Research Statement

Liyu Zhang

I am interested in theoretical computer science and its applications. I believe breakthroughs in theoretical computer science have the potential to affect all of computer science radically.

My Ph.D. dissertation is in computational complexity. This area of research raises fundamental questions, such as the problem of whether P equals NP. I focused on the following topics for my Ph.D. research.

Disjoint NP-Pairs

A disjoint NP-pair is a pair of disjoint, nonempty sets in NP. The study of disjoint NP-pairs is motivated by its connections to secure public-key cryptosystems and to propositional proof complexity. The latter subject is concerned with the complexity of proof systems for propositional logic, which itself is related to the problem of whether NP equals coNP [8]. In a reasonable formulation of public-key cryptosystems [18], the problem of cracking public-key cryptosystems is equivalent to separating certain disjoint NP-pairs. For this reason, answers to questions about the existence of NP-hard or P-inseparable disjoint NP-pairs informs us about the question of whether secure public-key cryptosystems exist or whether public-key cryptosystems can be NP-hard to crack. My dissertation concentrates primarily on disjoint NP-pairs and connections with propositional proof systems.

Razborov [25] was the first to discover a connection between disjoint NP-pairs and propositional proof systems. A proof system is optimal if its proofs of tautologies are not much longer than those of any other proof system. The problem is that we do not know whether optimal proof systems exist. Concerning the study of disjoint NP-pairs, we do not know whether there is a complete disjoint NP-pair. In fact some of the results I proved in Glaßer et al. [11] show that it is very hard to either prove or disprove existence of complete disjoint NP-pairs. Razborov discovered a connection between these problems. He defined a *canonical* disjoint NP-pair for every propositional proof system and showed that the canonical pair of an optimal propositional proof system is complete for the class of all disjoint NP-pairs. Also, it is a known fact [24] that the canonical NP-pair of every propositional proof system is P-separable if and only if the propositional proof system is simulated by an *automatizable* propositional proof system, a concept in the study of automated theorem proving. The recent paper by Glaßer et al. [12], shows that every disjoint NP-pair is equivalent to the canonical NP-pair of some propositional proof system. This is a surprising result. These results display an unexpected connection of existence of complete and/or non-trivial P-separable disjoint NP-pairs to propositional proof complexity and automated theorem proving. Due to this connection, it is interesting to study the degree structure of disjoint NP-pairs, which is also the degree structure of all canonical NP-pairs of

propositional proof systems. In Glaßer et al. [12] I proved that the degree structure of disjoint NP-pairs is "universal" in the sense that every countable distributive lattice can be embedded into it, assuming P-inseparable NP-pairs exist. This result tells us that the degree structure of disjoint NP-pairs is very complicated. (Similar results were shown for degrees of NP-sets by Ambos-Spies [1].)

Another interesting question regarding the relation between disjoint NP-pairs and propositional proof systems is to what extent canonical NP-pairs represent the properties of their associated propositional proof systems, or more precisely, how different can two propositional proof systems be if they have equivalent canonical NP-pairs. In my recent joint work with Glaßer and Selman [14], we show that the strength of canonical NP-pairs only relate to the strength of their corresponding proof systems in a very limited way.

Structural Properties of Complete Sets

One basic problem in computational complexity is what structural properties complete sets of different complexity classes have. For example, are all NP-complete sets dense, meaning that they have exponentially many strings per length? It is important to study such computational structure of complete sets, because they, by reductions of all the sets in the class to the complete sets, represent all of the structure that a class might have. For this reason, the study of structural properties of complete sets gave us a better understanding of the computational power of various complexity classes [5], and also might lead to proofs of separation results in complexity theory (see, for e.g. , Buhrman et al. [4].)

In Glaßer et al. [9, 10] we studied autoreducibility and mitoticity [4, 6, 7]. A set is r-autoreducible if the set reduces to itself under the reduction r without querying on the input string. Intuitively, autoreducible sets contain a certain amount of redundant information. The information about whether a string is in A can be retrieved from the membership of other strings in A. It was known that the Turing-complete (T) sets for many complexity classes are T-autoreducible [2, 4]. Glaßer et al. [9] proved similar results for the much stronger many-one (m) reductions. They showed that many-one complete sets of NP, PSPACE, all levels of the Polynomial Hierarchy, and all levels of the Boolean Hierarchy over NP are many-one autoreducible. The proofs are totally different from those used previously and involve clever applications of the so-called "left-set" technique introduced by Ogiwara and Watanabe [23]. However, the scenarios for the more general truth-table (tt) reductions, which have strength between the many-one reductions and the Turing reductions, remain much unknown and are interesting open problems for further research. In particular, we do not know whether tt-complete sets for NP are tt-autoreducible.

An NP-complete set L is mitotic if it can be partitioned by a set in P into two parts L_1 and L_2 such that L_1 and L_2 are both NP-complete. Notice that the parts L_1 and L_2 have the same information as does the original set. Glaßer et al. [10] proved that complete sets for complexity classes are m-mitotic. This is a surprising result and the proof is anything but straightforward. No direct proof is known. Instead we proved that m-autoreducible implies m-mitotic, and then apply the results described above.

In general, a set is r-mitotic if the set can be easily partitioned into two subsets that are equivalent under the reduction r. One can easily prove that r-mitoticity implies r-autoreducibility for any reduction r. The problem is whether autoreducibility implies mitoticity. Glaßer et al. [10]

showed that autoreducibility and mitoticity coincides for many-one reductions, which are the same reductions we use to define NP-complete sets. This solves an open problem in Buhrman and Torenvliet [5] in a very surprising way as intuitively and by definition, mitoticity seems to imply much more redundancy than autoreducibility. The key idea of the proof is a novel labeling algorithm using log-distances between two strings. The proof is complicated and involves a combinatorial argument. By combining the results in both papers [9, 10], we conclude that every NP-complete set can be easily split into two NP-complete sets. The same holds for many-one complete sets of many other complexity classes also: PSPACE, the Polynomial Hierarchy, EXP, etc. These are the most compelling and unexpected results we obtained.

In the same paper [10] the authors also showed that mitoticity coincides with autoreducibility for 1-*tt* reductions but not for any reduction weaker than 3-*tt*, where 3-*tt* is a weaker reduction than 1-*tt*, which in turn is weaker than the many-one reduction. This left the case for 2-*tt* as an interesting open problem. In a new paper with Glaßer et al. [15], I solved this open problem by proving that 2-*tt* autoreducibility does not imply 2-*tt* mitoticity. This result completes the picture as to for which reductions the two notions, autoreducibility and mitoticity, are or are not equivalent.

Future Work

I would certainly like to continue my research in the area of computational complexity, specifically on disjoint NP-pairs and structural properties of complete sets. Disjoint NP-pairs is a concept that researchers just started studying in complexity theory and provides us with a new perspective of looking at previously studied unsolved problems. Study of structural properties of complete sets has always been an important component of computational complexity research. Results in the area of structural properties of complete sets always bring about a new and better understanding of complexity classes. Both areas are very active and have many interesting and important open problems [13, 7].

I am also interested in applications of theoretical computer science. Computational complexity has close relations with areas such as cryptography [16, 17] or computer security where tools and concepts of computational complexity can and indeed need to be applied. I would like to explore these areas and will look to get applicable results by exploiting ideas and solutions from the computational complexity sector.

Another area I would like to explore as part of my future research is the design and analysis of algorithms. This is an area in theoretical computer science that is directly connected with tangible applications. The rapidly developing field of computer technology and industry gives rise to many practical problems that need algorithms that are grounded in theoretical concepts. As closely related areas, research in algorithm and complexity have collaborated on many new ideas and techniques in the last decades such as randomization [21], derandomization [22, 20, 19], and algebraic methods [3]. I plan to build upon my solid background in complexity theory and my early experience in algorithmic research [27, 26] to eventually work on the frontier of algorithmic research in the near future.

I will certainly enjoy collaborating with experts in all areas of computer science and conduct mutually beneficial research.

References

[1] K. Ambos-Spies. On the structure of the polynomial time degrees of recursive sets. Habilitationsschrift, Zur Erlangung der Venia Legendi Für das Fach Informatik an der Abteilung Informatik der Universität Dortmund, September 1984.

[2] R. Beigel and J. Feigenbaum. On being incoherent without being hard. *Computation Complexity*, 2(1):1–17, 1992.

[3] H. Buhrman, L. Fortnow, and T. Thierauf, editors. *Algebraic Methods in Computational Complexity*, number 04421 in Dagstuhl Seminar Proceedings. IBFI, Schloss Dagstuhl, Germany, 2005.

[4] H. Buhrman, L. Fortnow, D. van Melkebeek, and L. Torenvliet. Using autoreducibility to separate complexity classes. *SIAM Journal on Computing*, 29(5):1497–1520, 2000.

[5] H. Buhrman and L. Torenvliet. On the structure of complete sets. In *Proceedings 9th Structure in Complexity Theory*, pages 118–133, 1994.

[6] H. Buhrman and L. Torenvliet. Separating complexity classes using structural properties. In *Proceedings of the 19th IEEE Conference on Computational Complexity*, pages 130–138, 2004.

[7] H. Buhrman and L. Torenvliet. A Post's program for complexity theory. *Bulleting of the EATCS*, 85:41–51, 2005.

[8] Stephen A. Cook and Robert A. Reckhow. The relative efficiency of propositional proof systems. *The Journal of Symbolic Logic*, 44(1):36–50, 1979.

[9] C. Glaßer, M. Ogihara, A. Pavan, A. L. Selman, and L. Zhang. Autoreducibility, mitoticity, and immunity. In *Proceedings 30th International Symposium on Mathematical Foundations of Computer Science*, volume 3618 of *Lecture Notes in Computer Science*, pages 387–398. Springer-Verlag, 2005.

[10] C. Glaßer, A. Pavan, A. L. Selman, and L. Zhang. Redundancy in complete sets. In *Proceedings 23nd Symposium on Theoretical Aspects of Computer Science*, volume 3884 of *Lecture Notes in Computer Science*, pages 444–454. Springer, 2006.

[11] C. Glaßer, A. Selman, S. Sengupta, and L. Zhang. Disjoint NP-pairs. *SIAM Journal on Computing*, 33(6):1369–1416, 2004.

[12] C. Glaßer, A. Selman, and L. Zhang. Canonical pairs of proof systems and disjoint NP-pairs. In *Proceedings of the 30th International Symposium on Mathematical Foundations of Computer Science*, lecture Notes in Computer Science, 2005.

[13] C. Glaßer, A. Selman, and L. Zhang. Survey of disjoint NP-pairs and relations to propositional proof systems. In O. Goldreich, A. L. Rosenberg, and A. L. Selman, editors, *Theoretical Computer Science - Essays in Memory of Shimon Even*, volume 3895 of *Lecture Notes in Computer Science*. Springer, 2006.

[14] C. Glaßer, A. Selman, and L. Zhang. To what extent canonical NP-pairs represent their propositional proof systems. Work in progress, 2006.

[15] G. Glaßer, A. Selman, S. Travers, and L. Zhang. Non-mitotic sets. Technical Report TR06-090, Electronic Computational Complexity Colloquium, 2006. Also submitted to STACS'06.

[16] O. Goldreich. *Foundations of Cryptography, Volume I Basic Tools*. Cambridge University Press, 2001.

[17] O. Goldreich. *Foundations of Cryptography, Volume II Basic Applications*. Cambridge University Press, 2004.

[18] J. Grollmann and A. Selman. Complexity measures for public-key cryptosystems. *SIAM Journal on Computing*, 17(2):309–335, 1988.

[19] R. Impagliazzo. Hardness as randomness: a survey of universal derandomization. In *CoRR cs.CC/0304040: (2003)*. 2003.

[20] V. Kabanets. Derandomization: A brief overview. *Bulletin of EATCS*, 76:88–103, 2002.

[21] R. Motwani and P. Raghavan. *Randomized Algorithms*. Cambridge University Press, 1995.

[22] N. Nisan and A. Wigderson. Hardness vs randomness. *Journal of Computer and System Sciences*, 49(2):149–167, 1994.

[23] M. Ogiwara and O. Watanabe. On polynomial-time bounded truth-table reducibility of NP sets to sparse sets. *SIAM Journal of Computing*, 20(3):471–483, 1991.

[24] P. Pudlák. On reducibility and symmetry of disjoint NP-pairs. *Theoretical Computer Science*, 295:323–339, 2003.

[25] A. A. Razborov. On provably disjoint NP-pairs. Technical Report TR94-006, Electronic Computational Complexity Colloquium, 1994.

[26] L. Zhang, H. Zhu, and P. Zhang. A simple randomized algorithm for the union of sets problem. *Chinese Journal of Software*, 11(2):1587–1593, Science Press 2000.

[27] P. Zhang and L. Zhang. Dynamic broadcast paging. In *Proceedings 5th International Conference for Young Computer Scientists, part I*, Nanjing, China, August 1999. China Computer Federation, International Academic Publishers.