]-hardness. Then, in 2017, Bonnet *et al.* [BGL$^+$17] proved that general Maker-Breaker games are W[1]-complete, Avoider-Enforcer games are co-W[1]-complete and general Maker-Maker games are AW[*]-complete.

Notice that the parameterized results obtained on general positional game consider the number of moves as a parameter. This is mostly motivated by the fact that these problems are already PSPACE-hard for bounded rank hypergraphs. In Waiter-Client convention however, this is not the case, and therefore, we study the complexity of determining the winner of Waiter-Client games parameterized by the rank of the hypergraph. Note that, even if the outcome of Client-Waiter games are hereditary (see Lemma 11), in contrast with Maker-Breaker or Avoider-Enforcer games, when the number of moves is bounded, it is not. Indeed, Waiter can control where Client plays, the addition of isolated vertices can be used by Waiter to waste turns. Therefore, the number of moves in this convention is not as relevant as in the others.

High-level description of the results.

We show that, similarly to the Maker-Breaker and Enforcer-Avoider conventions, deciding the winner of a positional game in the Client-Waiter convention is PSPACE-complete. The result was already conjectured in 2009 [CMP09]. It is an improvement of [CMP11] where the problem is shown to be NP-hard. Moreover, we obtain the PSPACE-completeness even for 6-uniform hypergraphs.

Theorem 2. *For $k \geq 6$, Client-Waiter games are* PSPACE*-complete even restricted to k-uniform hypergraphs.*

The containment in PSPACE directly follows from Lemma 2.2 in [Sch78]. So the main point is the hardness part. To prove it, we reduce the problem of

deciding who has the win to the same problem in the following game.

Definition 3 (Paired SAT)**.** *Let φ be a 3-CNF Formula over a set of pairs of variables $X = \{(x_1, y_1), \dots, (x_n, y_n)\}$. The* Paired SAT*-game is played by two players, Satisfier and Falsifier as follows: while there is a variable that has not been assigned a valuation, Satisfier chooses a pair of variables (x_i, y_i) that she has not chosen yet and gives a valuation, $\top$ or $\bot$, to x_i. Then Falsifier gives a valuation to y_i. When all variables are instantiated, Satisfier wins if and only if the valuation they have provided to the x_is and y_is satisfies φ.*

The Paired SAT-game is a new variant of a CNF-game where the play order is closer to the Client-Waiter convention: the first player chooses, at each turn, which variables she plays on and which variable the second player will have to play on.

Again, deciding who wins on this new game is PSPACE-complete. It is proved in Section 3.1 by a reduction from the game 3-QBF (known to be PSPACE-complete since Schaefer's seminal work [Sch78]).

Theorem 4. *Deciding who is the winner of the* Paired SAT*-game is* PSPACE*-complete.*

Then Theorem 2 is obtained by reducing the Paired SAT-game to the Client-Waiter one. This is done by designing a gadget (given in Figure 2) which simulates a pair of variables (x_i, y_i) of the Paired SAT-game. Notice that the reduction only creates a hypergraph of rank 6. But, similarly to the Maker-Breaker games [RW21, Corollary 4] and the Avoider-Enforcer ones [GO23, Lemma 7], Lemma 22 shows that the hypergraph can be turned into a 6-uniform one afterwards.

In the Maker-Breaker convention, as said before, it is known [RW21] that deciding if a position is winning is PSPACE-complete over hypergraphs of rank at most 6. On the other side, the problem is easy for hypergraphs of rank 2 since Maker wins if and only if there are two adjacent 2-edges or the graph contains a singleton edge (result already noticed in [Kut05]). But only recently, after a serie of results [Kut05, RW20], Galliot, Gravier, and Savignon [Gal23, GGS22] showed that the problem is still polynomial for hypergraphs of rank at most 3. The same question arises in the Client-Waiter convention: despite being PSPACE-complete over hypergraphs of rank at most 6, what can we say about the complexity of the problem for hypergraphs of low rank? For hypergraphs of rank 2, it is easily seen that the problem is polynomial (Proposition 23). The question is already non-trivial for hypergraphs of rank 3.

We show that the problem of deciding if a position is winning in a rank 3 hypergraph reduces to the problem of finding a specific structure (called a Tadpole) in a hypergraph. Tadpoles have been generalized to hypergraphs by Galliot, Gravier and Sivignon [GGS22] to handle rank 3 Maker-Breaker games and their definition is recalled in Definition 26. Intuitively, given a and b two vertices of H, an *ab-tadpole* is given by the union of a path from a to b and a cycle containing b such that the structure is simple (two edges intersect only if they

are two consecutive edges of the path or the cycle, or if one is the last edge of the path and the other an edge of the cycle containing b) and linear (the intersection of two intersecting edges is of size exactly 1). More precisely, we require this structure is 3-uniform, simple and linear, which means that all edges have size 3, the intersection of two consecutive edges has always size one, and a same vertex can not happen at two different places of the structure. An *ab* tadpole is simply called *tadpole*. A tadpole is said rooted in a, if it is an ab-tadpole for some vertex b.

We consider the problem TADPOLE: Given a vertex a in a 3-uniform hypergraph H, decide if there is in H a tadpole rooted in a.

It is easy to check that a given subhypergraph is a tadpole, so this problem is in NP. We do not know if this problem can be tractable. We note however that it would be sufficient to loop for all vertex b and all triples of edges of the form $\{b, x_1, x_2\}, \{b, y_1, y_2\}, \{b, z_1, z_2\}$ and check if there are two *disjoint* simple linear paths linking the two sources a and x_1 to the targets y_1 and z_1 (b would be the contact between the path and the cycle). The complexity of the problem "Disjoint Connected Paths" for graphs has been a very fruitful research topic. In the case of two sources and two targets, the problem was shown to be tractable [Shi80, Sey80]. In fact, in their well-known result, Robertson and Seymour [RS95] showed that the problem continues to be tractable for a constant number of sources and targets. The case where the number of sources and targets is unbounded is one of the first NP-complete problems in Karp's list. For 3-uniform hypergraphs, it has been just proved recently [GGS23] that the simple linear connectivity problem (one source and one target) is tractable. A corollary of the next theorem is that if the "Disjoint Connected Paths" problem for two sources and two targets is still tractable for 3-uniform hypergraphs, then deciding who is winning in a Client-Waiter game on a hypergraph of rank 3 would also be tractable.

Theorem 5. *The problem of deciding if a given rank* 3 *hypergraph is a winning position for Client can be solved by a polynomial time algorithm which uses the problem* TADPOLE *as an oracle.*

In particular, the problem lies in the class $\Delta_2^{\mathsf{P}} = \mathsf{P}^{\mathsf{NP}}$.

In fact, during the proof of this theorem, we notice that this reduction is necessary. The problem of detecting a Tadpole can conversely be reduced to deciding on a winning position in a Client-Waiter game.

Proposition 6. *The problem* TADPOLE *is polynomial-time many-one reducible to the problem of deciding if Client has a winning strategy in a Client-Waiter game played on a rank* 3 *hypergraph.*

Then, we focus on the second convention Waiter-Client. Surprisingly, the complexity of deciding if a position is winning is very different in this convention.

Theorem 7. *Waiter-Client is* FPT *on* k*-uniform hypergraphs.*

More precisely deciding if a hypergraph $H = (V, E)$ is winning for Waiter can be decided in time $O(f(k)|E|\log|V|)$ where f is a computable function, i.e., in linear time when k is fixed.

This result is obtained from a structural analysis of k-uniform hypergraphs, using the famous sunflower lemma from Erdős and Rado [ER60]. This is a very natural approach since, intuitively, a huge sunflower (edges pairwise intersecting on a same center set) should be "reducible" in the sense that one could replace the sunflower by a unique edge given by the center set. Indeed, if Waiter can obtain the center and the sunflower is large enough, then she can make sure to get a petal and so a set of the sunflower. This intuition should be valid, but the main difficulty is to exactly state what is "huge", as all other potentially useful sunflowers interact. We were unable to find a simple strategy based on these lines. Instead, we describe a kernelization type algorithm which produces a subset of vertices (kernel) of the hypergraph H such that Waiter wins on H if and only if Waiter wins on the trace of H on the kernel. The argument is based on a process which ultimately reaches a fixed point, the major drawback being that the kernel size is ridiculously large. This kernelization provides an FPT algorithm for the Waiter-Client game on rank k hypergraphs. This also proves that if Waiter can win, she wins using only some function of k moves. However, the gap between the very large upper bound and the best known lower bound ($2^k - 1$ moves, required to win on 2^k disjoint edges of size k) indicates how little we understand strategies in Waiter-Client games.

Nevertheless, the fact that Waiter-Client is FPT for rank k hypergraphs could indicate that this convention is simpler to analyse than Maker-Breaker. It would be interesting to revisit the classical topics in which Maker-Breaker game was tried as a tool (for instance 2-colorability of hypergraphs or the Local Lemma) to see if Waiter-Client could provide more insight.

Complexity results for the various conventions are summarized in the table below.

Organization.

We start with some preliminary results and definition in Section 2. We then prove in Section 3 that Client-Waiter games are PSPACE-complete, even restricted to 6-uniform hypergraphs. In Section 4, we focus on rank 3 hypergraph in Client-Waiter convention and we prove that the decision problem of determining the outcome of the game is in Δ_2^{P}. Finally, in Section 5, we prove that Waiter-Client games are FPT parameterized by the rank.

References to add in the introduction.

- Introduction of Maker-Breaker (dixit Galliot,Gravier,Savignon) by Beck and Csirmaz in 82
- N: c'est pas l'introduction de Maker-Breaker (Chvatal Erdos en 1978 avaient déjà bossé sur un jeu Maker-Breaker), mais c'est l'introduction

Rank r	2	3	4, 5	6	7+
Maker -Breaker	P[Folklore]	P[Gal23]	Open	PSPACE-c [RW21]	PSPACE-c [RW21, Sch78]
Maker -Maker	P[Folklore]	Open	Open	Open	PSPACE-c [RW21, Bys04]
Avoider -Avoider	PSPACE-c [BH19]	PSPACE-c [BH19]	PSPACE-c [BH19]	PSPACE-c [BH19]	PSPACE-c [BH19]
Avoider -Enforcer	P [Gal23]	Open	Open	PSPACE-c [GO23]	PSPACE-c [GO23]
Client -Waiter	P Prop 23	$\Delta_2^{\mathsf{P}} = \mathsf{P}^{\mathsf{NP}}$ Cor 5	Open	PSPACE-c Thm 2	PSPACE-c Thm 2
Waiter -Client	P Prop 37	P Thm 7	P Thm 7	P Thm 7	FPT w.r.t. r Thm 7

Table 1: Complexity in the different conventions

du nom "Maker-Breaker" avant, on disait juste 1er joueur et 2e joueur. Si on prend également en compte les jeux pas introduits comme étant positionnels, on peut remonter encore (mettre Hex dedans serait de la triche, mais on peut au moins mentionner le shannon switching game)

2 Preliminaries

In this section, we first introduce the context of the game by providing some definitions, then we present some useful lemmas to handle Client-Waiter and Waiter-Client games.

Definition 8. *Let $H = (V, E)$ be a hypergraph and let k be an integer. H is said to have* rank k *if all its edges $e \in E$ have size at most k. H is said to be k-uniform if all its edges have size exactly k.*

Remark that, if H is a hypergraph, if it has an edge included in another, we can remove the largest one without changing the outcome of the game played on H. Therefore, we can consider that all hypergraphs considered in this paper are clutter.

Definition 9. *Let $H = (V, E)$ be a hypergraph. H is said to be a* clutter *if for any edges $e_1 \neq e_2 \in E$, we have $e_1 \not\subseteq e_2$ and $e_2 \not\subseteq e_1$.*

Definition 10. *The hypergraph $H' = (V', E')$ is a subhypergraph of $H = (V, E)$ if $V' \subseteq V$ and $E' \subseteq E$. If $A \subseteq V$, the induced subhypergraph $H_{|A}$ is the*

subhypergraph $(A, \{e \in E \mid e \subseteq A\})$. *The trace* $T_A(H)$ *of* H *on* A *is the hypergraph* $(A, \{e \cap A \mid e \in E\})$.

In this section, we present some general results about Client-Waiter and Waiter-Client games. The monotony of Client-Waiter and Waiter-Client conventions is immediate and folkloric: if "Maker" player (i.e. Client in Client-Waiter and Waiter in Waiter-Client) has a winning strategy on a sub-hypergraph, then he has one on the general hypergraph.

Lemma 11. *Let* $H = (V, E)$ *be a hypergraph and let* $H' = (V', E')$ *be a sub-hypergraph of* H. *If Client (resp. Waiter) wins the Client-Waiter (resp. Waiter-Client) game on* H', *then Client (resp. Waiter) wins the Client-Waiter (resp. Waiter-Client) game on* H.

In the Client-Waiter convention, edges of length 2 are forced moves for Waiter.

Lemma 12 (Proposition 9 from [CMP09])**.** *Let* $H = (V, E)$ *be a hypergraph. Let a game in the Client-Waiter convention. Let* $W, C \subset V$ *be the set of vertices already claimed by Waiter and Client respectively. If there exists* $e \in E$ *such that* $e \cap W = \emptyset$ *and* $|e \setminus C| = 2$, *then an optimal move for Waiter is to propose the two unclaimed element of* e *with her next move.*

3 6-uniform Client-Waiter games are PSPACE-complete

This section is dedicated to the proof of Theorem 2. As explained in the introduction, we start with hypergraphs of rank at most 6:

Proposition 13. *Computing the winner of a Client-Waiter game is* PSPACE*-complete, even restricted to hypergraphs of rank* 6.

We notice that the membership in PSPACE follows from an argument of Schaefer [Sch78].

Lemma 14. *Both Client-Waiter and Waiter-Client positional games are in* PSPACE.

Proof. Let $H = (V, E)$ be a hypergraph. Each turn, Waiter has the choice among at most $\binom{|V|}{2}$ moves and Client has the choice among 2 moves. Since the game ends in at most $|V|$ moves, it is in PSPACE using the same proof as in Lemma 2.2 from Schaefer [Sch78]. □

3.1 Quantified Boolean Formula and paired SAT

The most classical PSPACE-complete problem is 3-QBF, the quantified version of SAT. Our hardness proof is a reduction from 3-QBF, but since the roles of the

players in Client-Waiter games are very different, we introduce an intermediate problem Paired SAT.

First we recall the definition of 3-QBF, in its gaming version, as it was done by Rahman and Watson [RW21], and later Gledel and Oijid [GO23] to prove that Maker-Breaker and Avoider-Enforcer games are PSPACE-hard respectively.

Given a 3-CNF quantified formula $\varphi = \exists x_1, \forall y_1, \ldots, \exists x_n \forall y_n \psi$, where ψ is a 3-CNF without quantifier, the 3-QBF game is played by two players, Satisfier and Falsifier. Satisfier chooses the value of x_1, then Falsifier chooses the value of y_1, and so on until the last variable has its value chosen. At the end, a valuation ν of the variables is obtained, and Satisfier wins if and only if ν satisfies ψ.

Theorem 15 (Stockmeyer and Meyer [SM73])**.** *Determining if Satisfier has a winning strategy in the* 3-QBF *game is* PSPACE*-complete.*

The game Paired SAT is introduced in the introduction (Definition 3). This is a variant of 3-QBF where, at each turn, Satisfier chooses an index i and instantiates the variable x_i, and then Falsifier instantiates the variable y_i. The main idea of this game is to introduce a CNF-game mimicking the fact that one player chooses which variable the second player has to play on.

Theorem 16. *Determining the winner of the* Paired SAT*-game is* PSPACE*-complete.*

Proof. Since at any moment of the game, the number of options for a player is at most n and the game runs for at most n turns, determining the winner of the Paired SAT-game is in PSPACE using the same proof as Lemma 2.2 from Schaefer [Sch78].

We provide a reduction from 3-QBF. Let $\psi = \exists x_1 \forall y_1 \cdots \exists x_n \forall y_n, \varphi$ be a QBF formula. We construct an instance of the Paired SAT-game (φ', X) as follows:

- $X = \{(z_0, y_0), (x_1, t_1), (z_1, y_1), \ldots, (x_n, t_n), (z_n, y_n)\}$, where y_0, the z_is, and the t_js are new variables.
- $\varphi' = \varphi \wedge \bigwedge\limits_{1 \leq i \leq n} (y_{i-1} \oplus t_i \oplus z_i)$.

  where $\oplus$ refers to the XOR operator:

  $$a \oplus b \oplus c = (a \vee b \vee c) \wedge (\neg a \vee \neg b \vee c) \wedge (\neg a \vee b \vee \neg c) \wedge (a \vee \neg b \vee \neg c).$$

We prove that Satisfier wins on ψ if and only if she wins on (φ', X).

First note that whenever two values of $a \oplus b \oplus c$ are known, the player who chooses the last value can always decide to satisfy or not $(a \oplus b \oplus c)$. Suppose first that Satisfier has a winning strategy $\mathcal{S}$ on ψ, and consider the following strategy for him on (φ', X):

- Satisfier instantiates the pairs $(z_0, y_0), (x_1, t_1), (x_2, t_2), \ldots, (z_n, y_n)$ in that order.

- Whenever Satisfier has to choose a value for a variable x_i, he follows $\mathcal{S}$ with the corresponding values of the x_j and y_j for $1 \leq j < i$. This is always possible as the order was given above.

- Whenever Satisfier has to give a value to a variable z_i, she gives the value so that $y_{i-1} \oplus t_i \oplus z_i$ is satisfied (if $i = 0$ she can instantiate the variable z_0 by either $\top$ or $\bot$).

Following this strategy, all the clauses $(y_{i-1} \oplus t_i \oplus z_i)$ are satisfied as Satisfier always chooses the last vertex of these clauses (which appears in exactly one of them), and as $\mathcal{S}$ has a winning strategy in ψ, it satisfies φ, as the variables are chosen in the same order.

Now suppose that Falsifier has a winning strategy $\mathcal{S}$ on ψ, and consider the following strategy on (φ', X, Y):

- While Satisfier plays the pairs following the order $(z_0, y_0), (x_1, t_1), (x_2, t_2), \ldots, (z_n, y_n)$, Falsifier gives to the corresponding t_i the value $\bot$, and to y_i the value given by $\mathcal{S}$.

- If Satisfier plays a pair (x_j, t_j) before she should, Falsifier still gives the valuation $\bot$ to t_j and ignores this move while choosing the valuations of the y_i for $i \leq j$.

- If Satisfier plays a vertex z_j before he should, the first time it happens, all the unplayed variables in the clause $(y_{j-1} \oplus t_j \oplus z_j)$ will be played by Falsifier. Therefore, Falsifier can just win by choosing a good valuation for y_{j-1} and t_j.

Following this strategy, if Satisfier instantiates a variable z_i whereas there is $j \leq i$ such that the variable x_j has not yet been instantiated then Falsifier wins. Indeed, the first time it happens, either z_{i-1} or x_i has not been instantiated (otherwise it already happened when Satisfier instantiated z_{i-1}). Consequently, Falsifier wins through the clause $(y_{i-1} \oplus t_i \oplus z_i)$. Otherwise, each time Falsifier has to choose a valuation for a variable y_i, all the vertices x_j with $j \leq i$ have already been played and so, he can play according to $\mathcal{S}$. As $\mathcal{S}$ is a winning strategy, in both case, Falsifier can make a clause unsatisfied and therefore wins the game. □

3.2 Reduction to Client-Waiter games

3.2.1 Blocks in Client-Waiter games

The main tool of several reductions of positional games is pairing strategies. However, this cannot be applied to Client-Waiter games, since only Waiter has choices about how to make the pairs. We present *blocks-hypergraphs* and *block-strategies* that will be used similarly to pairing strategies to ensure that client can claim some vertices. A blocks-hypergraph is depicted in Figure 1. The idea of blocks was already used in the NP-hardness proof from [CMP11], but

we present here a more formal definition of them. Intuitively, Blocks are a generalisation of Lemma 12, which are blocks of size 2.

Definition 17 (Blocks)**.** *Let $H = (V, E)$ be a hypergraph. A* block $B \subset V$ *of size $2k$ is a set of vertices such that $|B| = 2k$ for some $k \geq 1$, and any set of $k + 1$ vertices of B is an edge.*

If H can be partitioned into blocks, we say that H is a blocks-hypergraph*.*

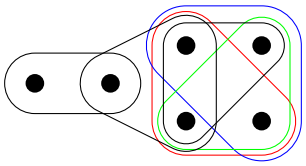

Figure 1: A blocks-hypergraph. The two vertices on the left form a block. The four on the right a second one. The hyperedge between them is in no block

Lemma 18. *Let $H = (V, E)$ be a hypergraph, and let B be a block of H. If Waiter has a winning strategy in H, she has to offer the vertices of B two by two.*

Proof. Suppose that Waiter has a winning strategy in which she does not offer all vertices of B two by two. Let $k = \frac{|B|}{2}$. The first time she presents a vertex $x \in B$ with a vertex $y \notin B$, Client can choose x. Then, each time Waiter offers at least one vertex in B, Client claims it. In the end, Client will claim at least $k + 1$ vertices of B and therefore wins. Thus, if Waiter has a winning strategy, she has to offer the vertices of B two by two. □

Corollary 19. *Let $H = (V, E)$ be a blocks-hypergraph. If Waiter has a winning strategy in H, any pair of vertices she offers belong to a same block of H.*

3.2.2 Construction of the hypergraph

We show now the reduction. The main idea of the reduction is that we construct a blocks-hypergraph, such that each block corresponds to the valuation that will be given to a variable.

Let (φ, X) be an instance of Paired SAT where $X = \{(x_1, y_1), \ldots, (x_n, y_n)\}$, and $\varphi = \bigwedge_{1 \leq j \leq m} C_j$ is a 3-CNF on the variables of X. We build a hypergraph $H = (V, E)$ as follows.

Let us define the set V of $8n$ vertices. Let $V = \bigcup_{1 \leq i \leq n} S_i \cup F_i$, with for $1 \leq i \leq n$, $S_i = \{s_i^0, s_i^T, s_i^F, s_i^1\}$ (gadget which encodes Satisfier's choice for the variable x_i) and $F_i = \{f_i^0, f_i^T, f_i^{T'}, f_i^F\}$ (gadget which encodes Falsifier's choice for the variable y_i).

Now we focus on the construction of the edges.

- The block-edges $B = \bigcup_{1 \leq i \leq n} B_i$, which make each S_i and each F_i a block:

$$B_i = \{H \subseteq S_i \mid |H| = 3\} \cup \{H \subseteq F_i \mid |H| = 3\}.$$

- The pair-edges $P = \bigcup_{1 \leq i \leq n} P_i$ (see Figure 2):

$$P_i = \left\{ \{s_i^0, s_i^T, f_i^0, f_i^T\}, \{s_i^0, f_i^F, f_i^T, s_i^F\}, \{s_i^0, f_i^F, s_i^T, f_i^{T'}\}, \{s_i^0, s_i^F, f_i^0, f_i^{T'}\} \right\}.$$

- The clause-edges. Each clause $C_j \in \varphi$ is a set of three literals $\{\ell_j^1, \ell_j^2, \ell_j^3\}$. We define first, for $1 \leq j \leq m$ and $k \in \{1, 2, 3\}$, the set H_j^k which encodes the property that the literal ℓ_j^k is instantiated to $\bot$.

$$H_j^k = \begin{cases} \{\{s_i^0, s_i^T\}\} & \text{if } \ell_j^k = x_i \\ \{\{s_i^0, s_i^F\}\} & \text{if } \ell_j^k = \neg x_i \\ \{\{f_i^0, f_i^T\}, \{f_i^0, f_i^{T'}\}\} & \text{if } \ell_j^k = y_i \\ \{\{f_i^F\}\} & \text{if } \ell_j^k = \neg y_i. \end{cases}$$

  We define now the set of edges:

$$C = \bigcup_{C_j \in \varphi} H_j.$$

  with $H_j = \left\{h_1 \cup h_2 \cup h_3 \mid \forall k \in \{1, 2, 3\},\ h_k \in H_j^k\right\}$
  For example, if $C_j = x_1 \vee y_1 \vee \neg y_2$, we have $H_j^1 = \{\{s_1^0, s_1^T\}\}$, $H_j^2 = \{\{f_1^0, f_1^T\}, \{f_1^0, f_1^{T'}\}\}$ and $H_j^3 = \{\{f_2^F\}\}$. Finally, we have two edges to encode C_j: $H_j = \{(s_1^0, s_1^T, f_1^0, f_1^T, f_2^F), (s_1^0, s_1^T, f_1^0, f_1^{T'}, f_2^F)\}$

The gadget for the pair (x_i, y_i) is depicted in Figure 2.
Intuitively, these edges are constructed in such a way that:

- The block-edges B force Waiter to always propose two vertices in the same set.
- The pair-edges P force Waiter to give to Client the choice of the value of y_i after she has made her choice for x_i.
- The clause-edges C which represent the clauses of φ and make the equivalence between a win of Waiter and a valuation that satisfies φ.

Finally, the reduction associates to the instance (φ, X) of Paired SAT the hypergraph $H = (V, E)$ with $E = B \cup P \cup C$ of Client-Waiter. This reduction is polynomial as B contains $8n$ edges, P contains $4n$ edges, and C contains at most $8m$ edges (where m is the number of clause of φ).

We define an underlying assignment of the variables, corresponding to the moves on the hypergraph as follows:

- If Client claims s_i^T, $x_i = \bot$

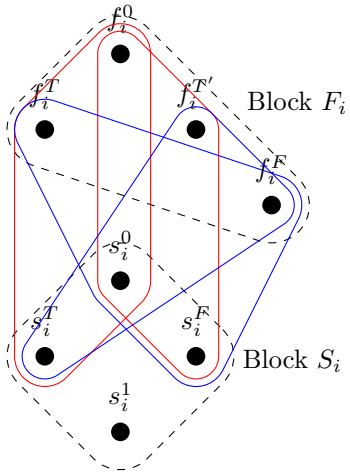

Figure 2: Gadget for the vertices in B_i. A dashed set represents a block, i.e. all hyperedges of size three are present in it.

- If Client claims s_i^F, $x_i = \top$
- If Client claims f_i^0 and one of f_i^T, $f_i^{T'}$, $y_i = \bot$
- If Client claims f_i^F, $y_i = \top$

We prove in next sections (Lemmas 21 and 21) that Waiter has a winning strategy on H if and only if Satisfier has a winning strategy on φ. Then, we can obtain the proof of Proposition 13.

Proof of Proposition 13. First, according to Lemma 14, Client-Waiter games are in PSPACE. We prove the hardness by a reduction from Paired SAT.

Let (φ, X) be an instance of Paired SAT. Consider the hypergraph H obtained from the reduction provided in Subsection 3.2.2. It has $O(|X|)$ vertices and $O(|X| + |\varphi|)$ edges, which is polynomial. According to Lemma 20 and Lemma 21, Satisfier wins in (φ, X) if and only if Waiter wins in H. Therefore, determining the winner of a Client-Waiter game is PSPACE-complete.

Moreover, as any edge of H has size at most 6, the problem is even PSPACE-complete restricted to hypergraphs of rank 6. □

3.2.3 Waiter's winning strategy

In this section, we prove that if Satisfier has a winning strategy in φ for the Paired SAT-game, then Waiter has a winning strategy in H for the Client-Waiter game.

Lemma 20. *If Satisfier has a winning strategy in φ, then Waiter has a winning strategy in H.*

Proof. Let $\mathcal{S}$ be a winning strategy for Satisfier, consider a strategy for Waiter as follows. If $\mathcal{S}$ selects an integer $1 \leq i \leq n$, and puts x_i to $\top$ (resp. $\bot$), Waiter plays in the block B_i and selects the pair (s_i^0, s_i^T) (resp. (s_i^0, s_i^F)). Then, she plays the pair corresponding to the two other vertices in the block S_i. To determine the value of y_i, she plays (f_i^F, f_i^T) (resp. $(f_i^F, f_i^{T'})$) and finally, the remaining pair of the block F_i. If Client chooses f_i^F, she considers that $y_i = \top$, otherwise, she considers that $y_i = \bot$ in $\mathcal{S}$.

As this strategy always propose vertices in blocks, Client cannot win with the edges in B:

- In the case Waiter offers (s_i^0, s_i^T), if Client does not choose s_i^0, as it is in all the edges of P_i, Waiter can not lose on P_i. Otherwise, Waiter claims s_i^T, and proposes the pairs (f_i^F, f_i^T) and $(f_i^{T'}, f_i^0)$ which cover all the remaining edges of P_i.

- In the other case Waiter offers (s_i^0, s_i^F), again if Client chooses s_i^0, Waiter cannot lose on P_i. Otherwise, Waiter claims s_i^F, and proposes the pairs $(f_i^F, f_i^{T'})$ and (f_i^T, f_i^0) which covers all the remaining edges of P_i.

We now consider the clause-edges:

Let C_j be a clause of φ. It is sufficient to prove that there exists $1 \leq k \leq 3$ such that Waiter claims a vertex in H_j^k. As $\mathcal{S}$ is a winning strategy for Satisfier in φ, there exists, at the end of the game, an index $1 \leq k \leq 3$ such that the assignment of the literal ℓ_j^k satisfies C_j.

- If $\ell_j^k = x_i$, by construction of the strategy, Waiter offers the pair (s_i^0, s_i^T), therefore she claims a vertex in H_j^k.

- If $\ell_j^k = \neg x_i$, by construction of the strategy, Waiter offers the pair (s_i^0, s_i^F), therefore she claims a vertex in H_j^k.

- If $\ell_j^k = y_i$, by construction of the strategy, Waiter has considered that y_i was put to $\top$ according to the choices of Client, which corresponds to the case where Client has chosen f_i^F. Therefore, she has claimed two of the three vertices $\{f_i^0, f_i^T, f_i^{T'}\}$, and so has a vertex in each set of H_j^k.

- If $\ell_j^k = \neg y_i$, by construction of the strategy, Waiter has considered that y_i was put to $\bot$ according to the choice of Client, which corresponds to the case where Client has not chosen f_i^F, therefore Waiter has claimed it, and thus has a vertex in H_j^k.

Finally, Client can not fill up an edge in H_j for any $1 \leq j \leq m$. So the described startegy is winning for Waiter. □

3.2.4 Client's winning strategy

We prove now the other direction.

Lemma 21. *If Falsifier has a winning strategy in φ, then Client has a winning strategy in H.*

Proof. Suppose now that Falsifier has a winning strategy $\mathcal{S}$ in φ. We provide a strategy for Client.

First, notice that, as H is a blocks-hypergraph, we can suppose that Waiter always offer vertices in blocks, according to Corollary 19. Moreover, since each block has size 4, once the first pair of a block is offered, Client knows that the second pair will be proposed at some point and can already decide her move on this second pair. Therefore it is sufficient to have a winning strategy when Waiter always offers simultaneously the two pairs of a same block.

Let $1 \leq i \leq n$ be an integer, and suppose that Waiter offers two disjoint pairs of a block S_i or F_i.

- If the pairs are in S_i, Client can ensure to claim s_i^0 and one of s_i^T and s_i^F. If he has claimed s_i^F, he considers that Satisfier has instantiated x_i to $\top$ and if he has claimed s_i^T, he considers x_i has been instantiated to $\bot$.
  - Assume Client claims s_i^0 and s_i^T. Since $\{s_i^0, s_i^T, f_i^T, f_i^0\}$ is a edge of P_i, by Lemma 12 Waiter will offer the pairs $\{f_i^T, f_i^0\}$ and $\{f_i^{T'}, f_i^F\}$. Client can follow $\mathcal{S}$ by picking f_i^0 and $f_i^{T'}$ if $\mathcal{S}$ instantiates y_i to $\bot$, and by picking f_i^T and f_i^F otherwise.
  - If Client claims s_i^0 and s_i^F, the result is similar by switching the roles of f_i^T and $f_i^{T'}$.
- If the pairs are in F_i, as F_i is a block of four vertices, Waiter has only three possible ways to pair them.
  - If the pairs are $\{f_i^0, f_i^T\}$ and $\{f_i^F, f_i^{T'}\}$, Client will claim either the two vertices (f_i^T, f_i^F), or the two vertices $(f_i^0, f_i^{T'})$. In both cases, Waiter will have to offer the pairs $\{s_i^0, s_i^F\}$ and $\{s_i^T, s_i^1\}$, and Client will claim s_i^0 and s_i^F. This case is already described above, and so the choice of the claiming pair (f_i^T, f_i^F) or $(f_i^0, f_i^{T'})$ can be done as previously in following $\mathcal{S}$.
  - If the pairs are $\{f_i^T, f_i^{T'}\}$ and $\{f_i^0, f_i^F\}$, Client can still claim either (f_i^T, f_i^F), or $(f_i^0, f_i^{T'})$ and the case is identical.
  - Otherwise the pairs are $\{f_i^T, f_i^F\}$ and $\{f_i^0, f_i^{T'}\}$. Client will claim either $(f_i^{T'}, f_i^F)$, or (f_i^0, f_i^T). In both cases, Waiter will have to offer the pairs $\{s_i^0, s_i^T\}$ and $\{s_i^F, s_i^1\}$, and Client will claim s_i^0 and s_i^T. This is again a case described above.

Following this strategy until the end, the underlying valuation of the claimed vertices is the one obtained by $\mathcal{S}$ in φ. By hypothesis, there exists a clause $C_j \in \varphi$ in which all literals are set to $\bot$. Thus Client wins by filling up the edge H_j. □

3.3 Reduction to 6-uniform hypergraphs

As it is done by Rahman and Watson for Maker-Breaker games [RW21], or by Gledel and Oijid for Avoider-Enforcer games [GO23], Proposition 13 can be strengthen by transforming the hypergraph of rank 6 into a k-uniform one (with $k \geq 6$). We achieved this with the following lemma:

Lemma 22. *Let $H = (V, E)$ be a hypergraph of rank k. Let $m = \min_{e \in E}(|e|)$. If $m < k$, there exists a hypergraph $H' = (V', E')$ of rank k where $\min_{e \in E}(|e|) = m+1$, having $|E'| \leq |E| + \binom{2k-2}{k}$ and $|V'| \leq |V| + 2(k-1)$ such that Client has a winning strategy in the Client-Waiter game on H if and only if he has one in H'.*

Proof. Let $H = (V, E)$ be a hypergraph of rank k. We construct the hypergraph $H' = (V', E')$. Let $A = \{a_1, \ldots, a_{2(k-1)}\}$ be $2(k-1)$ new vertices, and set $V' = V \cup A$. We make A a block, i.e. we define the set $U = \{B \subset A \mid |B| = k\}$ of edges. Then, we introduce $L = \{e \cup \{a_1\} \mid e \in E \text{ and } |e| = m\}$. Finally, we define our edges as follows:

$$E' = \{e \in E \mid |e| \geq m+1\} \cup L \cup U.$$

The construction is depicted in Figure 3.

Client wins in H if and only if he wins in H'. Indeed, suppose that Client wins in H, as A is a block, Waiter will have to offer the vertices of A two by two. Therefore, Client can claim a_1 when it will be proposed and apply the strategy for H on vertices of V. Then, if Client fills up an edge $e \in E$ with $|e| = m$, he also fills up $e \cup \{a_1\}$ in H', and if she fills up $e \in E$ with $|e| \geq m+1$, she also fills up e in H'. Reciprocally, if Waiter has a winning strategy in H, she can offer the same pairs in H' and the vertices of A two by two. Then, for any edge $e \in E'$, if $e \in E$ or $e \setminus \{a_1\} \in E$, Waiter has claimed a vertex of it by her winning strategy in H. Otherwise, $e \in U$, and she claims one vertex of it since she claims $k-1$ vertices of A. □

Hence, we obtain Theorem 2 by combining Proposition 13 and Lemma 22.

Proof. The hypergraph obtained in the proof of Proposition 13 has rank 6. Therefore, by applying $k-1$ times Lemma 22, with $m = 1, \ldots, k-1$, we obtain a k-uniform hypergraph having at most $2(k-1)^2$ more vertices and $(k-1)\binom{2k-2}{k}$ more edges. Thus, when k is fixed, this construction is still polynomial, and the hypergraph obtained is k-uniform. □

4 Rank 3 Client-Waiter equivalent to the problem of detecting a tadpole

Similarly to Maker-Breaker, the PSPACE-hardness of Client-Waiter games restricted to 6-uniform hypergraphs leads us to consider smaller rank hypergraphs. In this section, we first provide a linear time algorithm to compute the winner in rank 2 hypergraphs, then we prove that rank 3 hypergraphs are in Δ_2^P.

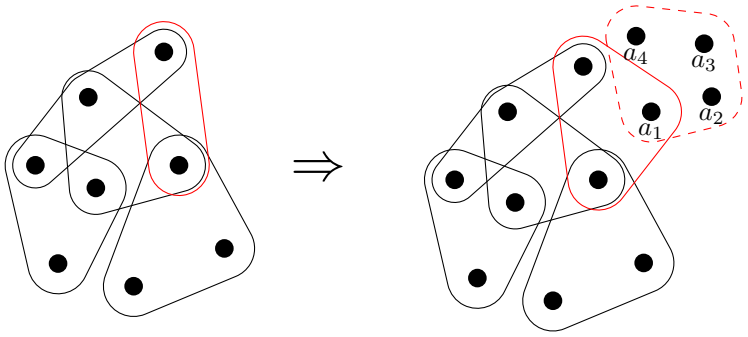

Figure 3: The construction of Lemma 22 with $k = 3$ and $m = 2$. The dashed set is a block and contains the four hyperedges of size 3. The resulting hypergraph is 3-uniform

4.1 Rank 2 games

Let us start with the easier case of rank 2 hypergraphs. We prove that Client-Waiter games are tractable restricted to them.

Proposition 23. *Let $H = (V, E)$ be a rank 2 hypergraph. The winner of a Client-Waiter game played on H can be computed in linear time.*

Proof. We prove that Client wins on H if and only if there exists an edge of size 1, or if two edges intersect. The result directly follows since these properties can be checked in linear time.

- If H has an edge $\{x\}$ of size 1, when Waiter proposes the vertex x, Client can claim it and win. As the last vertex goes to Client if the number of vertices if odd, Client can ensure to get x and win.
- If there exist two edges $\{a, b\}$ and $\{a, c\}$ which intersect, Client applies the following strategy:
  - Whenever Waiter offers two of these vertices, Client claims one of them, and he takes a if it is available.
  - If Waiter offers one of these vertices with any other, Client claims the one in $\{a, b, c\}$.
  - Otherwise, Client takes any vertex.

  Following this strategy, Client claims a and at least one vertex from $\{b, c\}$. Therefore, he wins. Once again, even if the number of vertices is odd, this strategy can be applied.

Reciprocally, if all edges have size 2 and do not intersect (H is in fact a graph, and even a matching), by always selecting a pair (a, b) such that (a, b) is an edge, Waiter gets one vertex from each edge and wins. □

4.2 Rank 3 games

We now focus on Client-Waiter games restricted to rank 3 hypergraphs. We show that testing if a position is winning for Client in a hypergraph H of rank 3 reduces to the problem of searching two structures (the a-snakes and the ab-tadpoles) in H. We start by defining them.

Definition 24. *Let $H = (V, E)$ be a hypergraph and $a, b \in V$.*

*A sequence of edges of H, $\mathcal{P} = (e_1, \dots, e_t)$, is an ab-*path *if $a \in e_1$, $b \in e_t$, and for $1 \leq i \leq t-1$ the edges e_i and e_{i+1} intersect. The number of edges t is called the* length *of $\mathcal{P}$. An a-*cycle *is an aa-path of length at least two. An ab-path is said* linear *if the size of the intersection of two consecutive edges is always one. An a-cycle of length at least 3 is* linear *if the aa-path is linear and if the intersection of the end edges is exactly $\{a\}$. We continue to call* linear *an a-cycle (e_1, e_2) of length 2 if $|e_1 \cap e_2| = 2$. An ab-path is said* simple *if a appears only in e_1, b appears only in e_t, and if whenever e_i and e_j intersect, then $|i - j| \leq 1$. Similarly, an a-cycle is simple if whenever e_i and e_j intersect, then $|i - j| \leq 1$ or $\{i, j\} = \{1, t\}$.*

An ab-path is also called an a-path or a path. A cycle is a-cycle for some a in V.

Definition 25. *Let a be a vertex of H. An a-*snake *is an a-path $(e_1, \dots, e_t)$ with $t \geq 1$ such that for all $1 \leq i \leq t-1$, e_i has exactly size 3, and e_t has size at most 2.*

Definition 26. *Let $H = (V, E)$ be a hypergraph and $a, b \in V$. If $a \neq b$, an ab-*tadpole *is a sequence of edges $T = (e_1, \dots, e_s, f_1, \dots, f_t)$ where:*

- *a belongs to e_1 and no other edge;*
- *b belongs to e_s, f_1, f_t and no other edge;*
- *$(e_1, \dots, e_s)$ is a 3-uniform simple linear ab-path $\mathcal{P}_T$;*
- *$(f_1, \dots, f_t)$ is a 3-uniform simple linear b-cycle $\mathcal{C}_T$;*
- *b is the only vertex which appears both in $\mathcal{P}_T$ and $\mathcal{C}_T$.*

If $a = b$, an ab-tadpole is just a 3-uniform simple linear a-cycle. When T is an ab-tadpole, we may simply say T is an a-tadpole, or even just a tadpole.

We consider the problem Tadpole: Given a vertex u in a 3-uniform hypergraph H, decide if there is a u-tadpole in H.

We denote by $o(H)$ the outcome of an optimal Client-Waiter game on the hypergraph H. We will write $o(H) = \mathcal{W}$ when Waiter has a winning strategy on H, and $o(H) = \mathcal{C}$ otherwise.

Let u be a vertex of H, we consider the following family u-$\mathcal{F}_H$ of subhypergraphs of H:

$$T \in u\text{-}\mathcal{F}_H \iff \begin{cases} T \text{ is a } u\text{-snake} \\ \text{or } T \text{ is a } u\text{-tadpole.} \end{cases}$$

Notice that 3-uniform simple linear u-cycles are particular cases of u-tadpoles, so such subhypergraphs are also in u-$\mathcal{F}_H$. When the hypergraph H is known, we will simply write u-$\mathcal{F}$.

We notice that Waiter has a winning strategy in $H = (V, E)$ which starts by offering a pair $\{u, v\}$ if and only if Waiter has a winning strategy in both trace hypergraphs $H_1 = T_{V\setminus\{u,v\}}(H_{|V\setminus\{u\}})$ and $H_2 = T_{V\setminus\{u,v\}}(H_{|V\setminus\{v\}})$. So to simplify notations, we will write H^{+v} for the trace $T_{V\setminus\{v\}}(H)$ and H^{-v} for the induced subhypergraph $H_{|V\setminus\{v\}}$. In particular, we can just write $H_1 = H^{+v-u}$ and $H_2 = H^{+u-v}$.

Lemma 27. *Let u be a vertex of H hypergraph of rank 3 and let $T \in u$-$\mathcal{F}$. Then $o(T^{+u}) = \mathcal{C}$.*

Proof. We do the proof by induction on the size (number of edges) of T. By definition, each subhypergraph of u-$\mathcal{F}$ contains at least one edge.

Assume that T contains exactly one edge. It means that T is a u-snake of length 1, i.e., one edge of length at most two. Since T^{+u} is an edge of size at most one, it is a winning position for Client.

Assume otherwise that T has at least two edges. Three cases can happen.

- T is a u-snake of length at least two. So T is a sequence of the form $(\{u, v, w\}, \{v, x, y\}, e_3, \ldots, e_\ell)$ where e_ℓ has size at most 2 and the other edges have size 3. Assume $o(T^{+u}) = \mathcal{W}$, then by Lemma 12, there is a strategy for Waiter which starts with the pair $\{v, w\}$. However, if Client selects v, then $(\{v, x, y\}, e_3, \ldots, e_\ell)$ is a v-snake of size smaller than T, and so the position is winning for Client by induction hypothesis. Hence $o(T) = \mathcal{C}$.

- T is a 3-uniform simple linear u-cycle. So T is of the form $(\{u, v, w\}, \{v, x, y\}, e_3, \ldots, e_\ell)$ where $(\{v, x, y\}, e_3, \ldots, e_\ell)$ is a 3-uniform simple linear path of positive length and e_ℓ contains u. The conclusion is similar to the previous case. Indeed, if in T^{+u} Waiter offers the pair $\{v, w\}$ and Client picks v, we obtain a v-snake of size smaller than T.

- T is a u-tadpole which is not a cycle. So T is a sequence of the form $(\{u, v, w\}, \{v, x, y\}, e_3, \ldots, e_\ell, e'_1, \ldots, e'_t)$ where e_ℓ contains a vertex b such that $(\{u, v, w\}, \{v, x, y\}, e_3, \ldots, e_\ell)$ is a 3-uniform simple linear ub-path, and $(e'_1, \ldots, e'_t)$ is a 3-uniform simple linear b-cycle. Assume $o(T) = \mathcal{W}$, then by Lemma 12, there is a strategy for Waiter which starts with the pair $\{v, w\}$. However, if Client selects v, then $(\{v, x, y\}, e_3, \ldots, e_\ell, e'_1, \ldots, e'_t)$ is a v-tadpole of size smaller than T, and so the position is winning for Client by induction hypothesis. Hence $o(T) = \mathcal{C}$. □

We consider the set of vertices which are reachable from u by a simple linear path.

Definition 28. *Let H be a hypergraph and u be a vertex of H. Let us denote by $\mathrm{CL}_H(u)$ the induced subhypergraph of H^{+u} on the set of the vertices which are reachable from u by a simple linear path.*

Notice that it is possible that v is reachable from u by a linear path and by a simple path with no simple linear path between u and v (see for example Figure 4).

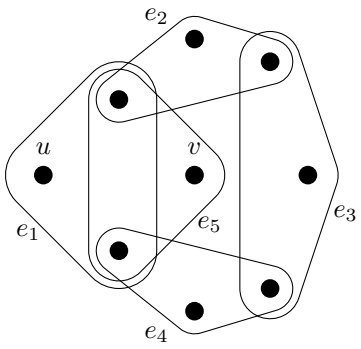

Figure 4: The vertices u and v are connected by the linear path $e_1e_2e_3e_4e_5$ and by the simple path e_1e_5 but no simple linear path connects them.

Definition 29. *Two vertices v and w are called* siblings *with respect to u if they are reachable from u by a simple linear path, but any such path to one of these vertices contain the other one. More formally, $v \sim_u w$ if and only if*

$$v \in \mathrm{CL}_H(u) \setminus \mathrm{CL}_{H^{-w}}(u) \text{ and } w \in \mathrm{CL}_H(u) \setminus \mathrm{CL}_{H^{-v}}(u).$$

Notice that siblings happen only by pairs in rank 3 hypergraphs.

Proposition 30. *Let H be a rank 3 hypergraph. Assume that both pairs (v, w_1) and (v, w_2) are siblings with respect to u. Then $w_1 = w_2$.*

Proof. Let $\mathcal{P}$ be a smallest simple linear path from u to v. Since, v is not in $\mathrm{CL}_{H^{-w_1}}(u) \cup \mathrm{CL}_{H^{-w_2}}(u)$, it means that w_1 and w_2 have to appear in $\mathcal{P}$. If $w_1 \neq w_2$, then either w_1 or w_2 appears before the last edge in $\mathcal{P}$. It contradicts the fact w_1 and w_2 do not belong to $\mathrm{CL}_{H^{-v}}(u)$. □

Definition 31. *Let u be a vertex of H hypergraph of rank 3. We extend the hypergraph $\mathrm{CL}_H(u)$ by adding, for each pair of siblings $v \sim_u w$, a new edge $\{v, w\}$. We call this hypergraph $\mathrm{CCL}_H(u)$ the* completed *of $\mathrm{CL}_H(u)$.*

Lemma 32. *Let u be a vertex of H hypergraph of rank 3. Then, u-$\mathcal{F}$ is non empty if and only if $o(\mathrm{CCL}_H(u)) = \mathcal{C}$.*

Proof. If $T \in u$-$\mathcal{F}$, then T^{+u} is a subhypergraph of $\mathrm{CL}_H(u) \subseteq \mathrm{CCL}_H(u)$. Then Lemmas 11 and 27 imply that $o(\mathrm{CCL}_H(u)) = \mathcal{C}$.

We prove that if u-$\mathcal{F}$ is empty then $o(\mathrm{CCL}_H(u)) = \mathcal{W}$ by induction on the number of vertices of $\mathrm{CL}_H(u)$. If $\mathrm{CL}_H(u)) = \emptyset$, then necessarily $o(\mathrm{CCL}_H(u)) = \mathcal{W}$. Otherwise, the number of vertices of $\mathrm{CL}_H(u)$ is a positive integer. It means that $\mathrm{CCL}_H(u)$ contains an edge e, which is the trace of an edge $\{u\} \cup e$ of H. If e has size at most 1, then $\{u\} \cup e$ is in u-$\mathcal{F}$. So $e = \{v, w\}$.

We have a first fact.

Fact 33. *Any vertex of $\mathrm{CL}_{H^{+u-w}}(v)$ is already present in $\mathrm{CL}_H(u)$.*

Proof. Let x be a vertex of $\mathrm{CL}_{H^{+u-w}}(v)$. It means there is an almost simple linear path $\mathcal{P}$ in H^{-w} from v to x where only the vertex u can be repeated. If $\mathcal{P}$ contains u, then, starting by its last occurrence, it contains a simple linear path from u to x. Otherwise, the concatenation of $(\{u, v, w\})$ with $\mathcal{P}$ is simple and linear. In both cases, we have that x is a vertex of $\mathrm{CL}_H(u)$. □

Consequently, every vertex of $\mathrm{CCL}_H(u)$ is neighbour of u or appears in some $\mathrm{CCL}_{H^{+u-w}}(v)$ for such a couple of vertices (v, w). Given such a (v, w), we show, that there is a winning strategy on $\mathrm{CCL}_{H^{+u-w}}(v)$ for Waiter, such that if this strategy is played on $\mathrm{CCL}_H(u)^{+v-w}$, the resulting hypergraph G is a subhypergraph of $\mathrm{CCL}_H(u)^{+v-w}$ and so of $\mathrm{CCL}_H(u)$. The result of the lemma follows by repeating the argument for each such couple (v, w) ($G \cap \mathrm{CCL}_H^{+u-w'}(v')$ is a subhypergraph of $\mathrm{CCL}_H^{+u-w'}(v')$ and so the strategy can be done on G by Lemma 11).

So let us fix such couple (v, w). Since v and w are vertices of $\mathrm{CL}_H(u) \setminus \mathrm{CL}_{H^{+u-w}}(v)$, the number of vertices of $\mathrm{CL}_{H^{+u-w}}(v)$ is strictly smaller than the one of $\mathrm{CL}_H(u)$.

Fact 34. *If u-$\mathcal{F}_H$ is empty, then v-$\mathcal{F}_{H^{+u-w}}$ is also empty.*

Proof. Let us show the contrapositive. Let $T \in v\text{-}\mathcal{F}_{H^{+u-w}}$.

- Assume first that T is reduced to one edge e. If e is an edge of H, then $(\{u, w, v\}, e)$ is a u-snake of H. Otherwise e is the trace of an edge e' of H of the form $e' = \{v, x, u\}$ where $x \neq w$ (e' can not be of length 2 since it would be contained in $\{u, v, w\}$ but H is a clutter). In particular, $(\{u, w, v\}, \{v, x, u\})$ is a 3-uniform simple linear u-cycle of length 2 and so a u-tadpole in u-$\mathcal{F}$.

- Assume then that T is a subhypergraph of H. In particular it is a v-tadpole in H^{-w} which does not contain u. Adding the edge $\{u, v, w\}$ at the beginning of T gives a u-tadpole in u-$\mathcal{F}_H$.

- Otherwise, T is a v-snake $(e_1, \ldots, e_p)$ such that e_p is the trace of an edge $e' = e_p \cup \{u\}$ in H. So adding $\{u, v, w\}$ in front of T and replacing e_p by e' gives a u-tadpole (in fact a u-cycle) in u-$\mathcal{F}$. □

By the second fact v-$\mathcal{F}_{H^{+u-w}}$ is empty and so by induction hypothesis Waiter has a winning strategy $\mathcal{S}$ in $\mathrm{CCL}_{H^{+u-w}}(v)$. So the first fact ensures that Waiter can simulate the strategy $\mathcal{S}$ on $H' = \mathrm{CCL}_H(u)^{+v-w} = (V', E')$. After playing $\mathcal{S}$, we obtain a new hypergraph G (depending on Client's choices). Let $V_{\mathcal{W}}$ and $V_{\mathcal{C}}$ be the sets of vertices claimed respectively by Waiter and Client during this phase. So G is the trace on $V' \setminus (V_{\mathcal{W}} \cup V_{\mathcal{C}})$ of the hypergraph $H'_{|V' \setminus V_{\mathcal{W}}}$. We show that G is a subhypergraph of H' (i.e., there is no edge in $H'_{|V' \setminus V_{\mathcal{W}}}$ intersecting both $V_{\mathcal{C}}$ and $V' \setminus (V_{\mathcal{W}} \cup V_{\mathcal{C}})$).

Suppose there exists e an edge of $H'_{|V' \setminus V_{\mathcal{W}}}$ such that $x \in e \cap V_{\mathcal{C}}$ and $y \in e \setminus V_{\mathcal{C}}$.

- If $e = \{x, y\}$ or if $e = \{x, y, z\}$ with z not in $V_{\mathcal{C}}$, then any simple linear path from v to x can be extended to a simple linear path from v to y by adding the edge e, which is impossible.

- Otherwise, $e = \{x, y, z\}$ with $z \in V_{\mathcal{C}}$. It implies that x and z are siblings with respect to v in H^{+u-w} (otherwise y would also be reachable from v). Then, there is an edge $\{x, z\}$ in $\mathrm{CCL}_{H^{+u-w}}(v)$. But having x and z in $V_{\mathcal{C}}$ contradicts the fact that $\mathcal{S}$ is winning in $\mathrm{CCL}_{H^{+u-w}}(v)$. □

In particular, Proposition 6 is a direct consequence of Lemma 32.

Corollary 35. *Let u, v be vertices of H hypergraph of rank 3 such that u-$\mathcal{F}_{H^{-v}}$ and v-$\mathcal{F}_{H^{-u}}$ are empty. If $o(H) = \mathcal{W}$, then Waiter has a winning strategy on H which starts with the pair $\{u, v\}$.*

Proof. Indeed, Waiter starts offering the pair $\{u, v\}$. Assume that Client picks u (the other case is symmetric). By hypothesis, u-$\mathcal{F}_{H^{-v}}$ is empty. By Lemma 32 Waiter has a winning strategy $\mathcal{S}$ in $\mathrm{CCL}_{H^{-v}}(u)$. Waiter can play in H^{+u-v} according to $\mathcal{S}$.

Claim 36. *At the end of the strategy $\mathcal{S}$, the obtained hypergraph is a subhypergraph of H.*

Proof of the claim. Assume that e is an edge of H^{+u-v} where none of its vertices are claimed by Waiter and where $x \in e$ is a vertex claimed by Client. Assume furthermore that there is $y \in e$ which is unclaimed, i.e., it does not belong to $\mathrm{CL}_{H^{-v}}(u)$. Two cases can happen.

1. Assume first that $e = \{x, y\}$ or $e = \{x, y, z\}$ where z does not belong to $\mathrm{CL}_{H^{-v}}(u)$ too. Since, $x \in \mathrm{CL}_{H^{-v}}(u)$, there exists a simple linear path $\mathcal{P}$ from u to x. Adding e at the end of $\mathcal{P}$ gives a simple linear path from u to y which is impossible.

2. So x and z belong to $\mathrm{CL}_{H^{-v}}(u)$. If one vertex of $\{x, z\}$ is reachable from u in H^{-v} by a linear simple path which does not contain the other vertex, it would again contradicts the fact that $y \notin \mathrm{CL}_{H^{-v}}(u)$. So x and z are siblings with respect to u. Again, this is impossible since $\{x, z\}$ would be a winning edge claimed by Client. ◇

The new hypergraph is a subhypergraph of H which is winning for Waiter by hypothesis. By Lemma 11, the new hypergraph is still winning for Waiter. □

Theorem 5 follows from Corollary 35. Indeed, it suffices to test if there is a couple of vertices (u, v) such that u-$\mathcal{F}_{H^{-v}}$ and v-$\mathcal{F}_{H^{-u}}$ are empty. If it is not the case, it is a win for Client by Lemma 32. If such couple is found, Corollary 35 ensures, we lose nothing by starting with this couple, and we can redo the test for the smaller hypergraph.

5 Waiter-Client games are FPT on k-uniform hypergraphs

In Oijid's thesis [Oij24], it is proved that if Waiter can win a rank 2 Waiter-Client game, she has a winning strategy in at most three moves. This leads to the following result:

Proposition 37 (Theorem 1.70 from [Oij24])**.** *Let $H = (V, E)$ be a rank 2 hypergraph. The winner of a Waiter-Client game played on H can be computed in polynomial time.*

We here extend this result, by proving that for any $k \geq 1$, the winner of a Waiter-Client game on a rank k hypergraph can be computed in FPT time parameterized by k. This result is a far-reaching generalization of the following easy fact: if a rank k hypergraph has 2^k disjoint edges, it is Waiter's win.

Let $H = (V, E)$ be a k-uniform hypergraph. We call *ℓ-sunflower* a set S of $\ell \geq 1$ edges of H pairwise intersecting on a fixed set C called *center* of S. When the center is empty, the sunflower simply consists of disjoint edges. We call *petal* of S every set $s \setminus C$ where $s \in S$. We authorize multisets in the definition, in particular, any edge e of H can be considered as an ℓ-sunflower for every $\ell \geq 1$ (the emptyset being a petal). Such a sunflower with empty petals is called *trivial*. The celebrated Sunflower Lemma from Erdős and Rado [ER60] asserts that every k-uniform hypergraph with at least $k!(\ell - 1)^k$ distinct edges contains a non trivial ℓ-sunflower. We first show that the number of inclusion-wise minimal centers is bounded in terms of k and ℓ. This is the first step of the FPT algorithm for the Waiter-Client game.

We say that an ℓ-sunflower S of H is *minimal* (with respect to inclusion) if no ℓ-sunflower of H has a center strictly included in the center of S. Let Y be a subset of vertices of H, we say that S is *outside* Y if all petals of S are disjoint from Y. In particular, every edge $e \in E$ forms a trivial ℓ-sunflower outside Y for every subset $Y \subseteq V$.

Lemma 38. *There exists a function mc_k for which every k-uniform hypergraph H has at most $mc_k(\ell)$ distinct centers of minimal ℓ-sunflower. Moreover, there exists a function omc_k such that whenever Y is a subset of vertices of size y, H has at most $omc_k(\ell, y)$ distinct centers of minimal ℓ-sunflower outside Y.*

Proof. We focus on the existence of mc_k, and postpone the outside case omc_k to the end of the proof.

We just have to show that if $C_1, \ldots, C_t$ are distinct centers of size c of a family of minimal ℓ-sunflowers $S_1, \ldots, S_t$, then t is bounded in terms of k and ℓ. Indeed, this implies the first part of the lemma since $mc_k(\ell)$ will be at most k times this bound. Let us fix $t' = 2k(\ell - 1) + 1$. We denote by P_i^j the j^{th} petal of S_i. If t is large enough, we can extract a subfamily of t' sunflowers from $S_1, \ldots, S_t$ which, free to reorder, is assumed to be $S_1, \ldots, S_{t'}$ with the following additional properties:

- all (distinct) centers $C_1, \ldots, C_{t'}$ form a sunflower with center X.

- for all $1 \leq j \leq \ell$, all petals $P_1^j, \ldots, P_{t'}^j$ form a sunflower with center Q_j.

Indeed, we just have for this to iterate $\ell + 1$ extractions using the Sunflower Lemma, one for the centers, and ℓ for the petals. To conclude, we now extract an ℓ-sunflower S centered at X in H. This is indeed a contradiction to the minimality of centers C_i. A simple greedy argument suffices for this.

Let us say that $B_i := C_i \setminus X$ and $D_i^j := P_i^j \setminus Q_j$. With this notation, the j^{th} edge of the sunflower S_i is $X \cup B_i \cup Q_j \cup D_i^j$.

Our first edge e_1 of S is the first edge of S_1, that is $C_1 \cup P_1^1$. Observe that e_1 intersects at most k of the (disjoint) subsets B_i and also at most k of the subsets D_i^2. Since $t' > 2k$, there is an index a such that e_1 is disjoint from $B_a \cup D_a^2$ (notice that a is automatically distinct from 1 since e_1 is disjoint of B_a). We pick the edge e_2 as the second edge of S_a, that is $X \cup B_a \cup Q_2 \cup D_a^2$. Since all edges of S_1 intersect on C_1, the edge e_1 is disjoint from Q_2. In particular $e_1 \cap e_2 = X$. By a similar argument, since $t' > 4k$, there is an index b such that $e_1 \cup e_2$ is disjoint from $B_b \cup D_b^3$. We now pick the edge e_3 as the third edge of S_b. Iterating the process leads to an ℓ-sunflower S centered on X, a contradiction.

The proof for sunflowers outside some set Y is similar, but we have to take care of the fact that the final sunflower S must have its petals outside Y. This is granted for the subsets Q_j and D_i^j, which are petals of the original ℓ-sunflowers $S_1, \ldots, S_t$ outside Y. But the (disjoint) sets B_i can intersect Y. Observe that this happens at most $|Y|$ times. Hence, choosing $t' = |Y| + 2k(\ell - 1) + 1$ leads to a contradiction. □

We now turn to the key-definition. Given some integer ℓ, we say that a set $K \subseteq V$ is an *ℓ-kernel* of H if for every edge $e \in E$, there exists $C \subseteq e \cap K$ and an ℓ-sunflower S outside K centered at C. Note that we can assume that S is minimal outside K. Observe that the union of all edges of H is an ℓ-kernel for every ℓ, and that if H has ℓ disjoint edges, then $\emptyset$ is an ℓ-kernel. We now show that there is always a bounded size kernel.

Theorem 39. *There exists a function f_k such that every k-uniform hypergraph H has an ℓ-kernel of size at most $f_k(\ell)$, for every integer ℓ.*

Proof. Let $X_0 = \emptyset$ and ℓ be some integer. We denote by C_0 the set of all centers of minimal ℓ-sunflowers, which is bounded by $mc_k(\ell)$ by Lemma 38. We then set $X_1 := \bigcup C_0$ and start to define our sequence (X_i) as follows: Once X_i is defined, we denote by C_i the set of all centers of minimal ℓ-sunflowers outside X_i and set $X_{i+1} := (\bigcup C_i) \cup X_i$. Our goal is to show that there exists a step i, bounded by a function on k and ℓ, such that $X_{i+1} = X_i$. Observe that when this step is reached, X_i is an ℓ-kernel. Indeed, let $e \in E$ be an edge. Note that e by itself is a trivial ℓ-sunflower outside X_i, hence there is a C included in e which is the center of a minimal ℓ-sunflower outside X_i. By definition of X_{i+1}, we have $C \subseteq X_{i+1} = X_i$, and therefore $C \subseteq X_i \cap e$, implying that X_i is an ℓ-kernel.

To reach our conclusion, we need to show that the size of X_i is upper bounded by some fixed function $g(i)$ (depending on k and ℓ). For this, we just have to

bound the size of X_{i+1} in terms of the size of X_i. This is easily obtained by Lemma 38, since the total number of minimal ℓ-sunflowers outside X_i is at most $omc_k(\ell, |X_i|)$.

Let us now denote, for every i, the sequence $s_i = (t_0(i), t_1(i), \ldots, t_k(i))$ where $t_c(i)$ denotes the number of distinct centers C of size c of minimal ℓ-sunflowers outside X_i. If $X_i \neq X_{i+1}$, we claim that s_{i+1} is lexicographically smaller than s_i. To see this, assume that c is minimum such that $t_c(i) \neq t_c(i+1)$ and suppose for contradiction that $t_c(i) < t_c(i+1)$. This means that there is a set C of size c which is the center of a minimal ℓ-sunflower outside X_{i+1}, but not the center of a minimal ℓ-sunflower outside X_i. In particular, C strictly contains C' which is the center of a minimal ℓ-sunflower outside X_i, but not the center of a minimal ℓ-sunflower outside X_{i+1} (by minimality of C). Since $t_{|C'|}(i) = t_{|C'|}(i+1)$, there is a set C'' of size $|C'|$ which is the center of a minimal ℓ-sunflower outside X_{i+1}, but not the center of a minimal ℓ-sunflower outside X_i. This process always find a smaller center which is minimum outside X_{i+1} but not X_i. This is a contradiction.

Note that the previous argument moreover shows that if $s_i = s_{i+1}$, then the centers of minimal ℓ-sunflowers outside X_i and outside X_{i+1} are the same, thus $X_i = X_{i+1}$.

Therefore, we just have to show that the sequence (s_i) is ultimately constant after some fixed number of steps. Let us examine for this how the sequence (s_i) evolves from i to $i+1$. At each step, the minimum index coordinate t_c which differs in s_i and s_{i+1} decreases. Meanwhile, the coordinates $t_{c'}$ where $c' > c$ can change in an arbitrary way, but no more than some fixed amount since the size of X_{i+1} is bounded by $g(i+1)$ (a crude upper bound for the increase of $t_{c'}$ would be for instance $omc_k(\ell, g(i+1))$). In all, this process terminates before some fixed number of steps depending on k and ℓ. □

Kernels are relevant for Waiter-Client games. Indeed, if K is a 2^k-kernel of a k-uniform hypergraph $H = (V, E)$ and $T_K(H)$ is the trace hypergraph on vertex set K and edge set $E_K = \{e \cap K : e \in E\}$, we have the following equivalence:

Lemma 40. *H is Waiter win if and only if $T_K(H)$ is Waiter win.*

Proof. Since $T_K(H)$ is obtained by reducing the size of some edges of H (and deleting isolated vertices), if Waiter has a winning strategy on H, it has one on $T_K(H)$.

Now assume that $T_K(H)$ is Waiter win. Waiter can play her strategy on $T_K(H)$ in order to select a set of vertices which contains some $e \cap K$ for $e \in E$. Since K is a kernel, there exists a 2^k-sunflower S outside K with center $C \subseteq e \cap K$. Now Waiter just has to play outside K to select one of the 2^k petals of S and win the game. □

We can now prove Theorem 7, asserting that Waiter-Client is FPT on k-uniform hypergraphs.

Proof. Let H be some input k-uniform hypergraph on n vertices and m edges. We first describe an algorithm running in time $O(h(k).(n+m)^2)$ which decides if H is Waiter-win.

The strategy is to compute a 2^k-kernel K as in Theorem 39 which size only depends on k. Then, by Lemma 40, we just have to (brute force) test in $O(BF(k))$ time if H_K is a Waiter win. The only point to check is how efficiently one can compute the sequence (X_i) as in the proof of Theorem 39.

As the argument is similar for getting X_{i+1} from X_i, we just describe how to compute X_1, that is the set of centers of minimal 2^k-flowers. An easy algorithm consists in (bottom up) testing all $2^k m$ subsets included in some edge of H and determine which ones are centers of 2^k-sunflowers (this single test being done in linear time, the computation of all centers can be achieved in quadratic time). Once computed, determining which centers are minimal can be done in linear time by dynamic programming. In all, each X_i can be computed in quadratic time, and the number of steps is bounded in terms of k.

To improve the complexity of the above algorithm to linear, we need to show that, while reading all edges of H, we can only keep constant size information for each possible center C to determine if it is the center of a 2^k-sunflower. For this, observe that whenever we have read $k!(k2^k-1)^k$ edges containing C, then there is a non trivial $k2^k$-sunflower S centered at C' containing C. Now we can replace all edges in S by C' since the existence of a 2^k-sunflower centered at C is left unchanged. We can then compute in linear time all the centers C, and proceed as previously. □

6 Open problems

A reasonable guess is that Waiter-Client game is NP-hard when the set of edges is explicit, i.e. the size of the input is the sum of the sizes of edges. This is our first open problem. This would highlight the fact that Fixed Parameter Tractability is probably the best one can expect, leaving open the order of magnitude of the complexity in k. Observe that the proof of Theorem 7 shows in particular that the minimum number of moves in an optimal winning strategy for Waiter is at most some value $os(k)$. Proposition 37 gives $os(2) = 3$. Alas, we have no decent bound to propose for $os(3)$, and a very optimistic analysis of the bound provided by Theorem 39 already gives an Ackermann type bound for $os(k)$. We believe that a much more reasonable upper bound can be achieved for $os(k)$.

The relationship between positional games and hypergraph colorability has driven a lot of research in Maker-Breaker. This is also the case here: Is it possible that Waiter's win in Clien-Waiter implies that the hypergraph is 2-colorable?

References

[ADF93] Karl A. Abrahamson, Rodney G. Downey, and Michael R. Fellows. Fixed-parameter intractability ii (extended abstract). In P. Enjalbert, A. Finkel, and K. W. Wagner, editors, *STACS 93*, pages 374–385, Berlin, Heidelberg, 1993. Springer Berlin Heidelberg.

[BB13] Małgorzata Bednarska-Bzdęga. On weight function methods in chooser–picker games. *Theoretical Computer Science*, 475:21–33, 2013.

[BBHL16] Małgorzata Bednarska-Bzdęga, Dan Hefetz, and Tomasz Łuczak. Picker–chooser fixed graph games. *Journal of Combinatorial Theory, Series B*, 119:122–154, 2016.

[Bec02] József Beck. Positional games and the second moment method. *Combinatorica*, 22:169–216, 2002.

[BGL+17] Édouard Bonnet, Serge Gaspers, Antonin Lambilliotte, Stefan Rümmele, and Abdallah Saffidine. The Parameterized Complexity of Positional Games. In Ioannis Chatzigiannakis, Piotr Indyk, Fabian Kuhn, and Anca Muscholl, editors, *44th International Colloquium on Automata, Languages, and Programming (ICALP 2017)*, volume 80 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 90:1–90:14, Dagstuhl, Germany, 2017. Schloss Dagstuhl – Leibniz-Zentrum für Informatik.

[BH19] Kyle Burke and Bob Hearn. Pspace-complete two-color placement games. *International Journal of Game Theory*, 48, 06 2019.

[BJS16] Édouard Bonnet, Florian Jamain, and Abdallah Saffidine. On the complexity of connection games. *Theoretical Computer Science*, 644:2–28, 2016. Recent Advances in Computer Games.

[Bys04] Jesper Makholm Byskov. Maker-maker and maker-breaker games are pspace-complete. *BRICS Report Series*, 11(14), 2004.

[CE78] Vašek Chvátal and Paul Erdős. Biased positional games. In *Annals of Discrete Mathematics*, volume 2, pages 221–229. Elsevier, 1978.

[CMP09] András Csernenszky, C. Ivett Mándity, and András Pluhár. On chooser–picker positional games. *Discrete Mathematics*, 309(16):5141–5146, 2009.

[CMP11] András Csernenszky, Ryan Martin, and András Pluhár. On the complexity of chooser-picker positional games. *Integers*, 2012:427–444, 11 2011.

[Cse10] András Csernenszky. The chooser-picker 7-in-a-row-game. *Publicationes Mathematicae*, 76, 04 2010.

[ER60] P. Erdös and R. Rado. Intersection theorems for systems of sets. *Journal of the London Mathematical Society*, s1-35(1):85–90, 1960.

[ES73] Paul Erdös and John L. Selfridge. On a combinatorial game. *Journal of Combinatorial Theory, Series A*, 14(3):298–301, 1973.

[Gal23] Florian Galliot. *Hypergraphes et jeu Maker-Breaker : une approche structurelle*. PhD thesis, Université Grenoble Alpes, 2023. Thèse de doctorat dirigée par Gravier, Sylvain et Sivignon, Isabelle Mathématiques et informatique Université Grenoble Alpes 2023.

[GGS22] Florian Galliot, Sylvain Gravier, and Isabelle Sivignon. Maker-breaker is solved in polynomial time on hypergraphs of rank 3. *arXiv preprint arXiv:2209.12819*, 2022.

[GGS23] Florian Galliot, Sylvain Gravier, and Isabelle Sivignon. (k- 2)-linear connected components in hypergraphs of rank k. *Discrete Mathematics & Theoretical Computer Science*, 25(Special issues), 2023.

[GO23] Valentin Gledel and Nacim Oijid. Avoidance Games Are PSPACE-Complete. In Petra Berenbrink, Patricia Bouyer, Anuj Dawar, and Mamadou Moustapha Kanté, editors, *40th International Symposium on Theoretical Aspects of Computer Science (STACS 2023)*, volume 254 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 34:1–34:19, Dagstuhl, Germany, 2023. Schloss Dagstuhl – Leibniz-Zentrum für Informatik.

[HJ63] Robert I. Hales and Alfred W. Jewett. Regularity and positional games. *Trans. Am. Math. Soc*, 106:222–229, 1963.

[HKSS14] Dan Hefetz, Michael Krivelevich, Miloš Stojaković, and Tibor Szabó. *Positional games*, volume 44. Springer, 2014.

[HKT16] Dan Hefetz, Michael Krivelevich, and Wei En Tan. Waiter–client and client–waiter planarity, colorability and minor games. *Discrete Mathematics*, 339(5):1525–1536, 2016.

[Kno12] Fiachra Knox. Two constructions relating to conjectures of beck on positional games. *arXiv preprint arXiv:1212.3345*, 2012.

[Kut05] Martin Kutz. Weak positional games on hypergraphs of rank three. *Discrete Mathematics & Theoretical Computer Science*, AE, European Conference on Combinatorics, Graph Theory and Applications (EuroComb '05), 2005.

[Nen23] Rajko Nenadov. Probabilistic intuition holds for a class of small subgraph games. *Proceedings of the American Mathematical Society*, 151(04):1495–1501, 2023.

[Oij24] Nacim Oijid. *Complexity of positional games on graphs*. PhD thesis, Université Claude Bernard, Lyon 1, 2024. Thèse de doctorat dirigée par Duchêne, Eric et Parreau, Aline.

[RS95] Neil Robertson and Paul D Seymour. Graph minors. xiii. the disjoint paths problem. *Journal of combinatorial theory, Series B*, 63(1):65–110, 1995.

[RW20] Md Lutfar Rahman and Thomas Watson. Tractable unordered 3-cnf games. In *Latin American Symposium on Theoretical Informatics*, pages 360–372. Springer, 2020.

[RW21] Md Lutfar Rahman and Thomas Watson. 6-uniform maker-breaker game is pspace-complete. In *Proceedings of the 38th International Symposium on Theoretical Aspects of Computer Science (STACS)*, 2021.

[Sch78] Thomas J. Schaefer. On the Complexity of Some Two-Person Perfect-Information Games. *Journal of computer and system Sciences*, 16:185–225, 1978.

[Sey80] Paul D Seymour. Disjoint paths in graphs. *Discrete mathematics*, 29(3):293–309, 1980.

[Shi80] Yossi Shiloach. A polynomial solution to the undirected two paths problem. *Journal of the ACM (JACM)*, 27(3):445–456, 1980.

[SM73] Larry J. Stockmeyer and Albert R. Meyer. Word problems requiring exponential time (preliminary report). In *Proceedings of the fifth annual ACM symposium on Theory of computing*, pages 1–9, 1973.