An FPT Variant Of The Shadow Problem With Kernelization

Stefan Porschen *

Abstract—**The shadow problem (SIS) gets as input a forest F, and a map that assigns subtrees, called shadows, to leaves of F.** SIS **asks whether there exists a set of $|F|$ leaves, one from each tree, such that no leaf lies in the shadow of another. Usually** SIS **is considered as a parameterized problem with parameter k bounding the cardinality of F, for which some fixed-parameter tractability time bounds have been proven, namely $O(n^2k^k)$ in [2] and $O(n^33^k)$ in [4], where n is the number of vertices in F. In this paper, we discuss a variant of** SIS **that essentially is characterized through a different parameterization using two independent parameters, namely k as above, and s bounding the shadow size. We provide a kernelization w.r.t. this parameterization, and prove a fixed-parameter tractability bound of $O(k \cdot n^2 + p(k,s)3^k)$ where p is a polynomial in the parameters k, s.**

Keywords: shadow-independent set problem, forest, fixed-parameter tractability, kernelization

1 Introduction

The *shadow-independent set problem (for short shadow problem or SIS)* is the graph theoretical formulation of a propositional logic problem, namely the falsifiability problem for pure implicational formulas; for details the reader is refered to [2]. SIS has also been investigated in a pure graph theoretical framework [4] likewise in the present paper. Input instances of SIS consist of a finite forest F over n vertices and a (partial) map σ, called *shadow map*, assigning leaves of F to its vertices. The relevant objects in SIS are the leaves of F, which obey a dependence structure impressed by the shadow map: For a leaf ℓ (in the domain of σ), we call the subtree of F rooted at vertex $\sigma(\ell)$ the *shadow* of ℓ, and we say that all leaves of that subtree are *shadow-dependent on* ℓ. SIS then asks for the existence of a set of mutually shadow-independent leaves in F exactly one of each tree. In general, this problem is NP-complete [2]. Nevertheless, from the point of view of parameterized complexity [1], SIS admits a natural parameterization where the parameter k bounds the number of trees in an input instance: $|F| \leq k$. According to this parameterization, SIS has been proven to belong to the complexity class FPT (fixed-parameter tractable problems) in [2], where an algorithm for SIS running in time $O(n^2k^k)$ is provided. Another algorithm has been designed by in [4] providing the time bound $O(n^33^k)$ substantially improving on the parameter factor.

An important feature exhibited by all problems in FPT is the existence of a so-called *kernelization* w.r.t. the parameter(s) [1]. By a kernelization one extracts in a preprocessing step the *kernel* of the problem which is the relevant part of the input instance bounded in size by a function f in the parameter(s). Thus kernelization leads to an overall bound of the form $O(p(n,k_1,\ldots,k_s)+f(k_1,\ldots,k_s))$, called *kernel form*, where p is a polynomial in the problem size n as well as in the parameters, and f is an arbitrary function depending on the parameters only. So far, no kernelization for SIS w.r.t. k was constructed, which however must exist [1]. A reason might be that SIS input instances (F,σ) consist of two independent components, and the standard parameter k only controls the forest part. To circumvent this difficulty, here we propose a variant of the shadow independent set problem which is defined through an additional parameter s bounding the number of leaves contained in shadows generated by σ. We are able to prove a fixed-parameter tractability time bound having the kernel form $O(k \cdot n^2 + p(k,s)3^k)$, with a polynomial $p(k,s)$ in the two parameters. The proof is based on the dynamic programming algorithm as designed for the standard parameterization of SIS provided in [4].

2 Basic definitions and notation

For a graph G, we denote by $V(G)$ its vertex set. In this paper we are mainly interested in finite forests F and (rooted) trees T (a tree is a connected forest, for details cf., e.g., [3]). Let $L(F)$ denote the set of leaves in F. In a rooted tree T, with root w, every vertex $x \in V(T)-\{w\}$ selects the unique proper subtree S_x of T rooted at x. Given a vertex $x \in V(F)$, we denote by $T(x) \in F$ the unique tree it belongs to; similarly for a vertex set $X \subset V(F)$ we write $T(X) \in F$, if X contains vertices of one tree only. Let 2^M denote the power set of a set M, and let $\overline{A} := M - A$ denote the complement of $A \subseteq M$. Given a (partial) map $f : M \to M'$, we denote its domain by $D(f) \subseteq M$ and its image by $I(f) \subseteq M'$. Recall that each $y \in I(f)$ has a non-empty *preimage* $f^{-1}(y) \subseteq D(f)$.

A SIS input instance (F,σ) consists of a finite forest F

*Institut für Informatik, Universität zu Köln, D-50969 Köln, Germany, Email: porschen@informatik.uni-koeln.de

and a (partial) map $\sigma : L(F) \to V(F)$, the *shadow map*. For $\ell \in D(\sigma)$, we call the unique subtree $S_{\sigma(\ell)}$ of F the *shadow* of ℓ as depicted in Figure 1.

Two leaves $\ell, \ell' \in L(F)$ are said to be *shadow-independent* if and only if $\ell \notin L(S_{\sigma(\ell')})$ and $\ell' \notin L(S_{\sigma(\ell)})$. The notion of shadow-independency canonically transfers to sets of leaves. A *transversal* in a forest F is defined to be a set $\Lambda \subseteq L(F)$ of leaves such that $|\Lambda \cap L(T)| = 1$, for each $T \in F$.

Definition 1 *The* shadow(-independent set) problem (SIS) *is defined through: input instance* $(F, \sigma) \in$ SIS *if and only if there exists a shadow-independent transversal in* F.

Throughout it is assumed that each input instance (F, σ) of SIS satisfies the following conditions:
(i): No leaf is mapped to the root of a tree in F, and
(ii): no leaf is mapped to a vertex of its own tree.
(These conditions essentially yield no loss of generality since an arbitrary input instance can be transformed fast into one satisfying (i), (ii): Leaves that are mapped to roots have to be marked and ignored since their shadows rule out whole trees and thus can never join a shadow-independent transversal in F. Similarly, a leaf generating a shadow in its own tree is of importance only if it is contained in its own shadow, in which case it has to be ignored, too. Otherwise such a leaf can contribute to any shadow-independent transversal. Marking corresponding leaves in an arbitrary SIS input instance accordingly needed $O(|V(F)|^2)$ time using depth first search, which is dominated by the time complexity of the preprocedure as stated in the final result.)

Definition 2 *The* parameterized problem SIS_k *with parameter* $k \in \mathbb{N}$ *is defined by:* $(F, \sigma, k) \in \mathrm{SIS}_k$ *if and only if* $|F| \leq k$ *and* $(F, \sigma) \in$ SIS.

The parameterized version of SIS as above has been studied in the past to some extent [2, 4]. In the following a different parameterization is defined introducing a second parameter bounding the size of shadows generated by σ:

$$s_\sigma := \max_{\ell \in D(\sigma)} |L(S_{\sigma(\ell)})|$$

Throughout we assume that $s_\sigma \geq 1$, equivalently $D(\sigma) \neq \varnothing$, otherwise the instance contains no shadow and thus is trivial.

Definition 3 *With (vector-)parameter* $\kappa = (k, s) \in \mathbb{N}^2$, *problem* SIS_κ *is defined through:* $(F, \sigma, \kappa) \in \mathrm{SIS}_\kappa$ *if and only if* $|F| \leq k, s_\sigma \leq s$, *and* $(F, \sigma) \in$ SIS.

A leaf is called *free* if it neither is contained in $D(\sigma)$ nor in $I(\sigma)$. Observe that a tree having a free leaf can be removed from a SIS input instance because it contributes, via its free leaf, to any shadow-independent transversal in the remaining forest.

3 Specifying the kernel of SIS_κ

In this section we provide the specification of a (problem) kernel for input instances (F, σ, κ) of SIS_κ. We show that the relevant part of the kernel is bounded in size by a function in the parameters k, s. It turns out to be useful, first to take a more abstract view on the shadow structure of (F, σ) by only considering those shadows that are not subtrees of other shadows. Note that two shadows in a tree T either are disjoint subtrees of T or one is a subtree of the other. Thus, in a tree T containing shadows there exists at least one shadow that is not subtree of any other shadow in T; we call such a shadow an *envelope*. An envelope usually contains several shadows as subtrees (cf. Fig. 2).

Let $B \subseteq I(\sigma)$ be the collection of all envelope roots in the input instance. Clearly, largest shadows in F are envelopes, hence $s_\sigma = \max_{b \in B} |L(S_b)|$. For envelope S_b, $b \in B$, we consider its *preenvelope*

$$H(b) := \bigcup_{x \in I(\sigma) \cap V(S_b)} \sigma^{-1}(x)$$

collecting the leaves of all non-empty preimages of vertices in S_b. A preenvelope $H(b)$ is composed of its tree fragments $H_T(b) := H(b) \cap L(T)$, for each $T \in F$. For convenience, let $\mathcal{H}(T) := \{H_T(b) : b \in B\}$ denote the collection of all different preenvelope fragments in T. Every leaf of a preenvelope fragment $H_T(b)$ either is contained in exactly one envelope of T. Or it lies outside of all envelopes contained in T, i.e., is contained in no shadow at all; we call the set of these leaves the *0-fragment* $H^0_T(b)$ of $H_T(b)$. Let $\mathcal{H}^0(W) := \{H^0_T(b) : b \in B, T \in W\}$ be the collection of all 0-fragments in subforest $W \subseteq F$ (if $W = \{T\}$ we simply write $\mathcal{H}^0(T)$). A basic object regarding kernelization is an *autonomous* tree which, as will turn out, can be characterized by the number of its preenvelope fragments.

Definition 4 *A tree* T *in* (F, σ) *is called* autonomous *if for every transversal* Λ *in* $F - \{T\}$ *there exists a leaf in* $L(T)$ *that is mutually shadow-independent of every leaf in* Λ.

From the definition directly follows that an autonomous tree can be removed from a SIS input instance since it always can complete a shadow-independent transversal in the remaining forest. Therefore, a SIS search algorithm can be restricted to the remaining forest:

Lemma 1 *If* T *in* (F, σ) *is autonomous then there exists a shadow-independent transversal in* F *if and only if there exists a shadow-independent transversal in* $F - \{T\}$.

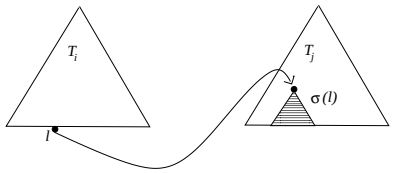

Figure 1: Leaf ℓ of T_i generates the shadow rooted at $\sigma(\ell)$ in T_j.

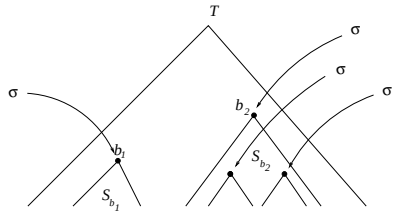

Figure 2: A tree T containing two envelopes S_{b_1}, S_{b_2}; arrows correspond to assignments of σ.

The key to specifying a problem kernel of SIS_κ is the observation that there either exists an autonomous tree in F or all its trees have a bounded number of preenvelope fragments. Basis for this observation is the following necessary condition for a tree to be autonomous.

Lemma 2 *A tree T in (F,σ) is autonomous if*

$$(*) \quad |\mathcal{H}(T)| \geq 1 + (|F|-1)(s_\sigma + 1)$$

Proof. Let $T \in F$ such that $|\mathcal{H}(T)| \geq 1+(|F|-1)(s_\sigma+1)$, and let Λ be an arbitrary transversal in $F - \{T\}$. In worst case, each $\ell \in \Lambda$ is mapped into $V(T)$ generating shadow $S_{\sigma(\ell)}$ in T, and all these shadows are mutually vertex-disjoint, i.e., envelopes. Because s_σ is the maximum number of leaves in a shadow of (F,σ) we have

$$\sum_{\ell\in\Lambda} |L(S_{\sigma(\ell)})| \leq (|F|-1)s_\sigma$$

In worst case all preenvelope fragments of T are single element sets, implying that Λ rules out at most $(|F|-1)s_\sigma$ preenvelope fragments in $\mathcal{H}(T)$. On the other hand, we have to assume that there is a set $\hat{\Lambda} \subseteq D(\sigma) \cap L(T)$ containing $|F|-1$ leaves lying outside of $\bigcup_{\ell\in\Lambda} L(S_{\sigma(\ell)})$ such that, for every $\hat{\ell} \in \hat{\Lambda}$, there exists $\ell \in \Lambda$ with $\ell \in L(S_{\sigma(\hat{\ell})})$. Again supposing that each $\hat{\ell} \in \hat{\Lambda}$ delivers a single element preenvelope fragment another $|F|-1$ preenvelope fragments in $\mathcal{H}(T)$ are ruled out. In summary, there are $r := |F|-1+(|F|-1)s_\sigma$ preenvelope fragments in $\mathcal{H}(T)$ which under worst circumstances cannot provide a leaf shadow-independent of all leaves in Λ. But by assumption, for T we have $|\mathcal{H}(T)| - r \geq 1$, so there is at least one preenvelope fragment left in $\mathcal{H}(T)$ containing at least one element $\hat{\ell}_0$. Although its shadow $S_{\sigma(\hat{\ell}_0)}$ must be contained in a tree of $\hat{F}$ it does not contain a member of Λ because envelopes, by construction, are mutually disjoint subtrees. Observe that $\hat{\ell}_0$ does not need to be the same leaf for every transversal in $F - \{T\}$. The situation is illustrated in Fig. 3. □

Hence, existence of autonomous trees in fact allows us to reduce the input instance: Recursively remove all autonomous trees. Then it is sufficient to decide SIS for the remaining input instance, called the *kernel* of the problem.

Definition 5 *The* kernel $(\hat{F},\hat{\sigma})$ *of a* SIS *instance* (F,σ) *is defined as follows:*
$\hat{F} \subseteq F$ is the largest subforest of F such that for every $T \in \hat{F}$ we have:
(i) there is no free leaf in T, (ii) T is not autonomous, and
$\hat{\sigma} := \sigma|\hat{F}$ is the restriction of σ to $\hat{F}$ (meaning $D(\hat{\sigma}) := D(\sigma) \cap [L(\hat{F}) - \{\ell \in L(\hat{F}) : \sigma(\ell) \in V(\overline{\hat{F}})\}]$, i.e., the domain is restricted to $L(\hat{F})$, and additionally all leaves are removed from the domain that were previously mapped to vertices of $F - \hat{F}$).

Proposition 1 *(1) For every (F,σ), the kernel $(\hat{F},\hat{\sigma})$ exists and is unique.*
*(2) $(F,\sigma) \in$ SIS *if and only if* $(\hat{F},\hat{\sigma}) \in$ SIS.
(3) Let B denote the set of all envelope roots in $\hat{F}$ and $L_B := \bigcup_{b\in B} L(S_b)$, then

$$(\alpha) \quad \sum_{T\in\hat{F}} |\mathcal{H}(T)| \leq \rho(|\hat{F}|, s_{\hat{\sigma}}), \quad (\beta) \quad |L_B| \leq s_{\hat{\sigma}} \cdot \rho(|\hat{F}|, s_{\hat{\sigma}})$$

with $\rho(x,y) := x(x-1)(y+1)$.

Proof. For a constructive proof of (1) consider:
Procedure KERNEL:
For input (F,σ), initially assign $\hat{F} \leftarrow F$, $\hat{\sigma} \leftarrow \sigma$, and $F' \leftarrow F$.
Loop (to be repeated as long as $F' \neq \varnothing$):
(A) Assign $F' \leftarrow \varnothing$.

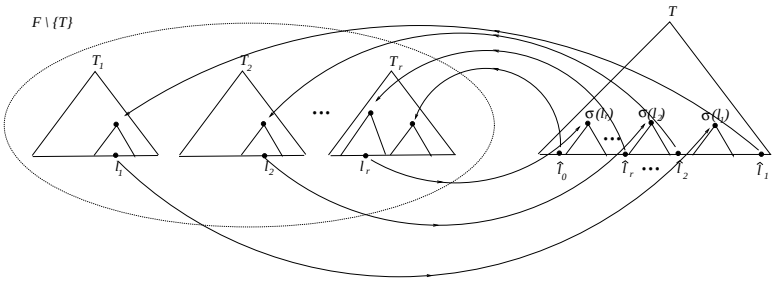

Figure 3: An autonomous tree T in the forest $F = \{T_1, \ldots, T_r, T\}$. Leaf $\hat{\ell}_0$ is mutually shadow-independent of all members of the transversal $\Lambda = \{\ell_1, \ldots, \ell_r\}$ in $F - \{T\}$. Arrows correspond to assignments of σ.

(B) Add to F' each tree in $\hat{F}$ containing a free leaf.
(C) Add to F' each tree in $\hat{F}$ satisfying condition $(*)$ of Lemma 2.
(D) Assign $\hat{F} \leftarrow \hat{F} - F'$ and $\hat{\sigma} \leftarrow \hat{\sigma}|\hat{F}$,
finishing the loop body and also the procedure.
It is obvious that the procedure is purely deterministic and thus yields a unique result for each fixed input instance. We claim that the resulting object $(\hat{F}, \hat{\sigma})$ meets Definition 5: Indeed, suppose $(\hat{F}, \hat{\sigma})$ returned by Procedure KERNEL contains an autonomous tree T (resp. a tree T containing a free leaf). Since the procedure stopped, container F' must be empty. Hence during the last iteration of the loop there was found neither an autonomous tree in Step (C) nor a tree containing a free leaf in Step (B) contradicting that T is autonomous (resp. contains a free leaf). Finally, each tree T that has been removed from (F, σ) during Procedure KERNEL contains a free leaf or satisfies relation $(*)$ of Lemma 2, for $\hat{F} \cup \{T\}$ and correspondingly extended $\hat{\sigma}$. Hence $(\hat{F}, \hat{\sigma})$ cannot be enlarged to an instance that contains a greater subforest of F and has kernel properties completing the proof of (1).
(2) obviously holds true if $F = \varnothing$. For $F \neq \varnothing$, we claim that each iteration of the loop in Procedure KERNEL preserves SIS-status meaning that the current instances at the beginning and at the end of each iteration either both belong to SIS or both do not, obviously implying (2). The claim is proven by induction on the number $n \in \mathbb{N}$ of loop iterations: At the end of the first iteration $n = 1$ we either have (i) $F' = \varnothing$ meaning $\hat{F} = F$, and the procedure stops, or (ii) $\hat{F} = F - F'$ where $F' = F_f \cup F_a \subseteq F$ is a subforest containing trees having a free leaf collected in F_f, or autonomous trees in F collected in F_a. In case (i) there is nothing to prove. In case (ii) it is obvious that if F admits a shadow-independent transversal, so does each subforest of F, especially $\hat{F}$. For the converse direction, assume that $\hat{F}$ admits a shadow-independent transversal Λ but F does not. Let Λ_f denote any transversal in F_f composed of free leaves, then clearly, $\Lambda \cup \Lambda_f$ is a shadow independent transversal in $\hat{F} \cup F_f$. If there is $T \in F_a$ then it satisfies $|\mathcal{H}(T)| \geq 1 + (|F| - 1)(s_\sigma + 1) \geq 1 + (|\hat{F} \cup F_f| - 1)(s_{\sigma|(\hat{F} \cup F_f)} + 1)$. Hence T is autonomous in $\hat{F} \cup F_f$ according to Lemma 2. Therefore $\Lambda \cup \Lambda_f$ can be enlarged to a shadow-independent transversal in $\hat{F} \cup F_f \cup \{T\}$ by Lemma 1. By analogous conclusion for each remaining tree in $F_a - \{T\}$, we obtain a shadow-independent transversal in $\hat{F} \cup F_f \cup F_a = F$ contradicting the assumption and establishing the induction base. If the claim is true at the end of the ith iteration of the loop, for all $i < n$, $n \geq 2$, then it also holds true at the end of its nth iteration by means of an analogous argumentation.
To verify (3), first observe that (β) follows from (α): Assume kernel $(\hat{F}, \hat{\sigma})$ contains at most $\rho(|\hat{F}|, s_{\hat{\sigma}})$ different preenvelopes. Because the number of envelopes in the kernel equals the number of its preenvelopes, and each envelope contains at most $s_{\hat{\sigma}}$ leaves, (β) holds true. So it remains to prove (α). According to (1), $(\hat{F}, \hat{\sigma})$ contains no autonomous tree. Thus, for each $T \in \hat{F}$, we have $|\mathcal{H}(T)| \leq (|\hat{F}| - 1)(s_{\hat{\sigma}} + 1)$ according to Lemma 2. Therefore $(\hat{F}, \hat{\sigma})$ contains at most

$$\sum_{T \in \hat{F}} |\mathcal{H}(T)| \leq |\hat{F}| \cdot (|\hat{F}| - 1)(s_{\hat{\sigma}} + 1) = \rho(|\hat{F}|, s_{\hat{\sigma}})$$

different preenvelope fragments. Because any preenvelope fragment belongs to exactly one preenvelope, the number of preenvelopes in $(\hat{F}, \hat{\sigma})$ is certainly upper bounded by $\rho(|\hat{F}|, s_{\hat{\sigma}})$.□

4 The main algorithm

In this section we provide an algorithm for SIS_κ that works in two parts: After the kernel (F, σ, κ) of an input instance is computed, two dynamic programming steps are performed consecutively to decide whether $(F, \sigma, \kappa) \in \mathrm{SIS}_\kappa$. These dynamic programs are based on Algorithm INITIAL (DP1) and Algorithm GLOBAL (DP2) in [4]. Whereas it turns out that we can use DP2 without modifications, DP1 needs to be adapted for our purposes resulting in DP1'. So we briefly recall the main ideas underlying DP1 to keep the presentation self-contained. Observe that a SIS input instance (F, σ) determines a digraph G, the *shadow graph*: Trees in F correspond to nodes and two nodes T_j, T_i are joined by an arc $T_j \to T_i$ if and only if there is a leaf-vertex in T_j that is mapped by σ to a vertex of T_i (cf. right part of Fig. 4). (We will call the

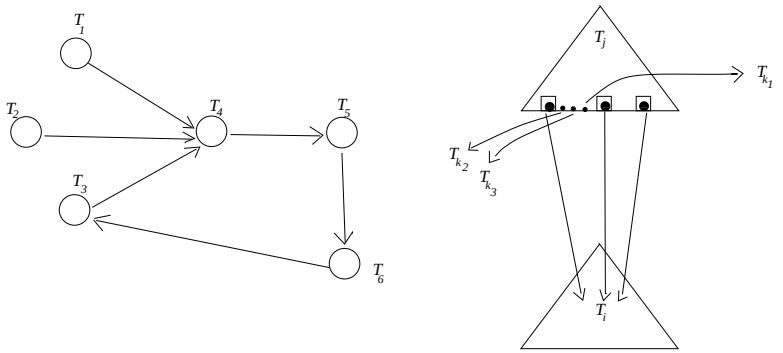

Figure 4: A shadow pattern over the forest $F = \{T_1, \ldots, T_6\}$ (left part), and shadow graph construction (right part): all mappings of leaf-vertices in T_j to vertices in T_i defined through σ are identified yielding arc $T_j \to T_i$ in the shadow graph.

vertices of the shadow graph *nodes*, so that the notion *vertex* is reserved for members in $V(F)$ having no structure. Similarly, we distinguish between the trees in F and directed trees as subgraphs of G called *G-trees*.)

Basic objects in DP1 are specific subgraphs of G, namely *shadow patterns*. A shadow pattern consists of all nodes in G, but selects *exactly one outgoing arc* of each node. Consequently, each connected component of a shadow pattern is a "cycle of G-trees". Figure 4 (left part), e.g., depicts a shadow pattern over $\{T_1, \ldots, T_6\}$, where $\{T_3, \ldots, T_6\}$ are roots of G-trees in a (directed) cycle, and only node T_4 is the root of a non-trivial G-tree with node set $\{T_1, T_2, T_4\}$. Observe that each transversal in F composed of leaves in $D(\sigma)$ determines a unique shadow pattern in G. It turns out to be useful that also the leaf-vertices in $\overline{D(\sigma)}$ can determine shadow patterns. To that end, we simply make σ a total map by assigning to each $\ell \in \overline{D(\sigma)}$ an arbitrary fixed vertex x of another tree, but define the shadow of ℓ to be empty. Hence x is a "dummy" image of ℓ, and the SIS-status of the instance is preserved. Afterwards *each* transversal in F defines a unique shadow pattern. There are only two possibilities for dividing a connected shadow pattern into two parts. E.g., consider the connected shadow pattern depicted in Fig. 4: Either cut one arc of the G-tree rooted at T_4 or cut two arcs of the cycle. Define a *shadow pattern part (spp)* to be an induced (not necessarily proper) *connected* subgraph of a connected shadow pattern. Hence a spp either is a cycle of G-trees called *spp of tree-cycle type* where each node has exactly one outgoing arc, or the spp is a root directed G-tree, where each node has at most one outgoing arc, and the root is the only node having no outgoing arc, called *spp of tree type*. Recursively, a connected induced subgraph of a spp is a spp, too. Thus, reversing the possible dividing operations we can combine two spp's yielding a larger one in exactly two ways: Connect the root of the first one (which must be of tree type) by an arc to any node of the second one (which can be of tree type or of tree-cycle type) yielding a larger spp of tree type, resp. of tree-cycle type. Or two spp's of tree type are combined by connecting either root via an arc to a node of the other spp yielding a spp of tree-cycle type. Abstractly a triple (W, o, i) defines a spp of tree type: $W \subset F$ is its node set, $o \in L(W)$ defines its root node $T(o) \in W$, and $i \in I(\sigma) \cap V(W)$ is any selected image vertex. We call o an *out-object* because it can define the source of an arc combining (W, o, i) to another spp, and we call i an *in-object* because it can define the sink of an arc coming from another spp. i is also allowed to be the empty set, then $(W, o, \varnothing)$ can be combined to another spp only in the first way as stated above. If o and i both are empty, then $(W, \varnothing, \varnothing)$ corresponds to a connected spp of tree-cycle type which cannot be enlarged by further combinations, since no source or sink vertices are reserved for combining arcs. Triples $(W, \varnothing, i)$ correspond to spp's of tree-cycle type where i is the sink of an arc coming from a spp of tree type. However, such configurations do not need to be considered, see below. Now suppose there are shadow-independent transversals inducing spp's of tree type in subforests W resp. W' with $W \cap W' = \varnothing$. For testing whether these spp's can be combined to one over $W \cup W'$ without disturbing shadow-independency, we have to try whether there are leaf-vertices $o \in L(W)$ resp. $o' \in L(W')$ which yield arcs into W' resp. W such that their shadows do not contain members of the transversals in W, W'. Starting with single tree subforests, in that way, it was possible in DP1 to test each subforest W of the input instance inductively whether it admits a spp induced by a shadow-independent transversal.

We now provide adaptation DP1' of DP1 yielding a bound of kernel form for SIS_κ instances. Instead of testing each single leaf-vertex of the kernel as out-object and each single image vertex i as in-object, we test *only leaves in L_B* as single out-objects, and only images of leaves in L_B as single in-objects. Leaves outside L_B are treated as *classes*, namely all leaves of the same 0-fragment of a preenvelope are tested simultaneously. Hence whole 0-fragments are out-objects and their envelope images are in-objects. So we define a set $\mathcal{O}$ of *generalized* out-objects

and a set $\mathcal{I}$ of *generalized* in-objects:

$$\mathcal{O} := L_B \cup \mathcal{H}^0(F) \cup \{e_o\} \quad \text{and} \quad \mathcal{I} := \sigma(L_B) \cup B \cup \{e_i\}$$

where e_o is the empty out-object and e_i is the empty in-object being instances of the empty set, and σ is assumed to be total. For subforest $W \subseteq F$, we define $\mathcal{O}_W := L_B(W) \cup \mathcal{H}^0(W)$, and $\mathcal{I}_W := \sigma(L_B(W)) \cup (B \cap V(W))$, with $L_B(W) := L_B \cap L(W)$.

Definition 6 *We extend σ (assumed to be made total) to a mapping $\Sigma : \mathcal{O} \to \mathcal{I}$ by*

$$\Sigma(O) := \left\{ \begin{array}{ll} \sigma(\ell), & O = \ell \in L_B \\ b, & O = H^0_T(b) \in \mathcal{H}^0(F) \\ e_i, & O = e_o \end{array} \right\} \in \mathcal{I}$$

If $O \in L_B$, and $I = \Sigma(O')$, for $O' \in \mathcal{H}^0(F)$, then for short we write $O \notin L(S_I)$ if there exists $\ell \in O'$ such that $O \notin L(S_{\sigma(\ell)})$, and we write $O \in L(S_I)$ otherwise. For $O' \in L_B$, expression $O \notin L(S_I)$, resp. $O \in L(S_I)$, has the usual meaning as $I = \sigma(O')$ is a unique image vertex then.

Definition 7 *For* SIS *input instance (F, σ) and Σ as above, define $Q : 2^F \times \mathcal{O} \times \mathcal{I} \to \{0, 1\}$ by $Q(W, O, I) = 1$ iff the following conditions are satisfied simultaneously:*
(i) $W \neq \varnothing, O \in \mathcal{O}_W \cup \{e_o\}, \Sigma(O) \in \mathcal{I}_{\overline{W}} \cup \{e_i\}$, there exists $O' \in \mathcal{O}_{\overline{W}} \cup \{e_o\}$ such that $I = \Sigma(O') \in \mathcal{I}_W \cup \{e_i\}$,
(ii) there exists a shadow-independent transversal Λ in W inducing a shadow pattern part over W such that $O \cap \Lambda \neq \varnothing$, (in case O is a single leaf ℓ, $O \cap \Lambda$ is identified with $\{\ell\} \cap \Lambda$)
(iii) for each $\ell \in \Lambda$ it holds that $\ell \notin L(S_I)$.

Algorithm DP1':
Input: SIS-kernel (F, σ) with $|F| \geq 2$, $s_\sigma \geq 1$.
Output: Values $Q(U) := Q(U, e_0, e_i), U \subseteq F, |U| \geq 2$.
(A): Initial values: $\forall T \in F, \forall (O, I) \in \mathcal{O}_{\{T\}} \times \mathcal{I}_{\{T\}}$ compute

$$Q(T, O, I) = \left\{ \begin{array}{l} 1, \; I = e_i \vee O \in \mathcal{H}^0(T) \vee (O \in L_B(\{T\}) : O \notin L(S_I)) \\ 0, \text{ else} \end{array} \right.$$

(B): $\forall U \subseteq F : |U| \geq 2, \forall O \in \mathcal{O}_U$ compute:

$$Q(U, O, e_i) = \bigvee_{\substack{W \subset U: \\ T(O) \notin W}} \; \bigvee_{\substack{O' \in \mathcal{O}_W: \\ \Sigma(O') \in \mathcal{I}_{\{T(O)\}} \\ \wedge O \notin L(S_{\Sigma(O')})}} Q(W, O', e_i) \wedge Q(\overline{W}, O, e_i)$$

(C): $\forall U \subseteq F : |U| \geq 2$ compute:

$$\begin{array}{rcl} \forall O & \in & \mathcal{O}_U, \forall I \in \mathcal{I}_{\{T(O)\}} : \\ Q(U, O, I) & = & \left\{ \begin{array}{ll} 1, & Q(U, O, e_i) = 1 \wedge O \notin L(S_I) \\ 0, & \text{else} \end{array} \right. \\ \\ \forall O & \in & \mathcal{O}_U, \forall I \in \mathcal{I}_U, T(I) \neq T(O) : \\ Q(U, O, I) & = & \displaystyle\bigvee_{\substack{W \subset U: \\ T(O) \notin W \\ \wedge T(I) \in W}} \; \bigvee_{O' \in \mathcal{O}_{\{T(I)\}}} Q(W, O', I) \wedge Q(\overline{W}, O, \Sigma(O')) \end{array}$$

(D): For each $U \subseteq F : |U| \geq 2$ compute:

$$\begin{array}{rcl} Q(U, e_o, e_i) & = & \displaystyle\bigvee_{W \subset U} \; \bigvee_{O, O' \in \mathcal{O}_U} Q(W, O, \Sigma(O')) \wedge Q(\overline{W}, O', \Sigma(O)) \\ & =: & Q(U) \end{array}$$

Theorem 1 *Let (F, σ) be the kernel of a* SIS *input instance, where σ has been made total. Assume that, for each pair $(O, I) \in \mathcal{O} \times \mathcal{I}$, the information whether $O \in L(S_I)$ can be accessed in $O(1)$ time. Via DP1' followed by DP2, i.e. Algorithm GLOBAL in [4], can be decided in $O([s_\sigma \cdot \rho(|F|, s_\sigma)]^3 3^{|F|})$ time whether (F, σ) admits a shadow-independent transversal, where $\rho(x, y) := x(x-1)(y+1)$.*

PROOF. Observe that a kernel containing only one tree cannot exist, because this tree is autonomous and thus is removed via Procedure KERNEL. So a kernel either is empty and therefore trivial or has size at least two which is assumed for the input of DP1'.
For proving correctness of DP1', we have to show that in Step (D) value $Q(U) := Q(U, e_0, e_i)$ is correctly calculated for each $U \subseteq F$. That means $Q(U) = 1$ if there exists a shadow-independent transversal in U inducing a *connected* spp (U, e_o, e_i), and $Q(U) = 0$ otherwise. (U, e_o, e_i) corresponding to a spp of tree-cycle type can be composed either via connecting two spp's $(W, O, \Sigma(O'))$, $(\overline{W}, O', \Sigma(O))$ of tree type, for $O \in \mathcal{O}_W$, $O' \in \mathcal{O}_{\overline{W}}$, and $W \subset U$ as calculated in Step (D). Or it can be combined via connecting a spp (W, O, e_i) of tree type and a spp $(\overline{W}, e_o, \Sigma(O))$ of tree-cycle type, for $O \in \mathcal{O}_W$. Observe that the latter combination type needs not to be considered because it is covered already by a combination of the first kind: Indeed, replace W by W' with $W \subset W' \subseteq U$ containing the root of that G-tree in the cycle of $(\overline{W}, e_o, \Sigma(O))$ which contains the node $\Sigma(O)$.
As long as O and O' both are single leaves in $L_B(U)$ correctness of (D) is justified by the arguments above. If O or O' is a 0-fragment we have to verify that no conflict can occur when simultaneously treating all leaves contained in O resp. O'. If both $O, O' \in \mathcal{H}^0(U)$ this is obvious because no leaf in O resp. O' is contained in the shadow of any leaf in O' resp. O. If w.l.o.g. $O \in \mathcal{H}^0(U)$ and $O' \in L_B(U)$, we only need to decide whether there is a leaf $\ell \in O$ such that $O' \notin L(S_{\sigma(\ell)})$ which without conflict tells us the value $Q(\overline{W}, O', \Sigma(O))$, by definition. Therefore $Q(W, O, \Sigma(O')) \wedge Q(\overline{W}, O', \Sigma(O))$ is correctly computed. Observe that a conflict could occur only if leaves of 0-fragments O, O' lied in shadows: $Q(W, O, \Sigma(O')) \wedge Q(\overline{W}, O', \Sigma(O)) = 1$ then would mean that there are $\ell_1 \in O, \ell_1' \in O'$ such that $\ell_1 \notin L(S_{\sigma(\ell_1')})$ and simultaneously that there are $\ell_2 \in O, \ell_2' \in O'$ such that $\ell_2' \notin L(S_{\sigma(\ell_2)})$. But there was no guarantee that $\ell_1 = \ell_2$ and $\ell_1' = \ell_2'$ which could obviously yield a conflict in case that e.g. ℓ_1 is contained in the shadow of ℓ_2. So we are sure to obtain no conflict because leaves in shadows, i.e. in $L_B(U)$, are touched as single objects and not

as classes.
To have made σ a total map without disturbing the shadow structure as described above guarantees that we touch each leaf in L_B as an out-object and therefore no possible candidate for a shadow-independent transversal is missed that might induce a connected spp. Clearly only leaves in L_B need to be considered for making σ a total map because the kernel contains no free leaves.
In Step (A), Step (B) and Step (C) all values necessary for Step (D) are provided. Correctness of Step (A) is obvious. Correctness of Steps (B) and (C) can be proven analogous to the argumentation above.
All calculations in (A) - (D) are of the same structure as those in DP1 for which the bound $O(n^3 \cdot 3^{|F|})$ has been proven ([4], Proposition 4.2) under the assumption that all operations for calulating values of Q can be performed in constant time. The only crucial part regarding these computations in DP1' concerns the decision whether $O \in L(S_I)$ or $O \notin L(S_I)$, for $O \in \mathcal{O}, I \in \mathcal{I}$, which has assumed to be managable in $O(1)$ time. (In the proof of Theorem 2 below we will provide a preprocedure making available these information appropriately.) Factor n in the bound stated above is the number of all vertices in F which is an upper bound for the number of all leaves in F, resp. the number of all image vertices w.r.t. the shadow map. Recall that leaves are the out-objects, and their images are the in-objects in DP1. Hence, to obtain the time complexity for calculating (A) - (D) we have to replace n by $|\mathcal{O}| \leq |L_B| + |\mathcal{H}^0(F)| + 1$ upper bounding the number of out- resp. in-objects in DP1'. According to Prop. 1 (3)(α), we have

$$|\mathcal{H}^0(F)| = \sum_{T \in F} |\mathcal{H}^0(T)| \leq \sum_{T \in F} |\mathcal{H}(T)| \leq \rho(|F|, s_\sigma)$$

and according to Prop. 1 (3)(β) we have $|L_B| \leq s_\sigma \cdot \rho(|F|, s_\sigma)$. Therefore, and because $s_\sigma \geq 1$, we have $|\mathcal{O}| \in O(s_\sigma \cdot \rho(|F|, s_\sigma))$, hence DP1' runs in $O([s_\sigma \cdot \rho(|F|, s_\sigma)]^3 3^{|F|})$ worst case time.
If during DP1' no shadow-independent transversal is found in F then there still can exist such a transversal inducing a *disconnected* shadow pattern. That means $(**)$: for appropriate $j \in \mathbb{N}$ there can exist a partition $F = U_1 \cup \cdots \cup U_j$ such that $Q(U_i) = 1$ with $|U_i| \geq 2$, for each $1 \leq i \leq j$. Since the values $Q(U_i)$ are independent of out- and in-objects, we are allowed to use dynamic program DP2 (i.e. Algorithm GLOBAL in [4]) without paying attention to 0-fragments as out-objects specifically. DP2 gets as input all values $Q(U)$, $U \subset F$, and outputs 1 if $(**)$ holds true, 0 otherwise. Correctness of DP2 as well as its worst case time complexity of $O(|F| \cdot 3^{|F|})$ are stated in [4], Prop. 4.1. Thus the running time of DP2 is dominated by that of DP1' completing the proof. □

So far we only know how to treat the kernel, next we state the final result presenting a bound for a general input instance:

Theorem 2 *Whether* $(F, \sigma, \kappa = (k, s)) \in \mathrm{SIS}_\kappa$ *can be decided in* $O(k \cdot n^2 + [s \cdot \rho(k, s)]^3 3^k)$ *time, where* $n = |V(F)|$ *and* $\rho(x, y) := x(x-1)(y+1)$.

Proof. Given input instance $(F, \sigma, \kappa = (k, s))$ we claim that in time $O(k \cdot n^2)$ we can compute its kernel, provide all information that DP1' needs to access in constant time, and moreover that we can decide whether $|F| \leq k$ and $s_\sigma \leq s$. This claim implies the theorem using DP1' and DP2 according to Theorem 1. We prove the claim via the preprocedure stated below. To that end, suppose that σ is represented, for each fixed $T \in F$, by array σ_T of length $|L(T)|$ which for each leaf of T in prescribed order contains its image vertex if existing and **nil** otherwise. We further assume that each vertex $x \in V(F)$ "knows" the tree $T(x) \in F$ it belongs to.

Preprocedure:
(1) For each fixed $T \in F$, maintain array A_T of Boolean having length $|V(T)|$ where vertices in $V(T)$ are assumed to have a prescribed order. Running once through each σ_T, $T \in F$, we fill all arrays A_T simultaneously: if $\sigma_T(\ell) = x$ we assign $A_{T(x)}(x) := 1$. Afterwards running once through each A_T, we assign 0 to all entries still having no value. Hence, A_T, $T \in F$, can be filled in $O(n)$ time. Observe that A_T contains the information which vertex of T appears as image of the shadow map.
(2) For each fixed $T \in F$, hold array B_T of Boolean having length $|V(T)|$, and array N_T of Boolean having length $|L(T)|$. These containers are filled via depth first search (DFS) as follows: For each $\ell \in L(T)$, start DFS in T, and for each vertex x on the path from ℓ to the root of T check in constant time whether $A_T(x) = 1$, i.e. ℓ lies in the shadow of x, and store for each ℓ the last current such vertex x in $last(\ell)$. After all DFS rounds were finished, assign $N_T(\ell) := 1$ iff $last(\ell) = \varnothing$, i.e. ℓ is contained in no shadow at all. Otherwise, assign $B_T(last(\ell)) := 1$, meaning that vertex $last(\ell)$ is an envelope root in that case. All other entries of $B_T, T \in F$, get value 0. Because each DFS runs in $O(n)$ time, Step (2) can be performed in $O(n^2)$ time.
(3) For each $x \in V(F) : B_{T(x)}(x) = 1$, compute by DFS set $L(S_x)$ of all leaves in envelope S_x and store cardinality $|L(S_x)|$. If there is $x \in V(F)$ s.t. $s < |L(S_x)|$ then reject the instance and stop. Hold array U of length k and for each $T \in F$ assign $U(T) := \max\{|L(S_x)| : x \in V(T) : B_T(x) = 1\}$. Assign $s_\sigma := \max_{T \in F} U(T)$; if $s_\sigma = 0$ then accept the instance and stop. Step (3) can be performed in $O(n^2)$ time.
(4) Computation of the number of preenvelope fragments $c_T := |\mathcal{H}(T)|$ in T, for all trees $T \in F$ simultaneously: Initially set $c_T := 0$, for each $T \in F$. For each fixed $x \in V(F) : B_{T(x)}(x) = 1$ do: for each $T \in F - \{T(x)\}$ do: if there is a first $\ell \in L(T)$ s.t. $\sigma_T(\ell) = x$ then increment $c_T := c_T + 1$ and continue with the next tree in $F - \{T(x)\}$. Step (4) can be executed in $O(n^2)$ time.
(5) Compute kernel $(\hat{F}, \hat{\sigma})$ of (F, σ) according to Proce-

dure KERNEL in proof of Prop. 1 (1): Since the values s_σ and c_T, $T \in F$, are already computed in Steps (3) and (4) above, we can find all initially existing autonomous trees in the input instance in time $O(k)$. For finding all trees initially containing a free leaf, we simply check, for each $T \in F$, whether there is $\ell \in L(T)$ s.t. $N_T(\ell) = 1$ and $\sigma_T(\ell) = \mathbf{nil}$, needing $O(n)$ time. Now assume that during an iteration of KERNEL subforest $W \neq \varnothing$ is detected to contain autonomous trees resp. free leaves w.r.t. the current instance $(\hat{F}, \hat{\sigma})$. Then we first have to calculate the new shadow map $\hat{\sigma}|(\hat{F} - W)$: For each $\ell \in L(\hat{F} - W)$, assign $\sigma_{T(\ell)}(\ell) \leftarrow \mathbf{nil}$ iff $\sigma_{T(\ell)}(\ell) \in V(W)$ needing $O(n)$ time. Now remove $T, B_T, N_T, A_T, \sigma_T$ for each $T \in W$. Finally, execute Steps (1) - (4) for updating $A_T, B_T, N_T, U(T)$, and c_T for each $T \in \hat{F} - W$ w.r.t. $\hat{\sigma}|(\hat{F}-W)$. If the forest of the kernel is empty then accept the input instance and stop. Since KERNEL performes at most $|F| \leq k$ rounds, and each iteration needs at most $O(n^2)$ time, Step (5) can be executed within $O(k \cdot n^2)$ time.

(6) Let (F, σ) be the kernel of the input instance as computed previously. Maintain table R of Boolean having size $|L(F)| \cdot |V(F)|$. For each leaf $\ell \in L(F)$, start DFS in $T(\ell)$, and for each vertex on the path from ℓ to the root of $T(\ell)$, assign $R(\ell, x) := 1$ if $A_{T(\ell)}(x) = 1$, otherwise assign 0. Step (6) can be performed in $O(n^2)$ time.

(7) For each $x \in V(F) : B_{T(x)}(x) = 1$, and each $T \in F$ fill set $H^0_T(x)$ (initially assumed to be empty) as follows: For each $\ell \in L(T)$ s.t. $\sigma_T(\ell) = x$, add ℓ to $H^0_T(x)$ iff $N_T(\ell) = 0$ meaning that ℓ is contained in no shadow and thus is a member of the 0-fragment $H^0_T(x)$. Step (7) can be executed in $O(n^2)$ time.

(8) Maintain table Z of Boolean having size $|L(F)| \cdot |\mathcal{H}^0(F)|$. For each $x \in V(F) : B_{T(x)}(x) = 1$, consider each fixed pair $(\ell, H^0_T(x))$ where $\ell \in L(S_x)$ and $H^0_T(x) \in \mathcal{H}^0(F), T \in F - \{T(x)\}$. Then, for each $\ell' \in H^0_T(x)$ check whether $R(\ell, \sigma_T(\ell')) = 0$, i.e., whether ℓ does not lie in the shadow of ℓ'. If a first such $\ell' \in H^0_T(x)$ can be found for ℓ then assign table entry $Z(\ell, H^0_T(x)) := 0$, otherwise set it to 1. Finally, assign value 0 to all entries of Z still having no value. Since $|\mathcal{H}^0(F)| \in O(k \cdot n)$, Step (8) can be performed in $O(k \cdot n^2)$.

(9) Finally, σ is modified to a total map as follows: Let $F = \{T_0, \ldots, T_{|F|-1}\}$. Run through each $\sigma_T, T \in F$: if $\sigma_T(\ell) = \mathbf{nil}$, and $\ell \in T_i$, then assign the root of $T_{i+1 \bmod |F|}$ to $\sigma_T(\ell)$. Observe that the information about the shadow structure does not be affected since the tables R and Z have already been constructed before. Step (9) can be executed in $O(n)$ time.

As shown, the time complexity of each step of the preprocedure is bounded by $O(k \cdot n^2)$. It correctly computes the kernel of the input instance according to Prop. 1, and makes σ a total map without modifying the shadow structure. Moreover we observe that in the tables R and Z all information is collected that DP1' needs to get access to in constant time completing the proof. □

5 Concluding remarks and open problems

So far, problem SIS, parameterized by k only, did not reveal a kernelization, especially the FPT-bounds provided in [2, 4] do not have kernel form. The reason might be that SIS input instances consist of two components, a forest and a shadow map, that essentially are independent of each other. In this paper we provided a kernelization and a FPT algorithm for SIS running in $O(k \cdot n^2 + [s \cdot \rho(k, s)]^3 3^k)$ time. The price to pay was the introduction of an additional parameter s bounding the shadow size. The new parameterization thus controls both components. An open question is, whether there is a completely different algorithmic approach to SIS_k running faster and exhibiting a kernelization based on parameter k only. A further feature of such an approach should be polynomial space amount, which cannot be achieved in the framework of dynamic programming. A distinct future work direction is to check whether the methods presented here can be extended to obtain also kernelizations for the generalized versions of the shadow problem as recently discussed in [5]. There the shadow map is replaced by a *shadow relation* and a fixed-parameter tractable algorithm w.r.t. standard parameter k is presented.

References

[1] Downey, R.G., Fellows, M.R., *Parameterized Complexity*, Springer-Verlag, New York, 1999.

[2] Franco, J., Goldsmith, J., Schlipf, Speckenmeyer, E., Swaminathan, R.P., "An algorithm for the class of pure implicational formulas," *Discrete Appl. Math.*, V96, pp. 89-106, 1999.

[3] Golumbic, M.C., *Algorithmic Graph Theory and Perfect Graphs*, Academic Press, New York, 1980.

[4] Heusch, P., Porschen, S., Speckenmeyer, E., "Improving a fixed-parameter tractability time bound for the shadow problem," *J. Comp. Syst. Science*, V67, pp. 772-788, 2003.

[5] Porschen, S., "On generalizations of the shadow independent set problem, " *Discrete Math.*, V37, pp. 1473-1485, 2007.