Parametrisierte Ansätze für schwere Graphprobleme: Algorithmen und Experimente

Algorithms and Experiments for Parameterized Approaches to Hard Graph Problems*

Falk Hüffner
School of Computer Science, Tel Aviv University, Israel

25. Oktober 2010

Zusammenfassung

Viele kombinatorische Probleme aus der Praxis sind *NP-schwer*; zur ihrer Lösung werden meist Heuristiken verwendet. *Parametrisierte Komplexität* ist ein neuerer Ansatz, der versucht, Strukturen von Praxisinstanzen auszunutzen. Ziel der Arbeit war es zu belegen, dass parametrisierten Komplexität, und insbesondere neuartige algorithmische Techniken, deren Entwicklung auf dieses Konzept zurückgeht, tatsächlich zu einsetzbaren Programmen für die exakte Lösung von Praxisinstanzen führt. Wir zeigen dies hier am Beispiel der Graphprobleme CLIQUE COVER und MINIMUM-WEIGHT PATH, die Anwendungen in der Bioinformatik und anderen Gebieten haben.

Many real-world problems are NP-hard; to solve them, usually heuristics are used. *Parameterized Complexity* is a recent approach that tries to exploit structures of real-world problem instances. The aim of the work was to establish that Parameterized Complexity, and in particular novel algorithmic techniques whose development was driven by this concept, can actually lead to practically useful programs for exactly solving real-world problem instances. We show this here using the example of CLIQUE COVER und MINIMUM-WEIGHT PATH, which have applications in computational biology and other areas.

F.2.2 [Analysis of Algorithms and Problem Complexity]: Nonnumerical Algorithms and Problems; G.2.2 [Discrete Mathematics]: Graph Theory—Graph algorithms
General Terms: Algorithms; Experimentation; Theory
Additional Key Words and Phrases: Parameterized complexity; Data reduction; Color-coding

1 Parametrisierte Komplexität

Viele kombinatorische Probleme aus der Praxis sind *NP-schwer*. Es ist inzwischen eine gängige Annahme, dass NP-Schwere eine inhärente kombinatorische Explosion im Lösungsraum impliziert, die zu Laufzeiten führt, die exponentiell mit der Eingabegröße wachsen. Dies bedeutet, dass große Instanzen nicht immer optimal gelöst werden können.

In der Praxis werden für NP-schwere Probleme meist *Heuristiken* eingesetzt, die keine Laufzeitgarantien oder Qualitätsgarantien haben, sondern darauf getrimmt sind, für typische Instanzen schnell gute Ergebnisse zu liefern. In vielen Fällen ist es nicht akzeptabel, dass ein Algorithmus in Grenzfällen extrem lange läuft oder deutlich suboptimale Lösungen liefert. Parametrisierte Komplexität [2] ist ein neuerer Ansatz, der in vielen Fällen einen Ausweg bietet, bei dem man weder auf Optimalität noch auf nützliche Laufzeitschranken verzichten muss. Die zentrale Beobachtung ist, dass Praxisinstanzen fast immer ausnutzbare Strukturen aufweisen. Die Idee ist, die strukturelle Komplexität mit einem *Parameter* zu messen, der üblicherweise eine positive Zahl k ist. Wir können dann eine zweidimensionale Komplexitäts-

*Die Promotion erfolgte an der Fakultät für Mathematik und Informatik der Friedrich-Schiller-Universität Jena. Die Gutachter waren Prof. Rolf Niedermeier (Friedrich-Schiller-Universität Jena), Prof. Michael R. Fellows (The University of Newcastle, Australien) und Prof. Peter Rossmanith (Rheinisch-Westfälische Technische Hochschule Aachen).

analyse vornehmen, bei der das Wachstum der Laufzeit sowohl mit der Eingabegröße n als auch mit k beschrieben wird. Die Hoffnung ist, dass man die kombinatorische Explosion auf den Parameter einschränken kann, sodass Instanzen schnell lösbar sind, solange der Parameter klein ist. Die Problemklasse FPT (*fixed-parameter tractable*) beschreibt die Problem/Parameter-Kombinationen, für die dies möglich ist.

Ein Beispiel ist die Aufgabe, von n Messungen möglichst wenige wegzulassen, sodass keine zwei Messungen mehr widersprüchlich sind. Ein guter Parameter ist hier die Anzahl zu löschender Messungen k, da dieser typischerweise klein sein wird. Es gibt FPT-Algorithmen für dieses Problem, deren Laufzeit nur mit k exponentiell wächst, und die somit sogar noch Instanzen mit $k = 120$ und $n = 1000$ lösen können.

Es gab bisher kaum Veröffentlichungen, die sich systematisch mit der Erforschung der praktischen Umsetzung der neuen Ideen befassen. Ziel meiner Arbeit war es zu belegen, dass das Konzept der parametrisierten Komplexität, und insbesondere auch neuartige algorithmische Techniken, deren Entwicklung auf dieses Konzept zurückgeht, tatsächlich zu einsetzbaren Programmen für die exakte Lösung von Praxisinstanzen führt. Eine solche neuartige Technik (Farbkodierung) wird in Abschnitt 3 vorgestellt. Parametrisierte Komplexität erlaubt aber auch einen neuen Blickwinkel für bekannte Techniken, wie im folgenden Abschnitt gezeigt wird.

2 Datenreduktion

Datenreduktion ist eine klassische Methode, mit schweren Problemen umzugehen: bevor der eigentliche Lösungsprozess gestartet wird, versucht man, die Größe der Eingabe zu verringern, indem man Teile entfernt oder vereinfacht, zum Beispiel weil man sofort sehen kann, dass sie irrelevant für die Lösung sind. Aufwendige Lösungsalgorithmen brauchen dann nur auf den verbliebenen „harten Kern“ der Instanz angewendet zu werden.

Einige Datenreduktionsregeln sind einfach und werden leicht von jedem entdeckt, der ein Problem angeht. Andere erfordern tiefere Einsichten in die kombinatorische Struktur des Problems. Sobald eine effektive Reduktionsregel gefunden wird, sollte sie in praktisch jedem Kontext zum Einsatz kommen, sei er heuristisch, approximativ oder exakt.

Datenreduktion wurde bisher meist als heuristische Aufgabe gesehen, denn in der klassischen eindimensionalen Komplexitätstheorie kann man nichts über die Qualität der Datenreduktion aussagen. Parametrisierte Komplexität hingegen stellt zur Einschätzung der Wirkung einer Datenreduktionsregel das Konzept eines *Problemkerns* bereit [1]. Eine Reduktion auf einen Problemkern ist eine in Polynomzeit laufende Datenreduktion, die eine Instanz so verkleinert, dass ihre Größe nur noch vom Parameter k abhängt, und nicht mehr von der ursprünglichen Eingabegröße n. Auf diese Weise ermöglicht das Problemkernkonzept einen fruchtbaren Dialog zwischen Praktikern und Theoretikern: Problemkerne können erklären, warum Reduktionsregeln in der Praxis so gut funktionieren; und die Suche nach Problemkernen kann zu neuen, mächtigen Datenreduktionsregeln führen, die auf tiefen strukturellen Einsichten beruhen. Insbesondere sind Problemkerne auch nützlich, wenn der Parameter nicht klein ist, da sie in Polynomzeit laufen.

Datenreduktion für Clique Cover. Beim CLIQUE COVER-Problem ist die Aufgabe, gegeben einen Graphen, eine kleinstmögliche Menge von Cliquen (vollständigen Teilgraphen) zu finden, die alle Kanten abdeckt. Es hat zahlreiche Anwendungen in der Compileroptimierung, bei geometrischen Berechnungsproblemen und in der angewandten Statistik. CLIQUE COVER ist NP-schwer, und es gibt keine praktisch brauchbaren Approximationsgarantien; deshalb sind bisher eingesetzte Verfahren meist heuristisch. Wir geben eine Reihe von Reduktionsregeln für CLIQUE COVER an, die es ermöglichen, zusammen mit einem simplen Brute-Force-Verfahren viele Praxisinstanzen in wenigen Sekunden zu lösen. Eine einfache Regel besagt, isolierte Knoten (also Knoten ohne Nachbarn) zu löschen. Hier ist die Korrektheit noch offensichtlich. Als Beispiel für eine aufwändigere Reduktionsregel sei die folgende genannt (Abbildung 1).

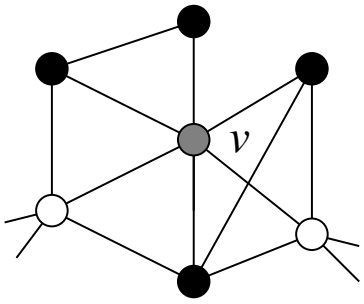

Abbildung 1: Partition der Nachbarn von v. Die zwei weißen Knoten sind Ausgänge, und die schwarzen Knoten sind Gefangene. Da jeder Ausgang einen Gefangenen als Nachbarn hat, ist Reduktionsregel 1 anwendbar, und v kann gelöscht werden.

Reduktionsregel 1. *Sei $N(v)$ die Menge der Nachbarknoten von v.*

Partitioniere die Nachbarschaft eines Knoten v in Gefangene p *mit* $N(p) \subseteq N(v)$ *und* Ausgänge x *mit* $N(x) \setminus N(v) \neq \emptyset$. *Wenn alle Ausgänge mindestens einen Gefangenen als Nachbarn haben, dann lösche v.*

Der Grund für die Korrektheit dieser weniger offensichtlichen Regel ist, dass v zu jeder Clique hinzugefügt werden kann, die einen Gefangenen enthält, und dass auf diese Weise alle an v angrenzenden Kanten abgedeckt werden können. Mit den beiden genannten Regeln ergibt sich das folgende Resultat, wobei k die Anzahl Cliquen.

Theorem 1. *Nach Anwendung der Reduktionsregeln hat die* CLIQUE COVER*-Instanz höchstens* 2^k *Knoten, d. h.* CLIQUE COVER *hat einen Problemkern.*

Die gefundenen Datenreduktionsregeln wurden zusammen mit einem simplen Brute-Force-Algorithmus auf verschiedenen Instanzen getestet. Von den Testinstanzen aus der Visualisierung von Getreideertragsdaten mit bis zu 124 Knoten und 4847 Kanten konnten alle in unter einer Sekunde gelöst werden. Mit einem Modell, dass es erlaubt, größere solcher Instanzen zu simulieren, konnten in unter 20 Sekunden noch Instanzen mit bis zu 300 Knoten gelöst werden. Dies deckt alle für diese Anwendung praxisrelevanten Größen ab.

3 Farbkodierung

Farbkodierung ist eine randomisierte Methode zum Finden kleiner Teilgraphen der Größe k in einem Graphen. Sie kann benutzt werden, um Kandidaten für Signalpfade in Proteininteraktionsnetzwerken zu finden, also eine Folge unterschiedlicher Proteine, bei der jedes mit dem vorigen stark wechselwirkt. Dies kann als das NP-schwere MINIMUM-WEIGHT PATH-Problem modelliert werden: finde in einem kantengewichteten Graphen einen einfachen Pfad einer gegebenen Länge mit minimalem Gewicht. Beispielsweise suchen wir im Proteininteraktionsnetzwerk der Hefe mit 4 400 Knoten und 14 300 Kanten nach Pfaden der Länge $k = 5$ bis 15.

Bei Farbkodierung wird jeder Knoten des Graphen zufällig mit einer von k Farben eingefärbt. Dabei hofft man, dass die k Knoten im optimalen Pfad alle unterschiedliche Farben erhalten (der Pfad *bunt ist*), denn unter dieser Annahme lässt sich mittels dynamischen Programmierens MINIMUM-WEIGHT PATH erheblich schneller lösen. Da dies aber meist nicht der Fall ist, wiederholt man diese Versuche, bis mit hoher Wahrscheinlichkeit mindestens einmal der Pfad tatsächlich bunt war. Das Verfahren liefert deswegen einen Festparameteralgorithmus, weil sowohl die Laufzeit der Suche nach einem bunten Pfad als auch die Anzahl Versuchswiederholungen, um eine vorgegebene Fehlerwahrscheinlichkeit ε zu erreichen, im exponentiellen Anteil nur von k abhängen.

Unsere Implementierung zeigt eine Beschleunigung von mehreren Größenordnungen gegenüber vorherigen Ergebnissen. Insbesondere kann sie für praktisch alle für die Suche nach Signalpfaden relevanten Parametereinstellungen Resultate in Sekunden liefern, was die interaktive Untersuchung solcher Pfade ermöglicht. Für diese Aufgabe haben wir die graphische Benutzeroberfläche FASPAD (Abbildung 2) entwickelt. Hier kann der Benutzer zu einem beliebigen Netzwerk und vorgegebener Pfadlänge die besten Pfadkandidaten finden und graphisch anzeigen lassen, wobei auch Überlappungen der Pfade und ihre Umgebung im Netzwerk angezeigt werden können.

4 Zusammenfassung

Neben den genannten Beispielen wurden in meiner Arbeit eine Reihe neuer Festparameteralgorithmen entwickelt und Implementierungen für vier Probleme angegeben, die nach unserem Stand des Wissens die jeweils schnellsten Löser sind. Dies unterstreicht die praktische Brauchbarkeit des parametrisierten Ansatzes, und insbesondere seine Nützlichkeit als Leitfaden für die Entwicklung neuartiger algorithmischer Techniken. Basierend auf diesen Erkenntnissen findet sich in der Dissertation eine Schritt-für-Schritt-Anleitung, wie man mit den Methoden der parametrisierten Komplexität effektiv zu optimalen Ergebnissen für NP-schwere Probleme kommt.

Die Implementierungen für CLIQUE COVER und MINIMUM-WEIGHT PATH sowie weitere Beispiele sind als freie Software unter der *GNU Public License* erhältlich bei http://theinf1.informatik.uni-jena.de/~hueffner/.

Literatur

[1] J. Guo and R. Niedermeier. Invitation to data reduction and problem kernelization. *ACM SIGACT News*, 38(1):31–45, 2007.

[2] R. Niedermeier. *Invitation to Fixed-Parameter Algorithms*. Number 31 in Oxford Lecture Series in Mathematics and Its Applications. Oxford University Press, 2006.

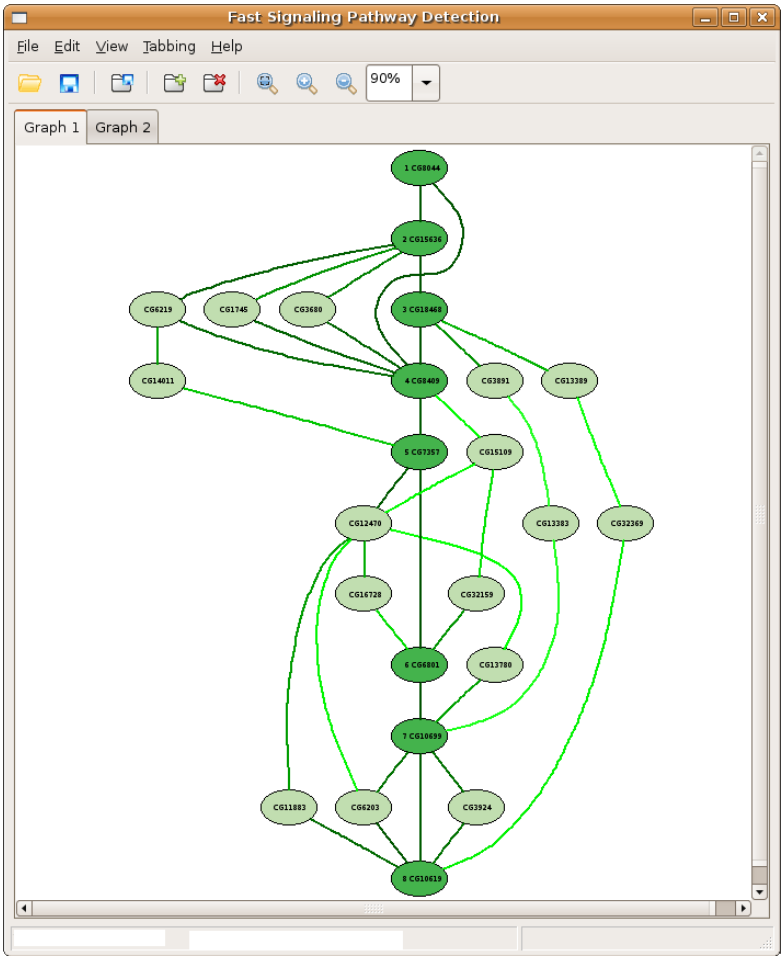

Abbildung 2: Pfad im Netzwerk der Fruchtfliege mit Kontext [**Abbildung über 2 Spalten**]

Dr. rer. nat. Falk Hüffner studierte Informatik an der Eberhard-Karls-Universität Tübingen. Nachdem er dort zunächst auch als wissenschaftlicher Angestellter bei der Gruppe von Dr. Rolf Niedermeier begann, wechselte er zusammen mit der Gruppe 2004 an die Friedrich-Schiller-Universität Jena, wo er 2007 seine Promotion abschloß. Seit Februar 2008 ist er als Postdoc an der Tel Aviv University im Bereich Computational Genomics in der Gruppe von Prof. Ron Shamir tätig.