Channel Assignment via Fast Zeta Transform

Marek Cygan and Łukasz Kowalik
Institute of Informatics, University of Warsaw
{cygan,kowalik}@mimuw.edu.pl

Abstract

We show an $O^*((\ell+1)^n)$-time algorithm for the channel assignment problem, where $\ell$ is the maximum edge weight. This improves on the previous $O^*((\ell+2)^n)$-time algorithm by Kral [4], as well as algorithms for important special cases, like $L(2,1)$-labelling. For the latter problem, our algorithm works in $O^*(3^n)$ time. The progress is achieved by applying the fast zeta transform in combination with the inclusion-exclusion principle.

1 Introduction

In the channel assignment problem, we are given a symmetric weight function $w : V^2 \to \mathbb{N}$ (we assume that $0 \in \mathbb{N}$). The elements of $V$ will be called vertices (as $w$ induces a graph on the vertex set $V$ with edges corresponding to positive values of $w$). We say that $w$ is $\ell$-bounded when for every $x, y \in V$ we have $w(x, y) \leq \ell$. An assignment $c : V \to \{1, \ldots, s\}$ is called proper when for each pair of vertices $x, y$ we have $|c(x) - c(y)| \geq w(x, y)$. The number $s$ is called the span of $c$. The goal is to find a proper assignment of minimum span. Note that the special case when $w$ is 1-bounded corresponds to the classical graph coloring problem.

In this paper we deal with exact algorithms for the channel assignment problem. As a generalization of graph coloring, the decision version of channel assignment is NP-complete. It follows that the existence of a polynomial-time algorithm is unlikely. As a consequence, researchers began to study exponential-time algorithms for the channel assignment problem. The asymptotic efficiency of these algorithms is measured in terms of $n = |V|$ and $\ell$, we assume that $\ell$ is a constant. The first non-trivial algorithm was proposed by McDiarmid [5] and had running time of $O(n^2(2\ell+1)^n)$. It was then improved by Kral [4] to $O(n(\ell+2)^n)$.

Here we improve the running time further to $O^*((\ell+1)^n)$. We also show that the number of all proper assignments can be found in the same time bound. Note that for $\ell = 1$ the running time of our algorithm matches

\footnote{By $O^*(\cdot)$ we suppress polynomially bounded terms.}
the time complexity of the currently fastest algorithm for graph coloring by Björklund, Husfeldt and Koivisto [1].

Our improvement is achieved by applying the fast zeta transform in combination with the inclusion-exclusion principle. The same ingredients were used also in a set partition problem in [1], however in our algorithm the fast zeta transform plays a different role. In particular, although channel assignment resembles a kind of set partition it does not seem to be possible to solve it by a direct application of the algorithm from [1].

Some special cases of the channel assignment problem received particular attention. An important example is the $L(p,q)$-labeling of graphs, where given an undirected graph $G = (V,E)$ one has to find an assignment $c : V \to \mathbb{N}$ such that if vertices $u$ and $v$ are adjacent then $|c(u) - c(v)| \geq p$ and if vertices $u$ and $v$ are at distance 2 then $|c(u) - c(v)| \geq q$. The goal is to minimize $\max_{v \in V} c(v)$. Clearly, the algorithmic problem of finding an $L(p,q)$-labeling reduces in polynomial time to the $\max\{p,q\}$-bounded channel assignment and we get an $O^*((\max\{p,q\} + 1)^n)$-time algorithm as an immediate corollary from our result. In particular, it gives an $O^*(3^n)$-time algorithm for the most researched subcase of $L(2,1)$-labeling. This improves over the algorithms by Havet et al. [2] running in time $O(3.873^n)$ and a recent improvement of Junosza-Szaniawski and Rzążewski [3] running in $O(3.562^n)$ time.

2 Deciding

In this section we consider the decision version of the problem, i.e. for a given $\ell$-bounded weight function $w$ and an integer $s \in \mathbb{N}$ we check whether there is a proper assignment of span at most $s$. Since the case $\ell = 1$ can be solved in $O^*((\ell + 1)^n) = O^*(2^n)$ time as described in [1], here we assume $\ell \geq 2$.

An assignment $c : V \to \mathbb{N}$ of span $s$ can be seen as a tuple $(I_1, I_2, \ldots, I_s)$, where $I_j = c^{-1}(j)$ for every $j = 1, \ldots, s$. We will relax the notion of assignment in that we will work with tuples of vertex sets $(I_1, I_2, \ldots, I_k)$, where the $I_j$'s are not necessarily disjoint. We say that a tuple $(I_1, I_2, \ldots, I_k)$ is proper, when for every $i, j \in \{1, \ldots, k\}$ if $x \in I_i$ and $y \in I_j$ then $|i - j| \geq w(x,y)$.

In what follows, $U$ denotes the set of all proper tuples $(I_1, \ldots, I_s)$ such that for each $j = 1, \ldots, s - \ell + 1$, the sets $I_j, I_{j+1}, \ldots, I_{j+\ell-1}$ are pairwise disjoint. A tuple with the last elements being empty sets is denoted as $(I_1, \ldots, I_{s-r}, \emptyset^r)$. For a subset $X \subseteq V$, we say that a tuple $(I_1, \ldots, I_j)$ lies in $X$ when for every $i = 1, \ldots, j$, we have $I_i \subseteq X$.

For $v \in V$, define $U_v = \{(I_1, \ldots, I_s) \in U : v \in \bigcup_{j=1}^s I_j\}$. Observe, that

**Proposition 1.** $|\bigcap_{v \in V} U_v| > 0$ iff there is a proper assignment of span $s$.

By the inclusion-exclusion principle, if we denote $\overline{U}_v = U - U_v$ and
\[ \bigcap_{v \in \emptyset} U_v = U, \text{ then} \]
\[ | \bigcap_{v \in V} U_v | = \sum_{Y \subseteq V} (-1)^{|Y|} | \bigcap_{v \in Y} U_v|. \quad (1) \]

Our algorithm computes \(| \bigcap_{v \in Y} U_v |\) using the above formula. The rest of the section is devoted to computing \(| \bigcap_{v \in Y} U_v |\) for a given \(Y \subseteq V\). If we denote \(X = V - Y\), then \(\bigcap_{v \in Y} U_v\) is just the set of tuples of \(U\) that lie in \(X\):
\[ \bigcap_{v \in Y} U_v = \{(I_1, \ldots , I_s) \in U : I_1, \ldots , I_s \subseteq X\}. \quad (2) \]

Our plan now is to compute the value of \(| \bigcap_{v \in Y} U_v |\) using dynamic programming accelerated by the fast zeta transform. More precisely, for every \(i = \ell - 1, \ldots , s\) and for every sequence \(J_1, \ldots , J_{\ell - 1}\) of pairwise disjoint subsets of \(X\) our algorithm computes the value of

\[ T^X_i (J_1, \ldots , J_{\ell - 1}) = |\{(I_1, \ldots , I_{i-(\ell - 1)}, J_1, \ldots , J_{\ell - 1}, \emptyset^{s-i}) \in U : \bigcup_{j=1}^{i-(\ell - 1)} I_j \subseteq X\}|, \quad (3) \]

that is, the number of tuples in \(U\) that lie in \(X\) and end with \(J_1, \ldots , J_{\ell - 1}\) followed by \(s - i\) empty sets. Then, clearly,
\[ \bigcap_{v \in Y} U_v = \sum_{J_1, \ldots , J_{\ell - 1} \subseteq X} T^X_{s-i} (J_1, \ldots , J_{\ell - 1}). \quad (4) \]

For every sequence of pairwise disjoint sets \(J_1, \ldots , J_{\ell - 1} \subseteq X\), we can initialize the value of \(T^X_{i-1} (J_1, \ldots , J_{\ell - 1})\) in polynomial time as follows:
\[ T^X_{i-1} (J_1, \ldots , J_{\ell - 1}) = [(J_1, \ldots , J_{\ell - 1}) \text{ is proper}]. \quad (5) \]

Then the algorithm finds the values of \(T^X_{j}\) for subsequent \(j = \ell , \ldots , s\). This is realized using the following formula:
\[ T^X_{j} (J_1, \ldots , J_{\ell - 1}) = [(J_1, \ldots , J_{\ell - 1}) \text{ is proper}]: \sum_{J_0 \subseteq X \cap \text{proper}(J_1, \ldots , J_{\ell - 1})} T^X_{j-1} (J_0, J_1, \ldots , J_{\ell - 2}), \quad (6) \]
where \(\text{proper}(J_1, \ldots , J_{\ell - 1})\) is the set of all vertices \(v \in V \setminus \bigcup_{j=1}^{\ell - 1} J_j\) such that for each \(j = 1, \ldots , \ell - 1\) and \(x \in J_j\) we have \(j \geq w(v, x)\).

Using the formula (6) explicitly, one can compute all the values of \(T^X_{j}\) from the values of \(T^X_{i-1}\) in \(O^*((\ell + 1)^{|X|})\) time, since there are \((\ell + 1)^{|X|}\) tuples \((J_0, \ldots , J_{\ell - 1})\) of disjoint subsets of \(X\). Now we describe how to speed it up to \(O^*(|X|)\).
Let \( S \) be a set and let \( f : 2^S \to \mathbb{Z} \) be a function on the lattice of all subsets of \( S \). The zeta transform is an operator which transforms \( f \) to another function \( (\zeta f) : 2^S \to \mathbb{Z} \) and it is defined as follows:

\[
(\zeta f)(Q) = \sum_{R \subseteq Q} f(R).
\]

A nice feature of the zeta transform is that given \( f \) (i.e. when the value of \( f(R) \) can be accessed in \( O(1) \) time for any \( R \)) there is an algorithm (called fast zeta transform or Yates’ algorithm, see [1, 7]) which computes \( \zeta f \) (i.e. the values of \( (\zeta f)(Q) \) for all subsets \( Q \subseteq S \)) using only \( O(2^{|S|}) \) arithmetic operations (additions).

Let us come back to our algorithm. In the faster version, for each \( i = \ell, \ldots, s \), we iterate over all sequences of disjoint subsets \( J_1, \ldots, J_{\ell-2} \subseteq X \). Then the values of \( T^X_{i-1}(J_1, \ldots, J_{\ell-1}) \) for all the \( 2^{|X|}-\sum_{j=1}^{\ell-2} |J_j| \) sets \( J_{\ell-1} \) that are disjoint with \( J_1, \ldots, J_{\ell-2} \) are computed in \( O^*(2^{|X|}-\sum_{j=1}^{\ell-2} |J_j|) \) time (that is in polynomial time per set!). To this end, we use the function \( f : 2^X \setminus \bigcup_{j=1}^{\ell-2} J_j \to \mathbb{Z} \), where

\[
f(S) = T^X_{i-1}(S, J_1, \ldots, J_{\ell-2}).
\]

We compute the function \( (\zeta f) \) with the fast zeta transform using \( O(2^{|X|}-\sum_{j=1}^{\ell-2} |J_j|) \) additions. Now, observe that by (1), for each \( J_{\ell-1} \subseteq X \) disjoint with \( J_1, \ldots, J_{\ell-2} \),

\[
T^X_{i-1}(J_1, \ldots, J_{\ell-1}) = [(J_1, \ldots, J_{\ell-1}) \text{ is proper}] \cdot (\zeta f)(X \cap \text{proper}(J_1, \ldots, J_{\ell-1})).
\]

It follows that for each \( i = \ell, \ldots, s \) the algorithm runs in time needed to perform the following number of additions:

\[
O\left( \sum_{J_1, \ldots, J_{\ell-2} \subseteq X \atop j \neq k \Rightarrow J_j \cap J_k = \emptyset} 2^{|X|}-\sum_{j=1}^{\ell-2} |J_j| \right) = O\left( \sum_{J_1, \ldots, J_{\ell-1} \subseteq X \atop j \neq k \Rightarrow J_j \cap J_k = \emptyset} 1 \right) = O(\ell^{|X|}). \tag{7}
\]

By (1) it follows that the whole decision algorithm runs in time needed to perform \( O(n(\ell + 1)^n) \) additions. The numbers being added are bounded by \( |\bigcap_{v \in Y} U_v| \leq 2^{ns} \leq 2^{ns\ell} \), where the last inequality follows from the fact that the minimum span is upper bounded by \( (n-1)\ell + 1 \) (see e.g. [3]). Hence a single addition is performed in \( O(n^2\ell) \) time.

**Corollary 2.** There is an algorithm which verifies whether the minimum span of an \( \ell \)-bounded instance of the channel assignment problem is bounded by \( s \) which uses \( O^*((\ell + 1)^n) \) time and \( O^*(\ell^n) \) space.
3 Counting

In this section we briefly describe how to modify the decision algorithm from Section 2 in order to make it count the number of proper assignments of span at most $s$. We follow the approach of Björklund et al. [1]. The trick is to modify the definition of $U$. Namely, now every tuple $(I_1, \ldots, I_s)$ from $U_v$ additionally satisfies the following condition:

$$\sum_{j=1}^{s} |I_j| = n. \tag{8}$$

Observe, that then $|\bigcap_{v \in V} U_v|$ equals the number of proper assignments of span at most $s$. Now, we add another dimension to the arrays $T^X$: $T^X_{i,k}(J_1, \ldots, J_{\ell-1}) = |\{(I_1, \ldots, I_{i-(\ell-1)}, J_1, \ldots, J_{\ell-1}, \emptyset^{s-i}) \in U : \bigcup_{j=1}^{i-(\ell-1)} I_j \subseteq X \text{ and } \sum_{j=1}^{i-(\ell-1)} |I_j| + \sum_{j=1}^{\ell-1} |J_j| = k\}|.

The dynamic programming algorithm from Section 2 can be easily modified to compute the values of $T^X_{i,k}(J_1, \ldots, J_{\ell-1})$ for all $i = \ell-1, \ldots, s$, $k = 0, \ldots, n$ and all sequences of $\ell-1$ pairwise disjoint subsets of $X$. The details are left to the reader.

Corollary 3. For any $\ell$-bounded instance of the channel assignment problem the number of the proper assignments of span at most $s$ can be computed in $O^*((\ell + 1)n)$ time and $O^*(\ell^n)$ space.

4 Finding

In order to find the assignment itself we can solve the extended version of the channel assignment problem, where we are additionally given a set of vertices $Z \subseteq V$ together with a function $c' : Z \to \{1, \ldots, s\}$. Then we are to check whether there exists a proper assignment $c : V \to \{1, \ldots, s\}$ satisfying $c|_Z = c'$. It is not hard to modify the presented algorithm to solve the extended version of the problem in $O^*((\ell + 1)^{n-|Z|})$ time. The details are left to the reader.

Now using the extended version of the channel assignment problem we can take any $v \in V \setminus Z$ and try each of the $s \leq (n-1)\ell + 1$ possible values of $c(v)$ one by one, each time using the algorithm for the extended channel assignment problem as a black box. When the value for $v$ is fixed in a similar manner we assign the value for the other vertices of $V \setminus Z$. Since $\sum_{i=1}^{n}(\ell + 1)^{n-i} < (\ell + 1)^n$, the algorithm for finding an assignment has a multiplicative overhead of $O(n\ell)$ over the running time of the decision version.
5 Open problems

In [6] Traxler has shown that for any constant $c$, the Constraint Satisfaction Problem (CSP) has no $O(c^n)$-time algorithm, assuming the Exponential Time Hypothesis (ETH). More precisely, he shows that ETH implies that CSP requires $d^{Ω(n)}$ time, where $d$ is the domain size. On the other hand, graph coloring, which is a variant of CSP with unbounded domain, admits a $O^*(2^n)$-time algorithm. The channel assignment problem is a generalization of graph coloring and a special case of CSP. In that context, the central open problem in the complexity of the channel assignment problem is to find a $O^*(c^n)$-time algorithm for a constant $c$ independent of $\ell$ or to show that such the algorithm does not exist, assuming ETH (or other well-established complexity conjecture).

References

[1] A. Björklund, T. Husfeldt, and M. Koivisto. Set Partitioning via Inclusion-Exclusion. *SIAM J. Comput.*, 39(2):546–563, 2009.

[2] F. Havet, M. Klazar, J. Kratochvíl, D. Kratsch, and M. Liedloff. Exact Algorithms for L(2, 1)-Labeling of Graphs. *Algorithmica*, 59(2):169–194, 2011.

[3] K. Junosza-Szaniawski and P. Rzążewski. On Improved Exact Algorithms for L(2,1)-Labeling of Graphs. In *Proc. IWOCA 2010*, LNCS 6460, pages 34–37, 2010.

[4] D. Král. An exact algorithm for the channel assignment problem. *Discrete Applied Mathematics*, 145(2):326–331, 2005.

[5] C. J. H. McDiarmid. On the span in channel assignment problems: bounds, computing and counting. *Discrete Mathematics*, 266(1-3):387–397, 2003.

[6] P. Traxler. The Time Complexity of Constraint Satisfaction. In *Proc. IWPEC 2008*, LNCS 5018, pages 190–201, 2008.

[7] F. Yates. The Design and Analysis of Factorial Experiments. *Imperial Bureau of Soil Sciences, Harpenden*, 1937.