Faster Deterministic Algorithms for r-Dimensional Matching Using Representative Sets

Prachi Goyal[1], Neeldhara Misra[1], and Fahad Panolan[2]

1 **Indian Institute of Science, Bangalore, India**
`prachi.goyal|neeldhara@csa.iisc.ernet.in`
2 **Institute of Mathematical Sciences, Chennai, India**
`fahad@imsc.res.in`

Abstract

Given a universe $U := U_1 \uplus \cdots \uplus U_r$, and a r-uniform family $\mathcal{F} \subseteq U_1 \times \cdots \times U_r$, the r-DIMENSIONAL MATCHING problem asks if $\mathcal{F}$ admits a collection of k mutually disjoint sets. The special case when $r = 3$ is the classic 3D-MATCHING problem. Recently, several improvements have been suggested for these (and closely related) problems in the setting of randomized parameterized algorithms. Also, many approaches have evolved for deterministic parameterized algorithms. For instance, for the 3D-MATCHING problem, a combination of color coding and iterative expansion yields a running time of $\mathcal{O}^*(2.80^{(3k)})$, and for the r-DIMENSIONAL MATCHING problem, a recently developed derandomization for known algebraic techniques leads to a running time of $\mathcal{O}^*(5.44^{(r-1)k})$.

In this work, we employ techniques based on dynamic programming and representative families, leading to a deterministic algorithm with running time $\mathcal{O}^*(2.85^{(r-1)k})$ for the r-DIMENSIONAL MATCHING problem. Further, we incorporate the principles of iterative expansion used in the literature [TALG 2012] to obtain a better algorithm for 3D-MATCHING, with a running time of $\mathcal{O}^*(2.003^{(3k)})$. Apart from the significantly improved running times, we believe that these algorithms demonstrate an interesting application of representative families in conjunction with more traditional techniques.

1998 ACM Subject Classification F.2.0 Analysis of Algorithms and Problem Complexity

Keywords and phrases 3-Dimensional Matching, Fixed-Parameter Algorithms, Iterative Expansion

Digital Object Identifier 10.4230/LIPIcs.FSTTCS.2013.237

1 Introduction

Given a universe $U := U_1 \uplus \cdots \uplus U_r$, and a r-uniform family $\mathcal{F} \subseteq U_1 \times \cdots \times U_r$, the r-DIMENSIONAL MATCHING problem asks if $\mathcal{F}$ admits a collection of k mutually disjoint sets. The special case when $r = 3$ can be viewed as an immediate generalization of the matching problem on bipartite graphs to three-partite, three-uniform hypergraphs. The question of finding the largest 3D-Matching is a classic optimization problem, and the decision version is listed as one of the six fundamental NP-complete problems in Garey and Johnson [9]. These problems may also be thought as restricted versions of the more general SET PACKING questions, where no restrictions are assumed on the universe.

r-Dimensional Matching. The question of r-DIMENSIONAL MATCHING has enjoyed substantial attention in the context of exact algorithms, and there have been several deterministic and randomized approaches to the problem (see Table 1). One of the earliest approaches [6] used the color-coding technique [1]. Further, in [7], an $\mathcal{O}(k^3)$ kernel was developed and the color-coding was combined with dynamic programming on the structure of the kernel to obtain an improvement. In [10], Koutis used an algebraic formulation of SET PACKING and proposed a randomized algorithm (derandomized with hash families). The randomized approaches saw further improvements in subsequent work [12, 11, 2], also based on algebraic techniques. The common theme in these developments is to express a parameterized problem in an algebraic framework by associating multilinear monomials with the combinatorial structures that are sought, ultimately arriving at multilinear monomial testing problem or polynomial identity testing problem.

In a recent development [4], the authors propose a derandomization method for these algebraic approaches, leading to a determinstic algorithm that solves the r-DIMENSIONAL MATCHING problem in time $\mathcal{O}^*(5.44^{(r-1)k})$[1].

Table 1 Algorithms for r-DIMENSIONAL MATCHING.

References	Randomized Algorithms	Deterministic Algorithms
Fellows *et al.*, 1999 [6]		$\mathcal{O}^*((rk)!(rk)^{3rk+1})$
Fellows *et al.*, 2008 [7]		$\mathcal{O}^*(2^{O(rk)})$
Koutis, 2005 [10]	$\mathcal{O}^*((4e)^{rk})$	$\mathcal{O}^*(25.6^{rk})$ (See also [3, 13])
Koutis, 2008 [12]	$\mathcal{O}^*(2^{rk})$	
Koutis and Williams, 2009 [11]	$\mathcal{O}^*(2^{(r-1)k})$	
Björklund *et al.*, 2010 [2]	$\mathcal{O}^*(2^{(r-2)k})$	
Chen and Chen, 2013 [4]		$\mathcal{O}^*(5.44^{(r-1)k})$
This work (Theorem 3.4)		$\mathcal{O}^*(2.851^{(r-1)k})$

3D-Matching. The 3D-MATCHING problem admits of even better algorithms than those obtained by using the algorithms above for the particular case of $r = 3$. Indeed, the works [13, 3] consider 3D-MATCHING separately and obtain better algorithms with deterministic running times $\mathcal{O}^*(2.77^{3k})$ and $\mathcal{O}^*(2.80^{3k})$. In fact, even in the present work, we will explore an improvement that is specific to 3D-MATCHING, resulting in a significantly faster algorithm. The approach in [3] uses a clever combination of dynamic programming (embedded in a color coding framework) and iterative expansion, derandomized using hash families. In our work, we obtain a deterministic algorithm with running time $\mathcal{O}^*(2.0034^{3k})$.

The Method of Representative Families and Our Contributions

We first consider the general problem of finding a maximum weight r-dimensional k-packing (that is, the sets have integer weights and the goal is to find a k-packing of weight at least W). Next, we focus on the 3D-MATCHING question separately, improving the general algorithm further for this special case. Our algorithms are an improvization of the work in [3], in that

[1] The $\mathcal{O}^*$ notation is used to suppress polynomial factors in the running time.

the main dynamic programming and iterative expansion wireframes stay intact, however, the color coding engine is replaced by an application of *representative families*.

In using representative families, which is a general approach based on uniform matroids, our attempt is somewhat different from the previous approaches to the problem. Matroids have been applied to design fixed-parameter tractable algorithms as early as [14], where the main tool was the notion of representative families. The idea of using representative families was introduced by Monien in [15]. We briefly describe the basic notion here, and the reader is referred to Section 2 for the details.

In a dynamic programming framework, one might think of every intermediate stage of the algorithm as a location storing several partial solutions, with the hope that one of them would "lead to" a final solution. In many settings, a solution (when it exists) can be viewed as a split into two disjoint parts — where one can be thought of as the partial part that we would like to store, and the other is the part that we are hoping to encounter down the line. The inherent disjointness of these parts suggests that perhaps all partial solutions need not be maintained, and instead, as long as there is some witness that can be evolved into a complete solution, correctness is guaranteed.

We note that the color coding approach is a rather clever way of capturing this notion, albeit with an element of randomization. The notion of representative families, on the other hand, formalizes this intuition in a combinatorial fashion. The definition of a representative family is as follows. Let $M = (E, \mathcal{I})$ be a matroid and let $\mathcal{S} = \{S_1, \ldots, S_t\}$ be a family of subsets of E of size p. A subfamily $\widehat{\mathcal{S}} \subseteq \mathcal{S}$ is *q-representative* for $\mathcal{S}$ if for every set $Y \subseteq E$ of size at most q, if there is a set $X \in \mathcal{S}$ disjoint from Y with $X \cup Y \in \mathcal{I}$, then there is a set $\widehat{X} \in \widehat{\mathcal{S}}$ disjoint from Y with $\widehat{X} \cup Y \in \mathcal{I}$.

For this definition to be useful, we will need to know that small representative families exist. A classical result due to Bollobás indicates that uniform matroids, involving sets of size at most p, admit a q-representative family with at most $\binom{p+q}{p}$ sets (subsequently generalized by Lovász to representable matroids). The next natural issue is that of computational overhead, which brings us to the question of whether these representative families be found, and at what cost. Fortunately, the combinatorial proofs turn out to be constructive, and algorithmic versions have been established. The most recent development in this line of work is in [8], where the authors describe an algorithm constructing a q-representative family of size at most $\binom{p+q}{p}$ in time bounded by a polynomial in $\binom{p+q}{p}$, t, and the time required for field operations, where t is the size of the input family.

Representative families can be applied in a very natural way to a large class of "matching and packing" problems. For instance, consider the DP table for 3-Set Packing that stores in $S[i]$ packings of size i. This approach can be improved by keeping only $(3k - 3i)$-representative families at every stage, leading to an algorithm with running time $\mathcal{O}^*(2.851^{3k})$.

Our focus is to improve this naive approach by more careful dynamic programming, leading to an algorithm with running time $\mathcal{O}^*(2.851^{(r-1)k})$ for r-dimensional matching (see Theorem 3.4). For 3D-Matching, we are able to apply iterative expansion as used in [3]. The idea of iterative expansion is to focus on the "improvement" version of the question, where the input includes a matching of size k and the question is if there is a matching of size $k+1$. Clearly, an efficient algorithm for this question can be run k times (starting with $k = 1$) to find a matching of size k. One of the reasons it is useful to have a matching of size k as input is Lemma 3.4 in [3], which states if there is an improved matching, then there is one that overlaps the given matching in a substantial way. The guarantee of this overlap can be exploited suitably to attain better running times. We use this result as well, except that we incorporate the advantages in the setting of representative families. The result is an algorithm with running time $\mathcal{O}^*(2.0034^{3k})$ for 3D-Matching (see Theorem 4.5).

Another advantage of our first approach is the fact that it works for the weighted version of the r-dimensional matching problem with only an additional factor of $\log W$ (where W is the target weight). This factor is linear in the input, and is thus essentially optimal. To the best of our knowledge, the known approaches either are not scalable to the weighted version of the question or incur an additional factor of W in the running time.

Organization. In Section 2, we discuss some of the definitions associated with matroids and representative families. In Section 3, we describe our first algorithm for the weighted version of r-dimensional matching. Finally, in Section 4, we show how iterative expansion can be used to obtain an improved algorithm for 3D-Matching. Owing to limitations of space, some proofs have been omitted. Such results are marked with a $\star$.

2 Preliminaries

In this section we summarize the important definitions. A more detailed introduction may be found in the full version. To begin with, we define the notion of a (min/max) representative family and state the theorems associated with efficient computations of such families.

▶ **Definition 2.1** (**q-Representative Family**,[8]). Given a matroid $M = (E, \mathcal{I})$ and a family $\mathcal{S}$ of subsets of E, we say that a subfamily $\widehat{\mathcal{S}} \subseteq \mathcal{S}$ is *q-representative* for $\mathcal{S}$ if the following holds: for every set $Y \subseteq E$ of size at most q, if there is a set $X \in \mathcal{S}$ disjoint from Y with $X \cup Y \in \mathcal{I}$, then there is a set $\widehat{X} \in \widehat{\mathcal{S}}$ disjoint from Y with $\widehat{X} \cup Y \in \mathcal{I}$. If $\widehat{\mathcal{S}} \subseteq \mathcal{S}$ is q-representative for $\mathcal{S}$ we write $\widehat{\mathcal{S}} \subseteq^q_{rep} \mathcal{S}$. Also, we say that $\widehat{X}$ is a q-representative for X with respect to Y.

▶ **Definition 2.2** (**Min/Max q-Representative Family**,[8]). Given a matroid $M = (E, \mathcal{I})$, a family $\mathcal{S}$ of subsets of E and a non-negative weight function $w : \mathcal{S} \to \mathbb{N}$, we say that a subfamily $\widehat{\mathcal{S}} \subseteq \mathcal{S}$ is *min q-representative* (*max q-representative*) for $\mathcal{S}$ if the following holds: for every set $Y \subseteq E$ of size at most q, if there is a set $X \in \mathcal{S}$ disjoint from Y with $X \cup Y \in \mathcal{I}$, then there is a set $\widehat{X} \in \widehat{\mathcal{S}}$ disjoint from Y with $\widehat{X} \cup Y \in \mathcal{I}$; and $w(\widehat{X}) \leq w(X)$ ($w(\widehat{X}) \geq w(X)$). We use $\widehat{\mathcal{S}} \subseteq^q_{minrep} \mathcal{S}$ ($\widehat{\mathcal{S}} \subseteq^q_{maxrep} \mathcal{S}$) to denote a min q-representative (max q-representative) family for $\mathcal{S}$. Also, we say that $\widehat{X}$ is a min q-representative (respectively, max q-representative) for X with respect to Y.

▶ **Theorem 2.3** ([8]). *Let $\mathcal{S} = \{S_1, \ldots, S_t\}$ be a family of sets of size p over a universe of size n. For a given q, a q-representative family $\widehat{\mathcal{S}} \subseteq \mathcal{S}$ for $\mathcal{S}$ with at most $\binom{p+q}{p} \cdot 2^{o(p+q)} \cdot \log n$ sets can be computed in time $\mathcal{O}((\frac{p+q}{q})^q \cdot 2^{o(p+q)} \cdot t \cdot \log n)$.*

▶ **Theorem 2.4** ([8]). *There is an algorithm that given a p-family $\mathcal{S}$ of sets over a universe U of size n, an integer q, and a non-negative weight function $w : \mathcal{S} \to \mathbb{N}$ with maximum value at most W, computes in time $\mathcal{O}(|\mathcal{S}| \cdot (\frac{p+q}{q})^q \cdot \log n + |\mathcal{S}| \cdot \log |\mathcal{S}| \cdot \log W)$ a subfamily $\widehat{\mathcal{S}}$ such that $|\widehat{\mathcal{S}}| = \binom{p+q}{p} \cdot 2^{o(p+q)} \cdot \log n$ and $\widehat{\mathcal{S}} \subseteq^q_{minrep} \mathcal{S}$ ($\widehat{\mathcal{S}} \subseteq^q_{maxrep} \mathcal{S}$).*

The problems we consider are the following:

r-DIMENSIONAL MATCHING **Parameter:** k
Input: A universe $U := U_1 \uplus \cdots \uplus U_r$, a family $\mathcal{F} \subseteq U_1 \times \cdots \times U_r$, and $k \in \mathbb{Z}^+$.
Question: Does $\mathcal{F}$ have a collection of at least k mutually disjoint sets?

WEIGHTED EXACT r-DIMENSIONAL MATCHING **Parameter:** k
Input: A universe $U := U_1 \uplus \cdots \uplus U_r$, a family $\mathcal{F} \subseteq U_1 \times \cdots \times U_r$, a non-negative weight function $w : \mathcal{F} \to \mathbb{N}$ and positive integers k, W.
Question: Does there exist $\mathcal{M} \subseteq \mathcal{F}$, of k mutually disjoint sets such that $w(\mathcal{M}) \geq W$?

3 A Dynamic Programming Approach

In this section, we describe an algorithm with running time $\mathcal{O}^*(2.851^{(r-1)k})$ for the WEIGHTED EXACT r-DIMENSIONAL MATCHING problem. For ease of presentation, we will describe the algorithm for WEIGHTED EXACT 3-DIMENSIONAL MATCHING here. The generalization to the problem of finding a weighted r-dimensional matching of size k appears in the full version of this work.

Let $(U_1, U_2, U_3, \mathcal{F}, w : \mathcal{F} \to \mathbb{N}, k, W)$ be an instance of WEIGHTED EXACT 3D-MATCHING. Recall that $\mathcal{F} \subseteq U_1 \times U_2 \times U_3$, and the problem involves finding k mutually disjoint sets in $\mathcal{F}$ whose weight is at least W. For a set $S \in \mathcal{F}$, we let $S[i]$ denote $S \cap U_i$, that is, $S[i]$ is the element of S that is from U_i. We sometimes refer to the element $S[i]$ as the i^{th} coordinate of S.

The improved dynamic programming approach involves iterating over the elements in U_3 (for the discussion in this section, this is an arbitrary and fixed choice). In particular, let $U_3 := \{c_1, c_2, \ldots, c_n\}$. Let $\mathcal{M} := \{S_1, S_2, \ldots, S_t\}$ be a 3D-Matching. Let $\mathcal{M}_3 := \{S_i[3] \mid i \in [t]\} \subseteq U_3$. We define the *maximum last index* of $\mathcal{M}$, denoted by $\lambda(\mathcal{M})$, as the largest index i for which $c_i \in \mathcal{M}_3$. For the empty matching $\emptyset$, $\lambda(\emptyset) = 0$. In the i^{th} iteration of our algorithm, we would like to store all matchings whose maximum last index is at most i, and we stop at our first encounter with a matching of size k and weight at least W. However, it turns out that storing all possible matchings at every stage is exponentially expensive — we potentially run into storing $O(n^{2i})$ matchings at the i^{th} step. In [3], this problem (for the unweighted case) is mitigated using color coding. We will now discuss how we can use the notion of max representative families instead, which has the dual advantage of being faster and deterministic.

We first introduce some notation. We let $\mathcal{Q}^{(i)} = \{\mathcal{M} \mid \lambda(\mathcal{M}) \leq i\}$ and $\mathcal{Q}_j^{(i)} := \{\mathcal{M} \mid \mathcal{M} \in \mathcal{Q}^{(i)}, |\mathcal{M}| = j\}$. Notice that the $\mathcal{Q}_j^{(i)}$'s constitute a partition of the set $\mathcal{Q}^{(i)}$ based on matching size, in other words, $\mathcal{Q}^{(i)} := \bigcup_{j=0}^{n} \mathcal{Q}_j^{(i)}$. Recall that a naive application of the method of representative families would lead to a running time of $\mathcal{O}(2.85^{3k})$. To use the representative families more efficiently, we would like to splice the elements of the matchings into two parts. We will first collect the parts of the matching that come from $(U_1 \cup U_2)$, and then store separately a map that completes the first part to the complete matching. This will allow us to apply the dynamic programming approach suggested above. To this end, for a matching $\mathcal{M}$, we let $\mathcal{M}_{12}$ denote the subset of $(U_1 \cup U_2)$ obtained by projecting the elements of $\mathcal{M}$ on their first two co-ordinates, that is, $\mathcal{M}_{12} := \bigcup_{S \in \mathcal{M}} \{S[1], S[2]\}$. On other hand, let

$$\gamma := \{(\mathcal{M}, \mathcal{M}_{12}) \mid \mathcal{M} \text{ is a matching in } \mathcal{F}\}.$$

Algorithm 1: A Simple DP Implementation for WEIGHTED EXACT 3D-MATCHING using Representative Families

```
Input: $(U_1, U_2, U_3, \mathcal{F}, w : \mathcal{F} \to \mathbb{N}, k, W)$, where $\mathcal{F} \subseteq U_1 \times U_2 \times U_3$, and $k, W \in \mathbb{N}$.
Output: A 3D-Matching of size $k$ and weight at least $W$ if it exists, NO otherwise.
1  $\mathcal{R}_0^{(0)} \leftarrow \{\emptyset\}$
2  $\mathcal{R}_j^{(0)} \leftarrow \emptyset$, for all $1 \leq j \leq k$
3  $\mathcal{L}^{(0)} \leftarrow \{\emptyset\}$
4  for $i \in \{1, 2, \ldots, n\}$ do
5      $\mathcal{L}^{(i)} \leftarrow \{\mathcal{M} \cup S \mid \mathcal{M} \in \mathcal{L}^{(i-1)}, S \in \mathcal{F}, \mathcal{M} \cap S = \emptyset, S[3] = i\} \cup \mathcal{L}^{(i-1)}$
6      for $j \in \{0, 1, \ldots, k\}$ do
7          $\mathcal{L}_j^{(i)} \leftarrow \{\mathcal{M} \mid \mathcal{M} \in \mathcal{L}^{(i)}, |\mathcal{M}| = j\}$
8          if $\mathcal{L}_k^{(i)} \neq \emptyset$ and $\exists \mathcal{M} \in \mathcal{L}_k^{(i)}$ such that $w(\mathcal{M}) \geq W$ then
9              return $\mathcal{M}$.
10         $\mathcal{P}_j^{(i)} \leftarrow \{\mathcal{M}_{12} \mid \mathcal{M} \in \mathcal{L}_j^{(i)}\}$;
11         $\gamma \leftarrow \{(\mathcal{M}, \mathcal{M}_{12}) \mid \mathcal{M}$ is a matching in $\mathcal{L}_j^{(i)}\}$
12         $w' \leftarrow \{(\mathcal{M}_{12}, w(\gamma^\dagger(\mathcal{M}_{12}))) \mid \mathcal{M}_{12} \in \mathcal{P}_j^{(i)}\}$
13         Let $\mathcal{R}_j^{(i)} \subseteq_{maxrep}^{2k-2j} \mathcal{P}_j^{(i)}$
14         $\mathcal{L}_j^{(i)} \leftarrow \{\gamma^\dagger(\mathcal{M}_{12}) \mid \mathcal{M}_{12} \in \mathcal{R}_j^{(i)}\}$
15     $\mathcal{L}^{(i)} \leftarrow \bigcup_{j=0}^{k} \mathcal{L}_j^{(i)}$
16 return NO;
```

In γ, we are merely storing the associations of $\mathcal{M}_{12}$ with the matchings that they "came from". Observe that γ might (by definition) contain multiple entries with the same second coordinate. On the other hand, when the algorithm stores the associations in γ, we will see that it will be enough to maintain one maximum weighted entry for each $\mathcal{M}_{12}$. To this end, we define the function $\gamma^\dagger$ as follows. Let $\leq$ be an arbitrary total order on the set of all matchings in $\mathcal{F}$. For a set $S \subseteq U_1 \cup U_2$, we define $\gamma^\dagger(S)$ as the smallest matching $\mathcal{M}$ (with respect to $\leq$) among the maximum weighted matchings in the set $\{\mathcal{M}' \mid (\mathcal{M}', S) \in \gamma\}$. If $\gamma^\dagger(S)$ is $\mathcal{M}$, then we say that $\mathcal{M}$ is the matching associated with S. Finally, for a set S and a matching $\mathcal{M}$, we abuse notation and say that $\mathcal{M}$ is disjoint from S (notationally, $\mathcal{M} \cap S = \emptyset$) to mean that $T \cap S = \emptyset$ for all $T \in \mathcal{M}$.

We are now ready to describe our first algorithm (see also Algorithm 1). In $\mathcal{L}^{(i)}$, we store carefully chosen 3D-Matchings whose maximum last index is at most i. We compute $\mathcal{L}^{(i)}$ by considering all elements $\mathcal{M}$ of $\mathcal{L}^{(i-1)}$, and checking if they can be extended to a matching whose maximum last index is at most i. For a fixed $\mathcal{M}$, this check is performed by iterating over all elements $S \in \mathcal{F}$ such that $S[3] = i$ and extending $\mathcal{M}$ to $\mathcal{M} \cup S$ whenever $\mathcal{M} \cap S = \emptyset$. Subsequently, we will compute a representative family of $\mathcal{L}^{(i)}$. Since $\mathcal{L}^{(i)}$ is a collection of matchings of varying sizes, we will need an appropriate version of $\mathcal{L}^{(i)}$ that will be suitable for Theorem 2.4. To this end, we first partition the set $\mathcal{L}^{(i)}$ — the part denoted by $\mathcal{L}_j^{(i)}$ contains all matchings from $\mathcal{L}^{(i)}$ of size j. Note that this is simply done to ensure uniformity of size. Of course, it is easily seen that if we have a matching of size k with weight at least W at this point, then we are done.

Next, we associate with every matching $\mathcal{M} \in \mathcal{L}_j^{(i)}$, a set that consists of the first two indices of every set in $\mathcal{M}$. Recall that this is denoted by $\mathcal{M}_{12}$. The collection of sets that correspond to matchings in $\mathcal{L}_j^{(i)}$ is denoted by $\mathcal{P}_j^{(i)}$. We use γ to store the associations between the sets and the original matchings. Note that γ might have multiple pairs with the same second index and same weight w for first index, but this will be irrelevant (it would simply mean that $\mathcal{M}_{12}$ can be "pulled back" to multiple matchings, each of which would be equally valid). Now we define a weight function $w' : \mathcal{P}_j^{(i)} \to \mathbb{N}$. $\forall \mathcal{M}_{12} \in \mathcal{P}_j^{(i)}$, we set $w'(\mathcal{M}_{12}) = w(\gamma^{\dagger}(\mathcal{M}_{12}))$. Now, the central step of the algorithm is to compute a max $(2k-2j)$-representative family $\mathcal{R}_j^{(i)}$ for $\mathcal{P}_j^{(i)}$. Once we have the representative family, we revise $\mathcal{L}_j^{(i)}$ to only include the associated matchings with sets in $\mathcal{R}_j^{(i)}$.

The correctness of the algorithm relies on the fact that at each step, instead of the complete family of partial solutions $\mathcal{Q}^{(i)}$, it suffices to store only a representative family of $\mathcal{Q}^{(i)}$. Also, we will show that the family computed by the algorithm, $\mathcal{L}^{(i)}$, is indeed a representative family for $\mathcal{Q}^{(i)}$. The analysis of the running time will be quite straightforward in the light of Theorem 2.4, and we will discuss it after arguing the correctness.

Let $\mathcal{X}_j^{(i)} := \{\mathcal{M}_{12} \mid \mathcal{M} \in \mathcal{Q}_j^{(i)}\}$. We assign a weight function $w' : \mathcal{X}_j^{(i)} \to \mathbb{N}$ as follows. $\forall \mathcal{M}_{12} \in \mathcal{X}_j^{(i)}$, $w'(\mathcal{M}_{12}) = \max\{w(\mathcal{M}^\star) \mid \mathcal{M}^\star \in \mathcal{Q}_j^{(i)}, \mathcal{M}_{12} = \mathcal{M}^\star_{12}\}$. Our first claim is the following.

▶ **Lemma 3.1** ($\star$)**.** *For all $0 \le i \le n$, and $0 \le j \le k-1$, the set $\mathcal{R}_j^{(i)}$ is a max $(2k-2j)$-representative family for $\mathcal{X}_j^{(i)}$.*

Observe that any solution constructed by Algorithm 1 is always a valid matching. Therefore, if there is no matching of size k, Algorithm 1 always returns a No. On the other hand, if given a Yes instance, we now show that Algorithm 1 always finds a 3D-Matching of size k with weight at least W.

▶ **Lemma 3.2.** *Let $(U_1, U_2, U_3, \mathcal{F}, w : \mathcal{F} \to \mathbb{N}, k, W)$ be a* Yes*-instance of* Weigted Exact 3D-Matching*. Then, Algorithm 1 successfully computes a 3D-Matching of size k with weight at least W.*

Proof. Let $(U_1, U_2, U_3, \mathcal{F}, w : \mathcal{F} \to \mathbb{Z}^+, k, W)$ be a Yes-instance of Weighted Exact 3D-Matching. Let $\mathcal{M} = \{S_1, S_2, \ldots, S_k\}$, be a k-sized 3D-Matching in $\mathcal{F}$ and $w(\mathcal{M}) \ge W$. Suppose $\lambda(\mathcal{M}) = i$. Without loss of generality, let $S_k[3] = c_i$. Finally, let $\mathcal{M}' = \mathcal{M} \setminus S_k$. Recall that $\mathcal{Q}_j^{(i)}$ is the set of all 3D-Matchings of size j with maximum last index at most i, and $\mathcal{X}_j^{(i)}$ contains the projections of these matchings on their first two coordinates. Therefore, $\mathcal{M}' \in \mathcal{Q}_{k-1}^{(i-1)}$ and $\mathcal{M}'_{12} \in \mathcal{X}_{k-1}^{(i-1)}$. Note that $w'(\mathcal{M}'_{12}) \ge w(\mathcal{M}')$.

By Lemma 3.1, we have that $\mathcal{R}_{k-1}^{(i-1)}$ is a max 2-representative family for $\mathcal{X}_{k-1}^{(i-1)}$. Let $Y = \{S_k[1], S_k[2]\}$. Since $\mathcal{R}_{k-1}^{(i-1)} \subseteq^2_{maxrep} \mathcal{X}_{k-1}^{(i-1)}$, we have that $\mathcal{R}_{k-1}^{(i-1)}$ contains a 2-representative $Z^\star$, for $\mathcal{M}'_{12}$ with respect to Y. So we have $w'(Z^\star) \ge w'(\mathcal{M}'_{12})$. Let $\mathcal{M}^\star$ be the matching associated with $Z^\star$. Therefore $w(\mathcal{M}^\star) = w'(Z^\star)$. It is easy to see that $\mathcal{M}^\star \cap S_k = \emptyset$ and $\lambda(\mathcal{M}^\star \cup S_k) = i$. Therefore, in Step 2 of Algorithm 1, we have that $\mathcal{L}^{(i)}$ contains the matching $\mathcal{M}^\star \cup S_k$ which is then classified in the set $\mathcal{L}_k^{(i)}$. Consider the weight of $\mathcal{M}^\star \cup S_k$.

$$\begin{aligned} w(\mathcal{M}^\star \cup S_k) &= w(\mathcal{M}^\star) + w(S_k) \\ &\ge w(\mathcal{M}') + w(S_k) \qquad (\text{Since } w(\mathcal{M}^\star) = w'(Z^\star) \ge w'(\mathcal{M}'_{12}) \ge w(\mathcal{M}')) \\ &\ge W \qquad (\text{Since } w(\mathcal{M}') + w(S_k) = w(\mathcal{M}) \ge W) \end{aligned}$$

Thus, the algorithm will return a matching of size k and weight at least W in Step 9. ◀

▶ **Lemma 3.3** ($\star$). *The running time of Algorithm 1 is bounded by* $O(2.851^{2k})$.

We remark that in case of unweigted 3D-Matching, if there exists a 3D-Matching of size at least k then there exists a 3D-Matching of size k, and so we can use Algorithm 1 to solve 3D-Matching. The extension of these ideas to the Weighted Exact r-Dimensional Matching is discussed in the full version of the paper. The final result is the following.

▶ **Theorem 3.4.** Weighted Exact r-Dimensional Matching *and* r-Dimensional Matching *can be solved deterministically in time* $\mathcal{O}^*(2.851^{(r-1)k})$.

4 Iterative Expansion and Representative Sets

In this section, we provide a faster algorithm for 3D-Matching using iterative expansion. The idea is to first solve the following improvement problem: given a 3D-Matching of size j as input, can we find a 3D-Matching of size $(j+1)$? Once we have an algorithm for solving the improvement question, we can use it as a subroutine k times to find a matching of size k, by starting with a trivial matching of size one. Therefore, for the rest of this section, we focus on the improvement problem:

3D-Matching Improvement **Parameter:** k
Input: A universe $U := U_1 \uplus U_2 \uplus U_3$, a family $\mathcal{F} \subseteq U_1 \times U_2 \times U_3$, and $\mathcal{K} \subseteq \mathcal{F}$, a 3D-Matching of size k.
Question: Does $\mathcal{F}$ have a 3D-Matching of size $(k+1)$?

Let $\mathcal{M}$ be a 3D-Matching given by $\{S_1, \dots, S_t\}$. Generalizing the notation in the previous section, we let $\mathcal{M}_{pq}$ denote the subset of $(U_p \cup U_q)$ obtained by projecting the elements of $\mathcal{M}$ on the p and q co-ordinates, that is, $\mathcal{M}_{pq} := \bigcup_{S \in \mathcal{M}} \{S[p], S[q]\}$. On other hand, let

$$\gamma_{pq} := \{(\mathcal{M}, \mathcal{M}_{pq}) \mid \mathcal{M} \text{ is a matching in } \mathcal{F}\}.$$

The definition of $\gamma^{\dagger}_{pq}$ is also as expected: For a set $S \subseteq U_p \cup U_q$, we define $\gamma^{\dagger}_{pq}(S)$ as the smallest matching $\mathcal{M}$ (with respect to $\leq$) for which $(\mathcal{M}, S) \in \gamma_{pq}$. Finally, for $r \in \{1,2,3\}$, let $\mathcal{M}_r := \{S_i[r] \mid i \in [t]\} \subseteq U_r$. Let $U_r = \{c_1, c_2, \dots, c_n\}$. We define the *maximum last index of* $\mathcal{M}$ *with respect to* r, denoted by $\lambda_r(\mathcal{M})$, as the largest index i for which $c_i \in \mathcal{M}_r$. For empty matching $\emptyset$, $\lambda_r(\mathcal{M}) = 0$. To use the method of iterative expansion to our advantage, we invoke the following result from [3], which states that if there is *some* matching of size $(k+1)$, then there is also one that has a large intersection with the given matching $\mathcal{K}$.

▶ **Lemma 4.1** (Lemma 3.4 and Theorem 3.6, [3]). *Let* $(U_1, U_2, U_3, \mathcal{F}, \mathcal{K})$ *be a* Yes*-instance of* 3D-Matching Improvement*, that is,* $\mathcal{F}$ *admits a matching* $\mathcal{M}$ *of size* $(k+1)$*. Then, there is also a matching* $\mathcal{M}^\star$ *of size* $(k+1)$ *such that there exist two indices* $p, q \in \{1,2,3\}$*, for which* $|\mathcal{M}^\star_{pq} \cap \mathcal{K}_{pq}| \geq (4/3)k$.

The improved algorithm for solving the 3D-Matching Improvement (see Algorithm 2) is along the lines of Algorithm 1, in that the dynamic programming procedure is very similar. The difference lies in how we compute the representative families (see lines 17-19, Algorithm 2), and this is also what brings about the improved running time.

We begin by considering the given matching, $\mathcal{K}$. By Lemma 4.1, we know that there is always a solution that has a large common intersection with $\mathcal{K}$ (if one solution exists).

In particular, if we denote this solution by $\mathcal{M}^\star$ then Lemma 4.1 indicates that there is a choice of coordinates $p, q \in \{1,2,3\}$ for which the number of elements in $\mathcal{K}_{pq} \cap \mathcal{M}^\star_{pq}$ is at least $(4/3)k$. So our first step involves guessing the coordinates p and q (Steps 1-2 in Algorithm 2). Once we have fixed a choice of p and q, we further guess the intersection $\mathcal{K}_{pq} \cap \mathcal{M}^\star_{pq}$. We do this by iterating over all subsets of $\mathcal{K}_{pq}$ (see Line 12, Algorithm 2). Let U be a fixed guess of the elements in the intersection. In $\mathcal{P}^{(i)}_{j,U}$ we store (as before) the projection of the matchings from $\mathcal{L}^{(i)}_j$ on U_p and U_q — however we are now only interested in matchings that satisfy $\mathcal{M}_{pq} \cap \mathcal{K}_{pq} = U$. In the next step, we remove the elements of U from every set in $\mathcal{P}^{(i)}_{j,U}$ to obtain $\mathcal{D}^{(i)}_{j,U}$. Now, we compute a $(2k/3 - 2j + |U| + 2)$-representative family of $\mathcal{D}^{(i)}_{j,U}$. The intuitive explanation is the following. Since $\mathcal{M}_{pq}$ already has $(4k/3)$ elements in common with $\mathcal{K}_{pq}$, we could isolate these elements, set them aside, and compute representative families only for the "rest". Due to this restrictive computation of representative families, the amount of time we spend on this step improves considerably.

Towards correctness, we define $\mathcal{Q}^{(i)}[\![pq]\!]$, which is analogous to the notion of $\mathcal{Q}^{(i)}$ in the previous section. The only difference is that the maximum last index value is considered along the r coordinate, where $r \in \{1,2,3\}$ is such that $r \neq p, q$. Further, it will be useful to define $\mathcal{Q}^{(i)}_{j,U}[\![pq]\!]$, which further partitions the collection $\mathcal{Q}^{(i)}[\![pq]\!]$ based on the matching size and the intersection with $\mathcal{K}_{pq}$, i.e, $\mathcal{Q}^{(i)}_{j,U}[\![pq]\!] := \{\mathcal{M} \mid \mathcal{M} \in \mathcal{Q}^{(i)}, |\mathcal{M}| = j, \mathcal{M}_{pq} \cap \mathcal{K}_{pq} = U\}$ Finally let $\mathcal{X}^{(i)}_{j,U}[\![pq]\!] := \{\mathcal{M}_{pq} \mid \mathcal{M} \in \mathcal{Q}^{(i)}_{j,U}[\![pq]\!]\}$.

In this setting, our first claim is that if we have a Yes instance of 3D-Matching Improvement, then for a correct guess of the coordinates p and q, at least one relevant partial solution is preserved in $\mathcal{L}^{(i)}$ at every stage of the algorithm.

▶ **Lemma 4.2.** *For all $0 \le i \le n$, for all $U \subseteq \mathcal{K}_{pq}$, and for all $0 \le j \le k$ such that $2j \le (2k/3) + |U| + 2$, we have that $\mathcal{R}^{(i)}_{j,U}[\![pq]\!]$ is a $\alpha(j,U)$-representative family for $\mathcal{X}^{(i)}_{j,U}[\![pq]\!]$, where $\alpha(j,U) = (2k/3) - (2j - |U|) + 2$.*

Proof. Towards a proof, we need to show that for all $Y \subseteq U_p \cup U_q$ such that $|Y| \le \alpha(j,U)$, if there exists $Z \in \mathcal{X}^{(i)}_{j,U}[\![pq]\!]$ such that $Y \cap Z = \emptyset$, then there also exists $Z^\star \in \mathcal{R}^{(i)}_{j,U}[\![pq]\!]$ such that $Z^\star \cap Y = \emptyset$. Note that since Y is assumed to be disjoint from Z, and $U \subseteq Z$ (since $Z \in \mathcal{X}^{(i)}_{j,U}[\![pq]\!]$), we have that $Y \cap U = \emptyset$. The proof of the lemma is by induction on i. The base case is when $i = 0$.

$$\mathcal{R}^{(0)}_{j,U}[\![pq]\!] = \mathcal{X}^{(0)}_{j,U}[\![pq]\!] = \begin{cases} \{\emptyset\} & \text{if } j = 0 \text{ and } U = \emptyset, \\ \emptyset & \text{Otherwise.} \end{cases}$$

Hence $\mathcal{R}^{(0)}_{j,U}[\![pq]\!]$ is $\alpha(j,U)$-representative family for $\mathcal{X}^{(0)}_{j,U}[\![pq]\!]$ for all $0 \le j \le k$ and $U \subseteq \mathcal{K}_{pq}$. The induction hypothesis states that $\mathcal{R}^{(i)}_{j,U}[\![pq]\!]$ is an $\alpha(j,U)$-representative family for $\mathcal{X}^{(i)}_{j,U}[\![pq]\!]$ for all $U \subseteq \mathcal{K}_{pq}$ and for all $0 \le j \le k$ such that $2j \le (2k/3) + |U| + 2$. For the induction step, we will show that $\mathcal{R}^{(i+1)}_{j,U}[\![pq]\!]$ is an $\alpha(j,U)$-representative family for $\mathcal{X}^{(i+1)}_{j,U}[\![pq]\!]$ for all $U \subseteq \mathcal{K}_{pq}$ and for all $0 \le j \le k$ such that $2j \le (2k/3) + |U| + 2$. As with the proof of Lemma 3.1, we note that the corner case when $j = 0$ is trivial. To this end, let $U \subseteq \mathcal{K}_{pq}$ and j be fixed. Let $Y \subseteq U_p \cup U_q$ such that $|Y| \le \alpha(j,U)$. Further, suppose there exists a set $Z \in \mathcal{X}^{(i+1)}_{j,U}[\![pq]\!]$ such that $Z \cap Y = \emptyset$. Let $\mathcal{M}^Z$ be the matching associated with Z. Since $\mathcal{M}^Z$ is derived from $\mathcal{X}^{(i+1)}_{j,U}[\![pq]\!]$, note that $\lambda_r(\mathcal{M}^Z) \le i + 1$. We distinguish two cases, depending on whether $\mathcal{M}^Z$ contains a set with c_{i+1} as the r-coordinate.

Algorithm 2: An Expansion Algorithm using Representative Families

```
Input: $(U_1, U_2, U_3, \mathcal{F}, \mathcal{K})$, where $\mathcal{F} \subseteq U_1 \times U_2 \times U_3$, $\mathcal{K} \subseteq \mathcal{F}$ is a 3D-Matching of size $k$
Output: A 3D-Matching of size $(k+1)$ if it exists, No otherwise.
for $r$ in $\{1,2,3\}$ do
    $p \leftarrow \max\{\{1,2,3\} \setminus \{r\}\}$ and $q \leftarrow \min\{\{1,2,3\} \setminus \{r\}\}$
    $\mathcal{R}^{(0)}_{0,\emptyset}[\![pq]\!] \leftarrow \{\emptyset\}$, $\mathcal{L}^{(0)} \leftarrow \{\emptyset\}$
    $\mathcal{R}^{(0)}_{j,U}[\![pq]\!] \leftarrow \emptyset$, if $j \neq 0$ or $U \neq \emptyset$, for all $0 \leq j \leq k$ and $U \subseteq \mathcal{K}_{pq}$
    for $i \in \{1, 2, \ldots, n\}$ do
        $\mathcal{L}^{(i)} \leftarrow \{\mathcal{M} \cup S \mid \mathcal{M} \in \mathcal{L}^{(i-1)}, S \in \mathcal{F}, \mathcal{M} \cap S = \emptyset, S[r] = i\} \cup \mathcal{L}^{(i-1)}$
        for $j \in \{0, 1, 2, \ldots, k+1\}$ do
            $\mathcal{L}^{(i)}_j \leftarrow \{\mathcal{M} \mid \mathcal{M} \in \mathcal{L}^{(i)}, |\mathcal{M}| = j\}$
            if $\mathcal{L}^{(i)}_{k+1} \neq \emptyset$ then
                return $\mathcal{M}$, where $\mathcal{M} \in \mathcal{L}^{(i)}_{k+1}$.
            $\gamma_{pq} \leftarrow \{(\mathcal{M}, \mathcal{M}_{pq}) \mid \mathcal{M}$ is a matching in $\mathcal{L}^{(i)}_j\}$;
            for $U \subseteq \mathcal{K}_{pq}$ do
                if $2j > (2/3)k + |U| + 2$ then
                    Continue;
                $\mathcal{P}^{(i)}_{j,U}[\![pq]\!] \leftarrow \{\mathcal{M}_{pq} \mid \mathcal{M} \in \mathcal{L}^{(i)}_j$ and $\mathcal{M}_{pq} \cap \mathcal{K}_{pq} = U\}$
                $\mathcal{D}^{(i)}_{j,U}[\![pq]\!] \leftarrow \{X \setminus U \mid X \in \mathcal{P}^{(i)}_{j,U}[\![pq]\!]\}$
                Let $\mathcal{T}^{(i)}_{j,U}[\![pq]\!] \subseteq^{\alpha(j,U)}_{rep} \mathcal{D}^{(i)}_{j,U}[\![pq]\!]$, where $\alpha(j,U) = (2k/3) - (2j - |U|) + 2$
                Let $\mathcal{R}^{(i)}_{j,U}[\![pq]\!] \leftarrow \{X \cup U \mid X \in \mathcal{T}^{(i)}_{j,U}[\![pq]\!]\}$
            $\mathcal{L}^{(i)}_j \leftarrow \bigcup_{U \subseteq \mathcal{K}_{pq}} \{\gamma^{\dagger}_{pq}(\mathcal{M}_{pq}) \mid \mathcal{M}_{pq} \in \mathcal{R}^{(i)}_{j,U}[\![pq]\!]\}$
        $\mathcal{L}^{(i)} \leftarrow \bigcup_{j=1}^{k} \mathcal{L}^{(i)}_j$
return No;
```

Case 1. $\mathcal{M}^Z$ contains a set with c_{i+1} as the r-coordinate.

Let $S \in \mathcal{M}^Z$ be such that $c_{i+1} \in S$. Define the smaller matching $\mathcal{M}^{Z\setminus S} := \mathcal{M}^Z \setminus S$. Notice that $|\mathcal{M}^{Z\setminus S}| = j-1$ and $\lambda_r(\mathcal{M}^{Z\setminus S}) \leq i$. Let $U_S := U \cap S$ and $U' = U \setminus U_S$. Therefore, $\mathcal{M}^{Z\setminus S} \in \mathcal{Q}^{(i)}_{j-1,U'}[\![pq]\!]$ and $\mathcal{M}^{Z\setminus S}_{pq} \in \mathcal{X}^{(i)}_{j-1,U'}[\![pq]\!]$. Let $A = \mathcal{M}^{Z\setminus S}_{pq}$ and $Y^S = Y \cup (\{S[p], S[q]\} \setminus U_S)$. It is easy to check that $A \cap Y^S = \emptyset$. We claim that $|Y^S| \leq \alpha(j-1, U')$:

$$|Y^S| = |Y| + 2 - |U_S| \leq (2k/3) - 2j + |U| + 2 + 2 - |U_S| = \alpha(j, U')$$

We also claim that $2(j-1) \leq (2k/3) + |U'| + 2$. Suppose not, then $2j > (2k/3) + |U'| + 4 > (2k/3) + |U| + 2$ (Since $|U| = |U'| + |U_S| \leq |U'| + 2$). This contradicts the assumption that $2j \leq (2k/3) + |U| + 2$. Hence, we are now in a situation where $A \in \mathcal{X}^{(i)}_{j-1,U'}[\![pq]\!]$, $|Y^S| \leq \alpha(j-1, U')$, $A \cap Y^S = \emptyset$ and $2(j-1) \leq (2k/3) + |U'| + 2$. By the induction hypothesis, we have that $\mathcal{R}^{(i)}_{j-1,U'}[\![pq]\!]$ contains a $\alpha(j-1, U')$-representative of A with respect to Y^S. Let us denote this representative by B. Note that $B \cap Y^S = \emptyset$ by definition. Also note that $B \cap U_S = \emptyset$ because $B \cap \mathcal{K}_{pq} = U'$. Let $\mathcal{M}^B$ be the matching associated with B. Since $B \in \mathcal{R}^{(i)}_{j-1,U'}[\![pq]\!]$, we have that $\mathcal{M}^B \in \mathcal{L}^{(i)}_{j-1}$. Since the p, q-coordinates of S are disjoint from B and the r-coordinate of S is c_{i+1}, we have that $\mathcal{M}^B \cap S = \emptyset$. Thus, the matching $\mathcal{M}^B \cup S \in \mathcal{L}^{(i+1)}_j$. Let us denote this matching by $\mathcal{M}^{B|S}$. It is easy to check that

$\mathcal{M}_{pq}^{B|S} \cap \mathcal{K}_{pq} = U$, and correspondingly, $\mathcal{M}_{pq}^{B|S} \in \mathcal{P}_{j,U}^{(i+1)}[\![pq]\!]$. Let $W := \mathcal{M}_{pq}^{B|S} \setminus U$. Note that $W \in \mathcal{D}_{j,U}^{(i+1)}[\![pq]\!]$ and $W \cap Y = \emptyset$. Since $\mathcal{T}_{j,U}^{(i+1)}[\![pq]\!]$ is an $\alpha(j,U)$-representative family for $\mathcal{D}_{j,U}^{(i+1)}[\![pq]\!]$, we have that $\mathcal{T}_{j,U}^{(i+1)}[\![pq]\!]$ contains an $\alpha(j,U)$-representative $W^\star$ for W with respect to Y. We now define $Z^\star$ to be $W^\star \cup U$. Note that $Z^\star \in \mathcal{R}_{j,U}^{(i+1)}[\![pq]\!]$. Since $W^\star$ is disjoint from Y by definition, and we know that $U \cap Y = \emptyset$, we conclude that $Z^\star$ is the desired representative.

Case 2. $\mathcal{M}^Z$ contains no set with c_{i+1} as the r-coordinate.

Recall that $Z \in \mathcal{X}_{j,U}^{(i+1)}[\![pq]\!]$, and therefore, $Z \cap \mathcal{K}_{pq} = U$. Since we additionally have that $\lambda(\mathcal{M}^Z) \leq i$, $\mathcal{M}^Z \in \mathcal{Q}_{j,U}^{(i)}[\![pq]\!]$. Consequently, $\mathcal{M}_{pq}^Z \in \mathcal{X}_{j,U}^{(i)}[\![pq]\!]$. By induction hypothesis there exists a $\alpha(j,|U|)$-representative, B, for $\mathcal{M}_{pq}^Z$ with respect to Y, in $\mathcal{R}_{j,U}^{(i)}[\![pq]\!]$ Note that $B \cap Y = \emptyset$. Thus the matching associated with B, say $\mathcal{M}^B$, belongs to $\mathcal{L}_j^{(i)}$. Further, since $\mathcal{L}_j^{(i+1)} \supseteq \mathcal{L}_j^{(i)}$, we have that $\mathcal{M}^B \in \mathcal{L}_j^{(i+1)}$. As before, this implies that $\mathcal{M}_{pq}^B \in \mathcal{P}_{j,U}^{(i+1)}[\![pq]\!]$. Note that this may be equivalently stated as $B \in \mathcal{P}_{j,U}^{(i+1)}[\![pq]\!]$. The rest of the argument is identical to the last part of the previous case, leading to a representative of Z as desired. ◀

We now establish the correctness of Algorithm 2, and then establish the running time.

▶ **Lemma 4.3.** *Let $(U_1, U_2, U_3, \mathcal{F}, \mathcal{K})$ be a* Yes*-instance of* 3D-Matching Improvement*. Then, Algorithm 2 successfully computes a 3D-Matching of size $(k+1)$.*

Proof. Since $(U_1, U_2, U_3, \mathcal{F}, \mathcal{K})$ be a Yes-instance of 3D-Matching Improvement, by Lemma 4.1, we have that there exists a matching $\mathcal{M}$ such that $|\mathcal{M}| = (k+1)$ and there exists $p, q \in \{1,2,3\}$ such that $|\mathcal{M}_{pq} \cap \mathcal{K}_{pq}| \geq (4k/3)$. Let $r \in \{1,2,3\}$ such that $r \neq p, q$. Let $\mathcal{M} = \{S_1, S_2, \ldots, S_k, S_{k+1}\}$. Suppose $\lambda_r(\mathcal{M}) = i$. Without loss of generality, let $S_{k+1}[r] = c_i$. Finally, let $\mathcal{M}'$ denote the matching $\mathcal{M} \setminus \{S_{k+1}\}$. Let $U = \mathcal{M}_{pq} \cap \mathcal{K}_{pq}$, $U_S = S_{k+1} \cap \mathcal{K}_{pq}$ and $U' = U \setminus U_S$. Note that $\mathcal{M}'_{pq} \in \mathcal{X}_{k,U'}^{(i-1)}$ and $2k \leq (2k/3) + |U'| + 2$ (since $|U'| \geq (4k/3) - 2$). Now consider the value $\alpha(k, U')$.

$$\alpha(k,U') = (2k/3) - 2k + |U'| + 2 \geq (2k/3) - 2k + (4k/3 - |U_S|) + 2 = 2 - |U_S|$$

By Lemma 4.2, we have that $\mathcal{R}_{k,U'}^{(i-1)}$ is a $\alpha(k,U')$-representative family for $\mathcal{X}_{k,U'}^{(i-1)}$, where $\alpha(k,U') \geq 2 - |U_S|$. Now consider $Y := \{S_k[p], S_k[q]\} \setminus U_S$. Note that $|Y| = 2 - |U_S|$. Since $\mathcal{M}'_{pq} \in \mathcal{X}_{k,U'}^{(i-1)}$ is clearly disjoint from Y, by the definition of a representative family, there exists some set, say $X \in \mathcal{R}_{j,U'}^{(i-1)}$, which is also disjoint from Y. Let $\mathcal{M}^\star$ be the matching associated with X, that is, let $\mathcal{M}^\star := \gamma_{pq}^\dagger(X)$. It is easily checked that $\mathcal{M}^\star \cup S_{k+1}$ is included in $\mathcal{L}_{k+1}^{(i)}$ and is therefore returned as a matching of size $(k+1)$ in Step 9 of Algorithm 2. ◀

▶ **Lemma 4.4** ($\star$). *The running time of Algorithm 2 is bounded by $\mathcal{O}^\star(2.0034^{3k})$.*

Putting together Lemmas 4.2, 4.3, and 4.4, we have the following theorem.

▶ **Theorem 4.5.** *The* 3D-Matching *problem can be solved in time $\mathcal{O}^\star(2.0034^{3k})$.*

5 Conclusions

We have demonstrated that when combined with techniques like iterative expansion, representative families can be used to a great advantage inside dynamic programming routines.

We were able to significantly improve on the state of the art for the of r-Dimensional Matching problem (even with weights), and we further improved this running time for the special case of the 3D-Matching problem. We already know that representative families can be applied to a variety of other packing problems. The naive approach, however, yields only incremental improvements in running times. The more dramatic improvements obtained for the matching problems considered in this work were due to the additional structure in the input: the partitioned universe was exploited in the dynamic programming.

It will be very interesting to see what other techniques can be combined with this method to obtain improved algorithms for other packing questions, such as the classic Set Packing problem, or the problem of packing paths of length two from an underlying graph.

References

1 Alon, Yuster, and Zwick. Color-coding. *JACM: Journal of the ACM*, 42, 1995.

2 Andreas Björklund, Thore Husfeldt, Petteri Kaski, and Mikko Koivisto. Narrow sieves for parameterized paths and packings. *CoRR*, abs/1007.1161, 2010.

3 Jianer Chen, Yang Liu, Songjian Lu, Sing-Hoi Sze, and Fenghui Zhang. Iterative expansion and color coding: An improved algorithm for 3D-matching. *ACM Transactions on Algorithms*, 8(1):6, 2012.

4 Shenshi Chen and Zhixiang Chen. Faster deterministic algorithms for packing, matching and t-dominating set problems. *CoRR*, abs/1306.3602, 2013.

5 R. G. Downey and M. R. Fellows. *Parameterized Complexity.* Springer, 1999.

6 R. G. Downey, M. R. Fellows, and M. Koblitz. Techniques for exponential parameterized reductions in vertex set problems. (Unpublished, reported in [5], §8.3).

7 Michael R. Fellows, Christian Knauer, Naomi Nishimura, Prabhakar Ragde, Frances A. Rosamond, Ulrike Stege, Dimitrios M. Thilikos, and Sue Whitesides. Faster fixed-parameter tractable algorithms for matching and packing problems. *Algorithmica*, 52(2):167–176, 2008.

8 Fedor V. Fomin, Daniel Lokshtanov, and Saket Saurabh. Efficient computation of representative sets with applications in parameterized and exact algorithms. *CoRR*, abs/1304.4626, 2013.

9 M. R. Garey and D. S. Johnson. *Computers and Intractability.* Freeman, San Francisco, 1979.

10 I. Koutis. A faster parameterized algorithm for set packing. *Information Processing Letters*, 94:7–9, 2005.

11 I. Koutis and R. Williams. Limits and applications of group algebras for parameterized problems. In *Proceedings of Automata, Languages and Programming, 36th International Colloquium, ICALP*, volume 5555 of *LNCS*, pages 653–664. Springer, 2009.

12 Ioannis Koutis. Faster algebraic algorithms for path and packing problems. In *Proceedings of Automata, Languages and Programming, 35th International Colloquium, ICALP*, volume 5125 of *Lecture Notes in Computer Science*, pages 575–586. Springer, 2008.

13 Y. Liu, S. Lu, J. Chen, and S.-H. Sze. Greedy localization and color-coding: improved matching and packing algorithms. In *International Workshop on Parameterized and Exact Computation IWPEC*, volume 4169 of *LNCS*, pages 84–95. Springer, 2006.

14 Dániel Marx. A parameterized view on matroid optimization problems. *Theor. Comput. Sci*, 410(44):4471–4479, 2009.

15 B. Monien. How to find long paths efficiently. *Ann. Discrete Math.*, 25:239–254, 1985.