Lecture 8 (10.05.2013) Scribe: Tomasz Kociumaka

Lecturer: Marek Cygan

Kernelization by matroids: Odd Cycle Transversal

1 Introduction

The main aim of this lecture is to give a polynomial kernel for the Odd Cycle Transversal problem parameterized by k. Recall the problem formulation:

Odd Cycle Transversal
Input: A graph $G = (V, E)$, a positive integer k
Problem: Does there exist $X \subseteq V$ of size $\leq k$ such that $G \setminus X$ is bipartite?

Our strategy is going to be different than previously. We shall *compress* Odd Cycle Transversal to a language $L \in NP$. Then, NP-completeness gives a polynomial reduction back to Odd Cycle Transversal. Recall the formal definition of compression.

Definition 1. Let $L_1 \subseteq \Sigma^* \times \mathbb{N}$ be a parameterized language and $L_2 \subseteq \Sigma^*$ be a language. A mapping $f : \Sigma^* \times \mathbb{N} \to \Sigma^*$ computable in polynomial time (with respect to inputs size) is called *a compression* if $(I, k) \in L_1 \iff f(I, k) \in L_2$. A compression is *polynomial* if $|f(I, k)| \leq k^{\mathcal{O}(1)}$.

Note that language L_2 in the definition of compression might be arbitrary, possibly undecidable. Kernelization can be viewed as special case of compression, where L_2 is equal to (the non-parameterized variant of) L_1.

In the following sections we give a polynomial compression of the Odd Cycle Transversal problem. We present an FPT algorithm, interpret it as a series of $f(k)$ min-cut queries and finally use matroid tools to encode the graph in $k^{\mathcal{O}(1)}$ bits so that we can still answer all relevant min-cut queries.

2 FPT algorithm for Odd Cycle Transversal

We present an $\mathcal{O}^\star(3^k)$ algorithm using *iterative compression*. This technique lets us reduce our problem to developing an $\mathcal{O}^\star(3^{|Y|})$ algorithm for the following problem:

OCT Compression
Input: A graph $G = (V, E)$, a positive integer k and a set $Y \subseteq V$ such that $G \setminus Y$ is bipartite.
Problem: Find a minimum $X \subseteq V$ such that $G \setminus X$ is bipartite.

Let us recall how iterative compression works. For more details and applications see Lecture 2 of Parameterized and Moderately Exponential Algorithms course `http://www.mimuw.edu.pl/~malcin/dydaktyka/2012-13/fpt/fpt_02_minors_iterative_compression.pdf`. Let $V = \{v_1, \ldots, v_n\}$ be the vertices of G and let $G_i = G[\{v_1, \ldots, v_i\}]$. The algorithm below solves standard Odd Cycle Transversal using $\mathcal{O}(n)$ calls to the subroutine `OCTCompression` solving the compression variant.

Observe that the algorithm is correct, because if X is a solution in G (i.e. $G \setminus X$ is bipartite), then $X \cap \{x_1, \ldots, x_k\}$ is a solution in G_i, and if X_i is a solution in G_i, then $X_i \cup \{v_{i+1}\}$ is a solution in G_{i+1}.

Algorithm 1: ODD CYCLE TRANVERSAL($G = (\{v_1, \ldots, v_n\}, E), k$)

$X_0 = \emptyset$;
for $i := 1$ ***to*** n **do**
  $Y_i := X_{i-1} \cup \{v_i\}$;
  $X_i :=$`OCTCompression`(G_i, Y_i, k);
  if $|X_i| > k$ **then return** NO;
return YES;

2.1 An $\mathcal{O}^\star(3^k)$ algorithm for OCT Compression

Note that any solution X induces a (not necessarily unique) partition of $G \setminus X$ into two stable sets V_L and V_R. In particular, Y is divided into $T = Y \cap X$, $L = Y \cap V_L$ and $R = Y \cap V_R$. The algorithm iterates over all partitions of Y into three sets T, L, R. Let us fix a single partition. Observe that Y_L and Y_R can be assumed to be stable, while Y_T can be already removed from the graph.

Since $G \setminus Y$ is bipartite, this gives a graph G' with a partition of $V(G')$ into four stable sets A, B, L, R. Now, it suffices to solve the following problem.

Auxiliary problem
Input: A graph $G = (V, E)$ with V partitioned into four stable sets A, B, L, R
Problem: Find the smallest $X \subseteq A \cup B$ such that $V \setminus X$ can be partitioned into stable sets V_L, V_R with $L \subseteq V_L$, $R \subseteq V_R$.

We shall see that the auxiliary problem can be solved in polynomial time. First, observe that it is equivalent to finding minimal X, which hits:

- all odd L to L and R to R walks,
- all even L to R walks.

Note, that if $G \setminus X$ does not have such walks, then we can set V_L as the set of vertices connected (in $G \setminus X$) to L by an even walk, or to R by an odd walk or disconnected both from L and R and belonging to A. Simultaneously V_R can be defined symmetrically as the set of vertices connected to R with an even walk, or to L an with odd walk or disconnected both from L and R and belonging to B.

Let $A_L = N(L) \cap A$ and analogously define A_R, B_L, B_R. Observe that if $X \subseteq A \cup B$ is a solution, then X must hit the following types of walk in $G[A \cup B]$

- from A_L to B_L,
- from A_L to A_R,
- from A_R to B_R,
- from B_L to B_R.

For example, in $G[A \cup B]$ any walk from A_L to B_L is odd. Extending such a walk by elements of L gives an odd walk from L to L. By similar arguments X must hit the remaining types of walks.

Observe that sets hitting these walks are exactly S, T-cuts in $G[A \cup B]$ for $S = A_R \cup B_L$ and $T = A_L \cup B_R$. We shall prove that the minimum S, T-cut C is an optimal solution to the auxiliary problem. We have already shown that any solution must be an S, T-cut, so it suffices to prove the following lemma.

Lemma 2. *If C is an S,T-cut, then C hits each odd walk from L to L, each odd walk from R to R, and each even walk from L to R.*

Proof. We shall give an inductive proof with respect to walk's length. Let P be a walk for which we need to show that C hits P. Observe, that if P contains an inner vertex from $L \cup R$, which divides P into P_1 and P_2, then P_1 or P_2 is also a walk C must hit, and by inductive hypothesis it does. Therefore, we may assume that all inner vertices of P lie within $A \cup B$. In this case, we remove the first and last vertex from P to obtain a walk P' in $G[A \cup B]$, which is of one of the following types:

- odd walk from $A_L \cup B_L$ to $A_L \cup B_L$,
- odd walk from $A_R \cup B_R$ to $A_R \cup B_R$,
- even walk from $A_L \cup B_L$ to $A_R \cup B_R$ (or vice versa).

Since $G[A \cup B]$ is bipartite with color classes A, B, an easy case-analysis shows that each kind of walk is either impossible in $G[A \cup B]$ or joins S with T, and then C hits it by definition. □

Consequently, to solve the auxiliary problem it suffices to perform a single min-cut computation. Together with the reductions described before, we obtain an FPT algorithm.

Theorem 3 ([3])**.** *The* Odd Cycle Transversal *problem can be solved in $\mathcal{O}^\star(3^k)$ time.*

3 Polynomial kernel for Odd Cycle Transversal

3.1 Matroid tools

The matroid tools we use are described in Part I and first section of Part II of scribe notes from WorKer's tutorial on matroids. We need Theorems 36 and 38 from these notes:

Theorem 4 ([2])**.** *Let $G = (V, A)$ be a digraph and let $S, T \subseteq V$. The corresponding gammoid has a representation using $\mathcal{O}(min(|T|, |S|log|T|) + log(\frac{1}{\varepsilon}) + log|V|)$-bit integers. Such a representation can be found in $\left(n + \log(\frac{1}{\varepsilon})\right)^{\mathcal{O}(1)}$ time with error probability ε.*

Theorem 5. *Let $G = (V, A)$ be a digraph and $X \subseteq V$ be a set of* terminals*. There exists a gammoid with ground set of size $2|X|$, which gives maximum S,T-flow values in $G \setminus R$ for any partition $X = S \cup T \cup R \cup U$. Moreover, such a gammoid can be constructed in polynomial time.*

To use Theorem 5 for undirected graphs, we replace each edge with a pair of arcs, one in each direction.

3.2 Reformulation of FPT algorithm

Observe that the FPT algorithm we have presented reduces finding OCT to computation of $\mathcal{O}^\star(3^k)$ minimum cuts in induced subgraphs of G. Nevertheless the sizes of the source and sink sets could be large, so Theorem 5 cannot be applied yet.

We were looking for the minimum $(A_R \cup B_L), (A_L \cup B_R)$-cut in $G[A \cup B]$. Instead, we consider a graph G', whose vertices are $A \cup B \cup Y_A \cup Y_B$, where Y_A and Y_B are copies of Y. Edges within $G'[A \cup B]$ are as in G, while for $y \in Y$ vertex y_A is connected to $N_G(y) \cap A$ and y_B to $N_G(y) \cap B$.

For a partition $Y = L \cup R \cup T$ we are going to compute the minimum $(R_A \cup L_B), (L_A \cup R_B)$-cut in $G' \setminus (T_A \cup T_B)$. This is sufficient due to the following lemma:

Lemma 6. *Let $G = (V, G)$ be a graph and $Y \subseteq V$ be such that $G \setminus Y$ is bipartite with color classes A, B. Then, the size of the minimum odd cycle transversal is the minimum over all partitions $Y = L \cup R \cup T$ of the following value:*

$$|T| + mincut_{G' \setminus (T_A \cup T_B)}(R_A \cup L_B, L_A \cup R_B).$$

Proof. First, consider a minimum odd cycle transversal X. Let $T = Y \cap X$ and let $V \setminus X = V_L \cup V_R$ be the partition color classes of $G \setminus X$. As we have seen before, $X \setminus T$ is a $(A_R \cup B_L), (A_L \cup B_R)$-cut in $G[A \cup B]$. It is easy to see that this implies it is also an $(R_A \cup L_B), (L_A \cup R_B)$-cut in $G' \setminus (T_A \cup T_B)$.

Now, consider the partition $Y = L \cup R \cup T$ minimizing the value we compute. If there are many optimal partitions, let us choose one maximizing $|T|$. Observe that in $H = G' \setminus (T_A \cup T_B)$ set we have $N_H(R_A \cup L_B) = A_R \cup B_L$ and $N_H(L_A \cup R_B) = A_L \cup B_R$. Thus, the minimum cut C either is also an $(A_R \cup B_L), (A_L \cup B_R)$-cut and it gives an odd cycle transversal as before, or contains an element of $y_A \in Y_A$ or $y_B \in Y_B$. In the latter case, if we move y to T, then $C \setminus \{y_A, y_B\}$ would be a cut we consider, so the value would not increase. By maximality of $|T|$ among optimal partitions, this is impossible. □

3.3 Polynomial compression for OCT Compression

Note that Theorem 5 to G' and $Y_A \cup Y_B$ shows that the values of all minimum cuts considered in Lemma 6 can be encoded within a gammoid over a ground set of size $\mathcal{O}(|Y|)$ for a graph with $\mathcal{O}(|V|)$ vertices.

Using Theorem 4, this matroid can be represented in $\tilde{\mathcal{O}}(|Y|^3 + \log |V|)$ bits and such a representation can be computed in polynomial time with high probability.

This is does not give a polynomial compression yet, since the size cannot be bounded by a function of $|Y|$. Nevertheless, if $|Y| \leq \log |V|$, then the $\mathcal{O}^\star(3^{|Y|})$-time algorithm works in time polynomial in $|V|$. Thus, in this case we can solve the problem and return a trivial positive or negative instance. Otherwise $\log |V| \leq |Y|$, so the representation has $\tilde{\mathcal{O}}(|Y|^3)$ bits, i.e. we have obtained a polynomial compression.

3.4 Polynomial compression for Odd Cycle Transversal

Observe that iterative compression technique does not give a polynomial compression for Odd Cycle Transversal. This is because subsequent calls of the subroutine solving OCT Compression depend on each other. Instead, we shall use an approximation algorithm:

Theorem 7 ([1])**.** *The* Odd Cycle Transversal *problem admits an $\mathcal{O}(\log n)$ approximation algorithm.*

Theorem 8 ([2])**.** *The* Odd Cycle Transversal *problem can be compressed to $\tilde{\mathcal{O}}(k^{4.5})$ bits.*

Proof. As before, if $k \leq \log n$, we solve the problem using $\mathcal{O}^\star(3^k)$ algorithm and return trivial instances. Otherwise, we apply the approximation algorithm, which returns a set Y. If $|Y| = \omega(k\sqrt{\log n})$, we return a trivial negative instance, since no odd cycle transversal of size $\leq k$ may exist. Otherwise we apply the polynomial compression for OCT Compression. We have $|Y| = \mathcal{O}(k\sqrt{\log n}) = \mathcal{O}(k^{1.5})$, so the compression outputs an instance of size $\tilde{\mathcal{O}}(k^{4.5})$. □

3.5 Polynomial kernel for Odd Cycle Transversal

As we have announced in the introduction, the polynomial kernel for Odd Cycle Transversal is relies on NP-completeness of the non-parameterized version of the problem.

Theorem 9 ([4]). *The* Odd Cycle Transversal *problem is NP-complete.*

It suffices to show that the output language of the polynomial compression we have presented is in NP, so that there is a polynomial reduction back to Odd Cycle Transversal. Indeed, it suffices to make a non-deterministic choice of a partition of Y into $L \cup R \cup T$. Then,

$$|T| + mincut_{G' \setminus (T_A \cup T_B)}(R_A \cup L_B, L_A \cup R_B)$$

can be computed from the gammoid in polynomial time and depending on whether this value is larger than k, we return the result.

Corollary 10. *The* Odd Cycle Transversal *problem parameterized by k admits a polynomial kernel.*

Note that the kernel is polynomial, but we cannot directly control the degree of the polynomial, since we use the generic reduction of NP problems to SAT.

References

[1] Amit Agarwal, Moses Charikar, Konstantin Makarychev, and Yury Makarychev. $\mathcal{O}(\sqrt{\log n})$ approximation algorithms for min UnCut, min 2CNF deletion, and directed cut problems. In Harold N. Gabow and Ronald Fagin, editors, *STOC*, pages 573–581. ACM, 2005.

[2] Stefan Kratsch and Magnus Wahlström. Compression via matroids: a randomized polynomial kernel for odd cycle transversal. In Yuval Rabani, editor, *SODA*, pages 94–103. SIAM, 2012.

[3] Bruce A. Reed, Kaleigh Smith, and Adrian Vetta. Finding odd cycle transversals. *Oper. Res. Lett.*, 32(4):299–301, 2004.

[4] Mihalis Yannakakis. Node- and edge-deletion NP-complete problems. In Richard J. Lipton, Walter A. Burkhard, Walter J. Savitch, Emily P. Friedman, and Alfred V. Aho, editors, *STOC*, pages 253–264. ACM, 1978.