FPT algorithms for path-transversals and cycle-transversals problems in graphs

S. Guillemot

INRIA, France

20 February 2008

Introduction

In short: We consider graph problems

- aiming at breaking some substructures in a graph (sets of paths or sets of cycles) by edge or vertex deletion;
- from the point of view of parameterized complexity: is there a solution with p deletions?

Examples: Some well-known examples:

- the Feedback Vertex Set problem [FLRS05,GGHNW06]: given a graph, remove p vertices to break each cycle (= to obtain a tree);
- the Graph Bipartization problem [RSV04,GGHNW06]: given a graph, remove p vertices to break each odd cycle (= to obtain a bipartite graph).

...and also the Directed Feedback Vertex Set problem which was recently proved FPT [CLLSR08].

The PATH COVER problem

We first consider a generic PATH COVER problem, and we describe an FPT algorithm for *homogeneous* instances.

Definition

A *path system* is a tuple $\sigma = (G, T, F, \mathcal{P})$ where (i) $G = (V, E)$ is an undirected graph, (ii) $T \subseteq V$ is a set of *terminals*, (iii) $F \subseteq V$ is a set of *forbidden vertices*, (iv) $\mathcal{P}$ is a set of paths in G joining terminals.

The PATH COVER problem takes a path system σ and seeks a set of vertices $S \subseteq V \backslash F$ which hits each path of $\mathcal{P}$.

Remarks:

- The cardinality of $\mathcal{P}$ can be exponential in $|V|$, hence we assume that we have some "oracle" for $\mathcal{P}$ using a polynomial-size description;
- Here we are interested in the parameterized problem: given a parameter p, is there a solution of cardinality $\leq p$?

The PATH COVER problem

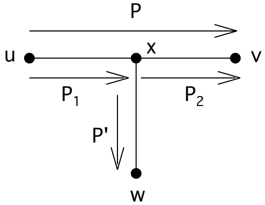
We first consider a generic PATH COVER problem, and we describe an FPT algorithm for *homogeneous* instances.

Definition

A path system $\sigma = (G, T, F, \mathcal{P})$ is *homogeneous* iff the following conditions hold:

1. for each path $P \in \mathcal{P}$, there exists a simple path $P' \in \mathcal{P}$ which is included in P;
2. for each path $P = P_1 x P_2 \in \mathcal{P}$ joining $u, v \in T$, for each $w \in T$ and P' joining x to w, one of $P_1 P'$, $\tilde{P'} P_2$ is in $\mathcal{P}$.

Illustration:



The PATH COVER problem

We first consider a generic PATH COVER problem, and we describe an FPT algorithm for *homogeneous* instances.

LP formulation

The problem can be formulated as an IP (integer program). We consider its LP (linear program) relaxation as well as the dual LP:

$$\begin{cases} \text{minimize } \sum_{v \in V} x_v \\ \text{subject to } \forall P \in \mathcal{P}, \sum_{v \in P} x_v \geq 1, \ldots \end{cases} \quad \begin{cases} \text{maximize } \sum_{P \in \mathcal{P}} f_P \\ \text{subject to } \forall v \in V \backslash F, \sum_{P \in \mathcal{P}: v \in P} f_P \leq 1, \ldots \end{cases}$$

If the instance is homogeneous then:

- the LP has a half-integral solution (generalizes a known property of MULTIWAY CUT [GVY94]);
- the PATH COVER problem can be solved in $O^*(4^p)$ time, using bounded search guided by half-integral solutions.

The Path Cover problem

We first consider a generic Path Cover problem, and we describe an FPT algorithm for *homogeneous* instances.

LP formulation

The problem can be formulated as an IP (integer program). We consider its LP (linear program) relaxation as well as the dual LP:

$$\left\{\begin{array}{l}\text{minimize } \sum_{v \in V} x_v \\ \text{subject to } \forall P \in \mathcal{P}, \sum_{v \in P} x_v \geq 1, \ldots\end{array}\right. \quad \left\{\begin{array}{l}\text{maximize } \sum_{P \in \mathcal{P}} f_P \\ \text{subject to } \forall v \in V \backslash F, \sum_{P \in \mathcal{P} : v \in P} f_P \leq 1, \ldots\end{array}\right.$$

Notations:

- we denote by opt^*_σ the cost of an optimal solution of the LP;
- we denote by opt_σ the cost of an optimal solution of the IP.

If the instance is homogeneous, then by half-integrality, we have $opt^*_\sigma \leq opt_\sigma \leq 2opt^*_\sigma$.

The PATH COVER problem

We first consider a generic PATH COVER problem, and we describe an FPT algorithm for *homogeneous* instances.

Principle of the algorithm

Suppose that σ is homogeneous. We solve the PATH COVER problem for the instance (σ, p) by a recursive algorithm. We proceed as follows:

- we solve the LP and compute opt^*_σ;
- based on this value, we either fall in a base case, or issue recursive calls for instances (σ', p') computed from (σ, p).

Base cases:

- if $opt^*_\sigma \leq \frac{p}{2}$, we answer "yes";
- if $opt^*_\sigma > p$, we answer "no".

This is correct since $opt^*_\sigma \leq opt_\sigma \leq 2opt^*_\sigma$ (by half-integrality).

The Path Cover problem

We first consider a generic Path Cover problem, and we describe an FPT algorithm for *homogeneous* instances.

Principle of the algorithm

Suppose that σ is homogeneous. We solve the Path Cover problem for the instance (σ, p) by a recursive algorithm. We proceed as follows:

- we solve the LP and compute opt^*_σ;
- based on this value, we either fall in a base case, or issue recursive calls for instances (σ', p') computed from (σ, p).

General case:

Choose a vertex u according to some criterion (not discussed here). Consider $\sigma' = (G, T, F \cup \{u\}, \mathcal{P})$ and $\sigma'' = (G \backslash \{u\}, T, F, \mathcal{P})$.
Clearly: (σ, p) is a positive instance iff one of $(\sigma', p), (\sigma'', p-1)$ is a positive instance.
This suggests issuing two recursive calls for these instances.

The Path Cover problem

We first consider a generic Path Cover problem, and we describe an FPT algorithm for *homogeneous* instances.

Principle of the algorithm

Suppose that σ is homogeneous. We solve the Path Cover problem for the instance (σ, p) by a recursive algorithm. We proceed as follows:

- we solve the LP and compute opt^*_σ;
- based on this value, we either fall in a base case, or issue recursive calls for instances (σ', p') computed from (σ, p).

General case:

Problem: the first recursive call (for (σ', p)) does not decrease the value of the parameter $\rightarrow$ no guarantee of termination.

Solution: we will compensate the fact that p does not change by an increase in opt^*.

Namely: we will issue these two recursive calls only when $opt^*_{\sigma'} > opt^*_\sigma$.

The PATH COVER problem

We first consider a generic PATH COVER problem, and we describe an FPT algorithm for *homogeneous* instances.

Principle of the algorithm

Suppose that σ is homogeneous. We solve the PATH COVER problem for the instance (σ, p) by a recursive algorithm. We proceed as follows:

- we solve the LP and compute opt^*_σ;
- based on this value, we either fall in a base case, or issue recursive calls for instances (σ', p') computed from (σ, p).

General case:

What do we do when $opt^*_{\sigma'} = opt^*_\sigma$?

It turns out that in this case $opt_{\sigma'} = opt_\sigma$ holds. The proof is involved and heavily relies on the assumption that the instance is homogeneous.

Thus, whenever $opt^*_{\sigma'} = opt^*_\sigma$, we issue only *one* recursive call for the equivalent instance (σ', p).

The PATH COVER problem

We first consider a generic PATH COVER problem, and we describe an FPT algorithm for *homogeneous* instances.

Algorithm SOLVEPATHCOVER(σ, p)

Compute opt^*_σ;
If $opt^*_\sigma \leq p/2$ then return "yes"; if $opt^*_\sigma > p$ then return "no";
Consider the instances σ', σ'' as before;
If $opt^*_\sigma = opt^*_{\sigma'}$ then return SOLVEPATHCOVER(σ', p);
Else return (SOLVEPATHCOVER(σ', p) *or* SOLVEPATHCOVER($\sigma'', p-1$)).

Analysis:
For an instance (σ, p), define $k = 2p + 1 - 2opt^*_\sigma$, then in the two recursive calls of the last line the values of p, k are:

- for the first call: $p, \leq k - 1$ (since opt^* increases by at least $1/2$);
- for the second call: $p - 1, \leq k$ (since p decreases by 1 while opt^* increases by at most 1).

The PATH COVER problem

We first consider a generic PATH COVER problem, and we describe an FPT algorithm for *homogeneous* instances.

Algorithm SOLVEPATHCOVER(σ, p)

Compute opt^*_σ;
If $opt^*_\sigma \leq p/2$ then return "yes"; if $opt^*_\sigma > p$ then return "no";
Consider the instances σ', σ'' as before;
If $opt^*_\sigma = opt^*_{\sigma'}$ then return SOLVEPATHCOVER(σ', p);
Else return (SOLVEPATHCOVER(σ', p) *or* SOLVEPATHCOVER($\sigma'', p-1$)).

Analysis:
We obtain a recurrence of the form

$$\begin{cases} T(p,k) & \leq 1 \text{ if } p = 0 \text{ or } k = 0 \\ T(p,k) & \leq T(p,k-1) + T(p-1,k) \text{ otherwise} \end{cases}$$

which solves to $T(p,k) \leq 2^{p+k}$. The $O^*(4^p)$ running time is obtained by observing that $k \leq p$ always holds for internal nodes of the search tree.

Graph problems

The algorithm for PATH COVER gives rise to alternative or new fpt-algorithms for several graph problems.

First kind: separation problems

We are given a graph with distinguished vertices called *terminals*, and the objective is to break some paths between terminals.

The MULTIWAY CUT problem aims at disconnecting each pairs of terminals. The MULTICUT problem aims at disconnecting specified pairs of terminals.

Remarks:

- problems already considered in [M06,CLS07] from the point of view of parameterized complexity;
- there are two parameters of interest: p = number of deletions, k = number of terminals.

Graph problems

The algorithm for PATH COVER gives rise to alternative or new fpt-algorithms for several graph problems.

First kind: separation problems

We are given a graph with distinguished vertices called *terminals*, and the objective is to break some paths between terminals.

The MULTIWAY CUT problem aims at disconnecting each pairs of terminals. The MULTICUT problem aims at disconnecting specified pairs of terminals.

Results:

Problem	k, p	p
MULTIWAY CUT		$O^*(4^p)$ algorithm
MULTICUT	$O^*((8k)^p)$ algorithms	Open

Graph problems

The algorithm for PATH COVER gives rise to alternative or new fpt-algorithms for several graph problems.

First kind: separation problems

We are given a graph with distinguished vertices called *terminals*, and the objective is to break some paths between terminals.

The MULTIWAY CUT problem aims at disconnecting each pairs of terminals. The MULTICUT problem aims at disconnecting specified pairs of terminals.

Results:

Problem	k, p	p
MULTIWAY CUT		$O^*(4^p)$ algorithm
MULTICUT	$O^*((8k)^p)$ algorithms	Open

Straightforward reduction to PATH COVER. A different $O^*(4^p)$ algorithm was obtained by [CLS07], improving a previous FPT algorithm by [M06].

Graph problems

The algorithm for PATH COVER gives rise to alternative or new fpt-algorithms for several graph problems.

First kind: separation problems

We are given a graph with distinguished vertices called *terminals*, and the objective is to break some paths between terminals.

The MULTIWAY CUT problem aims at disconnecting each pairs of terminals. The MULTICUT problem aims at disconnecting specified pairs of terminals.

Results:

Problem	k, p	p
MULTIWAY CUT		$O^*(4^p)$ algorithm
MULTICUT	$O^*((8k)^p)$ algorithms	Open

Enumerates $O^*((2k)^p)$ realizable partitions, and for each partition solves a PATH COVER problem in $O^*(4^p)$ time. Improves an FPT algorithm by [M06].

Graph problems

The algorithm for PATH COVER gives rise to alternative or new fpt-algorithms for several graph problems.

First kind: separation problems

We are given a graph with distinguished vertices called *terminals*, and the objective is to break some paths between terminals.

The MULTIWAY CUT problem aims at disconnecting each pairs of terminals. The MULTICUT problem aims at disconnecting specified pairs of terminals.

Results:

Problem	k, p	p
MULTIWAY CUT		$O^*(4^p)$ algorithm
MULTICUT	$O^*((8k)^p)$ algorithms	Open

Is MULTICUT FPT for the single parameter p? Open question already mentioned in [M06].

Graph problems

The algorithm for PATH COVER gives rise to alternative or new fpt-algorithms for several graph problems.

Second kind: group feedback problems

Let Γ be a group, we are given a digraph G s.t. each arc a is labelled by an element $\lambda(a) \in \Gamma$. A *nonnull cycle* is a cycle $x_1 \rightarrow_{a_1} x_2 \rightarrow_{a_2} \ldots x_m \rightarrow_{a_m} x_1$ s.t. $\lambda(a_1)\ldots\lambda(a_m) \neq 1_\Gamma$.

The GROUP FEEDBACK SET problems aim at breaking each nonnull cycle of G.

Remarks:

- the GRAPH BIPARTIZATION problem is a special case of the GROUP FEEDBACK SET problem with $\Gamma = \mathbb{Z}_2$;
- the parameters of interest are: p = the number of deletions, s = the cardinality of Γ.

Graph problems

The algorithm for PATH COVER gives rise to alternative or new fpt-algorithms for several graph problems.

Second kind: group feedback problems

Let Γ be a group, we are given a digraph G s.t. each arc a is labelled by an element $\lambda(a) \in \Gamma$. A *nonnull cycle* is a cycle $x_1 \rightarrow_{a_1} x_2 \rightarrow_{a_2} ... x_m \rightarrow_{a_m} x_1$ s.t. $\lambda(a_1)...\lambda(a_m) \neq 1_\Gamma$.

The GROUP FEEDBACK SET problems aim at breaking each nonnull cycle of G.

Results:

Problem	s, p	p
GROUP FEEDBACK ARC SET	$O^*((4s+1)^p)$	$O^*((8p+1)^p)$
GROUP FEEDBACK VERTEX SET	$O^*((4s+1)^p)$	Open

Graph problems

The algorithm for PATH COVER gives rise to alternative or new fpt-algorithms for several graph problems.

Second kind: group feedback problems

Let Γ be a group, we are given a digraph G s.t. each arc a is labelled by an element $\lambda(a) \in \Gamma$. A *nonnull cycle* is a cycle $x_1 \rightarrow_{a_1} x_2 \rightarrow_{a_2} \dots x_m \rightarrow_{a_m} x_1$ s.t. $\lambda(a_1)\dots\lambda(a_m) \neq 1_\Gamma$.

The GROUP FEEDBACK SET problems aim at breaking each nonnull cycle of G.

Results:

Problem	s, p	p
GROUP FEEDBACK ARC SET	$O^*((4s+1)^p)$	$O^*((8p+1)^p)$
GROUP FEEDBACK VERTEX SET	$O^*((4s+1)^p)$	Open

Uses *iterative compression* similar to the algorithm of [RSV04] for GRAPH BIPARTIZATION; at each compression step, solves $O(s^p)$ PATH COVER problems.

Graph problems

The algorithm for Path Cover gives rise to alternative or new fpt-algorithms for several graph problems.

Second kind: group feedback problems

Let Γ be a group, we are given a digraph G s.t. each arc a is labelled by an element $\lambda(a) \in \Gamma$. A *nonnull cycle* is a cycle $x_1 \rightarrow_{a_1} x_2 \rightarrow_{a_2} ... x_m \rightarrow_{a_m} x_1$ s.t. $\lambda(a_1)...\lambda(a_m) \neq 1_\Gamma$.

The Group Feedback Set problems aim at breaking each nonnull cycle of G.

Results:

Problem	s, p	p
Group Feedback Arc Set	$O^*((4s+1)^p)$	$O^*((8p+1)^p)$
Group Feedback Vertex Set	$O^*((4s+1)^p)$	Open

Adaptation of the previous algorithm, by restricting the number of Path Cover problems to solve.

Graph problems

The algorithm for PATH COVER gives rise to alternative or new fpt-algorithms for several graph problems.

Second kind: group feedback problems

Let Γ be a group, we are given a digraph G s.t. each arc a is labelled by an element $\lambda(a) \in \Gamma$. A *nonnull cycle* is a cycle $x_1 \rightarrow_{a_1} x_2 \rightarrow_{a_2} \ldots x_m \rightarrow_{a_m} x_1$ s.t. $\lambda(a_1) \ldots \lambda(a_m) \neq 1_\Gamma$.

The GROUP FEEDBACK SET problems aim at breaking each nonnull cycle of G.

Results:

Problem	s, p	p
GROUP FEEDBACK ARC SET	$O^*((4s+1)^p)$	$O^*((8p+1)^p)$
GROUP FEEDBACK VERTEX SET	$O^*((4s+1)^p)$	Open

Open question: is GROUP FEEDBACK VERTEX SET FPT for the single parameter p?

Conclusion

Summary:

- a $O^*(4^p)$ time algorithm for the generic Path Cover problem, relying on a LP formulation and a half-integrality property of the LP.
- yields alternative or new fpt algorithms for various graph problems: separation problems and group feedback set problems.

Open questions:

- for several graph problems considered: existence of an fpt algorithm for the single parameter p?
- adapt the fpt results to variants of the group feedback set problems? An example: satisfiability of systems of linear equations with two equations per variable, allowing at most p unsatisfied equations.