Research Statement of Yury Lifshits

November 2006

Five-Year Objective: identify, formalize and solve key algorithmic/data structure problems for future web technologies.

I. PhD Research: Algorithms on Compressed Texts

How can one minimize data storage space, without compromising too much on the query processing time? I am addressing this problem using data compression perspective in my PhD thesis. Namely, I have studied what kind of queries could be answered quickly on LZ-compressed files.

My main contributions are an $O(n^3)$ algorithm for compressed equivalence problem and an $O(n^2m)$ algorithm for fully compressed pattern matching, where n and m are sizes of compressed text and compressed pattern [Paper][Slides]. Original lengths might be exponentially longer. These algorithms are both simpler and faster than any of the previous ones. Next, jointly with my coauthors I constructed the first known algorithm for solving window subsequence problem (bounded subsequence matching) for compressed texts [Paper][Slides].

Not all classical problems can be solved in time polynomially depending on the size of compressed input. Jointly with Markus Lohrey we proved that checking whether compressed pattern can be embedded as a subsequence (i.e. with gaps) into compressed text is Θ_2-hard [Paper][Slides]. Moreover, I showed that computing Hamming distance between two LZ-compressed texts is #P-complete which seems to be one of the most surprising results in the field.

Also, I am interested in suggesting new compression schemes. Since classic compression is usually based on periodicity and repetition, with Juhani Karhumäki we try to generalize these notions [Paper][Slides]. We call a word to be tiling periodic if it can be split in a finite number of copies of a partially defined word. For example, the word XXYY is tiling periodic, since it consists of two copies of the partially defined word X_Y. We present the first known algorithm for finding minimal tiling periods.

II. Other Results

Besides PhD research I (1) proved a lower bound on the size of minimal NFA corresponding to a regular expression [Paper]; (2) proved an upper bound on the number of parts in the joint decomposition of k-connected graphs by several cutting k-sets [SpringerLink]; (3) described several formalizations for software protection based on code obfuscation [Paper]; (4) jointly with Dmitri Pavlov we constructed a new algorithm for solving mean payoff games with the best known upper bound on working time among deterministic algorithms [Paper][Slides]; (5) with research group of Danièle Beauquier we proposed the first known algorithm for detection *information leak* for any given pair (system, property) [Slides].

III. Directions for Future Research

Succinct Data Structures. We need methods for query-specific compression: for a given "query problem" to develop a data structure such that: (A) query time is linearly comparable to that for the classical data structures and (B) for some kind of "regular data" (e.g. with low entropy or with compact generating description) the size of our data structure is smaller than the original input size. Tasks: (1) unify existing approaches used in XML/pattern matching/data compression communities, (2) define theoretical measures of compression power of succinct data structures, (3) choose an industrial application and develop succinct data structures in that specific setting.

Large-Scale Filtering. Personal news aggregation works as follows. Every user has a preference profile, every news item has its own description. The filtering problem is to find, say, ten most appropriate news items for every user. In practice we may have at least 10^6 news per day and 10^8 users. Tasks: (1) find fast algorithms for all-to-all filtering problem, (2) suggest data structures for storing profiles and news, (3) study filtering in dynamic settings: with profiles and descriptions quickly evolving in time, (4) describe spam prevention mechanisms for large filtering systems.

Large-Scale Matching. Consider optimal ads (sponsored links) distribution over the huge set of websites. Every website has an audience description and every ad has a target description. We want to choose a single ad for every website in order to maximize the effectiveness ratio displays/clicks. Tasks: (1) state ads distribution as an optimization problem, (2) find algorithms that can approximately solve this problem faster than (#websites)×(#ads), (3) introduce feedback to the model: after every click on any ad we receive some additional knowledge about the world and can use it for improvement of our matching.

Tag Propagation. Consider the web graph of hyperlinks where *some* websites are labelled by keywords, e.g. Del.icio.us tags or DMOZ categories. Can we extend this labelling to the whole web? Tasks: (1) define formulas for tag "propagation", (2) construct a fast algorithm for computing, say, ten most relevant tags of arbitrary website.

Structure Discovery. Recall that *folksonomy* is a set of triples (object, tag, user) where every user sets arbitrary tags to his favorite objects. Here all tags "live on the same level". Can we automatically find the most relevant hierarchy (taxonomy) for them? Tasks: (1) fix a format of tag description and define an optimality criteria for tags taxonomy, (2) construct a fast algorithm for computing optimal taxonomy, (3) study interplay with algorithms for constructing phylogeny tree in bioinformatics.

IV. Research Objectives

My goal is to find and solve algorithmic problems that are central for the future web technologies. I organize my activities in the following seven steps: (1) understand technology trends and choose important challenges, (2) survey theoretical results around, (3) suggest and discuss formalizations for the chosen problems, (4) create a list of open problems, (5) construct algorithms and write papers, (6) perform experimental analysis, (7) organize public promotion for results.

V. Remark on Teaching

I consider teaching as an integral part of research, as a tool for exploring and rapidly learning new topics. As a Russain saying puts it: "I kept explaining, until I myself understood". My teaching principles are: (1) make lectures highly interactive, (2) promote the course and attract large audience, (3) get feedback from everywhere, (4) provide full technical support: course website, slides, electronic notes, video & audio recording, links to original papers. I developed and taught courses in the following areas:

Web: Algorithms for Internet (website), String Algorithms (website).

Cryptography/Security: Obfuscation and Cryptogaphy (website), Modern Problems of Cryptography (website), Program Obfuscation (website).

Mixture of advanced topics: Modern Problems of Theoretical Computer Science (website), Invitation to Computer Science (website).