Average-case performance of heuristics for three-dimensional random assignment problems

Alan Frieze* Gregory B. Sorkin†

20 Jan 2012

Abstract

Beautiful formulas are known for the expected cost of random two-dimensional assignment problems, but in higher dimensions, even the scaling is not known. In 3 dimensions and above, the problem has natural "Axial" and "Planar" versions, both of which are NP-hard. For 3-dimensional Axial random assignment instances of size n, the cost scales as $\Omega(1/n)$, and a main result of the present paper is the first polynomial-time algorithm that, with high probability, finds a solution of cost $O(n^{-1+\varepsilon})$, for arbitrary positive ε (or indeed ε going slowly to 0). For 3-dimensional Planar assignment, the lower bound is $\Omega(n)$, and we give a new efficient matching-based algorithm that returns a solution with expected cost $O(n \log n)$.

1 Introduction

An instance of the (two dimensional) assignment problem may be thought of as an $n \times n$ cost array $C_{i,j}$, a candidate solution is a permutation $\pi\colon [n] \mapsto [n]$, its cost is $\sum_{i=1}^n C_{i,\pi(i)}$, and an optimal solution is one minimizing the cost. If the cost matrix represents, for example, the costs of assigning various jobs i to machines j, where each machine can accommodate only one job, then the problem's solution represents the cheapest way of assigning the jobs to machines. It may equivalently be formulated as an integer linear program, minimizing the sum of selected elements consistent with the selection of exactly one element from each row and from each column, i.e., minimizing $\sum_{i,j} X_{i,j} C_{i,j}$ where $X_{i,j} \in \{0,1\}$, $(\forall i)\ \sum_j X_{i,j} = 1$ and $(\forall j)\ \sum_j X_{i,j} = 1$. This is a network flow problem, thus its linear relaxation with $X_{i,j} \in [0,1]$ has integer extreme points, and the problem may be solved in polynomial time.

The *random assignment problem*, in its most popular form, is the case when the entries of the cost matrix C are i.i.d. Exp(1) random variables (independent, identically distributed exponential random variables with parameter 1). Since anyway the problem can be solved in polynomial time, the focus for the random case is on the cost's expectation as a function of n,

$$f(n) = \mathbf{E}\left[\min_{\pi} \sum_{i=1}^{n} C_{i,\pi(i)}\right] = \mathbf{E}\left[\min_{X_{i,j}} \sum_{i,j} X_{i,j} C_{i,j}\right]$$

with $X_{i,j} \in \{0,1\}$ subject to the row and column constraints as above. This problem has received a great deal of study over several decades. It was considered from an operations research perspective in the 1960s [Don69], an asymptotic conjecture $f(n) \to \pi^2/6 = \zeta(2)$ was formulated by statistical physicists Mézard and Parisi in the 1980s based on the mathematically sophisticated but non-rigorous "replica method" [MP85, MP87], an exact conjecture $f(n) = \sum_{i=1}^n 1/i^2$ was hazarded by

*Research supported by NSF grant DMS-6721878, Department of Mathematical Sciences, Carnegie Mellon University, Pittsburgh PA15213, e-mail alan@random.math.cmu.edu

†Department of Mathematical Sciences, IBM T.J. Watson Research Center, Yorktown Heights NY 10598, e-mail sorkin@watson.ibm.com

Parisi in the late 1990s [Par98] and a generalization to partial matchings and non-square matrices made by Coppersmith and Sorkin [CS99], the Mézard–Parisi conjecture was proved by Aldous in a pair of papers in 1992 and 2001 [Ald92, Ald01], the Coppersmith–Sorkin conjecture was proved simultaneously in 2004 by two papers using two very different methods [NPS05, LW04]. A further generalisation of these conjectures was made by Buck, Chan and Robbins [1]. This was proved by Wastlund in [4]. The study of other aspects of the random assignment problem and related problems is ongoing, for example by Wästlund in [Wäs09].

In higher dimensions there are two natural generalizations of the assignment problem. For example in three dimensions, the Axial assignment problem is, given an $n \times n \times n$ matrix (or "tensor" or "array") C, to find a solution $X_{i,j,k}$ minimizing $\sum_{i,j,k} X_{i,j,k} C_{i,j,k}$ where $X_{i,j,k} \in \{0,1\}$ and there is one selected value per "plane" of the array:

$$(\forall i) \sum_{j,k} X_{i,j,k} = 1, \quad (\forall j) \sum_{i,k} X_{i,j,k} = 1, \quad (\forall k) \sum_{i,k} X_{i,j,k} = 1. \tag{1}$$

Equivalently, it is to determine $\min_{\pi,\sigma} \sum_{i=1}^{n} C_{i,\pi(i),\sigma(i)}$, the minimum taken over a pair of permutations. The Planar three dimensional assignment problem is similar but with one selected value per "line" of the array:

$$(\forall i,j) \sum_{k} X_{i,j,k} = 1, \quad (\forall j,k) \sum_{i} X_{i,j,k} = 1, \quad (\forall i,k) \sum_{j} X_{i,j,k} = 1. \tag{2}$$

The generalizations to higher dimensions are clear. In 3 dimensions and higher, the Axial and Planar assignment problems are both NP-hard. The Axial case $d = 3$ was one of the original problems listed by Karp [Kar72]. The complexity of the Planar problem was established by Frieze [Fri83].

The *multidimensional random assignment problem* we consider here is the case when the entries of the cost matrix are i.i.d. Exp(1)random variables. In this random setting, there are two natural questions. First, are there polynomial-time algorithms that find optimal or near-optimal solutions **whp**? Second, what is the expected cost of a minimum assignment? A random two-dimensional assignment instance has limiting expected cost $\zeta(2)$, and Frieze showed that the expected cost of a minimum spanning tree in the complete graph with random Exp(1) edge weights tends to $\zeta(3)$ [Fri85], so it is tantalizing to wonder if there might be similarly beautiful expressions for the expected cost in multi-dimensional versions of the random assignment problem. However, we do not even know how the cost scales with n.

Some of the characteristics and applications of these problems are discussed in a recent book by Burkard, Dell'Amico and Martello [BDM09]. Much less is known about the probabilistic behavior of these problems for $d \geq 3$ and even less is known about polynomial time algorithms for constructing good solutions. First consider the "Axial problem" with constraints (1). Grundel, Krokhmal and Pardalos [GOPP05] replace the Exp(1) assumption with more general distributions. Their result most relevant to our discussion is that if $F^{-1}(x) = O(x^{\beta})$ for some $\beta > 0$ as $x \to 0^{+}$ then the minimum value $Z_{d,n}^{P} \to 0$ **whp**. Here $F(x)$ is the probability that $C_{i,j,k} \leq x$. We will see in Section 3.3.1 that this is not difficult to prove $Z_{d,n}^{P} \to 0$ **whp** for $C_{i,j,k} = Exp(1)$, explaining our interest in improving the rate of convergence to zero. Kravtsov [3] describes a class of greedy algorithms that work well if Exp(1) is replaced by uniform over $\{1, 2, \ldots, n^{\alpha}\}$ where $\alpha < 1$ depends on the particular algorithm. The lower bound of 1 means that the minimum is at least n and this is not a difficult target, asymptotically.

The "Planar problem" with constraints (2) was considered by Dichkovskaya and Kravtsov [2]. Here they discuss a "greedy" algorithm similar to that proposed by us. Their analysis is quite different and their distribution is once again of the form uniform over $\{1, 2, \ldots, n^\alpha\}$ where $\alpha < 1$. This makes the minimum at least n^2 and this is not a difficult target, asymptotically.

Statistical physicists have conjectures based on "cavity" calculations [MMR04, MMR05], but there is no such nice constant as $\zeta(2)$, and no certainty even that the conjectured scaling is correct.

2 Summary of results, methods, and limitations

2.1 Axial assignment

For the Axial d-dimensional assignment problem, there is an easy lower bound of $\Omega(1/n^{d-2})$ on the expected cost (see Theorem 1). Our main result (Theorem 2) is for the case $d = 3$. Here we give an algorithm that for any constant $\varepsilon > 0$ runs in time polynomial in n and yields a solution of expected cost $O(1/n^{1-\varepsilon})$, and where ε may be taken slowly to 0 for a "mildly exponential time" algorithm yielding an $n^{o(1)}$ approximation in the average case. Not only is this the first nearly tight upper bound obtained algorithmically, it is the only such bound except for one (see Theorem 1) following from a recent non-constructive result on hypergraph factors by Johansson, Kahn and Vu [JKV08].

Our algorithm may be thought of as an extension of one in [CS99] for 2-dimensional assignment. In that case, a bipartite matching was augmented by an alternating path of bounded length, with care taken to in regard to "conditioning" of the cost matrix. Here, partial assignments are augmented with a "bounded depth alternating path tree", a tree in which a newly added element displaces two previously selected elements, those two elements are replaced in a way displacing four selected elements, and so on, until all the displaced elements are replaced by elements in a non-conflicting, "unassigned" set (see Figure 2.1).

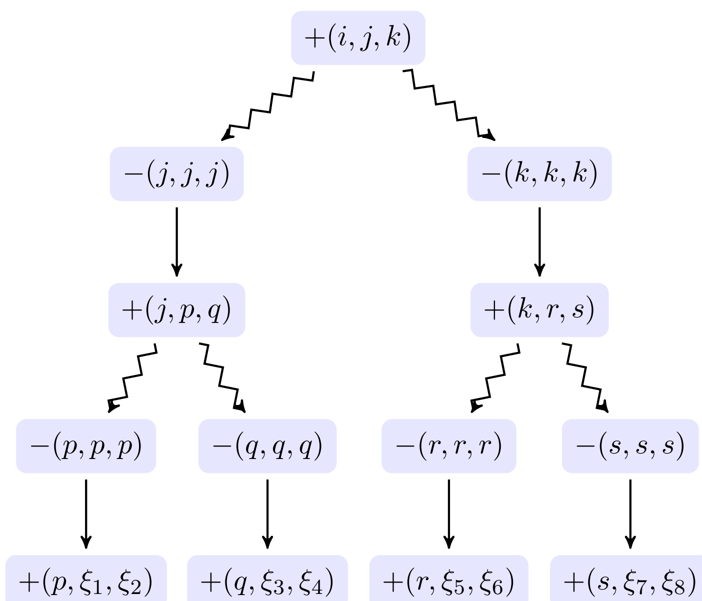

Figure 1: Diagram of alternating-path tree; see Section 3.3.2. Adding new first coordinate i to the partial assignment using hyperedge (i, j, k) implies deletion of previous assignment elements (j, j, j) and (k, k, k), with first coordinates j and k then reassigned respectively to elements (j, p, q) and (k, r, s), displacing four more existing assignment elements (p, p, p) etc., whose first coordinates are finally reassigned to unused second and third coordinates as (p, ξ_1, ξ_2) etc.

2.2 Planar assignment

Our second main result (Theorem 3) is for Planar 3-dimensional assignment, and uses a greedy algorithm: once assignment elements are chosen, they are never replaced. The 3-dimensional assignment consists of n 2-dimensional assignments, with constraints between them. (Ignoring these constraints gives a relaxed 3-dimensional problem consisting of n 2-dimensional problems. Its expected minimum cost is $n\sum_{i=1}^{n} 1/i^2$, proving that the true 3-dimensional cost has expectation $\Omega(n)$.) The first assignment is chosen greedily (at expected cost $\sum_{i=1}^{n} 1/i^2 \leq \pi^2/6$) from a complete bipartite graph $K_{n,n}$. The second assignment is similar, but on a graph $K_{n,n}$ from which the edges of the first matching have been subtracted, so that each vertex has degree $n-2$, and similarly for the subsequent matchings, down to the last one where each vertex has degree 1 and the matching is forced.

The structure of the ith intermediate graph — $K_{n,n}$ minus a union of $i-1$ edge-disjoint perfect matchings — is complex, but a general result of Dyer, Frieze, and McDiarmid [DFM86] is insensitive to the details and gives an upper bound of $2n/(n-i+1)$ for the expectation of its minimum-cost perfect matching, for a total expectation of $O(n \log n)$, close to the $\Omega(n)$ lower bound. Algorithmically there is no difficulty since we are just solving a set of minimum-cost assignment problems.

As in the Axial case, though, our approach to the Planar problem falters for dimensions $d \geq 4$. Here, the 3-dimensional assignment problems are not necessarily regular, and by the end, there can even be 3-dimensional instances with no assignment at all: the greedy algorithm can fail. Perhaps this is not a problem in the average case, but at the least it necessitates a more careful analysis.

2.3 Structure of the paper

We deal with Axial assignment in Section 3, starting with easily proved non-constructive bounds (but where the upper bound relies on some heavy machinery). The next task is to describe and analyze a "bounded depth alternating path tree algorithm" (`BDAPTA`=`BDAPTA`(k)) where $2k$ is the depth of the search tree. This will prove Theorem 2. The description of the algorithm is given later along with the analysis. We first analyze a two level version in Section 3.3, providing intuition. The general case is analyzed in Section 3.4, completing the proof.

The Planar problem is considered in Section 4. The lower bound of Theorem 3 is proved in Section 4.2, and its upper bound in Section 4.3.

3 Multi-Dimensional Axial Version

3.1 Simple bounds

Theorem 1

$$\Omega\left(\frac{1}{n^{d-2}}\right) \leq Z_{d,n}^P \leq O\left(\frac{\log n}{n^{d-2}}\right) \quad \textbf{whp}.$$

Proof. Clearly

$$Z_{d,n}^P \geq \sum_{i_1=1}^{n} \min_{i_2,\ldots,i_d} C_{i_1,\ldots,i_d}.$$

Each term in the above sum is distributed as $\text{Exp}(n^{d-1})$ and so has expectation $1/n^{d-1}$ and variance $1/n^{2d-2}$. The Chebyshev inequality implies that the sum is concentrated around the mean.

For the upper bound we use a recent result of Johansson, Kahn and Vu [JKV08] on perfect matchings in random d-uniform hypergraphs. This implies that **whp** there is a solution that only uses d-tuples of weight at most $\frac{K \log n}{n^{d-1}}$ [1]. The upper bound follows immediately. It should be noted that their proof is non-constructive. □

3.2 Main theorem

We fix the dimension to $d = 3$ for the rest of Section 3. We remind the reader that a description of `BDAPTA`(k) will follow in Sections 3.3 and 3.4.

Theorem 2 *Suppose that $1 \leq k \leq \gamma \log_2 \log n$ where γ is any constant strictly less than 1/2. Then,* **whp***:*

(a) *Algorithm `BDAPTA`(k) runs in time $O(n^{2^{k+2}})$.*

(b) *The cost $C(T)$ of the set of triples T output by `BDAPTA`(k) satisfies $C(T) = O(2^k n^{-1+\theta_k} \log n)$* **whp***, where $\theta_k = \frac{1}{2^{k+1}-1}$.*

As already noted, we are not aware of any other polynomial time algorithm that will **whp** find a solution of value $O(n^{-1+\varepsilon})$, for arbitrary positive ε.

3.3 Two Level Version of `BDAPTA`

In this section we consider a two level version of the algorithm `BDAPTA`. In this way we hope that to make it easier to understand the general version that is described in Section 3.4. With reference to Theorem 2, the two-level version means taking $k = 3$, $\theta = \theta_3 = 1/7$.

The heuristic has three phases:

3.3.1 Greedy Phase

The first phase is a simple greedy procedure.

Greedy Phase

1. Let $n_1 = n - n^{1-\theta}$, $J = K = [n]$, and $T = \emptyset$.[2]
2. For $i = 1, \ldots, n_1$ do the following:
   - Let $C_{i,j,k} = \min\{C_{i,j',k'} : j' \in J, k' \in K\}$;
   - Add (i, j, k) to T and remove j from J and k from K.

At the end of this procedure the triples in T provide a partial assignment. Let

$$Z_1 = \sum_{(i,j,k) \in T} C_{i,j,k}.$$

Lemma 1

$$Z_1 \leq \frac{2}{n^{1-\theta}} \qquad \textbf{whp}.$$

[1] In truth [JKV08] does not deal with d-partite hypergraphs. We have however verified that their result can be extended to this case.

[2] We will often pretend that some expressions are integer. Formally, we should round up or down but it will not matter.

Proof. We observe that if $(i,j,k) \in I$ then $C_{i,j,k}$ is the minimum of $(n-i+1)^2$ independent copies of Exp(1) and is therefore distributed as $\text{Exp}((n-i+1)^2)$. Furthermore, the random variables $C_{i,j,k}, (i,j,k) \in T$ are independent. Using the facts that an $\text{Exp}\,\lambda$ random variable has mean $1/\lambda$ and variance $1/\lambda^2$,

$$\mathbf{E}(Z_1) = \sum_{i=1}^{n_1} \frac{1}{(n-i+1)^2} \leq \int_{x=1}^{n_1+1} \frac{dx}{(n-x+1)^2} \leq \frac{1}{n^{1-\theta}}.$$

Now

$$\mathbf{Var}(Z_1) = \sum_{i=1}^{n_1} \frac{1}{(n-i+1)^4} \leq \frac{3}{n^{3(1-\theta)}} = o(\mathbf{E}(Z_1)^2)$$

and the lemma follows from the Chebyshev inequality. □

Remark 2 If we replace n_1 by $n-\omega$ where $\omega = \omega(n) \to \infty$ and then do exhaustive search for the optimum solution in the remaining ω size problem then we see (i) the expected optimum value is bounded by $O\left(\frac{1}{\omega} + \frac{\log \omega}{\omega}\right)$ (greedy plus exhaustive search costs) and (ii) the running time is bounded by $O\left(n^3 + \omega^{2\omega}\right)$. So, if $\omega = O(\log n / \log\log n)$ then the algorithm is polynomial and produces a solution with an expected (and **whp**) $o(1)$ cost.

3.3.2 Main Phase

The aim of this phase is to increase the size of the partial assignment defined by T to $n - O(1)$. Let $I = I(T)$ be the set of first coordinates assigned in T, i.e., $I = I(T) = \{i : \exists j,k \text{ s.t. } (i,j,k) \in T\}$. Relabeling if necessary, without loss of generality we may assume that $I = [|T|]$. This phase will be split into *rounds*. We choose a small constant $0 < \alpha \ll 1$ and let $\beta = 1 - \alpha$. The aim of a round is to reduce the size of the set of unmatched first coordinates $X(T) = [n] \setminus I(T)$ by a factor β while increasing the total cost of the matching only by an acceptably small amount. Thus we let $x_1 = n - n_1$ and $x_t = \beta^{t-1} x_1$ for $t \geq 2$. The aim of round t is to reduce $|X(T)|$ from x_t to x_{t+1}. We continue this for $t_0 = \log_{1/\beta}(x_1/L)$ rounds where L is a large positive constant. Thus at the end of the Main Phase, if successful, we will have a partial assignment of size at least $n - 2L$.

So suppose now that we are at the start of a round and that $|X(T)| = x_t$. This is true for $t = 1$. Next let $w_0 = 2\gamma n^{-12/7} \log n$ where $\gamma = 2^{k+2} - 2 = 6$ and

$$w_t = 2\gamma n^{-6/7} x_t^{-8/7} \log^{1/7} n \qquad \text{for } t \geq 1.$$

At the start of each round we will *refresh* the array C with independent exponentials, at some cost. By this we mean that we replace C by a new array C' where $C_{i,j,k} \leq C'_{i,j,k} + w_{t-1}$ and the entries of C' are i.i.d. Exp(1) random variables. More precisely, suppose that during the previous round we determined the precise values for all $C_{i,j,k} \leq w_{t-1}$ and left our state of knowledge for the other $C_{i,j,k}$ as being at least w_{t-1}. Then the memoryless property of exponentials means that

$$C'_{i,j,k} = \begin{cases} C_{i,j,k} - w_{t-1} & \text{when } C_{i,j,k} > w_{t-1} \\ \text{fresh } X_{i,j,k} \sim \text{Exp}(1) & \text{otherwise} \end{cases}$$

has the claimed property. Thus we can start a round with a fresh matrix of independent exponentials at the expense of adding another w_{t-1} to each cost. We note also that we can **whp** carry out the Greedy Phase only looking at those $C_{i,j,k}$ of value less than w_0.

Let T_t denote the value of T at the start of round t and let $I_t = I(T_t), X_t = X(T_t)$. In round t we will add $A_t = [n - x_t + 1, n - x_{t+1}]$ to I_t. By relabeling if necessary we will assume that at the start of round t we have $T = \{(i,i,i) : 1 \leq i \leq n - x_t\}$. To add $i \in A_t$ to I_t we find distinct indices $j,k,p,q,r,s \in I_t$ (distinctness is not strictly necessary) and replace 6 of the triples in T_t by 7 new triples (see Figure 2.1:

$$+(i,j,k) - (j,j,j) - (k,k,k) + (j,p,q) + (k,r,s) - (p,p,p) - (q,q,q) - (r,r,r) - (s,s,s) + (p,\xi_1,\xi_2) + (q,\xi_3,\xi_4) + (r,\xi_5,\xi_6) + (s,\xi_7,\xi_8), \quad (3)$$

where $\xi_1,\ldots,\xi_8$ are distinct members of X_t, and each of the triples added in (3) is required to have (refreshed) cost at most w_t. Roughly, we are assigning a new 1-coordinate i, this collides with previously used 2-coordinate j and 3-coordinate k, so the (j,j,j) and (k,k,k) elements are removed from the existing assignment, 1-coordinates j and k are re-added as (j,p,q) and (k,r,s) thus colliding with the previous assignment elements (p,p,p), (q,q,q), (r,r,r), and (s,s,s), and finally 1-coordinates p, q, r, s are re-added as (p,ξ_1,ξ_2) etc., where the ξ_i are elements *not* previously assigned. One may think of (3) as a binary tree version of an alternating-path construction; we will control the cost despite the tree's expansion.

Putting $W_t = w_0 + w_1 + \cdots + w_t$ we see that if we can add one element to T at a cost of at most w_t in refreshed costs, then in reality it costs us at most W_t; step (3) increases the cost by $\leq 7W_t$. Success in a round means doing this $x_t - x_{t+1}$ times, in which case the additional cost of the Main Phase will be at most 7 times

$$\sum_{t=1}^{t_0}(x_t - x_{t+1})W_t \leq x_1 w_0 + \sum_{t=1}^{t_0} x_t w_t \leq 2\gamma n^{-6/7}\log n + 2\gamma x_1^{-1/7} n^{-6/7}\log^{1/7} n \sum_{t=1}^{t_0}\beta^{-t/7} = O(n^{-6/7}\log n). \quad (4)$$

We must now show that **whp** it is possible to add $x_t - x_{t+1} = \alpha x_t$ triples in round t with a (refreshed) cost of at most $7w_t$ per triple. For this we fix t and drop the suffix t from all quantities that use it. We will treat refreshed costs as actual costs and drop the word "refreshed".

We let $\nu = n - x$ and partition $[\nu]$ into $\gamma = 6$ sets $Q_{a,b}, 1 \leq a \leq 2,\ 1 \leq b \leq 2^a$ of nearly equal size. Our choices of $p,q,\ldots,r,s$ are such that $j \in Q_{1,1}, k \in Q_{1,2}, p \in Q_{2,1}, \ldots, s \in Q_{2,4}$. We now estimate the number of choices $p \in Q_{2,1}$ for assigning $p \in Q_{2,1}$. The number of choices is distributed as the binomial $\mathrm{Bin}(\nu/\gamma, 1 - e^{-wx^2}) = \mathrm{Bin}(\nu/\gamma, (1-o(1))wx^2)$. Here $1 - e^{-wx^2}$ is the probability that for a given p, there exist ξ_1, ξ_2 such that $C_{p,\xi_1,\xi_2} \leq w$. Note that

$$wx^2 = 2(x/n)^{6/7}\log^{1/7} n = o(1) \text{ and that } wnx^2 \gg \log n$$

and so the Chernoff bounds imply that, **qs**,[3] we can choose a set P of size exactly $wnx^2/2\gamma = o(n)$, such that for each $p \in P$ there is at least one choice $\xi_1, \xi_2 \in X$ such that the triple (p,ξ_1,ξ_2) is *good*, i.e., $C_{p,\xi_1,\xi_2} \leq w$. Given this set of choices P,the number of choices for $q \in Q_{2,2}$ is distributed as the binomial $\mathrm{Bin}(\nu/\gamma, 1 - e^{-wx^2})$ and we can once again **qs** choose a set Q such that $|Q| = wnx^2/2\gamma$ and each $q \in Q$ is in some good triple (q,ξ_3,ξ_4) where $\xi_3,\xi_4 \in X$. Similarly, we can choose sets $R \subseteq Q_{2,3}$, $S \subseteq Q_{2,4}$ of choices for r, s, of size $wnx^2/2\gamma$.

Observation 3 *Each $\xi \in X$ is in at most $\mathrm{Bin}(x\nu, 1 - e^{-w})$ good triples of the form $(p \in P, \xi', \xi'')$ and so* **qs** *it is in at most $2wnx$ such triples. (Here $wnx = \Omega\left(\frac{n\log n}{x}\right)^{1/7} \gg \log n$).*

[3] A sequence of events $\mathcal{E}_n, n \geq 0$ are said to occur *quite surely*, **qs**, if $\mathbf{Pr}(\mathcal{E}_n) = 1 - O(n^{-c})$ for any constant $c > 0$.

We now discuss our choices for j and k. For a fixed $j \in Q_{1,1}$ there are $w^2n^2x^4/\gamma^2$ pairs in $P \times Q$ and each has a probability $1-e^{-w}$ of forming a good triple (j,p,q). Let $j \in L$ be *useful* if there is such a pair and *useless* otherwise. Then

$$\mathbf{Pr}(j \text{ is useless}) \leq \exp\left\{-\frac{w^3n^2x^4}{4\gamma^2}\right\} \leq 1 - \frac{w^3n^2x^4}{6\gamma^2}.$$

It follows that the number of useful $j \in L$ dominates $(1-o(1))\,\mathrm{Bin}(n/\gamma, w^3n^2x^4/5\gamma^2)$ and so **qs** we can choose a set J of useful $j \in L$ of size

$$\frac{w^3n^3x^4}{8\gamma^3} = n^{3/7}x^{4/7}\log^{3/7} n = o(n).$$

We can by a similar argument choose a same size set K of useful $k \in Q_{1,2}$.

Observation 4 *A fixed p is in at most $\mathrm{Bin}(wn^2x^2, 1-e^{-w})$ good triples (j,p,q) where $(j,q) \in J \times Q$ and so* **qs** *every p is in at most $2w^2n^2x^2$ such triples.*

Suppose then that in the middle of a round we have added $y < \alpha x$ triples to T. The number of $\xi \in X$ that can be used in a good triple (p,ξ,η) will have been reduced by y. The number of η will have been reduced by the same amount. It follows from Observation 3 that the number of choices for p will have been reduced by at most $2\alpha x \times 2wnx/\gamma$. By Observation 4 this reduces the number of choices for j by at most $2\alpha x \times 2wnx/\gamma \times 2w^2n^2x^2 + 7\alpha x \ll |J| = w^3n^3x^4/8\gamma^3$. The additional term $+7\alpha x$ accounts for the choices we lost because they have previously been used in this round. So our next i will get a choice of at least $\mathrm{Bin}((w^3n^3x^4/8\gamma^3)^2, 1-e^{-w})$ choices for a good triple (i,j,k). So the expected number of choices is at least $w^7n^6x^8/64\gamma^6 = 2\gamma\log n$ and then the probability there is no choice is $o(n^{-1})$. This is sufficient to ensure that **whp** there is always at least one choice for every i.

3.3.3 Final Phase

We now have to add only $O(1)$ indices to I. At this point there is a problem with the bottom-up approach of the previous phase if $x < 8$, clearest in the case $x = 1$, say the single element n, when each of $\xi_1, \ldots, \xi_8$ would have to be n, leading to an illegal assignment. Thus instead we will work top down. The details of this will cause more conditioning of the matrix, and therefore we refresh C after each increase in I, at an extra cost of $w = \gamma n^{-6/7}\log^{1/7} n$. So, if successful, the cost of this round is $O(W_{t_0} + w) = O(n^{-6/7}\log^{1/7} n)$.

Let us now replace the notation of (3) by

$$+(i,j,k) - (j,j,j) - (k,k,k) + (j,p,q) + (k,r,s) - (p,p,p) - (q,q,q) - (r,r,r) - (s,s,s) + (p,i_2,p) + (q,q,j) + (r,s,i_3) + (s,k,r), \quad (5)$$

We hold to the convention that the triples of I are of the form (j,j,j). i_2, i_3 are unused 2- and 3-coordinates respectively.

Fix j (and thus its previously matched companion indices j_1, j_3) and let Z_j be the number of choices for p, q (with their previously matched companion indices p_1, p_3, q_1, q_2) such that $C(j_1,p,q), C(p_1,i_2,p_3), C(q_1,q_2,j_3) \leq w$. This has the distribution $B_1(B_2(n,w)B_3(n,w),w)$ where B_1, B_2, B_3 denote independent binomials, with B_2 counting the good choices for p, B_3 those for q, and B_1 those for j using these p and q possibilities. Using Chernoff bounds on the binomials B_2, B_3 we see that **whp** Z_j dominates $B(n^2w^2/2, w)$ which dominates $\mathrm{Be}(n^2w^3/3)$, the Bernoulli random variable that is 1 with probability $n^2w^3/3$ and 0 otherwise. The same holds for index k and (5) has been constructed so that choices for j, k are independent. So, the number of choices for j, k dominates $\mathrm{Bin}(n^2, w(n^2w^3/3)^2)$ which has expectation $\Omega(\log n)$ and so is non-zero **whp**.

This completes the analysis of `BDAPTA` when there are two levels.

3.4 General 3-Dimensional Version

We follow the same three phase strategy. k is a positive integer, $2 \leq k \leq \gamma \log\log n$.

3.4.1 Greedy Phase

This is much as before. Proceed as in Section 3.3.1 but taking $\theta = \theta_k$ (recall θ's definition from Theorem 2) and defining n_1 accordingly. Lemma 1 continues to hold.

3.4.2 Main Phase

Let

$$\alpha = 2^{-2k-2}\left(1 - \sqrt{2/3}\right)$$

and as in Section 3.3.2 let

$$\beta = 1 - \alpha, t_0 = \log_{1/\beta}(x_1/L) \text{ and } x_t = \beta^{t-1}x_1 = \beta^{t-1}n^{1-\theta}, t = 1, \ldots, t_0.$$

Let I_t, X_t, A_t have the same meaning as well. Now let

$$\gamma = 2^{k+1} - 2 \text{ and } w_0 = 2\gamma n^{-2(1-\theta)}\log n$$

and

$$w_t = 2\gamma x_t^{-1-\theta}n^{\theta-1}\log^\theta n \qquad \text{for } t \geq 1$$

and

$$W_t = w_0 + w_1 + \cdots + w_t = O(2^k n^{\theta-1}\log^\theta n).$$

The aim of round t is once again to add $x_t - x_{t+1}$ new indices to I_t using triples with (refreshed) cost at most w_t. We will assume that at the start of round t we have $T = \{(i, i, i) : 1 \leq i \leq n - x_t\}$. In analogy with (3), to add $i \in A_t$ to I_t we will add $2^{k+1} - 1$ triples to T and remove $2^{k+1} - 2$ triples, in which case the additional cost of the Main Phase will be at most $2^{k+1} - 1$ times

$$\sum_{t=1}^{t_0}(x_t - x_{t+1})W_t \leq x_1 w_0 + \sum_{t=1}^{t_0} x_t w_t$$

$$\leq 2\gamma n^{\theta-1}\log n + 2\gamma x_1^{-\theta}n^{\theta-1}\log^\theta n \sum_{t=1}^{t_0}\beta^{-\theta t} = O(2^k n^{\theta-1}\log n). \quad (6)$$

The notation used in (3) is obviously insufficient. We imagine a rooted tree Γ of triples of depth $2k$. The root will be $\rho = (i_0, j_0, k_0)$ where i_0 is the index to be added to I_t. The root is at level zero and there are $2k + 1$ levels $0, 1, \ldots, 2k$ in all. The triples at odd levels are to be deleted from T and the triples at even levels are to be added to T. We considered each level of Γ to be ordered so it makes sense to talk of the ith vertex of level $2l$ where $1 \leq i \leq 2^l$.

We partition $[\nu]$ into γ sets $\mathcal{Q} = \{Q_{i,l}, 1 \leq i \leq 2^l,\ 1 \leq l \leq k\}$ of (near) equal size. The ith triple at an odd level $2l - 1$, $l \geq 1$ will, by assumption, have the form (p, p, p) where $p \in Q_{i,l}$. This triple will have one child (p,a,b) which will replace the parent triple in 1-plane p. Here if $l<k$ then $a\in Q_{2i-1,l+1}, b\in Q_{2i,l+1}$ and if $l=k$ then $a,b\in X_t$.

A triple $u=(p,a,b)$ at an even level will have two children. By construction, (having made changes as we work down from the root), u will be the unique triple in 1-plane p, but now we will have two triples in 2-plane a and 3-plane b. Thus the children of u will be (a,a,a) and (b,b,b).

This construction defines a tree corresponding to adding $2^{k+1}-1$ and removing $2^{k+1}-2$ triples from T.

Our partition into $Q_{i,j}$'s ensures that adding the triples at even levels and deleting the triples at odd levels yields a partial assignment. Indeed, if $u=(p,a,b)$ is the ith triple at level $2l$, then the blocks of the partition $\mathcal{Q}$ containing p,a,b are uniquely defined by i,l. Thus no clashes are possible because of the additions of triples elsewhere in the tree.

We ensure that if $u=(p,a,b)$ is a triple at an even level then $C_{p,a,b}\leq w$. We call such a tree *feasible*.

We now have to show that **whp** there is always at least one such tree Γ for each $i\in A_t$. We take the same *bottom-up* approach that we did in Section 3.3. We fix t and drop the suffix t from all quantities that use it. We now estimate the number of choices for a $p\in Q_{1,k}$ that can be in a triple (p,x,y) at level $2k$. The number of choices for $p\in Q_{1,k}$ is distributed as the binomial $\mathrm{Bin}(\nu/\gamma,1-e^{-wx^2})=\mathrm{Bin}(\nu/\gamma,(1-o(1))wx^2)$. Note that $wx^2=\gamma(x/n)^{1-\theta}\log^\theta n=o(1)$ and that $wnx^2=\tilde{\Omega}(n^\theta)\gg\log n$. (Here our notation $f(n)\gg g(n)$ means that $f(n)/g(n)\to\infty$ with n). So the Chernoff bounds imply that **qs** we can choose a set P of size exactly $wnx^2/2\gamma=o(n)$, such that for each $p\in P$ there is at least one choice ξ_1,ξ_2 such that the triple (p,ξ_1,ξ_2) is *good*, i.e., $C_{p,\xi_1,\xi_2}\leq w$. We will in fact be able to choose 2^k disjoint sets $P_{i,k}\subseteq Q_{i,k}$.

Observation 5 *Each $\xi\in X$ is in fewer than $\mathrm{Bin}(x\nu,1-e^{-w})$ good triples of the form $(p\in P_{l,k},\xi,.)$ and so* **qs** *it is in at most $2wnx$ such triples. (Here $wnx=\Omega\left(\frac{n\log n}{x}\right)^\theta\gg\log n$).*

Let
$$\nu_0=\frac{wnx^2}{2\gamma}\text{ and }\nu_l=\frac{wn}{2\gamma}\nu_{l-1}^2\text{ for }1\leq l<k. \tag{7}$$
The solution to this recurrence is
$$\nu_l=\left(\frac{wn}{2\gamma}\right)^{2^{l+1}-1}x^{2^{l+1}}=(n\log n)^{(2^{l+1}-1)\theta}x^{(2^{k+1}-2^{l+1})\theta}.$$
Observe that ν_l increases with l. Note also that if $l\leq k-2$ then
$$w\nu_l^2\leq w\nu_{k-2}^2=2\gamma\left(\frac{x}{n}\right)^{2^k\theta}\log^{(2^k-1)\theta}n=o(1), \tag{8}$$
$$wn\nu_l\geq wn\nu_0=\frac{w^2n^2x^2}{\gamma}=2\gamma\left(\frac{n\log n}{x}\right)^{2\theta}\gg\log n. \tag{9}$$
Suppose now that $l\leq k-1$ and $1\leq r\leq 2^l$. We will prove by induction on l that **qs** there are at least ν_l choices for the rth triple $u=(p,a,b)$ at this level with the following two properties: (i) $C_u\leq w$ and (ii) there exists a feasible tree Γ_u with u as root and depth $2l+1$. Our analysis above has proved the base case of $l=0$. Imagine now that we are filling in the possibilities for the rth triple (p,a,b) at level $k-l$. (We fill in these possibilities level by level starting at level $2k$). Imagine also that we have identified ν_{l-1} choices for each of a,b. This follows from our inductive assumption, so for example a will have to be a possible selection for the first component of the rth triple at level $2(k-(l-1))$.

For a fixed $p \in Q_{i,l}$, conditional on our having selected exactly ν_{l-1} choices A, B for a, b, let p be *useful* if there is a pair $(a, b) \in A \times B$ with $C_{p,a,b} \le w$ and *useless* otherwise. Then, using (8),

$$\mathbf{Pr}(p \text{ is useless}) \le \exp\left\{-w\nu_{l-1}^2\right\} \le 1 - \frac{2w\nu_{l-1}^2}{3}.$$

It follows that the number of useful p dominates $(1 - o(1))\,\mathrm{Bin}(n/\gamma, 2w\nu_{l-1}^2/3)$. It follows that **qs** we can choose a set $P_{i,l} \subseteq Q_{i,l}$ of useful p's of size $\nu_l = wn\nu_{l-1}^2/2\gamma = o(n)$.

Observation 6 *A fixed a is in at most $\mathrm{Bin}(n\nu_{l-1}, 1 - e^{-w})$ good triples (p, a, b) feasible for level $2(k - l)$ and so* **qs** *every a is in at most $2wn\nu_{l-1}$ such triples, see* (9).

This completes our induction. We now apply the above to show that round t succeeds **whp**.

Suppose that in the middle of a round we have added $y < \alpha x$ triples to T. The number of $\xi \in X$ that can be used in a good triple (p, ξ, η) at level $2k$ will have been reduced by y. Thus the number of choices for p in any triple in this level will have been reduced by at most $2^k \times 2 \times \alpha x \times 2wnx$, see Observation 5. This reduces the number of choices for p in a triple at level $2(k-1)$ by at most $2^{k+2}\alpha wnx^2 \times 2wn\nu_0 = 2^{k+3}\alpha wn\nu_0^2$, see Observation 6. So let μ_l denote the number of choices for p in triples $p(,.,.)$ at level $2(k - l)$ that are forbidden by choices further down the tree. We have just argued that $\mu_1 \le 2^{k+3}\alpha wn\nu_0^2$. In general we can use Observation 6 to conservatively argue that

$$\mu_l \le 2wn\nu_{l-1}(\mu_{l-1} + 2^{k+1}\alpha x).$$

It follows that for $l \ge 2$ we have

$$\frac{\mu_l}{\nu_l} \le 2\frac{\mu_{l-1}}{\nu_{l-1}} + \frac{2^{k+2}\alpha x}{\nu_{l-1}} \le 2\frac{\mu_{l-1}}{\nu_{l-1}} + 2^{k+2}\alpha\left(\frac{x}{n\log n}\right)^{(2^l-1)\theta} \le 2\frac{\mu_{l-1}}{\nu_{l-1}} + 2^{k+2}\alpha\left(\frac{x}{n\log n}\right)^{\theta}.$$

It follows that

$$\frac{\mu_{k-1}}{\nu_{k-1}} \le 2^{k-2}\frac{\mu_1}{\nu_1} + 2^{2k+1}\alpha\left(\frac{x}{n\log n}\right)^{\theta} \le 2^{2k+2}\alpha.$$

We see that at the root there will still be at least $(1 - 2^{2k+2}\alpha)\nu_{k-1}$ choices for j_0, k_0. So i_0 will get a choice of at least $\mathrm{Bin}((1 - 2^{2k+2}\alpha)^2\nu_{k-1}^2, 1 - e^{-w})$ choices for a good triple (i_0, j_0, k_0). So the expected number of choices is at least $2w\nu_{k-1}^2/3$, our choice of α implies this. Now $w\nu_{k-1}^2 = 2\gamma\log n$ and this is sufficient to ensure that **whp** there is always at least one choice for every i_0.

3.4.3 Final Phase

We can execute the Main Phase so long as $x \ge 2^{k+1}$. Now assume that $1 \le x < 2^{k+1}$. We now have to add $\le 2^{k+1}$ indices to I. This time we refresh C at most 2^{k+1} times at an extra cost of $w_f = O\left(\frac{2^k\log^\theta n}{n^{1-\theta}}\right)$ each time we add an index. So, if successful, the cost of this round is

$$O(2^{2k}(W_{t_0} + w_f)) = O\left(\frac{2^{2k}\log^\theta n}{n^{1-\theta}}\right).$$

We will revert to a top-down appraoch as we did in Section 3.3.3. It will not be quite so straightforward to describe. The aim will be to build a feasible tree of triples Γ for each addition to I. Suppose that we have a partial assignment S of size at least $n - 2^{k+1}$ and that we wish to add index i to $I = I(S)$. As in Section 3.3.3, we are holding to the convention that all triples in S are of the form (j, j, j)). We will also use a partition of $[|S|]$ into γ sets $Q_{i,l}, 1 \leq i \leq 2^l$, $1 \leq l \leq k$ of (near) equal size. Once again, the ith triple at an odd level $2l - 1$, $l \geq 1$ will have the form (p, p, p) where $p \in Q_{i,l}$. This triple will have one child (p, a, b) which will replace the parent triple in 1-plane p. Here if $l < k$ then $a \in Q_{2i-1,l+1}, b \in Q_{2i,l+1}$ and if $l = k$ then $a, b \in X_t$.

To build Γ we will have to build it out of subtrees with suitable properties. A tree of depth $2l$ is built out of trees of depth $2(l - 1)$. We want to choose these trees so that the additional cost of these changes is small. We proceed inductively. We use the notation of (7) with $\gamma = 2^{k+2} - 2$ and $x = 1$.

Our inductive assumption is as follows: Suppose that C is unconditioned. Then for each $j \in [n] \setminus I(S)$ and $1 \leq l \leq k$ and $1 \leq i \leq 2^l$ we can in $O(n^{2l})$ time **qs** find a set $P \subseteq Q_{2i-1,l}$ of size ν_{l-1} and a collection $Q_p \subseteq Q_{2i,l}, p \in P$ of sets of size ν_{l-1} such that for each $p \in P, q \in Q_p$ there is an assignment S' with $(j, p, q) \in S'$ and $|I(S')| = |I(S)| + 1$ and $|S' \setminus S| \leq 2^{l+1} - 1$ and $C(S') = C(S) + C(j, p, q) + O(lw_f)$. Here S' arises from S through changes indicated by a tree of depth $2l$.

This is true for $l = k$ since we can make the changes

$$+(j, p, q) - (p, p, p) - (q, q, q) + (p, i_2, p) + (q, q, i_3)$$

where i_2, i_3 are unused 2- and 3-coordinates respectively. The number of choices for p, q constitute an independent pair of $(1 - o(1)) \operatorname{Bin}(n/\gamma, w)$ random variables and $\nu_0 = wn/2\gamma$.

For the inductive step, we first refresh the matrix C. Now fix $p \in Q_{1,l}, q \in Q_{2,l}$. We will check to see if it is suitable to have $p \in P, q \in Q_p$. First let $S' = S \setminus \{(p, p, p)\}$ and apply the induction hypothesis to generate ν_{l-2}^2 choices of assignment S_{p_1,p_2} containing the triple (p, p_1, p_2). For each such p_1, p_2 let $S'(p_1, p_2) = S(p_1, p_2) \setminus \{(q, q, q)\}$. Now apply the induction hypothesis to generate ν_{l-2}^2 choices of assignment $S_{p_1,p_2;q_1,q_2}$ containing the triples $(p, p_1, p_2), (q, q_1, q_2)$.

After this, the number of choices for p_1, p_2 where $C(p, p_1, p_2) \leq w$ is $(1 - o(1))B(n/\gamma, w\nu_{l-2})$ and this will **qs** be at least $\nu_{l-1} = wn\nu_{l-2}/2\gamma$. Similarly, for each such p_1, p_2 there will **qs** be at least ν_{l-1} choices for q, completing the induction. Notice that because these events occur **qs** we do not have to refresh C for every p_1, p_2 since we can use the union bound over instances p_1, p_2. We can build subtrees

Going back to level $l = 1$, it follows that we can **qs** find sets $P, Q_p, p \in P$ and trees of depth $2(l-1)$ below each p, q.. Now refresh C one more time. Let We have $(1 - o(1)) \operatorname{Bin}(\nu_{k-1}^2, w)$ choices of j, k which can be used to add $i \notin I$ to I at a cost of $O(w)$. In expectation this is $2 \log n$ and so we succeed **whp**.

3.4.4 Execution time

For the execution time of the algorithm we observe that to add a triple to I involves replacing $2^{k+1} - 2$ triples by $2^{k+1} - 1$ triples. This involves the construction of a tree T where for the $2^{k+1} - 1$ vertices that we add, we have n^2 choices to make, in the worst-case and for which the other choices are forced. This gives a running time of $O\left(n \times (n^2)^{2^{k+1}-1}\right)$ time.

This completes the proof of Theorem 2. □

3.5 Problems with $d > 3$

Given the expansion of the tree in Figure 1 it is perhaps surprising that this approach works, but for $d = 3$ it does. However, for $d > 3$, a more general approach fails. For any $d \geq 2$, our algorithm iterates a restricted form of local augmentation. (The main technical aspect of our result is restricting the augmentation so as to control the conditioning of the random cost matrix.) The last iteration can without loss of generality be thought of as replacing the k selected d-tples, $\{(i,i,\ldots,i), 1 \le i \le k\}$, with $k+1$ d-tples from $[k+1]^d$. When k is a constant, the probability that there is any new such assignment of cost $\le w$ (even without the restrictions our algorithm imposes) is at most $\binom{n}{k}(k!)^{d-1} \cdot w^k = O((nw)^k)$, so for the algorithm to have any hope of succeeding, for the last iteration alone we must budget a cost contribution $w = \Omega(n^{-1})$. That proves to be satisfactory (even taking all the iterations into account) for 2-dimensional assignment where the total cost we are aiming for is $\Theta(1)$, and for 3-dimensional assignment with its lower bound of $\Omega(n^{-1})$, but it is plainly unsatisfactory for dimension 4, where the lower bound (which we imagine is close to the truth) is $O(n^{-(d-2)}) = O(n^{-2})$.

4 Multi-Dimensional Planar Version

4.1 Main theorem

Here we give our main theorem for Planar assignment.

Theorem 3 *The optimal solution value $Z_{d,n}^A$ satisfies the following:*

(a) $Z_{d,n}^A = \Omega(n^{d-2})$ **whp** *for $d \ge 3$.*

(b) *When $d = 3$ there is a polynomial time algorithm that finds a solution with cost Z where $Z = O(n \log n)$* **whp**.

The theorem is proved in the next two sections.

4.2 Lower bound

It is clear that $Z_{d,n}^A \ge Z_1 + Z_2 + \cdots + Z_{n^{d-2}}$ where Z_i is the minimum cost of the 2-dimensional assignment with cost matrix $A_{j,k} = C_{i_1,\ldots,i_{d-2},j,k}$. We know that $Z_j \ge (1-o(1))\zeta(2)$ **whp** and the Z_i's are independent. It follows that **whp** $Z_{3,n}^A \ge (1-o(1))n^{d-2}\zeta(2) > 3n^{d-2}/2$.

4.3 Upper bound for $d = 3$

For the upper bound we need a result of Dyer, Frieze and McDiarmid [DFM86]. We will not state it in full generality, instead we will tailor its statement to precisely what is needed. Suppose that we have a linear program

$$P: \qquad \text{Minimize } c^T x \text{ subject to } Ax = b,\ x \ge 0.$$

Here A is an $m \times n$ matrix and the cost vector $c = (c_1, c_2, \ldots, c_n)$ is a sequence of independent copies of Exp(1). Let Z_P denote the minimum of this linear program. Note that Z_P is a random variable. Next let y be *any* feasible solution to P.

Theorem 4 ([DFM86])

$$\mathbf{E}(Z_P) \le m \max_{j=1,2,\ldots,n} y_j. \tag{10}$$

Furthermore, Z_P is at most $1 + o(1)$ times the RHS of (10), **whp**.

Now consider the following greedy-type algorithm. We find a minimum 2-dimensional assignment for 1-plane $i = 1$, we then find a minimum assignment for 1-plane $i = 2$, consistent with choice for 1-plane $i = 1$, and so on:
Greedy

1. For $i = 1, \ldots, n$ do the following:
   - Let $G = K_{n,n} \setminus (M_1 \cup M_2 \cup \cdots M_{i-1})$;
   - If $(j, k) \in E(G)$ let $A_{j,k} = C_{i,j,k}$.
   - Let M_i be a minimum cost matching of G using edge weights A.

The output, $M_1, M_2, \ldots, M_n$ defines a set of triples $T = \{(i, j, k) : (j, k) \in M_i\}$. We claim that if $Z_i = A(M_i)$ then

$$\mathbf{E}(Z_i) \leq \frac{2n}{n - i + 1}. \tag{11}$$

For this we apply Theorem 4 to the following linear program which we note always has an integer optimum solution,

$$\begin{aligned}
\text{Minimize} \sum_{(j,k) \in E(G)} A_{j,k} x_{j,k} \quad & \text{subject to} \\
\sum_{k:\, (j,k) \in E(G)} x_{j,k} = 1, \quad & j = 1, 2, \ldots, n \\
\sum_{j:\, (j,k) \in E(G)} x_{j,k} = 1, \quad & k = 1, 2, \ldots, n \\
x_{j,k} \geq 0, \quad & j, k = 1, 2, \ldots, n.
\end{aligned}$$

We note that there are $2n$ constraints and that $x_{j,k} = 1/(n - i + 1)$ is a feasible solution. With Theorem 4, this implies (11) and the upper bound in Theorem 3 for the case $d = 3$.

4.4 Higher dimensions

As noted in the introductory summary of results, this approach appears to fail for dimensions $d \geq 4$: it can generate 3-dimensional instances with no assignment at all. Perhaps this does not happen in a typical case, but a small example is shown below, for a 4-dimensional instance of size 4 (so, with 4^4 elements). Suppose that in the first 3-dimensional sub-array considered (shown as four 4×4 2-dimensional arrays), the elements selected are those indicated by 1s in the table below; this will happen, for example, if the corresponding input elements have tiny cost and the others have large cost.

```
0 1 0 0   0 0 0 1   0 0 1 0   1 0 0 0
0 0 1 0   1 0 0 0   0 1 0 0   0 0 0 1
1 0 0 0   0 1 0 0   0 0 0 1   0 0 1 0
0 0 0 1   0 0 1 0   1 0 0 0   0 1 0 0
```

Then, in the second 3-dimensional sub-array, perhaps the following selection gives the cheapest assignment. Again, this may occur: it is not blocked by the first sub-array assignment.

```
0 0 1 0   0 1 0 0   0 0 0 1   0 0 0 0
0 1 0 0   0 0 1 0   1 0 0 0   0 0 0 0
0 0 0 1   1 0 0 0   0 1 0 0   0 0 0 0
1 0 0 0   0 0 0 1   0 0 1 0   0 0 0 0
```

At this point, there is no feasible assignment for the third sub-array.

5 Conclusions

For the 2-dimensional random assignment problem, we know the limiting expected cost, and a given instance can be solved in polynomial time. As noted in the Introduction, much less is known about multidimensional assignment problems, and as far as we are aware, nothing was known about polynomial-time algorithms solving these problems well on average. For the 3-dimensional Planar assignment problem, the present paper is the first to prove an upper bound within a logarithmic factor of the obvious $\Omega(n)$ lower bound (likely the true answer), doing so by analyzing a simple and fast greedy algorithm. For the 3-dimensional Axial assignment problem, we give the first upper bounds within polylogarithmic factors of the obvious $\Omega(1/n)$ lower bound. One of our upper bounds is a trivial application of a result of Johansson, Kahn and Vu [JKV08]; we cannot really take credit for it. The second, however, comes from analyzing an algorithm that is the first that solves this problem well, on average. For any $\varepsilon \geq (\log n)^{-\gamma}$, for any constant $\gamma < 1/2$, if we run our algorithm for $O(n^{2+2/\varepsilon})$) time then **whp** the algorithm delivers a solution within a factor $O(n^{\varepsilon} \log^2 n)$ of the expected minimum; we know of no other such approximation. As discussed in Section 2, our results do not extend to $d \geq 4$.

We are left with open questions including these:

P1 What are the growth rates of $\mathbf{E}[Z^P_{d,n}]$ and $\mathbf{E}[Z^A_{d,n}]$ for $d \geq 3$?

P2 Are there asymptotically optimal, polynomial time algorithms for solving these problems when $d \geq 3$?

P3 For $d > 3$, are there polynomial time algorithms yielding solutions within logarithmic or $O(n^e)$ factors for Planar and Axial assignment problems (as we have given for $d = 3$)?

P4 Frieze [Fri74] gave a bilinear programming formulation of the 3-dimensional Axial problem. There is a natural heuristic associated with this formulation (see Appendix). What are its asymptotic properties?

References

[Ald92] David Aldous, *Asymptotics in the random assignment problem*, Probab. Theory Related Fields **93** (1992), no. 4, 507–534.

[Ald01] David J. Aldous, *The $\zeta(2)$ limit in the random assignment problem*, Random Structures Algorithms **18** (2001), no. 4, 381–418.

[1] M. Buck, C. Chan and D. Robbins, *On the expected value of the minimum assignment*, Random Structures Algorithms **21** (2002), no. 2, 295–301.

[BDM09] Rainer Burkard, Mauro Dell'Amico, and Silvano Martello, *Assignment problems*, Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2009.

[CS99] Don Coppersmith and Gregory B. Sorkin, *Constructive bounds and exact expectations for the random assignment problem*, Random Structures Algorithms **15** (1999), no. 2, 113–144.

[2] S. Dichkovskaya and M. Kravtsov, *Investigation of polynomial algorithms for solving the three-index palanar assignment problem*, Computational Mathematics and Mathematical Physics **46** (2006) 212–217.

[Don69] W.E. Donath, *Algorithm and average-value bounds for assignment problems*, IBM Journal of Research and Development **13** (1969), no. 4, 380–386.

[DFM86] Martin E. Dyer, Alan M. Frieze, and Colin J.H. McDiarmid, *On linear programs with random costs*, Mathematical Programming **35** (1986), 3–16.

[Fri74] A. M. Frieze, *A bilinear programming formulation of the* 3*-dimensional assignment problem*, Math. Programming **7** (1974), 376–379.

[Fri83] ______, *Complexity of a* 3*-dimensional assignment problem*, European J. Oper. Res. **13** (1983), no. 2, 161–164.

[Fri85] ______, *On the value of a random minimum spanning tree problem*, Discrete Appl. Math. **10** (1985), no. 1, 47–56.

[FY81] A.M. Frieze and J. Yadegar, *An algorithm for solving 3-dimensional assignment problems with application to scheduling a teaching practice*, Journal of the Operational Research Society (1981), no. 32, 989–995.

[GOPP05] P. Krokhmal, D. Grundel and P. Pardalos, *Asymptotic behavior of the expected value of the random multidimensional assignment problems*, Mathematical Programming Series B **109** (2007) 525-551.

[JKV08] Anders Johansson, Jeff Kahn, and Van Vu, *Factors in random graphs*, Random Structures Algorithms **33** (2008), no. 1, 1–28.

[Kar72] Richard M. Karp, *Reducibility among combinatorial problems*, Complexity of computer computations (Proc. Sympos., IBM Thomas J. Watson Res. Center, Yorktown Heights, N.Y., 1972), Plenum, New York, 1972, pp. 85–103.

[3] V. Kratsov, *Polynomial algorithms for finding the asymptoticall optimal plan of the multiindex asignment problem*, Cybernetics and Systems Analysis, **41** no. 6 (2005) 940–944.

[LW04] Svante Linusson and Johan Wästlund, *A proof of Parisi's conjecture on the random assignment problem*, Probab. Theory Related Fields **128** (2004), no. 3, 419–440.

[MMR04] O. C. Martin, M. Mezard, and O. Rivoire, *A frozen glass phase in the multi-index matching problem.*

[MMR05] O. C. Martin, M. Mézard, and O. Rivoire, *Random multi-index matching problems*, J. Stat. Mech. Theory Exp. (2005), no. 9, P09006, 36 pp. (electronic).

[MP85] M. Mézard and G. Parisi, *Replicas and optimization*, J. Physique Lettres **46** (1985), 771–778.

[MP87] ______, *On the solution of the random link matching problems*, J. Physique Lettres **48** (1987), 1451–1459.

[NPS05] Chandra Nair, Balaji Prabhakar, and Mayank Sharma, *Proofs of the Parisi and Coppersmith-Sorkin random assignment conjectures*, Random Structures Algorithms **27** (2005), no. 4, 413–444. MR MR2178256 (2006e:90050)

[Par98] Giorgio Parisi, *A conjecture on random bipartite matching*, Physics e-Print archive, http://xxx.lanl.gov/ps/cond-mat/9801176, January 1998.

[4] Johan Wästlund, *A Proof of a Conjecture of Buck, Chan and Robbins on the Expected Value of the Minimum Assignment*, Random Structures and Algorithms **26** 1-2 (2005), 237-251

[Wäs09] Johan Wästlund, *An easy proof of the zeta(2) limit in the random assignment problem*, Electronic Communications in Probability (2009), no. 14, 261–269.

A Bilinear Programming Formulation

Frieze [Fri74] re-formulated the 3-dimensional Axial problem as

$$\text{Minimize} \sum_{i,j,k=1}^{n} C_{i,j,k} y_{i,j} z_{i,k} \text{ subject to } x, y \in P_A$$

where P_A is the bipartite matching polyhedron $\sum_{i=1}^{n} x_{i,j} = 1 = \sum_{j=1}^{n} x_{i,j}$, for all $1 \leq i, j \leq n$.

Now denote the objective above by $C(y, z)$. The following heuristic was used successfully in a practical situation [FY81]:

1. Choose y_0, z_0 arbitrarily; $Z_0 = C(y_0, z_0)$; $i = 0$.
2. Repeat until $Z_{i+1} = Z_i$.
   - Let y_{i+1} maximize $C(y, z_i)$.
   - Let z_{i+1} maximize $C(y_{i+1}, z)$.
   - $Z_{i+1} = C(y_{i+1}, z_{i+1})$.
   - $i = i + 1$.