Multivariant Algorithmics

Rod Downey
Victoria University
Wellington
April, 2016

The Goals

- "Understand computation"
- What does it mean to be a computable process?
- What does it mean to be an efficiently computable process?
- How to find a best algorithm for a process?
- How to know you cannot do better?
- How to cope with apparently hard problems?
- What has this got to do with computation on real data.

The Role of Complexity theory

- Provide mathematical models for computational tasks.
- Assign cost to computational processes.
- Explain why some processes seem hard.
- Provide algorithms for more efficient computation.
- Explain the actual behaviour of algorithms *in practice*.
- Is theory fulfilling this role?

Algorithms and Complexity

- Basic idea count the steps as a function of the size of the input. (Hartmanis, Sterns, Edmonds)
- Classical idealization Efficient= P = polynomial time= those problems which have algorithms running in $O(n^c)$ where n is the size of input.
- Controversial in the 1960's.
- How to prove something is not e.g. in P?
- Kind of a holy grail in complexity theory.

NP

- Russians (Levin) perabor, Cook identified nondeterministic polynomial time.
- These are defined as problems L such that there is a polynomial time checking process $R(x, y)$ such that

$$x \in L \text{ iff } \exists y(|y| \leq |x|^c \wedge R(x, y)).$$

- Examples: SATISFIABILITY, VERTEX COVER (vertices cover edges), DOMINATING SET (vertices cover vertices), HAMILTON CYCLE. (Karp)
- NP completeness is everywhere.
- We cannot prove P$\neq$NP, etc.

What we are able to prove about lower bounds for NP problems

But what's the REAL cost

- In the early 1990's a reasonably successful systematic coping strategy was developed.
- Building on early work of Fellows and Langston, Mike Fellows and I suggested that focusing on the parameters of the input would yield insight into feasibility.
- When is the only thing you know about an input is its size?
- Answer Never, except in cryptography by design.
- We know things like planarity, engineering bounds, topology, depth, inductive designs.

Two Basic Examples

- VERTEX COVER
  Input: A Graph G.
  Parameter : A positive integer k.
  Question: Does G have a size k vertex cover? (Vertices cover edges.)
- DOMINATING SET
  Input: A Graph G.
  Parameter : A positive integer k.
  Question: Does G have a size k dominating set? (Vertices cover vertices.)

- VERTEX COVER is solvable by an algorithm $\mathcal{O}$ in time $f(k)|G|$, a behaviour we call fixed parameter tractability, (Specifically $1.28^k k^2 + c|G|$, with c a small absolute constant independent of k.)
- Whereas the only known algorithm for DOMINATING SET is complete search of the possible k-subsets, which takes time $\Omega(|G|^k)$.

- In the below I will mostly talk for convenience about graphs.
- I could just as easily be talking about many other areas.
- In the Computer Journal alone, there is biological, artificial intelligence, constraint satisfaction, geometric problems, scheduling, cognitive science, voting, combinatorial optimization, phylogeny. Model checking is the basis of Flum-Grohe.

Basic Definition(s)

- Setting : Languages $L \subseteq \Sigma^* \times \Sigma^*$.
- Example (Graph, Parameter).
- We say that a language L is fixed parameter tractable if there is a algorithm M, a constant C and a function f such that for all x, k,

$$(x, k) \in L \text{ iff } M(x) = 1 \text{ and}$$

$$\text{the running time of } M(x) \text{ is} f(k)|x|^C.$$

Parameters

- Without even going into details, think of all the graphs you have given names to and each has a relevant parameter: planar, bounded genus, bounded cutwidth, pathwidth, treewidth, degree, interval, etc, etc.
- Also nature is kind in that for many practical problems the input (often designed by us) is nicely ordered.

Positive techniques

- Elementary ones
- Almost elementary
- Logical metatheorems
- Limits

Kernelization

- I believe that the most important practical technique is called kernelization.
- pre-processing, or reducing For parameterized complexity $(I, k) \mapsto (I', k')$ and $|I'| \leq g(k)$.

Karsten Weihe's Train Problem

- Train Covering by Stations
  Instance: A bipartite graph $G = (V_S \cup V_T, E)$, where the set of vertices V_S represents railway stations and the set of vertices V_T represents trains. E contains an edge $(s, t), s \in V_s, t \in V_T$, iff the train t stops at the station s.
  Problem: Find a minimum set $V' \subseteq V_S$ such that V' covers V_T, that is, for every vertex $t \in V_T$, there is some $s \in V'$ such that $(s, t) \in E$.

Weihe's Solution

- Reduction Rule TCS1:
  Let $N(t)$ denote the neighbours of t in V_S. If $N(t) \subseteq N(t')$ then remove t' and all adjacent edges of t' from G. If there is a station that covers t, then this station also covers t'.
- Reduction Rule TCS2:
  Let $N(s)$ denote the neighbours of s in V_T. If $N(s) \subseteq N(s')$ then remove s and all adjacent edges of s from G. If there is a train covered by s, then this train is also covered by s'.

- European train schedule, gave a graph consisting of around $1.6 \cdot 10^5$ vertices and $1.6 \cdot 10^6$ edges.
- Solved in minutes.
- This has also been applied in practice as a subroutine in practical heuristical algorithms.

The Idea

- Reduce the parameterized problem to a kernel whose size depends solely on the parameter
- As compared to the classical case where this process is a central heuristic we get a provable performance guarantee.
- We remark that often the performance is much better than we should expect especially when elementary methods are used.

Vertex Cover

- Reduction Rule VC1:
  Remove all isolated vertices.
- Reduction Rule VC2:
  For any degree one vertex v, add its single neighbour u to the solution set and remove u and all of its incident edges from the graph.
- Note $(G, k) \rightarrow (G', k-1)$.
- (S. Buss) Reduction Rule VC3:
  If there is a vertex v of degree at least $k+1$, add v to the solution set and remove v and all of its incident edges from the graph.
- The result is a graph with no vertices of degree $> k$ and this can have a VC of size k only if it has $< k^2$ many edges.

Strategies for improving I: Bounded Search Trees

- Buss's algorithm gives crudely a $2n + k^{k^2}$ algorithm for k-VC.
- Here is another algorithm: (DF) Take any edge $e = v_1 v_2$. either v_1 or v_2 is in any VC. Begin a tree T with first children v_1 and v_2. At each child delete all edges covered by the v_i.
- repeat to depth k.
- Gives a $O(2^k \cdot n)$ algorithm.
- Now combine the two: Gives a $2n + 2^k k^2$ algorithm.

- It is worth remarking that there are problems notably FPT by bounded search tree (type checking in ML) that are not known to have polynomial size kernels, and some “provably” don’t.
- Another easy example for bounded search trees is PLANAR INDEPENDENT SET. (Start with a degree 5 vertex, branching rule of size 6)

Pruning trees and clever reduction rules

- If G has paths of degree 2, then there are simple reduction rules to deal with them first. Thus we consider that G is of min degree 3.
  Branching Rule VC2:
  If there is a degree two vertex v in G, with neighbours w_1 and w_2, then either both w_1 and w_2 are in a minimum size cover, or v together with all other neighbours of w_1 and w_2 are in a minimum size cover.
- Now when considering the kernel, for each vertex considered either v is included or all of its neighbours (at least) $\{p, q\}$ are included.
- Now the tree looks different. The first child nodes are labeled v or $\{p, q\}$, and on the right branch the parameter drops by 2 instead of 1. or similarly with the w_i case.

- Now the size of the search tree and hence the time complexity is determined by some recurrence relation.
- many, many versions of this idea with increasingly sophisticated reduction rules.
- This method has a 2005 (Fomin, Grandoni, Kratsch) incarnation called measure and conquer where the branching rules are given *rational valued* weights, and decisions as to what to do are figured out by optimization.
- For example the best exact algorithm for SET COVER and DOMINATING SET use this. (van Rooij-Bodlaender point out that this can be used for algorithm design as well.)
- Jianer Chen and others use this in many FPT algorithms such as the state of the art for FEEDBACK VERTEX SET and VERTEX COVER.

Shrink the Kernel

Theorem (Nemhauser and Trotter (1975))

There is a 2k kernel.

- Proof use LP $\{x_i \mid i \in V\}$. Solve

$$\sum_{i \in V} x_i \leq k, \text{ subject to}$$

$$x_i + x_j \geq 1, \text{ for } ij \in E(G),$$

$$x_i \in \{0, \frac{1}{2}, 1\}.$$

- The current champion using this approach is a $O^*(1.286^k)$ (Chen01) The best is $O^*(1.2745^k)$(Chen10 using this, iterative compression, struction, measure and conquer, and other methods).
- Here the useful O^* notation only looks at the exponential part of the algorithm.

Interactions

- Now we can ask lots of questions. How small can the kernel be?
- Notice that applying the kernelization to the unbounded problem yields a approximation algorithm.
- Using the PCP theorem we know that no kernel can be smaller that 1.36 k unless P=NP (Dinur and Safra) as no better approximation is possible. Is this tight?
- Assuming the "Unique Games Conjecture" the $2k$ kernel is tight (Khot etc).
- Actually we know that no $O^*(1+\epsilon)^k)$ algorithm is possible unless ETH fails.
- ETH n-variable 3SAT is not in DTIME($2^{o(n)}$). That is, not only does $P \neq NP$ but you can't beat complete search meaningfully.

Interleaving

- (Niedermeier and Rossmanith, 2000) showed that iteratively combining kernelization and bounded search trees often performs much better than either one alone or one followed by the other.
- Begin a search tree, and apply kernelization, then continue etc. Analysing the combinatorics shows a significant reduction in time complexity, which is very effective in practice.

Iterative Compression

- Reed, Smith and Vetta 2004. For the problem of "within k of being bipartite" (by deletion of edges).

Definition (Compression Routine)

A compression routine is an algorithm that, given a problem instance I and a solution of size k, either calculates a smaller solution or proves that the given solution is of minimum size.

An Example, VC again!

- $(G = (V, E), k)$, start with $V' = \emptyset$, and (solution) $C = \emptyset$.
- Add a new vertex v to both V' and C, $V' \leftarrow V' \cup \{v\},\ C \leftarrow C \cup \{v\}$.
- Now call the compression routine on the pair $(G[V'], C)$, where $G[V']$ is the subgraph induced by V' in G, to obtain a new solution C'. If $|C'| > k$ then we output NO, otherwise we set $C \leftarrow C'$.
- If we successfully complete the nth step where $V' = V$, we output C with $|C| \leq k$. Note that C will be an optimal solution for G. (Algo runs in time $O(2^k mn)$.)

- This was first successfully applied by Reed, Smith, Vetta to GRAPH BIPARTITIZATION. The algorithm is similar, building a minimal bipartitization at each step and using what we can call acceptable partitions for the search step.
- The best now is $O^*(3.83^k)$, and it works better with algorithm engineering (Gray Codes, tree pruning) with (e.g.) biological data Hüffner 2004.
- It is a crucial step for the best two algorithms for VERTEX COVER (Chen, Kanj, Xia 2010, $O^*(1.2745^k)$ and FEEDBACK VERTEX SET (Can I remove k vertices and get an acyclic graph?) (Cao, Chen, Liu, 2009).

Practice

- I remark that in practice these methods work much better than we might expect.
- Langston's work with irradiated mice, ETH group in Zurich, Karsten Weihe
- See The Computer Journal especially articles by Langston et al.

Less practical algorithms

- In what follows we look at algorithms that in general seem less practical but can sometimes work in practice.

Colour Coding

- k-Subgraph Isomorphism
  Instance: $G = (V, E)$ and a graph $H = (V^H, E^H)$ with $|V^H| = k$.
  Parameter: A positive integer k (or V^H).
  Question: Is H isomorphic to a subgraph in G?
- e.g. k-Path. Is H is a path of length k.
- Idea: Randomly colour the vertices of G with k colours and expect that there is a colourful solution; all the vertices of V' have different colours.
- G uniformly at random with k colors, a set of k distinct vertices will obtain different colours with probability $(k!)/k^k$. This probability is lower-bounded by e^{-k}, so we need to repeat the process e^k times to have high probability of obtaining the required colouring.

Derandomize

- k-Path has a $2^{O(k)} \cdot log\,|V|$ colourings, and $k!$ orderings. k-path in time $O(k \cdot |V|^2)$, using derandomization techniques.
- Recent improvements using multilinear detection and finite fields. (Kanj, Lokshtanov and Saurabh)

Bounded Width Metrics

- Graphs constructed inductively. Treewidth, Pathwidth, Branchwidth, Cliquewidth mixed width etc. k-Inductive graphs, plus old favourites such as planarity etc, which can be viewed as local width. Recently nowhere density. e.g.:

Definition

Let $G = (V, E)$ be a graph. A tree decomposition, TD, of G is a pair $(T, \mathcal{X})$ where

1. $T = (I, F)$ is a tree, and
2. $\mathcal{X} = \{X_i \mid i \in I\}$ is a family of subsets of V, one for each node of T, such that

(i) $\bigcup_{i \in I} X_i = V$,

(ii) *for every edge* $\{v, w\} \in E$, *there is an* $i \in I$ *with* $v \in X_i$ *and* $w \in X_i$, *and*

(iii) *for all* $i, j, k \in I$, *if* j *is on the path from* i *to* k *in* T, *then* $X_i \cap X_k \subseteq X_j$.

- This gives the following well-known definition.

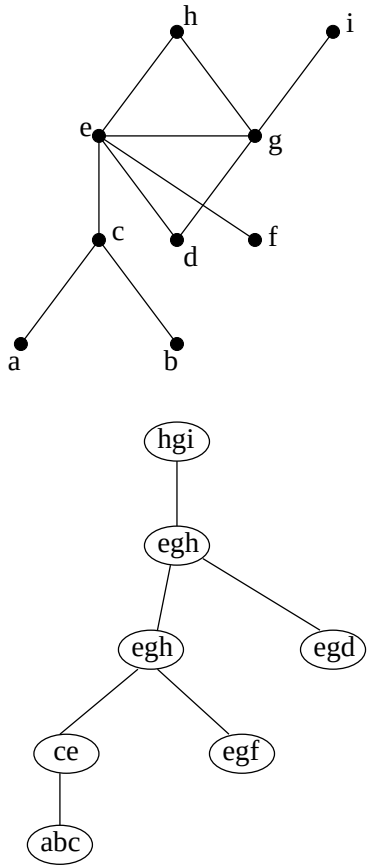
DEFINITION

The width of a tree decomposition $((I, F), \{X_i \mid i \in I\})$ is $\max_{i \in I} |X_i| - 1$. The treewidth of a graph G, denoted by $tw(G)$, is the minimum width over all possible tree decompositions of G.

The canonical method

- The following refers to any of these inductively defined graphs families. Notes that many commercial constructions of, for example chips are inductively defined.
  1. Find a bounded-width tree (path) decomposition of the input graph that exhibits the underlying tree (path) structure.
  2. Perform dynamic programming on this decomposition to solve the problem.

An example for Independent Set



$\emptyset$	a	b	c	ab	ac	bc	abc
0	1	1	1	2	-	-	-

Bodlaender's Theorem

- The following theorem is shows that treewidth is FPT. Improves many earlier results showing this. The constant is about 2^{35k^3}.

Theorem (Bodlaender)

k-Treewidth *is linear time FPT*

- Not practical because of large hidden O term.
- Unknown if there is a practical FPT treewidth algorithm
- Nevertheless approximation and algorithms specific to known decomps run well at least sometimes.

Monadic Second Order Logic

- Two sorted structure with variables for sets of objects.
- 1. **Additional atomic formulas:** For all set variables X and individual variables y, Xy is an MSO-formula.
  2. **Set quantification:** If ϕ is an MSO-formula and X is a set variable, then $\exists X\ \phi$ is an MSO -formula, and $\forall X\ \phi$ is an MSO-formula.
- Eg k-col

$$\exists X_1,,,\exists X_k \left(\forall x \bigvee_{i=1}^{k} X_i x \wedge \forall x \forall y \left(E(x,y) \rightarrow \bigwedge_{i=1}^{k} \neg (X_i x \wedge X_i y) \right) \right)$$

Model Checking

- Instance: A structure $\mathcal{A} \in \mathcal{D}$, and a sentence (no free variables) $\phi \in \Phi$.
  Question: Does $\mathcal{A}$ satisfy ϕ?
- PSPACE-complete for FO and MSO.

Courcelle's and Seese's Theorems

Theorem (Courcelle 1990)

The model-checking problem for MSO restricted to graphs of bounded treewidth is linear-time fixed-parameter tractable.

Detlef Seese has proved a converse to Courcelle's theorem.

Theorem (Seese 1991)

Suppose that $\mathcal{F}$ is any family of graphs for which the model-checking problem for MSO is decidable, then there is a number n such that, for all $G \in \mathcal{F}$, the treewidth of G is less than n.

The Frick Grohe Theorem

- considers the treewidth growth rate for families of graphs
- Examples Bounded degree, bounded treewidth, bounded genus, excluding a minor

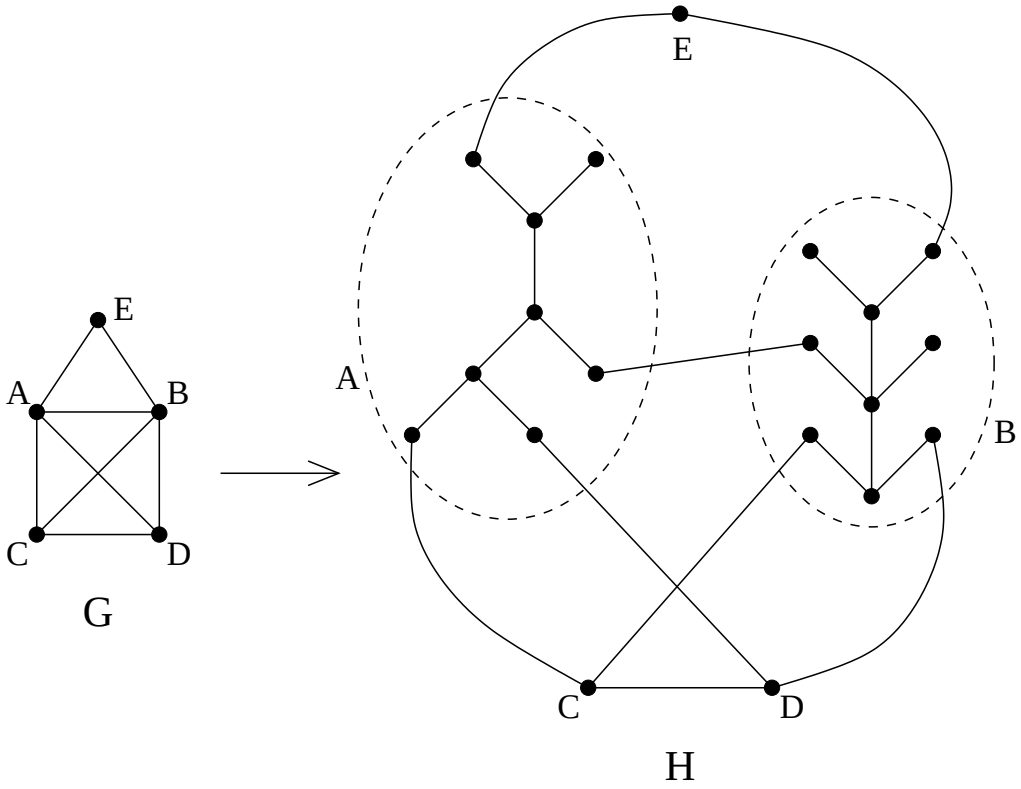
Theorem (Frick and Grohe 1999)

Parameterized problems that can be described as model-checking problems for FO are fixed-parameter tractable on classes of graphs of bounded local treewidth.

- For example Dominating set, Independent Set, or Subgraph isomorphism are FPT on planar graphs, or on graphs of bounded degree
- Recent extension of this for nowhere dense graphs.

More exotic methods

- minor ordering



- Robertson-Seymour Finite graphs are WQO's under minor ordering. $H \leq_{minor} G$ is $O(|G|^3)$ FPT for a fixed H.

- **THEOREM (MINOR-CLOSED MEMBERSHIP)**

  If $\mathcal{F}$ is a minor-closed class of graphs then membership of a graph G in $\mathcal{F}$ can be determined in time $O(f(k) \cdot |G|^3)$, where k is the collective size of the graphs in the obstruction set for $\mathcal{F}$.

  - Likely I won't have time to discuss what this means but see DF for more details.

- Natural basic hardness class: $W[1]$. Does not matter what it is, save to say that the analog of Cook's Theorem is SHORT NONDETERMINISTIC TURING MACHINE ACCEPTANCE
  Instance: A nondeterministic Turing Machine M and a positive integer k.
  Parameter: k.
  Question: Does M have a computation path accepting the empty string in at most k steps?

- If one believes the philosophical argument that Cook's Theorem provides compelling evidence that SAT is intractable, then one surely must believe the same for the parametric intractability of SHORT NONDETERMINISTIC TURING MACHINE ACCEPTANCE.
- Moreover, recent work has shown that if SHORT NTM is fpt then n-variable 3SAT is in DTIME($2^{o(n)}$)

- Given two parameterized languages $L, \widehat{L} \subseteq \Sigma^* \times \Sigma^*$, say $L \leq_{FPT} \widehat{L}$ iff there are (computable) $f, x \mapsto x', k \mapsto k'$ and a constant c, such that for all x,

$$(x, k) \in L \text{ iff } (x', k') \in \widehat{L},$$

  in time $f(k)|x|^c$.
- Lots of technical question still open here.

Analog of the Cook-Levin Theorem

Theorem (Downey, Fellows, Cai, Chen)

Weighted 3Sat$\equiv_{FTP}$ Short NTM Acceptance.

Weighted 3Sat

Input: A 3 CNF formula ϕ

Parameter: k

Question: Does ϕ has a satisfying assignment of Hamming weight k, meaning exactly k literals made true.

- Many hardness results, like e.g. Dominating Set, Clique, Independent Set, etc

W-Hierarchy

- Think about the usual poly reduction from SAT to 3SAT. It takes a clause of size p, and turns it into many clauses of size 3. But the weight control goes awry. A weight 4 assignment could go to anything.
- We don't think Weighted CNF Sat$\leq_{ftp}$Weighted 3 Sat.
- Gives rise to a hierarchy:

$$W[1] \subseteq W[2] \subseteq W[3] \ldots W[SAT] \subseteq W[P] \subseteq XP.$$

- XP is quite important, it is the languages which are in DTime$(n^f(k))$ with various levels of uniformity, depending on the choice of reductions.

- Notice that there are at least two ways to parameterize: Parameterize the part of the problem you want to look at and to parameterize the problem itself.
- This point of view makes this sometime a promise problem. Input something, promise it is parameterized, and ask questions about it.
- Two interpretations one with certificate one only with a promise. e.g. CLIQUEWIDTH, PATHWIDTH.
- Some recent work “lowers the hardness barrier”; perhaps giving better inapproximability results.

ETH

- Recall the exponential time hypothesis is (ETH) n-variable 3-SATISFIABILITY is not solvable in DTIME($2^{o(n)}$). (Impagliazzo Paturi and Zane.) Recall, this refinement of $P \neq NP$ says not only does $P \neq NP$ but you can't beat complete search meaningfully.
- This is seen an important refinement of $P \neq NP$ that is widely held to be true.
- It is related to FPT as we now see.

The Minimob

- INPUT A parametrically miniature problem
  QUESTION Is it a yes instance?
  e.g. INPUT a graph G of size $k \log n$ with n, k in unary.
  Does it have a vertex cover of size d?
- Get mini Vertex cover, mini Dominating set, Minisat etc.
- Core problem: minicircuitsat.

Theorem (Chor, Fellows and Juedes ; Downey et. al.)

The $M[1]$ complete problems such as Min-3Sat *are in FPT iff the exponential time hypothesis fails.*

- That is, more or less, EPT is the "same" as $M[1] \neq FPT$.
- And now we have a method of demonstrating no good subexponential algorithm; Show M[1] hardness.
- Chen-Grohe established an isomorphism between the complexity degree structures.

XP-optimality

- This new programme regards the classes like W[1] as artifacts of the basic problem of proving hardness under reasonable assumptions, and strikes at membership of XP, and realizes the dream of upper and lower bounds matching.
- Eg Independent Set and Dominating Set which certainly are in XP. But what's the best exponent we can hope for for slice k? They are clearly solvable in time $O(n^{k+1})$.

Theorem (Chen et. al 05)

The following hold:

(i) Independent Set *cannot be solved in time* $n^{o(k)}$ *unless* FPT=M[1].

(ii) Dominating Set *cannot be solved in time* $n^{o(k)}$ *unless* FPT=M[2].

- The proofs recycle and miniaturize various NP and W[1] completeness results.
- Many recent results of similar ilk based on ETH or SETH, such as results on treewidth etc.
- For example, Most known algorithms for e.g. treewidth problems are optimal.

Where else?

- Another area is approximation. Here we ask for an algorithm which either says "no solution of size k" or here is one of size $2k$ (say).
- For example Bin Packing is has to $(k, 2k)$-approx, but k-Independent Dominating Set has not approx of the form $(k, F(k))$ for any computable F unless $FPT = W[1]$. (DFMccartin)
- Flum Grohe show that all natural $W[P]$ complete problems don't have approx of the form $(k, F(k))$ for any computable F unless $FPT = W[P]$.
- Yijia Chen and Bingkai Lin (submitted) recently solved the 30 year old problem showing that Dominating Set has no multiplicative approximation unless $W[1] = FPT$. (Allowing new inapproximability proofs without PCP!)

Remember kernelization?

- When can we show that a FPT problem likely has no polynomial size kernel?
- Notice that if P=NP then all have constant size kernel, so some reasonable assumption is needed.

A GENERIC LOWER BOUND ENGINE

DEFINITION (BODLAENDER, DOWNEY, FELLOWS, HERMELIN)

An OR-distillation algorithm for a classical problem $L \subseteq \Sigma^*$ (like SAT is an algorithm that

- receives as input a sequence $(x_1, \ldots, x_t)$, with $x_i \in \Sigma^*$ for each $1 \leq i \leq t$,
- uses time polynomial in $\sum_{i=1}^{t} |x_i|$,
- and outputs a string $y \in \Sigma^*$ with
  1. $y \in L \iff x_i \in L$ for some $1 \leq i \leq t$.
  2. $|y|$ is polynomial in $\max_{1 \leq i \leq t} |x_i|$.

- Similarly AND-distillation.

The Fortnow-Santhanam Lemma

Lemma (Fortnow and Santhanam 2007)

If any NP complete problem has a distillation algorithm then $PH = \Sigma_3^P$. That is, the polynomial time hierarchy collapses to three or fewer levels

- Here Σ_3^p is $NP^{NP^{NP}}$.
- Strictly speaking the prove that $co - NP \subseteq NP \backslash poly$.

How does this relate to kernelization?

- Bodlaender, Downey, Fellows, Hermelin identified a property called composition and proved the following.

Lemma (Bodlaender, Downey, Fellows, Hermelin)

Let L be a compositional parameterized problem whose derived classical problem L_c is NP-complete. If L has a polynomial kernel, then L_c is also distillable.

- One guarantee that a problem is compositional if the disjoint union of two instances is a yes iff at least one is a yes with the same parameter. (BDFH)

Examples

- k-Path, k-Cycle, k-Cheap Tour, k-Exact Cycle, and k-Bounded Treewidth Subgraph
- k, σ-Short Nondeterministic Turing Machine Computation (Needs work)
- Many recent examples, Bodlaender, Kratch, Lokshantov, Saurabh etc. Also using (poly,poly)-reductions, co-nondeterminism, small interactive protocols, etc.
- Now a big cottage industry.

AND-composition and Distillation

- A then PhD student Andrew Drucker from MIT (2012) who has shown this also implies collapse. This implies all the below don't have poly kernels, unless.... The proof is remarkable.
- Applications: Graph width metrics:
- Cutwidth, Treewidth, problems with treewidth promises, eg.. colouring

Other results

- Burhmann and Hitchcock: There are no subexponential size hard sets for NP unless PH collapses. (Ie many hard instances)
- Using transformations, Bodlaender, Thomasse and Yeo show that Disjoint Cycles, Hamilton Circuit Parameterized by Treewidth etc don't have poly kernels unless collapse.
- Also the important Disjoint Paths, famously FPT by Robertson and Seymour.
- Similarly using Dell-Mecklebeek, Kratz showed the non-poly-kernelizability of k-Ramsey.
- Fernau et. al. have shown that there are problems with Poly Turing Kernels but no poly kernels unless collapse.(!), and these are natural related to spanning trees (Namely Directed k Leaf Spanning Tree).

The biggest challenge

- Explain why heuristics deliver. e.g.
- Parosh-Abdullah infinite state verification,
- Simplex method
- Sat solvers (Remember: Sat was supposed to be toxic)
- Huge linear programmes.
- Incorporate bounded rationality.
- Practical off shelf toolkit.
- Combine with, or beat things like Map Reduce.

Some References

- Parameterized Complexity, springer 1999 DF
- Invitation to Parameterized Algorithms, 2006 Niedermeier, OUP
- Parameterized Complexity Theory, 2006, Springer Flum and Grohe
- Theory of Computing Systems, Vol. 41, October 2007
- Parameterized Complexity for the Skeptic, D,
- The Computer Journal, (ed Downey, Fellows, Langston)
- Confronting intractability via parameters, Downey Thilikos, Computing Reviews
- Fundamentals of Parameterized Complexity, Downey-Fellows.
- Parameterized Algorithms, Cygan et. al.

What should you do?

- You should buy that new wonderful book...(and its friends)
- Thanks!