TITLE: Parameterized Complexity for Practical Computing

1 Challenges, Background, and The Research Project

Many of the computational tasks vitally necessary for computer applications in all areas of science, industry, and society are *NP-hard* [16], which means that they are unlikely to be solvable optimally on even the fastest computers in any reasonable time. These intrinsically intractable computational problems must still be addressed somehow. Overwhelmingly, they are tackled with what are called *heuristics*, a term that denotes intuitive/ad-hoc rules, such as the often used *Greedy Heuristic*, which makes choices based on what looks best at the moment. Heuristics often do a good job, but have no guarantees of efficiency or quality of solutions. A central contemporary challenge of theoretical computer science is to explain the surprising effectiveness of heuristics on real world input data for NP-hard problems. This Project takes up that challenge in the framework of Parameterized Complexity (PC) with three novel lines of attack that dramatically go beyond existing knowledge and develop methods new to the field.

The **primary objective** *of this Project is to develop new parameterized algorithms and complexity theory to understand the effectiveness of practical heuristics on real-world datasets, and to systematically design and improve heuristics, based on theory and experiments.*

The Algorithms Group in the Informatics Department at the University of Bergen is arguably currently the strongest research group in the world in parameterized complexity *theory*. This assertion can be easily checked just by counting the number of papers in the area at top international theory conferences such as FOCS, STOC, SODA, ICALP, STACS and ESA. Theoretical research leading to applications is of increasing interest in the parameterized complexity community. The *Parameterized Algorithms and Computational Experiments Challenge* created in 2016 had 27 teams competing from 11 countries, with the initial organizational impetus coming from Prof. Frances Rosamond, a key member of my Bergen team for this Project. The Project is large-scale with ambitious objectives. I am an excellent theory builder with high standards and have every confidence that the results will show the next big directions for both theory and applications. Based on my track record for leading and building research community, this Project will be well-coordinated and successful, and has already attracted considerable interest from potential collaborators who know how I have led the field. With this Project and my team, I intend to lead the University of Bergen Algorithms Group towards becoming the foremost research formation worldwide in *both* Theoretical *and* Applied Parameterized Algorithms and Complexity.

1.1 Background: Parameterized Complexity

I am credited as being the primary founder of the field of parameterized complexity beginning with the FOCS 1989 paper [1], for which I was the lead author, followed by a series of papers with my main partner, Rod Downey. Together with Downey, I wrote the first book on PC [9]. We recently published a second book [10]. Together these have been cited about 5000 times. Other textbooks and monographs iclude [21, 18, 6] with another soon to be published. Parameterized complexity theory has been based on two key insights, both associated with great waves of research. I fully expect that a third key insight, central to this Project, will also generate a wave of research productivity. I will now discuss these insights.

Insight 1: Secondary measurements beyond overall input size can crucially affect computational complexity. Such measurements may be many things, such as: *input structure, size of the solution sought, degree of approximation,* or *the number of electrons* — or an aggregate of several such secondary measurements.
Example: The ML TYPE CHECKING **problem** . Consider the problem of checking the consistency of type declarations for programs written in the high-level logic-based programming language ML. Paying attention only to the size n of a legal ML program, ML TYPE CHECKING is highly intractable, known to be complete for EXP (exponential time). *The ML compiler shouldn't work!* But it does.

The explanation is that human-composed programs typically have a maximum type-declaration nesting depth of $k \leq 6$. The compiler employs a type-checking subroutine that runs in time $O(2^k n)$, where n is the size of the program and k is the maximum nesting depth of the type declarations; This is clearly practical.

One can reflect that naturally occurring programs have small nesting depth because the programs would otherwise tax the thinking processes and abilities of the programmer creating them. In this way, natural input distributions have *inherited* parameters (relevant structural regularities and restrictions) due to the natural complexity constraints on the processes that *generate* typical inputs. The Project will explore this theme in a formal way in Objective 3, parameterizing on how typical inputs are generated, termed *reverse kernelization*, will be explored, innovatively using visualization techniques as a way of discovering appropriate generative parameterizations.

The ML TYPE-CHECKING problem concretely illustrates the central notion of *fixed-parameter tractability* (FPT) which refers to the possibility of solving a parameterized problem in time $f(k)n^c$ where:
• f is an arbitrary function of the *parameter* k (which can be aggregate of several secondary measurements),
• n, is the total input size in bits,
• c is a constant independent of k and n.
FPT is thus a natural generalization of the classical notion of *polynomial time*, solvability in time $O(n^c)$. FPT is able to accommodate finer complexity distinctions based on relevant secondary measurements captured by the parameter k. In practical terms, the availability of an FPT algorithm can make a huge difference. For example, for the modest numbers $k = 10$ and $n = 100$, an FPT algorithm with running time $2^k n$ runs about a thousand trillion times faster than an algorithm with running time $O(n^k)$. Many natural parameterized problems have brute force algorithms running in time $O(n^{O(k)})$ (e.g., by trying all k-subsets of the vertex set of a graph). Not all parameterized problems admit an FPT algorithm. Hardness for the parameterized complexity class $W[1]$ is the natural analog of NP-hardness.

Insight 2: FPT can be viewed (equivalently!) in terms of polynomial-time kernelization. If we call kernelization the **second great wave** of research in parameterized complexity theory, then again I am credited for the seminal ideas and papers. The very idea of focusing on polynomial sized kernels as a target for PC research was pioneered by me and recognized (jointly with three others) by a Nerode Prize honor for fundamental work in this area. I invented the term *kernelization* in the early days of parameterized complexity theory. This insight spurred the development of an almost entirely new field, especially after interest began to focus on the objective of *polynomial kernels* (where the function g in the following lemma is a polynomial function of k.)
Lemma. ([11, 5]) *A parameterized decision problem Π is FPT if and only if it is* polynomial-time kernelizable*, that is, if and only if there is a polynomial-time transformation taking an instance (x, k) to an instance (x', k') where:*
(1) (x, k) is a yes-instance if and only if (x', k') is a yes-instance,
(2) $k' \leq k$,
(3) $|(x', k')| \leq g(k)$ for some function g.
Kernelization lower bound theory and the clash between upper and lower bounds has made kernelization a subject of intense interest. Because polynomial-time kernelization is essentially a straightforward model of **preprocessing**, universally important in practical computing, the kernelization view of FPT is firmly established as relevant to practical computing. There has been transfer of theory into practical computing applications with the caveat that pieces of theoretical results are often cherry picked by practitioners.

Insight 3: FPT is really about polynomial time extremal structure theory and this structure theory can be used in a variety of ways. The celebrated Graph Minors project that provided a major impetus to the development of PC theory already substantiates this view. Every concrete FPT result can be viewed as launching an associated *structure theory project*. My research Project focuses on the situation where an FPT problem admits such an associated structure theory that is algorithmically well-behaved, captured by the notion of *smooth FPT*, that leads to an entirely new level of PC theory of very wide reach.

A concrete example of smooth FPT is afforded by the MAX LEAF SPANNING TREE problem. The optimization form of this problem is: *given a graph G, find a spanning tree of G with a maximum number of leaves*. Parameterized by the number of leaves, this can be proved to be FPT in a variety of ways (including graph minor theory). Consider now the following sequence of reflections:

• FPT is necessarily about P-time kernelization, so we are necessarily concerned with discovering a kernelization algorithm with the smallest achievable kernel size-bounding function g.

• To prove the kernelization bound naturally and systematically involves: (1) finding polynomial-time applicable reduction rules that to be useful for solution construction need to be polynomial-time invertible to take

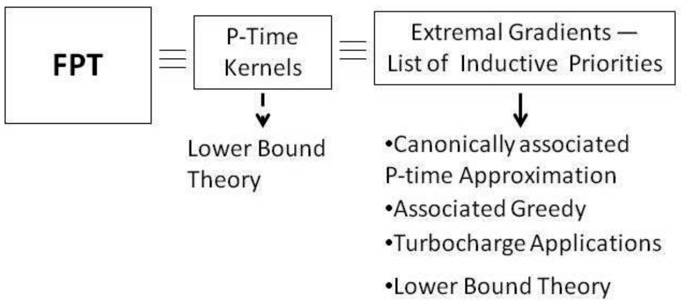

FPT
P-Time Kernels
Extremal Gradients — List of Inductive Priorities
Lower Bound Theory
•Canonically associated P-time Approximation
•Associated Greedy
•Turbocharge Applications
•Lower Bound Theory

Figure 1: FPT extremal induction leads to mathematical theory.

a solution for the kernel back to a solution for the original instance, and (2) in natural mathematical argumentation, a proof by induction (by minimum counterexample), which begins by inducting on k, the number of leaves, and a structural study of the boundary where the answer is "yes" for k, and "no" for $k + 1$. Being extremal combinatorics (modulo the polynomial-time requirements on the reduction rules) this will naturally involve a series of inductive priorities regarding the minimum counterexample. Implicitly (the boundary), the zeroeth priority is the number of leaves k.

It is a principal, and **potentially revolutionary thesis** of this Project, that *smooth FPT* inductive gradients provide a new view of combinatorial optimization with wide applicability, partly because *the structural insights are indifferent to k being small.* A key objective is to explore this both theoretically and experimentally. A key notion in this regard is *parametric duality.*

There are many mathematical techniques and approaches for proving that a parameterized problem is FPT, including some methods (such as well-quasi-ordering) that are nonconstructive: you end up knowing that an FPT algorithm exists, without knowing what it is. Early in my career, I wrote the first pioneering, now heavily cited, papers exploring this fascinating issue and offering some general ways of making such results constructive. This work involved substantial aspects of theory building, as did my later pioneering research at the root of the first and second great waves of PC research. This Project is centered at the far other end of the spectrum of FPT methods, in a highly constructive form of FPT algorithm design, and disciplined argumentation for the result. Basically, this form of FPT algorithm design, situated in the kernelization view of FPT, requires (essentially) that every aspect of the algorithm and its argumentation, is executable and interpretable in terms of polynomial time subroutines. We do know that smooth FPT is a proper subset of anything-goes FPT, and that there are several natural and important examples of smooth FPT.

The virtue of smooth FPT is that its results can (we hypothesize) be exported and used in a variety of ways. In this Project, we focus primarily on how smooth FPT results might be used to design and empower heuristic algorithms for NP-hard optimization problems based on the structure of the inductive argumentation for the kernel bound. We also pursue a secondary agenda of developing the *fundamental theory of smooth FPT*. We fully expect the outcomes of this Project to seed a **third great wave** of research in parameterized complexity theory — and this time, *a great new wave of research* in its practical applications, which must necessarily, realistically, and in an intellectually exciting way, be (primarily) through the systematic, mathematically empowered design of heuristics. Success in this project would be a revolutionary advance in both theoretical and applied parameterized algorithms and complexity, with substantial impact on combinatorial optimization in general.

1.2 The Research Project

Three novel and interrelated research objectives will be investigated theoretically and empirically in my project.
• Objective 1 is to investigate the third key PC insight, discussed above, that FPT is really about polynomial time extremal structure theory. We aim for a theory of *smooth FPT* that will enable systematic transfer of PC results to the design of heuristics. We also expect approximation and exponential algorithms based on FPT extremal gradients and smooth kernelization that asymptotically out-perform current best-known theoretical results.
• Objective 2 is to develop new paradigms for using FPT to improve current heuristics as well as design new ones, and to explore theoretically how these various strategies interrelate. We will investigate turbocharging heuristics experimentally in a variety of application areas, and also FPT turbocharging kernelization, local search, and polynomial time approximation algorithms obtained from FPT extremal gradients.

• Objective 3 aims PC in another thoroughly new direction we call *reverse kernelization* that uses parameterization to model typical data input generation. These are all bold new directions and the new theory will be tested empirically.

1.2.1 Objective 1: INDUCTIVE GRADIENTS AND SMOOTH KERNELIZATION

My project will explore, both theoretically and empirically/experimentally, using *smooth FPT* extremal structure gradients to systematically FPT-turbocharge various important and familiar heuristic algorithmic strategies, and to develop novel heuristic algorithmic strategies built around the possibilities of FPT subroutines as key components. Preliminary results primarily due to my research on the MAX LEAF SPANNING TREE problem, will be an important initial wedge for both the theoretical and the experimental directions of my project, which we will drive deep to reveal other key combinatorial optimization problems Π for which either Π or its parametric dual is FPT, as the theory and experiments develop, with an expected feedback loop.

My project will introduce the concept of using *polynomial time extremal structure* and extremal arguments to refine the gradient for various heuristics, and thereby adjust the *quality* of solution. The MAX LEAF SPANNING TREE problem will be used as an example to demonstrate the approach. More specifically, the inductive gradient is an artifact of the inductive argument by minimum counterexample and uses *inductive priorities*.

Smooth Kernelization: Smooth kernelization refers to being able to prove an FPT result in such a structured manner, where: (1) the reduction rules are P-time applicable, (2) P-time invertible and (3) solution-quality preserving; and, (4) the argumentation for the kernelization bound has a standard structure of proof by minimum counterexample, with a witness structure (WS) and inductive priorities (IP), (5) where the structural claims registered against the WS and IP can be translated into P-time subroutines to improve the witness structure. Many FPT parameterized problems admit smooth kernelization, although provably (modulo accepted complexity hypotheses) some do not.

1.2.2 Objective 2: TURBOCHARGING HEURISTICS

My project will develop general methods for *turbocharging* heuristics that utilizes the nondeterminism found in Greedy and other heuristics that allows for a limited amount of back-up when the heurisic encounters a *moment of regret*. A moment of regret in Graph Coloring, for example, would be when one is forced to use a new color. I invented the term "turbocharging heuristics" and designed a turbocharged greedy heuristic for MINIMUM DOMINATING SET [8]. The project will consider standard heuristics for foundational optimization problems (k-coloring, dominating set, etc). We will identify key moments in the execution where the heuristic either fails or needs to increase the cost of the solution. We will design parameterized algorithms for improving solutions and plug these as subroutines into the existing heuristics, thereby turbocharging the heuristic. The strategy can be applied wherever limited back-up is allowed. We will explore turbocharged (and even double turbocharged) Greedy and other heuristics, as well as turbocharged *kernelization* and turbocharged *approximation*. The algorithms and turbocharged heuristics developed in this project will be tested on standard benchmarks and their performance compared systematically with packages such as CPLEX, ILP, SAT-solvers. Practitioners do not tend to read theory. This project leads to the possibility of packaging the theoretically designed turbocharged algorithms into easy-to-use modules for practitioners to insert as subroutines into heuristics they already use.

1.2.3 Objective 3. REVERSE KERNELIZATION

This objective begins an investigation of an innovative generative parameterization of natural datasets based on narrative about where the dataset may have arose. The "Silk Road" narrative, for example, may begin with a line (imagine a trail used by pilgrams), then interstices (resting points for the camels), and then offshoots (made by firewood gatherers). The Silk Road starts from a "seed" graph of size k (part of the parameterization), where we alternately subdivide edges and then add an independent set, building up a structure. This project will provide a detailed complexity map of the hierarchy of special graph classes resulting from the reverse kernelization methodology. Note that one is no longer limited to randomly generating graphs and there is a lot of control. It is clear that many graphs, such as Weihe's model of all the trains in Europe [23], must have a lot of structure. But, we don't know what that structure is. We know, for example, that Weihe's train graph problem (identify a small subset of stations that can service all the trains) can be reconfigured as the Feature

Set problem (identify small sets of cancer-causing genes), and the same reduction rules apply [19, 20]. Similar reduction rules are being used by Pablo Moscato researching breast cancer [4], and by Kitty Meeks researching spread of virus [12].

Parameterizing "typical inputs" as part of rich aggregate parameterization in a generative way is a thoroughly new research direction with potentially high gain. The Visualization Group in the Informatics Department at University of Bergen is keen to work with us on developing visualization tools that shed light on modeling where data comes from.

2 The Project Plan and Methodology

In this section, we describe in more detail the three research objectives in which the Project is structured and detailed work packages for each objective. We also propose to carry out a thorough empirical evaluation of our algorithms. Experimentation, the last task in an algorithm engineering cycle, usually raises new questions and provides novel insights that inform the design of new algorithms, leading to another cycle of design, analysis, implementation, and experimentation [22].

2.1 Objective 1: Inductive Gradients and Smooth Kernelization

The objective of systematizing the use of FPT extremal gradients in heuristics is of high originality and inserting them into existing heuristics packages has the potential for high payoff.

Objectives include: • Systematic means of developing inductive gradients for FPT problems, • of translating inductive gradients into greedy and other algorithms, • of translating inductive gradients into approximation algorithms, • of using inductive gradients in local search, and • meta-theorems linking these deployments of inductive gradients. Objectives include: • Systematic use of inductive gradients to FPT turbocharge derived approximation algorithms, • of inductive gradients in FPT turbocharging of greedy and other heuristics. • Lower bound theory for some of these uses of inductive gradients.

Most gradients for local search heuristics only reflect the basic cost of a solution. The focus of this research direction is to use extremal inductive argumentation in FPT kernelization to find tighter measurements of solution quality. The inductive gradient is an artifact of the inductive argument by minimum counterexample establishing a kernelization bound. The inductive gradients (priorities) essentially measure the quality of a solution – how close is the tree to *being ready to add a new leaf*, or making sensible something like: "The graph has a spanning tree with 14.789 leaves, which is better than one with only 14.123 leaves," as a metaphor for three further inductive priorities – even though, considering the trees involved as discrete structures, they both have 14 leaves. This tighter measurement of solution quality is a powerful product of extremal inductive argumentation in FPT kernelization. FPT is the *signal* that mathematical structure is available to be exploited algorithmically. Graph Minor Theory is driven by the question: *What is the structure of a graph G that excludes a graph H as a minor?* MINOR TESTING IS FPT and every FPT parameterized problem launches an algorithmically constrained extremal structure theory project.

It is a natural conjecture that if there are two different extremal inductive gradients G_1 and G_2 arising from two different "P-time extremal" kernelization arguments, where the second is associated with a better kernel bound, then the (*greedy heuristic, approximation algorithm, turbocharge efficiency factor*) canonically derived from G_2 might be better than the (*greedy heuristic, approximation algorithm, turbocharge efficiency factor*) canonically derived from G_1. This is intuitive but unexplored, and it could turn out to be wrong in practice, because the deeper inductive gradient may be slower, and not worth the extra effort. Only computational experimentation can tell — the issue is not open to worst-case asymptotic theoretical investigation. The correct question is rather: "How *much* depth of inductive gradient is worthwhile?" In some sense, we cannot lose, because using only the zeroeth inductive priority is just the usual local search heuristic (e.g., the only considers the number of leaves).

2.1.1 Approach and Methodology for Inductive Gradients and Smooth Kernelization

Next we give an example to illustrate the general idea.

Case study – MAXLEAF: Recall the MAXLEAF problem. Given an undirected graph, the MAX LEAF SPANNING TREE problem is to find a spanning tree with as many leaves as possible.

MAXLEAF	
Input:	A graph $G = (V, E)$ and an integer k.
Question:	Is there a spanning tree of G with k leaves?

In [13], we developed a polynomial-time extremal structure theory for graphs of bounded max leaf number and proved a polynomial-time kernelization bound based on a collection of P-time reduction rules

such that if the reduced instance has more than $3.75k$ vertices then it is always a 'Yes' instance. Based on this result, the standard FPT scheme is as follows:

1. Kernelize the input by applying the data-reduction rules of [14] to obtain a reduced instance of the problem.
2. If the reduced instance has more than $3.75k$ vertices, then output 'Yes'.
3. If the reduced instance has at most $3.75k$ vertices, then solve the the reduced instance using exhaustive search.

This is a valid FPT algorithm, but a more useful way of establishing a kernelization lemma is to study the *boundary* between 'Yes' and 'No' instances. In general, an instance of a parameterized problem consists of a pair (G, k), so the boundary is located by holding G fixed while varying k. Tackling the problem from a "boundary" point of view we were able to prove the following lemma:

Lemma 1 (Boundary Lemma [14]). *If (G, k) is a reduced instance of* MAXLEAF *with (G, k) being a 'Yes' instance and $(G, k + 1)$ being a 'No' instance, then $|V(G)| \leq 3.75k$.*

There are two points to note in relation to this result. Firstly, the proof of the lemma works with a witness structure that is not a spanning tree of G with k leaves, but rather a subtree that has k leaves. And then, the improved kernelization outcomes of the lemma is essentially due to the fact that the argument employs a more refined "better solution" gradient. More specifically, the proof is by minimum counterexample and uses *inductive priorities*. The priorities not only include the number of leaves but also five other priorities that contribute to the quality of a solution.

Now turning our attention to an r-opt local search heuristic, we can show how the Boundary Lemma inspires a more effective approach. The typical form that a local search heuristic takes is the following: Maintain a current solution consisting of a spanning tree T of G, where the value of T is the number of leaves of T.

The local search now repeatedly performs the following routine until no better solution is found. Delete r edges from T, reconnect the resulting forest in all other possible ways by adding r edges into a spanning tree T'. If the value of T' is greater than the value of T then update the current solution to T'. This process is then repeated for a different set of r edges.

Next we argue how the Boundary Lemma could be used for a more refined local search in two ways. First, the local search can be conducted based on maintaining a "current witness structure" rather than a complete solution. Second, the proof of the Boundary Lemma uses *inductive priorities* to select the "best" solution. The same idea can be used to define a "better" solution gradient for the local search. That is, there are many solutions that have k leaves, but one might be better than the others for future iterations. The inductive priorities can be used to choose the "best" one based on the structural insights on the boundary between 'Yes" and 'No' instances.

The generalization of the usual setup for local search inspired by the mathematical structure will be explored. It is expected that the most practical results come from using a subset of the inductive priorities to refine the local search gradient.

Experiments will explore using the extremal program in turbocharging algorithms in applied contexts, such as deploying a gradient to improve a heuristic. A local search heuristic for MAX LEAF may maintain a *current solution* spanning tree T, and then repeatedly delete 3 edges in all possible ways (perhaps 4 is too expensive) and try all possible ways of rejoining the pieces to try to get a spanning tree with more leaves. Working with kernelized graphs, we will explore 3-mutations of the witness structure of the extremal argument to determine if the pieces can be reassembled into a "better" witness structure (for k leaves) according to the inductive gradient.

Our main aim is to explore, and mathematically systematize the use of sophisticated inductive gradients in the design of heuristics. The general idea of using structural properties to refine the gradient can also be used for defining "moments of regret" for greedy heuristics or improving kernelization bounds in Direction 2.

2.1.2 Workplan and Workpackages for Inductive Gradients and Smooth Kernelization:

There are three workpackages for Inductive Gradients and Smooth Kernelization that are devoted to theory (GT1, GT2, and GT3), and three parallel, accompanying workpackages devoted to empirical experiments (GE1, GE2, and GE3). The workpackages will inform each other and contribute to further developments in both directions. The research in each workpackage will be conducted by a team consisting of one research postdoc and one PhD student. All teams will be supervised and with joint research by the project investigator.

Workpackages GT1 and GE1 will perform a thorough case study of MAXLEAF from a theoretical and experimental point of view. The objectives are to strengthen the structural properties, propose a list of inductive priorities inspired by the structural properties, and perform an extensive experimental evaluation using the inductive priorities. Workpackage GT1 aims to develop for each of three basic graph problems, a series of increasingly powerful (and necessarily, increasingly more elaborate) smooth kernelization results, with their associated inductive gradients. A model for this is the substantial, but incomplete results concerning the MAX LEAF SPANNING TREE problem [cites]. A high priority for this work package is to complete that project. The parametric dual of MAX LEAF is Connected Dominating Set which is not FPT, so using local search for a "better" (fewer number of vertices) has no gradient. It is generally the case that the parametric dual of an FPT problem is W-hard but, we can use a tree in the complement FPT Max Leaf to guide the search. Gradients can be transferred from an FPT problem to its dual. For our prototype MAX LEAF, we have early exploration of how different strengths of the boundary lemma yield different kernelization bounds. The other two basic graph problems on which we focus are MINIMUM GRAPH COLORING and NONBLOCKER. MINIMUM GRAPH COLORING parameterized by the number of colors is not FPT, but the parametric dual problem, SAVING k COLORS is FPT, so the gradient(s) will be developed working with this problem. NONBLOCKER is the parametric dual of the basic graph problem MINIMUM DOMINATING SET, and is known to be FPT. A secondary objective of this workpackage is to explore the systematic translation of smooth kernelization results into polynomial time approximation algorithms.

Experimental workpackages GE1 in tandem with GT1 will conduct basic computational experiments that explore such issues as whether gradient local search heuristics have performance superior to ordinary local search heuristics, beginning with simple FPT inductive gradients for the MAX LEAF problem, and advancing to more sophisticated inductive gradients based on more powerful smooth kernelization results, as developed in workpackage GT1. We will also conduct computational experiments with greedy algorithms developed from smooth kernelization inductive gradients, and with Concorde style heuristics for the ML problem based on gradients. Secondary objectives include experimenting with the transfer of gradients to parametric dual problems such as the transfer from ML to MINIMUM CONNECTED DOMINATING SET.

Workpackages GT2 and GE2 will broaden the identification of inductive gradients based on smooth kernelization to FPT problems of a different character: *graph layout problems* such as MIN CUT LINEAR ARRANGEMENT (which is FPT, but does not have a polynomial kernel) and FPT problems about strings of symbols. A secondary objective of GT2 is to explore how inductive gradients can be used in worst-case exponential exact algorithms. Workpackage GE2 focuses on experiments with gradients for the other basic problems that will be developed in the theory workpackages. This will be supplemented by experiments with translating gradients to novel greedy algorithms based on smooth inductive gradients, and will involve experimental exploration of the interactions and trade-offs between gradients and kernelization. A secondary focus will be on exploring experimentally the transfer of gradients to the parametrically dual problems of these targets.

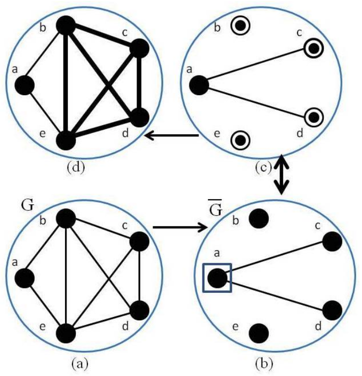


Figure 2: Duality is used to find Clique, which is W-hard. To find the clique in (a), find the minimum Vertex Cover (FPT) in its complement (b). Vertex Cover and Independent Set (W-hard) are duals (c). Use the IS to obtain the clique (d).

Workpackages GT3 and GE3 aim to develop the general theory of smooth kernelization inductive gradients, and their uses in designing and enhancing heuristics. Secondary aims are to develop a general theory of smooth FPT (which we know to be different from general FPT [cite Lokshtanov]), including the development of lower bound methods for the issue of whether an FPT problem has a smooth polynomial kernelization.

2.2 Objective 2: Turbocharging Heuristics

I will explore and develop new paradigms for using FPT results (both FPT subroutines and inductive gradients) to strengthen established heuristic methodologies such as *greedy heuristics*, *pre-processing* and *local search*. This research direction involves both theoretical work and computational experimentation, with an emphasis on the latter, and a feedback loop. Some concrete examples will help to illustrate the myriad opportunities.

Example 1: FPT Turbocharging a Greedy Heuristic for GRAPH COLORING**.** A common heuristic strategy for the MINIMUM GRAPH COLORING problem (*color the vertices of a graph with the minimum possible number of colors, such that no two adjacent vertices receive the same color*) is a variation on the following two stage process (for a single run):
(1) Order the vertices from high degree to low degree.
(2) Progressively go through the list. At a vertex v on the list, if a color is available (perhaps there are several), choose one to be the color of v. Else, one must introduce a new color to handle v.

We could say that it is a *moment of regret*, if at v, we are forced to introduce a new color, bringing the total number of colors used (up to v) to $t + 1$. A moment of regret, because our goal is to minimize the total number of colors used. Variations on this strategy involve various policies on how to choose an available color. Notice that at the level described, both (1) and (2) are “essentially nondeterministic”. *FPT turbocharging* seeks to back up k steps from the moment of regret, and to explore whether the last k steps of (2) *could have been executed differently* (within the bounds of the nondeterminism inherent in (2)) in order to arrive at v without having to use a fresh color. This is termed *strict turbocharging*. An alternative, *permissive turbocharging* allows that the k vertices preceding v on the list (which is held fixed), but only those k, can be recolored in any way so that when we reach v it is not necessary to use another color, in other words, to reach v having used at most t colors. Suitably formalized, both forms of turbocharging are FPT in the *operational parameter k* (which we are free to choose) are FPT. In a pioneering paper, Hartung and Niedermeier reported successful experiments with such a turbocharged greedy algorithm, as compared with the best available heuristics on standard benchmarks [**?**]. Recently, turbocharging greedy heuristics for finding tree decompositions has been explored [17] with successful results in comparison with standard heuristics, competing on standard benchmarks.

Example 2: FPT Turbocharging a Greedy Heuristic for MINIMUM DOMINATING SET**.**

The NP-hard MINIMUM DOMINATING SET (MDS) problem admits a greedy heuristic somewhat similar to Graph Coloring: (1) order vertices from low to high degree, (2) if a vertex is undominated by the set D of vertices so-far chosen, choose a vertex in $N[v]$ of greatest *utility* (number of so-far undominated neighbors). Down the list, vertices have high degree and likely a neighbor in D. Both (1) and (2) are *essentially nondeterministic* polynomial-time procedures; in (1) there could be many vertices of the same degree from which to choose; in (2) there could be many vertices of the same utility. The quality of the dominating set computed in (2) where the ordering is fixed, could depend on the ordering established in (1). The technical problem we explore is the following. As (2) is executed, we may have a *moment of regret*: reaching v and having to augment D. We investigate redoing the last k steps of nondeterminism, to reach the same point in the vertex ordering *with no regret*. Formalized, this results in an FPT turbocharged greedy heuristic with the parameter *k* being the amount of “back-up” [8], performing well on standard benchmarks [2]. As above, there may be more than one nondeterministic step in a single heuristic, which may lead to “double-turbocharging”. Since this is an *operational parameterization*, we will experiment with *k* to find the right trade-off between the FPT *f(k)* that we can afford, and the improvement in the performance of the procedure. Nondeterminism can be found in ordering, scoring, and other procedures. We will inventory and publicize these opportunities.

Example 3: FPT Turbocharging Preprocessing and Kernelization. Preprocessing and kernelization typically involve a variety of local reduction rules that are exhaustively employed until the kernelized instance is reached. Again, this is an “essentially nondeterministic” procedure, as at any given step, and for any of the reduction rules, there may be multiple opportunities to apply the rule to the current reduced instance. There is a clear *moment of regret*: when there are no further opportunities to apply the reduction rules to shrink the input! Thus arises the possibility of *turbocharging kernelization*.

Example 4: Gradient Local Search for a Dual Problem. Suppose we will to explore FPT gradient

turbocharging of a local search heuristic for the important (because of local area network applications and others) problem of MINIMUM CONNECTED DOMINATING SET (CDS). At first glance, this solution-sized parameterized problem: *Does G have a connected dominating set of size at most* k seems impossible to gradient turbocharge, because it is not FPT! However, a graph has a connected dominating set of size k if and only if it has a spanning tree with at least $n - k$ leaves. This suggests the following approach.
Step 1: Kernelize the given graph G according to the reduction rules for the smooth FPT kernelization for the MAX LEAF problem, obtaining the reduced graph G'.
Step 2: At each step of the local search heuristic for CDS, move to "the best possible" *current solution* as the CDS solution space is explored, interpreting this to mean *the quality of the associated solution to the dual* MAX LEAF *problem*, evaluated according to the structural inductive gradient for MAX LEAF.
Step 3: After a solution S' for CDS for G' is decided upon at the conclusion of the local search, lift this to a solution S for the original graph G by undoing Step 1, using the fact that the (smooth) kernelization rules are P-time invertible, and the nonobvious fact that solution quality for CDS can be lifted compatibly with ML kernelization, due to the well-behaved parametric duality. Theoretical work is required to show that this all fits together properly.

2.2.1 Approach and Methodology for Turbocharging Heuristics

At the theoretical level, we plan to explore to explore several themes:
• Commonalities between the FPT results required to FPT-turbocharge local search heuristics, and the FPT results required to FPT-turbocharge greedy algorithms (either strictly or permissively). Both involve exploring a limited region with respect to a solution that is mostly fixed, so there should be general ways of connecting these two contexts and perhaps proving metatheorems about when the relevant parameterized problems are FPT based on some form of general logical characterization.
• An abstract point of view about what a greedy heuristic *is*. Is there a useful notion of a greedy heuristic *canonically associated* with a kernelization inductive gradient? Can we prove general results about when such associated greedy heuristics can be gradient-turbocharged? • Another related parameterization paradigm for which a few results are currently known (I have been a major pioneer in these) is the *dynamic problem* paradigm. Here too there is the possibility of systematic connections to FPT-turbocharging of local search heuristics, and there should be logic-based metatheorems about when dynamic problems are FPT, for various kinds of mathematical objects (not only graphs).

At the experimental level, the following themes are important:
• We need to design experimental protocols to test various concepts and hypotheses, such as the hypothesis that nontrivial gradient turbocharging of local search heuristics is sound. Complications come from the fact that each local search step for a gradient-turbocharged local search heuristic may be slower, so we will need to design experimental protocols to test whether a gradient-turbocharged heuristic is actually "smarter" at all in practice, and to what extent (a proof-of-concept demonstration)?
• In some greedy algorithms (such as the one for MINIMUM GRAPH COLORING there are several essentially nondeterministic phases. We need to experimentally explore whether (or to what extent) turbocharging both is useful in practical terms.
• The interaction of turbocharging greedy or local search heuristics with the implicit inductive structure in reverse kernelization needs to be explored both theoretically and experimentally.

2.2.2 Workplan and Workpackages for Turbocharging Heuristics:

The workpackages will build turbocharging heuristics case studies showing theoretical and empirical design and analysis of new turbocharged algorithms. The workpackages will investigate permissive variants and lower bounds. Included will be the investigation of confluence and approximate confluence, that is, how the ordering of reduction rules affect efficiency. The research will investigate how the fpt extremal inductive gradients of Direction 1 can be used to define more "fine-grained" regret. Extremal gradients can be canonically translated into approximation algorithms, so the possibility of *turbocharging polynomial time approximation* will be investigated. Targets include approximate confluence, *k*-local search, iterative compression, general frameworks for upper and lower bounds, and transfer results, possibly connected to logic. The experimental workpackages will also investigate how large a search space (measured as the distance to the given solution) can we allow depending on the size of F, the part of the solution that we fix, and how much more effective do the heuristics become by enlarging the search space.

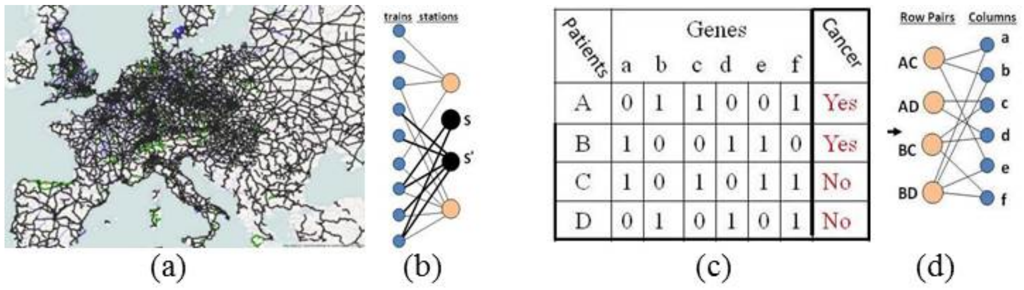
Workpackages [TT1] and [TE1] will inventory nondeterminism found in ordering, scoring, kernelization and other procedures. Workpackage [TT1] will explore such questions as: when is nontrivial turbo-kernelization possible or impossible? Some impossibility results are known, through the theory of confluent graph rewriting systems. We will also explore whether there may be provable limits of a quantitative nature concerning what turbo kernelization may achieve. We will explore, for the various smooth inductive gradients for ML, under what circumstances the turbocharging problem is FPT or W-hard, for both permissive and strict turbocharging of greedy algorithms for ML and it's parametric dual CDS. A secondary objective of this work package will be to prove a nontrivial FPT^{NP} result for turbocharging a P^{NP} greedy heuristic. The experimental workpackage [TE1] has the principal aim of beginning experimental exploration of turbocharging. This involves not only the turbocharging of greedy heuristics, but also of possibilities for turbocharging other kinds of essentially nondeterministic subroutines, including kernelization. With the starting point of two nontrivial smooth inductive gradients for the max Leaf problem [cite big Max leaf paper], we will experiment with both FPT and XP turbocharging of ML kernelization. We will also explore experimentally gradient turbocharging of the connected dominating set problem by transferring gradients from the parametric dual problem ML. A key goal will be to show that some degree of turbocharging is useful in FPT kernelization.

Workpackage [TT2] will explore, for the smooth inductive gradients developed in the inductive gradient workpackages for our basic targeted graph problems, when the turbocharging problem, for both strict and permissive gradient turbocharging, is FPT. We will also attempt to establish the FPT results needed to turbocharge popular greedy heuristics for graph layout and stringology problems, and perhaps other not-about-graphs problems. We will also target to determine, for list-based greedy heuristics, such as the usual one for graph coloring, under what circumstances can the list building phase be FPT turbocharged, both strictly and permissively. Workpackage [TE2] will experiment with gradient turbocharging for the basic problems for which smooth gradients have been developed, that is, for Nonblocker and it's dual problem Dominating Set, and for the Saving k Colors problem and it's dual problem Minimum Graph Coloring. We will also investigate how useful double turbocharging is in practice. A secondary objective of this work package is to experiment with FPT^{NP} turbocharging of such problems as finding a small number of vertices that hit all maximal cliques, which does have a natural P^{NP} greedy heuristic. This is an important objective, because success will substantially expand the program suggested by Szeider and deHaan [7], which has already been validated for some problems in the area of computational logic and artificial intelligence, but is so far unexplored in practical terms for combinatorial optimization problems.

The principal focus of workpackage [TT3] is to develop general theory concerning the interrelationship of the various kinds of turbocharging (for example, turbocharging greedy heuristics and turbocharging kernelization), and local search heuristics, gradient turbocharging of local search heuristics, gradient recombination in memetic or Concorde style heuristics, turbo-iterative compression and dynamic problems. A specific objective of this package is to prove a logic-based meta-theorem concerning FPT dynamic problems, and FPT permissive local search. A secondary specific objective is to prove a logic-based meta-theorem relating turbo kernelization and dynamic problem FPT. The experimental workpackage [TE3] will broaden the investigation of the practicality of turbocharging for problems that are not about graphs, such as problems concerning strings of symbols, and for graph problems that have a different character, such as layout problems. In this workpackage, we may be addressing problems that we have not yet chosen at this time, based on our experiences in the earlier work packages. The outcomes of the earlier FPT^{NP} turbocharging experiments may broaden the exploration to further combinatorial optimization problems in the Szeider-deHaan paradigm.

2.3 Objective 3: Reverse Kernelization

This novel, high-gain objective concerns real datasets in practical computing situations. They offer many relevant secondary measurements affecting computational complexity, however they have mostly been studied in an *ad hoc approach*, failing to capture the abundant structure available. I started the parameter ecology program [19], where one parameterizes on any kind of structural measurement, and in that context studies the complexity of optimizing some other measurement (Fig. 6). Seminal results of Faisal Abu-Khzam [3] show that a considerable number of problem features can be included in a *rich, aggregate parameterization*, and that contrary to expectation, adding parameters does not make the problem more difficult, and it is still



Patients	Genes a	b	c	d	e	f	Cancer
A	0	1	1	0	0	1	Yes
B	1	0	0	1	1	0	Yes
C	1	0	1	0	1	1	No
D	0	1	0	1	0	1	No

Figure 3: (a) Trains of Europe. (b) Bipartite graph of Trains and Stations. (c) Feature Selection sketch chart of microarray data. Columns are genes. Rows represent patients. Goal is to identify a small set of genes that distinguishes which patients have cancer or not. (d) Feature Selection as a bipartite graph. Patients A and C differ in genes a, b and e. The Feature Selection graph follows reduction rules similar to that of the Train Graph.

mathematically elegant. The breakthrough here is to conceptualize in a complexity framework, how typical inputs arise, and how they can be incorporated into rich aggregate parameterizations of input structure.

A specific target is to explain Karsten Weihe's train problem, and its connection to Feature Selection (Fig 7). Feature Selection is one of the most fundamental problems in the fields of pattern recognition, machine learning and data mining where the aim is to eliminate noisy, irrelevant or misleading features while retaining a suitably high accuracy in representing the original features. In biology, feature selection numbers may be enormous. If the column space is microarray gene expression data, then the column space is around 35,000. The aim of efficient feature selection is to determine a small number of genes to pay attention to in distinguishing *cancer* versus *non-cancer*. Other application areas which can benefit from generative models are spread of disease in networks of animals or humans and generative models of voters in Election Theory.

Case study - Karsten Weihe's Train Problem and the Feature Selection Problem An optimization problem concerning the train systems of Europe was described by Karsten Weihe [23]. The graphs are large, on the order of 10,000 vertices. The problem can be represented by a bipartite graph with vertices partitioned into two sets S (stations) and T (trains). An edge represents that a train t stops at a station s. The problem is to find a minimum number of stations to service all the trains. This is a special case of HITTING SET, and therefore NP-complete.

The following two reduction rules can be applied to simplify (pre-process) the input to the problem. Let $N(s)$ denote the set of trains that stop at station s, and let $N(t)$ denote the set of stations at which the train t stops.

1. If $N(s) \subseteq N(s')$ then delete s.
2. If $N(t) \subseteq N(t')$ then delete t'.

Applications of these reduction rules cascade, preserving at each step enough information to obtain an optimal solution. Weihe found that, remarkably, these two simple reduction rules were strong enough to "digest" the original, huge input graph into a *problem kernel* consisting of disjoint components of size at most 50, small enough to allow the problem to then be solved optimally by brute force. Perhaps, more surprisingly, we found that the FEATURE SET problem; for example, identifying a few genes to distinguish which patients have cancer, has the same structure as the train graph. This project will investigate rich, aggregate parameters as routinely mathematically doable when part of the parameterization is concerned with the generative description of inputs, applying well-quasi-ordering as an FPT classifier, and also standard kernelization arguments. Novel uses of color-coding enabled by generative parameterization will be studied.

A key step, partly because the problem is so well-known, is a generative parameterization explanation of why the reduction rules work so well for Weihe's Train Graph and why they also work on FEATURE SELECTION with respect to cancer and other data. If the parameter is the *size k of the largest connected component after the rules have been exhaustively applied* then we clearly have an FPT algorithm. For each connected component we pay exponentially in k to compute a minimum-sized dominating set. Is this cheating? Or are there computing situations where FPT is identified by parameters that are *emergent*? We anticipate in the case of train and other graphs, there is an available narrative about how they typically arise.

2.3.1 Approach and Methodology for Reverse Kernelization

Our methodology investigates *generative* and *emergent* parameterizations with a narrative we call *reverse kernelization*. The process generates "typical" problem instances by starting with a seed and performing various operations. A graph of bounded VERTEX COVER number can be thought of as starting with a *seed graph* of size k, and arbitrarily connecting an independent set to the seed. Let $ISI(k)$ denote all of the graphs that can be obtained from a seed graph of size k by first adding an independent set, then subdividing, and then adding another independent set. A minimum dominating set can be computed in FPT time for graphs in $I(k)$ (bounded vertex cover number) and for graphs in $S(k)$ (bounded max leaf number) — but what about $ISI(k)$? This class of graphs contains both $SI(k)$ and $IS(k)$, which are incomparable, but both of which simultaneously generalize $I(k)$ and $S(k)$. *A Silk Road* generative narrative might start with a simple graph of a few trails, with vertices being towns along the route. This *seed* of size k, part of the parameterization, grew more complicated. Merchants opened hostels along the route — the seed graph was subdivided. These needed firewood, so additional vertices were added, representing camps of wood-gatherers. The subdivided seed is the same as Max Leaf. The W[1]-hard Minimum Dominating Set becomes FPT when the parameter is the Max Leaf number of the input graph.

We investigate how to find and use narratives about how the typical inputs arise. In contexts such as the research of Kitty Meeks on epidemiological transmission on *animal contact graphs* [12], detailed information for generative parameterization is readily available.

2.3.2 Workplan and Workpackages for Reverse Kernelization:

Workpackage [RT1] initiates exploration of the research direction that we term *reverse kernelization*. We will develop what we term the *JAT Hierarchy* of special graph classes, described by reverse kernelization classes generated by the two operators: *I: add an independent set* and *S: subdivide the edges*. We will investigate theoretically for which of these classes is the un-parameterized optimization problem FPT or W-hard, for the basic problems on which we focus (ML, SC, Nonblocker, MinCut). We will also explore the containment relations that structure this hierarchy. We will build upon previous unpublished work of the chief investigator investigating the parameterized complexity of these optimization problems, when the parameter is: *the vertex cover number* (which is essentially the reverse kernelization class $revker(I)$, and *the max leaf number* (which is essentially the reverse kernelization class $revker(S)$). A possibly related, but important secondary objective is to try to theoretically solve the conundrum of the famous Train Problem. Workpackage [RE1] will explore visualization of kernelization for the Train Problem, in collaboration with the Visualization Group at the University of Bergen. The hypothesis is that insights gained by visualization can be translated into reverse kernelization parameterizations. A second objective is to investigate visualization of historical data about the development of train networks, as a second way of using visualization to develop reverse kernelization parameterizations of typical instances.

The focus of workpackage [RT2] will be on proving logic-based FPT meta-theorems for problems where the inputs are restricted to the classes of the JAT hierarchy. We will also investigate the issue of designing heuristics for the unrestricted Max Leaf (ML) problem for these classes. When ML is FPT for one of these classes, how should the general gradient for ML interact with the FPT gradient for such results? We will begin with the ML problem, and then seek to extend to the other basic graph problems that are targeted in this project. Workpackage [RE2] will focus on computational experiments in gradient local search and gradient kernelization for the reverse kernelization class of graphs in the JAT Hierarchy. A secondary objective is to use visualization to explore kernelization for planar graph problems on real-world instances and to use this to define appropriate reverse kernelization classes.

The objectives of [RT3] are to explore how visualization and reverse kernelization can be used to explain theoretically other mysteriously efficient preprocessing heuristics on real-world data sets, beyond the train problem. As a secondary objective, we will investigate the possibility of meta-theorems about other hierarchies of reverse kernelization classes beyond the JAT hierarchy. In conjunction with theory, workpackage [RE3] will conduct the interesting experiments of workpackages [GT1] and [GE2] four inputs restricted to the reverse kernelization classes of the JAT hierarchy. A secondary objective is to extend these computational experiments to other reverse kernelization classes beyond the JAT hierarchy.

The Visualization Group in the Informatics Department at University of Bergen is keen to work with us on developing visualization tools that shed light on modeling where data comes from.

	Year 1	Year 2	Year 3	Year 4	Year 5
Gradient Theory	GT1	GT2	GT3		Finish book for CUP
Gradient Experiment	GE1	GE2		GE3	
Turbo Theory	TT1	TT2		TT3	
Turbo Experiment	TE1	TE2		TE3	
RevKer Theory	RT1	RT2		RT3	
RevKer Experiment	RE1	RE2		RE3	

Figure 4: Workplan and workpackages.

Neural nets are beyond the scope of this Project, but future research might investigate whether neural nets can recognize samples of graphs from different reverse kernelization classes.

Integration of Theory and Experiment

One of the major challenges of the Project is that it will require a close integration of theory development and computational experimentation. The computational experiments are project objectives in their own right, and will be of several different kinds, for example:

– Experiments with random or synthetic data in order to gather evidence that certain heuristic design strategies actually work (“proof of concept”). For example, it will be very interesting to see to what extent local search heuristics that utilize FPT inductive gradients might be superior to simple local search heuristics. In order to make the comparison fair, it will be necessary to adjust for the fact that a single run of a gradient local search may be slower than a single run of a simple local search heuristic. Local search heuristics are commonly based on taking the best result from many different runs, so one way to make the comparison fair would be to give the simple local search heuristic more runs. Such issues will require careful design of the computational experiments.

– Experiments with real data or standard benchmarks. In this experimental context it would be meaningful to compare an FPT-turbocharged version of a standard heuristic, against the performance of the non-turbocharged version, again perhaps adjusting for the fact that FPT-turbocharging entails some time complexity costs for a single run beyond those of the non-turbocharged version. Another issue for experiments in this context will be that the amount of turbocharging is a key parameter k; when $k = 0$ one simply has the non-turbocharged heuristic, so the question really becomes *how much* FPT-turbocharging is worth the extra cost, since this will generally be an exponential function of k.

The algorithms and turbocharged heuristics developed in this project will be tested on standard benchmarks and their performance compared systematically with packages such as CPLEX, ILP, SAT-solvers. We note that several of my team members have been involved in the FPT computational challenge (PACE) that will be in its third year when the project commences, and that has required careful consideration in its design. We expect that experiment will inform theory and vice-versa.

The timeline for the workpackages is presented in Figure 1, with the second half of year 5 dedicated to finishing the book for Cambridge University Press.

Feasibility and Timeliness My project is thoroughly modern and pathbreaking for a number of reasons related to the contemporary ecology of practical computing. The memory hierarchy structure of modern machines has evolved in ways that make FPT algorithms more practical in terms of hidden constants than they used to be. Due to the open source revolution, sophisticated code is much easier to compose. • Powerful, highly engineered *solver packages* for a variety of NP-complete problems, such as SAT or ILP, are now widely and conveniently available and frequently used in heuristic computing strategies. • The practical computing ecology in many important applied areas, such as Bioinformatics, routinely relies on visualization of data as part of the cycle. • The parameterized algorithms and complexity community has warmed to the idea of delivering more practical impact. (The advent of the PACE Challenge is one testimony to this.) • My explorations on how visualization can be used to identify appropriate generative parameterizations that capture how typical inputs arise will inevitably lead to challenges of FPT algorithm design where the aggregate parameter includes information on how the input arises, a challenge that we can now have some confidence of being able to meet. For another example, in the exploration of FPT-turbocharging greedy heuristics, I will spend some energy exploring the possibility of FPT^{NP}-turbocharging P^{NP} greedy heuristics for some combinatorial optimization problems that are complete for higher levels of the Polynomial Hierarchy (beyond NP). Here, a P^{NP} *greedy heuristic*, refers to a polynomial time greedy heuristic algorithm that makes calls to a SAT-solver. I see only success.

3 The Host Institution, The Research Team, International Cooperation

Host Institution. The University of Bergen's Algorithms Research Group, in the Department of Informatics (UiB), Norway is the perfect location for the project. The Algorithms Research Group provides a highly stimulating research environment with 8 faculty members, large international contact with frequent guests and intensive publication activity. It is one of the leading algorithms centres in Europe with excellent researchers all with expertise relevant to the project. The group currently manages 3 ERC grants, led by close collaborators *Fedor Fomin, Saket Saurabh*, and *Daniel Lokshtanov*. *Pinar Heggernes* with two large Research Council of Norway projects is an expert in Graph Algorithms; *Fredrik Manne* is expert in Parallel Processing and algorithm engineering; *Jan Arne Telle* is expert in width parameters; and *Frances Rosamond* is expert in parameterized complexity kernelization. The University is home to other excellent researchers whose knowledge is relevant to the project, including *Inge Jonassen*, Bioinformatics Group and *Helwig Hauser* from the Visualization Group.

The department's infrastructure (network, file and mail servers, printers, library) can be used for the project, and the administrative staff of the department (secretaries, system administrator, etc.) will be available for support. UiB has a Cray supercomputer (at purchase, it was one of the top 50 in the world) which we will use for our experimental evaluation when needed.

The Research Team. The Project will be led by me, Prof. Michael Fellows. I am a full professor at the Institute for Informatics, hired under the Elite Toppforsk Program in 2016. I will take overall responsibility for the Project, fully engage in research in every part of the Project and lead the team. I will also work on foundational aspects of the research, and will initiate new and intensify existing international collaborations. I will also act as a mentor together with chief collaborator Rosamond. In the past, I have successfully lead research teams as the Chief Investigator of research projects. Prof. Rosamond joins the project as an experienced scientist in parameterized algorithms and complexity as well as Editor of the *Parameterized Complexity Newsletter*, manager of the PC wiki (www.fpt.wikidot.com), and publicist of IPEC. She will contribute to research, dissemination, mentoring and assist with Project management.

This is a very large-scale project with extremely ambitious aims. Because of the diversity of topics that require experience and knowledge in a range of sophisticated mathematical and computing areas we request funding for three PostDoc researchers and three PhD students for three years each. The PostDocs and the PhD students will work on theoretical questions as well as on empirical evaluations. It is important that the PostDocs have strong expertise on parameterized algorithms, kernelization, approximation algorithms and interest in bridging algorithmic theory to applied problems. Some of the programming that will be used require state-of-art knowledge early on, thus the need for researchers at the PostDoc level. The researchers will be recruited internationally to secure the need for expert competence. The PhD students will focus on specific tasks well-suited to their research direction and interests.

I am well-known as one of the founders of parameterized complexity and I am the foremost leading expert on the areas of this project, and the high reputation of UiB will ease the search for outstanding PostDocs with the appropriate expertise. We will work actively to ensure gender balance in the composition of the team. We have potential candidates in an Erasmus Scholar and 2 research visitors coming in September, all are female, who are interested in the research topics of the project.

International Collaboration. One of the purposes of Toppforsk funding is to promote Norwegian research groups internationally. Discussions about the research directions proposed in this Project have already generated considerable international and local interest.

As soon as colleagues hear about the project they want to join because they know I am an excellent theory builder with high standards and therefore this will be the next big direction for theory. Based on my track record for building community, they have confidence that this project will be well coordinated and successful. Sixteen international experts in parameterized complexity from 8 countries, each with their own research teams have volunteered to contribute expertise, following my lead. I have worked with all of them in the past. The gain here is not only applicative, but also contributes to the scientific community in terms of follow-on research, beyond the scope of this project. Collaborators are listed in the Application Form.

I have a track record of delivering ground-breaking results relevant to every one of the research objectives. For Objective 1, it is my insight to use extremal arguments for kernelization and I did the work on MAX LEAF that will be the key exemplar of this project. Objective 2, I created the notion of Turbocharging Heuristics and have designed the first dynamic algorithms for this. For Objective 3, I developed the Parameter Ecology Table and proved many of the results in it. Given that I have teamed up with the world-leading experts on all of the topics that are most relevant to the project, under my leadership the objectives will be more than met.

References

[1] Karl R. Abrahamson, John A. Ellis, Michael R. Fellows, and Manuel E. Mata. On the complexity of fixed parameter problems (extended abstract). In *30th Annual Symposium on Foundations of Computer Science, Research Triangle Park, North Carolina, USA, 30 October - 1 November 1989*, pages 210–215. IEEE Computer Society, 1989.

[2] F. Abu-Khzam, J. Egan, M. Fellows, F. Rosamond, and P. Shaw. On the parameterized complexity of dynamic problems with connectivity constraints. In *Combinatorial Optimization and Applications – 8th Intern'l Conf (COCOA)*, LNCS. Springer.

[3] Faisal N. Abu-Khzam. The multi-parameterized cluster editing problem. In Peter Widmayer, Yinfeng Xu, and Binhai Zhu, editors, *Combinatorial Optimization and Applications - 7th International Conference, COCOA 2013, Chengdu, China, December 12-14, 2013, Proceedings*, volume 8287 of *Lecture Notes in Computer Science*, pages 284–294. Springer, 2013.

[4] Regina Berretta and Pablo Moscato. Cancer biomarker discovery: the entropic hallmark. *PLoS One*, 5(8):e12262, 2010.

[5] Liming Cai, Jianer Chen, Rodney G. Downey, and Michael R. Fellows. Advice classes of parameterized tractability. *Ann. Pure Appl. Logic*, 84(1):119–138, 1997.

[6] M. Cygan, F. Foman, . Kowalik, D. Lokshtanov, D. Marx, M. Pilipczuk, M. Pilipczuk, and S. Saurabh. *Parameterized Algorithms*. Springer, 2015.

[7] Ronald de Haan and Stefan Szeider. Parameterized complexity classes beyond para-np. *Journal of Computer and System Sciences*, 87:16 – 57, 2017.

[8] R. Downey, J. Egan, M. Fellows, F. Rosamond, and P. Shaw. Dynamic dominating set and turbo-charging greedy heuristics. *Tsinghua Journal of Science and Technology*, 19(4):329–337, 2014.

[9] R. Downey and M. Fellows. *Parameterized Complexity*. Springer, 1999.

[10] Rodney G. Downey and Michael R. Fellows. *Fundamentals of Parameterized Complexity*. Texts in Computer Science. Springer, 2014.

[11] Rodney G. Downey, Michael R. Fellows, and Ulrike Stege. Parameterized complexity: A framework for systematically confronting computational intractability. In Ronald L. Graham, Jan Kratochvíl, Jaroslav Nesetril, and Fred S. Roberts, editors, *Contemporary Trends in Discrete Mathematics: From DIMACS and DIMATIA to the Future, Proceedings of a DIMACS Workshop, Stirín Castle, Czech Republic, May 19-25, 1997*, volume 49 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, pages 49–100. DIMACS/AMS, 1997.

[12] Jessica Enright and Kitty Meeks. Deleting edges to restrict the size of an epidemic: A new application for treewidth. *Algorithmica*, pages 1–33, 2017.

[13] V. Estivill-Castro, M. Fellows, M. Langston, and F. Rosamond. Fpt is p-time extremal structure i. In *Proc 1st Algorithms and Complexity in Durham Workshop*, volume 4, pages 1–41, 2005.

[14] Vladimir Estivill-Castro, Michael R. Fellows, Michael A. Langston, and Frances A. Rosamond. FPT is p-Time extremal structure i. In *Proceedings of the 1st Algorithms and Complexity in Durham Workshop*, volume 4, pages 1–41, 2005.

[15] M. Fellows, D. Lokshtanov, N. Misra, M. Mnich, F. Rosamond, and S. Saurabh. The complexity ecology of parameters: An illustration using bounded max leaf number. *Theory Comput. Syst.*, 45(4):822–848, 2009.

[16] Michael R. Garey and David S. Johnson. *Computers and Intractability, A Guide to the Theory of NP-Completeness*. W.H. Freeman and Company, 1979.

[17] Serge Gaspers, Joachim Gudmundsson, Mitchell Jones, Julián Mestre, and Stefan Rümmele. Turbocharging treewidth heuristics. In Jiong Guo and Danny Hermelin, editors, *11th International Symposium on Parameterized and Exact Computation, IPEC 2016, August 24-26, 2016, Aarhus, Denmark*, volume 63 of *LIPIcs*, pages 13:1–13:13. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2016.

[18] M. Grohe and J. Flum. *Parameterized Complexity Theory*. Springer, 2009.

[19] Bart M. P. Jansen Michael Fellows and Frances Rosamond. Towards fully multivariate algorithmics: Parameter ecology and the deconstruction of computational complexity. *European J. Combinatorics*, 34(3):541–566, 2013.

[20] Serge Gaspers Michael R. Fellows and Frances A. Rosamond. Parameterizing by the number of numbers. *Theory of Computing Systems*, 50(4):675–693, 2012.

[21] Rolf Niedermeier. *Invitation to Fixed-Parameter Algorithms*. Oxford University Press, 2006.

[22] Peter Sanders. Algorithm engineering – an attempt at a definition using sorting as an example. In *Proceedings of the 12th Algorithm Engineering and Experimentation Workshop*, pages 55–61.

[23] K. Weihe. Covering trains by stations or the power of data reduction. In *Proc 1st Workshop on Algorithms and Experiments (ALEX 1998)*, pages 62–75. 1998.