To appear in *Optimization Letters*,
http://dx.doi.org/10.1007/s11590-011-0311-5.

Parameterized Computational Complexity of Finding Small-Diameter Subgraphs*

Alexander Schäfer · Christian Komusiewicz · Hannes Moser · Rolf Niedermeier

the date of receipt and acceptance should be inserted later

Abstract Finding subgraphs of small diameter in undirected graphs has been seemingly unexplored from a parameterized complexity perspective. We perform the first parameterized complexity study on the corresponding NP-hard s-Club problem. We consider two parameters: the solution size and its dual.

Keywords Clique relaxation · s-club · NP-hard problem · problem kernel · polynomial-time preprocessing · branching algorithm

1 Introduction

The search for cohesive subgraphs is an important task in network analysis. The standard model for a cohesive subgraph is the complete subgraph, called clique. In many applications, however, the clique requirement is too strict. To deal with this problem, the application of clique relaxations has been proposed [1, 3, 21]. Cliques are precisely the graphs that have diameter 1. We study a concept that relaxes this diameter requirement, so-called s-clubs: an *s-club* is a graph that has diameter at most s [1]. Hence, 1-clubs are cliques; in this

* The main work for this article was done while all authors were with the Friedrich-Schiller-Universität Jena.

The results of this article are part of the first author's diploma thesis [20].

C. K. was supported by a PhD fellowship of the Carl-Zeiss-Stiftung and by the Deutsche Forschungsgemeinschaft, project PABI, NI 369/7.

H. M. was supported by the Deutsche Forschungsgemeinschaft, project AREG, NI 369/9.

A. Schäfer
Max Planck Institute for Human Cognitive and Brain Sciences, Department of Neurology, Stephanstraße 1A, D-04103 Leipzig, Germany, E-mail: aschaefer@cbs.mpg.de

C. Komusiewicz · R. Niedermeier
Institut für Softwaretechnik und Theoretische Informatik, Technische Universität Berlin, D-10587 Berlin, Germany E-mail: {christian.komusiewicz, rolf.niedermeier}@tu-berlin.de

work we study s-clubs for $s > 1$. The s-club concept was defined in the context of social sciences [1], and it has recently been used in the analysis of social [17] and biological [19] networks. The task of finding an s-club of order k can be formalized as follows:

s-Club
Input: An undirected graph $G = (V, E)$ with integer $k \geq 1$.
Question: Is there a set of vertices $S \subseteq V$ of size k such that $G[S]$ has diameter at most s?

s-Club is NP-complete [7], even in graphs of diameter $s + 1$ [3]. For $s \geq 2$, every graph contains an s-club of order $\Delta(G)+1$, where $\Delta(G)$ is the maximum degree of G, namely the graph that is induced by the closed neighborhood of a vertex with maximum degree. Hence, s-Club is nontrivially posed only if $k > \Delta(G) + 1$. In contrast, for every constant $s \geq 2$, there is no polynomial-time algorithm for deciding whether a graph G has an s-club of order at least $\Delta(G) + 2$, unless P=NP [8]. Finding a maximum-cardinality s-club is inapproximable within a factor of $|V|^{1/3-\epsilon}$, for any $\epsilon > 0$, unless P=NP [16]. For even s, the inapproximability result has been strengthened to a factor of $|V|^{1/2-\epsilon}$ [2]. Integer linear programs [3, 7] as well as heuristics [6] have been proposed for s-Club. Since 1-Club is equivalent to Clique, it is W[1]-hard with respect to the parameter solution size [9].

We initiate the study of s-Club for $s \geq 2$ within the framework of parameterized computational complexity [10, 12, 18]. We study two parameters and their influence on the computational complexity of s-Club: the solution size k and its dual $d := |V| - k$, which we refer to as "size of the deletion set". For parameter k as well as parameter d we obtain fixed-parameter tractability. This contrasts the W[1]-completeness of 1-Club or, equivalently, Clique parameterized by the solution size k. In detail, our results are as follows: for the parameter k the problem does not admit a polynomial-size many-to-one kernel but a k^2-vertex Turing kernel for even s and a k^3-vertex kernel for odd s. This shows the potential and limitations of polynomial-time preprocessing and contributes to the currently very short list of parameterized problems on general graphs for which both nonexistence of a polynomial-size problem kernel and existence of a polynomial-size Turing kernel are known.[1] Furthermore, we show that s-Club can be solved in $O((k-2)^k \cdot k! \cdot k^3|V| + |V||E|)$ time by a branching algorithm. For the dual parameter d, we present a simple branching algorithm that runs in $O(2^d \cdot |V||E|)$ time.

Preliminaries. We only consider undirected graphs $G = (V, E)$, with vertex set V and edge set $E \subseteq \{\{u, v\} \mid u, v \in V\}$. Throughout this work, let $n := |V|$ and $m := |E|$. Furthermore, we assume that $n = O(m)$ since we can remove isolated vertices in linear time from G. The *distance* $d(u, v)$ between two vertices u and v is the length of a shortest path between u and v. The *diameter* of a graph G is the maximum of all distances between any two vertices

[1] To our knowledge, k-Leaf Out-Branching and k-Leaf Out-Tree are so far the only other problems for which this has been shown [11].

u and v of G. The (open) *i-neighborhood* $N_i(v) := \{u \mid 1 \leq d(u,v) \leq i\}$ of v is the set of vertices that have distance at most i to v. The *closed i-neighborhood* of v is $N_i[v] := N_i(v) \cup \{v\}$. The *exact i-neighborhood* $N_i^\circ(v) := \{u \mid d(u,v) = i\}$ of v is the set of vertices that have distance exactly i to v. For a vertex set S, we use $G[S]$ to denote the subgraph of G induced by S having edge set $E' = \{\{u,v\} \in E \mid u,v \in S\}$.

Parameterized complexity [10, 12, 18] is a two-dimensional framework for studying the computational complexity of problems. One dimension is the input size n (as in classical complexity theory), and the other one is a parameter k. A parameterized problem L is *fixed-parameter tractable* if there is an algorithm that decides in $f(k) \cdot \text{poly}(n)$ time whether $(x,k) \in L$, where f is a computable function depending only on k. Data reduction aims at the elimination of polynomial-time "solvable" parts of the input data, to obtain "hard" cores [4, 14]. A parameterized problem L admits a *(many-to-one) problem kernel* if there is a polynomial-time transformation of any instance (I,k) to an instance (I',k') such that $(I,k) \in L \Leftrightarrow (I',k') \in L$, $|I'| \leq g(k)$, and $k' \leq k$. A problem kernel is *polynomial* if $g(k)$ is a polynomial function. While a parameterized problem is fixed-parameter tractable if and only if it has a problem kernel [10], polynomial many-to-one kernels presumably cannot be obtained for every fixed-parameter tractable problem [5]. An extension of the classical problem kernel definition are so-called *Turing kernels*. We use a definition of Turing kernels given by Lokshtanov [15]. It relies on the notion of t-oracles: A t-oracle for a parameterized problem L is an "oracle" that takes as input (I,k) with $|I| \leq t$, $k \leq t$, and decides $(I,k) \in L$ in constant time. A parameterized problem L admits *a $g(k)$-size Turing kernel* if there is an algorithm which, given an input (I,k) together with a $g(k)$-oracle for L, decides whether $(I,k) \in L$ in time polynomial in $|I|$ and k. A Turing kernel is polynomial if g is a polynomial function.

2 Turing Kernels for the Parameter Solution Size

Using a methodology due to Bodlaender et al. [5], we show that s-Club does not admit a polynomial (many-to-one) problem kernel. It relies on the notion of compositional parameterized problems for which there can be no polynomial problem kernels under some complexity-theoretic assumptions. Afterwards, we contrast this result by showing a polynomial Turing kernel for s-Club.

Lemma 1 *[5, 13] Let L be a compositional parameterized problem whose "unparameterized version $\hat{L}$" is NP-complete. Then, L does not admit a polynomial kernel unless* $\text{NP} \subseteq \text{coNP/poly}$.

We use the following characterization of compositional graph problems for showing that s-Club is compositional:

Lemma 2 *[5] Let L be a parameterized graph problem such that for any pair of graphs G_1 and G_2, and any integer $k \in \mathbb{N}$, we have*

$$((G_1,k) \in L \vee (G_2,k) \in L) \Leftrightarrow (G_1 \uplus G_2, k) \in L,$$

where $G_1 \uplus G_2$ is the disjoint union of G_1 and G_2. Then L is compositional.

Lemma 3 *s-Club is compositional.*

Proof In order to apply Lemma 2, we show that (G_1, k) or (G_2, k) is a yes-instance $\Leftrightarrow$ $(G_1 \uplus G_2, k)$ is a yes-instance. The "$\Rightarrow$"-direction follows from the fact that every subgraph of G_1 or G_2 is also a subgraph of $G_1 \uplus G_2$. The "$\Leftarrow$"-direction follows from the fact that every s-club is connected and that every connected subgraph of $G_1 \uplus G_2$ is either a subgraph of G_1 or a subgraph of G_2. □

By using the facts that s-Club is compositional, that its unparameterized version is NP-complete [7], and by applying Lemma 1, we arrive at the following.

Theorem 1 *s-Club does not admit a polynomial kernel with respect to the parameter "solution size" unless* NP $\subseteq$ coNP/poly.

Next, we contrast this result by showing a polynomial Turing kernel for s-Club, $s \geq 2$. The following simple observation is the basis for our Turing kernelization for s-Club:

Observation 1 *If a vertex v is part of an s-club S, then only vertices in the s-neighborhood of v can also be part of S.*

The main idea of our Turing kernelization is that either s-Club has an "easy-to-find" solution or the size of the s-neighborhood of each vertex in G is bounded from above by a function polynomial in k. The following rule finds these easy order-k s-clubs of a graph:

Rule 1 *If there exists a vertex $v \in V$ with $|N_{\lfloor s/2 \rfloor}(v)| \geq k - 1$, then return "yes".*

Lemma 4 *Rule 1 is correct and can be exhaustively performed in $O(nm)$ time.*

Proof Let v be a vertex with $|N_{\lfloor s/2 \rfloor}(v)| \geq k - 1$ and let $S := N_{\lfloor s/2 \rfloor}[v]$. We show that $G[S]$ is an s-club, that is, the pairwise distance of any two vertices $x, y \in N_{\lfloor s/2 \rfloor}[v]$ is at most s. Since $x, y \in N_{\lfloor s/2 \rfloor}[v]$ and by the definition of $N_{\lfloor s/2 \rfloor}[v]$, the distance from x to v is at most $\lfloor s/2 \rfloor$ and the distance from v to y is at most $\lfloor s/2 \rfloor$. Consequently, the distance between x and y is at most s. This proves the correctness of Rule 1. As to the running time, computing $N_{\lfloor s/2 \rfloor}[v]$ can be done in $O(m)$ time for each vertex by breadth-first search. Overall, Rule 1 thus runs in $O(nm)$ time. □

Obviously, Rule 1 can be modified to output the corresponding s-club as well. In the following, we describe how to obtain a polynomial Turing kernel. For each vertex v of G, we query the t-oracle for an s-club of order k within $N_s[v]$. Due to Observation 1 it is obvious that there exists an s-club of order k in G if and only if there exists an s-club of order k in one of these s-neighborhoods. Hence, the Turing kernel is correct. It remains to bound the

size of each s-neighborhood by some function $g(k)$. Therefore, we assume in the following that Rule 1 has been applied and show that this upper-bounds the s-neighborhood of each vertex $v \in V$. For this we distinguish between even and odd values of s. First, we show that s-Club admits a k^2-vertex Turing kernel if s is even, then we show that s-Club admits a k^3-vertex Turing kernel if s is odd.

Using the exact i-neighborhood $N_i^\circ(v)$, we observe that $N_s[v]$ is expressible by the union of two sets:

Lemma 5 *When s is even, $N_s[v] = N_{s/2}[v] \cup N_{s/2}(N_{s/2}^\circ(v))$.*

Proof Lemma 5 can be seen as follows. By definition, any vertex that has distance at most $s/2$ from v is in $N_{s/2}[v]$. Any vertex x with $s/2 \leq d(v,x) \leq s$ has distance at most $s/2$ to at least one vertex in $N_{s/2}^\circ(v)$ and thus belongs to $N_{s/2}(N_{s/2}^\circ(v))$. □

Using Lemma 5, we show the following.

Theorem 2 *For even s, s-Club admits a k^2-vertex Turing kernel, which can be computed in $O(nm)$ time.*

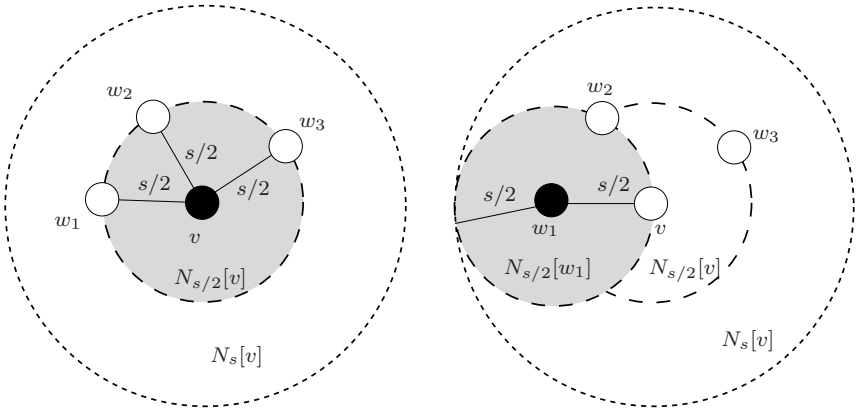
Proof The Turing kernelization consists of applying Rule 1, and then creating for each $v \in V$ a graph that is induced by $N_s[v]$. The running time follows from Lemma 4 and the fact that for each $v \in V$ the graph induced by $N_s[v]$ can be constructed in $O(m)$ time, by first applying breadth-first search to obtain $N_s[v]$ and then adding the edges of G with both of endpoints in $N_s[v]$.

It remains to bound the number of vertices in $N_s[v]$. Using Lemma 5, this can be done by bounding the sizes of $N_{s/2}[v]$ and $N_{s/2}(N_{s/2}^\circ(v))$. After Rule 1 has been applied, $N_{s/2}[v]$ contains at most k vertices, as illustrated in Figure 1(a). The size of $N_{s/2}^\circ(v)$ is thus also at most $k-1$. Furthermore, for each vertex $w \in N_{s/2}^\circ(v)$, the size of $N_{s/2}(w)$ is also upper-bounded by $k-1$, as illustrated in Figure 1(b). Thus, the size of $N_{s/2}(N_{s/2}^\circ(v))$ is at most k^2-2k+1. Hence, the size of $N_{s/2}[v] \cup N_{s/2}(N_{s/2}^\circ(v))$ is at most $k + k^2 - 2k + 1 \leq k^2$. By Lemma 5, $|N_s[v]| \leq k^2$ follows. □

For odd s, the argumentation is different: instead of bounding the size of two subsets of $N_s[v]$, we now bound the size of three subsets. We denote $N_{\lfloor s/2 \rfloor}$ by $N_{(s-1)/2}$ in the following; this is equivalent since s is odd. Using the $(s-1)/2$-neighborhood, we can express $N_s[v]$ by the union of three sets:

Lemma 6 $N_s[v] \subseteq N_{(s-1)/2}[v] \cup N_{(s-1)/2}(N_{(s-1)/2}^\circ(v)) \cup N_{(s-1)/2}(N_{s-1}^\circ(v))$.

Proof By definition, any vertex with distance at most $(s-1)/2$ to v is in $N_{(s-1)/2}[v]$. Any vertex y with $(s-1)/2 \leq d(v,y) \leq s-1$ has distance at most $(s-1)/2$ to at least one vertex in $N_{(s-1)/2}^\circ(v)$ and thus belongs to $N_{(s-1)/2}(N_{(s-1)/2}^\circ(v))$. Any vertex z with $s-1 \leq d(v,z) \leq 3(s-1)/2$ has distance at most $(s-1)/2$ to at least one vertex in $N_{s-1}^\circ(v)$ and thus belongs to $N_{(s-1)/2}(N_{s-1}^\circ(v))$. Since $3(s-1)/2 = 1.5s - 1.5 \geq s$ for $s \geq 3$, we have considered all vertices in $N_s[v]$, and the claim follows. □



(a) Rule 1 bounds the $s/2$-neighborhood of a vertex v to at most $k-1$ vertices.

(b) Rule 1 bounds the $s/2$-neighborhood of a vertex $w_1 \in N^\circ_{s/2}(v)$ to at most $k-1$ vertices.

Fig. 1 Turing kernelization via Rule 1 for s-Club with even s.

Using Lemma 6, we show the following.

Theorem 3 *For odd $s \geq 3$, s-Club admits a k^3-vertex Turing kernel, which can be computed in $O(nm)$ time.*

Proof The kernelization and its running time analysis are completely analogous to Theorem 2. Hence, we only bound the size of $N_s[v]$ to show the kernel size by upper-bounding the number of vertices in each of the three sets $N_{(s-1)/2}[v]$, $N_{(s-1)/2}(N^\circ_{(s-1)/2}(v))$ and $N_{(s-1)/2}(N^\circ_{s-1}(v))$.

After application of Rule 1, the closed $(s-1)/2$-neighborhood of v consists of at most k vertices. Hence, $|N^\circ_{(s-1)/2}(v)| \leq k-1$. For each vertex $w \in N^\circ_{(s-1)/2}(v)$ the size of $N_{(s-1)/2}(w)$ is at most $k-1$. Thus, the size of $N_{(s-1)/2}(N^\circ_{(s-1)/2}(v))$ is at most k^2-2k+1. Therefore, also the size of $N^\circ_{s-1}(v)$ is at most k^2-2k+1. For each vertex $x \in N^\circ_{s-1}(v)$ we have $|N_{(s-1)/2}(x)| \leq k-1$. Thus the size of $N_{(s-1)/2}(N^\circ_{s-1}(v))$ is at most k^3-3k^2+3k-1. By Claim 6, the kernel thus contains at most $k+k^2-2k+1+k^3-3k^2+3k-1 \leq k^3$ vertices. □

3 Two Branching Algorithms

We present two branching algorithms for s-Club. First, we present an algorithm for the parameter solution size k. The main idea of this algorithm is to test for each $v \in V$ whether there is an s-club that contains v. This is done by considering $\{v\}$ as "seed" of an s-club and then branching into all possibilities to extend it. This branching is performed recursively until the considered set has size k. For a size-k set S, we then check whether $G[S]$ is an s-club. The

Algorithm: *Branch-s-club* (G,S,k)
Input: A graph G, a set S, an integer k
Output: An order-k s-club S' such that $S' \supseteq S$ if such an s-club exists.

```
1    if |S| < k then
2        for each u ∈ S
3            for each w ∈ {N_1(u)\S}
4                call Branch-s-club(G,S ∪ {w},k)
5    else
6        if G[S] is an s-club then output S
```

Fig. 2 Pseudo-code of the branching algorithm for parameter solution size k.

pseudo-code of the branching algorithm *Branch-s-club* is presented in Figure 2. Next, we show that a call to *Branch-s-club*(S, G, k) returns an order-k s-club that comprises S if such an s-club exists.

Lemma 7 *The algorithm Branch-s-club given in Figure 2 is correct.*

Proof First, the size of the set S is checked. If $|S| < k$, then at least one vertex needs to be added in order to obtain an order-k s-club. Furthermore, since we can assume that the s-club is connected, we must add a vertex that is a neighbor of some vertex $u \in S$. Hence, we branch first into all possible choices for u (Line 2), then we branch into all possible choices for adding a neighbor $w \notin S$ of u (Line 3). For each choice of w, we recursively call the algorithm with $S \cup \{w\}$ (Line 4).

If $S = k$, then we test in Line 6 whether S is an s-club. If this is the case, then we output S. Clearly, all connected size-k vertex sets that contain S will be checked whether they form an s-club. □

In general, the algorithm *Branch-s-club* does not yield fixed-parameter tractability for parameter k since in Line 3 it branches into an unbounded number of cases. The number of choices is bounded, however, when the algorithm runs on a graph to which Rule 1 has been applied.

Theorem 4 s-Club, $s \geq 2$, *can be solved in* $O((k-2)^k \cdot k! \cdot kn + nm)$ *time.*

Proof The algorithm works as follows: First, apply Rule 1 to the input graph G. If Rule 1 does not return a trivial s-club, call *Branch-s-club* for each $v \in V$. The correctness of this approach follows from Lemma 7. It remains to bound the running time of this algorithm.

The exhaustive application of Rule 1 takes $O(nm)$ time. Then, *Branch-s-club* is called for each $v \in V$. For each v, the number of recursive calls can be upper-bounded as follows. The algorithm branches into all possibilities to add a neighbor of a vertex in S. In the first branching step, that is, when $S = \{v\}$, there are $k - 1$ possible choices after application of Rule 1. When $|S| > 1$, we consider the neighbors of each vertex in S. Since each vertex of S has at most $k-1$ neighbors and at least one neighbor in S, the number of vertices to

consider is at most $k-2$. Hence, the number of recursive calls is $|S| \cdot (k-2)$. Branching is performed as long as $|S| \leq k-1$. The overall number of recursive calls thus is

$$(k-1) \cdot 2 \cdot (k-2) \cdot \ldots \cdot (k-1) \cdot (k-2) < (k-1)! \cdot (k-2)^{k-1}.$$

Testing whether S is an s-club takes $O(k^3)$ time using breadth-first search from each vertex in S. Overall, we need $O(nm)$ time for Rule 1, then we make n calls to a function which calls itself up to $(k-2)^{k-1} \cdot (k-1)!$ times, and additionally tests the at most $(k-2)^{k-1} \cdot (k-1)!$ resulting sets in $O(k^3)$ time. The overall running time bound follows. □

Finally, we describe a simple fixed-parameter algorithm for the dual parameter $d := n-k$. The idea is to branch on vertices that have distance at least $s+1$: at least one of them is not part of the solution.

Theorem 5 *s-Club can be solved in $O(2^d \cdot nm)$ time, where $d := n-k$.*

Proof The search tree strategy proceeds as follows: Search for a pair of vertices u and v with $d(u,v) \geq s+1$. If no such pair exists, then the graph is already an s-club and no further vertices need to be deleted. The search for these pairs takes $O(nm)$ running time using breadth-first search starting from each of the n vertices in the graph. Since u and v have too high distance from each other, at least one of them has to be deleted. Hence, branch into the two subcases of deleting either u or v and set $d := d-1$. Branching stops if $d = 0$ or an s-club has been found. Hence, the size of the search tree is at most 2^d. □

4 Conclusion

We conclude with several open questions. First, concerning the parameter solution size k, is there a linear Turing kernel and is there an algorithm with running time $c^k \cdot \text{poly}(n)$ for constant c? Second, concerning the dual parameter d, is there a polynomial kernel for this parameter and can the exponential part of the running time be improved to $o(2^d)$? Finally, many questions arise concerning the complexity of s-Club in special graph classes. For example, the problem can be formulated in monadic second-order logic [20] leading to several tractability results. For instance, s-Club is fixed-parameter tractable parameterized by the treewidth of the input graph and fixed-parameter tractable parameterized by s in case the input graph is planar [20]. What is the running time of direct combinatorial algorithms for these special cases?

References

1. R. D. Alba. A graph-theoretic definition of a sociometric clique. *Journal of Mathematical Sociology*, 3:3–113, 1973.

2. Y. Asahiro, E. Miyano, and K. Samizo. Approximating maximum diameter-bounded subgraphs. In *Proceedings of the 9th Latin American Theoretical Informatics Symposium (LATIN'10)*, volume 6034 of *Lecture Notes in Computer Science*, pages 615–626. Springer, 2010.
3. B. Balasundaram, S. Butenko, and S. Trukhanov. Novel approaches for analyzing biological networks. *Journal of Combinatorial Optimization*, 10(1):23–39, 2005.
4. H. L. Bodlaender. Kernelization: New upper and lower bound techniques. In *Proceedings of the 4th International Workshop on Parameterized and Exact Computation (IWPEC'09)*, volume 5917 of *Lecture Notes in Computer Science*, pages 17–37. Springer, 2009.
5. H. L. Bodlaender, R. G. Downey, M. R. Fellows, and D. Hermelin. On problems without polynomial kernels. *Journal of Computer and System Sciences*, 75(8):423–434, 2009.
6. J. Bourjolly, G. Laporte, and G. Pesant. Heuristics for finding k-clubs in an undirected graph. *Computers and Operations Research*, 27(6):559–569, 2000.
7. J. Bourjolly, G. Laporte, and G. Pesant. An exact algorithm for the maximum k-club problem in an undirected graph. *European Journal of Operational Research*, 138(1):21–28, 2002.
8. S. Butenko and O. Prokopyev. On k-club and k-clique numbers in graphs. Technical report, Texas A and M University, 2007.
9. R. G. Downey and M. R. Fellows. Fixed-parameter tractability and completeness II: On completeness for W[1]. *Theoretical Computer Science*, 141(1&2):109–131, 1995.
10. R. G. Downey and M. R. Fellows. *Parameterized Complexity*. Springer, 1999.
11. H. Fernau, F. V. Fomin, D. Lokshtanov, D. Raible, S. Saurabh, and Y. Villanger. Kernel(s) for problems with no kernel: On out-trees with many leaves. In *Proceedings of the 26th International Symposium on Theoretical Aspects of Computer Science (STACS'09)*, volume 3 of *LIPIcs*, pages 421–432. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, Germany, 2009.
12. J. Flum and M. Grohe. *Parameterized Complexity Theory*. Springer, 2006.
13. L. Fortnow and R. Santhanam. Infeasibility of instance compression and succinct pcps for np. *Journal of Computer and System Sciences*, 77(1):91–106, 2011.
14. J. Guo and R. Niedermeier. Invitation to data reduction and problem kernelization. *ACM SIGACT News*, 38(1):31–45, 2007.
15. D. Lokshtanov. *New Methods in Parameterized Algorithms and Complexity*. PhD thesis, Universitetet i Bergen, Bergen, Norway, 2009.
16. J. Marincek and B. Mohar. On approximating the maximum diameter ratio of graphs. *Discrete Mathematics*, 244(1-3):323–330, 2002.
17. N. Memon, K. C. Kristoffersen, D. L. Hicks, and H. L. Larsen. Detecting critical regions in covert networks: A case study of 9/11 terrorists network. In *Proceedings of the 2nd International Conference on Availability, Reli-*

ability and Security (ARES'07), pages 861–870. IEEE Computer Society, 2007.
18. R. Niedermeier. *Invitation to Fixed-Parameter Algorithms*. Number 31 in Oxford Lecture Series in Mathematics and Its Applications. Oxford University Press, 2006.
19. S. Pasupuleti. Detection of protein complexes in protein interaction networks using n-clubs. In *Proceedings of the 6th European Conference on Evolutionary Computation, Machine Learning and Data Mining in Bioinformatics (EvoBIO'08)*, volume 4973 of *LNCS*, pages 153–164. Springer, 2008.
20. A. Schäfer. Exact algorithms for s-club finding and related problems, 2009. Diploma Thesis, Friedrich-Schiller-Universität Jena.
21. S. B. Seidman and B. L. Foster. A graph-theoretic generalization of the clique concept. *Journal of Mathematical Sociology*, 6:139–154, 1978.