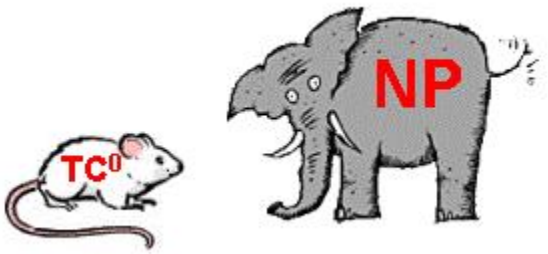
The Complexity Zoo

Scott Aaronson
www.ScottAaronson.com

LaTeX Translation by
Chris Bourke
cbourke@cse.unl.edu

417 classes and counting



Contents

1 About This Document

What is this? Well its a PDF version of the website www.ComplexityZoo.com typeset in LaTeX using the `complexity` package. Well, what's that? The original Complexity Zoo is a website created by Scott Aaronson which contains a (more or less) comprehensive list of Complexity Classes studied in the area of theoretical computer science known as Computational Complexity.

I took on the (mostly painless, thank god for regular expressions) task of translating the Zoo's HTML code to LaTeX for two reasons. First, as a regular Zoo patron, I thought, "what better way to honor such an endeavor than to spruce up the cages a bit and typeset them all in beautiful LaTeX."

Second, I thought it would be a perfect project to develop `complexity`, a LaTeX package I've created that defines commands to typeset (almost) all of the complexity classes you'll find here (along with some handy options that allow you to conveniently change the fonts with a single option parameters). To get the package, visit my own home page at `http://www.cse.unl.edu/~cbourke/`.

In addition, I've used the `hyperref` package to preserve the ability to link to references, internal anchors, etc. In general, any text in red represents an internal link that will take you to the reagent page (mainly for complexity classes). Green links will are used for references and will take you to the bibliography entry it refers to. Finally, magenta links refer to external URLs (for which you'll need a web browser/email program).
The content itself is entirely due to Scott. Other than these few paragraphs, all the original effort was his. I take full responsibility for all errors in translation and would appreciate any and all corrections.

Thanks to Scott for allowing me to pursue this project and for the original efforts made in the web-based ComplexityZoo.

–Chris Bourke cbourke@cse.unl.edu

2 Introductory Essay

I created the Complexity Zoo with three audiences in mind.

First, me. Before my zookeeping foray, I spent a week trying to put AM outside QMA relative to an oracle, only to learn that this followed trivially from two known results: that of Vereshchagin [Ver92] that AM is outside PP relative to an oracle, and that of Kitaev and Watrous (unpublished, but mentioned in [Wat00]) that QMA is in PP. What to do next? One option would be to work on putting SZK outside QMA relative to an oracle. But instead I decided that, before hoisting one more brick onto the vast edifice of complexity, I'd do well to take stock of what was already known. Some theorists seem able to hold in their minds, in one instant, the results of every FOCS, STOC, and Complexity paper ever published. Not every proof, of course - but those can be looked up if one knows the results and to whom they're due.

I am not one of those theorists. The sprawling web of known relations among complexity classes—containments, oracle separations, random oracle separations, lowness results, the occasional inequality - is not fixed in my memory like the English language. And so it's largely for my own benefit that I recorded a chunk of what's known in one unwieldy HTML file.

The second audience is other theorists and theory students, who might find the Zoo to have a few advantages as a reference. First, inspired by Eric Weisstein's famed World of Mathematics, it links liberally between entries. Second, it can be updated regularly—much of its current content is not yet in any book as far as I know. Third, it takes a democratic approach to complexity classes, with big hitters like NP listed alongside the lesser-known mAL, PODN, and QACC_0. Any class is fair game so long as something nontrivial has been said about it in the literature.

The third audience comprises programmers, mathematicians, physicists, and anyone else who bumps into complexity classes occasionally. With this audience in mind I've kept the writing informal; those who need a precise definition of a class should consult the references. I hope that non-theorists, even if they don't understand everything, will at least find some amusement in the many exotic beasts that complexity theory has uncovered.

Comments, corrections, and additions (of classes, results, references...) are most welcome; send to aaronson@cs.berkeley.edu.

2.1 Recommended Further Reading

- [Pap94a] *Computational Complexity* (1994) by C. H. Papadimitriou. The standard text, and an ideal starting place for beginners.

- [Joh90] "A Catalog of Complexity Classes" (1990) by D. S. Johnson (Chapter 2 in the *Handbook of Theoretical Computer Science*, Volume A). Close in spirit to the Zoo.

- [HO02] *The Complexity Theory Companion* (2002), by L. A. Hemaspaandra and M. Ogihara. A lively guide to structural complexity. Has a "Rogues' Gallery" of classes at the end, including such obscure zoo denizens as US and SF_k (though Prof. Hemaspaandra emphasizes to me that the book is "*mostly* about NP, PH, etc. - pretty normal, standard stuff").

- http://fortnow.com/lance/complog/: Lance Fortnow's Computational Complexity Web Log. Includes a "Complexity Class of the Week."

2.2 Other Theory Compendia

- [GJ79] *Computers and Intractability: A Guide to the Theory of NP-Completeness* (1979) by M. R. Garey and D. S. Johnson.

- http://www.nada.kth.se/~viggo/wwwcompendium/ A Compendium of NP Optimization Problems, web site by P. Crescenzi and V. Kann et al.

2.3 Errors?

Omissions? Misattributions? Your favorite class not here? Email aaronson at ias dot edu. Please include a reference, or better yet a link to a paper if available.

3 Pronunciation Guide

Unfortunately, there are no accepted conventions for pronouncing the names of rarer complexity classes. Usually the safest bet is just to spell out a whole acronym; even when vowels are present, this is easier than wrapping one's tongue around (say) PODN or NISZK. The major exceptions are words and word prefixes, which are pronounced as such: for example, co, mod, log, lin, few, poly, exp, gap, time, space, amp, av, sel, sat. Also, the "TAS" in PTAS, FPTAS, and EPTAS is generally pronounced "tahz."

As for symbols, the "+" of $\oplus$L (which actually has a circle around it in LaTeX rendering) is pronounced "parity"; while the "#" of #P is pronounced "sharp" (some textbooks also list "pound" or "number" as acceptable, but I've never heard the latter two used in real life). The "/" of P/poly is pronounced "slash." The act of subscripting is left unvocalized, so that $\mathsf{Mod}_k\mathsf{P}$ is "mod-k-p" rather than "mod-sub-k-p." Superscripting, on the other hand, *is* vocalized: P^{NP} is "P to the NP" (but not "P to the NP power" - it's an oracle, for Godsakes).

Finally, Lance Fortnow has informed me that $\mathsf{C}_{=}\mathsf{P}$ and its cousin $\mathsf{C}_{=}\mathsf{L}$ are pronounced "C equals P" and "C equals L" respectively. This could lead to confusion in sentences such as "coNQP equals C equals P."

Glossary

0-1-NP$_{\mathbb{C}}$
1NAuxPDA^{p}
#AC^{0}
#L
#L/poly
#P
#W[t]
⊕EXP
⊕L
⊕L/poly
⊕P
⊕SAC^{1}
A$_{0}$PP
AC
AC^{0}
AC$^{0}[m]$
ACC^{0}
AH
AL
AlgP/poly
AM
AM-EXP
AM ∩ coAM
AM[polylog]
AmpMP
AmpPBQP
AP
AP
APP
APP
APX
AUC-SPACE($f(n)$)
AuxPDA
AVBPP
AvE
AvP
AW[P]
AWPP
AW[SAT]
AW[$*$]
AW[t]
βP
BH
BPE
BPEE
BP$_{\mathsf{H}}$SPACE($f(n)$)
BPL
BP · NP
BPP
BPP^{cc}
BPP^{KT}
BPP-OBDD
BPP$_{path}$
BPQP
BPSPACE($f(n)$)
BPTIME($f(n)$)
BQNC
BQNP
BQP
BQP/log
BQP/poly
BQP/qlog
BQP/qpoly
BQP-OBDD
BQP/poly
BQTIME($f(n)$)
k-BWBP
C$_{=}$AC^{0}
C$_{=}$L
C$_{=}$P
CFL
CLOG
CH
Check
C$_{k}$P
CNP
coAM
coC$_{=}$P
cofrIP
Coh
coMA
coMod$_{k}$P
compIP
compNP
coNE
coNEXP
coNL
coNP
coNP^{cc}
coNP/poly
coNQP
coRE
coRNC
coRP
coSL
coUCC
coUP
CP
CSIZE($f(n)$)
CSL
CZK
D#P
DCFL
Δ_{2}P
δ-BPP
δ-RP
DET
DiffAC^{0}
DisNP
DistNP
DP
DQP
DSPACE($f(n)$)
DTIME($f(n)$)
DTISP($t(n), s(n)$)
Dyn-FO
Dyn-ThC^{0}
E
EE
EEE
EESPACE
EEXP
EH
ELEMENTARY
EL$_{k}$P
EPTAS
k-EQBP
EQP
EQTIME($f(n)$)
ESPACE
ExistsBPP
ExistsNISZK
EXP
EXP/poly
EXPSPACE
FBQP
Few
FewP
FH
FNL
FNL/poly
FNP
FO($t(n)$)
FOLL
FP
FP$^{NP[log]}$
FPR
FPRAS
FPT
FPT$_{nu}$
FPT$_{su}$
FPTAS
FQMA
frIP
F-TAPE($f(n)$)
F-TIME($f(n)$)
GA
GAN-SPACE($f(n)$)
GapAC^{0}
GapL
GapP
GC($s(n), \mathcal{C}$)
GCSL
GI

GPCD$(r(n), q(n))$
G$[t]$
HeurBPP
HeurBPTIME$(f(n))$
H$_k$P
HVSZK
IC[log, poly]
IP
IPP
L
LIN
L$_k$P
LOGCFL
LogFew
LogFewNL
LOGNP
LOGSNP
L/poly
LWPP
MA
MA′
MAC^0
MA-E
MA-EXP
mAL
MaxNP
MaxPB
MaxSNP
MaxSNP^0
comNL
MinPB
MIP
MIP$^*[2, 1]$
MIP$_{\mathsf{EXP}}$
(M$_k$)P
mL
mNC^1
mNL
mNP
Mod$_k$L
Mod$_k$P
ModP
ModZ$_k$L
mP
MP
MPC
mP/poly
mTC^0
NAuxPDA^p
NC
NC^0
NC^1
NC^2
NE
NE/poly
NEE
NEEE
NEEXP
NEXP
NEXP/poly
NIQSZK
NISZK
NISZK$_h$
NL
NL/poly
NLIN
NLOG
NP
NPC
NP$^{\mathsf{cc}}$
NP$_{\mathbb{C}}$
NPI
NP ∩ coNP
(NP ∩ coNP)/poly
NP/log
NPMV
NPMV-sel
NPMV$_t$
NPMV$_t sel$
NPO
NPOPB
NP/poly
(NP, P-samplable)
NP$_{\mathbb{R}}$
NPSPACE
NPSV
NPSV-sel
NPSV$_t$
NPSV$_t$-Sel
NQP
NSPACE$(f(n))$
NT
NTIME$(f(n))$
OCQ
OptP
P
P/log
P/poly
P$^{\#\mathsf{P}}$
P$^{\#P[1]}$
PAC^0
PBP
k-PBP
P$_{\mathbb{C}}$
P$^{\mathsf{cc}}$
PCD$(r(n), q(n))$
P-close
PCP$(r(n), q(n))$
PermUP
PEXP
PF
PFCHK$(t(n))$
PH
PH$^{\mathsf{cc}}$
Φ_2P
PhP
$\Pi_2 P$
PINC
PIO
P$^{\mathsf{K}}$
PKC
PL
PL$_1$
PL$_\infty$
PL
PLL
PLS
P$^{\mathsf{NP}}$
P$^{\mathsf{NP}[k]}$
P$^{\mathsf{NP}[\log]}$
P$^{\mathsf{NP}[\log^2]}$
P-OBDD
PODN
polyL
PostBQP
PP
PP/poly
PPA
PPAD
PPADS
PPP
P$^{\mathsf{PP}}$
PPSPACE
PQUERY
PR
P$_{\mathbb{R}}$
Pr$_{\mathsf{H}}$SPACE$(f(n))$
PromiseBPP
PromiseBQP
PromiseP
PromiseRP
PrSPACE$(f(n))$
P-Sel
PSK
PSPACE
PT$_1$
PTAPE
PTAS
PT/WK$(f(n), g(n))$
PZK
QAC^0
QAC$^0[m]$
QACC^0
QAM
QCFL
QCMA
QH
QIP

QIP(2)
QMA
QMA^{+}
QMA(2)
QMA$_{\log}$
QMAM
QMIP
QMIP$_{\text{le}}$
QMIP$_{\text{ne}}$
QNC^{0}
QNC$^{0}_{f}$
QNC^{1}
QP
QPLIN
QPSPACE
QSZK
R
RE
REG
RevSPACE$(f(n))$
R$_{\text{H}}$L
RL
RNC
RP
RPP
RSPACE$(f(n))$
S$_2$P
S$_2$-EXP $\cdot$ P$^{\text{NP}}$
SAC
SAC^{0}
SAC^{1}
SAPTIME
SBP
SC
SEH
SelfNP
SF$_k$
Σ_2P
SKC
SL
SLICEWISEPSPACE
SNP
SO-E
SP
SP
span-P
SPARSE
SPL
SPP
SUBEXP
symP
SZK
SZK$_h$
TALLY
TC^{0}
TFNP
Θ_2P
TreeBQP
TREE-REGULAR
UAP
UCC
UE
UL
UL$/$poly
UP
US
VNC$_k$
VNP$_k$
VP$_k$
VQP$_k$
W$[1]$
WAPP
W[P]
WPP
W[SAT]
W$[*]$
W$[t]$
W$^{*}[t]$
XOR-MIP$^{*}[2,1]*[2,1]$
XP
XP$_{\text{uniform}}$
YACC
ZPE
ZPP
ZPTIME$(f(n))$

4 Complexity Classes

Warning: Please do not feed oracles to the complexity classes! These classes require a specially balanced diet to ensure proper relativization.

0-1-$\mathsf{NP}_{\mathbb{C}}$: Binary Restriction of NP Over The Complex Numbers

- The intersection of $\mathsf{NP}_{\mathbb{C}}$ with $\{0,1\}^*$ (i.e. the set of binary strings).
- Contains NP.
- Is contained in PSPACE, and in AM assuming the Extended Riemann Hypothesis [Koi96].

1$\mathsf{NAuxPDA}^p$: One-Way $\mathsf{NAuxPDA}^p$

- Defined in [Bra77], where it was also shown that 1$\mathsf{NAuxPDA}^p$ strictly contains CFL and is strictly contained in LOGCFL.

#AC^0: Sharp-AC^0

- The class of functions from $\{0,1\}^n$ to nonnegative integers computed by polynomial-size constant-depth arithmetic circuits, using addition and multiplication gates and the constants 0 and 1.
- Contained in GapAC^0.

#L: Sharp-L

- Has the same relation to L as #P does to P.
- #L is contained in DET [AJ93].

#L/poly: Nonuniform #L

- Has the same relation to #L as P/poly does to P.

#P: Sharp-P

- The class of function problems of the form "compute $f(x)$," where f is the number of accepting paths of an NP machine.
- The canonical #P-complete problem is #SAT: given a Boolean formula, compute how many satisfying assignments it has.
- Defined in [Val79b], where it was also shown that the problem of counting the number of perfect matchings in a bipartite graph (or equivalently, computing the permanent of a 0-1 matrix) is #P-complete.
- What makes that interesting is that the associated decision problem (whether a bipartite graph *has* a perfect matching) is in P.

- PH is in $\mathsf{P}^{\#\mathsf{P}}$ [Tod89].
- Any function in #P can be approximated to within a polynomial factor in BPP with an NP oracle [Sto85].

#W[t]: Sharp-W[t]

- Roughly, the analogue of #P for parameterized complexity. I.e. the class of parameterized counting problems that are fixed-parameter parsimonious reducible to the following problem: #WSAT Given a Boolean formula, count the number of satisfying assignments of Hamming weight k (where k is the parameter).
- Defined in [FG02], which should be consulted for the full definition. [FG02] also showed that there exist #W[1]-complete problems whose corresponding decision problems are fixed-parameter tractable (i.e. in FPT).

⊕EXP: Parity EXP

- The exponential-time analogue of ⊕P.
- There exists an oracle relative to which ⊕EXP = ZPP [BBF98]

⊕L: Parity L

- Has the same relation to L as ⊕P does to P.
- Contains SL [KW93].
- Solving a linear system over $\mathbb{Z}_2$ is complete for ⊕L [Dam90].
- $\oplus\mathsf{L}^{\oplus\mathsf{L}} = \oplus\mathsf{L}$ [HRV00].

⊕L/poly: Nonuniform ⊕L

- Has the same relation to ⊕L as P/poly does to P.
- Contains NL/poly [GL96].

⊕P: Parity P

- The class of decision problems solvable by an NP machine such that
  - If the answer is "yes," then the number of accepting paths is odd.
  - If the answer is "no," then the number of accepting paths is even.
- Defined in [PZ83].
- Contains graph isomorphism; indeed, graph isomorphism is low for ⊕P [AK02].
- Contains FewP [CH89].

- There exists an oracle relative to which P = $\oplus$P but P is not equal to NP (and indeed NP = EXP) [BBF98].

$\oplus$SAC^1: Parity SAC^1

- Has the same relation to SAC^1 as $\oplus$P does to P.
- Contains SAC^1 [GW96].

A$_0$PP: One-Sided Analog of AWPP

- Same as SBP, except that f is a GapP rather than #P function.
- Defined in [Vya03], where the following was also shown:
  - A$_0$PP contains QMA, AWPP, and coC$_=$P.
  - A$_0$PP is contained in PP.
  - If A$_0$PP =PP then PH is contained in PP.

AC: Unbounded Fanin Polylogarithmic-Depth Circuits

- AC$_i$ is the class of decision problems solvable by a uniform family of Boolean circuits, with polynomial size, depth $O(\log^i(n))$, and unbounded fanin. The gates allowed are AND, OR, and NOT.
- Then AC is the union of AC$_i$ over all nonnegative i.
- AC$_i$ is contained in NC$_{i+1}$; thus, AC =NC.
- Contains NL.
- For a random oracle A, AC$_i^A$ is strictly contained in AC$_{i+1}^A$, and AC^A is strictly contained in P^A, with probability 1 [Mil92].

AC^0: Unbounded Fanin Constant-Depth Circuits

- An especially important subclass of AC, corresponding to constant-depth, unbounded-fanin, polynomial-size circuits with AND, OR, and NOT gates.
- Computing the parity or majority of n bits is not in AC^0 [FSS84].
- There are functions in AC^0 that are pseudorandom for all statistical tests in AC^0 [NW94]. But there are no functions in AC^0 that are pseudorandom for all statistical tests in QP (quasipolynomial time) [LMN93].
- [LMN93] showed furthermore that functions with AC^0 circuits of depth d are learnable in QP, given their outputs on $O(2^{\log^{O(d)}(n)})$ randomly chosen inputs. On the other hand, this learning algorithm is essentially optimal, unless there is a $2^{n^{o(1)}}$ algorithm for factoring [Kha93].

- Although there are no good pseudorandom *functions* in AC^0, [IN96] showed that there are pseudorandom *generators* that stretch n bits to $n+\Theta(\log n)$, assuming the hardness of a problem based on subset sum.
- AC^0 contains NC^0, and is contained in QAC^0 and MAC^0.
- In descriptive complexity, uniform AC^0 can be characterized as the class of problems expressible by first-order predicates with addition and multiplication operators - or indeed, with ordering and multiplication, or ordering and division (see [Lee02]).
- [BLMS98] showed the following problem is complete for depth-k AC^0 circuits (with a uniformity condition):
  - Given a grid graph of polynomial length and width k, decide whether there is a path between vertices s and t (which can be given as part of the input).

$\mathsf{AC}^0[m]$: AC^0 With MOD_m Gates

- Same as AC^0, but now "MOD_m" gates (for a specific m) are allowed in addition to AND, OR, and NOT gates. (A MOD m gate outputs 0 if the sum of its inputs is congruent to 0 modulo m, and 1 otherwise.)
- If m is a power of a prime p, then for any prime $q \neq p$, deciding whether the sum of n bits is congruent to 0 modulo q is not in $\mathsf{AC}^0[m]$ [Raz87], [Smo87]. It follows that, for any such m, $\mathsf{AC}^0[m]$ is strictly contained in NC^1.
- However, if m is a product of distinct primes (i.e. 6), then it is not even known whether $\mathsf{AC}^0[m] = \mathsf{NP}$!
- See also: $\mathsf{QAC}^0[m]$.

ACC^0: AC^0 With Arbitrary MOD Gates

- Same as $\mathsf{AC}^0[m]$, but now the constant-depth circuit can contain MOD_m gates for *any* m.
- Contained in TC^0.
- Indeed, can be simulated by depth-3 threshold circuits of quasipolynomial size [GK93].
- According to [All96], there is no good evidence for the existence of cryptographically secure functions in ACC^0. On the other hand, no nontrivial lower bounds against ACC^0 are known either. Thus, this class represents the current frontier for circuit lower bounds.
- Contains 4-PBP [BT88].
- See also: QACC^0.

AH: Arithmetic Hierarchy

- The analog of PH in computability theory.
- Let $\Delta_0 = \Sigma_0 = \Pi_0 = \mathsf{R}$. Then for $i > 0$, let
  - $\Delta_i = \mathsf{R}$ with Σ_{i-1} oracle.
  - $\Sigma_i = \mathsf{RE}$ with Σ_{i-1} oracle.
  - $\Pi_i = \mathsf{coRE}$ with Σ_{i-1} oracle.

  Then AH is the union of these classes for all nonnegative constant i.
- Each level of AH strictly contains the levels below it.

AL: Alternating L

- Same as AP, but for logarithmic-space instead of polynomial-time.
- AL = P [CKS81].

AlgP/poly: Polynomial-Size Algebraic Circuits

- The class of multivariate polynomials over the integers that can be evaluated using a polynomial (in the input size n) number of additions, subtractions, and multiplications, together with the constants -1 and 1. The class is nonuniform, in the sense that the polynomial for each input size n can be completely different.
- Named in [Imp02], though it has been considered since the 1970's.
- If BPP (or even BPP is contained in NE), then either NEXP is not in P/poly, or else the permanent polynomial of a matrix is not in AlgP/poly [KI02].

AM: Arthur-Merlin

- The class of decision problems for which a "yes" answer can be verified by an *Arthur-Merlin protocol*, as follows.

  Arthur, a BPP (i.e. probabilistic polynomial-time) verifier, generates a "challenge string" based on the input, and sends it to Merlin, who has unbounded computational resources. Merlin sends back a response, and then Arthur decides whether to accept. Given an algorithm for Arthur, we require that

  1. If the answer is "yes," then Merlin can act in such a way that Arthur accepts with probability at least 2/3 (over the choice of Arthur's random bits).
  2. If the answer is "no," then however Merlin acts, Arthur will reject with probability at least 2/3.

It turns out that, without loss of generality, we can take the protocol to be *public-coin* [GS86]: that is, the "challenge string" is just a sequence of uniform random bits. So, Arthur never needs to hide information from Merlin.

- Furthermore, define $\mathsf{AM}[k]$ similarly to AM, except that Arthur and Merlin have k rounds of interaction. Then for all constant $k > 2$, $\mathsf{AM}[k] = \mathsf{AM}[2] = \mathsf{AM}$ [BM88].
- AM contains graph nonisomorphism.
- Contains NP, BPP, and SZK, and is contained in $\Pi_2\mathsf{P}$ and NP/poly.
- If AM contains coNP then PH collapses to $\Sigma_2\mathsf{P} \cap \Pi_2\mathsf{P}$ [BHZ87].
- There exists an oracle relative to which AM is not contained in PP [Ver92].
- AM = NP under the assumption that some language in NE intersect coNE requires nondeterministic circuits of size $2^{\Omega(n)}$ ([MV99], improving [KvM99]). (A nondeterministic circuit C has two inputs, x and y, and accepts on x if there exists a y such that $C(x, y) = 1$.)

AM-EXP: Exponential-Time AM

- Same as AM, except that Arthur is exponential-time and can exchange exponentially long messages with Merlin.
- Contains MA-EXP, and is contained in EH and indeed $\mathsf{S}_2\text{-EXP} \cdot \mathsf{P}^{\mathsf{NP}}$.
- If coNP is contained in AM[polylog] then EH collapses to AM-EXP [PV04].

$\mathsf{AM} \cap \mathsf{coAM}$:

- The class of decision problems for which both "yes" and "no" answers can be verified by an AM protocol.
- If EXP requires exponential time even for AM protocols, then $\mathsf{AM} \cap \mathsf{coAM} = \mathsf{NP} \cap \mathsf{coNP}$ [GSTS03].
- There exists an oracle relative to which $\mathsf{AM} \cap \mathsf{coAM}$ is not contained in PP [Ver95].

AM[polylog]: AM With Polylog Rounds

- Same as AM, except that we allow $\mathsf{polylog}(n)$ rounds of interaction between Arthur and Merlin instead of a constant number.
- Not much is known about AM[polylog]—for example, whether it sits in PH. However, [SS04] show that if AM[polylog] contains coNP, then EH collapses to $\mathsf{S}_2\text{-EXP} \cdot \mathsf{P}^{\mathsf{NP}}$. ([PV04] improved the collapse to AM-EXP.)

AmpMP: Amplifiable MP

- The class of decision problems such that for some #P function $f(x, 0^m)$,
  1. The answer on input x is "yes" if and only if the middle bit of $f(x)$ is 1.
  2. The m bits of $f(x)$ to the left and right of the middle bit are all 0.
- Defined in [GKR+95].
- Contained in MP.

AmpP-BQP: BQP Restricted To AmpP States

- Similar to TreeBQP except that the quantum computer's state at each time step is restricted to being exponentially close to a state in AmpP (that is, a state for which the amplitudes are computable by a classical polynomial-size circuit).
- Defined in [Aar04c], where it was also observed that AmpP-BQP is contained in the third level of PH, just as TreeBQP is.

AP: Alternating P

- An *alternating Turing machine* is a nondeterministic machine with two kinds of states, AND states and OR states. It accepts if and only if the tree of all computation paths, considered as an AND-OR tree, evaluates to 1. (Here "Accept" corresponds to 1 and "Reject" to 0.)
- Then AP is the class of decision problems solvable in polynomial time by an alternating Turing machine.
- AP = PSPACE [CKS81].

APP: Approximable in Polynomial Time

- The "other" AP.
- The class of real-valued functions from $\{0,1\}^n$ to $[0,1]$ that can be approximated within any $\epsilon > 0$ by a deterministic Turing machine in time polynomial in n and $1/\epsilon$.
- Defined by [KRC00], who also showed that the set of AP machines is in RE.

APP: Approximable in Probabilistic Polynomial Time

- The class of real-valued functions from $\{0,1\}^n$ to $[0,1]$ that can be approximated within any $\epsilon > 0$ by a probabilistic Turing machine in time polynomial in n and $1/\epsilon$.
- Defined by [KRC00], who also show the following:
  - Approximating the acceptance probability of a Boolean circuit is APP-complete. The authors argue that this makes APP a more natural class than BPP, since the latter is not believed to have complete problems.

- If APP = AP, then BPP = P.
- On the other hand, there exists an oracle relative to which BPP = P but APP does not equal AP.

- Interestingly, it is unclear whether the set of APP machines is in RE.

APP: Amplified PP

- The "other" APP.
- Roughly, the class of decision problems for which the following holds. For all polynomials $p(n)$, there exist GapP functions f and g such that for all inputs x with $n = |x|$,
  1. If the answer is "yes" then $f(x)/g(1^n) > 1 - 2^{-p(n)}$.
  2. If the answer is "no" then $f(x)/g(1^n) < 2^{-p(n)}$.
- Defined in [Li93], where the following was also shown:
  - APP is contained in PP, and indeed is low for PP.
  - APP is closed under intersection, union, and complement.
- APP contains AWPP [Fen03].

APX: Approximable

- The subclass of NPO problems that admit constant-factor approximation algorithms. (I.e., there is a polynomial-time algorithm that is guaranteed to find a solution within a constant factor of the optimum cost.)
- Contains PTAS.
- Equals the closure of MaxSNP and of MaxNP under PTAS reduction [KMSV99], [CT94].
- Defined in [ACG+99].

AUC-SPACE($f(n)$): Randomized Alternating $f(n)$-Space

- The class of problems decidable by an $O(f(n))$-space Turing machine with three kinds of quantifiers: existential, universal, and randomized.
- Contains GAN-SPACE($f(n)$).
- AUC-SPACE(poly(n)) = SAPTIME = PSPACE [Pap83].
- [Con92] shows that AUC-SPACE($\log n$) has a natural complete problem, and is contained in NP $\cap$ coNP.

AuxPDA: Auxiliary Pushdown Automata

- Equivalent to NAuxPDA^p without the running-time restriction.
- Equals P [Coo71a].

AVBPP: Average-Case BPP

- Defined in [OW93] to be the class of decision problems that have a good average-case BPP algorithm, whenever the input is chosen from an efficiently samplable distribution.
- Note that this is *not* the same as the BPP version of AvP.

AvE: Average Exponential-Time With Linear Exponent

- Has the same relation to E as AvP does to P.

AvP: Average Polynomial-Time

- A *distributional problem* consists of a decision problem A, and a probability distribution μ over problem instances.
- A function f, from strings to integers, is *polynomial on μ-average* if there exists a constant $\epsilon > 0$ such that the expectation of $f^\epsilon(x)$ is finite, when x is drawn from μ. Then (A, μ) is in AvP if there is an algorithm for A whose running time is polynomial on μ-average.
- This convoluted definition is due to Levin [Lev86], who realized that simpler definitions lead to classes that fail to satisfy basic closure properties. Also see [Gol97] for more information.
- If AvP = DistNP then EXP = NEXP [BDCGL92].
- See also: (NP, P-samplable).

AW[P]: Alternating W[P]

- Same as AW[SAT] but with "circuit" instead of "formula."
- Has the same relation to AW[SAT] as W[P] has to W[SAT].
- Defined in [DF99].

AWPP: Almost WPP

- The class of decision problems solvable by an NP machine such that for some polynomial-time computable (i.e. FP) function f,
  1. If the answer is "no," then the difference between the number of accepting and rejecting paths is at most $2^{-\mathsf{poly}(n)} f(x)$.
  2. If the answer is "yes," then the difference is between $(1 - 2^{-\mathsf{poly}(n)} f(x)$ and $f(x)$.

- Defined in [FFK94].
- Contains BQP [FR98], WAPP [BGM03], LWPP, and WPP.
- Contained in APP [Fen03].

AW[SAT]: Alternating W[SAT]

- Basically has the same relation to W[SAT] as PSPACE does to NP.
- The class of decision problems of the form $(x, r, k_1, \cdots, k_r)$ (with $r, k_1, \cdots, k_r$ parameters), that are fixed-parameter reducible to the following problem, for some constant h:
  - Parameterized QBFSAT: Given a Boolean formula F (with no restriction on depth), over disjoint variable sets $\{S_1, \cdots, S_r\}$. Does there exist an assignment to S_1 of Hamming weight k_1, such that for all assignments to S_2 of Hamming weight k_2, etc. (alternating "there exists" and "for all'), F is satisfied?
- See W[1] for the definition of fixed-parameter reducibility.
- Defined in [DF99].
- Contains AW[$*$], and is contained in AW[P].

AW[$*$]: Alternating W[$*$]

- The union of AW[t] over all t.

AW[t]: Alternating W[t]

- Has the same relation to W[t] as PSPACE does to NP.
- Same as AW[SAT], except that the formula F can have depth at most t.
- Defined in [DF99].
- Contained in AW[$*$].

βP: Limited-Nondeterminism NP

- β_kP is the class of decision problems solvable by a polynomial-time Turing machine that makes $O(\log^k n)$ nondeterministic transitions, with the same acceptance mechanism as NP. Equivalently, the machine receives a purported proof of size $O(\log^k n)$ that the answer is "yes."
- Then βP is the union of β_kP over all constant k.
- Defined in [KF84]. See also the survey [GLM96].

- There exist oracles relative to which basically any consistent inclusion structure among the β_kP's can be realized [BG98].
- β_2P contains LOGNP and LOGSNP.

BH: Boolean Hierarchy Over NP

- The smallest class that contains NP and is closed under union, intersection, and complement.
- The levels are defined as follows:
  - $\mathsf{BH}_1 = \mathsf{NP}$.
  - BH_{2i} is the class of languages that are the intersection of a BH_{2i-1} language with a coNP language.
  - BH_{2i+1} is the class of languages that are the union of a BH_{2i} language with an NP language.

  Then BH is the union over all i of BH_i.
- For more detail see [CGH+88].
- Contained in Δ_2P.
- If BH collapses at any level, then PH collapses to Σ_3P [Kad88], and indeed to a class smaller than $\mathsf{P}^{\mathsf{NP}[\log]}$ [HHH98], [RW01].
- See also: QH.

BPE: Bounded-Error Probabilistic E

- Has the same relation to E as BPP does to P.
- EE = BPE if and only if EXP = BPP [IKW01].

BPEE: Bounded-Error Probabilistic EE

- Has the same relation to EE as BPP does to P.

$\mathsf{BP}_\mathsf{H}\mathsf{SPACE}(f(n))$: Bounded-Error Halting Probabilistic $f(n)$-Space

- The class of decision problems solvable in $O(f(n))$-space with error probability at most 1/3, by a Turing machine that halts on every input *and* every random tape setting.
- Contains $\mathsf{R}_\mathsf{H}\mathsf{SPACE}(f(n))$.
- Is contained in $\mathsf{DSPACE}(f(n)^{3/2})$ [SZ95].

BPL: Bounded-Error Probabilistic L

- Has the same relation to L as BPP does to P. The Turing machine has to halt with probability 1 on every input.
- Contained in SC [Nis92] and in PL.

BP · NP: Probabilistic NP

- Equals AM.

BPP: Bounded-Error Probabilistic Polynomial-Time

- The class of decision problems solvable by an NP machine such that

  1. If the answer is "yes" then at least 2/3 of the computation paths accept.
  2. If the answer is "no" then at most 1/3 of the computation paths accept.

  (Here all computation paths have the same length.)
- Often identified as the class of feasible problems for a computer with access to a genuine random-number source.
- Defined in [Gil77].
- Contained in Σ_2P $\cap$ Π_2P [Lau83], and indeed in ZPP with an NP oracle.
- If BPP contains NP, then RP = NP [Ko82] and PH is contained in BPP [Zac88].
- If any problem in E requires circuits of size $2^{\Omega(n)}$, then BPP = P [IW97].
- Indeed, *any* proof that BPP = P requires showing either that NEXP is not in P/poly, or else that #P requires superpolynomial-size arithmetic circuits [KI02].
- BPP is not known or believed to contain complete problems. [Sip82], [HH86] give oracles relative to which BPP has no complete problems.
- There exist oracles relative to which P = RP but still P is not equal to BPP [BF99].
- In contrast to the case of P, it is unknown whether BPP collapses to BPTIME(n^c) for some fixed constant c. However, [Bar02] and [FS04] have shown hierarchy theorems for BPP with a small amount of advice.
- See also: $\mathsf{BPP}_{\mathsf{path}}$.

BPP^{cc}: Communication Complexity BPP

- The analogue of P^{cc} for bounded-error probabilistic communication complexity.
- Does not equal P^{cc}, and is not contained in NP^{cc}, because of the EQUALITY problem.

- Defined in [BFS86].

$\mathsf{BPP}^{\mathsf{KT}}$: BPP With Time-Bounded Kolmogorov Complexity Oracle

- BPP with an oracle that, given a string x, returns the minimum over all programs P that output x_i on input i, of the length of P plus the maximum time taken by P on any input.
- A similar class was defined in [ABK+02], where it was also shown that in $\mathsf{BPP}^{\mathsf{KT}}$ one can factor, compute discrete logarithms, and more generally invert any one-way function on a non-negligible fraction of inputs.
- See also: P^{K}.

BPP-OBDD: Polynomial-Size Bounded-Error Ordered Binary Decision Diagram

- Same as P-OBDD, except that probabilistic transitions are allowed and the OBDD need only accept with probability at least 2/3.
- Does not contain the integer multiplication problem [AK98].
- Strictly contained in BQP-OBDD [NHK00].

$\mathsf{BPP}_{\mathsf{path}}$: Threshold BPP

- Same as BPP, except that now the computation paths need not all have the same length.
- Defined in [HHT97], where the following was also shown:
  - $\mathsf{BPP}_{\mathsf{path}}$ contains MA and $\mathsf{P}^{\mathsf{NP}[\log]}$, and is contained in PP and BPP with an NP oracle.
  - $\mathsf{BPP}_{\mathsf{path}}$ is closed under complementation, intersection, and union.
  - If $\mathsf{BPP}_{\mathsf{path}} = \mathsf{BPP}_{\mathsf{path}}^{\mathsf{BPP}_{\mathsf{path}}}$, then PH collapses to $\mathsf{BPP}_{\mathsf{path}}$.
  - If $\mathsf{BPP}_{\mathsf{path}}$ contains $\Sigma_2\mathsf{P}$, then PH collapses to BPP to the NP.
- There exists an oracle relative to which $\mathsf{BPP}_{\mathsf{path}}$ is not contained in $\Sigma_2\mathsf{P}$ [BGM03].

BPQP: Bounded-Error Probabilistic QP

- Equals $\mathsf{BPTIME}(2^{O(\log^k n)})$; that is, the class of problems solvable in quasipolynomial-time on a bounded-error machine.
- Defined in [CNS99], where the following was also shown:
  - If either (1) #P does not have a subexponential-time bounded-error algorithm, or (2) EXP does not have subexponential-size circuits, then the BPQP hierarchy is strict—that is, for all $a < b$ at least 1, $\mathsf{BPTIME}(2^{\log^a n})$ is strictly contained in $\mathsf{BPTIME}(2^{\log^b n})$.

BPSPACE($f(n)$): Bounded-Error Probabilistic $f(n)$-Space

- The class of decision problems solvable in $O(f(n))$-space with error probability at most 1/3, by a Turing machine that halts with probability 1 on every input.
- Contains RSPACE($f(n)$) and BP$_{\mathsf{H}}$SPACE($f(n)$).

BPTIME($f(n)$): Bounded-Error Probabilistic $f(n)$-Time

- Same as BPP, but with $f(n)$-time (for some constructible function f) rather than polynomial-time machines.
- Defined in [Gil77].
- BPTIME($n^{\log n}$) does not equal BPTIME(2^{n^ϵ}) for any $\epsilon > 0$ [KV88]. Proving a stronger time hierarchy theorem for BPTIME is a longstanding open problem; see [BH97] for details.
- [Bar02] has shown the following:
  - If we allow a small number of advice bits (say $\log n$), then there is a strict hierarchy: for every d at least 1, BPTIME(n^d)/($\log n$) does not equal BPTIME(n^{d+1})/($\log n$).
  - In the uniform setting, if BPP has complete problems then BPTIME(n^d) does not equal BPTIME(n^{d+1}).
  - BPTIME(n) does not equal NP.
- Subsequently, [FS04] managed to reduce the number of advice bits to only 1: BPTIME(n^d)/1 $\neq$ BPTIME(n^{d+1})/1. They also proved a hierarchy theorem for HeurBPTIME.
- For another bounded-error hierarchy result, see BPQP.

BQNC: Bounded-Error Quantum NC

- The class of decision problems solvable by polylogarithmic-depth quantum circuits with bounded probability of error. (A uniformity condition may also be imposed.)
- Has the same relation to NC as BQP does to P.
- [CW00] showed that factoring is in ZPP with a BQNC oracle.
- Is incomparable with BPP as far as anyone knows.
- See also: RNC.

BQNP: Alternate Name for QMA

BQP: Bounded-Error Quantum Polynomial-Time

- The class of decision problems solvable in polynomial time by a quantum Turing machine, with at most 1/3 probability of error.

- One can equivalently define BQP as the class of decision problems solvable by a uniform family of polynomial-size quantum circuits, with at most 1/3 probability of error [Yao93]. Any universal gate set can be used as a basis; however, a technicality is that the transition amplitudes must be efficiently computable, since otherwise one could use them to encode the solutions to hard problems (see [ADH97]).
- BQP is often identified as the class of feasible problems for quantum computers.
- Contains the factoring and discrete logarithm problems [Sho97], the hidden Legendre symbol problem [vDHI02], the Pell's equation and principal ideal problems [Hal02], and some other problems not thought to be in BPP.
- Defined in [BV97], where it is also shown that BQP contains BPP and is contained in P with a #P oracle.
- $\mathsf{BQP}^{\mathsf{BQP}} = \mathsf{BQP}$ [BV97].
- [ADH97] showed that BQP is contained in PP, and [FR98] showed that BQP is contained in AWPP.
- There exist oracles relative to which:
  - BQP does not equal BPP [BV97].
  - BQP is not contained in MA [Wat00].
  - BQP is not contained in $\mathsf{Mod}_{p^k}\mathsf{P}$ for prime p [dGV02].
  - NP, and indeed $\mathsf{NP} \cap \mathsf{coNP}$, are not contained in BQP [BBBV97].
  - SZK is not contained in BQP [Aar02b].

BQP/log: BQP With Logarithmic-Size Classical Advice

- Same as BQP/poly except that the advice is $O(\log n)$ bits instead of a polynomial number.
- Strictly contained in BQP/qlog [NY03b].

BQP/poly: BQP With Polynomial-Size Classical Advice

- Can also be defined as the class of problems solvable by a nonuniform family of polynomial-size quantum circuits, just as P/poly is the class solvable by a nonuniform family of polynomial-size classical circuits.
- Contains BQP/qlog, and is contained in BQP/qpoly.
- Does not contain ESPACE [NY03b].

BQP/qlog: BQP With Logarithmic-Size Quantum Advice

- Same as BQP/log except that the advice is quantum instead of classical.
- Strictly contains BQP/log [NY03b].
- Contained in BQP/poly.

BQP/qpoly: BQP With Polynomial-Size Quantum Advice

- The class of problems solvable by a BQP machine that receives a quantum state ψ_n as advice, which depends only on the input length n.
- As with all other advice classes in the Zoo, the acceptance probability does *not* need to be bounded away from 1/2 if the machine is given bad advice (thus, we are discussing the class that [NY03b] call BQP/$*$Qpoly).
- Does not contain EESPACE [NY03b].
- [Aar05] shows the following:
  - There exists an oracle relative to which BQP/qpoly does not contain NP.
  - BQP/qpoly is contained in PP/poly.
- An oracle separation between BQP/qpoly and BQP/poly is presently unknown. An unrelativized separation is too much to hope for, since it would imply that PP is not contained in P/poly.
- Contains BQP/poly.

BQP-OBDD: Polynomial-Size Bounded-Error Quantum Ordered Binary Decision Diagram

- Same as P-OBDD, except that unitary (quantum) transitions are allowed and the OBDD need only accept with probability at least 2/3.
- Strictly contains BPP-OBDD [NHK00]. BQTIME($f(n)$): Bounded-Error Quantum $f(n)$-Time
- Same as BQP, but with $f(n)$-time (for some constructible function f) rather than polynomial-time machines.
- Defined in [BV97].

BQP/poly: BQP With Polynomial-Size Advice and Truth-Table Queries

- Same as BQP/poly, except that the machine only gets to make *nonadaptive* queries to whatever oracle it might have.
- Defined in [NY03a], where it was also shown that P is not contained in BQP/poly relative to an oracle.

k-BWBP: Bounded-Width Branching Program

- Alternate name for k-PBP.

$\mathsf{C_{=}AC^0}$: Exact-Counting $\mathsf{AC^0}$

- The class of problems for which there exists a $\mathsf{DiffAC^0}$ function f such that the answer is "yes" on input x if and only if $f(x) = 0$.
- Equals $\mathsf{TC^0}$ and $\mathsf{PAC^0}$ under logspace uniformity [ABL98].

$\mathsf{C_{=}L}$: Exact-Counting L

- Has the same relation to L as $\mathsf{C_{=}P}$ does to P.
- $\mathsf{C_{=}L^{C_{=}L}} = \mathsf{L^{C_{=}L}}$ [ABO99].

$\mathsf{C_{=}P}$: Exact-Counting Polynomial-Time

- The class of decision problems solvable by an NP machine such that the number of accepting paths exactly equals the number of rejecting paths, if and only if the answer is "yes."
- Equals coNQP [FGHP98].

CFL: Context-Free Languages

- Does not equal QCFL [MC00].
- Contained in LOGCFL.
- Strictly contains DCFL [Bra77].

CH: Counting Hierarchy

- The union of the $\mathsf{C_kP}$'s over all constant k.
- Contained in PSPACE.
- It is an open problem whether there exists an oracle relative to which CH is infinite, or even unequal to PSPACE. This is closely related to the problem of whether $\mathsf{TC^0} = \mathsf{NC^1}$.

Check: Checkable Languages

- The class of problems such that a program P that allegedly solves them can be *checked* efficiently. That is, f is in Check if there exists a BPP algorithm C such that for all programs P and inputs x,

  1. If $P(y) = f(y)$ for all inputs y, then $C^{P(x)}$ (C with oracle access to P) accepts with probability at least 2/3.

2. If $P(x)$ is not equal to $f(x)$ then $C^{P(x)}$ accepts with probability at most $1/3$.

- Introduced in [BK89], where it was also shown that Check equals frIP $\cap$ cofrIP.
- Check is contained in NEXP $\cap$ coNEXP [FRS88].
- [BG94] show that if NEE is not contained in BPEE then NP is not contained in Check.

$\mathsf{C}_k\mathsf{P}$: k^{th} Level of CH

- Defined as follows:
  - $\mathsf{C}_0\mathsf{P} = \mathsf{P}$
  - $\mathsf{C}_1\mathsf{P} = \mathsf{PP}$
  - $\mathsf{C}_2\mathsf{P} = \mathsf{PP}^{\mathsf{PP}}$
  - In general, $\mathsf{C}_{k+1}\mathsf{P}$ is PP with a $\mathsf{C}_k\mathsf{P}$ oracle

  The union of the $\mathsf{C}_k\mathsf{P}$'s is called the counting hierarchy, CH.
- Defined in [Wag86].
- See [Tor91] or [AW90] for more information.

CLOG: Continuous Logarithmic-Time

- Roughly, the class of continuous problems solvable by an ordinary differential equation (ODE) with convergence time logarithmic in the size of the input. The vector field of the ODE is specified by an NC^1 formula, with n parameters that represent the input. The point to which the ODE converges (assuming it does) is the output.
- Defined in [BHSF02], which should be consulted for more details.
- [BHSF02] show that finding the maximum of n integers is in CLOG. Thus, CLOG is best thought of as the continuous-time analog of NC^1, not of DTIME$(\log n)$.
- Contained in CP.

CNP: Continuous NP

- A nondeterministic analog of CP.
- Defined in [SF98], which should be consulted for the definition (it has something to do with strange attractors, I think).
- The authors raise the question of whether CP equals CNP.

coAM: Complement of AM
coC$_=$P: Complement of C$_=$P

- Equals NQP [FGHP98].

cofrIP: Complement of frIP
Coh: Coherent Languages

- The class of problems L that are *efficiently autoreducible*, in the sense that given an input x and access to an oracle for L, a BPP machine can compute $L(x)$ by querying L only on points that differ from x.
- Defined in [Yao90].
- [BG94] show that, assuming NEE is not contained in BPEE, Coh $\cap$ NP is not contained in any of compNP, Check, or frIP.

coMA: Complement of MA
coMod_kP: Complement of Mod_kP
compIP: Competitive IP Proof System

- Same as compNP but for interactive (IP) proofs instead of NP proofs.
- More formally, compIP is the class of decision problems L in IP = PSPACE such that, if the answer is "yes," then that can be proven by an interactive protocol between a BPP verifier and a prover, a BPP machine with access only to an oracle for L.
- Assuming NEE is not contained in BPEE, NP (and indeed NP $\cap$ Coh) is not contained in compIP [BG94].

compNP: Competitive NP Proof System

- The class of decision problems L in NP such that, if the answer is "yes," then a proof can be constructed in polynomial time given access only to an oracle for L.
- Contains NPC.
- [BG94] show that compNP is contained in frIP, and that assuming NEE is not contained in BPEE, compNP does not equal NP.

coNE: Complement of NE
coNEXP: Complement of NEXP

- Contained in NEXP/poly (folklore result reported in Fortnow's weblog: `http://fortnow.com/lance/complog/`).

coNL: Complement of NL

- Equals NL [Imm88] [Sze87].

coNP: Complement of NP

- If NP = coNP, then any inconsistent Boolean formula of size n has a proof of inconsistency of size polynomial in n.
- If NP does not equal coNP, then P does not equal NP. But the other direction is not known.
- See also: NP ∩ coNP.
- Every problem in coNP has an IP (interactive proof) system, where moreover the prover can be restricted to $\mathsf{BPP}^{\#\mathsf{P}}$.

coNP$^{\mathsf{cc}}$: Complement of NP$^{\mathsf{cc}}$
coNP/poly: Complement of NP/poly

- If NP is contained in coNP/poly then PH collapses to $\mathsf{S}_2\mathsf{P}^{\mathsf{NP}}$ [CCHO01].
- $\boxed{\mathsf{NP}^{\mathsf{NP}^{\mathsf{NP}^{(\mathsf{coNP/poly}\cap\mathsf{NP})}}} = \mathsf{NP}^{\mathsf{NP}^{\mathsf{NP}}}}$ [HNOS96]
- *Note:* At the suggestion of Luis Antuñes, the above specimen of the Complexity Zoo has been locked in a cage.

coNQP: Complement of NQP

- Equals $\mathsf{C}_{=}\mathsf{P}$ [FGHP98].

coRE: Complement of RE

- Does not equal RE.
- The problem "given a computable predicate P, is P true of all positive integers?" is coRE-complete.

coRNC: Complement of RNC

- Contains the problem of whether a bipartite graph has a perfect matching [Kar86].

coRP: Complement of RP

- Defined in [Gil77].
- Contains the problem of testing whether an integer is prime [SS77].

coSL: Complement of SL
coUCC: Complement of UCC

- [Tor00] showed the following problem complete for coUCC under L reductions:

  > Given a colored graph G with at most two vertices having any given color, does G have any nontrivial automorphisms?

coUP: Complement of UP
CP: Continuous P

- Same as CLOG, except that the convergence time can be polynomial rather than logarithmic in the input size.
- Defined in [BHSF02] and [SF98].
- Finding a maximum flow, which is P-complete, can be done in CP [BHSF02]. Based on this the authors argue that "P is contained in CP," but this seems hard to formalize, since CP is not a complexity class in the usual sense. They also conjecture that "CP is contained in P" (i.e. the class of ODE's they consider can be integrated efficiently on a standard Turing machine), but this is open.
- Contained in CNP.

CSIZE($f(n)$): Circuit Size f(n)

- The class of decision problems solvable by a (nonuniform) family of Boolean circuits of size $O(f(n))$.
- So for example, CSIZE(poly(n)) (the union of CSIZE(n^k) over all k) equals P/poly.
- Defined in [SM02] among other places.

CSL: Context Sensitive Languages

- The class of languages generated by context-sensitive grammars.
- Equals NSPACE(n) [Kur64].

CZK: Computational Zero-Knowledge

- Same as SZK, except that now the two distributions are merely required to be *computationally indistinguishable* by any BPP algorithm; they don't have to be statistically close. (The "two distributions" are (1) the distribution over Arthur's view of his interaction with Merlin, conditioned on Arthur's random coins, and (2) the distribution over views that Arthur can *simulate* without Merlin's help.)
- Assuming the existence of one-way functions, CZK contains NP [GMW91], and indeed equals IP = PSPACE [BOGG+90].
- On the other hand, if one-way functions do not exist then CZK = AVBPP [OW93].
- Contains PZK and SZK.

D#P: Alternate Name for $\mathrm{P}^{\#\mathrm{P}}$
DCFL: Deterministic CFL

- The class of languages accepted by deterministic pushdown automata.
- Defined in [GG66], where it was also shown that DCFL is strictly contained in CFL and strictly contains REG.

Δ_2P: P With NP Oracle

- A level of PH, the polynomial hierarchy.
- Contains BH.
- There exists an oracle relative to which Δ_2P is not contained in PP [Bei94].
- There exists another oracle relative to which Δ_2P is contained in P/poly, and indeed has linear-size circuits [Wil85].

δ-BPP: δ-Semi-Random BPP

- Same as BPP, except that the random bit source is biased as follows. Each bit could depend on all the previous bits in arbitrarily complicated ways; the only promise is that the bit is 1 with probability in the range $[\delta, 1-\delta]$, conditioned on all previous bits.
- So clearly 0-BPP = P and 1/2-BPP = BPP.
- It turns out that, for any $\delta > 0$, δ-BPP = BPP [VV85], [Zuc91].

δ-RP: δ-Semi-Random RP

- Same as δ-BPP, but for RP instead of BPP.
- For any $\delta > 0$, δ-RP = RP [VV85].

DET: Determinant

- The class of decision problems reducible in L to the problem of computing the determinant of an n-by-n matrix of n-bit integers.
- Defined in [Coo85].
- Contained in NC^2, and contains NL and PL [BCP83].
- Graph isomorphism is hard for DET under L-reductions [Tor00].

DiffAC^0: Difference #AC^0

- The class of functions from $\{0,1\}^n$ to integers expressible as the difference of two #AC^0 functions.
- Equals GapAC^0 under logspace uniformity [ABL98].

DisNP: Disjoint NP Pairs

- The class of pairs $\langle A, B \rangle$, where A and B are NP problems whose sets of "yes" instances are nonempty and disjoint.
- If there exists an optimal propositional proof system, then DisNP has a complete pair [Raz94]. On the other hand, there exists an oracle relative to which DisNP does not have a complete pair [GSSZ03].
- If P does not equal UP, then DisNP contains pairs not separated by any set in P [GS88]. On the other hand, there exists an oracle relative to which P does not equal NP but still DisNP does not contain any P-inseparable pairs [HS92].

DistNP: Distributional NP

- (also called (NP, P-computable) or RNP)
- A *distributional problem* consists of a decision problem A, and a probability distribution μ over problem instances.
- (A, μ) is in DistNP if A is in NP, and μ is P-computable (meaning that its cumulative density function can be evaluated in polynomial time).
- DistNP has complete problems [Gur87], although unlike for NP this is not immediate.
- Any DistNP-complete problem is also complete for (NP, P-samplable) [IL90].

DP: Difference Polynomial-Time

- DP = BH_2, the second level of the Boolean hierarchy.
- Defined in [PY84].

DQP: Dynamical Quantum Polynomial-Time

- The class of decision problems solvable by a BQP machine with oracle access to a *dynamical simulator*. When given a polynomial-size quantum circuit, the simulator returns a sample from the distribution over "classical histories" induced by the circuit. The simulator can adversarially choose any history distribution that satisfies the axioms of "symmetry" and "locality" – so that the DQP algorithm has to work for any distribution satisfying these axioms.
- See [Aar02a] for a full definition.
- There it is also shown that SZK is contained in DQP.
- Contains BQP, and is contained in PP [Aar02a].
- There exists an oracle relative to which DQP does not contain NP [Aar02a].

DSPACE($f(n)$): Deterministic $f(n)$-Space

- The class of decision problems solvable by a Turing machine in space $O(f(n))$.
- The Space Hierarchy Theorem: For constructible $f(n)$ greater than $\log n$, DSPACE($f(n)$) is strictly contained in DSPACE($f(n)\log(f(n))$) [HS65].
- For space constructible $f(n)$, strictly contains DTIME($f(n)$) [HPV77].
- DSPACE(n) does not equal NP (though we have no idea if one contains the other)!
- See also: NSPACE($f(n)$).

DTIME($f(n)$): Deterministic $f(n)$-Time

- The class of decision problems solvable by a Turing machine in time $O(f(n))$.
- The Time Hierarchy Theorem: For constructible $f(n)$ greater than n, DTIME($f(n)$) is strictly contained in DTIME($f(n)\log(f(n))\log\log(f(n))$) [HS65].
- For any space constructible $f(n)$, DTIME($f(n)$) is strictly contained in DSPACE($f(n)$) [HPV77].
- Also, DTIME(n) is strictly contained in NTIME(n) [PPST83] (this result does not work for arbitrary $f(n)$).
- For any constructible superpolynomial $f(n)$, DTIME($f(n)$) with PP oracle is not in P/poly (see [All96]).

DTISP($t(n), s(n)$): Simultaneous $t(n)$-Time and $s(n)$-Space

- The class of decision problems solvable by a Turing machine that uses time $O(t(n))$ and space $O(s(n))$ simultaneously.
- Thus SC = DTISP(poly, polylog) for example.
- Defined in [Nis92], where it was also shown that for all space-constructible $s(n) = \Omega(\log n)$, BPSPACE($s(n)$) is contained in DTISP($2^{O(s(n))}, s^2(n)$).

Dyn-FO: Dynamic FO

- The class of *dynamic* problems solvable using first-order predicates.
- Basically what this means is that an algorithm maintains some polynomial-size data structure (say a graph), and receives a sequence of updates (add this edge, delete that one, etc.). For each update, it computes a new value for the data structure in FO – that is, for each bit of the data structure,there is an FO function representing the new value of that bit, which takes as input both the update and the previous value of the data structure. At the end the algorithm needs to answer some question (i.e. is the graph connected?).

- See [HI02] for more information, and a complete problem for Dyn-FO.
- See also Dyn-ThC^0.

Dyn-ThC^0: Dynamic Threshold Circuits

- Same as Dyn-FO, except that now updates are computed via constant-depth predicates that have "COUNT" available, in addition to AND, OR, and NOT—so it's a uniform version of TC^0 rather than of AC^0.
- See [HI02] for more information.

E: Exponential Time With Linear Exponent

- Equals $\mathsf{DTIME}(2^{O(n)})$.
- Does not equal NP [Boo72] or PSPACE [Boo74].
- There exists a problem that is complete for E under polynomial-time Turing reductions but not polynomial-time truth-table reductions [Wat87].
- Problems hard for BPP under Turing reductions have measure 1 in E [AS94].
- It follows that, if the problems complete for E under Turing reductions do not have measure 1 in E, then BPP does not equal EXP.

IS89 gives an oracle relative to which E = NE but still there is an exponential-time binary predicate whose corresponding *search* problem is not in E.

- Contrast with EXP.

EE: Double-Exponential Time With Linear Exponent

- Equals $\mathsf{DTIME}(2^{2^{O(n)}})$.
- EE = BPE if and only if EXP = BPP [IKW01].
- Contained in EEXP and NEE.

EEE: Triple-Exponential Time With Linear Exponent

- Equals $\mathsf{DTIME}(2^{2^{2^{O(n)}}})$.
- In contrast to the case of EE, it is not known whether EEE = BPEE implies EE = BPE [IKW01].

EESPACE: Double-Exponential Space With Linear Exponent

- Equals $\mathsf{DSPACE}(2^{2^{O(n)}})$.

- Is not contained in BQP/qpoly [NY03b].

EEXP: Double-Exponential Time

- Equals DTIME($2^{2^{p(n)}}$) for $p(n)$ a polynomial.
- Contains EE, and is contained in NEEXP.

EH: Exponential-Time Hierarchy

- Has roughly the same relationship to EXP as PH does to P.
- More formally, EH is defined as the union of E, NE, $\mathsf{NE}^{\mathsf{NP}}$, NE with $\Sigma_2\mathsf{P}$ oracle, and so on.
- See [Har87a] for more information.
- If coNP is contained in AM[polylog], then EH collapses to $\mathsf{S}_2\text{-EXP} \cdot \mathsf{P}^{\mathsf{NP}}$ [SS04] and indeed $\mathsf{AM}_{\mathsf{EXP}}$ [PV04].
- On the other hand, coNE is contained in NE/poly, so perhaps it wouldn't be so surprising if NE collapses.
- Contained in SEH.

ELEMENTARY:

- Equals the union of DTIME(2^n), DTIME(2^{2^n}), DTIME($2^{2^{2^n}}$), and so on.
- Contained in PR.

$\mathsf{EL}_k\mathsf{P}$: Extended Low Hierarchy

- An extension of $\mathsf{L}_k\mathsf{P}$.
- The class of problems A such that $\Sigma_k\mathsf{P}^A$ is contained in $\Sigma_{k-1}\mathsf{P}^{A,\mathsf{NP}}$.
- Defined in [BBS86].

EPTAS: Efficient Polynomial-Time Approximation Scheme

- The class of optimization problems such that, given an instance of length n, we can find a solution within a factor $1+\epsilon$ of the optimum in time $f(\epsilon)p(n)$, where $p(n)$ is a polynomial and f is arbitrary.
- Contains FPTAS and is contained in PTAS.
- Defined in [CT97], where the following was also shown:
  - If $\mathsf{FPT} = \mathsf{XP}_{\mathsf{uniform}}$ then EPTAS = PTAS.

- If EPTAS = PTAS then FPT = W[P].
- If FPT is strictly contained in W[1], then there is a natural problem that is in PTAS but not in EPTAS. (See [CT97] for the statement of the problem, since it's not *that* natural.)

k-EQBP: Width-k Polynomial-Time Exact Quantum Branching Programs

- See k-PBP for the definition of a classical branching program.
- A quantum branching program is the natural quantum generalization: we have a quantum state in a Hilbert space of dimension k. Each step t consists of applying a unitary matrix $U^{(t)}(x_i)$: that is, $U^{(t)}$ depends on a single bit x_i of the input. (So these are the quantum analogues of so-called *oblivious* branching programs.) In the end we measure to decide whether to accept; there must be zero probability of error.
- Defined in [AMP02], where it was also shown that NC^1 is contained in 2-EQBP.
- k-BQBP can be defined similarly.

EQP: Exact Quantum Polynomial-Time

- Same as BQP, except that the quantum algorithm must return the correct answer with probability 1.
- Defined in [BV97], where it was also shown that there exists an oracle relative to which EQP is not in NP.
- There is an oracle relative to which EQP is not in $\mathsf{Mod}_{p^k}\mathsf{P}$ where p is prime [dGV02].
- $\mathsf{P}_{\|}^{\mathsf{NP}[2k]}$ is contained in $\mathsf{EQP}_{\|}^{\mathsf{NP}[k]}$, where $_{\|}^{\mathsf{NP}[k]}$ denotes k nonadaptive oracle queries to NP (queries that cannot depend on the results of previous queries) [BvD99].

EQTIME($f(n)$): Exact Quantum f(n)-Time

- Same as EQP, but with $f(n)$-time (for some constructible function f) rather than polynomial-time machines.
- Defined in [BV97].

ESPACE: Exponential Space With Linear Exponent

- Equals DSPACE($2^{O(n)}$).
- If E = ESPACE then P = BPP [HY84].
- Indeed if E has nonzero measure in ESPACE then P = BPP [Lut91].
- ESPACE is not contained in P/poly [Kan82].

- See also: EXPSPACE.
- Is not contained in BQP/poly [NY03b].

ExistsBPP: BPP With Existential Operator

- The class of problems for which there exists a BPP machine M such that, for all inputs x,
  - If the answer is "yes" then there exists a y such that $M(x, y)$ accepts.
  - If the answer is "no" then for all y, $M(x, y)$ rejects.
- Contains NP and BPP, and is contained in MA and SBP.
- ExistsBPP seems *obviously* equal to MA, yet [FFKL93] constructed an oracle relative to which they're unequal! Here is the difference: if the answer is "yes," MA requires only that there exist a y such that for at least $2/3$ of random strings r, $M(x, y, r)$ accepts (where M is a P predicate). For all other y's, the proportion of r's such that $M(x, y, r)$ accepts can be arbitrary (say, $1/2$). For ExistsBPP, by contrast, the probability that $M(x, y)$ accepts must *always* be either at most $1/3$ or at least $2/3$, for all y's.

ExistsNISZK: NISZK With Existential Operator

- Contains NP and NISZK, and is contained in the third level of PH.

EXP: Exponential Time

- Equals the union of DTIME$(2^{p(n)})$ over all polynomials p.
- Also equals P with E oracle.
- If L = P then PSPACE = EXP.
- If EXP is in P/poly then EXP = MA [BFL91].
- Problems complete for EXP under many-one reductions have measure 0 in EXP [May94a], [JL95].
- There exist oracles relative to which
  - EXP = NP = ZPP [Hel84],
  - EXP = NEXP but still P does not equal NP [Dek76],
  - EXP does not equal PSPACE [Dek76].
- [BT04] show the following rather striking result: let A be many-one complete for EXP, and let S be any set in P of subexponential density. Then $A \setminus S$ is Turing-complete for EXP.

EXP/poly: Exponential Time With Polynomial-Size Advice

- The class of decision problems solvable in EXP with the help of a polynomial-length advice string that depends only on the input length.
- Contains BQP/qpoly [Aar05].

EXPSPACE: Exponential Space

- Equals the union of DSPACE($2^{p(n)}$) over all polynomials $p(n)$.
- See also: ESPACE.
- Given a first-order statement about real numbers, involving only addition and comparison (no multiplication), we can decide in EXPSPACE whether it's true or not [Ber80].

FBQP: Function BQP

- Has the same relation to BQP as FNP does to NP.
- There exists an oracle relative to which PLS is not contained in FBQP [Aar04b].

Few: FewP With Flexible Acceptance Mechanism

- The class of decision problems solvable by an NP machine such that
  1. The number of accepting paths A is bounded by a polynomial in the size of the input x.
  2. For some polynomial-time predicate Q, $Q(x, A)$ is true if and only if the answer is "yes".
- Also called FewPaths.
- Defined in [CH89].
- Contains FewP, and is contained in $\mathsf{P}^{\mathsf{FewP}}$ [Köb89] and in SPP [FFK94].
- See also the survey [Tor90].

FewP: NP With Few Witnesses

- The class of decision problems solvable by an NP machine such that
  1. If the answer is "no," then all computation paths reject.
  2. If the answer is "yes,' then at least one path accepts; furthermore,the number of accepting paths is upper-bounded by a polynomial in n, the size of the input.
- Defined in [AR88].

- Is contained in $\oplus$P [CH89].
- There exists an oracle relative to which P, UP, FewP, and NP are all distinct [Rub88].
- Also, there exists an oracle relative to which FewP does not have a Turing-complete set [HJV93].
- Contained in Few.
- See also the survey [Tor90].

FH: Fourier Hierarchy

- FH_k is the class of problems solvable by a uniform family of polynomial-size quantum circuits, with k levels of Hadamard gates and all other gates preserving the computational basis. (Conditional phase flip gates are fine, for example.) Thus
  - $\mathsf{FH}_0 = \mathsf{P}$
  - $\mathsf{FH}_1 = \mathsf{BPP}$
  - FH_2 contains factoring, because of Kitaev's phase estimation algorithm
- It is an open problem to show that the Fourier hierarchy is infinite relative to an oracle (that is, FH_k is strictly contained in FH_{k+1}).
- Defined in [Shi03].

FNL: Function NL

- Has the same relation to NL as FNP does to NP.
- Defined by [AJ93], who also showed that if NL = UL, then FNL is contained in #L.

FNL/poly: Nonuniform FNL

- Has the same relation to FNL as P/poly does to P.
- Contained in #L/poly [RA00].

FNP: Function NP

- The class of function problems of the following form:

  > Given an input x and a polynomial-time predicate $F(x, y)$, if there exists a y satisfying $F(x, y)$ then output any such y, otherwise output "no."

- FNP generalizes NP, which is defined in terms of decision problems only.
- Actually the word "function" is misleading, since there could be many valid outputs y. That's unavoidable, since given a predicate F there's no "syntactic" criterion ensuring that y is unique.

- FP = FNP if and only if P = NP.
- Contains TFNP.
- A basic question about FNP problems is whether they're *self-reducible*; that is, whether they reduce to the corresponding NP decision problems. Although this is true for all NPC problems, [BG94] shows that if EE does not equal NEE, then there is a problem in NP such that *no* corresponding FNP problem can be reduced to it. [BG94] cites Impagliazzo and Sudan as giving the same conclusion under the assumption that NE does not equal coNE.

FO($t(n)$): First-Order

- The class of decision problems for which a "yes" answer can be expressed by a first-order logic predicate, with a block of restricted quantifiers repeated $t(n)$ times. See [Imm98] for a full definition.
- FO(poly(n)) = P (see [Var82] for example).
- FO(poly(n)) is contained in SO-E.

FOLL: First-Order $\log \log n$

- The class of decision problems solvable by a uniform family of polynomial-size, unbounded-fanin, depth $O(\log \log n)$ circuits with AND, OR, and NOT gates.
- Defined in [BKLM01], where it was also shown that many problems on finite groups are in FOLL.
- Contains AC^0, and is contained in AC^1.
- Is not known to be comparable to L, SL, or NL.

FP: Function Polynomial-Time

- Sometimes defined as the class of functions computable in polynomial time by a Turing machine. (Generalizes P, which is defined in terms of decision problems only.)
- However, if we want to compare FP to FNP, we should instead define it as the class of FNP problems (that is, predicates $P(x, y)$) for which there exists a polynomial-time algorithm that, given x, outputs *any* y such that $P(x, y)$. That is, there could be more than one valid output, even though any given algorithm only returns one of them.
- FP = FNP if and only if P = NP.
- If $\mathsf{FP}^{\mathsf{NP}} = \mathsf{FP}^{\mathsf{NP}[\log]}$ (that is, allowed only a logarithmic number of queries), then P = NP [Kre88]. The corresponding result for P^{NP} versus $\mathsf{P}^{\mathsf{NP}[\log]}$ is not known, and indeed fails relative to some oracles (see [Har87b]).

$\mathsf{FP}^{\mathsf{NP}[\log]}$: FP With Logarithmically Many Queries To NP

- Given a graph, the problem of outputting the size of its maximum clique is complete for $\mathsf{FP}^{\mathsf{NP}[\log]}$.

FPR: Fixed-Parameter Randomized

- Has the same relation to FPT as R does to P.
- Defined in [AR01], where it was shown that, if the Resolution proof system is *automatizable* (that is, if a refutation can always be found in time polynomial in the length of the shortest refutation), then W[P] is contained in FPR.

FPRAS: Fully Polynomial Randomized Approximation Scheme

- The subclass of #P counting problems whose answer, y, is approximable in the following sense. There exists a randomized algorithm that, with probability at least $1 - \delta$, approximates y to within an ϵ multiplicative factor in time polynomial in n (the input size), $1/\epsilon$, and $\log(1/\delta)$.
- The permanent of a nonnegative matrix is in FPRAS [JSV01].

FPT: Fixed-Parameter Tractable

- The class of decision problems of the form $\langle x, k \rangle$, k a parameter, that are solvable in time $f(k)p(|x|)$, where f is arbitrary and $p(n)$ is a polynomial.
- The basic class of the theory of *fixed-parameter tractability*, as described by Downey and Fellows [DF99].
- Contained in $\mathsf{FPT}_{\mathsf{nu}}$, W[1], and FPR.
- Contains FPTAS [CC97], as well as $\mathsf{FPT}_{\mathsf{su}}$.
- Contains PTAS unless FPT = W[1] [Baz95].

$\mathsf{FPT}_{\mathsf{nu}}$: Fixed-Parameter Tractable (nonuniform)

- Same as FPT except that the algorithm can vary with the parameter k (though its running time must always be $O(p(|x|))$, for a fixed polynomial $p(n)$).
- An alternate view is that a single algorithm can take a polynomial-length advice string, depending on k.
- Defined in [DF99] (though they did not use our notation).

$\mathsf{FPT}_{\mathsf{su}}$: Fixed-Parameter Tractable (strongly uniform)

- Same as FPT except that f has to be recursive.

- Defined in [DF99] (though they did not use our notation).

FPTAS: Fully Polynomial-Time Approximation Scheme

- The subclass of NPO problems that admit an approximation scheme in the following sense. For any $\epsilon > 0$, there is an algorithm that is guaranteed to find a solution whose cost is within a $1 + \epsilon$ factor of the optimum cost. Furthermore, the running time of the algorithm is polynomial in n (the size of the problem) and in $1/\epsilon$.
- Contained in PTAS.
- Defined in [ACG+99].
- Contained in FPT [CC97].

FQMA: Function QMA

- The class of problems for which the task is to output a quantum certificate for a QMA problem, when such a certificate exists. Thus, the desired output is a quantum state.
- Defined in [JWB], where it is also shown that state preparation for 3-local Hamiltonians is FQMA-complete. The authors also observe that, in contrast to the case of FNP versus NP, there is no obvious reduction of FQMA problems to QMA problems.

frIP: Function-Restricted IP Proof Systems

- The class of problems L that have a *decider* in the following sense. There exists a BPP machine D such that for all inputs x,
  1. If the answer is "yes" then $D^L(x)$ (D with oracle for L) accepts with probability at least $2/3$.
  2. If the answer is "no" then $D^A(x)$ accepts with probability at most $1/3$ for all oracles A.
- Contains compIP [BG94] and Check [BK89].
- Contained in MIP = NEXP [FRS88].
- Assuming NEE is not contained in BPEE, NP (and indeed NP $\cap$ Coh) is not contained in frIP [BG94].

F-TAPE($f(n)$): Provable DSPACE($f(n)$) For Formal System $\mathcal{F}$

- The class of decision problems that can be *proven* to be solvable in $O(f(n))$ space on a deterministic Turing machine, from the axioms of formal system $\mathcal{F}$.
- Defined in [Har78].

- See also F-TIME($f(n)$). The results about F-TAPE mirror those about F-TIME, but in some cases are sharper.

F-TIME($f(n)$): Provable DTIME($f(n)$) For Formal System $\mathcal{F}$

- The class of decision problems that can be *proven* to be solvable in $O(f(n))$ time on a deterministic Turing machine, from the axioms of formal system $\mathcal{F}$.
- Defined in [Har78], where the following was also shown:
  - If F-TIME($f(n)$) = DTIME($f(n)$), then DTIME($f(n)$) is strictly contained in DTIME($f(n)g(n)$) for any nondecreasing, unbounded, recursive $g(n)$.
  - There exist recursive, monotonically increasing $f(n)$ such that F-TIME($f(n)$) is strictly contained in DTIME($f(n)$).

  See also F-TAPE($f(n)$).

GA: Graph Automorphism

- Can be defined as the class of problems polynomial-time Turing reducible to the problem of deciding whether a given graph has any nontrivial automorphisms.
- Contains P and is contained in GI.
- See [KSTT93] for much more information about GA.

GAN-SPACE($f(n)$): Games Against Nature $f(n)$-Space

- The class of problems decidable by an $O(f(n))$-space Turing machine with two kinds of quantifiers: existential and randomized.
- Contains NSPACE($f(n)$) and BPSPACE($f(n)$), and is contained in AUC-SPACE($f(n)$).
- By linear programming, GAN-SPACE($\log n$) is contained in P.

GapAC^0: Gap #AC^0

- The class of functions from $\{0,1\}^n$ to integers computable by constant-depth, polynomial-size arithmetic circuits with addition and multiplication gates and the constants 0, 1, and -1. (The only difference from #AC^0 is the ability to subtract, using the constant -1.)
- Equals DiffAC^0 under logspace uniformity [ABL98].

GapL: Gap Logarithmic-Space

- Has the same relation to L as GapP does to P.

GapP: Gap Polynomial-Time

- The class of functions $f(x)$ such that for some NP machine, $f(x)$ is the number of accepting paths minus the number of rejecting paths.
- Equivalently, the closure of the #P functions under subtraction.
- Defined in [FFK94] and independently [Gup95].

GC$(s(n), \mathcal{C})$: Guess and Check

- The class of decision problems for which a "yes" answer can be verified by

  1. guessing $s(n)$ bits, then
  2. verifying the answer in complexity class $\mathcal{C}$.

  For example, GC$(p(n), \text{P}) = \text{NP}$ where $p(n)$ is a polynomial.
- Defined in [CC93].
- Umans [Uma98] has shown that given a DNF expression ϕ, the Shortest Implicant problem (is there a conjunction of at most k negated or non-negated literals that implies ϕ?) is GC$(\log^2 n, \text{coNP})$-complete.

GCSL: Growing CSL

- The class of languages generated by context-sensitive grammars in which the right-hand side of each transformation is strictly longer than the left-hand side.
- Defined in [DW86].

GI: Graph Isomorphism

- Can be defined as the class of problems polynomial-time Turing reducible to the problem of deciding whether two graphs are isomorphic.
- Contains GA and is contained in NP and in coAM (indeed in SZK). So in particular, if graph isomorphism is NP-complete (that is, if GI = NP), then PH collapses.
- See [KSTT93] for much more information about GI.

GPCD$(r(n), q(n))$: Generalized Probabilistically Checkable Debate

- Same as PCD$(r(n), q(n))$, except that now the verifier is allowed nonadaptively to query $O(q(n))$ *rounds* of the debate, with no restriction on the number of bits it looks at within each round.
- Defined in [CFLS93], who also showed that PCD$(\log n, q(n)) =$ GPCD$(\log n, q(n))$ for every $q(n)$.

G$[t]$: Stratification of Fixed-Parameter Tractable Problems

- (Basically) the class of decision problems of the form $\langle x, k \rangle$ (k a parameter), that are solvable by a parameterized family of circuits with unbounded fanin and depth t. A uniformity condition may also be imposed.
- Defined in [DF99], which should be consulted for the full definition.
- Uniform $G[\mathsf{P}]$ (i.e. with no restriction on depth) is equal to FPT.

HeurBPP: Heuristic BPP

- The class of problems for which a $1 - 1/\mathsf{poly}(n)$ fraction of instances are solvable by a BPP machine.
- [FS04] showed a strict hierarchy theorem for HeurBPP; thus, HeurBPP does not equal $\mathsf{HeurBPTIME}(n^c)$ for any fixed c.

$\mathsf{HeurBPTIME}(f(n))$: Heuristic $\mathsf{BPTIME}(f(n))$

- The class of problems for which a $1 - 1/\mathsf{poly}(n)$ fraction of instances are solvable by a $\mathsf{BPTIME}(f(n))$ machine.
- Thus HeurBPP is the union of $\mathsf{HeurBPTIME}(n^c)$ over all c.

$\mathsf{H}_k\mathsf{P}$: High Hierarchy In NP

- The class of problems $A \in \mathsf{NP}$ such that $\Sigma_k\mathsf{P}^A = \Sigma_{k+1}\mathsf{P}$; that is, adding A as an oracle increases the power of the k^{th} level of the polynomial hierarchy by a maximum amount.
- For all k, H_k is contained in H_{k+1}.
- Defined in [Sch83].
- See also $\mathsf{L}_k\mathsf{P}$.

HVSZK: Honest-Verifier SZK

- The class of decision problems that have SZK protocols assuming an honest verifier (i.e. one who doesn't try to learn more about the problem by deviating from the protocol).
- Equals SZK [Oka96].

$\mathsf{IC}[\log, \mathsf{poly}]$: Logarithmic Instance Complexity, Polynomial Time

- The class of decision problems such that, for every n-bit string x, there exists a program A of size $O(\log n)$ that, given x as input, "correctly decides" the answer on x in time polynomial in n. This means:
  - There exists a polynomial p such that for any input y, A returns either "yes", "no", or "I don't know" in time $p(|y|)$.

- Whenever A returns "yes" or "no", it is correct.
- A returns either "yes" or "no" on x.

- Defined in [OKSW94]; see also [LV97].
- If NP is contained in IC[log, poly], then P = NP [OKSW94]. Indeed, any self-reducible problem in IC[log, poly] is also in P.
- Strictly contains P/log, and is strictly contained in P/poly.

IP: Interactive Proof

- The class of decision problems for which a "yes" answer can be verified by an *interactive proof*. Here a BPP (i.e. probabilistic polynomial-time) verifier sends messages back and forth with an all-powerful prover. They can have polynomially many rounds of interaction. Given the verifier's algorithm, at the end:
  1. If the answer is "yes," the prover must be able to behave in such a way that the verifier accepts with probability at least 2/3 (over the choice of the verifier's random bits).
  2. If the answer is "no," then however the prover behaves the verifier must reject with probability at least 2/3.
- IP contains PH [LFKN90], and indeed (this was discovered only a few days later) equals PSPACE [Sha90].
- See also: MIP, QIP, MA, AM.

IPP: Unbounded IP

- Same as IP, except that if the answer is "yes," there need only be a prover strategy that causes the verifier to accept with probability greater than 1/2, while if the answer is "no," then for all prover strategies the verifier accepts with probability less than 1/2.
- Defined in [CCG+94], where it was also shown that IPP = PSPACE relative to *all* oracles. Since IP is strictly contained in PSPACE relative to a random oracle, the authors interpreted this as evidence against the Random Oracle Hypothesis (a slight change in definition can cause the behavior of a class relative to a random oracle to change drastically).
- See also: PPSPACE.

L: Logarithmic Space

- The class of decision problems solvable by a Turing machine restricted to use an amount of memory logarithmic in the size of the input, n. (The input itself is not counted as part of the memory.)

- L contains NC^1 [Bor77], and is contained in generalizations including NL, L/poly, SL, RL, $\oplus$L, and $\mathsf{Mod}_k\mathsf{L}$.
- [Rei04] has shown that L = SL. In other words, undirected graph connectivity is solvable in deterministic logarithmic space.

LIN: Linear Time

- The class of decision problems solvable by a deterministic Turing machine in linear time.
- Contained in NLIN.

$\mathsf{L}_k\mathsf{P}$: Low Hierarchy In NP

- The class of problems A such that $\Sigma_k\mathsf{P}^A = \Sigma_k\mathsf{P}$; that is, adding A as an oracle does not increase the power of the k^{th} level of the polynomial hierarchy.
- $\mathsf{L}_1\mathsf{P} = \mathsf{NP} \cap \mathsf{coNP}$.
- For all k, L_k is contained in L_{k+1} and in NP.
- Defined in [Sch83].
- See also $\mathsf{H}_k\mathsf{P}$.

LOGCFL: Logarithmically Reducible to CFL

- The class of decision problems reducible in L to the problem of deciding membership in a context-free language.
- Equivalently, LOGCFL is the class of decision problems solvable by a uniform family of AC^1 circuits, in which no AND gate has fan-in exceeding 2 (see e.g. [Joh90], p. 137).
- LOGCFL is closed under complement [BCD+89].
- Contains NL [Sud78].

LogFew: Logspace-Bounded Few

- The class of decision problems solvable by an NL machine such that
  1. The number of accepting paths on input x is $f(x)$, and
  2. The answer is "yes' if and only if $R(x, f(x)) = 1$, where R is some predicate computable in L.
- Defined in [BDHM92], where it was also shown that LogFew is contained in $\mathsf{Mod}_k\mathsf{L}$ for all $k > 1$.

LogFewNL: Logspace-Bounded FewP

- Same as FewP but for logspace-bounded (i.e. NL) machines.
- Defined in [BDHM92], where it was also shown that LogFewNL is contained in $\mathsf{ModZ}_k\mathsf{L}$ for all $k > 1$.

LOGNP: Logarithmically-Restricted NP

- LOGNP_0 is the class of decision problems expressible in logical form as

  > The set of I for which there exists a subset $S = \{s_1, \cdots, s_{\log n}\}$ of $\{1, \cdots, n\}$ of size $\log n$, such that for all x there exists y such that for all j, a quantifier-free first-order predicate $\varphi(I, s_j, x, y, j)$ holds. (Here x and y are tuples of a constant number of variables.)

  Then LOGNP is the class of decision problems reducible in polynomial-time to a problem in LOGNP_0.
- Defined in [PY96], where it was also shown that the following problem is complete for LOGNP under many-one reductions:

  > *Vapnik-Chernonenkis (V-C) Dimension.* Given a family $\mathcal{F}$ of subsets of a set U, find a subset of $S \subseteq U$ of maximum cardinality, such that every subset of S can be written as the intersection of S with some set in $\mathcal{F}$.
- Contains LOGSNP, and is contained in βP (indeed β_2P).

LOGSNP: Logarithmically-Restricted SNP

- LOGSNP_0 is the class of decision problems expressible in logical form as

  > The set of I for which there exists a subset $S = \{s_1, \cdots, s_{\log n}\}$ of $\{1, \cdots, n\}$ of size $\log n$, such that for all x there exists j such that a quantifier-free first-order predicate $\varphi(I, s_j, x, j)$ holds. (Here x is a tuple of a constant number of variables.)

  Then LOGSNP is the class of decision problems reducible in polynomial-time to a problem in LOGSNP_0.
- Defined in [PY96].
- Contained in LOGNP, as well as QP (actually $\mathsf{DTIME}(n^{\log n})$).
- If P = LOGSNP, then for every constructible $f(n) > n$, $\mathsf{NTIME}(f(n))$ is contained in $\mathsf{DTIME}(g(n)^{\sqrt{(g(n))}})$, where $g(n) = O(f(n) \log f(n))$ [FK97].

L/poly: Nonuniform Logarithmic Space

- Has the same relation to L as P/poly does to P.
- Equals PBP [Cob66].
- Contains SL [AKL+79].

LWPP: Length-Dependent Wide PP

- The class of decision problems solvable by an NP machine such that
  1. If the answer is "no," then the number of accepting computation paths exactly equals the number of rejecting paths.
  2. If the answer is "yes," then these numbers differ by a function $f(|x|)$ computable in polynomial time (i.e. FP). Here $|x|$ is the length of the input x.
- Defined in [FFK94], where it was also shown that LWPP is low for PP and $C_{=}P$. (I.e. adding LWPP as an oracle does not increase the power of these classes.)
- Contained in WPP and AWPP.
- Contains SPP.
- Also, contains the graph isomorphism problem [KSTT92].
- Contains a whole litter of problems for solvable black-box groups: group intersection, group factorization, coset intersection, and double-coset membership [Vin04a]

MA: Merlin-Arthur

- The class of decision problems solvable by a *Merlin-Arthur protocol*, which goes as follows. Merlin, who has unbounded computational resources, sends Arthur a polynomial-size purported proof that the answer to the problem is "yes." Arthur must verify the proof in BPP (i.e. probabilistic polynomial-time), so that
  1. If the answer is "yes," then there exists a proof such that Arthur accepts with probability at least 2/3.
  2. If the answer is "no," then for all proofs Arthur accepts with probability at most 1/3.
- Contains NP and ExistsBPP, and is contained in AM and in QMA.
- Also contained in $\Sigma_2 P \cap \Sigma_2 P$.
- There exists an oracle relative to which BQP is not in MA [Wat00].
- Equals NP under a derandomization assumption.
- See also: MA-E, MA-EXP.

MA′: Sparse MA

- The subclass of MA such that for each input size n, there is a sparse set S_n that Merlin's proof string always belongs to (no matter what the input is).
- Defined in [KSTT93], where it is also observed that if graph isomorphism is in P/poly, then the complement of graph isomorphism is in MA′.

MAC^0: Majority of AC^0

- Same as AC^0, except now we're allowed a single unbounded-fanin majority gate at the root.
- Defined in [JKS02].
- MAC^0 is strictly contained in TC^0 [ABFR94].

MA-E: Exponential-Time MA With Linear Exponent

- Same as MA, except now Arthur is E instead of polynomial-time.
- If MA-E = NEE then MA = NEXP ∩ coNEXP [IKW01].

MA-EXP: Exponential-Time MA

- Same as MA, except now Arthur is EXP instead of polynomial-time, and the message from Merlin can be exponentially long.
- There is a problem in MA-EXP that does not have polynomial-size circuits [BFT98]. On the other hand, there is an oracle relative to which every problem in MA-EXP does have polynomial-size circuits.
- [MVW99] considered the best circuit lower bound obtainable for a problem in MA-EXP, using current techniques. They found that this bound is *half-exponential*: i.e. a function f such that $f(f(n)) = 2^n$. Such functions exist, but are not expressible using standard asymptotic notation.

mAL: Monotone AL

- Defined in [GS90]. Equals mP by definition.

MaxNP: Maximization NP

- Has the same relation to NP as MaxSNP does to SNP.
- Contains MaxPB.
- The closure of MaxNP under PTAS reduction is APX [KMSV99], [CT94].

MaxPB: MaxNP Polynomially Bounded

- The subclass of MaxNP problems for which the cost function is guaranteed always to be bounded by a polynomial.
- MinPB can be defined similarly.
- Defined in [KT94].
- The closure of MaxPB under PTAS reductions equals NPOPB [CVKT99].

MaxSNP: Maximization SNP

- The class of optimization problems reducible by an "L-reduction" to a problem in MaxSNP_0. (*Note:* "L" stands for linear – this is *not* the same as an L reduction! For more details see [PY88].)
- Defined in [PY88], where the following was also shown:
  - Max3SAT is MaxSNP-complete. (Max3SAT is the problem of finding an assignment that maximizes the number of satisfied clauses in a CNF formula with at most 3 literals per clause.)
  - Any problem in MaxSNP can be approximated to within a fixed ratio.
- The closure of MaxSNP under PTAS reduction is APX [KMSV99], [CT94].

MaxSNP_0:

- The class of function problems expressible as "find a relation such that the set of k-tuples for which a given SNP predicate holds has maximum cardinality."
- For example (see [Pap94a]), the Max-Cut problem can be expressed as follows:

  > Given a graph G, find a subset S of vertices that maximizes the number of pairs (u, v) of vertices such that $u \in S$, and $v \notin S$, and G has an edge from u to v.

- Defined in [PY88].

comNL: Complement of mNL

- Defined in [GS90], where it was also shown that comNL does not equal mNL.
- See also: mL.

MinPB: MinNP Polynomially Bounded

- Same as MaxPB but for minimization instead of maximization problems.

MIP: Multi-Prover Interactive Proof

- Same as IP, except that now the verifier can exchange messages with many provers, not just one. The provers cannot communicate with each other during the execution of the protocol, so the verifier can "cross-check" their assertions (as with suspects in separate interrogation rooms).
- Defined in [BOGKW88].
- Let $\mathsf{MIP}[k]$ be the class of decision problems for which a "yes" answer can be verified with k provers. Then for all $k > 2$, $\mathsf{MIP}[k] = \mathsf{MIP}[2] = \mathsf{MIP}$ [BOGKW88].
- MIP equals NEXP [BFL91].

$\mathsf{MIP}^*[2,1]$: 2-Prover, 1-Round MIP With Quantum Provers

- Same as MIP[2], except that now only one round is allowed, and the two provers can share arbitrarily many entangled qubits. The verifier is classical, as are all messages between the provers and verifier.
- Defined in [CHTW04], where evidence was given suggesting that MIP^* does not "obviously" equal NEXP. By contrast, $\mathsf{MIP}[2,1]$, the corresponding class without entanglement, equals NEXP.
- Indeed, the relationship between MIP^* and $\mathsf{MIP} = \mathsf{NEXP}$ is completely unknown—either could contain the other, or they could be incomparable.
- It is also unknown whether increasing the number of provers or rounds changes $\mathsf{MIP}^*[2,1]$.
- Contains $\mathsf{XOR\text{-}MIP}^*[2,1]^*[2,1]$.

$\mathsf{MIP}_{\mathsf{EXP}}$: Exponential-Time Multi-Prover Interactive Proof

- The exponential-time analogue of MIP.
- In the unrelativized world, equals NEEXP.
- There exists an oracle relative to which $\mathsf{MIP}_{\mathsf{EXP}}$ equals the intersection of P/poly, P^{NP}, and $\oplus\mathsf{P}$ [BFT98].

$(M_k)\mathsf{P}$: Acceptance Mechanism by Monoid M_k

- A *monoid* is a set with an associative operation and an identity element (so it's like a group, except that it need not have inverses).
- Then $(M_k)\mathsf{P}$ is the class of decision problems solvable by an NP machine with the following acceptance mechanism. The i^{th} computation path (under some lexicographic ordering) outputs an element m_i of M_k. Then the machine accepts if and only if $m_1 m_2 \cdots m_s$ is the identity (where s is the number of paths).

- Defined by Hertrampf [Her97], who also showed the following (in the special case M is a group):
  - If G is any nonsolvable group (for example S_5, the symmetric group on 5 elements), then $(G)\mathsf{P} = \mathsf{PSPACE}$.
  - $(\mathbb{Z}_k)\mathsf{P} = \mathsf{coMod}_k\mathsf{P}$, where $\mathbb{Z}_k$ is the cyclic group on k elements.
  - If $|G| = k$, then $(G)P$ contains $\mathsf{coMod}_k\mathsf{P}$.

mL: Monotone L

- The class of decision problems solvable by a family of monotone log-width polynomial-size leveled circuits. (A *leveled* circuit is one where gates on each level can depend only on the level immediately below it.)
- Defined in [GS90], who raise as an open problem to define a uniform version of mL.
- Strictly contains mNC^1 [GS91].
- Contained in (nonuniform versions of) mNL and comNL.

mNC^1: Monotone NC^1

- The class of decision problems solvable by a family of monotone NC^1 circuits (i.e. AND and OR gates only).
- A uniformity condition could also be imposed.
- Defined in [GS90].
- Strictly contained in mNL [KW88], and indeed in mL [GS91].
- Strictly contains mTC^0 [Yao89].

mNL: Monotone NL

- See mP for the definition of a monotone nondeterministic Turing machine, due to [GS90].
- mNL is the class of decision problems solvable by a monotone nondeterministic log-space Turing machine.
- mNL does not equal comNL [GS90], in contrast to the case for NL and coNL.
- Also, mNL strictly contains mNC^1 [KW88].
- See also: mL.

mNP: Monotone NP

- The class of decision problems for which a "yes" answer can be verified in mP (that is, monotone polynomial-time). The monotonicity requirement applies only to the input bits, not to the bits that are guessed nondeterministically. So, in the corresponding circuit, one can have NOT gates so long as they depend only on the nondeterministic guess bits.
- Defined in [GS90], where it was also shown that mNP is "trivial": that is, it contains exactly the monotone problems in NP.
- Strictly contains mP [Raz85b].

$\mathsf{Mod}_k\mathsf{L}$: Mod-k L

- Has the same relation to L as $\mathsf{Mod}_k\mathsf{P}$ does to P.
- For any prime k, $\mathsf{Mod}_k\mathsf{L}$ contains SL [KW93].
- For any prime k, $\mathsf{Mod}_k\mathsf{L}^{\mathsf{Mod}_k\mathsf{L}} = \mathsf{Mod}_k\mathsf{L}$ [HRV00].
- For any $k > 1$, contains LogFew [BDHM92].

$\mathsf{Mod}_k\mathsf{P}$: Mod-k Polynomial-Time

- For any $k > 1$: The class of decision problems solvable by an NP machine such that the number of accepting paths is divisible by k, if and only if the answer is "no."
- $\mathsf{Mod}_2\mathsf{P}$ is more commonly known as $\oplus\mathsf{P}$ "parity-P."
- For every k, $\mathsf{Mod}_k\mathsf{P}$ contains graph isomorphism [AK02].
- Equals $\mathsf{coMod}_k\mathsf{P}$ (that is, closed under complement) whenever k is a prime power [BG92].
- Defined in [CH89], [Her90].
- For prime p, there exists an oracle relative to which $\mathsf{Mod}_{p^k}\mathsf{P}$ does not contain EQP [dGV02].

ModP: $\mathsf{Mod}_k\mathsf{P}$ With Arbitrary k

- The class of decision problems solvable by a $\mathsf{Mod}_k\mathsf{P}$ machine where k can varydepending on the input. The only requirement is that 0^k be computable in polynomial time.
- Defined in [KT96], where it was also shown that ModP is contained in AmpMP.

$\mathsf{ModZ}_k\mathsf{L}$: Restricted $\mathsf{Mod}_k\mathsf{L}$

- The class of decision problems solvable by a nondeterministic logspace Turing machine, such that

1. If the answer is "yes," then the number of accepting paths is not congruent to 0 mod k.
2. If the answer is "no," then there are no accepting paths.

Defined in [BDHM92], where it was also shown that $\mathsf{ModZ}_k\mathsf{L}$ contains LogFewNL for all $k > 1$.

- Contained in $\mathsf{Mod}_k\mathsf{L}$ and in NL.

mP: Monotone P

- The definition of this class, due to [GS90], is notobvious. First, a *monotone nondeterministic Turing machine* is one such that, whenever it can make a transition with a 0 on its inputtape, it can also make that same transition with a 1 on its input tape. (This restriction does not apply to the work tape.) A *monotone alternating Turing machine* is subject to the restriction that it can only reference an input bit x by,"there exists a z at most x," or "for all z at least x."

- Then applying the result of [CKS81] that P = AL, mP is defined to be mAL: the class of decision problems solvable by a monotone alternating log-space Turing machine.

- Actually there's a caveat: A monotone Turing machine or circuit can first negate the *entire* input, then perform a monotone computation. That way it becomes meaningful to talk about whether a monotone complexity class is closed under complement.

- Strictly contained in mNP [Raz85b].

- Deciding whether a bipartite graph has a perfect matching, despite being both a monotone problem and in P, requires monotone circuits of superpolynomial size [Raz85a]. Letting MONO be the class of monotone problems,it follows that mP is strictly contained in MONO $\cap$ P.

- See also: mNC^1, mL, mNL,comNL.

MP: Middle-Bit P

- The class of decision problems such that for some #P function f, the answer on input x is "yes' if and only if the middle bit of $f(x)$ is 1.

- Defined in [GKR+95].

- Contains AmpMP.

- MP with ModP oracle equals MP with #P oracle [KT96].

MPC: Monotone Planar Circuits

- The class of decision problems solvable by a family of *monotone stratified planar circuits* (a uniformity condition may also be imposed).
- Such a circuit can contain only AND and OR gates of bounded fan-in. It must be embeddable in the plane with no wires crossing. Furthermore, the input bits can only be accessed at the bottom level, where they are listed in order $(x_1, \cdots, x_n)$.
- Defined in [DC89].
- [BLMS99] showed that we can assume without loss of generality that the circuit has width n and depth n^3.

mP/poly: Monotone P/poly

- The class of decision problems solvable by a nonuniform family of polynomial-size Boolean circuits with only AND and OR gates, no NOT gates. (Or rather, following the definitions of [GS90], the entire input can be negated as long as there are no other negations.)
- More straightforward to define than mP.

mTC^0: Monotone TC^0

- The class of decision problems solvable by a family of monotone TC^0 circuits (i.e.constant-depth, polynomial-size, AND, OR, and threshold gates, but no NOT gates).
- A uniformity condition could also be imposed.
- Defined in [GS90].
- Strictly contained in mNC^1 [Yao89].

$\mathsf{NAuxPDA}^p$: Nondeterministic Auxiliary Pushdown Automata

- The class of problems solvable by nondeterministic logarithmic-space and polynomial-time Turing machines with auxiliary pushdown.
- Equals LOGCFL [Sud78].

NC: Nick's Class

- (Named in honor of Nick Pippenger.)
- NC_i is the class of decision problems solvable by a uniform family of Boolean circuits, with polynomial size, depth $O(\log^i(n))$, and fan-in 2.
- Then NC is the union of NC_i over all nonnegative i.
- Also, NC equals the union of $\mathsf{PT/WK}(\log^k n, n^k)$ over all constants k.

- NC_i is contained in AC_i; thus, $\mathsf{NC} = \mathsf{AC}$.
- Contains NL.
- Generalizations include RNC and BQNC.
- [IN96] construct a candidate pseudorandom generator in NC based on the subset sum problem.
- For a random oracle A, NC_i^A is strictly contained in NC_{i+1}^A, and NC^A is strictly contained in P^A, with probability 1 [Mil92].

NC^0: Level 0 of NC

- By definition, a decision problem in NC^0 can depend on only a constant number of bits of the input. Thus, NC^0 usually refers to *functions* computable by constant-depth, bounded fan-in circuits.
- There is a family of permutations computable by a uniform family of NC^0 circuits that is$\mathsf{P}$-hard to invert [Hås88].
- Recently [AIK04] solved a longstanding open problem by showing that there exist pseudorandom generators and one-way functions in NC^0, based on (for example) the hardness of factoring. Specifically, in these generators every bit of the output depends on only 4 input bits. Whether the dependence can be reduced to 3 bits under the same cryptographic assumptions is open, but [AIK04] have some partial results in this direction. It is known that the dependence cannot be reduced to 2 bits.

NC^1: Level 1 of NC

- See NC for definition.
- [KV94] give a family of functions that is computable in NC^1, but not efficiently learnable unless there exists an efficient algorithm for factoring Blum integers.
- Was shown to equal 5-PBP [Bar89]. On the other hand, width 5 is necessary unless $\mathsf{NC}^1 = \mathsf{ACC}^0$ [BT88].
- As an application of this result, NC^1 can be simulated on a quantum computer with three qubits, one initialized to a pure state and the remaining two in the maximally mixed state [ASV00].Surprisingly, [AMP02] showed that only a *single* qubit is needed to simulate NC^1 - i.e. that NC^1 is contained in 2-EQBP. (Complex amplitudes are needed for this result.)
- Is contained in L [Bor77].
- Contains TC^0.

- NC^1 contains the integer division problem [BCH86], even if an L-uniformity condition is imposed [CDL01].

NC^2: Level 2 of NC

- See NC for definition.
- Contains NL.

NE: Nondeterministic E

- Nondeterministic exponential time with linear exponent (i.e. $\mathsf{NTIME}(2^{O(n)})$).
- $\mathsf{P}^{\mathsf{NE}} = \mathsf{NP}^{\mathsf{NE}}$ [Hem89].
- Contained in NEXP.

NE/poly: Nonuniform NE

- Contains coNE, just as NEXP/poly contains coNEXP.

NEE: Nondeterministic EE

- Nondeterministic double-exponential time with linear exponent (i.e.$\mathsf{NTIME}(2^{2^{O(n)}})$).
- If MA-E = NEE then MA = NEXP $\cap$ coNEXP [IKW01].
- Contained in NEEXP.

NEEE: Nondeterministic EEE

- Nondeterministic triple-exponential time with linear exponent.

NEEXP: Nondeterministic EEXP

- Nondeterministic double-exponential time (i.e. $\mathsf{NTIME}(2^{2^{p(n)}})$ for $p(n)$ a polynomial).
- Equals $\mathsf{MIP}_{\mathsf{EXP}}$.

NEXP: Nondeterministic EXP

- Nondeterministic exponential time (i.e. $\mathsf{NTIME}(2^{p(n)})$ for p a polynomial).
- Equals MIP [BFL91].
- NEXP is in P/poly if and only if NEXP = MA [IKW01].
- [KI02] show the following:
  - If P = RP, then NEXP is not computable by polynomial-size arithmetic circuits.

- If P = BPP and if checking whether a Boolean circuit computes a function that is close to a low-degree polynomial over a finite field is in P, then NEXP is not in P/poly.
- If NEXP is in P/poly, then matrix permanent is NEXP-complete.

Does not equal NP [SFM78].

- Does not equal EXP if and only if there is a sparse set in NP that is not in P.
- There exists an oracle relative to which EXP = NEXP but still P does not equal NP [Dek76].
- The theory of reals with addition (see EXPSPACE) is hard for NEXP [FR74].

NEXP/poly: Nonuniform NEXP

- Contains coNEXP (folklore result reported in Fortnow's weblog: http://fortnow.com/lance/complog/.

NIQSZK: Non-Interactive QSZK

- Has the same relation to QSZK as NISZK does to SZK.
- Defined in [Kob02],where it was also shown that the following promise problem is complete for NIQSZK. Given a quantum circuit, we are promised that the state it prepares (when run on the all-0 state, and tracing out non-output qubits) has trace distance either at most 1/3 or at least 2/3 from the maximally mixed state. The problem is to output "no" in the former case and "yes" in the latter.
- NIQPZK can be defined similarly.

NISZK: Non-Interactive SZK

- Defined in [SCPY98].
- Contained in SZK.
- [GSV99] showed the following:
  - If SZK does not equal BPP then NISZK does not equal BPP.
  - NISZK equals SZK if and only if NISZK is closed under complement.
  - NISZK has natural complete promise problems:
    * *Statistical Distance from Uniform (SDU)*: Given a circuit, consider the distribution over outputs when the circuit is given a uniformly random n-bit string. We're promised that the trace distance between this distribution and the uniform distribution is either at most 1/3 or at least 2/3. The problem is to output "yes" in the former case and "no" in the latter.

* *Entropy Approximation (EA)*: Now we're promised that the entropy of the circuit's output distribution is either at least $k+1$ or at most $k-1$. The problem is to output "yes" in the former case and "no" in the latter.

- NIPZK can be defined similarly.

NISZK$_h$: NISZK With Limited Help

- The non-interactive analogue of SZK$_h$.
- Defined in [BOG03], where the following was also shown:
  - NISZK$_h$ contains NISZK and is contained in SZK.
  - Graph Isomorphism is in NISZK$_h$.
  - The following problem is complete for NISZK$_h$:

    Given two functions from $\{0,1\}^n$ to $\{0,1\}^n$ (specified by circuits), decide whether their ranges are almost equal or almost disjoint, given that one of these is the case.
  - The quantum lower bound for the set comparison problem in [Aar02b] implies an oracle relative to which NISZK$_h$ is not in BQP.

NL: Nondeterministic Logarithmic-Space

- Has the same relation to L as NP does to P.
- In a breakthrough result, was shown to equal coNL [Imm88] [Sze87]. (Though contrast to mNL.)
- Is contained in LOGCFL [Sud78], as well as NC^2.
- Is contained in UL/poly [RA00].
- Deciding whether a bipartite graph has a perfect matching is hard for NL [KUW86].

NL/poly: Nonuniform NL

- Has the same relation to NL as P/poly does to P.
- Is contained in $\oplus$L/poly [GL96], as well as SAC^1.
- Equals UL/poly [RA00].

NLOG: NL With Nondeterministic Oracle Tape

- Same as NL – but if there's an oracle, then NLOG can make queries nondeterministically on a polynomial-size, one-way oracle tape. (NL, by contrast, can use nondeterministic transitions only on the work tape; oracle queries have to be deterministic.)

- See [LL76] or [HCKM88] for more information.
- Although NLOG is contained in P, there exists an oracle relative to which that is not the case. This illustrates that care is needed when defining oracle access mechanisms.

NLIN: Nondeterministic LIN

- Has the same relation to LIN as NP does to P.

NP: Nondeterministic Polynomial-Time

- The class of dashed hopes and idle dreams.
- More formally: an "NP machine" is a nondeterministic polynomial-time Turing machine.
- Then NP is the class of decision problems solvable by an NP machine such that
  1. If the answer is "yes," at least one computation path accepts.
  2. If the answer is "no," all computation paths reject.
- Equivalently, NP is the class of decision problems such that,if the answer is "yes," then there is a proof of this fact, of length polynomial in the size of the input, that can be verified in P (i.e. by a deterministic polynomial-time algorithm). On the other hand,if the answer is "no," then the algorithm must declare invalid any purported proof that the answer is "yes."
- For example, the SAT problem is to decide whether a given Boolean formula has any satisfying truth assignments. SAT is in NP, since a "yes" answer can be proved by just exhibiting a satisfying assignment.
- A decision problem is NP-complete if (1) it is in NP, and (2) any problem in NP can be reduced to it (under some notion of reduction). The class of NP-complete problems is sometimes called NPC.
- That NP-complete problems exist is immediate from the definition. The seminal result of Cook [Coo71b], Karp [Kar72], and Levin [Lev73] is that many *natural* problems (that have nothing to do with Turing machines) are NP-complete.
- The first such problem to be shown NP-complete was SAT [Coo71b]. Other classic NP-complete problems include:
  - 3-Colorability: Given a graph, can each vertex be colored red, green, or blue so that no two neighboring vertices have the same color?
  - Hamiltonian Cycle: Given a graph, is there a cycle that visits each vertex exactly once?

  - Traveling Salesperson: Given a set of n cities, and the distance between each pair of cities, is there a route that visits each city exactly once before returning to the starting city, and has length at most T?
  - Maximum Clique: Given a graph,are there k vertices all of which are neighbors of each other?
  - Subset Sum: Given a collection of integers, is there a subset of the integers that sums to exactly X?

- For many, many more NP-complete problems, see [GJ79].

- NP contains P. I've discovered a marvelous proof that NP and P are unequal, but this web page is too small to contain it. Too bad, since otherwise I'd be eligible for $1,000,000 [Ins00].

- There exists an oracle relative to which P and NP are unequal [BGS75]. Indeed, P and NP are unequal relative to a random oracle with probability 1 [BG81] (see [AFvM01] for a novel take on this result). Though randomoracle results are not always indicative about the unrelativized case [CCG+94].

- There even exists an oracle relative to which the P versus NP problem is outside the usual axiomsof set theory [HH76].

- If we restrict to *monotone* classes, mP is strictly contained in mNP [Raz85b].

- Perhaps the most important insight anyone has had into P versus NP is to be found in [RR97]. There the authors show that no "natural proof" can separate P from NP (or more precisely, place NP outside of P/poly), unless secure pseudorandom generators do not exist. A proof is "natural" if it satisfies two conditions called *constructivity* and *largeness*; essentially all lower bound techniques known to date satisfy these conditions. To obtain unnatural proof techniques, some people suspect we need to relate P versus NP to heavy-duty "traditional" mathematics, for instance algebraic geometry. See [MS02] (and the survey article [Reg02]) for a development of this point of view.

- For more on P versus NP (circa 1992) see [Sip92]. For an opinion poll, see [Gas02].

- If P equals NP, then NP equals its complement coNP. Whether NP equals coNP is also open. NP and coNP can be extended to the polynomial hierarchy PH.

- The set of decision problems in NP, but not in P or NPC, is sometimes called NPI. If P does not equal NP then NPI is nonempty [Lad75].

- Probabilistic generalizations of NP include MA and AM. If NP is in coAM (or BPP) then PH collapses to Σ_2P [BHZ87].

- PH also collapses to Σ_2P if NP is in P/poly [KL82].

- There exist oracles relative to which NP is not in BQP [BBBV97].
- An alternate characterization is $\mathsf{NP} = \mathsf{PCP}(\log n, O(1))$ [ALM+98].
- Also, [Fag74] gave a logical characterization of NP,which leads to the subclass SNP.

NPC: NP-Complete

- The class of decision problems such that (1) they're in NP and (2) every problem in NP is reducible to them (under some notion of reduction). In other words, the hardest problems in NP.
- Two notions of reduction from problem A to problem B are usually considered:
  1. *Karp* or *many-one* reductions. Here a polynomial-time algorithm is given as input an instance of problem A, and must produce as output an instance of problem B.
  2. *Turing* reductions. Here the algorithm for problem B can make arbitrarily many calls to an oracle for problem A.

  Some examples of NP-complete problems are discussed under the entry for NP.
- The classic reference on NPC is [GJ79].
- Unless P = NP, NPC does not containany sparse problems: that is, problems such that the number of "yes" instances of size n is upper-bounded by a polynomial in n [Mah82].
- A famous conjecture [BH77]asserts that all NP-complete problems are polynomial-time isomorphic—i.e. between any two problems, there is a one-to-one and onto Karp reduction. If that's true, the NP-complete problems could be interpreted as mere "relabelings" of one another.
- NP-complete problems are *p-superterse* unless P = NP [BKS95]. This means that, given k Boolean formulas $F_1, \cdots, F_k$, if you can rule out even *one* of the 2^k possibilities in polynomial time (e.g., "if $F_1, \cdots, F_{k-1}$ are all unsatisfiable then F_k is satisfiable"), then P = NP.

$\mathsf{NP}_{\mathbb{C}}$: NP Over The Complex Numbers

- An analog of NP for Turing machines over a complex number field.
- Defined in [BCSS98].
- It is unknown whether $\mathsf{P}_{\mathbb{C}} = \mathsf{NP}_{\mathbb{C}}$, nor are implications known among this question, $\mathsf{P}_{\mathbb{R}}$ versus $\mathsf{NP}_{\mathbb{R}}$, and P versus NP.
- However, [CKK+95] show that if P/poly does not equal NP/poly then $\mathsf{P}_{\mathbb{C}}$ does not equal $\mathsf{NP}_{\mathbb{C}}$.

- [BCSS98] show the following striking result. For a positive integer n, let $t(n)$ denote the minimum number of additions, subtractions, and multiplications needed to construct n, starting from 1. If for every sequence $\{n_k\}$ of positive integers, $t(n_k k!)$ grows faster than polylogarithmically in k, then $\mathsf{P}_{\mathbb{C}}$ does not equal $\mathsf{NP}_{\mathbb{C}}$.
- See also VNP_k.

$\mathsf{NP}^{\mathsf{cc}}$: Communication Complexity NP

- The analogue of P^{cc} for nondeterministic communication complexity. Both communication bits and nondeterministic guess bits count toward the complexity.
- Does not equal P^{cc} or $\mathsf{coNP}^{\mathsf{cc}}$ because of the EQUALITY problem. Also, does not contain $\mathsf{BPP}^{\mathsf{cc}}$ because of that problem.
- Defined in [BFS86].
- Contained in $\mathsf{PH}^{\mathsf{cc}}$.

NPI: NP-Intermediate

- Sometimes used to denote the set of decision problems in NP that are neither NP-complete (that is, in NPC) nor in P.
- Is thought to contain (for example) decision versions of factoring and graph isomorphism.
- Is nonempty if P does not equal NP [Lad75]. Indeed, under this assumption, it contains an infinite number of distinct polynomial-time equivalence classes.

$\mathsf{NP} \cap \mathsf{coNP}$:

- The class of problems in both NP and coNP.
- Contains factoring [Pra75].
- Contains graph isomorphism under the assumption that some language in $\mathsf{NE} \cap \mathsf{coNE}$ requires nondeterministic circuits of size $2^{\Omega(n)}$ ([MV99], improving [KvM99]). (A nondeterministic circuit C has two inputs, x and y, and accepts on x if there exists a y such that $C(x, y) = 1$.)
- Is not believed to contain complete problems.

$(\mathsf{NP} \cap \mathsf{coNP})/\mathsf{poly}$: Nonuniform $\mathsf{NP} \cap \mathsf{coNP}$

- Has the same relation to $\mathsf{NP} \cap \mathsf{coNP}$ as P/poly does to P.
- If NP is contained in $(\mathsf{NP} \cap \mathsf{coNP})/\mathsf{poly}$, then PH collapses to $\mathsf{S}_2\mathsf{P}^{\mathsf{NP} \cap \mathsf{coNP}}$ [CCHO01].

NP/log: NP With Logarithmic Advice

- Same as NP/poly, except that now the advice string is logarithmic-size.

NPMV: NP Multiple Value

- The class of all (possibly partial, possibly multi valued) functions computed by an NP machine as follows: ignore the rejecting paths, and consider any output of an accepting path to be "one of the outputs."
- Contains NPSV and NPMV_t.
- Defined in [BLS84].
- Contrast with FNP.

NPMV-sel: NPMV Selective

- Has the same relation to NPMV as P-Sel does to P.
- Defined in [HHN+95].

NPMV_t: NPMV Total

- The class of all (possibly multivalued) NPMV functions that are total (that is, defined for every input).

NPMV_t-Sel: NPMV_t Selective

- Has the same relation to NPMV_t as P-Sel does to P.
- Defined in [HHN+95].

NPO: NP Optimization

- The class of function problems of the form, "Find any n-bit string x that maximizes a cost function C(x), where C is computable in FP (i.e. polynomial-time)."
- Defined in [ACG+99].
- Contains APX and NPOPB.

NPOPB: NPO Polynomially Bounded

- The subclass of NPO problems for which the cost function is guaranteed always to be bounded by a polynomial in n (the input size).
- See [ACG+99].
- NPOPB equals the closure of MaxPB under PTAS reductions [CVKT99].

NP/poly: Nonuniform NP

- Has the same relation to NP as P/poly does to P.
- Contains AM. On the other hand, if NP/poly contains coNP then PH collapses to the third level.
- NP/poly-natural proofs cannot show that circuit families are outside P/poly, under a pseudorandomness assumption [Rud97].

(NP, P-samplable): Average NP With Samplable Distributions

- See AvP for basic notions of average-case complexity.
- (NP, P-samplable) is the same as DistNP, except that the distribution μ only needs to be*samplable* in polynomial time. μ's cumulative density function does not need to be computable in polynomial time.
- Any problem complete for DistNP is also complete for (NP, P-samplable) [IL90].

$\mathsf{NP}_{\mathbb{R}}$: NP Over The Reals

- An analog of NP for Turing machines over a real number field.
- Defined in [BCSS98].
- It is unknown whether $\mathsf{P}_{\mathbb{R}} = \mathsf{NP}_{\mathbb{R}}$, nor are implications known among this question, $\mathsf{P}_{\mathbb{C}}$ versus $\mathsf{NP}_{\mathbb{C}}$, and P versusNP.
- Also, in contrast to the case of $\mathsf{NP}_{\mathbb{C}}$, it is an open problem to show that P/poly distinct from NP/poly implies $\mathsf{P}_{\mathbb{R}}$ distinct from $\mathsf{NP}_{\mathbb{R}}$. The difference is that in the real case, a comparison (or greater-than) operator is available, and it is not known how much power this yields in comparison to the complex case.
- See also VNP_k.

NPSPACE: Nondeterministic PSPACE

- Equals PSPACE [Sav70].
- On the other hand, this result does not relativize if we allow strings of unbounded length to be written to the oracle tape. In particular, there exists an oracle relative to which NPSPACE is not contained in EXP [GTWB91].

NPSV: NP Single Value

- The class of NPMV functions that are single-valued (i.e., such that every accepting path outputs the same value).

- Defined in [BLS84].
- Contains NPSV$_t$.
- P = NP if and only if FP = NPSV.

NPSV-sel: NPSV Selective

- Has the same relation to NPSV as P-Sel does to P.
- Defined in [HHN+95].

NPSV$_t$: NPSV Total

- The class of all NPSV functions that are total (that is, defined on every input).
- Contained in NPMV$_t$.

NPSV$_t$-Sel: NPSV$_t$ Selective

- Has the same relation to NPSV$_t$ as P-Sel does to P.
- Also known as NP-sel.
- Defined in [HHN+95].

NQP: Nondeterministic Quantum Polynomial-Time

- The class of decision problems solvable by a BQP machine such that a particular `|Accept>` state has nonzero amplitude at the end of the computation, if and only if the answer is "yes."
- Defined in [ADH97].
- Turns out to equal coC$_=$P [FGHP98].
- Contrast with QMA.

NSPACE($f(n)$): Nondeterministic $f(n)$-Space

- Same as NPSPACE, but with $f(n)$-space (for some constructible function f) rather thanpolynomial-space machines.
- Contained in DSPACE($f(n)^2$) [Sav70], and indeed RevSPACE($f(n)^2$) [CP95].
- NPSPACE(n^k) is strictly contained in NPSPACE($n^{k+\epsilon}$) for $\epsilon > 0$ [Iba72] (actually the hierarchy theorem is stronger than this, but pretty technical to state).

NT: Near-Testable

- The class of decision problems that, on input x, are solvable in polynomial time given the answer on input $x-1$ (that is, the lexicographic predecessor of x).
- Defined by [GJY87] (*why* they defined it is a separate question).

NTIME($f(n)$): Nondeterministic $f(n)$-Time

- Same as NP, but with $f(n)$-time (for some constructible function f) rather than polynomial-time machines.
- The Nondeterministic Time Hierarchy Theorem: If f and g are time-constructible and $f(n+1) = o(g)$, then NTIME($f(n)$) does not equal NTIME($g(n)$) [SFM78] (this is actually stronger than the hierarchy theorem for DTIME).
- NTIME(n) strictly contains DTIME(n) [PPST83] (this result does not work for arbitrary $f(n)$).
- For any constructible superpolynomial f, NTIME($f(n)$) with NP oracle is not in P/poly [Kan82].

OCQ: One Clean Qubit

- The class of problems solvable by a BQP machine in which a single qubit is initialized to the "0' state, and the remaining qubits are initialized to the maximally mixed state. (This definition is not known to be robust, so one also needs to specify a gate set.)
- We also need to stipulate that there are no "strong measurements" – intermediate measurements on which later operations are conditioned – since otherwise we can do all of BQP by first initializing the computer to the all-0 state. Parker and Plenio [PP00] failed to appreciate this point.
- Defined by [ASV00] (though they didn't use the name OCQ), who also showed that if OCQ = BQP, something other than gate-by-gate simulation will be needed to show this.

OptP: Optimum Polynomial-Time

- The class of functions computable by taking the maximum of the output values over all accepting paths of an NP machine.
- Defined in [Kre88].
- Contrast with FNP.

P: Polynomial-Time

- The class that started it all.

- The class of decision problems solvable in polynomial time by a Turing machine. (See also FP, for function problems.)
- Defined in [Edm65], [Cob64], [Rab60], and other seminal early papers.
- Contains some highly nontrivial problems, including linear programming [Kha79] and finding a maximum matching in a general graph [Edm65].
- Contains the problem of testing whether an integer is prime, *assuming* the generalized Riemann hypothesis [Mil76].
- Since the Zoo went up, a proof that primality testing is in P with no assumptions was announced [AKS02]!
- A decision problem is P-complete if it is in P, and if every problem in P can be reduced to it in L (logarithmic space). The canonical P-complete problem is *circuit evaluation*: given a Boolean circuit and an input, decide what the circuit outputs when given the input.
- Important subclasses of P include L, NL, NC, andSC.
- P is contained in NP, but whether they're equal seemed to be an open problem when I last checked.
- Efforts to generalize P resulted in BPP and BQP.
- The nonuniform version is P/poly, the monotone version is mP, and versions over the real and complex number fields are $\mathsf{P}_{\mathbb{R}}$ and $\mathsf{P}_{\mathbb{C}}$ respectively.

P/log: P With Logarithmic Advice

- Same as P/poly, except that the advice string for input size n can have length at most logarithmic in n, rather than polynomial.
- Strictly contained in IC[log, poly].
- If NP is contained in P/log then P = NP.

P/poly: Nonuniform Polynomial-Time

- The class of decision problems solvable by a family of polynomial-size Boolean circuits. The family can be *nonuniform*; that is, there could be a completely different circuit for each input length.
- Equivalently, P/poly is the class of decision problems solvable by a polynomial-time Turing machine that receives an "advice string,' that depends only on the size n of the input, and that itself has size upper-bounded by a polynomial in n.

- P/poly contains BPP [KL82].
- [KL82] showed that, if P/poly containsNP, then PH collapses to the second level, Σ_2P.
- They also showed:
  - If PSPACE is in P/poly then PSPACE equals $\Sigma_2 P \cap \Sigma_2 P$.
  - If EXP is in P/poly then EXP = Σ_2P.
- It was later shown that, if NP is contained in P/poly, then PH collapses to ZPP^{NP} [KW98] and indeed S_2P [Cai01]. This seems close to optimal, since there exists an oracle relative to which the collapse cannot be improved to Δ_2P [Wil85].
- If NP is not contained in P/poly, then P does not equal NP.Much of the effort toward separating P from NP is based on this observation. However, a "natural proof' as defined by [RR97] cannot be used to show NP is outside P/poly, if there is any pseudorandom generator in P/poly that has hardness $2^{\Omega(n^\epsilon)}$ for some $\epsilon > 0$.
- If NP is contained in P/poly, then MA = AM [AKSS95]
- The monotone version of P/poly is mP/poly.
- P/poly has measure 0 in E with Σ_2P oracle [May94b].
- Strictly contains IC[log, poly] and P/ log.

$P^{\#P}$: P With #P Oracle

- I decided this class is so important that it deserves an entry of its own, apart from #P.
- Contains PH [Tod89], and is contained in PSPACE.
- Equals P^{PP} (exercise for the visitor).

$P^{\#P[1]}$: P With Single Query To #P Oracle

- Contains PH [Tod89].

PAC^0: Probabilistic AC^0

- The Political Action Committee for computational complexity research.
- The class of problems for which there exists a $DiffAC^0$ function f such that the answer is "yes" on input x if and only if $f(x) > 0$.
- Equals TC^0 and $C_=AC^0$ under logspace uniformity [ABL98].

PBP: Polynomial-Size Branching Program

- Same as k-PBP but with no width restriction.

- Equals L/poly [Cob66].
- Contains P-OBDD.

k-PBP: Polynomial-Size Width-k Branching Program

- A *branching program* is a directed acyclic graph with a designatedstart vertex. Each (non-sink) vertex is labeled by an input bit to query,and has two outgoing edges, one of which is followed if the input bit is 0, the other if the bit is 1. A sink vertex can be either an "accept" or a "reject" vertex.
- The *size* of the branching program is the number of vertices. The branching program has *width* k if the vertices can be sorted into levels, each with at most k vertices,such that each edge goes from a level to the one immediately after it.
- Then k-PBP is the class of decision problems solvable by a family of polynomial-size, width-k branching programs. (A uniformity condition may also be imposed.)
- k-PBP equals (nonuniform) NC^1 for constant k at least 5 [Bar89]. On the other hand, 4-PBP is in ACC^0 [BT88].
- Contained in k-EQBP, as well as PBP.

$\mathsf{P}_{\mathbb{C}}$: Polynomial-Time Over The Complex Numbers

- An analog of P for Turing machines over a complex number field.
- Defined in [BCSS98].
- See also $\mathsf{P}_{\mathbb{R}}$, $\mathsf{NP}_{\mathbb{C}}$, $\mathsf{NP}_{\mathbb{R}}$, VP_k.

P^{cc}: Communication Complexity P

- In a two-party communication complexity problem, Alice and Bob have n-bit strings x and y respectively, and they wish to evaluate some Boolean function $f(x, y)$ using as few bits of communication as possible. P^{cc} is the class of (infinite families of) f's, such that the amount of communication needed is only $O(\mathsf{polylog}(n))$, even if Alice and Bob are restricted to a deterministic protocol.
- Is strictly contained in $\mathsf{NP}^{\mathsf{cc}}$ and in $\mathsf{BPP}^{\mathsf{cc}}$ because of the EQUALITY problem.
- Equals $\mathsf{NP}^{\mathsf{cc}} \cap \mathsf{coNP}^{\mathsf{cc}}$.
- Defined in [BFS86].

$\mathsf{PCD}(r(n), q(n))$: Probabilistically Checkable Debate

- The class of decision problems decidable by a *probabilistically checkable debate system*, as follows.

- Two debaters B and C alternate writing strings on a "debate tape," with B arguing that the answer is "yes" and C arguing the answer is "no." Then a polynomial-time verifier flips $O(r(n))$ random coins and makes $O(q(n))$ *nonadaptive* queries to the debate tape (meaning that they depend only on the input and the random coins, not the results of previous queries). The verifier then outputs an answer, which should be correct with high probability.
- Defined in [CFLS93], who also showed that $\mathsf{PCD}(\log n, 1) = \mathsf{PSPACE}$. This result was used to show that certain problems are PSPACE-hard even to approximate.
- Contained in $\mathsf{GPCD}(r(n), q(n))$.

P-close: Problems Close to P

- The class of decision problems solvable by a polynomial-time algorithm that outputs the wrong answer on only a sparse (that is, polynomially-bounded) set of instances.

$\mathsf{PCP}(r(n), q(n))$: Probabilistically Checkable Proof

- The class of decision problems such that a "yes" answer can be verified by a *probabilistically checkable proof*, as follows.
- The verifier is a polynomial-time Turing machine with access to $O(r(n))$ uniformly random bits. It has random access to a *proof* (which might be exponentially long), but can query only $O(q(n))$ bits of the proof.
- Then we require the following:
  1. If the answer is "yes," there exists a proof such that the verifier accepts with certainty.
  2. If the answer is "no," then for all proofs the verifier rejects with probability at least $1/2$ (over the choice of the $O(r(n))$ random bits).
- Defined in [AS98].
- By definition $\mathsf{NP} = \mathsf{PCP}(0, \mathsf{poly}(n))$.
- $\mathsf{MIP} = \mathsf{PCP}(\mathsf{poly}(n), \mathsf{poly}(n))$.
- $\mathsf{PCP}(r(n), q(n))$ is contained in $\mathsf{NTIME}(2^{O(r(n))}q(n) + \mathsf{poly}(n))$.
- $\mathsf{NP} = PCP(\log n, \log n)$ [AS98].
- In fact, $\mathsf{NP} = PCP(\log n, 1)$ [ALM+98]!
- On the other hand, if NP is contained in $\mathsf{PCP}(o(\log n), o(\log n))$, then $\mathsf{P} = \mathsf{NP}$ [FGL+91].

- Also, even though there exists an oracle relative to which NP = EXP [Hel84], if we could show there exists an oracle relative to which $\mathsf{PCP}(\log n, 1) = \mathsf{EXP}$, then we'd have proved P not equal to NP [For94].
- Another weird oracle fact: since NP does not equal NEXP [SFM78], $\mathsf{PCP}(0, \log n)$ does not equal $\mathsf{PCP}(0, \mathsf{poly}(n))$. However, there exist oracles relative to which the latter inequality is false [HCC+92].

PermUP: Self-Permuting UP

- The class of languages L in UP such that the mapping from an input x to the unique witness for x is a permutation of L.
- Contains P.
- Defined in [HT03], where it was also shown that the closure of PermUP under polynomial-time one-to-one reductions is UP.
- On the other hand, they show that if PermUP = UP then E = UE.
- See also: SelfNP.

PEXP: Probabilistic Exponential-Time

- Has the same relation to EXP as PP does to P.
- Is not contained in P/poly [BFT98].

PF: Alternate Name for FP
PFCHK($t(n)$): Proof-Checker

- The class of decision problems solvable in time $O(t(n))$ by a nondeterministic Turing machine, as follows. The machine is given oracle access to a *proof string* of unbounded length.
  - If the answer is "yes," then there exists a value of the proof string such that all computation paths accept.
  - If the answer is "no," then for all values of the proof string, there exists a computation path that rejects.
- Credited in [For94] to S. Arora, R. Impagliazzo, and U. Vazirani.
- An interesting question is whether NP = $\mathsf{PFCHK}(\log n)$ relative to all possible oracles. Fortnow [For94] observes that the answer depends on what oracle access mechanism is used.

PH: Polynomial-Time Hierarchy

- Let $\Delta_0\mathsf{P} = \Sigma_0 P = \Pi_0\mathsf{P} = \mathsf{P}$. Then for $i > 0$, let
  - $\Delta_i\mathsf{P} = \mathsf{P}$ with $\Sigma_{i-1}\mathsf{P}$ oracle.
  - $\Sigma_i\mathsf{P} = \mathsf{NP}$ with $\Sigma_{i-1}\mathsf{P}$ oracle.
  - $\Pi_i\mathsf{P} = \mathsf{coNP}$ with $\Sigma_{i-1}\mathsf{P}$ oracle.

  Then PH is the union of these classes for all nonnegative constant i.
- Defined in [Sto76].
- Contained in P with a PP oracle [Tod89].
- Contains BPP [Lau83].
- Relative to a random oracle, PH is strictly contained in PSPACE with probability 1 [Cai86].
- Furthermore, there exist oracles separating any $\Sigma_i\mathsf{P}$ from $\Sigma_{i+1}\mathsf{P}$. On the other hand, it is unknown whether $\Sigma_i\mathsf{P}$ is strictly contained in $\Sigma_{i+1}\mathsf{P}$ relative to a *random* oracle with probability 1 (see [Hås87]). Book [Boo94] shows that if PH collapses relative to arandom oracle with probability 1, then it collapses unrelativized.

$\mathsf{PH}^{\mathsf{cc}}$: Communication Complexity PH

- The obvious generalization of $\mathsf{NP}^{\mathsf{cc}}$ and $\mathsf{coNP}^{\mathsf{cc}}$ to a nondeterministic hierarchy.
- It is unknown whether Σ_2^{cc} equals Π_2^{cc}.
- Defined in [BFS86], where it was also shown (among other things) that $\mathsf{BPP}^{\mathsf{cc}}$ is contained in $\Sigma_2^{\mathsf{cc}} \cap \Pi_2^{\mathsf{cc}}$.

$\Phi_2\mathsf{P}$: Second Level of the Symmetric Hierarchy, Alternative Definition

- The class of problems for which there exists a polynomial-time predicate $P(x, y, z)$ such that for all x, if the answer on input x is "yes," then
  1. For all y, there exists a z for which $P(x, y, z)$.
  2. For all z, there exists a y for which $P(x, y, z)$.

  Contained in $\Sigma_2\mathsf{P}$ and $\Pi_2\mathsf{P}$.
- Defined in [Can96], where it was also observed that $\Phi_2\mathsf{P} = \mathsf{S}_2\mathsf{P}$.

PhP: Physical Polynomial-Time

- Defined by Valiant [Val03] to be "the class of physically constructible polynomial resource computers" (characterizing what "can be computed in the physical world in practice"). There he says that PhP contains P and BPP, but that it is open whether PhP contains BQP, since no scalable quantum computing proposal has been demonstrated beyond reasonable doubt.

- For whatever it's worth, the present zookeeper has more qualms about admitting DTIME(n^{1000}) into PhP than BQTIME(n^2). According to the "holographic principle," the number of bits available to any one computation is at most 10^{123} (roughly the inverse of the cosmological constant)—and hence there are "classical polynomial-time algorithms" that can never be executed for fundamental physical reasons. (The key issue, of course, is what counts as a "fundamental physical reason," since every model of computation idealizes some aspects of the world. For example, in reality there aren't infinitely many possible inputs, again because of the holographic bound.)

Σ_2P: coNP With NP Oracle

- Complement of Σ_2P.

- Along with Σ_2P, comprises the second level of PH, the polynomial hierarchy. For any fixed k, there is a problem in $\Pi_2\mathsf{P} \cap \Sigma_2\mathsf{P}$ that cannot be solved by circuits of size n^k [Kan82].

PINC: Polynomial Ignorance of Names of Classes

- (By which I mean, I have no idea what PINC stands for—tell me if you know!)

- The class of function problems, $f : \{0,1\}^n \rightarrow \{0,1\}^m$, such that the k^{th} output bit is computable in time polynomial in n and k.

- Defined in [JY88].

- Contained in PIO. This containment is strict, since if $m = 2^n$ (say), then computing the first bit of $f(x)$ might be EXP-complete.

PIO: Polynomial Input Output

- The class of function problems, $f : \{0,1\}^n \rightarrow \{0,1\}^m$, such that $f(x)$ is computable in time polynomial in n and m. Allows us to discuss whether a function is "efficiently computable" or not, even if the output is too long to write down in polynomial time.

- Defined in [Yan81].

- Strictly contains PINC.

P^K: P With Kolmogorov-Complexity Oracle

- P equipped with an oracle that, given a string x, returns the length of the shortest program that outputs x.
- A similar class was defined in [ABK+02], where it was also shown that P^{K} contains PSPACE. It is not known whether P^{K} contains all of R, or even any recursive problem not in PSPACE.
- See also: $\mathsf{BPP}^{\mathsf{KT}}$.

PKC: Perfect Knowledge Complexity

- Has the same relation to PZK as SKC does to SZK.
- Defined in [GP91].

PL: Probabilistic L

- Has the same relation to L that PP has to P.
- Contains BPL.
- $\mathsf{PL}^{\mathsf{PL}} = \mathsf{PL}$ (see [HO02]).

PL_1: Polynomially-Bounded L_1 Spectral Norm

- The class of Boolean functions $f : \{-1,1\}^n \rightarrow \{-1,1\}$ such that the sum of absolute values of Fourier coefficients of f is bounded by a polynomial in n.
- Defined in [BS90], where it was also shown that PL_1 is contained in PT_1 (and this inclusion is strict).

PL_∞: Polynomially-Bounded L_∞^{-1} Spectral Norm

- The class of Boolean functions $f : \{-1,1\}^n \rightarrow \{-1,1\}$ such that the maximum of $|\alpha|^{-1}$,over all Fourier coefficients α of f, is upper-bounded by a polynomial in n.
- Defined in [BS90], where it was also shown that PL_∞ contains PT_1 (and this inclusion is strict).

PL: Polynomial Leaf

- Defined in [Pap90].
- I believe it's the same as PPA.

PLL: Polynomial Local Lemma

- The class of TFNP function problems that are guaranteed to have a solution because of theLovsz Local Lemma. Defined in [Pap94b].

PLS: Polynomial Local Search

- The subclass of TFNP function problems that are guaranteedto have a solution because of the lemma that "every finite directed acyclicgraph has a sink."
- More precisely, for each input, there's a finite set of *solutions* (i.e. strings), and a polynomial-time algorithm that computes a *cost* for each solution, and a *neighboring solution* of lower cost provided that one exists. Then the problem is to return any solution that has costless than or equal to all of its neighbors. (In other words, a local optimum.)
- (*Note:* In the Zookeeper's humble opinion, PLS *should* have been defined as follows: there exist polynomial-time algorithms that compute the cost of a solution, and the set of *all* neighbors of a given solution, not just a single solution of lower cost. Of course we'd require that every solution has only polynomially many neighbors. The two definitions are not obviously equivalent, and it's conceivable that knowing all the neighbors would be helpful—for example, in simulated annealing one sometimes makes uphill moves.)
- Defined in [JPY88], [PY88].
- There exists an oracle relative to which PLS is not contained in FBQP [Aar04b].
- Also, there exist oracles relative to which PLS is not contained in PPA [BOM04], and PPA and PPP are not contained in PLS [Mor01].
- Whether PLS is not in PPP relative to some oracle remains open.

P^{NP}: P With Oracle Access To NP

- See $\Delta_2\mathsf{P}$.

$\mathsf{P}^{\mathsf{NP}}[k]$: P With k NP Queries(for constant k)

- Equals P with 2^{k-1} parallel queries to NP (i.e. queries that do not depend on the outcomes of previous queries) ([BH91] and [Hem89] independently).
- If $\mathsf{P}^{\mathsf{NP}[1]} = \mathsf{P}^{\mathsf{NP}[2]}$, then $\mathsf{P}^{\mathsf{NP}[1]} = \mathsf{P}^{\mathsf{NP}[\log]}$ and indeed PH collapses to $\Delta_3\mathsf{P}$ (attributed in [Har87b] to J. Kadin).

$\mathsf{P}^{\mathsf{NP}[\log]}$: P With Log NP Queries

- The class of decision problems solvable by a P machine, that can make $O(\log n)$ queries to an NP oracle (where n is the length of the input).
- Equals $\mathsf{P}_{\|}^{\mathsf{NP}}$, the class of decision problems solvable by a P machine that can make polynomially many *nonadaptive* queries to an NP oracle (i.e. queries that do not depend on the outcomes of previous queries) ([BH91] and [Hem89] independently).

- $\mathsf{P}^{\mathsf{NP}[\log]}$ is contained in PP [BHW89].
- Determining the winner in an election system proposed in 1876 by Charles Dodgson (a.k.a. Lewis Carroll) has been shown to be complete for $\mathsf{P}^{\mathsf{NP}[\log]}$ [HHR97].
- Contains $\mathsf{P}^{\mathsf{NP}[k]}$ for all constants k.

$\mathsf{P}^{\mathsf{NP}[\log^2]}$: P With $\log^2$ NP Queries

- Same as $\mathsf{P}^{\mathsf{NP}[\log]}$, except that now $\log^2$ queries can be made.
- The model-checking problem for a certain temporal logic is $\mathsf{P}^{\mathsf{NP}[\log^2]}$-complete [Sch03].
- For all k, P with $\log^k$ adaptive queries to NP coincides with P with $\log^{k+1}$ nonadaptive queries [CS92].

P-OBDD: Polynomial-Size Ordered Binary Decision Diagram

- An *ordered binary decision diagram (OBDD)* is a branching program (see k-PBP), with the additional constraint that if x_i is queried before x_j on any path, then $i < j$.
- Then P-OBDD is the class of decision problems solvable by polynomial-size OBDD's.
- Contained in PBP, as well as BPP-OBDD.

PODN: Polynomial Odd Degree Node

- The subclass of TFNP function problems that are guaranteed to have a solution because of the lemma that "every finite graph has an even number of odd-degree nodes."
- Equals PPA [Pap90].

polyL: Polylogarithmic Space

- Equals DSPACE($\log^c n$).
- In contrast to L, which is contained in P, it is not known if polyL is contained in P or vice versa. On the other hand, we do know that polyL does not equal P, since (for example) polyL does not have complete problems under many-to-one logspace reductions.

PostBQP: BQP With Postselection

- The class of decision problems solvable by a BQP machine such that
  - If the answer is "yes" then the second qubit has at least 2/3 probability of being measured 1, *conditioned* on the first qubit having been measured 1.
  - If the answer is "no" then the second qubit has at most 1/3 probability of being measured 1, conditioned on the first qubit having been measured 1.
  - On any input, the first qubit has a nonzero probability of being measured 1.

- Defined in [Aar04a], where it is also shown that PostBQP equals PP.
- [Aar04a] also gives the following alternate characterizations of PostBQP (and therefore of PP):
  - The quantum analogue of $\mathsf{BPP}_{\mathsf{path}}$.
  - The class of problems solvable in quantum polynomial time if we allow arbitrary linear operations (not just unitary ones). Before measuring, we divide all amplitudes by a normalizing factor to make the probabilities sum to 1.
  - The class of problems solvable in quantum polynomial time if we take the probability of measuring a basis state with amplitude α to be not $|\alpha|^2$ but $|\alpha|^p$, where p is an even integer greater than 2. (Again we need to divide all amplitudes by a normalizing factor to make the probabilities sum to 1.)

PP: Probabilistic Polynomial-Time

- The class of decision problems solvable by an NP machine such that
  1. If the answer is "yes" then at least 1/2 of computation paths accept.
  2. If the answer is "no" then less than 1/2 of computation paths accept.

  Defined in [Gil77].
- PP is closed under union and intersection [BRS91](this was an open problem for 14 years).
- Contains $\mathsf{P}^{\mathsf{NP}[\log]}$ [BHW89].
- Equals $\mathsf{PP}^{\mathsf{BPP}}$ [KSTT89b] as well as PostBQP [Aar04a].
- However, there exists an oracle relative to which PP does not contain $\Delta_2\mathsf{P}$ [Bei94].
- PH is in P^{PP} [Tod89].
- BQP is low for PP; i.e. $\mathsf{PP}^{\mathsf{BQP}} = \mathsf{PP}$ [FR98].
- For a random oracle A, PP^A is strictly contained in PSPACE^A with probability 1 [ABFR94].
- For any fixed k, there exists a language in PP that does not have circuits of size n^k [Vin04b].
- PP can be generalized to the counting hierarchy CH.

PP/poly: Nonuniform PP

- Contains BQP/qpoly [Aar05].

- If PP/poly = P/poly then PP is contained in P/poly. Indeed this is true with any syntactically defined class in place of PP. An implication is that any unrelativized separation of BQP/qpoly from BQP/poly would imply that PP does not have polynomial-size circuits.

PPA: Polynomial Parity Argument

- Defined in [Pap94b]; see also [BCE+95].
- The subclass of TFNP function problems that are guaranteed to have a solution because of the lemma that "all graphs of maximum degree 2 have an even number of leaves."
- More precisely, there's a polynomial-time algorithm that, given any string, computes its "neighbor" strings (of which there are at most two). Then given a leaf string (i.e. one with only one neighbor), the problem is to output another leaf string.
- As an example, suppose you're given a cubic graph (one where every vertex has degree 3), and a Hamiltonian cycle H on that graph. Then by making a sequence of modifications to H (albeit possibly exponentially many), it is always possible to find a second Hamilton cycle (see [Pap94a]). So this problem is in PPA.
- Another problem in PPA is finding an Arrow-Debreu equilibrium, giventhe goods and utility functions of traders in a marketplace.
- Contained in TFNP.
- Contains PPAD.
- There exist oracles relative to which PPA does not contain PLS [BOM04] and PPP [BCE+95]. There also exists an oracle relative to which PPA is not contained in PPP [BCE+95].

PPAD: Polynomial Parity Argument (Directed)

- Defined in [Pap94b]; see also [BCE+95].
- Same as PPA, except now the graph is directed, and we're asked to find either a source or a sink.
- Contained in PPA and PPADS.
- NASH, the problem of finding a Nash equilibrium in a game of two or more players with specified utilities, is in PPAD [Pap94b].
- There exists an oracle relative to which PPP is not contained in PPAD [BCE+95].

PPADS: Polynomial Parity Argument (Directed, Sink)

- Defined in [Pap94b]; see also [BCE+95].
- Same as PPA, except now the graph is directed, and we're asked to find a sink.
- Contained in PPP.
- Contains PPAD.

PPP: Polynomial Pigeonhole Principle

- Defined in [Pap94b]; see also [BCE+95].
- The subclass of TFNP function problems that are guaranteed to have a solution because of the Pigeonhole Principle.
- More precisely, we're given a Boolean circuit, that maps n-bit strings to n-bit strings. The problem is to return *either* an input that maps to 0^n, *or* two inputs that map to the same output.
- Contained in TFNP.
- Contains PPADS.
- [BCE+95] give oracles relative to which PPP is not contained in PPA and PPAD, and PPA is not contained in PPP.
- [Mor01] gives an oracle relative to which PPP is not contained in PLS.
- Whether PLS is not contained in PPP relative to some oracle remains open.

P^{PP}: P With PP Oracle

- A level of the counting hierarchy CH.
- It is not known whether there exists an oracle relative to which P^{PP} does not equal PSPACE.
- Contains $\mathsf{PP}^{\mathsf{PH}}$ [Tod89].
- Equals $\mathsf{P}^{\#\mathsf{P}}$ (exercise for the visitor).

PQUERY: Polynomial Queries

- The class of decision problems solvable in polynomial space using at most a polynomial number of queries to the oracle.
- Thus, PQUERY = PSPACE, but PQUERY^A does not equal PSPACE^A for some oracles A.

- Defined in [Kur83], where it was actually put forward as a serious argument (!!) against believing relativization results.

PPSPACE: Probabilistic PSPACE

- Same as IPP, except that IPP uses private coins while PPSPACE uses public coins.
- Can also be defined as a probabilistic version of PSPACE.
- Equals PSPACE.
- Defined in [Pap83].

PR: Primitive Recursive Functions

- Basically, the class of functions definable by recursively building up arithmetic functions: addition, multiplication, exponentiation, tetration, etc. What's *not* allowed is to "diagonalize" a whole series of such functions to produce an even faster-growing one. Thus, the Ackermann function was proposed in 1928 as an example of a recursive function that's not primitive recursive, showing that PR is strictly contained in R.
- An interesting difference is that PR functions can be explicitly enumerated, whereas functions in R cannot be (since otherwise the halting problem would be decidable). That is, PR is a "syntactic" class whereas R is "semantic."
- On the other hand, we can "enumerate" any RE set by a PR function in the following sense: given an input (M, k), where M is a Turing machine and k is an integer, if M halts within k steps then output M; otherwise output nothing. Then the union of the outputs, over all possible inputs (M, k), is exactly the set of M that halt.
- PR strictly contains ELEMENTARY.

$\mathsf{P}_{\mathbb{R}}$: Polynomial-Time Over The Reals

- An analog of P for Turing machines over a real number field.
- Defined in [BCSS98].
- See also $\mathsf{P}_{\mathbb{C}}$, $\mathsf{NP}_{\mathbb{C}}$, $\mathsf{NP}_{\mathbb{R}}$,VP_k.

$\mathsf{Pr_HSPACE}(f(n))$: Unbounded-Error Halting Probabilistic $f(n)$-Space

- Has the same relation to DSPACE$(f(n))$ as PP does to P. The Turing machine has to halt on every input *and* every setting of the random tape.
- Equals PrSPACE$(f(n))$ [Jun85].

PromiseBPP: Promise-Problem BPP

- Same as PromiseRP, but for BPP instead of RP.
- Defined in [BF99].

PromiseBQP: Promise-Problem BQP

- Same as PromiseBQP, but for BQP instead of BPP.
- If PromiseBQP = PromiseP then BQP/poly = P/poly.

PromiseP: Promise-Problem P

- The class of promise problems solvable by a P machine.

PromiseRP: Promise-Problem RP

- The class of promise problems solvable by an RP machine. I.e., the machine must accept with probability at least 1/2 for "yes" inputs, and with probability 0 for "no" inputs, but could have acceptance probability between 0 and 1/2 for inputs that do not satisfy the promise.
- Defined in [BF99], where it was also shown that BPP is in $\mathsf{RP}^{\mathsf{PromiseRP}[1]}$ (i.e. with a single oracle query to PromiseRP).
- Contained in PromiseBPP.

PrSPACE($f(n)$): Unbounded-Error Probabilistic $f(n)$-Space

- Has the same relation to DSPACE($f(n)$) as PP does to P. The Turing machine has to halt with probability 1 on every input.
- Contained in DSPACE($f(n)^2$) [BCP83].
- Equals $\mathsf{Pr_H SPACE}(f(n))$ [Jun85].

P-Sel: P-Selective Sets

- The class of decision problems for which there's a polynomial-time algorithm with the following property. Whenever it's given two instances, a "yes"and a "no" instance, the algorithm can always decide which is the "yes" instance.
- Defined in [Sel79], where it was also shown that if NP is contained in P-Sel then P = NP.
- There exist P-selective sets that are not recursive (i.e. not in R).

PSK: Polynomial Sink

- Yeah, I'm told that's what the S and K stand for. Go figure.

- The class of total function problems definable as follows: given a directed graph of indegree and outdegree at most 1, and given a source, find a sink.
- Defined in [Pap90].
- Equals PPADS.

PSPACE: Polynomial-Space

- The class of decision problems solvable by a Turing machine in polynomial space.
- Equals NPSPACE [Sav70], AP [CKS81], IP [Sha90], and, assuming the existence of one-way functions,CZK [BOGG+90].
- Contains P with #P oracle.
- A canonical PSPACE-complete problem is *Quantified Boolean Formula (QBF)*: Given a Boolean formula withuniversal and existential quantifiers, decide whether it's true or false.
- Relative to a random oracle, PSPACE strictly contains PH with probability 1 [Cai86].
- PSPACE has a complete problem that is both downward self-reducible and random self-reducible [TV02]. It is the largest class with such a complete problem.
- Contained in EXP. There exists an oracle relative to which this containment is proper [Dek76].

PT_1: Polynomial Threshold Functions

- The class of Boolean functions $f : \{-1,1\}^n \to \{-1,1\}$ such that $f(x) = \text{sgn}(p(x))$, where p is a polynomial having a number of terms polynomial in n.
- Defined in [BS90], where it was also shown that PT_1 contains PL_1 (and this inclusion is strict), and that PT_1 is contained in PL_∞ (and this inclusion is strict).

PTAPE: Archaic for PSPACE

PTAS: Polynomial-Time Approximation Scheme

- The subclass of NPO problems that admit an *approximation scheme*in the following sense. For any $\epsilon > 0$, there is a polynomial-time algorithm that is guaranteed to find a solution whose cost is within a $1 + \epsilon$ factor of the optimum cost. (However, the exponent of the polynomial might depend strongly on ϵ.)
- Contains FPTAS, and is contained in APX.
- As an example, the Traveling Salesman Problem in the Euclidean plane is in PTAS [Aro96].

- Defined in [ACG+99].

PT/WK($f(n), g(n)$): Parallel Time $f(n)$ / Work $g(n)$

- The class of decision problems solvable by a uniform family of Booleancircuits with depth upper-bounded by $f(n)$ and size (number of gates) upper-bounded by g(n).
- The union of PT/WK($\log^k n, n^k$) over all constants k equals NC.

PZK: Perfect Zero Knowledge

- Same as SZK, but now the two distributions must be *identical*, not merely statistically close. (The "two distributions" are (1) the distribution over Arthur's view of his interaction with Merlin, conditioned on Arthur's random coins, and (2) the distribution over views that Arthur can *simulate* without Merlin's help.)
- Contained in SZK.
- See also: CZK.

QAC^0: Quantum AC^0

- The class of decision problems solvable by a family of constant-depth,polynomial-size quantum circuits. Here each layer of the circuit is a tensor product of one-qubit gates and Toffoli gates, or is a tensor product of controlled-NOT gates.
- A uniformity condition may also be imposed.
- Defined in [Moo99], where it was also shown that $\mathsf{QAC}^0 = \mathsf{QAC}^0[2] = \mathsf{QACC}^0$.

$\mathsf{QAC}^0[m]$: Quantum $\mathsf{AC}^0[m]$

- Same as QAC^0, except that now Mod_m gates are also allowed. A Mod_m gate computes whether the sum of a given set of bits is congruent to 0 modulo m, and exclusive-OR's the answer into another bit.
- Defined in [Moo99].

QACC^0: Quantum ACC^0

- Same as $\mathsf{QAC}^0[m]$, except that Mod-m gates are allowed for any m.
- Defined in [Moo99].
- [GHP00] showed that QACC^0 equals $\mathsf{QAC}^0[p]$ for any prime p.

QAM: Quantum AM

- The class of decision problems for which a "yes" answer can be verified by a public-coin quantum AM protocol, as follows. Arthur generates a uniformly random (classical) string and sends it to Merlin. Merlin responds with a polynomial-size quantum certificate, on which Arthur can perform any BQP operation. The completeness and soundness requirements are the same as for AM.
- Defined by Watrous [Wat02b].
- Contained in QIP(2) and in PSPACE.
- Contains QMA = QIP(1).

QCFL: Quantum CFL

- The class of decision problems recognized by quantum context-free languages, which are defined in [MC00]. The authors also showed that QCFL does not equal CFL.

QCMA: Quantum Classical MA

- The class of decision problems for which a "yes" answer can be verified by a *quantum* computer with access to a *classical* proof.
- Contains MA, and is contained in QMA.
- No oracle separation between QCMA and QMA is currently known.

QH: Query Hierarchy Over NP

- QH_i is defined to be $\mathsf{P}^{\mathsf{NP}[k]}$; that is, P with k queries to an NP oracle (where k is a constant). Then QH is the union of QH_i over all nonnegative i.
- QH = BH [Wag90]; thus, either both hierarchies are infinite or both collapse to some finite level.

QIP: Quantum IP

- The class of decision problems such that a "yes" answer can be verified by a *quantum interactive proof*. Here the verifier is a BQP (i.e. quantum polynomial-time) algorithm, while the prover has unbounded computational resources (though cannot violate the linearity of quantum mechanics). The prover and verifier exchange a polynomial number of messages, which can be quantum states. Thus, the verifier's and prover's states may become entangled during the course of the protocol. Given the verifier's algorithm,we require that

  1. If the answer is "yes," then the prover can behave in such a way that the verifier accepts with probability at least 2/3.
  2. If the answer is "no," then however the prover behaves, the verifier rejects with probability at least 2/3.

Let QIP(k) be QIP where the prover and verifier are restricted to exchanging k messages (with the prover going last).

- Defined in [Wat03], where it was also shown that PSPACE is in QIP(3).
- Subsequently [KW00] showed that for all $k > 3$, QIP(k) = QIP(3) = QIP.
- QIP is contained in EXP [KW00].
- QIP(1) is more commonly known as QMA.
- See also: QIP(2), QSZK.

QIP(2): 2-Round Quantum IP

- See QIP for definition.
- Contains QSZK [Wat02a].

QMA: Quantum MA

- The class of decision problems such that a "yes" answer can be verified by a 1-round quantum interactive proof. That is, a BQP (i.e. quantum polynomial-time) verifier is given a quantum state (the "proof"). We require that

  1. If the answer is "yes," then there exists a state such that verifier accepts with probability at least 2/3.
  2. If the answer is "no," then for all states the verifier rejects with probability at least 2/3.

  QMA = QIP(1).
- Defined in [Wat00], where it is also shown that *group non-membership* is in QMA. That is: let G be a group, whose elements are represented by polynomial-size strings. We're given a "black box" that correctly multiplies and inverts elements of G. Then given elements g and $h_1, \cdots, h_k$, we can verify in QMA that g is not in the subgroup generated by $h_1, \cdots, h_k$.
- Based on this, [Wat00] gives an oracle relative to which MA is strictly contained in QMA.
- Kitaev and Watrous (unpublished) showed QMA is contained in PP. Combining that result with [Ver92], one can obtain an oracle relative to which AM is not in QMA.
- Kitaev ([KSV02], see also [AN]) showed that the following problem is complete for QMA:

5-Local Hamiltonians. Given an n-qubit Hilbert space, as well as a collection $H_1, \cdots, H_k$ of Hamiltonians (i.e. Hermitian positive semidefinite matrices), each of which acts on at most 5 qubits of the space. Also given reals a, b such that $b - a = \Theta(1/\mathsf{poly}(n))$. Decide whether the smallest eigenvalue of $H = H_1 + \cdots + H_k$ is less than a or greater than b, promised that one of these is the case.

Subsequently Kempe and Regev [KR03] showed that even 3-Local Hamiltonians is QMA-complete. In recent unpublished work, the same duo has hit rock bottom (assuming P does not equal QMA), by showing 2-local Hamiltonians QMA-complete.

- Compare to NQP.
- If QMA = PP then PP contains PH [Vya03]. This result uses the fact that QMA is contained in $\mathsf{A}_0\mathsf{PP}$.
- See also: QSZK, QMA(2), QMA^+.

QMA^+: QMA With Super-Verifier

- Same as QMA, except now the verifier can directly obtain the *probability* that a given observable of the certificate state, if measured, would equal 1. (In the usual model, by contrast, one can only sample an observable.)
- Defined in [AR03], where it was also shown that $\mathsf{QMA}^+ = \mathsf{QMA}$.

QMA(2): Quantum MA With Multiple Certificates

- Same as QMA, except that now the verifier is given *two* polynomial-size quantum certificates, which are guaranteed to be unentangled.
- Defined in [KMY01]. It is unknown whether $\mathsf{QMA}(k) = \mathsf{QMA}(2)$ for all $k > 2$, and also whether $\mathsf{QMA}(2) = \mathsf{QMA}$.

$\mathsf{QMA}_{\log}$: QMA With Logarithmic-Size Proofs

- Same as QMA except that the quantum proof has $O(\log n)$ qubits instead of a polynomial number.
- Equals BQP (Watrous, unpublished).

QMAM: Quantum Merlin-Arthur-Merlin Public-Coin Interactive Proofs

- The class of decision problems for which a "yes" answer can be verified by a public-coin quantum MAM protocol, as follows. Merlin sends a polynomial-size quantum state to Arthur. Arthur then flips some classical coins (in fact, he only has to flip *one* without loss of generality) and sends the outcome to Merlin. At this stage Arthur is not yet allowed to perform any quantum operations. Merlin then sends Arthur another quantum state. Finally, Arthur performs a BQP operation on both of the states simultaneously, and either accepts or rejects. The completeness and soundness requirements are the same as for AM. Also, Merlin's messages might be entangled.

- Defined by Watrous [Wat02b], who also showed that QMAM = QIP(3) = QIP.
- Hence QMAM contains PSPACE.

QMIP: Quantum Multi-Prover Interactive Proofs

- The quantum generalization of MIP, and the multi-prover generalization of QIP.
- A quantum multi-prover interactive proof system is the same as a classical one, except that all messages and verifier computations are quantum. As in MIP, there is no communication among the provers; however, the provers share unlimited prior entanglement. The number of provers and number of rounds can both be polynomial in n.
- Defined in [KM02].
- Fascinatingly, no relationship between QMIP and NEXP is known. We don't know whether allowing the provers unlimited prior entanglement makes the class more powerful, less powerful, or both!

$\mathsf{QMIP}_{\mathsf{le}}$: Quantum Multi-Prover Interactive Proofs With Limited Prior Entanglement

- Same as QMIP, except that now the provers share only a polynomial number of EPR pairs, instead of an unlimited number.
- Defined in [KM02], where it was also shown that $\mathsf{QMIP}_{\mathsf{le}}$ is contained in NEXP = $\mathsf{QMIP}_{\mathsf{ne}}$.

$\mathsf{QMIP}_{\mathsf{ne}}$: Quantum Multi-Prover Interactive Proofs With No Prior Entanglement

- Same as QMIP, except that now the provers have no prior entanglement.
- Defined in [KM02], where it was also shown that $\mathsf{QMIP}_{\mathsf{ne}}$ = NEXP. Thus, $\mathsf{QMIP}_{\mathsf{ne}}$ contains $\mathsf{QMIP}_{\mathsf{le}}$.

QNC^0: Quantum NC^0

- Constant-depth quantum circuits without fanout gates.
- Defined in [Špalek03].
- Contained in QNC^0_f.

QNC^0_f: Quantum NC^0 With Unbounded Fanout

- Constant-depth quantum circuits with unbounded-fanout gates.
- Defined in [Špalek03].
- Contains QNC^0, and is contained in QACC^0.

QNC^1: Quantum NC^1

- Same as BQNC^1, but for the exact rather than bounded-error case.
- In contrast to NC^1, it is not clear how to simulate QNC^1 on a quantum computer in which one qubit is initialized to a pure state,and the remaining qubits are in the maximally mixed state [ASV00].
- See also [MN02].

QP: Quasipolynomial-Time

- Equals DTIME$(2^{\text{polylog}(n)})$.

QPLIN: Linear Quasipolynomial-Time

- Equals DTIME$(n^{O(\log n)})$.
- Has the same relationship to QP that E does to EXP.

QPSPACE: Quasipolynomial-Space

- Equals DSPACE$(2^{\text{polylog}(n)})$.
- According to [BG94], Beigel and Feigenbaum and (independently) Krawczyk showed that QPSPACE is not contained in Check.

QSZK: Quantum Statistical Zero-Knowledge

- A quantum analog of SZK (or more precisely HVSZK).
- Arthur is a BQP (i.e. quantum) verifier who can exchange quantum messages with Merlin. So Arthur and Merlin's states may become entangled during the course of the protocol.
- Arthur's "view" of his interaction with Merlin is taken to be the sequence of mixed states he has, over all steps of the protocol. The zero-knowledge requirement is that each of these states must have trace distance at most (say) 1/10 from a state that Arthur could prepare himself (in BQP), without help from Merlin. Arthur is assumed to be an honest verifier.
- Defined in [Wat02a], where the following was also shown:
  - QSZK is contained in PSPACE.
  - QSZK is closed under complement.
  - Any protocol can be parallelized to consist of two messages, so that QSZK is in QIP(2).

- One can assume without loss of generality that protocols are public-coin, as for SZK.
- QSZK has a natural complete promise problem, called *Quantum State Distinguishability* (QSD). We are given quantum circuits Q_0 and Q_1. Let ρ_0 and ρ_1 be the mixed states they produce respectively, when run on the all-0 state (and when non-output qubits are traced out). We are promised that the trace distance between ρ_0 and ρ_1 is either at most α or at least β, where α and β are constants in $[0,1]$ satisfying $\alpha < \beta^2$. The problem is to decide which of these is the case.

R: Recursive Languages

- The class of decision problems solvable by a Turing machine. Often identified with the class of "effectively computable" functions (the *Church-Turing thesis*).
- Defined in [Tur36], [Chu41], and other seminal early papers.
- Equals RE $\cap$ coRE.
- Strictly contains PR, the primitive recursive functions (see [Kle71]).

RE: Recursively Enumerable Languages

- The class of decision problems for which a "yes" answer can be verified by a Turing machine in a finite amount of time. (If the answer is "no," on the other hand, the machine might never halt.)
- Equivalently, the class of decision problems for which a Turing machine can list all the "yes" instances, one by one (this is what "enumerable" means).
- A problem C is complete for RE if (1) C is in RE and (2) any problem in RE can be reduced to C by a Turing machine.
- Actually there are two types of reduction: M-reductions (for *many-one*), in which a single instance of the original problem is mapped to an instance of C, and T-reductions (for *Turing*), in which an algorithm for the original problem can make arbitrarily many calls to an oracle for C.
- RE-complete sets are also called *creative* sets for some reason.
- The canonical RE-complete problem is the *halting problem*: i.e., given a Turing machine, does it halt when started on a blank tape?
- The famous unsolvability of the halting problem [Tur36] implies that R does not equal RE.
- Also, RE does not equal coRE.
- RE and coRE can be generalized to the *arithmetic hierarchy* AH.

- There are problems in RE that are neither RE-complete under T-reductions, nor in R [Fri57] [Muc56]. This is the resolution of *Post's problem* [Pos44].
- Indeed, RE contains infinitely many nonequivalent "T-degrees.' (A T-degree is a class of problems, all of which can be T-reduced to one another.) The structure of the T-degrees has been studied in more detail than you can possibly imagine [Sho99].

REG: Regular Languages

- The class of decision problems solvable by deterministic finite automata (DFA's).
- Equals the class solvable by nondeterministic finite automata (NDFA's).
- Equals DSPACE($O(1)$) [She59],which equals DSPACE($o(\log \log n)$) [HIS65].
- Includes, i.e., "Is the parity of the input odd?," but not "Are the majority of bits in the input 1's?" This is sometimes expressed as "finite automata can't count."
- Contained in NC^1.
- See e.g. [Koz97], [Gur89] for basic results on regular languages.

RevSPACE($f(n)$): Reversible $f(n)$-Space

- The class of decision problems solvable in space $O(f(n))$ by a reversible Turing machine (a deterministic Turing machine for which every configuration has at most one immediate predecessor).
- Was shown to equal DSPACE($f(n)$) [LMT97].

$\mathsf{R}_H\mathsf{L}$: Randomized Halting Logarithmic-Space

- Has the same relation to L as RP does to P. The randomized machine must halt for every input *and* every setting of the random tape.
- Contains undirected reachability (is there a path from vertex u to vertex v in an undirected graph?) [AKL+79].
- Contained in RL.

RL: Randomized Logarithmic-Space

- Has the same relation to L as RP does to P. The randomized machine must halt with probability 1 on any input.
- Contains $\mathsf{R_HL}$.
- Contained in SC [Nis92].

RNC: Randomized NC

- Has the same relation to NC as RP does to P.
- Contains the maximum matching problem for bipartite graphs [MVV87].
- Contained in BQNC.
- See also: coRNC.

RP: Randomized Polynomial-Time

- The class of decision problems solvable by an NP machine such that
  1. If the answer is "yes," at least 1/2 of computation paths accept.
  2. If the answer is "no," all computation paths reject.
- Defined in [Gil77].
- Contains the problem of testing whether an integer is prime [AH87].
- For other problems in RP, see the standard text on randomized algorithms, [MR95].
- See also: coRP, ZPP, BPP.

RPP: Restricted Pseudo Polynomial-Time

- The class of decision problems $\langle x, m \rangle$ (where x is an input of length $|x| = n$ and m is an integer parameter), that are solvable by a nondeterministic (i.e. NP) machine in $\mathsf{poly}(n + m)$ time and $O(m + \log n)$ space simultaneously.
- Defined in [Mon80].
- See also FPT.

RSPACE($f(n)$): Randomized $f(n)$-Space

- Same as RL, but for $O(f(n))$-space instead of logarithmic-space.
- Contained in NSPACE($f(n)$) and BPSPACE($f(n)$).

$\mathsf{S}_2\mathsf{P}$: Second Level of the Symmetric Hierarchy

- The class of decision problems for which there is a polynomial-time predicate P such that, on input x,
  1. If the answer is "yes," then there exists a y such that for all z, $P(x, y, z)$ is true.
  2. If the answer is "no," then there exists a z such that for all y, $P(x, y, z)$ is false.
- Note that this differs from Σ_2P in that the quantifiers in the second condition are reversed.

- Defined in [RS98], where it was also shown that $\mathsf{S_2P}$ contains MA and $\Delta_2\mathsf{P}$.
- Contained in $\mathsf{ZPP^{NP}}$ [Cai01].

$\mathsf{S_2\text{-}EXP \cdot P^{NP}}$: Don't Ask

- One of the caged classes of the Complexity Zoo.
- Has been implicated in a collapse scandal involving AM[polylog], coNP, and EH.

SAC: Semi-Unbounded-Fanin AC

- SAC_k is the class of decision problems solvable by a family of depth-$O(\log^k n)$ circuits with unbounded-fanin OR and bounded-fanin AND gates. Negations are only allowed at the input level.
- A uniformity condition may also be imposed.
- Defined by [BCD+89], who also showed that SAC_k is closed under complement for every $k > 0$.

$\mathsf{SAC^0}$: Semi-Unbounded-Fanin $\mathsf{AC^0}$

- See SAC for definition.
- Not closed under complement [BCD+89].

$\mathsf{SAC^1}$: Semi-Unbounded-Fanin $\mathsf{AC^1}$

- See SAC for definition.
- Equals LOGCFL [Ven91].
- Contained in $\oplus\mathsf{SAC^1}$ [GW96].

SAPTIME: Stochastic Alternating Polynomial-Time

- The class of problems solvable by a polynomial-time Turing machine with three kinds of quantifiers: existential, universal, and randomized.
- Defined in [Pap83], where it was also observed that SAPTIME = PSPACE.

SBP: Small Bounded-Error Probability

- The class of decision problems for which the following holds. There exists a #P function f and an FP function g such that, for all inputs x,
  1. If the answer is "yes" then $f(x) > g(x)$.
  2. If the answer is "no" then $f(x) < g(x)/2$.

- Defined in [BGM03], where the following was also shown:
  - SBP contains MA, WAPP, and ExistsBPP.
  - SBP is contained in AM and $\mathsf{BPP}_{\mathsf{path}}$.
  - There exists an oracle relative to which SBP is not contained in $\Sigma_2\mathsf{P}$.
  - SBP is closed under union.

SC: Steve's Class

- (Named in honor of Stephen Cook.)
- The class of decision problems solvable by a Turing machine that simultaneously uses polynomial time and polylogarithmic space.
- Note that SC might *not* equal $\mathsf{P} \cap \mathsf{DSPACE}(\mathsf{polylog}(n))$, since for the latter, it suffices to have two separate algorithms: one polynomial-time and the other polylogarithmic-space.
- Deterministic context-free languages (DCFL's) can be recognized in SC [Coo79].
- SC contains RL and BPL [Nis92].
- SC equals $\mathsf{DTISP}(\mathsf{poly}, \mathsf{polylog})$.

SE: Subexponentially-Solvable Search Problems

- The class of FNP search problems solvable in $O(2^\epsilon)$ time for every $\epsilon > 0$.
- Defined in [IPZ01], who also gave reductions showing that if any of k-SAT, k-colorability, k-set cover, clique, or vertex cover is in SE, then all of them are.

SEH: Strong Exponential Hierarchy

- The union of NE, $\mathsf{NP}^{\mathsf{NE}}$, $\mathsf{NP}^{\mathsf{NP}^{\mathsf{NE}}}$, and so on.
- Is called "strong" to contrast it with the ordinary Exponential Hierarchy EH, which it contains.
- Note that we would get the same class if we replaced NE by NEXP.
- SEH collapses to P^{NE} [Hem89].

SelfNP: Self-Witnessing NP

- The class of languages L in NP such that the union, over all x in L, of the set of valid witnesses for x equals L itself.
- Defined in [HT03], where it was shown that the closure of SelfNP under polynomial-time many-one reductions is NP.

- They also show that if SelfNP = NP, then E = NE; and that SAT is contained in SelfNP.
- See also: PermUP.

SF_k: Width-k Bottleneck Turing Machines

- The class of decision problems solvable by a k-bottleneck Turing machine. This is a machine that, once every $p(n)$ steps (where $p(n)$ is some polynomial), erases everything on the tape except for a single "safe-storage," which can take on any integer value from 1 to k. (There's also a counter recording how many erasings have occurred so far.)
- Defined in [CF91], where it was also shown that $\mathsf{SF}_5 = \mathsf{PSPACE}$.
- The complexity of SF_2, SF_3, and SF_4 was studied in [Ogi94] and [Her97]. The following result of those authors is among the caged beasts of the Complexity Zoo:

  SF_4 is contained in $\oplus\mathsf{P}^{\mathsf{Mod}_3\mathsf{P}^{\oplus\mathsf{P}^{\mathsf{Mod}_3\mathsf{P}^{\oplus\mathsf{P}}}}}$

- (Here the BP operator means that one makes the class into a bounded-error probabilistic class, the same way one makes P into BPP and NP into MA.)

$\Sigma_2\mathsf{P}$: NP With NP Oracle

- Complement of $\Sigma_2\mathsf{P}$.
- Along with $\Sigma_2\mathsf{P}$, comprises the second level of PH, thepolynomial hierarchy.
- [Uma98] has shown that the following problems are complete for $\Sigma_2\mathsf{P}$:
  - *Minimum equivalent DNF.* Given a DNF formula F and integer k, is there a DNF formula equivalent to F with k or fewer occurrences of literals?
  - *Shortest implicant.* Given a DNF formula F and integer k, is there a conjunction of k or fewer literals that implies F?
- For any fixed k, there is a problem in $\Sigma_2\mathsf{P} \cap \Sigma_2\mathsf{P}$ that cannot be solved by circuits of size n^k [Kan82].

SKC: Statistical Knowledge Complexity

- A hierarchy of generalizations of SZK, in which Arthur is allowed to gain *some* information from his interaction with Merlin.
- Defined in [GP91].
- There are several variants (which we only describe roughly), including:
  - $\mathsf{SKC}_{hint}(k(n))$: Hint sense. The simulator can reproduce Arthur's view of the protocol if given a hint string of size $k(n)$.

- $\mathsf{SKC}_{hint}(k(n))$: Strict oracle sense. The simulator can reproduce Arthur's view if allowed $k(n)$ queries to an oracle O.
- $\mathsf{SKC}_{avg}(k(n))$: Average oracle sense. For each input, the *expected* number of queries the simulator makes to oracle O is at most $k(n)$.
- $\mathsf{SKC}_{ent}(k(n))$: Entropy sense. Defined in [ABV95]. For each input, the expectation (over Arthur's random coins) of $-\log(P)$ is at most $k(n)$, where P is the probability that the view output by the simulator equals the view resulting from the actual protocol.

- See also: PKC.

SL: Symmetric Logarithmic-Space

- The class of problems solvable by a nondeterministic Turing machine in logarithmic space, such that
  1. If the answer is "yes," one or more computation paths accept.
  2. If the answer is "no," all paths reject.
  3. If the machine can make a nondeterministic transition from configuration A to configuration B, then it can also transition from B to A. (This is what "symmetric" means.)
- Defined in [LP82].
- The reachability problem (is there a path from vertex s to vertex t?) for undirected graphs is complete for SL, under L-reduction.
- SL contains L, and is contained in NL.
- It follows from [AKL+79] that SL is contained in L/poly.
- [KW93] showed that SL is contained in $\oplus$L, as well as Mod_kL for every prime k.
- SL is also contained in $\mathsf{DSPACE}(\log^{3/2} n)$ [NSW92], and indeed in $\mathsf{DSPACE}(\log^{4/3} n)$ [ATSWZ00].
- [NTS95] showed that SL equals coSL, and furthermore that $\mathsf{SL}^{\mathsf{SL}} = \mathsf{SL}$ (that is, the symmetric logspace hierarchy collapses).
- News flash! [Rei04] has shown that SL = L.

SLICEWISEPSPACE

- The parameterized version of PSPACE.
- Same as FPT, except that now on input $\langle x, k \rangle$ (k a parameter), the *space* used must be $f(k)p(|x|)$, where $p(n)$ is a polynomial.

- If P = PSPACE, then FPT = SLICEWISEPSPACE.
- Defined in [DF99].

SNP: Strict NP

- [Fag74] showed that NP is precisely the class of decision problems reducible to a graph-theoretic property expressible in second-order existential logic. For example (see [Pap94a]), the property "graph G has a Hamiltonian path" is expressible as

  > There exists a relation $P(u, v)$ on vertices of G, such that $P(u, u)$ is false, and for all distinct u, v either $P(u, v)$ or $P(v, u)$, and $P(u, v)$ and $P(v, w)$ implies $P(u, w)$, and if $P(u, w)$ and there does not exist a v for which $P(u, v)$ and $P(v, w)$, then there is an edge from u to w.

- (Here the relation $P(u, v)$ defines a *total order* on vertices, such that any two consecutive vertices must be adjacent.)
- Then SNP is the class of decision problems reducible to a graph-theoretic predicate with *only*universal quantifiers over vertices, no existential quantifiers. As an example, k-SAT (CNF satisfiability with at most k literals per clause, for k a constant)is in SNP. But general SAT is not in SNP, basically because we're not allowed to say, "There exists a literal in this clause that satisfies the clause."
- See also: MaxSNP.

SO-E: Second Order Existential

- The class of decision problems for which a "yes" answer is expressible by a proposition with second-order existential quantifiers followed by a first-order formula. See [Imm98] for a full definition.
- SO-E = NP [Fag74].
- Contains FO(poly(n)).

SP: Semi-Efficient Parallel

- The class of problems in P for which the best parallel algorithm (using a polynomial number of processors) is faster than the best serial algorithm by a factor of $\Omega(n^\epsilon)$ for some $\epsilon > 0$.
- Defined in [KRS90].

SP: Alternate Name for $\mathsf{XP}_{\mathsf{uniform}}$
span-P: Span Polynomial-Time

- The class of functions computable as $|S|$, where S is the set of output values returned by the accepting paths of an NP machine.

- Defined in [KSTT89a], where it is also shown that span-P contains #P and OptP; and that span-P = #P if and only if UP = NP.

SPARSE: Sparse Languages

- The class of decision problems for which the number of "yes" instances of size n is upper-bounded by a polynomial in n. If SPARSE intersects NPC then P = NP [Mah82].
- Contains TALLY.

SPL: Stoic PL

- Has the same relation to PL as SPP does to PP.
- Contains the maximum matching and perfect matching problems under a pseudorandom assumption [ARZ99].
- Contains UL.
- Contained in $\mathsf{C}_{=}\mathsf{L}$ and $\mathsf{Mod}_k\mathsf{L}$.
- Equals the set of problems low for GapL.

SPP: Stoic PP

- The class of decision problems solvable by an NP machine such that

  1. If the answer is "no," then the number of accepting computation paths exactly equals the number of rejecting paths.
  2. If the answer is "yes," then these numbers differ by 2.

  (A technicality: If the total number of paths is even then the numbers can't differ by 1.)

- Defined in [FFK94], where it was also shown that SPP is low for PP, $\mathsf{C}_{=}\mathsf{P}$, $\mathsf{Mod}_k\mathsf{P}$, and SPP itself. (I.e. adding SPP as an oracle does not increase the power of these classes.)
- Independently defined in [OH93], who called the class XP.
- Contained in LWPP, $\mathsf{C}_{=}\mathsf{P}$, and WPP among other classes.
- Contains FewP; indeed, FewP is low for SPP, so that $\mathsf{SPP}^{\mathsf{FewP}} = \mathsf{SPP}$ [FFK94].
- Contains the problem of deciding whether a graph has any nontrivial automorphisms [KSTT92].
- Indeed, contains graph isomorphism [AK02].

- Contains a whole gaggle of problems for solvable black-box groups: solvability testing, membership testing, subgroup testing, normality testing, order verification, nilpotetence testing, group isomorphism, and group intersection [Vin04a]

- [AK02] also showed that the Hidden Subgroup Problem for permutation groups, of interest in quantum computing, is in $\mathsf{FP}^{\mathsf{SPP}}$.

SUBEXP: Deterministic Subexponential-Time

- The intersection of $\mathsf{DTIME}(2^{n^\epsilon})$ over all $\epsilon > 0$. (Note that the algorithm used may vary with ϵ.)

symP: Alternate Name for $\mathsf{S_2P}$
SZK: Statistical Zero Knowledge

- The class of decision problems for which a "yes" answer can be verified by a *statistical zero-knowledge proof protocol*. In such a protocol, we have a BPP (i.e. probabilistic polynomial-time) verifier, Arthur, and a prover, Merlin, who has unbounded computational resources. By sending messages back and forth with Merlin, Arthur must become convinced (with high probability) that the answer is "yes," *without learning anything else about the problem* (statistically).

- What does that mean? For each choice of random coins, Arthur has a "view" of his entire interaction with Merlin, consisting of his random coins as well as all messages sent back and forth. Then the distribution over views resulting from interaction with Merlin has to be statistically close to a distribution that Arthur could generate himself (in polynomial-time), without interacting with Merlin. (Here "statistically close" means that, say, the trace distance is at most 1/10.)

- The most famous example of such a protocol is for *graph nonisomorphism*. Given two graphs G and H, Arthur can pick one of the graphs (each with probability 1/2), permute its vertices randomly, send the resulting graph to Merlin,and ask, "Which graph did I start with, G or H?" If G and H are non-isomorphic, Merlin can always answer correctly (since he can use exponential time), but if they're isomorphic, he can answer correctly with probability at most 1/2. Thus, if Merlin always gives the correct answer, Arthur becomes convinced the graphs are not isomorphic. On the other hand, Arthur already *knew* which graph (G or H) he started with, so he could simulate his entire view of the interaction himself, without Merlin's help.

- If that sounds like a complicated definition, well, it is. But it turns out that SZK has extremely nice properties. [Oka96] showed that:

  - SZK is closed under complement. I.e. Arthur can verify in zero-knowledge that two graphs *are* isomorphic, not only that they aren't.
  - We can assume without loss of generality that the whole interaction consists of a constant number of messages.

– Amazingly, we can also assume without loss of generality that the protocol is *public-coin*. I.e. Arthur doesn't need to hide any of his random bits, as he did in the graph nonisomorphism protocol above, but can just send them all to Merlin!

– Finally, we can assume without loss of generality that the verifier (Arthur) is *honest*. A dishonest verifier would be one that tries to learn something about the problem (violating the zero-knowledge requirement) by deviating from the protocol.

Subsequently, [SV97] showed that SZK has a natural complete promise problem, called *Statistical Difference (SD)*. Given two polynomial-size circuits, C_0 and C_1, let D_0 and D_1 be the distributions over their respective outputs when they're given as input a uniformly random n-bit string. We're promised that D_0 and D_1 have trace distance either at most $1/3$ or at least $2/3$; the problem is to decide which is the case.

- *Note:* The constants $1/3$ and $2/3$ can be amplified to $2^{-\mathsf{poly}(n)}$ and $1-2^{-\mathsf{poly}(n)}$ respectively. But it is crucial that $(2/3)^2 > 1/3$.
- Another complete promise problem for SZK is *Entropy Difference (ED)* [GV99]. Here we're promised that either $\mathcal{H}(D_0) > \mathcal{H}(D_1)+1$ or $\mathcal{H}(D_1) > \mathcal{H}(D_0)+1$, where the distributions D_0 and D_1 are as above, and $\mathcal{H}$ denotes Shannon entropy. The problem is to determine which is the case.
- If any hard-on-average language is in SZK, then one-way functions exist [Ost91].
- Zero-knowledge proofs were first studied in [GMW91], [GMR89].
- Contains PZK and NISZK, and is contained in AM $\cap$ coAM, as well as CZK and QSZK.
- There exists an oracle relative to which SZK is not in BQP [Aar02b].
- Contained in DQP [Aar02a].

SZK_h: SZK With Limited Help

- The class of decision problems for which a "yes" answer can be verified by a statistical zero-knowledge proof protocol, and the prover and verifier both have access to a string computed by a trusted probabilistic polynomial-time third party with access to the input.
- Defined in [BOG03], where it was also shown that $\mathsf{SZK}_h = \mathsf{SZK}$.
- Contains NISZK_h.

TALLY: Tally Languages

- The class of decision problems for which every "yes" instance has the form 0^n (i.e. inputs are encoded in unary). If TALLY intersects NPC then P = NP [Mah82].

- Contained in SPARSE.

TC^0: Constant-Depth Threshold Circuits

- The class of decision problems solvable by polynomial-size, constant-depth circuits with unbounded fanin, which can use AND, OR, and NOT gates (as in AC^0) as well as *threshold* gates. A threshold gate returns 1 if at least half of its inputs are 1, and 0 otherwise.
- A uniformity requirement is sometimes also placed.
- TC^0 contains ACC^0, and is contained in NC^1.
- TC^0 circuits of depth 3 are strictly more powerful than TC^0 circuits of depth 2 [HMP+93].
- TC^0 circuits of depth 3 and quasipolynomial size can simulate all of ACC^0 [GK93].
- [NR97] give a candidate pseudorandom function family computable in TC^0, that is secure assuming a subexponential lower bound on the hardness of factoring. (See also [NRR01] for an improvement of this construction, as well as [Kha93].)
- One implication is that, assuming such a bound, there is no *natural proof* in the sense of [RR97] separating TC^0 from $\mathsf{P/poly}$. (It is important for this that a *function family*, and not just a candidate pseudorandom *generator*, is computable in TC^0.) Another implication is that functions in TC^0 are likely to be difficult to learn.
- The permanent of a 0-1 matrix cannot be computed in *uniform* TC^0 [All99].
- In a breakthrough result [Hes01] (building on [BCH86] and [CDL01]), integer division was shown to be in L-uniform TC^0. Indeed division is *complete* for this class under AC^0 reductions.

TFNP: Total Function NP

- The class of function problems of the following form:

  > Given an input x and a polynomial-time predicate $F(x,y)$, output any y satisfying $F(x,y)$. (Such a y is promised to exist.)

- Defined in [MP91].
- Contained in FNP.
- Subclasses include PPA, PPP, and PLS.

$\Theta_2\mathsf{P}$: Alternate name for $\mathsf{P}^{\mathsf{NP}[\log]}$

TreeBQP: BQP Restricted To Tree States

- The class of languages accepted by a BQP machine subject to the constraint that at every time step t, the machine's state is exponentially close to a *tree state*—that is, a state expressible by a polynomial-size tree of additions and tensor products (together with complex constants and $|0>$ and $|1>$ leaf nodes).

- More formally, a uniform classical polynomial-time algorithm generates a sequence of gates $g^{(1)}, \cdots, g^{(p(n))}$. Each $g^{(t)}$ can be either be selected from some finite universal basis of unitary gates (the choice turns out not to matter), or can be a 1-qubit measurement. When we perform a measurement, the state evolves to one of two possible pure states, with the usual probabilities, rather than to a mixed state. We require that the final gate $g^(p(n))$ is a measurement of the first qubit. If at least one intermediate state was more than distance $2^{-\Omega(n)}$ away from the nearest state of tree size at most p(n), then the outcome of the final measurement is chosen adversarially; otherwise it is given by the usual Born probabilities. The measurement must return 1 with probability at least 2/3 if the input is in the language, and with probability at most 1/3 otherwise.

- Contains BPP, and is contained in BQP.

- Defined in [Aar04c], where it was also shown that TreeBQP is contained in the third level of PH, which might provide weak evidence that TreeBQP does not equal BQP.

TREE-REGULAR

- Same as REG, except that now the inputs are *trees* (say, binary trees) instead of strings. Each vertex is labeled with a symbol from a fixed alphabet. Evaluation begins at the leaves and proceeds to the root. The state of the finite automaton at each vertex v is a function of (1) the states at v's children (if any), and (2) the symbol at v. The tree is in the language if and only if the automaton is in an "accept" state at the root.

- See [Koz92] for example.

UAP: Unambiguous Alternating Polynomial-Time

- Same as AP, except we are promised that each existential quantifier has at most one "yes" path, and each universal quantifier has at most one "no" path.

- Contains UP.

- Defined in [NR98], where it was also shown that, even though AP = PSPACE, it is unlikely that the same is true for UAP, since UAP is contained in SPP.

- [CGRS04] have also shown that $\mathsf{UAP}^{\mathsf{UAP}} = \mathsf{UAP}$, and that UAP contains the Graph Isomorphism problem.

UCC: Unique Connected Component

- The class of problems reducible in L to the problem of whether an undirected graph has a unique connected component.

- See [AG00] for more information.
- Contained in SL.
- See also coUCC.

UL: Unambiguous L

- Has the same relation to L as UP does to P.
- If UL = NL, then FNL is contained in #L [AJ93].

UL/poly: Nonuniform UL

- Has the same relation to UL as P/poly does to P.
- Equals NL/poly [RA00] (a corollary is that UL/poly is closed under complement).

UE: Unambiguous Exponential-Time With Linear Exponent

- Has the same relation to E as UP does to P.

UP: Unambiguous Polynomial-Time

- The class of decision problems solvable by an NP machine such that
  1. If the answer is "yes," exactly one computation path accepts.
  2. If the answer is "no," all computation paths reject.
- One-way functions exist if and only if P does not equal UP ([GS88] and independently [Ko85]).
- One-way permutations exist if and only if P does not equal UP $\cap$ coUP [HT03].
- There exists an oracle relative to which P is strictly contained in UP is strictly contained in NP [Rac82]; indeed, these classes are distinct with probability 1 relative to a random oracle [Bei89].
- NP is contained in $\mathsf{RP}^{\mathsf{UP}}$ [VV86]. On the other hand, [BBF98] give an oracle relative to which P = UP but still P does not equal NP.
- UP is not known or believed to contain complete problems. [Sip82], [HH86] give oracles relative to which UP has no complete problems.

US: Unique Polynomial-Time

- The all-American counting class.
- The class of decision problems solvable by an NP machine such that the answer is "yes" if and only if exactly one computation path accepts.

- In contrast to UP, a machine can legally have more than one accepting path - that just means that the corresponding input is not in the language.
- Defined in [BG82].
- Contains coNP.

VNC_k: Valiant NC Over Field k

- Has the same relation to VP_k as NC does to P.
- More formally, the class of VP_k problems computable by a straight-line program of depth polylogarithmic in n.
- Surprisingly, $\mathsf{VNC}_k = \mathsf{VP}_k$ for any k [VSBR83].

VNP_k: Valiant NP Over Field k

- A superclass of VP_k in Valiant's algebraic complexity theory, but not *quite* the analogue of NP.
- A problem is in VNP_k if there exists a polynomial p with the following properties:
  - p is computable in VP_k; that is, by a polynomial-size straight-line program.
  - The inputs to p are constants $c_1, \cdots, c_m, e_1, \cdots, e_h$ and indeterminates $x_1, \cdots, x_n$ over the base field k.
  - When p is summed over all 2^h possible assignments of $\{0, 1\}$ to each of $e_1, \cdots, e_h$, the result is some specified polynomial q.
- Originated in [Val79a].
- If the field k has characteristic greater than 2, then the permanent of an n-by-n matrix of indeterminates is VNP_k-complete under a type of reduction called p-projections ([Val79a]; see also [Br00]).
- A central conjecture is that for all k, VP_k is not equal to VNP_k. Bürgisser [Br00] shows that if this were false then:
  - If k is finite, $\mathsf{NC}^2/\mathsf{poly} = \mathsf{P}/\mathsf{poly} = \mathsf{NP}/\mathsf{poly} = \mathsf{PH}/\mathsf{poly}$.
  - If k has characteristic 0, then assuming the Generalized Riemann Hypothesis (GRH), $\mathsf{NC}^3/\mathsf{poly} = \mathsf{P}/\mathsf{poly} = \mathsf{NP}/\mathsf{poly} = \mathsf{PH}/\mathsf{poly}$, and $\#\mathsf{P}/\mathsf{poly} = \mathsf{FP}/\mathsf{poly}$.

  In both cases, PH collapses to $\Sigma_2\mathsf{P}$.

VP_k: Valiant P Over Field k

- The class of efficiently-solvable problems in Valiant's algebraic complexity theory.

- More formally, the input consists of constants $c_1, \cdots, c_m$ and indeterminates $x^1, \cdots, x_n$ over a base field k (for instance, the complex numbers or $\mathbb{Z}_2$). The desired output is a collection of polynomials over the x_i's. The complexity is the minimum number of pairwise additions, subtractions, and multiplications needed by a straight-line program to produce these polynomials. VP_k is the class of problems whose complexity is polynomial in n. (Hence, VP_k is a nonuniform class, in contrast to $\mathsf{P}_{\mathbb{C}}$ and$\mathsf{P}_{\mathbb{R}}$.)
- Originated in [Val79a]; see [Br00] for more information.
- Contained in VNP_k and VQP_k, and contains VNC_k.

VQP_k: Valiant QP Over Field k

- Has the same relation to VP_k as QP does to P.
- Originated in [Val79a].
- The determinant of an n-by-n matrix of indeterminates is VQP_k-complete under a type of reduction called qp-projections (see [Br00] for example). It is an open problem whether the determinant is VP_k-complete.

W[1]: Weighted Analog of NP

- The class of decision problems of the form $\langle x, k \rangle$ (k a parameter), that are *fixed-parameter reducible* to the following:

  > Weighted 3SAT: Given a 3SAT formula, does it have a satisfying assignment of Hamming weight k?

  A fixed-parameter reduction is a Turing reduction that takes time at most $f(k)p(|x|)$, where f is an arbitrary function and p is a polynomial. Also, if the input is $\langle x, k \rangle$, then all Weighted 3SAT instances the algorithm queries about must have the form $\langle x, k' \rangle$ where k' is at most k.
- Contains FPT.
- Defined in [DF99], where the following is also shown:
  - If FPT = W[1] then NP is contained in $\mathsf{DTIME}(2^{o(n)})$.
- W[1] can be generalized to W[t].

WAPP: Weak Almost-Wide PP

- The class of decision problems for which there exists a #P function f, and an $\epsilon > 0$, such that for all inputs x,
  1. If the answer is "yes" then $f(x) > (1+\epsilon)2^{\mathsf{poly}(|x|)}$.

2. If the answer is "no" then $f(x) < (1-\epsilon)2^{\mathsf{poly}(|x|)}$.

- Defined in [BGM03], where it is also shown that WAPP is contained in AWPP and SBP.

W[P]: Weighted Circuit Satisfiability

- The class of decision problems of the form $\langle x, k\rangle$ (k a parameter), that are fixed-parameter reducible to the following problem, for some constant h:

  > Weighted Circuit-SAT: Given a Boolean circuit C (with no restriction on depth), does C have a satisfying assignment of Hamming weight k?

- See W[1] for the definition of fixed-parameter reducibility.
- Defined in [DF99].
- Contains W[SAT].

WPP: Wide PP

- The class of decision problems solvable by an NP machine such that

  1. If the answer is "no," then the number of accepting computation paths exactly equals the number of rejectingpaths.
  2. If the answer is "yes," then these numbers differ by a function $f(x)$ computable in polynomial time (i.e. FP).

  Defined in [FFK94].

- Contained in $\mathsf{C_{=}P} \cap \mathsf{coC_{=}P}$, as well as AWPP.
- Contains SPP and LWPP.

W[SAT]: Weighted Satisfiability

- The class of decision problems of the form $\langle x, k\rangle$ (k a parameter), that are fixed-parameter reducible to the following problem, for some constant h:

  > Weighted SAT: Given a Boolean formula F (with no restriction on depth), does F have a satisfying assignment of Hamming weight k?

- See W[1] for the definition of fixed-parameter reducibility.
- Defined in [DF99].
- Contains W[t] for every t, and is contained in W[P].

W[$*$]: Union of W[t]'s

- The union of W[t] over all t.

W[t]: Nondeterministic Fixed-Parameter Hierarchy

- A generalization of W[1].
- The class of decision problems of the form $\langle x, k \rangle$ (k a parameter), that are fixed-parameter reducible to the following problem, for some constant h:

  > Weighted Weft-t Depth-h Circuit-SAT: Given a Boolean circuit C, with a mixture of fanin-2 and unbounded-fanin gates. The number unbounded-fanin gates along any path to the root is at most t, and the total depth (fanin-2 and unbounded-fanin) is at most h. Does C have a satisfying assignment of Hamming weight k?

- See W[1] for the definition of fixed-parameter reducibility.
- Defined in [DF99].
- Contained in W[SAT] and in W*[t].

W*[t]: W[t] With Parameter-Dependent Depth

- Same as W[t], except that now the circuit depth can depend on the parameter k rather than being constant. (The number of unbounded-fanin gates along any path to the root is still at most t.)
- W*[1] = W[1] [DFT96], and W*[2] = W[2] [DF97], but the problem is open for larger t.

XOR-MIP*[2, 1]: MIP*[2, 1] With XOR Restriction

- Same as MIP*[2, 1], but with the further restriction that both provers send only a single bit to the verifier, and the verifier's output is the exclusive-OR of those bits.
- Defined by [CHTW04], whose motivation was a connection to the Bell and CHSH inequalities of quantum physics.
- Interestingly, [CHTW04] showed that XOR-MIP*[2, 1] is contained in EXP, from which it follows that XOR-MIP*[2, 1] cannot capture the full power of multi-prover interactive proofs unless EXP = NEXP.

XP: Fixed-Parameter Tractable for Each Parameter

- The class of decision problems of the form $\langle x, k \rangle$ (k a parameter) that are solvable in time $O(|x|^{f(k)})$ for some function f. The algorithm used may depend on k.
- Defined in [DF99].
- Contains $\mathsf{XP}_{\mathsf{uniform}}$.
- Strictly contains FPT (by diagonalization).

$\mathsf{XP}_{\mathsf{uniform}}$: Uniform XP

- Same as XP except that the algorithm used must be the same for each k (though it can take k as input).
- Defined in [DF99].

YACC: Yet Another Complexity Class

- A term of derision, used against a complexity class.

ZPE: Zero-Error Probabilistic E

- Same as ZPP, but with $2^{O(n)}$-time instead of polynomial-time.
- ZPE = EE if and only if ZPP = EXP [IKW01].

ZPP: Zero-Error Probabilistic Polynomial-Time

- Defined to be the intersection of RP and coRP.
- The class of problems solvable by randomized algorithms that *always* return the correct answer, and whose *expected* running time (on any input) is polynomial.
- Defined in [Gil77].
- Contains the problem of testing whether an integer is prime [SS77] [AH87].
- In contrast to BPP and RP, it is not known whether showing ZPP = P requires proving superpolynomial circuit lower bounds [KI02].
- There exists an oracle relative to which ZPP = EXP [Hel84].

ZPTIME($f(n)$): Zero-Error Probabilistic $f(n)$-Time

- Same as ZPP, but with $O(f(n))$-time instead of polynomial-time.
- For any constructible superpolynomial f, ZPTIME($f(n)$) with NP oracle is not contained in P/poly [KW98].

5 Special Zoo Exhibit: Classes of Quantum States and Probability Distributions

24 classes and counting

A whole new phylum of the Complexity kingdom has recently been identified. This phylum consists of classes, not of problems or languages, but of *quantum states* and *probability distributions*. Well, actually, infinite *families* of states and distributions, one for each number of bits n. Admittedly, computer scientists have been talking about the complexity of sampling from probability distributions for years, but they haven't tended to organize those distributions into *classes* designated by *inscrutable sequences of capital letters*. This needs to change. The present Zookeeper has started the analogous project for quantum states; indeed, he puts the classes below in a special exhibit to avoid the appearance of favoritism toward his own classes in the main Zoo.

AmpP	OTree	Σ_2	TreeSize(f(n))
Basis	ProbP	Σ_3	TSH
Circuit	ψP	$\otimes_4 \cap \otimes_4$	Vidal
Dist	Pure	$\otimes_1$	
FPAUS	Samplable	$\otimes_2$	
Mixed	Separable	$\otimes_3$	
MOTree	Σ_1	Tree	

AmpP AmpP: States with Polytime Computable Amplitudes

- The class of Pure quantum state families $|\psi_n>=\Sigma_x \alpha_x |x>$ such that for all n, b, there exists a classical circuit of size $p(n+b)$ that outputs α_x to b bits of precision given x as input, for some polynomial p.
- Defined in [Aar04c], where the following was also shown:
  - If BQP = $\mathsf{P}^{\#\mathsf{P}}$ then AmpP is contained in ψP.
  - If AmpP is contained in ψP then NP BQP/poly.
  - If P = $\mathsf{P}^{\#\mathsf{P}}$ then ψP = AmpP.
  - If ψP is contained in AmpP then BQP is contained in P/poly.
- AmpP contains Circuit and Tree, and is contained in Pure.

Basis: Computational Basis States

- The class of quantum state families of the form $|x>$ where x is an n-bit string.
- The zeroth level of TSH.
- Equals Pure intersect Dist.

Circuit: Circuit States

- A generalization of Tree, where we allow amplitudes to be computed by polynomial-size multilinear *circuits* rather than just multilinear formulas.
- Defined in [Aar04c], where it was also observed that Circuit contains Vidal (indeed, strictly contains it).
- Contained in AmpP.

Dist: Classical Probability Distributions

- The class of classical probability distribution families over n-bit strings.
- Contains Basis, and is contained in Mixed.

FPAUS: Fully Polynomial Almost Uniform Samplable

- The class of probability distribution families D_n over n-bit strings such that (1) D_n is the uniform distribution over some subset, and (2) there exists a uniform probabilistic algorithm, running in time polynomial in n and $\log(1/\delta)$, that outputs a sample from a distribution at most δ from D_n in variation distance.
- See [JW04] for more information.

Mixed: Mixed Quantum States

- The class of mixed quantum state families of n qubits.
- Contains Pure and Dist.

MOTree: Manifestly Orthogonal Tree States

- The class of quantum state families in Tree representable by polynomial-size trees in which all additions are of states that are mutually *manifestly orthogonal*. Two states of n qubits, $\Sigma_x \alpha_x |x>$ and $\Sigma_x \beta_x |x>$, are manifestly orthogonal if $\alpha_x \beta_x = 0$ for all x.
- Defined in [Aar04c], where the following was also shown:
  - MOTree is *strictly* contained in Tree. Indeed, MOTree does not contain Σ_2.
  - MOTree strictly contains $\otimes_2$.

OTree: Orthogonal Tree States

- The class of quantum state families in Tree representable by polynomial-size trees in which all additions are of mutually orthogonal states.
- Contains MOTree.

- Defined in [Aar04c], where it was also shown that OTree is contained in ψP.
- Showing a separation between Tree and OTree remains an excellent open problem.

ProbP: Distributions With Polytime-Computable Probabilities
ψP: Pure States Preparable by Polynomial-Size Circuits

- The class of Pure quantum state families $|\psi_n>$ such that for all n and $\epsilon > 0$, there exists a quantum circuit of size $p(n+\log(1/\epsilon))$ that maps the all-0 state to a state some subsystem of which has trace distance at most $1-\epsilon$ from $|\psi_n>$, for some polynomial p. Because of the Solovay-Kitaev Theorem (see [NC00]), the definition of ψP is invariant under the choice of universal gate set.
- The following was shown in [Aar04c]:
  - ψP is not contained in Tree.
  - OTree is contained (indeed strictly contained) in ψP.
  - If BQP = $\mathsf{P}^{\#\mathsf{P}}$ then AmpP is contained in ψP.
  - If AmpP is contained in ψP then NP is contained in BQP/poly.
  - If P = $\mathsf{P}^{\#\mathsf{P}}$ then ψP = AmpP.
  - If ψP is contained in AmpP then BQP is contained in P/poly.
- Whether Tree is contained in ψP remains an intriguing open problem.

Pure: Pure Quantum States

- The class of quantum state families of n qubits that have the form $\Sigma_x \alpha_x |x>$; that is, that cannot be written as a nontrivial probability distribution over two other quantum states.
- Contains Basis, and is contained in Mixed.

Samplable: Polytime-Samplable Distributions
Separable: Separable Mixed States

- The class of Mixed quantum state families of n qubits that can be written as a probability distribution over $\oplus_1$ states.
- A well-known open problem asks whether a quantum computer that is in a separable state at every time step can be simulated in BPP, or alternatively, whether such a computer can simulate BQP.

Σ_1: First Sum Class in TSH

- The class of Pure quantum state families that can be written as superpositions over a polynomial number of computational basis states.

- Strictly contains Basis, and is strictly contained in Σ_2 and $\otimes_2$.

Σ_2: Second Sum Class in TSH

- The class of Pure quantum state families that can be written as superpositions over a polynomial number of Separable (or equivalently $\otimes_1$) states.
- Strictly contains Σ_1 and $\otimes_1$, and is strictly contained in Σ_3 and $\otimes_3$.

Σ_3: Third Sum Class in TSH

- The class of Pure quantum state families that can be written as superpositions over a polynomial number of $\otimes_2$ states.
- Strictly contains Σ_2 and $\otimes_2$, and is contained in $\Sigma_4 \cap \otimes_4$ (strict containment is not yet known for this and higher levels of TSH).

$\Sigma_4 \cap \otimes_4$: Intersection of Fourth Levels of TSH

- Does not equal $\otimes_3$ [Aar04c].

$\otimes_1$: First Tensor Class in TSH

- Equals Separable intersect Pure.
- Strictly contains Basis, and is strictly contained in Σ_2 and $\otimes_2$.

$\otimes_2$: Second Tensor Class in TSH

- The class of Pure quantum state families that can be written as a tensor product of Σ_1 states.
- Defined in [Aar04c], where the following was also shown:
  - $\otimes_2$ is strictly contained in $\otimes_3$, Σ_3, and MOTree.
  - $\otimes_2$ is not contained in Vidal.

$\otimes_3$: Third Tensor Class in TSH

- The class of Pure quantum state families that can be written as a tensor product of Σ_2 states.
- Strictly contains Σ_2 and $\otimes_2$, and is strictly contained in $\Sigma_4 \cap \otimes_4$ [Aar04c].

Tree: Tree States

- The class of Pure quantum state families $|\psi_n>$ such that $TS(|\psi_n>) = O(p(n))$ for some polynomial p, where TS, or *tree size*, is defined as follows.

- A *quantum state tree* is a rooted tree where each leaf vertex is labeled with either $|0>$ or $|1>$, and each non-leaf vertex is labeled with either $+$ or $\otimes$. Each vertex v is also labeled with a subset $S(v)$ of $\{1,...,n\}$ such that

  1. If v is a leaf then $|S(v)| = 1$.
  2. If v is the root then $S(v) = \{1,...,n\}$.
  3. If v is a $+$ gate and w is a child of v, then $S(w) = S(v)$.
  4. If v is a $\otimes$ gate and $w_1,...,w_k$ are the children of v, then $S(w_1),...,S(w_k)$ are pairwise disjoint and form a partition of $S(v)$.

- Finally, if v is a $+$ gate, then the outgoing edges of v are labeled with complex numbers. For each v, the subtree rooted at v represents a quantum state in the obvious way (we require this state to be normalized for each v).

- The tree size $TS(|\psi>)$ is then the minimum size of a tree representing $|\psi>$, where size means number of vertices.

- Defined in [Aar04c], where the following was also shown:

  – ψP is not contained in Tree.
  – Tree strictly contains MOTree.
  – Any state family in Tree can be represented by trees of polynomial size and logarithmic depth.
  – Most quantum states cannot even be well-approximated by states of subexponential tree size.

- Contains OTree and TSH, and is contained in Circuit.

TreeSize(f(n)): Pure Quantum States of Tree Size $O(f(n))$

- The class of Pure quantum state families that have $O(f(n))$ tree size in the sense of [Aar04c]. So for example, Tree equals the union over all k of TreeSize(n^k).

- TreeSize($n^{O(\log n)}$) contains Vidal [Aar04c].

TSH: Tensor-Sum Hierarchy

- The class of quantum state families in Tree represented by trees of polynomial size and constant depth, where the depth is just the maximum length of a path from the root to a leaf.

- The levels of TSH are $\otimes_1$, $\otimes_2$, $\otimes_3$, $\cdots$ (corresponding to trees whose root vertex is $\otimes$), and Σ_1, Σ_2, Σ_3, $\cdots$ (corresponding to trees whose root vertex is $+$). (We have $\otimes_0 = \Sigma_0$ = Basis.)

- Whether TSH = Tree and whether TSH is contained in ψP remain intriguing open problems.

Vidal: States of Polynomially-Bounded Schmidt Rank

- The class of Pure quantum state families that are polynomially entangled in the sense of Vidal [Vid03]. More precisely, given a partition of $\{1, ..., n\}$ into A and B, let $\chi_A(|\psi_n>)$ be the minimum k for which $|\psi_n>$ can be written as $\Sigma_{i=1}^{k} \alpha_i |\varphi_i^A> \otimes |\varphi_i^B>$, where $|\varphi_i^A>$ and $|\varphi_i^B>$ are states of qubits in A and B respectively. ($\chi_A(|\psi_n>$ is known as the *Schmidt rank*.) Let $\chi(|\psi_n>) = \max_A \chi_A(\psi_n>)$. Then $|\psi_n>$ is contained in Vidal if and only if $\chi(|\psi_n>$ is at most $p(n)$ for some polynomial p.

- [Vid03] shows that, if a quantum computer is in Vidal at every time step, then it can be simulated classically with only polynomial slowdown.

- [Aar04c] observes the following:

  - Vidal is strictly contained in Circuit and in TreeSize($n^{O(\log n)}$).
  - Vidal strictly contains Σ_2.
  - Vidal does not contain $\otimes_2$.

6 Acknowledgements

I thank Esteve del Acebo, Dorit Aharonov, Eric Allender, Simon Anders, Jesse Andrews, Luis Antues, Albert Atserias, Chris Bourke, Amit Chakrabarti, Hana Chockler, Matthew Cook, Alex Fabrikant, Craig Feinstein, Lance Fortnow, Lee Graham, Xiaoyang Gu, Dan Gutfreund, Satoshi Hada, Mitchell Harris, Julia Kempe, Hirotada Kobayashi, Andrew Landahl, James Lee, Troy Lee, Daniel Marx, Gatis Midrijanis, David Molnar, Cris Moore, Elchanan Mossel, Rana Mourad, Alan Nash, Gabriel Nivasch, Periklis Papakonstantinou, Dan Pless, Chris Pollett, Jibran Rashid, Oded Regev, Hein Rhrig, Mugizi Rwebangira, Cem Say, Philippe Schnoebelen, Barbara Terhal, Luca Trevisan, N. V. Vinodchandran, Qut Walters, John Watrous, Hoeteck Wee, Ryan Williams, Ronald de Wolf, Shengyu Zhang, and Standa Zivny for helpful comments. I especially thank Lane Hemaspaandra and Bodo Manthey for sending me a great deal of relevant information; Bill Gasarch for preparing a summary of communication complexity classes; Lawrence Ip for suggesting the 'table of contents' and pronunciation guide; and Alex Halderman for helping me set up the domain name.

7 Bibliography

References

[Aar02a] Scott Aaronson. Quantum computing and dynamical quantum models. *submitted*, 2002.

[Aar02b] Scott Aaronson. Quantum lower bound for the collision problem. In *STOC '02: Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, pages 635–642. ACM Press, 2002.

[Aar04a] Scott Aaronson. Is quantum mechanics an island in theoryspace? In *Proceedings of Quantum Theory: Reconsideration of Foundations (A. Khrennikov, ed.)*. Vxj University Press, 2004.

[Aar04b] Scott Aaronson. Lower bounds for local search by quantum arguments. In *STOC '04: Proceedings of the thirty-sixth annual ACM symposium on Theory of computing*, pages 465–474. ACM Press, 2004.

[Aar04c] Scott Aaronson. Multilinear formulas and skepticism of quantum computing. In *STOC '04: Proceedings of the thirty-sixth annual ACM symposium on Theory of computing*, pages 118–127. ACM Press, 2004.

[Aar05] Scott Aaronson. Limitations of quantum advice and one-way communication. *Theory of Computing*, 1:1–28, 2005.

[ABFR94] J. Aspnes, R. Beigel, M. L. Furst, and S. Rudich. The expressive power of voting polynomials. *Combinatorica*, 14(2):135–148, 1994.

[ABK+02] E. Allender, H. Buhrman, M. Kouck, D. van Melkebeek, and D. Ronneburger. Power from random strings. In *Proceedings of IEEE FOCS'2002*, pages 669–678, 2002.

[ABL98] Andris Ambainis, David A. Mix Barrington, and Huong LeThanh. On counting AC^0 circuits with negative constants. In *MFCS '98: Proceedings of the 23rd International Symposium on Mathematical Foundations of Computer Science*, pages 409–417. Springer-Verlag, 1998.

[ABO99] Eric Allender, Robert Beals, and Mitsunori Ogihara. The complexity of matrix rank and feasible systems of linear equations. *Computational Complexity 8(2):99-126*, 1999.

[ABV95] William Aiello, Mihir Bellare, and Ramarathnam Venkatesan. Knowledge on the average—perfect, statistical and logarithmic. In *STOC '95: Proceedings of the twenty-seventh annual ACM symposium on Theory of computing*, pages 469–478. ACM Press, 1995.

[ACG+99] G. Ausiello, P. Crescenzi, G. Gambosi, V. Kann, A. Marchetti-Spaccamela, and M. Protasi. *Complexity and Approximation: Combinatorial optimization problems and their approximability properties*. Springer-Verlag, 1999.

[ADH97] Leonard M. Adleman, Jonathan DeMarrais, and Ming-Deh A. Huang. Quantum computability. *SIAM J. Comput.*, 26(5):1524–1540, 1997.

[AFvM01] L. Antuñes, L. Fortnow, and D. van Melkebeek. Computational depth. In *Proceedings of IEEE Complexity'01*, pages 266–273, 2001.

[AG00] Carme Alvarez and Raymond Greenlaw. A compendium of problems complete for symmetric logarithmic space. *Comput. Complex.*, 9(2):123–145, 2000.

[AH87] L. Adleman and M. Huang. Recognizing primes in random polynomial time. In *STOC '87: Proceedings of the nineteenth annual ACM conference on Theory of computing*, pages 462–469. ACM Press, 1987.

[AH91] William Aiello and Johan Hastad. Statistical zero-knowledge languages can be recognized in two rounds. *J. Comput. Syst. Sci.*, 42(3):327–345, 1991.

[AIK04] B. Applebaum, Y. Ishai, and E. Kushilevitz. Cryptography in NC^0. In *Proceedings of IEEE FOCS 2004*, 2004.

[AJ93] Carme Alvarez and Birgit Jenner. A very hard log-space counting class. *Theor. Comput. Sci.*, 107(1):3–30, 1993.

[AK98] Farid M. Ablayev and Marek Karpinski. On the power of randomized ordered branching programs. *Electronic Colloquium on Computational Complexity (ECCC)*, 5(4), 1998.

[AK02] V. Arvind and P. Kurur. Graph isomorphism is in SPP. *Electronic Colloquium on Computational Complexity (ECCC)*, TR02-037, 2002.

[AKL+79] R. Aleliunas, R. M. Karp, R. J. Lipton, L. Lovsz, and C. Rackoff. Random walks, traversal sequences, and the complexity of maze problems. In *Proceedings of IEEE FOCS'79*, pages 218–223, 1979.

[AKS02] M. Agrawal, N. Kayal, and N. Saxena. Primes is in p, 2002.

[AKSS95] V. Arvind, J. Köbler, U. Schöning, and R. Schuler. If NP has polynomial-size circuits, then $\mathsf{MA} = \mathsf{AM}$. *Theoretical Computer Science*, 137, 1995.

[All96] E. Allender. Circuit complexity before the dawn of the new millennium. In *Proceedings of the 16th Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS) Lecture Notes in Computer Science 1180*, pages 1–18, 1996.

[All99] E. Allender. The permanent requires large uniform threshold circuits. *Chicago Journal of Theoretical Computer Science* 7, 1999.

[ALM+98] Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. Proof verification and the hardness of approximation problems. *J. ACM*, 45(3):501–555, 1998.

[AMP02] F. Ablayev, C. Moore, and C. Pollett. Quantum and stochastic branching programs of bounded width. In *Proceedings of the International Colloquium on Automata, Languages, and Programming (ICALP)*, 2002.

[AN] D. Aharonov and T. Naveh. Quantum NP - a survey.

[AR88] E. Allender and R. Rubinstein. P-printable sets. *SIAM Journal on Computing*, 17(6):1193–1202, 1988.

[AR01] M. Alekhnovich and A. Razborov. Resolution is not automatizable unless W[P] is tractable. In *FOCS '01: Proceedings of the 42nd IEEE symposium on Foundations of Computer Science*, page 210. IEEE Computer Society, 2001.

[AR03] D. Aharonov and O. Regev. A lattice problem in quantum np. *Submitted*, 2003.

[Aro96] S. Arora. Polynomial time approximation schemes for euclidean tsp and other geometric problems. In *FOCS '96: Proceedings of the 37th Annual Symposium on Foundations of Computer Science*, page 2. IEEE Computer Society, 1996.

[ARZ99] Eric Allender, Klaus Reinhardt, and Shiyu Zhou. Isolation, matching, and counting uniform and nonuniform upper bounds. *J. Comput. Syst. Sci.*, 59(2):164–181, 1999.

[AS94] E. Allender and M. Strauss. Measure on small complexity classes with applications for bpp. In *Proceedings of IEEE FOCS'94*, pages 807–818, 1994.

[AS98] Sanjeev Arora and Shmuel Safra. Probabilistic checking of proofs: a new characterization of NP. *J. ACM*, 45(1):70–122, 1998.

[ASV00] Andris Ambainis, Leonard J. Schulman, and Umesh V. Vazirani. Computing with highly mixed states (extended abstract). In *STOC '00: Proceedings of the thirty-second annual ACM symposium on Theory of computing*, pages 697–704. ACM Press, 2000.

[ATSWZ00] R. Armoni, A. Ta-Shma, A. Wigderson, and S. Zhou. An $O(\log^{4/3}(n))$ algorithm for (s,t) connectivity in undirected graphs. *Journal of the ACM*, 47(2):294–311, 2000.

[AW90] E. Allender and K. W. Wagner. Counting hierarchies: polynomial time and constant depth circuits, 1990.

[Bar89] D. A. M. Barrington. Bounded-width polynomial-size branching programs can recognize exactly those languages in NC1. *Journal of Computer and System Sciences*, 38:150–164, 1989.

[Bar02] Boaz Barak. A probabilistic-time hierarchy theorem for "slightly non-uniform" algorithms. In *RANDOM '02: Proceedings of the 6th International Workshop on Randomization and Approximation Techniques*, pages 194–208. Springer-Verlag, 2002.

[Baz95] C. Bazgan. *Ph.D. Thesis*. PhD thesis, INRIA, Orsay, France, 1995.

[BBBV97] Charles H. Bennett, Ethan Bernstein, Gilles Brassard, and Umesh Vazirani. Strengths and weaknesses of quantum computing. *SIAM J. Comput.*, 26(5):1510–1523, 1997.

[BBF98] Richard Beigel, Harry Buhrman, and Lance Fortnow. Np might not be as easy as detecting unique solutions. In *STOC '98: Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pages 203–208. ACM Press, 1998.

[BBS86] J. Balczar, R. Book, and U. Schning. Sparse sets, lowness, and highness. *SIAM Journal on Computing*, 15:739–747, 1986.

[BCD+89] A. Borodin, S. A. Cook, P. W. Dymond, W. L. Ruzzo, and M. Tompa. Two applications of inductive counting for complementation problems. *SIAM J. Comput.*, 18(3):559–578, 1989.

[BCE+95] Paul Beame, Stephen Cook, Jeff Edmonds, Russell Impagliazzo, and Toniann Pitassi. The relative complexity of NP search problems. In *STOC '95: Proceedings of the twenty-seventh annual ACM symposium on Theory of computing*, pages 303–314. ACM Press, 1995.

[BCH86] Paul W Beame, Stephen A Cook, and H James Hoover. Log depth circuits for division and related problems. *SIAM J. Comput.*, 15(4):994–1003, 1986.

[BCP83] A. Borodin, S. A. Cook, and N. Pippenger. Parallel computations for well-endowed rings and space-bounded probabilistic machines. *Information and Control*, 58:113–136, 1983.

[BCSS98] Lenore Blum, Felipe Cucker, Michael Shub, and Steve Smale. *Complexity and real computation*. Springer-Verlag New York, Inc., 1998.

[BDCGL92] S. Ben-David, B. Chor, O. Goldreich, and M. Luby. On the theory of average case complexity. *Journal of Computer and System Sciences*, 44(2):193–219, 1992.

[BDHM92] G. Buntrock, C. Damm, U. Hertrampf, and Ch. Meinel. Structure and importance of logspace-mod-classes. *Mathematical Systems Theory*, 25:223–237, 1992.

[Bei89] R. Beigel. On the relativized power of additional accepting paths. In *Proceedings of IEEE Complexity'89*, pages 216–224, 1989.

[Bei94] R. Beigel. Perceptrons, PP, and the polynomial hierarchy. *Computational Complexity*, 4:339–349, 1994.

[Ber80] L. Berman. The complexity of logical theories. *Theoretical Computer Science*, 11:71–78, 1980.

[BF99] H. Buhrman and L. Fortnow. One-sided versus two-sided randomness. In *Proceedings of the 16th Symposium on Theoretical Aspects of Computer Science (STACS)*, pages 100–109, 1999.

[BFL91] L. Babai, Lance Fortnow, and Carsten Lund. Non-deterministic exponential time has two-prover interactive protocols. *Computational Complexity*, 1:3–40, 1991.

[BFS86] L. Babai, P. Frankl, and J. Simon. Complexity classes in communication complexity theory. In *Proceedings of IEEE FOCS'86*, pages 337–347, 1986.

[BFT98] H. Buhrman, L. Fortnow, and T. Thierauf. Nonrelativizing separations. In *Proceedings of IEEE Complexity'98*, pages 8–12, 1998.

[BG81] C. H. Bennett and J. Gill. Relative to a random oracle A, $\mathsf{P}^A \neq \mathsf{NP}^A \neq \mathsf{coNP}^A$ with probability 1. *SIAM Journal on Computing*, 10(1):96–113, 1981.

[BG82] A. Blass and Y. Gurevich. On the unique satisfiability problem. *Information and Control*, 55(1-3):80–88, 1982.

[BG92] R. Beigel and J. Gill. Counting classes: thresholds, parity, mods, and fewness. *Theoretical Computer Science*, 103(1):3–23, 1992.

[BG94] Mihir Bellare and Shafi Goldwasser. The complexity of decision versus search. *SIAM J. Comput.*, 23(1):97–119, 1994.

[BG98] R. Beigel and J. Goldsmith. Downward separation fails catastrophically for limited nondeterminism classes. *SIAM Journal on Computing*, 17(5):142–1429, 1998.

[BGM03] E. Böhler, C. Glaßer, and D. Meister. Error-bounded probabilistic computations between MA and AM. In *28th International Symposium on Mathematical Foundations of Computer Science*, 2003.

[BGS75] T. Baker, J. Gill, and R. Solovay. Relativizations of the $\mathsf{P} \stackrel{?}{=} \mathsf{NP}$ question. *SIAM Journal on Computing*, 4:431–442, 1975.

[BH77] L. Berman and J. Hartmanis. On isomorphism and density of NP and other complete sets. *SIAM Journal on Computing*, 6:305–322, 1977.

[BH91] S. R. Buss and L. Hay. On truth-table reducibility to SAT. *Information and Computation*, 91(1):86–102, 1991.

[BH97] C. Berg and J. Hästad. On the BPP hierarchy problem. Technical Report TRITA-NA-9702, Royal Institute of Technology, Sweden, 1997.

[BHSF02] A. Ben-Hur, H. T. Siegelmann, and S. Fishman. A theory of complexity for continuous time dynamics. *Journal of Complexity*, 18(1):51–86, 2002.

[BHW89] R. Beigel, L. Hemachandra, and G. Wechsung. On the power of probabilistic polynomial time. In *Proceedings of IEEE Complexity'89*, pages 225–230, 1989.

[BHZ87] R. B. Boppana, J. Hastad, and S. Zachos. Does co-np have short interactive proofs? *Inf. Process. Lett.*, 25(2):127–132, 1987.

[BK89] M. Blum and S. Kanna. Designing programs that check their work. In *STOC '89: Proceedings of the twenty-first annual ACM symposium on Theory of computing*, pages 86–97. ACM Press, 1989.

[BKLM01] David A. Mix Barrington, Peter Kadau, Klaus-Jörn Lange, and Pierre McKenzie. On the complexity of some problems on groups input as multiplication tables. *J. Comput. Syst. Sci.*, 63(2):186–200, 2001.

[BKS95] R. Beigel, M. Kummer, and F. Stephan. Approximable sets. *Information and Computation*, 120(2):304–314, 1995.

[BLMS98] D. A. M. Barrington, C.-J. Lu, P. B. Miltersen, and S. Skyum. Searching constant width mazes captures the AC^0 hierarchy. In *Proceedings of the 1998 Symposium of Theoretical Aspects of Computer Science (STACS'98)*, 1998.

[BLMS99] D. A. M. Barrington, C.-J. Lu, P. B. Miltersen, and S. Skyum. On monotone planar circuits. In *Proceedings of IEEE Complexity'99*, 1999.

[BLS84] R. Book, T. Long, and A. Selman. Quantitative relativizations of complexity classes. *SIAM Journal on Computing*, 13(3):461–487, 1984.

[BM88] L. Babai and Shlomo Moran. Arthur-merlin games: a randomized proof system, and a hierarchy of complexity class. *J. Comput. Syst. Sci.*, 36(2):254–276, 1988.

[BOG03] Michael Ben-Or and Danny Gutfreund. Trading help for interaction in statistical zero-knowledge proofs. *J. Cryptology*, 16(2):95–116, 2003.

[BOGG+90] M. Ben-Or, O. Goldreich, S. Goldwasser, J. Håstad, J. Kilian, S. Micali, and P. Rogaway. Everything provable is provable in zero-knowledge. In *CRYPTO '88: Proceedings on Advances in cryptology*, pages 37–56. Springer-Verlag New York, Inc., 1990.

[BOGKW88] Michael Ben-Or, Shafi Goldwasser, Joe Kilian, and Avi Widgerson. Multi-prover interactive proofs: how to remove intractability. In *STOC '88: Proceedings of the twentieth annual ACM symposium on Theory of computing*, pages 113–131. ACM Press, 1988.

[BOM04] J. Buresh-Oppenheim and T. Morioka. Relativized NP search problems and propositional proof systems. In *Proceedings of IEEE Complexity 2004*, pages 54–67, 2004.

[Boo72] R. Book. On languages accepted in polynomial time. *SIAM Journal on Computing*, 1(4):281–287, 1972.

[Boo74] R. Book. Comparing complexity classes. *Journal of Computer and System Sciences*, 3(9):213–229, 1974.

[Boo94] R. Book. On collapsing the polynomial-time hierarchy. *Information Processing Letters*, 52(5):235–237, 1994.

[Bor77] A. Borodin. On relating time and space to size and depth. *SIAM Journal on Computing*, 6:733–744, 1977.

[Bra77] F.-J. Brandenburg. On one-way auxiliary pushdown automata. In *Proceedings of the Third GI-Conference on Theoretical Computer Science LNCS*, volume 48, pages 132–144. Springer, 1977.

[BRS91] R. Beigel, N. Reingold, and D. A. Spielman. PP is closed under intersection. In *Proceedings of ACM STOC'91*, pages 1–9, 1991.

[BS90] J. Bruck and R. Smolensky. Polynomial threshold functions, AC^0 functions and spectral norms. In *Proceedings of IEEE FOCS'90*, pages 632–641, 1990.

[BT88] D. A. M. Barrington and D. Thrien. Finite monoids and the fine structure of NC1. *Journal of the ACM*, 35(4):941–952, 1988.

[BT04] H. Buhrman and L. Torenvliet. Separating complexity classes using structural properties. In *Proceedings of IEEE Complexity 2004*, pages 130–138, 2004.

[BV97] E. Bernstein and U. Vazirani. Quantum complexity theory. *SIAM Journal on Computing*, 26(5):1411–1473, 1997.

[BvD99] H. Buhrman and W. van Dam. Bounded quantum query complexity. In *Proceedings of IEEE Complexity'99*, pages 149–156, 1999.

[Br00] P. Brgisser. *Completeness and Reduction in Algebraic Complexity Theory*, volume 7. Springer Series in Algorithms and Computation in Mathematics, 2000.

[Cai86] J Y Cai. With probability one, a random oracle separates pspace from the polynomial-time hierarchy. In *STOC '86: Proceedings of the eighteenth annual ACM symposium on Theory of computing*, pages 21–29. ACM Press, 1986.

[Cai01] J.-Y. Cai. $\Sigma_2\mathsf{P}$ is contained in $\mathsf{ZPP}^{\mathsf{NP}}$. In *Proceedings of IEEE FOCS'2001*, pages 620–629, 2001.

[Can96] R. Canetti. More on BPP and the polynomial-time hierarchy. *Information Processing Letters*, 57:237–241, 1996.

[CC93] L. Cai and J. Chen. On fixed-parameter tractability and approximability of NP-hard optimization problems. In *Proceedings of ISTCS'93 - Israel Symposium on Theory of Computing and Systems*, pages 118–126, 1993.

[CC97] L. Cai and J. Chen. On fixed–parameter tractability and approximability of NP optimization problems. *Journal of Computer and System Sciences*, 54(3):465–474, 1997.

[CCG+94] R. Chang, B. Chor, O. Goldreich, J. Hartmanis, J. Håstad, D. Ranjan, and P. Rohatgi. The random oracle hypothesis is false. *Journal of Computer and System Sciences*, 49(1):24–39, 1994.

[CCHO01] J.-Y. Cai, V. Chakaravarthy, L. Hemaspaandra, and M. Ogihara. Some karp-lipton-type theorems based on s2. Technical Report TR-759, University of Rochester Computer Science, November 2001.

[CDL01] A. Chiu, G. Davida, and B. Litow. Division in logspace–uniform nc1. *Theoretical Informatics and Applications*, 35(3):259, 2001.

[CF91] Jin-Yi Cai and Merrick L. Furst. PSPACE survives constant-width bottlenecks. *Int. J. Found. Comput. Sci.*, 2(1):67–76, 1991.

[CFLS93] A. Condon, J. Feigenbaum, C. Lund, and P. Shor. Probabilistically checkable debate systems and approximation algorithms for pspace–hard functions (extended abstract). In *Proceedings of ACM STOC'93*, pages 305–314, 1993.

[CGH+88] Jin-Yi Cai, Thomas Gundermann, Juris Hartmanis, Lane A. Hemachandra, Vivian Sewelson, Klaus Wagner, and Gerd Wechsung. The boolean hierarchy i: structural properties. *SIAM J. Comput.*, 17(6):1232–1252, 1988.

[CGRS04] M. Crasmaru, C. Glaßer, K. W. Regan, and S. Sengupta. A protocol for serializing unique strategies. *submitted*, 2004.

[CH89] Jin-Yi Cai and L. A. Hemachandra. On the power of parity polynomial time. In *Proceedings of the Symposium on Theoretical Aspects of Computer Science (STACS), Lecture Notes in Computer Science 349*, pages 229–240. Springer, 1989.

[CHTW04] R. Cleve, P. Høyer, B. Toner, and J. Watrous. Consequences and limits of nonlocal strategies. In *Proceedings of IEEE Complexity*, pages 236–249, 2004.

[Chu41] A. Church. The calculi of lambda-conversion. *Annals of Mathematical Studies*, 6, 1941.

[CKK+95] F. Cucker, M. Karpinski, P. Koiran, T. Lickteig, and K. Werther. On real turing machines that toss coins. In *Proceedings of ACM STOC'95*, pages 335–342, 1995.

[CKS81] A. K. Chandra, D. C. Kozen, and L. J. Stockmeyer. Alternation. *Journal of the ACM*, 28:114–133, 1981.

[CM01] M. Cryan and P. B. Miltersen. On pseudorandom generators in NC^0. In *Proceedings of the 26th International Symposium on Mathematical Foundations of Computer Science (MFCS)*, pages 272–284, 2001.

[CNS99] Jin-Yi Cai, Ajay Nerurkar, and D. Sivakumar. Hardness and hierarchy theorems for probabilistic quasi-polynomial time. In *STOC '99: Proceedings of the thirty-first annual ACM symposium on Theory of computing*, pages 726–735. ACM Press, 1999.

[Cob64] A. Cobham. The intrinsic computational difficulty of functions. In *Proceedings of the 1964 Congress on Logic, Mathematics and the Methodology of Science*, pages 24–30, 1964.

[Cob66] A. Cobham. The recognition problem for the set of perfect squares. In *Proceedings of the 7th Symposium on Switching and Automata Theory*, pages 78–87, 1966.

[Con92] A. Condon. The complexity of stochastic games. *Information and Computation*, 96(2):203–224, 1992.

[Coo71a] S. A. Cook. Characterizations of pushdown machines in terms of time–bounded computers. *Journal of the ACM*, 18(1):4–18, 1971.

[Coo71b] S. A. Cook. The complexity of theorem-proving procedures. In *Proceedings of ACM STOC'71*, pages 151–158, 1971.

[Coo79] S. A. Cook. Deterministic cfl's are accepted simultaneously in polynomial time and log squared space. In *Proceedings of ACM STOC'79*, pages 338–345, 1979.

[Coo85] S. A. Cook. A taxonomy of problems with fast parallel algorithms. *Information and Control*, 64:2–22, 1985.

[CP95] P. Crescenzi and C. Papadimitriou. Reversible simulation of space–bounded computations. *Theoretical Computer Science*, 143:159–165, 1995.

[CS92] J. Castro and C. Seara. Characterizations of some complexity classes between $\Theta_2\mathsf{P}$ and $\Delta_2\mathsf{P}$. In *Proceedings of STACS 1992*, pages 305–317, 1992.

[CT94] P. Crescenzi and L. Trevisan. An approximation scheme preserving reducibility and its applications. In *Proceedings of 14th Annual Conference on Foundations of Software Technology and Theoretical Computer Computer Science (FSTTCS)*, volume Lecture Notes in Computer Science 880, pages 330–341. Springer-Verlag, 1994.

[CT97] M. Cesati and L. Trevisan. On the efficiency of polynomial time approximation schemes. *Electronic Colloquium on Computational Complexity (ECCC)*, TR97-001, 1997.

[CVKT99] P. Crescenzi, R. Silvestri V. Kann, and L. Trevisan. Structure in approximation classes. *SIAM Journal on Computing*, 28:1759–1782, 1999.

[CW00] R. Cleve and J. Watrous. Fast parallel circuits for the quantum fourier transform. In *Proceedings of IEEE Focs'2000*, pages 526–536, 2000.

[Dam90] C. Damm. Problems complete for $\oplus\mathsf{L}$. *Information Processing Letters*, 36:247–250, 1990.

[DC89] P. W. Dymond and S. Cook. Complexity theory of parallel time and hardware. *Information and Computation*, 80:205–226, 1989.

[Dek76] M. I. Dekhtyar. On the relativization of deterministic and nondeterministic complexity classes. *Mathematical Foundations of Computer Science*, LNCS 45:255–259, 1976.

[DF97] R. G. Downey and M. R. Fellows. Threshold dominating sets and an improved characterization of $\mathsf{W}[2]$. *Theoretical Computer Science*, 189, 1997.

[DF99] R. G. Downey and M. R. Fellows. Parameterized complexity. *Springer-Verlag Monographs in Computer Science*, 1999.

[DFT96] R. G. Downey, M. R. Fellows, and U. Taylor. On the parameteric complexity of relational database queries and a sharper characterization of W[1]. In *Combinatorics, Complexity, and Logic, Proceedings of DMTCS'96*, pages 194–213. Springer–Verlag, 1996.

[dGV02] M. de Graaf and P. Valiant. Comparing EQP and $\mathsf{MOD}_{p^k}\mathsf{P}$ using polynomial degree lower bounds. *CoRR*, 2002.

[DW86] E. Dahlhaus and M. K. Warmuth. Membership for growing context-sensitive grammars is polynomial. *Journal of Computer and System Sciences*, 33:456–472, 1986.

[Edm65] J. Edmonds. Paths, trees, and flowers. *Canadian Journal of Mathematics*, 17(3):449–467, 1965.

[Fag74] R. Fagin. Generalized first-order spectra and polynomial-time recognizable sets. In *Complexity of Computation (R. M. Karp, ed.), SIAM–AMS Proceedings*, volume 7, 1974.

[Fen03] S. Fenner. PP-lowness and a simple definition of AWPP. *Theory of Computing Systems*, 36:199–212, 2003.

[FFK94] S. Fenner, L. Fortnow, and S. Kurtz. Gap–definable counting classes. *Journal of Computer and System Sciences*, 48(1):116–148, 1994.

[FFKL93] S. Fenner, L. Fortnow, S. Kurtz, and L. Li. An oracle builder's toolkit. In *Proceedings of Structure in Complexity Theory*, pages 120–131, 1993.

[FG02] J. Flum and M. Grohe. The parameterized complexity of counting problems. In *Proceedings of IEEE FOCS'2002*, 2002.

[FGHP98] S. Fenner, F. Green, S. Homer, and R. Pruim. Quantum NP is hard for PH. In *Proceedings of the Sixth Italian Conference on Theoretical Computer Science, World–Scientific*, pages 241–252, 1998.

[FGL+91] U. Feige, S. Goldwasser, L. Lovász, S. Safra, and M. Szegedy. Approximating clique is almost NP-complete. In *Proceedings of IEEE FOCS'91*, pages 2–12, 1991.

[FK97] U. Feige and J. Kilian. Limited versus polynomial nondeterminism. *Chicago Journal of Theoretical Computer Science*, 1997.

[For94] L. Fortnow. The role of relativization in complexity theory. *Bulletin of the EATCS 52*, February 1994.

[FR74] M. J. Fischer and M. O. Rabin. Super–exponential complexity of presburger arithmetic, in complexity of computation (r. m. karp, ed.). *SIAM–AMS Symposium on Applied Mathematics*, 1974.

[FR98] L. Fortnow and J. D. Rogers. Complexity limitations on quantum computation. In *Proceedings of IEEE Complexity'98*, pages 202–209, 1998.

[Fri57] R. M. Friedberg. Two recursively enumerable sets of incomparable degrees of unsolvability. In *Proceedings of the National Academy of Sciences*, volume 43, pages 236–238, 1957.

[FRS88] L. Fortnow, J. Rompel, and M. Sipser. On the power of multiprover interactive protocols. In *Proceedings of IEEE Complexity'88*, 1988.

[FS04] L. Fortnow and R. Santhanam. Hierarchy theorems for probabilistic polynomial time. In *Proceedings of IEEE FOCS'2004*, 2004.

[FSS84] M. Furst, J. Saxe, and M. Sipser. Parity, circuits, and the polynomial hierarchy. *Mathematical Systems Theory*, 17:13–27, 1984.

[Gas02] W. I. Gasarch. The $\mathsf{P} \stackrel{?}{=} \mathsf{NP}$ poll. *SIGACT News Complexity Theory Column 36 (L. A. Hemaspaandra, ed.)*, 2002.

[GG66] S. Ginsburg and S. Greibach. Deterministic context free languages. *Information and Control*, 9:203–220, 1966.

[GHP00] F. Green, S. Homer, and C. Pollett. On the complexity of quantum ACC. In *Proceeedings of IEEE Complexity'2000*, pages 250–262, 2000. See also: F. Green, S. Homer, C. Moore, and S. Pollett. Counting, fanout, and the complexity of quantum ACC, quant–ph/0106017, 2001.

[Gil77] J. Gill. Computational complexity of probabilistic turing machines. *SIAM Journal on Computing*, 6(4):675–695, 1977.

[GJ79] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of* NP*-Completeness*. Freeman, 1979.

[GJY87] J. Goldsmith, D. Joseph, and P. Young. Self-reducible, P-selective, near-testable, and P-cheatable sets: the effect of internal structure on the complexity of a set (preliminary abstract). In *Proceedings of IEEE Complexity'87*, 1987.

[GK93] Mikael Goldmann and Marek Karpinski. Simulating threshold circuits by majority circuits. In *STOC '93: Proceedings of the twenty-fifth annual ACM symposium on Theory of computing*, pages 551–560. ACM Press, 1993.

[GKR+95] F. Green, J. Köbler, K. W. Regan, T. Schwentick, and J. Torán. The power of the middle bit of a $\#\mathsf{P}$ function. *Journal of Computer and System Sciences*, 50(3):456–467, 1995.

[GLM96] J. Goldsmith, M. A. Levy, and M. Mundhenk. Limited nondeterminism. *SIGACT News*, 27(2):20–29, 1996.

[GMR89] S. Goldwasser, S. Micali, and C. Rackoff. The knowledge complexity of interactive proof systems. *SIAM Journal on Computing*, 18(1):186–208, 1989.

[GMW91] O. Goldreich, S. Micali, and A. Wigderson. Proofs that yield nothing but their validity or all languages in NP have zero-knowledge proof systems. *Journal of the ACM*, 38(1):691–729, 1991.

[Gol97] O. Goldreich. Notes on levin's theory of average-case complexity. ftp://theory.lcs.mit.edu/pub/people/oded/aver--notes.ps, 1997.

[GP86] L. Goldschlager and I. Parberry. On the construction of parallel computers from various bases of boolean functions. *Theoretical Computer Science*, 43(1):43–58, 1986.

[GP91] O. Goldreich and E. Petrank. Quantifying knowledge complexity. In *Proceedings of IEEE FOCS'91*, pages 59–68, 1991.

[GS86] S. Goldwasser and M. Sipser. Private coins versus public coins in interactive proof systems. In *Proceedings of ACM STOC'86*, pages 58–68, 1986.

[GS88] J. Grollman and A. L. Selman. Complexity measures for public-key cryptosystems. *SIAM Journal on Computing*, 17:309–335, 1988.

[GS90] M. Grigni and M. Sipser. Monotone complexity. In *Proceedings of LMS Workshop on Boolean Function Complexity (M. S. Paterson, ed.)*. Cambridge University Press, 1990.

[GS91] M. Grigni and M. Sipser. Monotone separation of nc1 from logspace. In *Proceedings of IEEE Complexity'91*, pages 294–298, 1991.

[GSSZ03] C. Glaßer, A. L. Selman, S. Sengupta, and L. Zhang. Disjoint NP-pairs. *Electronic Colloquium on Computational Complexity (ECCC)*, TR03–011, 2003.

[GSTS03] D. Gutfreund, R. Shaltiel, and A. Ta-Shma. Uniform hardness vs. randomness tradeoffs for arthur-merlin games. In *Proceedings of IEEE Complexity'2003*, 2003.

[GSV99] O. Goldreich, A. Sahai, and S. Vadhan. Can statistical zero knowledge be made non–interactive? or on the relationship of SZK and NISZK. *Electronic Colloquium on Computational Complexity (ECCC)*, TR99-013, 1999. Abstract appeared in CRYPTO'99.

[GTWB91] R. Gavaldá, L. Torenvliet, O. Watanabe, and J. Balcázar. Generalized kolmogorov complexity in relativized separations. In *Proceedings of MFCS'91 (Mathematical Foundations of Computer Science), Springer–Verlag Lecture Notes in Computer Science*, volume 452, pages 269–276, 1991.

[Gup95] S. Gupta. Closure properties and witness reduction. *Journal of Computer and System Sciences*, 50(3):412–432, 1995.

[Gur87] Y. Gurevich. Complete and incomplete randomized NP problems. In *Proceedings of IEEE FOCS'87*, pages 111–117, 1987.

[Gur89] E. Gurari. *An Introduction to the Theory of Computation*. Computer Science Press, 1989. http://www.cis.ohio--state.edu/~gurari/theory--bk/theory--bk.html.

[GV99] O. Goldreich and S. Vadhan. Comparing entropies in statistical zero-knowledge with applications to the structure of szk. In *Proceedings of IEEE Complexity'99*, pages 54–73, 1999.

[GW96] A. Gál and A. Wigderson. Boolean complexity classes vs. their arithmetic analogs. *Random Structures and Algorithms*, 9:1–13, 1996.

[Hal02] S. Hallgren. Polynomial-time quantum algorithms for pell's equation and the principal ideal problem. In *Proceedings of ACM STOC'2002*, 2002.

[Har78] J. Hartmanis. Feasible computations and provable complexity properties. *SIAM*, 1978.

[Har87a] J. Hartmanis. The collapsing hierarchies. *Bulletin of the EATCS*, 33, October 1987. http://external.nj.nec.com/homepages/fortnow/beatcs/column33.ps.

[Har87b] J. Hartmanis. Sparse complete sets for NP and the optimal collapse of the polynomial hierarchy. *Bulletin of the EATCS*, 32, June 1987. http://external.nj.nec.com/homepages/fortnow/beatcs/column32.ps.

[Hås87] J. Håstad. Computational limitations for small-depth circuits, 1987.

[Hås88] J. Håstad. Oneway permutations in NC^0. *Information Processing Letters*, 26:153–155, 1988.

[HCC+92] J. Hartmanis, R. Chang, S. Chari, D. Ranjan, and P. Rohatgi. Relativization: a revisionistic retrospective. *Bulletin of the EATCS*, 47, June 1992. http://external.nj.nec.com/homepages/fortnow/beatcs/column47.ps.

[HCKM88] J. Hartmanis, R. Chang, J. Kadin, and S. G. Mitchell. Some observations about relativization of space bounded computations. *Bulletin of the EATCS*, 35, June 1988. http://external.nj.nec.com/homepages/fortnow/beatcs/column35.ps.

[Hel84] H. Heller. Relativized polynomial hierarchies extending two levels. *Mathematical Systems Theory*, 17(2):71–84, 1984.

[Hem89] L. Hemachandra. The strong exponential hierarchy collapses. *Journal of Computer and System Sciences*, 39(3):299–322, 1989.

[Her90] U. Hertrampf. Relations among MOD-classes. *Theoretical Computer Science*, 74:325–328, 1990.

[Her97] U. Hertrampf. Acceptance by transformation monoids (with an application to local self-reductions). In *Proceedings of IEEE Complexity'97*, pages 213–224, 1997.

[Hes01] W. Hesse. Division is in uniform TC^0. In *Proceedings of the International Colloquium on Automata, Languages, and Programming (ICALP)*, 2001.

[HH76] J. Hartmanis and J. Hopcroft. Independence results in computer science. *ACM SIGACT News*, 8(4):13–24, 1976.

[HH86] J. Hartmanis and L. Hemachandra. Complexity classes without machines: on complete languages for UP. In *Proceedings of ICALP'86, Springer–Verlag Lecture Notes in Computer Science*, volume 226, pages 123–135, 1986.

[HHH98] E. Hemaspaandra, L. Hemaspaandra, and H. Hempel. What's up with downward collapse: using the easy–hard technique to link boolean and polynomial hierarchy collapses. *SIGACT News*, 29(3):10–22, 1998.

[HHN+95] L. Hemaspaandra, A. Hoene, A. Naik, M. Ogihara, A. Selman, T. Thierauf, and J. Wang. Nondeterministically selective sets. *International Journal of Foundations of Computer Science (IJFCS)*, 6(4):403–416, 1995.

[HHR97] E. Hemaspaandra, L. Hemaspaandra, and J. Rothe. Exact analysis of dodgson elections: Lewis carroll's 1876 voting system is complete for parallel access to NP. In *Proceedings of ICALP'97, Springer-Verlag Lecture Notes in Computer Science*, 1997.

[HHT97] Y. Han, L. Hemaspaandra, and T. Thierauf. Threshold computation and cryptographic security. *SIAM Journal on Computing*, 26(1):59–78, 1997.

[HI02] W. Hesse and N. Immerman. Complete problems for dynamic complexity classes. In *Proceedings of Logic in Computer Science (LICS)*, 2002.

[HIS65] J. Hartmanis, P. L. Lewis II, and R. E. Stearns. Hierarchies of memory-limited computations. In *Proceedings of the 6th Annual IEEE Symposium on Switching Circuit Theory and Logic Design*, pages 179–190, 1965.

[HJV93] L. Hemaspaandra, R. Jain, and N. K. Vereshchagin. Banishing robust turing completeness. *International Journal of Foundations of Computer Science*, 3-4:245–265, 1993.

[HMP+93] A. Hajnal, W. Maass, P. Pudlák, M. Szegedy, and G. Turán. Threshold circuits of bounded depth. *Journal of Computer and System Sciences*, 46(2):129–154, 1993.

[HNOS96] L. Hemaspaandra, A. Naik, M. Ogihara, and A. Selman. Computing solutions uniquely collapses the polynomial hierarchy. *SIAM Journal on Computing*, 25(4):697–708, 1996.

[HO02] L. Hemaspaandra and M. Ogihara. *The Complexity Theory Companion*. Springer-Verlag, 2002. See also http://www.cs.rochester.edu/u/lane/=companion/.

[HPV77] J. Hopcroft, W. Paul, and L. Valiant. On time versus space. *Journal of the ACM*, 24(2):332–337, 1977.

[HRV00] U. Hertrampf, S. Reith, and H. Vollmer. A note on closure properties of logspace MOD classes. *Information Processing Letters*, 75(3):91–93, 2000.

[HS65] J. Hartmanis and R. E. Stearns. On the computational complexity of algorithms. *Transactions of the AMS*, 117:285–305, 1965.

[HS92] S. Homer and A. L. Selman. Oracles for structural properties: the isomorphism problem and public-key cryptography. *Journal of Computer and System Sciences*, 44(2):287–301, 1992.

[HT03] C. M. Homan and M. Thakur. One-way permutations and self-witnessing languages. Technical report, Rochester, 2003. http://www.cs.rochester.edu/trs/theory--trs.html.

[HY84] J. Hartmanis and Y. Yesha. Computation times of NP sets of different densities. *Theoretical Computer Science*, 34:17–32, 1984.

[Iba72] O. Ibarra. A note concerning nondeterministic tape complexities. *Journal of the ACM*, 4:608–612, 1972.

[IKW01] R. Impagliazzo, V. Kabanets, and A. Wigderson. In search of an easy witness: exponential time vs. probabilistic polynomial time. In *Proceedings of IEEE Complexity'01*, 2001.

[IL90] R. Impagliazzo and L. A. Levin. No better ways to generate hard NP instances than picking uniformly at random. In *Proceedings of IEEE FOCS'90*, pages 812–821, 1990.

[Imm88] N. Immerman. Nondeterministic space is closed under complement. *SIAM Journal on Computing*, 17:935–938, 1988.

[Imm98] N. Immerman. *Descriptive Complexity*. Springer Graduate Texts in Computer Science, 1998.

[Imp02] R. Impagliazzo. Hardness as randomness: a survey of universal derandomization. In *Proceedings of the ICM, Beijing*, volume 3, pages 649–658, 2002.

[IN96] R. Impagliazzo and M. Naor. Efficient cryptographic schemes provably as secure as subset sum. *Journal of Cryptology*, 9(4):199–216, 1996.

[Ins00] Clay Mathematics Institute. The P versus NP problem (a millennium prize problem), with official problem description by s. cook. http://www.claymath.org/prizeproblems/pvsnp.htm, 2000.

[IPZ01] R. Impagliazzo, R. Paturi, and F. Zane. Which problems have strongly exponential complexity. *Journal of Computer and System Sciences*, 63(4):512–530, 2001. http://cm.bell--labs.com/cm/ms/who/francis/papers/focs98--subexp.pdf.

[IW97] R. Impagliazzo and A. Wigderson. P = BPP if E requires exponential circuits: derandomizing the XOR lemma. In *Proceedings of ACM STOC'97*, pages 220–229, 1997.

[JKS02] J. C. Jackson, A. R. Klivans, and R. A. Servedio. Learnability beyond AC^0. In *Proceedings of ACM STOC'2002*, pages 776–784, 2002.

[JL95] D. W. Juedes and J. H. Lutz. The complexity and distribution of hard problems. *SIAM Journal on Computing*, 24(2):279–295, 1995.

[Joh90] David S. Johnson. *A catalog of complexity classes*, pages 67–161. MIT Press, 1990.

[JPY88] D. S. Johnson, C. H. Papadimitriou, and M. Yannakakis. How easy is local search? *Journal of Computer and System Sciences*, 37:79–100, 1988.

[JSV01] M. Jerrum, A. Sinclair, and E. Vigoda. A polynomial–time approximation algorithm for the permanent of a matrix with non-negative entries. In *Proceedings of ACM STOC'2001*, pages 712–721, 2001.

[Jun85] H. Jung. On probabilistic time and space. In *Proceedings of 12th International Colloquium on Automata, Languages, and Programming (ICALP), Lecture Notes in Computer Science*, volume 194, pages 310–317, 1985.

[JW04] M. Jerrum and U. Wagner. Counting, sampling, and integrating: Algorithms and complexity. http://www.dcs.ed.ac.uk/home/mrj/ETHbook/chap3.ps, 2004.

[JWB] D. Janzing, P. Wocjan, and T. Beth. Cooling and low energy state preparation for 3-local hamiltonians are FQMA-complete.

[JY88] D. S. Johnson and M. Yannakakis. On generating all maximal independent sets. *Information Processing Letters*, 27(3):119–123, 1988.

[Kad88] J. Kadin. The polynomial time hierarchy collapses if the boolean hierarchy collapses. *SIAM Journal on Computing*, 17:1263–1282, 1988.

[Kan82] R. Kannan. Circuit–size lower bounds and non–reducibility to sparse sets. *Information and Control*, 55:40–56, 1982.

[Kar72] R. M. Karp. *Complexity of Computer Computations (J. W. Thatcher and R. E. Miller, eds.)*, chapter Reducibility among combinatorial problems. Plenum Press, 1972.

[Kar86] H. Karloff. A las vegas algorithm for maximum matching. *Combinatorica*, 6:387–392, 1986.

[KF84] C. M. R. Kintala and P. Fischer. Refining nondeterminism in relativized complexity classes. *SIAM Journal on Computing*, 13:329–337, 1984.

[Kha79] L. G. Khachiyan. A polynomial algorithm for linear programming. *Soviet Math Doklady*, 20:191–194, 1979.

[Kha93] M. Kharitonov. Cryptographic hardness of distribution-specific learning. In *Proceedings of ACM STOC'93*, pages 372–381, 1993.

[KI02] V. Kabanets and R. Impagliazzo. Derandomizing polynomial identity tests means proving circuit lower bounds. *Electronic Colloquium on Computational Complexity (ECCC)*, TR02-055, 2002.

[KL82] R. M. Karp and R. J. Lipton. Turing machines that take advice. *Enseign. Math.*, 28:191–201, 1982.

[Kle71] S. C. Kleene. *Introduction to Metamathematics*. Elsevier, 1971.

[KM02] H. Kobayashi and K. Matsumoto. Quantum multi-prover interactive proof systems with limited prior entanglement. In *Proceedings of ISAAC'2002*, pages 115–127, 2002.

[KMSV99] S. Khanna, R. Motwani, M. Sudan, and U. Vazirani. On syntactic versus computational views of approximability. *SIAM Journal on Computing*, 28:164–191, 1999.

[KMY01] H. Kobayashi, K. Matsumoto, and T. Yamakami. Quantum certificate verification: single versus multiple quantum certificates. *quant-ph/0110006*, 2001.

[Ko82] K. Ko. Some observations on the probabilistic algorithms and NP-hard problems. *Information Processing Letters*, 14(1):39–43, 1982.

[Ko85] K. Ko. On some natural complete operators. *Theoretical Computer Science*, 37(1):1–30, 1985.

[Köb89] J. Köbler. *Strukturelle Komplexität von Anzahlproblemen*. PhD thesis, Universität Stuttgart, 1989.

[Kob02] H. Kobayashi. Non-interactive quantum statistical and perfect zero-knowledge. *quant-ph/0207158*, 2002.

[Koi96] P. Koiran. Hilbert's nullstellensatz is in the polynomial hierarchy. *Journal of Complexity*, 12(4):273–286, 1996.

[Koz92] D. C. Kozen. On the myhill-nerode theorem for trees. *Bulletin of the EATCS*, 47:170–173, 1992.

[Koz97] D. C. Kozen. *Automata and Computability*. Springer-Verlag, 1997.

[KR03] J. Kempe and O. Regev. 3-local hamiltonian is QMA-complete. *Quantum Information & Computation (QIC)*, 3(3):258–264, 2003.

[KRC00] V. Kabanets, C. Rackoff, and S. A. Cook. Efficiently approximable real–valued functions. *Electronic Colloquium on Computational Complexity (ECCC)*, TR00-034, 2000.

[Kre88] M. Krentel. The complexity of optimization problems. *Journal of Computer and System Sciences*, 36:490–509, 1988.

[KRS90] C. P. Kruskal, L. Rudolph, and M. Snir. A complexity theory of efficient parallel algorithms. *Theoretical Computer Science*, 71:95–132, 1990.

[KSTT89a] J. Köbler, U. Schöning, S. Toda, and J. Torán. On counting and approximation. *Acta Informatica*, 26:363–379, 1989.

[KSTT89b] J. Köbler, U. Schöning, S. Toda, and J. Torán. Turing machines with few accepting computations and low sets for PP. In *Proceedings of IEEE Complexity'89*, pages 208–215, 1989.

[KSTT92] J. Köbler, U. Schöning, S. Toda, and J. Torán. Graph isomorphism is low for PP. *Computational Complexity*, 2:301–330, 1992.

[KSTT93] J. Köbler, U. Schöning, S. Toda, and J. Torán. *The Graph Isomorphism Problem: Its Structural Complexity*. Birkhäuser, 1993.

[KSV02] A. Kitaev, A. Shen, and M. N. Vyalyi. Classical and quantum computation. *American Mathematical Society*, 2002.

[KT94] P. G. Koliatis and M. N. Thakur. Logical definability of NP optimization problems. *Information and Computation*, 115:321–353, 1994.

[KT96] J. Köbler and S. Toda. On the power of generalized mod–classes. *Mathematical Systems Theory*, 29(1):33–46, 1996.

[Kur64] S. Y. Kuroda. Classes of languages and linear-bounded automata. *Information and Control*, 7:207–233, 1964.

[Kur83] S. Kurtz. On the random oracle hypothesis. *Information and Control*, 57:40–47, 1983.

[KUW86] R. Karp, E. Upfal, and A. Wigderson. Constructing a perfect matching is in random NC. *Combinatorica 6*, pages 35–48, 1986.

[KV88] M. Karpinski and R. Verbeek. Randomness, provability, and the separation of monte carlo time and space. *Lecture Notes in Computer Science 270*, pages 189–207, 1988.

[KV94] M. Kearns and L. Valiant. Cryptographic limitations on learning boolean formulae and finite automata. *Journal of the ACM*, 41(1):67–95, 1994.

[KvM99] A. Klivans and D. van Melkebeek. Graph nonisomorphism has subexponential size proofs unless the polynomial hierarchy collapses. In *Proceedings of ACM STOC'99*, pages 659–667, 1999.

[KW88] M. Karchmer and A. Wigderson. Monotone circuits for connectivity require superlogarithmic depth. In *Proceedings of ACM STOC'88*, pages 539–550, 1988.

[KW93] M. Karchmer and A. Wigderson. On span programs. In *Proceedings of IEEE Complexity'93*, pages 102–111, 1993.

[KW98] J. Kbler and O. Watanabe. New collapse consequences of NP having small circuits. *SIAM Journal on Computing*, 28(1):311–324, 1998.

[KW00] A. Kitaev and J. Watrous. Parallelization, amplification, and exponential time simulation of quantum interactive proof systems. In *Proceedings of ACM STOC'2000*, pages 608–617, 2000.

[Lad75] R. Ladner. On the structure of polynomial time reducibility. *Journal of the ACM*, 22:155–171, 1975.

[Lau83] C. Lautemann. BPP and the polynomial time hierarchy. *Information Processing Letters*, 17:215–218, 1983.

[Lee02] T. Lee. Arithmetical definability over finite structures. Technical Report PP-2002-04, University of Amsterdam, 2002.

[Lev73] L. A. Levin. Universal search problems. *Problemy Peredachi Informatsii*, 9(3):265–266, 1973. (in Russian).

[Lev86] L. A. Levin. Average case complete problems. *SIAM Journal on Computing*, 15:285–286, 1986.

[LFKN90] C. Lund, L. Fortnow, H. Karloff, and N. Nisan. Algebraic methods for interactive proofs. In *Proceedings of IEEE FOCS'90*, pages 1–10, 1990.

[Li93] L. Li. *On the Counting Functions*. PhD thesis, University of Chicago, 1993.

[LL76] R. Ladner and N. A. Lynch. Relativization of questions about log space computability. *Mathematical Systems Theory*, 10:19–32, 1976.

[LMN93] N. Linial, Y. Mansour, and N. Nisan. Constant depth circuits, fourier transform, and learnability. *Journal of the ACM*, 40(3):607–620, 1993.

[LMT97] K. Lange, P. McKenzie, and A. Tapp. Reversible space equals deterministic space (extended abstract). In *Proceedings of IEEE FOCS'97*, pages 45–50, 1997.

[LP82] H. R. Lewis and C. H. Papadimitriou. Symmetric space-bounded computation. *Theoretical Computer Science*, 19:161–187, 1982.

[Lut91] J. H. Lutz. An upward measure separation theorem. *Theoretical Computer Science*, 81:127–135, 1991.

[LV97] M. Li and P. Vitnyi. *An Introduction to Kolmogorov Complexity and Its Applications*. Springer, second edition, 1997.

[Mah82] S. R. Mahaney. Sparse complete sets for NP: Solution of a conjecture by berman and hartmanis. *Journal of Computer and System Sciences 25*, pages 130–143, 1982.

[May94a] E. Mayordomo. Almost every set in exponential time is P-bi-immune. *Theoretical Computer Science*, 136(2):487–506, 1994.

[May94b] E. Mayordomo. *Contributions to the study of resource-bounded measure*. PhD thesis, Universitat Politecnica de Catalunya, 1994.

[MC00] C. Moore and J. P. Crutchfield. Quantum automata and quantum grammars. *Theoretical Computer Science*, 237:275–206, 2000.

[Mil76] G. Miller. Riemann's hypothesis and tests for primality. *Journal of Computer and System Sciences*, 13:300–317, 1976.

[Mil92] P. B. Miltersen. Circuit depth relative to a random oracle. *Information Processing Letters*, 42(6):295–298, 1992.

[MN02] C. Moore and M. Nilsson. Parallel quantum computation and quantum codes. *SIAM Journal on Computing*, 31(3):799–815, 2002.

[Mon80] B. Monien. On a subclass of pseudopolynomial problems. In *Mathematical Foundations of Computer Science (MFCS'80), Springer LNCS 88*, pages 414–425, 1980.

[Moo99] C. Moore. Quantum circuits: fanout, parity, and counting. *Electronic Colloquium on Computational Complexity (ECCC)*, TR99-032, 1999.

[Mor01] T. Morioka. Classification of search problems and their definability in bounded arithmetic. Master's thesis, University of Toronto, 2001.

[MP91] N. Megiddo and C. H. Papadimitriou. On total functions, existence theorems, and computational complexity. *Theoretical Computer Science*, 81:317–324, 1991.

[MR95] R. Motwani and P. Raghavan. *Randomized Algorithms*. Cambridge University Press, 1995.

[MS02] K. Mulmuley and M. Sohoni. Geometric complexity theory i: An approach to the P vs. NP and related problems. *SIAM Journal on Computing*, 31(2):496–526, 2002.

[Muc56] A. A. Muchnik. On the unsolvability of the problem of reducibility in the theory of algorithms. *Doklady Akademii Nauk SSSR 108*, pages 194–197, 1956.

[MV99] P. B. Miltersen and N. V. Vinodchandran. Derandomizing arthur-merlin games using hitting sets. In *Proceedings of IEEE FOCS'99*, pages 71–80, 1999.

[MVV87] K. Mulmuley, U. V. Vazirani, and V. V. Vazirani. Matching is as easy as matrix inversion. In *Proceedings of ACM STOC'87*, pages 345–354, 1987.

[MVW99] P. B. Miltersen, N. V. Vinodchandran, and O. Watanabe. Super-polynomial versus half-exponential circuit size in the exponential hierarchy. In *Proceedings of the 5th Annual Conference on Computing and Combinatorics (COCOON'99), Lecture Notes in Computer Science 1627*, pages 210–220. Springer-Verlag, 1999.

[NC00] M. Nielsen and I. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, 2000.

[NHK00] M. Nakanishi, K. Hamaguchi, and T. Kashiwabara. Ordered quantum branching programs are more powerful than ordered probabilistic branching programs under a bounded-width restriction. In *Proceedings of COCOON'2000 (Computing and Combinatorics), Springer–Verlag Lecture Notes in Computer Science 1858*, pages 467–476, 2000.

[Nis92] N. Nisan. RL is contained in SC. In *Proceedings of ACM STOC'92*, pages 619–623, 1992.

[NR97] M. Naor and O. Reingold. Number-theoretic constructions of efficient pseudo-random functions. In *Proceedings of IEEE FOCS'97*, pages 458–467, 1997.

[NR98] R. Niedermeier and P. Rossmanith. Unambiguous computations and locally definable acceptance types. *Theoretical Computer Science*, 194:137–161, 1998.

[NRR01] M. Naor, O. Reingold, and A. Rosen. Pseudo–random functions and factoring. *Electronic Colloquium on Computational Complexity (ECCC)*, TR01-064, 2001.

[NSW92] N. Nisan, E. Szemerdi, and A. Wigderson. Undirected connectivity in $\emptyset(\log^{1.5}(n))$ space. In *Proceedings of IEEE FOCS'92*, pages 24–29, 1992.

[NTS95] N. Nisan and A. Ta-Shma. Symmetric logspace is closed under complement. In *Proceedings of ACM STOC'95*, pages 140–146, 1995.

[NW94] N. Nisan and A. Wigderson. Hardness versus randomness. *Journal of Computer and System Sciences*, 49:149–167, 1994.

[NY03a] H. Nishimura and T. Yamakami. An algorithmic argument for query complexity lower bounds of advised quantum computation. *Quantum Physics*, 2003.

[NY03b] H. Nishimura and T. Yamakami. Polynomial time quantum computation with advice. *Quantum Physics*, 2003.

[Ogi94] M. Ogihara. On serializable languages. *International Journal of Foundations of Computer Science*, 5(3-4):303–318, 1994.

[OH93] M. Ogihara and L. Hemachandra. A complexity theory for feasible closure properties. *Journal of Computer and System Sciences*, 46(3):295–325, 1993.

[Oka96] T. Okamoto. On relationships between statistical zero-knowledge proofs. In *Proceedings of ACM STOC'96*, 1996.

[OKSW94] P. Orponen, K. I. Ko, U. Schöning, and O. Watanabe. Instance complexity. *Journal of the ACM*, 41:96–121, 1994.

[Ost91] R. Ostrovsky. One-way functions, hard on average problems and statistical zero–knowledge proofs. In *Proceedings of IEEE Complexity'91*, pages 51–59, 1991.

[OW93] R. Ostrovsky and A. Wigderson. One-way functions are essential for non–trivial zero–knowledge. In *Proceedings of the 2nd Israel Symposium on Theory of Computing and Systems (ISTCS–93)*, 1993.

[Pap83] C. H. Papadimitriou. Games against nature. In *Proceedings of IEEE FOCS'83*, pages 446–450, 1983.

[Pap90] C. H. Papadimitriou. On graph–theoretic lemmata and complexity classes. In *Proceedings of IEEE FOCS'90*, pages 794–801, 1990.

[Pap94a] C. H. Papadimitriou. *Computational Complexity*. Addison-Wesley, 1994.

[Pap94b] C. H. Papadimitriou. On the complexity of the parity argument and other inefficient proofs of existence. *Journal of Computer and System Sciences*, 48(3):498–532, 1994.

[Pos44] E. L. Post. Recursively enumerable sets of positive integers and their decision problems. *Bulletin of the American Mathematical Society 50*, pages 284–316, 1944.

[PP00] S. Parker and M. B. Plenio. Efficient factorization with a single pure qubit and $\log n$ mixed qubits. *Physical Review Letters*, 85(3049), 2000.

[PPST83] W. J. Paul, N. Pippenger, E. Szemerdi, and W. T. Trotter. On determinism versus nondeterminism and related problems. In *Proceedings of IEEE FOCS'83*, pages 429–438, 1983.

[Pra75] V. R. Pratt. Every prime has a succinct certificate. *SIAM Journal on Computing*, 4:214–220, 1975.

[PV04] A. Pavan and N. V. Vinodchandran, 2004.

[PY84] C. H. Papadimitriou and M. Yannakakis. The complexity of facets (and some facets of complexity). *Journal of Computer and System Sciences*, 28:244–259, 1984.

[PY88] C. H. Papadimitriou and M. Yannakakis. Optimization, approximation, and complexity classes. In *Proceedings of ACM STOC'88*, pages 229–234, 1988.

[PY96] C. H. Papadimitriou and M. Yannakakis. On limited nondeterminism and the complexity of the vc dimension. *Journal of Computer and System Sciences*, 53(2):161–170, 1996.

[PZ83] C. H. Papadimitriou and S. Zachos. Two remarks on the power of counting. In *Proceedings of the 6th GI Conference in Theoretical Computer Science, Lecture Notes in Computer Science*, volume 145, pages 269–276. Springer-Verlag, 1983.

[RA00] K. Reinhardt and E. Allender. Making nondeterminism unambiguous. *SIAM Journal on Computing*, 29:1118–1131, 2000.

[Rab60] M. O. Rabin. Degree of difficulty of computing a function and a partial ordering of recursive sets. Technical Report 2, Hebrew University, 1960.

[Rac82] C. Rackoff. Relativized questions involving probabilistic algorithms. *Journal of the ACM*, 29(1):261–268, 1982.

[Raz85a] A. A. Razborov. A lower bound on the monotone network complexity of the logical permanent. *Mat. Zametky*, 37(6):887–900, 1985. English translation in Russian Mathematical Notes 37, pp. 485–493.

[Raz85b] A. A. Razborov. Lower bounds on the monotone complexity of some boolean functions. *Dokl. Akad. Nauk SSSR*, 281(4):798–801, 1985. English translation in Soviet Math. Dokl. 31:354–357.

[Raz87] A. A. Razborov. Lower bounds for the size of circuits of bounded depth with basis $\{\&, +\}$. *Mathematicheskie Zametki*, 41(4):598–607, 1987. English translation in Math. Notes. USSR 41(4), pp.333–338.

[Raz94] A. A. Razborov. On provably disjoint NP-pairs. *Electronic Colloquium on Computational Complexity (ECCC)*, TR94-006, 1994.

[Reg02] K. Regan. Understanding the mulmuley–sohoni approach to P vs. NP. *Bulletin of the EATCS*, 78, October 2002. http://people.cs.uchicago.edu/~fortnow/beatcs/column78.pdf.

[Rei04] O. Reingold. Undirected *st*-connectivity in log-space. manuscript, 2004. http://www.wisdom.weizmann.ac.il/~reingold/publications/sl.ps.

[RR97] A. A. Razborov and S. Rudich. Natural proofs. *Journal of Computer and System Sciences*, 55(1):24–35, 1997.

[RS98] A. Russell and R. Sundaram. Symmetric alternation captures BPP. *Computational Complexity*, 7(2):152–162, 1998.

[Rub88] R. Rubinstein. *Structural Complexity Classes of Sparse Sets: Intractability, Data Compression, and Printability.* PhD thesis, Northeastern University, 1988.

[Rud97] S. Rudich. Super-bits, demi-bits, and NP/qpoly-natural proofs. In *RANDOM: International Workshop on Randomization and Approximation Techniques in Computer Science, Lecture Notes in Computer Science*. Springer-Verlag, 1997.

[RW01] S. Reith and K. Wagner. On boolean lowness and boolean highness. *Theoretical Computer Science*, 261(2):305–321, 2001.

[Sav70] W. Savitch. Relationships between nondeterministic and deterministic tape complexities. *Journal of Computer and System Sciences*, 4(2):177–192, 1970.

[Sch83] U. Schöning. A low and a high hierarchy within NP. *Journal of Computer and System Sciences*, 27:14–28, 1983.

[Sch03] P. Schnoebelen. Oracle circuits for branching—time model checking. In *Proceedings of ICALP 2003*, pages 790–801, 2003.

[SCPY98] A. De Santis, G. Di Crescenzo, G. Persiano, and M. Yung. Image density is complete for non–interactive szk. In *Proceedings of the 25th International Colloquium on Automata, Languages, and Programming (ICALP), Lecture Notes in Computer Science*, pages 784–795, 1998. (Note: Some results in the paper were later retracted.).

[Sel79] A. Selman. P-selective sets, tally languages, and the behavior of polynomial time reducibilities in NP. *Mathematical Systems Theory*, 13(1):55–65, 1979.

[SF98] H. T. Siegelmann and S. Fishman. Analog computation with dynamical systems. *Physica*, 120D:214, 1998.

[SFM78] Joel I. Seiferas, Michael J. Fischer, and Albert R. Meyer. Separating nondeterministic time complexity classes. *J. ACM*, 25(1):146–167, 1978.

[Sha90] A. Shamir. IP = PSPACE. In *Proceedings of IEEE FOCS'90*, pages 11–15, 1990.

[She59] J. C. Shepherdson. The reduction of two-way automata to one-way automata. *IBM Journal of Research and Development*, 3:198–200, 1959.

[Shi03] Y. Shi. Quantum and classical tradeoffs. manuscript, 2003.

[Sho97] Peter W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM J. Comput.*, 26(5):1484–1509, 1997.

[Sho99] R. A. Shore. *Handbook of Recursion Theory*, chapter The recursively enumerable degrees, pages 169–197. E. Griffor ed, North-Holland, Amsterdam, 1999.

[Sip82] Michael Sipser. On relativization and the existence of complete sets. In *Proceedings of the 9th Colloquium on Automata, Languages and Programming*, pages 523–531. Springer-Verlag, 1982.

[Sip92] Michael Sipser. The history and status of the P versus NP question. In *STOC '92: Proceedings of the twenty-fourth annual ACM symposium on Theory of computing*, pages 603–618. ACM Press, 1992.

[SM02] L. J. Stockmeyer and A. R. Meyer. Cosmological lower bound on the circuit complexity of a small problem in logic. *Journal of the ACM*, 49(6):753–784, 2002.

[Smo87] R. Smolensky. Algebraic methods in the theory of lower bounds for boolean circuit complexity. In *STOC '87: Proceedings of the nineteenth annual ACM conference on Theory of computing*, pages 77–82. ACM Press, 1987.

[SP98] U. Schöning and R. Pruim. *Gems of Theoretical Computer Science*. Springer-Verlag, 1998.

[Špalek03] R. Špalek. Quantum circuits with unbounded fan-out. In *Proceedings of the 20th International Symposium on Theoretical Aspects of Computer Science (STACS'03), LNCS 2607*, pages 234–246, 2003.

[SS77] R. Solovay and V. Strassen. A fast monte-carlo test for primality. *SIAM Journal on Computing*, 6:84–86, 1977.

[SS04] A. Selman and S. Sengupta. Polylogarithmic-round interactive proofs for coNP collapse the exponential hierarchy. In *Proceedings of IEEE Complexity 2004*, pages 82–90, 2004.

[Sto76] L. J. Stockmeyer. The polynomial hierarchy. *Theoretical Computer Science*, 3:1–22, 1976.

[Sto85] L. J. Stockmeyer. On approximation algorithms for #P. *SIAM Journal on Computing*, 14:849–861, 1985.

[Sud78] I. H. Sudborough. On the tape complexity of deterministic context-free languages. *J. ACM*, 25(3):405–414, 1978.

[SV97] A. Sahai and S. Vadhan. A complete promise problem for statistical zero–knowledge. In *Proceedings of IEEE FOCS'97*, 1997.

[SZ95] M. Saks and S. Zhou. RSPACE(s) is contained in DSPACE($s3/2$). In *Proceedings of IEEE FOCS'95*, pages 344–353, 1995.

[Sze87] R. Szelepcsnyi. The method of forcing for nondeterministic automata. *Bulletin of the EATCS*, 33:96–100, 1987.

[Tar89] G. Tardos. Query complexity, or why is it difficult to separate $\mathsf{NP}^A \cap \mathsf{coNP}^A$ from P^A by random oracles A. *Combinatorica*, 9:385–392, 1989.

[Tod89] S. Toda. On the computational power of PP and $\oplus$P. In *Proceedings of IEEE FOCS'89*, pages 514–519, 1989.

[Tor90] Jacobo Torán. Counting the number of solutions. In *Proceedings of 15th Conference on Mathematical Foundations of Computer Science (MFCS)*, pages 121–135. Springer-Verlag Lecture Notes in Computer Science 452, 1990.

[Tor91] Jacobo Torán. Complexity classes defined by counting quantifiers. *J. ACM*, 38(3):752–773, 1991.

[Tor00] Jacobo Torán. On the hardness of graph isomorphism. In *Proceedings of IEEE FOCS'2000*, pages 180–186, 2000.

[Tur36] Alan M. Turing. On computable numbers, with an application to the entscheidungsproblem. *Proceedings of the London Mathematical Society*, 2(42):230–265, 1936.

[TV02] L. Trevisan and S. Vadhan. Pseudorandomness and average-case complexity via uniform reductions. In *Proceedings of CCC'2002*, pages 129–138, 2002.

[Uma98] Christopher Umans. The minimum equivalent dnf problem and shortest implicants. In *FOCS '98: Proceedings of the 39th Annual Symposium on Foundations of Computer Science*, page 556. IEEE Computer Society, 1998.

[Val79a] L. G. Valiant. Completeness classes in algebra. In *STOC '79: Proceedings of the eleventh annual ACM symposium on Theory of computing*, pages 249–261. ACM Press, 1979.

[Val79b] L. G. Valiant. The complexity of computing the permanent. *Theoretical Computer Science*, 8:189–201, 1979.

[Val03] Leslie G. Valiant. Three problems in computer science. *J. ACM*, 50(1):96–99, 2003.

[Var82] M. Vardi. Complexity of relational query languages. In *Proceedings of ACM STOC'82*, pages 137–146, 1982.

[vDHI02] W. van Dam, S. Hallgren, and L. Ip. Quantum algorithms for hidden coset problems. *submitted*, 2002.

[Ven91] H. Venkateswaran. Properties that characterize LOGCFL. *Journal of Computer and System Sciences*, 43(2):380–404, 1991.

[Ver92] N. K. Vereshchagin. On the power of PP. In *Proceedings of IEEE Complexity'92*, pages 138–143, 1992.

[Ver95] N. K. Vereshchagin. Oracle separation of complexity classes and lower bounds for perceptrons solving separation problems. *Izvestiya Mathematics*, 59(6):1103–1122, 1995.

[Vid03] G. Vidal. Efficient classical simulation of slightly entangled quantum computations. *Physical Review Letters*, 91(147902), 2003.

[Vin04a] N. V. Vinodchandran. Counting complexity of solvable group problems. *SIAM Journal on Computing*, 33(4):852–869, 2004.

[Vin04b] N. V. Vinodchandran. A note on the circuit complexity of PP. In *ECCC Technical Report Series TR04-056*, 2004.

[VSBR83] L. G. Valiant, S. Skyum, S. Berkowitz, and C. Rackoff. Fast parallel computation of polynomials using few processors. *SIAM Journal on Computing*, 12(4):641–644, 1983.

[VV85] U. V. Vazirani and V. V. Vazirani. Random polynomial time equals semi-random polynomial time. In *Proceedings of IEEE FOCS'85*, pages 417–428, 1985.

[VV86] L. G. Valiant and V. V. Vazirani. NP is as easy as detecting unique solutions. *Theor. Comput. Sci.*, 47(1):85–93, 1986.

[Vya03] M. Vyalyi. QMA = PP implies that PP contains PH. In *ECCC Technical Report Series TR03-021*, 2003.

[Wag86] Klaus W. Wagner. The complexity of combinatorial problems with succinct input representation. *Acta Inf.*, 23(3):325–356, 1986.

[Wag90] Klaus W. Wagner. Bounded query classes. *SIAM J. Comput.*, 19(5):833–846, 1990.

[Wat87] Osamu Watanabe. Comparison of polynomial time completeness notions. *Theoretical Computer Science*, 53:249–265, 1987.

[Wat99] J. Watrous. Space-bounded quantum complexity. *Journal of Computer and System Sciences*, 59(2):281–326, 1999.

[Wat00] J. Watrous. Succinct quantum proofs for properties of finite groups. In *FOCS '00: Proceedings of the 41st Annual Symposium on Foundations of Computer Science*, page 537. IEEE Computer Society, 2000.

[Wat02a] J. Watrous. Limits on the power of quantum statistical zero-knowledge. In *Proceedings of IEEE FOCS'2002*, 2002.

[Wat02b] J. Watrous. Talk given at msri workshop on quantum algorithms and complexity, September 2002.

[Wat03] John Watrous. Pspace has constant-round quantum interactive proof systems. *Theor. Comput. Sci.*, 292(3):575–588, 2003.

[Wil85] C. Wilson. Relativized circuit complexity. *Journal of Computer and System Sciences*, 31:169–181, 1985.

[Yan81] M. Yannakakis. Algorithms for acyclic database schemas. In *Proceedings of VLDB (Very Large Databases)*, 1981.

[Yao85] Andrew C-C. Yao. Separating the polynomial-time hierarchy by oracles. In *Proc. 26th annual symposium on Foundations of computer science*, pages 1–10. IEEE Press, 1985.

[Yao89] A. C. Yao. Circuits and local computation. In *STOC '89: Proceedings of the twenty-first annual ACM symposium on Theory of computing*, pages 186–196. ACM Press, 1989.

[Yao90] A. C.-C. Yao. Coherent functions and program checkers. In *STOC '90: Proceedings of the twenty-second annual ACM symposium on Theory of computing*, pages 84–94. ACM Press, 1990.

[Yao93] A. C.-C. Yao. Quantum circuit complexity. In *Proceedings of IEEE FOCS'93*, pages 352–361, 1993.

[Zac88] Stathis Zachos. Probabilistic quantifiers and games. *J. Comput. Syst. Sci.*, 36(3):433–451, 1988.

[Zuc91] David Zuckerman. Simulating bpp using a general weak random source. In *Proceedings of the 32nd annual symposium on Foundations of computer science*, pages 79–89. IEEE Computer Society Press, 1991.