Basic Parametric Complexity II: Intractability

Rod Downey
Victoria University
Wellington
Isaac Newton Institute, Cambridge

BCTCS, April 2012

This lecture:

- Basic Definitions
- Basic Hardness results
- Kernelization lower bounds

Reminder

- A mathematical idealization is to identify "Feasible" with P. (I won't even bother looking at the problems with this.)
- With this assumption, the theory of NP-hardness is an excellent vehicle for mapping an outer boundary of intractability, for all practical purposes.
- Indeed, assuming the reasonable current working assumption that NTM acceptance is $\Omega(2^n)$, NP-hardness allows for practical lower bound for exact solution for problems.
- A very difficult practical and theoretical problem is "How can we deal with P?".
- More importantly how can we deal with $P - FEASIBLE$, and map a further boundary of intractability.

- Arora 1996 gave a $O(n^{\frac{3000}{\epsilon}})$ PTAS for EUCLIDEAN TSP
- Chekuri and Khanna 2000 gave a $O(n^{12(\log(1/\epsilon)/\epsilon^8)})$ PTAS for MULTIPLE KNAPSACK
- Shamir and Tsur 1998 gave a $O(n^{2^{2^{\frac{1}{\epsilon}}}-1)})$ PTAS for MAXIMUM SUBFOREST
- Chen and Miranda 1999 gave a $O(n^{(3mm!)^{\frac{m}{\epsilon}+1}})$ PTAS for GENERAL MULTIPROCESSOR JOB SCHEDULING
- Erlebach et al. 2001 gave a $O(n^{\frac{4}{\pi}(\frac{1}{\epsilon^2}+1)^2(\frac{1}{\epsilon^2}+2)^2})$ PTAS for MAXIMUM INDEPENDENT SET for geometric graphs.

- Deng, Feng, Zhang and Zhu (2001) gave a $O(n^{5\log_{1+\epsilon}(1+(1/\epsilon))})$ PTAS for UNBOUNDED BATCH SCHEDULING.
- Shachnai and Tamir (2000) gave a $O(n^{64/\epsilon+(log(1/\epsilon)/\epsilon^8)})$ PTAS for CLASS-CONSTRAINED PACKING PROBLEM (3 cols).

Reference	Running Time for a 20% Error
Arora (Ar96)	$O(n^{15000})$
Chekuri and Khanna (CK00)	$O(n^{9,375,000})$
Shamir and Tsur (ST98)	$O(n^{958,267,391})$
Chen and Miranda (CM99)	$> O(n^{10^{60}})$ (4 Processors)
Erlebach et al. (EJS01)	$O(n^{523,804})$
Deng et. al (DFZZ01)	$O(n^{50})$
Shachnai and Tamir (ST00)	$O(n^{1021570})$

The Running Times for Some Recent PTAS's with 20% Error.

What is the problem here?

- Arora (1997) gave a PTAS running in nearly linear time for Euclidian TSP. What is the difference between this and the PTAS's in the table. Can't we simply argue that with more effort all of these will eventually have truly feasible PTAS's.
- The principal problem with the baddies is that these algorithms have a factor of $\frac{1}{\epsilon}$ (or worse) in their exponents.
- By analogy with the situation of *NP* completeness, we have some problem that has an exponential algorithm. Can't we argue that with more effort, we'll find a much better algorithm? As in Garey and Johnson's famous cartoon, we cannot seem to prove a better algorithm. BUT we prove that it is NP hard.

I'M DUBIOUS; EXAMPLE?

- Then assuming the working hypothesis that there is basically no way to figure out if a NTM has an accepting path of length n except trying all possibilities there is no hope for an exact solution with running time significantly better than 2^n. (Or at least no polynomial time algorithm.)
- Our new working hypothesis that there is basically no way to figure out if a NTM has an accepting path of length k except trying all possibilities. Note that there are $\Omega(n^k)$ possibilities. (Or at least no way to get the "k" out of the exponent or an algorithm deciding k-STEP NTM.)
- One then defines the appropriate reductions from k-STEP TURING MACHINE HALTING to the PTAS using $k = \frac{1}{\epsilon}$ as a parameter to argue that if we can "get rid" of the k from the exponent then it can only be if the working hypothesis is wrong.

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,

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

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

- 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 intractible, 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 COOK'S THEOREM

- Analog of Cook's 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 weigth k, meaning exactly k literals made true.

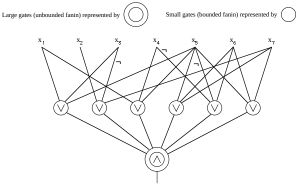
W-HIERARCHY

- 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 heirarchy:

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

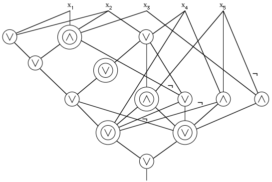
- XP is quite important, it is the languages which are in DTIME$(n^{f}(k))$ with various levels of uniformity, depending on the choice of reductions.

THE CIRCUIT VIEW



A 3CNF Formula is a large and of small or's.

$\varphi = (x_1 + x_3 + x_5)(x_2 + \neg x_3 + x_7)(x_1 + x_4 + x_5)(x_5 + x_6 + x_7)(\neg x_4 + \neg x_5 + x_6)(x_5 + x_6 + x_7)$

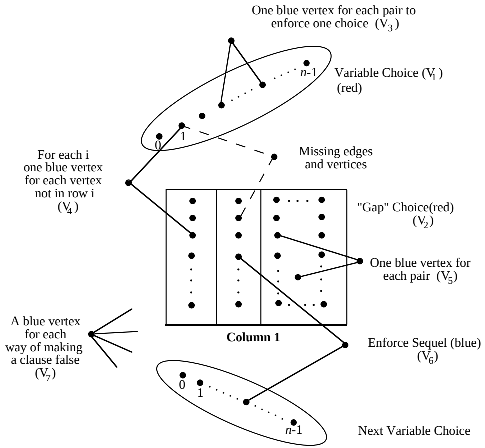


A Weft 2 Depth 5 Decision Circuit.

FIGURE: Examples of circuits

- Example: $W[1,s] \equiv_{fpt}$ Antimonotone $W[1,s]$.
- $W[1,s]$ is the problem based around weight k for circuits of depth 2, and maximum fanin for the top Or gates of size s. (board) Specifically, $W[1,s]$ are the problems fpt reducible to weighted s-SAT.
- Red/Blue nonblocker: Input A Red/Blue bipartite graph $G = (V_R \cup V_B = V, E)$ with the blue vertices of degree $\leq s$. Question is there a set of red vertices V' of size k, such that each blue vertex has a neighbour not in V'?
- $\Pi_{u \in V_B} \Sigma_{x_i \in N[u] \cap V_R} \overline{x_i}$.

- X is a boolean expressiuon in CNF of max clause size s.
- m clauses $C_1, \ldots, C_m$.
- Construct $G = (V_R \cup V_B, E)$ with a nonblocker of size $2k$ iff X has a satisfying assignment of weight k.


One blue vertex for each pair to enforce one choice (V_3)
Variable Choice (V_1) (red)
0
1
n-1
For each i one blue vertex for each vertex not in row i (V_4)
Missing edges and vertices
"Gap" Choice(red) (V_2)
One blue vertex for each pair (V_5)
A blue vertex for each way of making a clause false (V_7)
Column 1
Enforce Sequel (blue) (V_6)
0
1
n-1
Next Variable Choice

Theorem

$W[1,s] = W[1,2] =$ Antimonotone $W[1,2] = W[1]$.

- Given an antimonotone $W[1,s]$ circuit C, we construct an antimonotone $W[1,2]$ circuit such that $W[1,s]$ has a weight k accepting input iff C' has a weight k' one where

$$k' = k2^k + \sum_{i=2}^{s} \binom{k}{i}$$

- Let the input variable $x[j], j = 1, \ldots, n$ to C, create new variables for each possible set of at most s and at least 2 of the $x[i]$'s. Let $A_1, \ldots, A_p$ be the enumeration of all such sets. These are the circuit inputs. Think of them as variables $v[i]$ representing A_i.
- Rearrange the circuit using these variables and enforcement variables.

- For each top or gate g choose the correct A_i's for the input lines to g in C, all negated of course.
- Now add an enforcement mechanism for consistency of the $v[i]$. This is done by 2^k copies of each of the $x[j]$, $x[j,d] : d = 1, \ldots, 2^k$. Now write out the exponentially many implications saying that the simulation is faithful.
- Details DF or see FG, for their approach using logic.

- CLIQUE is $W[1]$-complete as is INDEPENDENT SET.

THEOREM (DF, CAI AND CHEN)

SHORT TURING MACHINE ACCEPTANCE *is* $W[1]$ *complete.*

- Generic reduction for hardness from CLIQUE. (write the vertices on the tape and check in $\binom{k}{2}$ moves if they are adjacent.) Membership is another generic simuation by a circuit.

- *XP* has k-CAT AND MOUSE GAME and some other games ((DF99a)),
- $W[P]$ has LINEAR INEQUALITIES, SHORT SATISFIABILITY, WEIGHTED CIRCUIT SATISFIABILITY ((ADF95)) and MINIMUM AXIOM SET((DFKHW94)).
- Then there are a number of quite im portant problems from combinatorial pattern matching which are $W[t]$ hard for all t: LONGEST COMMON SUBSEQUENCE ($k =$ number of seqs.,$|\Sigma|$-two parameters) ((BDFHW95)), FEASIBLE REGISTER ASSIGNMENT, TRIANGULATING COLORED GRAPHS, BANDWIDTH, PROPER INTERVAL GRAPH COMPLETION ((BFH94)), DOMINO TREEWIDTH ((BE97)) and BOUNDED PERSISTE NCE PATHWIDTH ((McC03)).
- $W[2]$ include WEIGHTED $\{0,1\}$ INTEGER PROGRAMMING, DOMINATING SET ((DF95a)), TOURNAMENT DOMINATING SET ((DF95c)) UNIT LENGTH PRECEDENCE CONSTRAINED SCHEDULING (hard) ((BF95)), SHORTEST COMMON SUPERSEQUENCE (k)(hard) ((FHK95)), MAXIMUM LIKELIHOOD DECODING (hard), WEIGHT DISTRIBUTION IN LINEAR CODES (hard), NEAREST VECTOR IN INTEGER LATTICES (hard) ((DFVW99)), SHORT PERMUTATION GROUP FACTORIZATION (hard).
- $W[1]$ we have a collection including k-STEP DERIVATION FOR CONTEXT SENSITIVE GRAMMARS, SHORT NTM COMPUTATION, SHORT POST CORRESPONDENCE, SQUARE TILING ((CCDF96)), WEIGHTED q–CNF SATISFIABILITY ((DF95b)), VAPNIK–CHERVONENKIS DIMENSION ((DEF93)) LONGEST COMMON SUBSEQUENCE (k, $m =$ LENGTH OF COMMON SUBSEQ.) ((BDFW95)), CLIQUE, INDEPENDENT SET ((DF95b)), and MONOTONE DATA COMPLEXITY FOR RELATIONAL DATABASES

- (Chandra and Merlin, 77) the complexity of query languages in the study of database theory.
- Vardi 1982 notes that classical complexity seemed wrong: suggested evaluation of a query when the size of the query was fixed as a function of the size of the database

- (Standard sort of problem) Input: A boolean query φ and a database instance I.
  Parameter: Some parameter of φ, such as the size of φ. (or its complexity etc)
  Problem: Evaluate φ in I.
- Yananakakis 1995 suggested that parameterized complexity good framework to address this.
- (Downey-Fellows-Taylor, 95 Papadimitriou-Yannakakis 97) The morning edition of the news is bad.
- Papadimitriou and Yannakakis systematically also looked at other parameters such as bounding the number of variables following ideas of Vardi (Va95). They looked at positive queries, conjunctive queries, first order theories and datalog ones and found them to be all $W[1]$ hard and at various levels of the W-hierarch y.
- Other analyses look at other parametric aspects and give even more bad news. (e.g. Demri, Laroussinie and Schnoebelen (DLS02).)

- You might well ask now, with “good” news like this provided by parameterized complexity, what use is it? You could argue that once we knew these problems were NP-hard and likely PSPACE complete. Now we know that even when you bound the obvious parameters then they are still hard!
- One interpretation is that we should learn to live with this by searching for new coping strategies

- The parametric point of view is to continue the dialog with the problem. To cope by finding new, and maybe more appropriate parameters.
- (Frick and Grohe (FrG02), Flum and Grohe (FG02a)) Let C be a class of relational database instances such that underlying graph of instances in C are any of the following forms: bounded degree, bounded treewidth, bounded local treewidth, planar or have an excluded minor. Then the query evaluation problem for the relational calculus on C is FPT.
- Frick and Grohe also looked at things beyond query languages, such as XML and temporal logics such as LTL and CTL* are used for specification languages for automated verification, proving sometimes practical FPT results.

- 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 hypthesis is (ETH) n-variable 3-SATISFIABILITY is not solvable in DTIME($2^{o(n)}$). (Impagliazzo Paturi and Zane.)
- 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 minature problem QUESTION Is it in the class
  e.g. INPUT a graph G of size $klogn$ with n 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 insomorphism between the complexity degree structures.
- Fellows conjectures that PCP like techniques will show

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 ofXP.
- Eg INDEPENDENT SET and DOMINATING SET which certainly are in XP. But what's the best exponent we can hope for for slice k?

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].

- We can formulate a notion of counting complexity and get $\#W[1]$ (Flum and Grohe, McCartin). A sample theorem:

THEOREM (FG)

Counting k-cycles in a graph is $\#W[1]$-complete. (The existence problem is FPT, as we see next time.)

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 $FPT = W[P]$.

Where else?

- EPT and bounded alternation. (Flum, Grohe and Weyer) A parameterized problem is in EPT iff it is solvable in time $2^{0(k)}|x|^c$.
- Partial formulation of an operator calculus such as *BP* analog. (Downey, Fellows, Regan) (Moritz Meuller) Unknown if analogs of, say, Toda's Theorem holds.
- A few results on parameterized logspace, and even less on parameterized PSPACE, except through game analogs.

ETP

- (Flum, Grohe and Weyer) A parameterized problem is in EPT iff it is solvable in time $2^{0(k)}|x|^c$.
- Things shown in FPT by elementary methods tend to be in EPT. e.g. Vertex Cover.
- (EPT reductions)

$$L \leq_{EPT} L' \text{ iff } \langle x, k\rangle \in L \text{ iff } \langle x', k'\rangle \in L',$$

where $x \mapsto x'$ in time $2^{0(k)}|x|^c$, but

$$\langle k, x\rangle \mapsto k' \text{ has } k' \leq d(k + \log |x|).$$

- Notice that this is not parameterized.
- This gives the E-hierarchy.
- Flum, Grohe and Weyer have shown that the various modle checking problems from logic are complete at higher levels of this hierarchy.
- Earlier Frick and Grohe had shown that model checking for trees is not solvable in FPT with the $f(k)$ elementary in

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.

DEFINITION (BODLAENDER, DOWNEY, FELLOWS, HERMELIN)

labelDefinition: DistillationA OR-distillation algorithm for a classical problem $L \subseteq \Sigma^*$ 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.

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 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 $co - NP \subseteq NP \backslash poly$.

THE PROOF

- Let L be NP complete. We show that $\overline{L}$ is in NP\poly if L has dist.
- Let $\overline{L}_n = \{x \notin L : |x| \leq n\}$.
- Given any $x_1, \ldots, x_t \in \overline{L}_n$, the distillation algorithm $\mathcal{A}$ maps $(x_1, \ldots, x_t)$ to some $y \in \overline{L}_{n^c}$, where c is some constant independent of t.

- The main part of the proof consists in showing that there exists a set $S_n \subseteq \overline{L}_{n^c}$, with $|S_n|$ polynomially bounded in n, such that for any $x \in \Sigma^{\leq n}$ (PHP) we have the following:
  - If $x \in \overline{L}_n$, then there exist strings $x_1, \ldots, x_t \in \Sigma^{\leq n}$ with $x_i = x$ for some i, $1 \leq i \leq t$, such that $\mathcal{A}(x_1, \ldots, x_t) \in S_n$.
  - If $x \notin \overline{L}_n$, then for all strings $x_1, \ldots, x_t \in \Sigma^{\leq n}$ with $x_i = x$ for some i, $1 \leq i \leq t$, we have $\mathcal{A}(x_1, \ldots, x_t) \notin S_n$.
- to decide if $x \in \overline{L}$, guess t strings $x_1, \ldots, x_t \in \Sigma^{\leq n}$, and checks whether one of them is x. If not, it immediately rejects. Otherwise, it computes $\mathcal{A}(x_1, \ldots, x_t)$, and accepts iff the output is in S_n. It is immediate to verify that M correctly determines (in the non-deterministic sense) whether $x \in \overline{L}_n$.

How does this relate to kernelization?

Definition (Bodlaender, Downey, Fellows, Hermelin)

A OR-composition algorithm for a parameterized problem $L \subseteq \Sigma^* \times \mathbb{N}$ is an algorithm that

- receives as input a sequence $((x_1, k), \dots, (x_t, k))$, with $(x_i, k) \in \Sigma^* \times \mathbb{N}^+$ for each $1 \leq i \leq t$,
- uses time polynomial in $\sum_{i=1}^{t} |x_i| + k$,
- and outputs $(y, k') \in \Sigma^* \times \mathbb{N}^+$ with
  1. $(y, k') \in L \iff (x_i, k) \in L$ for some $1 \leq i \leq t$.
  2. k' is polynomial in k.

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.

Examples

Lemma (Bodlaender, Downey, Fellows, Hermelin)

Let L be a parameterized graph problem such that for any pair of graphs G_1 and G_2, and any integer $k \in \mathbb{N}$, we have $(G_1, k) \in L \vee (G_2, k) \in L \iff (G_1 \cup G_2, k) \in L$, where $G_1 \cup G_2$ is the disjoint union of G_1 and G_2. Then L is compositional.

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, Lokshantox, Saurabh etc. Also using (poly,poly)-reductions.

AND-composition and Distillation

- Three days ago I received a manuscript from a student Andrew Drucker from MIT who has shown this also implies collapse. This implies all the below don't have poly kernels.
- Oracle results
- Applications: Graph width metrics:
- Cutwidth, Treewidth, problems with treeewidth promises, eg.. colouring

Other results

- BDFH show that there are problems in ETP (FPT in time $O^*(2^{O(k)})$) without polynomial time kernels.
- Fortnow and Santhanam: Satisfiability does not have PCP's of size polynomial in the number of variables unless PH collapse.
- The Harnik-Noar approach to constructing collision resistant hash functions won't work unless PH collapses.
- Burhmann and Hitchcock: There are no subexponential size hard sets for NP unless PH collapses. (Ie many hard instances)
- Chen Flum Müller: Many results, e.g. parameterized SAT has no subexponential "normal" (strong) kernelization unless ETH fails.

- Using transformations, Bodlaender, Thomass'/ e and Yeo show that DISJOINT CYCLES, HAMILTON CIRCUIT PARAMETERIZED BY TREEWIDT H etc don't have poly kernels unless collapse.
- Also the important DISJOINT PATHS, famously FPT by Robertson and Se ymour.
- 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).

Turing kernelizations

- Possible to avoid the material above. e.g. Binkele-Raible, Fernau, Fomin, Lokshantov, Saurabh and Villanger, k-Leaf Out Tree (directed spanning tree with k-leaves)
- The rooted case has a poly kernel.
- The unrooted case does not unless .....
- So it has a poly Turing Kernel
- no lower bounds by recent work on completeness.

DEFINITION (TURING KERNELIZATION)

A *Turing Kernel* consists of

1. (I) Three parameterized languages L_1 and L_2 (typically $L_1 = L_2$) with $L_i \subset \Sigma^* \times \mathbb{N}$ and $L_3 \subseteq \Sigma^* \times L_1$
   (II) and a computable function g
   (III) and a polynomial time computable function $f : \Sigma^* \times \Sigma^* \times \mathbb{N} \to \Sigma^* \times \mathbb{N}$, $\langle \sigma, \tau, k \rangle \mapsto \langle \rho, k' \rangle$ with $|\tau| \leq |\sigma|$ *and* $|f(\langle \sigma, \tau, k \rangle)| \leq g(k)$ such that
   (IV) for all σ, τ, k,

$$\langle \sigma, \tau, k \rangle \in L_3 \text{ iff } \langle \rho, k' \rangle \in L_2.$$

2. Plus an oracle Turing procedure Φ, running in polynomial time on L_1, with oracle L_2, such that on input $\langle \sigma, k \rangle$, if the procedure queries $\langle \tau, k \rangle$ then it answers yes iff $f(\langle \sigma, \tau, k \rangle) \in L_3$

- The idea is that on input $\langle \sigma, k \rangle$ Φ works like a normal polynomial time machine except on oracle queries, it converts the query to a query of the kernel *determined by the query* τ.
- In the case of k-LEAF OUT BRANCHING,
  (I) L_1 are pairs $\langle G, k \rangle$ consisting of digraphs with k or more leaf outbranchings.
  (II) L_2 are pairs $\langle \hat{G}, k \rangle$ consisting of *rooted* digraphs (the input $\hat{G}$ would specify a root r) with with k or more leaf outbranchings.
  (III) L_3 are triples $\langle r, G, k \rangle$ consisting of yes instances of whether G has a k or greater leaf outbranching rooted at r.

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, proceedings CCC, Aarhus, (see my homepage)
- The Computer Journal, (ed Downey, Fellows, Langston)
- Confronting intractability via parameters, Downey Thilikos, Computing Reviews
- Fundamentals of Parameterized Complexity, Downey-Fellows, this year.

What should you do?

- You should buy that new wonderful book...(and its friends)
- muchas gratsias