RWTH Aachen **WS 2018**
Lehrgebiet Theoretische Informatik Problem Set 5
Dreier—Kuinke—Rossmanith 19.11.2018

Parameterized Algorithms Tutorial

Tutorial Exercise T1

Let G be a graph and let $S \subseteq V(G)$ be some vertex subset. Show that the following properties are MSO-expressible:

- S is a vertex cover of G
- S is an independent set of G
- G is a connected graph
- S induces a cycle in G
- G has a Hamiltonian path
- S induces an even cycle in G

Proposed Solution

- Vertex cover: $vc(S) = \forall x \forall y (\neg xEy \vee x \in S \vee y \in S)$
- Independent set: $is(S) = \forall x \forall y (\neg xEy \vee x \notin S \vee y \notin S)$
- Connected: Let us introduce a slightly more general formula.

  $$con(S) = \forall A \forall B ((A \subseteq S \wedge B \subseteq S) \rightarrow \exists a \exists b (a \in A \wedge b \in B \wedge aEb))$$

  Where $con(S)$ means that $G[S]$ is connected ($con(V)$ is the formula we need for this part of the exercise)
- Cycle:

  $$\begin{aligned} cycle(S) = {} & con(S) \wedge \forall x \exists a \exists p \forall y ((x \in S \wedge y \in S) \\ & \rightarrow a \neq p \wedge a \in S \wedge p \in S \\ & \wedge (xEy \rightarrow y = a \vee y = p)) \end{aligned}$$

  If we would leave out the connectedness-condition, our formula would also be satisfied by a collection of disjoint cycles.
- Hamiltonian path:

  $$\begin{aligned} hampath = {} & \exists F \subseteq E \exists s \exists t \, path(s, t, V, F) \\ path(s, t, S, F) = {} & \forall x ((x \in S \wedge x \neq s \wedge x \neq t) \\ & \rightarrow \exists a \exists p \forall y (a \in S \wedge p \in S \wedge a \neq p \wedge (xFy \rightarrow y = a \vee y = p))) \end{aligned}$$

- Even cycle:

  $$\begin{aligned} evencycle(S) = {} & cycle(S) \wedge bipartite(S) \\ bipartite(S) = {} & \exists A \exists B \forall a \forall b ((a \in S \wedge b \in S \wedge aEb) \\ & \rightarrow (a \in A \wedge b \in B) \vee (b \in A \wedge a \in B)) \end{aligned}$$

Tutorial Exercise T2

The problem 3-Colorability is defined as follows: Given a graph G decide if it is possible to assign every node of G one of three colors, such that no two nodes with the same color are adjacent. This problem is fpt parameterized by the treewidth of the graph. Give an algorithm that solves the problem given a tree decomposition of width w in time $O(3^w \cdot w \cdot n)$.

Proposed Solution

First convert the tree decomposition into a nice tree decomposition $\mathcal{T}$ in linear time. If X is a bag of the tree decomposition, then let V_X be all nodes which are contained in X and in bag which is a descendant of X in T. We will solve this in the usual way, by computing a table for every bag which will contain entries as follows: For all possible 3-colorings c of $G[V_X]$ we want an entry $c(X)$, i.e. the colors of c for the nodes in X, in the table for the bag X. Such a table can have at most 3^w entries. Having such a table for the root would provide the solution for the decision problem, since by the definition of the table the graph is 3-colorable iff the table for the root bag is not empty. Assume we want to generate the table for some bag X. Let us do it case by case depending on what X can be:

leaf bag Generate a table that contains three entries, one for every color we can give to the only node in the bag.

forget bag Delete the forgotten node from every entry and delete duplicates.

introduce bag When a node x is introduced go through every coloring c of the nodes of the bag in the table of the child of X. If x does not have a neighbor in the bag with color c' add new a entry extending c with x colored with the color c'. This is correct since by the properties of tree decompositions x can only have neighbors which are contained in X in the graph $G[V_X]$.

join bag Only keep the entries which are also contained in both tables of the child bags of X.

Since these operation suffice to compute the table for the root bag in a bottom up fashion, all operations take time $O(3^w)$ for tables which at most 3^w entries and a nice tree decomposition has at most $w \cdot n$ bags it follows that the running time is $O(3^w \cdot w \cdot n)$.

Tutorial Exercise T3

Let k be a constant and consider the class of graphs that have a vertex cover of size *exactly* k. Does this class define a hereditary property? What can you say about the class of graphs that have a vertex cover of size *at most* k? In case you believe that the property is hereditary, how large is the forbidden set?

Proposed Solution

The property asking for a vertex cover of *exactly* size k is not hereditary: especially all graphs of size *smaller* than k do not have this property. As these occur as subgraphs of graphs which have it, the property is note hereditary.

The property asking for a vertex cover of size *at most* k is hereditary by the following argument: say G is a graph with a vertex cover S of size k. Then for every subgraph $G' \leq G$, the set $S \cap V(G')$ is a vertex cover.

We will now prove that a finite set of forbidden subgraph suffices to characterize this property. Consider the set $\mathcal{F}$ of all graphs which have a minumum vertex cover of size $k+1$. For any such graph G, let S be a vertex cover of that size. As we saw with the Buzz kernel, any vertex of degree at least $k+2$ must be inside S. We will use this fact to prove that a set of graphs of *bounded size* suffices as a forbidden set, which implies that this set is finite.

First, take all graphs $G \in \mathcal{F}$ where all vertices in S have a degree bounded by $k+1$. Call this set $\mathcal{F}_{small}$. We can diretly see that this set is finite as it contains graphs of size at most $O(k^2)$.

Now, consider graphs $G \in \mathcal{F}$ where a set of vertices $S' \subseteq S$ have degree larger than $k+1$. We want to argue that if the degree of a vertex $v \in S'$ is *very* high, we can find a subgraph of G which also has S as a minumum vertex cover—which of course means that G does not need to be included in the forbidden set.

Suppose there exists a vertex $v \in V(G) \setminus S$ which is not adjacent to any vertex of $S \setminus S'$ such that in $G - v$, all vertices of S' *still* have degree $> k+1$. Then clearly, S is still a vertex cover of $G - v$ and no smaller vertex cover can exist: the vertices of S' must be included in it anyway, as their degree is still large, and the vertices of $S \setminus S'$ are not affected by the removeal of v (any smaller cover of $G - v$ would imply a smaller cover for G which contradicts our initial assumption).

Now, we build our set $\mathcal{F}_{large}$ as follows: include any graph from $\mathcal{F} \setminus \mathcal{F}_{small}$ which does *not* have such a vertex.

Homework H1

Let G be a graph and let $S \subseteq V(G)$ be some vertex subset. Show that the following properties are MSO-expressible:

- S is a dominating set of G
- S is a distance-2 dominating set of G
- P is a longest path in G
- S is a Steiner tree in G

Proposed Solution

- U is a dominating set iff
$$\forall x(x \in U \vee \exists y adj(x,y) \wedge y \in U)).$$
- U is a distance-2 dominating set iff
$$\forall x(x \in U \vee \exists y((adj(x,y) \wedge y \in U) \vee \exists z((adj(x,y) \wedge adj(y,z) \wedge z \in U))))$$
- P is a path in G iff
$$\exists s \exists t \exists F(s \in P \wedge t \in P \wedge path(s,t,P,F).$$
- S is a Steiner graph with terminal set T iff
$$(\forall R(\neg((\forall x(x \notin R \vee x \in S)) \wedge (\exists x(x \in R)) \wedge (\exists x(x \notin R \wedge x \in S)))$$
$$\vee \exists x \exists y(adj(x,y) \wedge x \in R \wedge y \notin R \wedge y \in S)) \wedge (\forall x(x \in U \vee x \notin T)).$$
This graph is automatically a tree, if S is of minimal cardinality.

Homework H2

The problem DOMINATING SET is defined as follows: Given a graph G find the smallest set $S \subseteq V(G)$ such that every node of G is either in S or has a neighbor in S. This problem is fpt parameterized by the treewidth of the graph. Give an algorithm that solves the problem given a tree decomposition of width w in time $O(9^w \cdot w \cdot n)$.

Proposed Solution

We will use the same notation as in for the solution of T1. We will also solve it in a very similar manner. The interpretation of the tables for the bags will this time be somewhat more complicated: In the table we will have an entry which represent partitions of the current bag X intro three sets S, D and F and are associated with a number s. Such an entry is in the table only if there exists a partial dominating set of $G[V_X]$ of size s where every node not in the bag X is dominated, the nodes in S are in the dominating set, the nodes in D are dominated but not in the dominating set and the nodes in F are not dominated. Because we are trying to minimize the size of the dominating set between two entries with the same sets but different sizes s we will only keep the entry where the size is smaller. This means that, since there are only 3^w ways to partition a set of size w into three sets, the tables have at most 3^w entries. If we can compute such a table we would be able to read the size if a minimal dominating set by taking the minimum size over all the entries where F is empty. We can then again describe the operations needed to compute these tables in a bottom up fashion a nice tree decomposition.

leaf bag Create three entries, such that x is contained only in S, D and F respectively. If the nodes is in S set s to one, otherwise to zero.

forget bag Take the whole table of the child bag. Remove all the entries where the forgotten node x is contained in F. Then remove x from every set in the remaining table. We can do this because we only want entries that represent partial dominating sets that dominate every node which is not in the bag X.

introduce bag Take every entry (S, D, F, s) and create three new entries as follows:

- Put x in S, all the neighbors of x in X in the set D and increase s by one. This is correct since by the properties of tree decompositions x can not dominate anything in $G[V_X]$ which is not in the bag X.
- Put x in D if x has a neighbor in S. Keep s.
- Put x in F if x does not have a neighbor in S. Keep s.

join bag Take the tables t_1 and t_2 from the children bags of X. For every pair of entries $(S_1, D_1, F_1, s_1) \in t_1$ and $(S_2, D_2, F_2, s_2) \in t_2$ generate a new entry (S, D, F, s) where

- $S = S_1 \cup S_2$,
- $D = (D_1 \cup D_2) \setminus (S_1 \cup S_2)$,
- $F = X \setminus (S \cup D)$ and
- $s = s_1 + s_2 - |D_1 \cap D_2|$.

This is correct since by the properties of tree decompositions when merging two entries we only have to take care that the solution agrees on the join bag. Since we have to look at all pairs this step takes time $O((3^w)^2) = O(9^w)$.

By using this algorithm on a nice tree decomposition, since the join operation has the worst running time, we have an algorithm with the desired running time.

Homework H3

Let Π be a hereditary property that *excludes* some clique and some independent set. Show that Π-Induced Subgraph is FPT.

Proposed Solution

Let the size of the smallest independent set and smallest clique excluded by Π be s and t, respectively. Then no graph with $R(s,t)$ vertices or more satisfies Π. Since s and t are constants, this implies that Π is finite and hence the Induced Subgraph problem is *constant* time solvable.