ISSN 2186-7437

NII Shonan Meeting Report

No. 2023-184

New Frontiers in Locality in Computation

Guy Even
Pierre Fraigniaud
Gregory Schwartzman

March 20 – 23, 2023

National Institute of Informatics
2-1-2 Hitotsubashi, Chiyoda-Ku, Tokyo, Japan

Overview of the Meeting

Locality is a recurring theme in computation. In fact, computation, as defined by Turing, is a sequence of steps each of which is determined by a constant amount of information. The notion of computing based on partial information is common to many fields, such as: property testing, distributed computation, data structures, sublinear algorithms, and streaming algorithms. The goal of this workshop was to explore commonalities in these fields, to improve our understanding of possibilities and limitations of locality in computation, to study reductions and relations between local models of computations, and to assess which practical aspects are captured by various models of local computation.

Locality, as a computational paradigm, has been considered from different perspectives, resulting with a variety of computational models. We briefly review a few such models in an informal manner:

1. In property testing, we are given access to an object via probes, and the goal is to decide whether the object satisfies a specific property or is far from satisfying the property. This decision is based on making as few probes as possible. For example, the object could be a function $f : \{1, \ldots, n\} \to \{1, \ldots, n\}$, and we wish to test whether the function is monotone. Locality is captured in property testing by the fact that the decision is based on very little information. Indeed, the input is accessed via probes that reveal only a very small amount of information (e.g., each probe reveals $f(x)$ on a requested argument x). The challenge in property testing is to understand the effect of the object's size and the precision of the test on the number of probes required to test a given property. This challenge is addressed via algorithms and lower bounds. An algorithm provides an upper bound on the number of probes, and a lower bound tells us how many probes must be made by any algorithm.

2. In distributed computing, there is a network in which every node has a local input. The nodes communicate via the network links in synchronous rounds, and the goal is to compute local outputs that satisfy some common (global) objective. For example, the goal could be to compute a minimum spanning tree in the network. Thus, each node in the end knows which of the links that are incident to it belong to the minimum spanning tree. The goal in distributed algorithms is to design algorithms that minimize the number of communication rounds.

   Locality is captured in distributed computing by the fact that nodes communicate only with their neighbors. The challenge in distributed computing is to understand the effect of the network's size and topology on the number of communication rounds required to complete the computation. This challenge is addressed via algorithms and lower bounds. An algorithm provides an upper bound on the number of rounds, and a lower bound tell us how many rounds must be used by any algorithm.

3. In data structures, we are interested in finding ways to organize data so that operations over this data are fast. For example, one may want to maintain a set of elements subject to operations such as: insertions, deletions, and various types of queries. The goals in data structures is twofold: (i) find data structures that require as little space as possible,

and (ii) support the operations as fast as possible. Locality is captured in data structures by the fact that memory is accessed in blocks. Hence, each read or write to memory does not access a single bit or word but rather a whole block of words. Accessed blocks are stored in a special fast memory called a cache. Accesses to words in the cache are much faster than accessing a word whose block is not in the cache. In fact, access times to slow memory usually dominate the computation time. Hence, it is desirable to pack information so that operations access as few blocks as possible. Indeed, classical data structures such as linked lists and search trees that employ pointers are not efficient with respect to this definition of locality. The challenge in data structures is to understand the tradeoffs between the space requirements and the time it takes to process various operations. This challenge is addressed via algorithms and lower bounds. An algorithm provides a data structure with upper bounds on the running times of operations, while a lower bound sets minimal limits on the space of a data structure or the time required to perform operations.

4. In sublinear algorithms, we are interested in developing algorithms that perform approximate optimization while reading only a miniscule fraction of the input. The interest in sublinear algorithms is motivated by the fact that modern data sets are too big to be entirely read within a reasonable amount of time. For example, consider the task of studying common features in genomes of different species. With such huge data sets, even linear time algorithms may be too slow, hence the need for sublinear algorithms. Locality is captured in sublinear algorithms by the fact that the decision is only based on a miniscule part of the input. The challenge in sublinear algorithms is to minimize the running time and in particular the fraction of the input that needs to be read. This challenge is addressed via algorithms and lower bounds. An algorithm provides an upper bound on the fraction of the input that needs to be read (as well as on the running time), and a lower bound tell us how much of the input must be read by any algorithm.

5. In streaming algorithms, we are allowed to read the input once with a bounded space algorithm. In particular, we cannot save the whole input in our memory. Streaming algorithms occur naturally in communication networks where a great deal of information traverses a switch. Such a switch has limited memory resources, operates under strict performance requirements, and must nevertheless monitor the traffic to detect events such as cyber attacks. Locality is captured in streaming algorithms by the fact that the algorithm has very limited space. Hence the algorithm maintains a reduced or compressed view of the input it read so far. The challenge is to design algorithms that can deduce interesting properties of the input in spite of space limitations. This challenge is addressed via algorithms and lower bounds. An algorithm provides an upper bound on the required space, while a lower bound tells us how much space must be used used by any algorithm.

Fields dealing with locality in computation are rather active, keep growing in popularity year by year (papers in these fields are consistently presented in all major theoretical computer science conferences such as STOC, FOCS, and

SODA). These fields build on the classical field of algorithm design, while taking into account the constraints imposed by various models of local computation. Many new algorithms and algorithmic techniques have been developed recently. These recent results are promising candidates for becoming the foundational basis for the field of local computation.

The meeting brought together researchers from various branches of the dynamic field of local computation where they shared their recent results and insights. The participants of the meeting came from various countries and institutions, specifically from Israel, Japan, Europe and the USA.

The main research topics that were be discussed in the meeting are:

Commonalities and Differences between Local Models of Computation. Common to all these models of local computation is the information bottleneck. In fact, modern data sets are too big to fit in the fast memory of a single computer, and data is often distributed among several computers or locations. The differences between the models is how the algorithm accesses the data: on one extreme, property testing allows the algorithm to choose the probes. On the other extreme, streaming algorithms may read the input only once, and have severe space limitations (which limits what can be remembered about the input read so far).

Algorithmic techniques have been developed in each of the fields mentioned above. In distributed computing, decomposition algorithms have been playing a major role in the development of algorithms. In property testing, a different form of decomposition, Szemeredi's Regularity Lemma, has been employed (e.g., testing for triangle freeness). In sublinear algorithms and streaming algorithms, dimension reduction algorithms and low rank matrix approximations have been employed. This rich and sophisticated set of algorithmic tools are tailored for the model at hand. We wish to explore the possibilities of exporting these techniques between the various models.

Reductions between Models of Local Computation. An ambitious goal is to devise methods that transform an algorithm in one model of local computation to another. Indeed, such relations have been suggested before: for example, how to turn a sublinear algorithm that reads inputs that are "close" to each other into a distributed algorithm. Along these lines, one might ask whether there exist families of distributed algorithms that can be transformed into streaming algorithms. Such reductions are important both for algorithms and lower bounds. Indeed, if we knew of a lower bound in model B and a reduction from model A to model B, then this would imply a lower bound in model A.

New Abstractions of Locality for New Applications. The model of streaming algorithms was devised to capture the computational limits and demands of data network switches. Newer applications that include blockchains (e.g. Bitcoin), autonomous vehicles, and modern cellular networks (e.g. 5G) pose other challenges that need to be distilled. What aspects of these applications are captured by existing models of local computation? Can we suggest other models that capture additional characteristics of newer applications?

The Power of Randomization in Local Computation. In some models, such as property testing and sublinear algorithms, the need for randomness is inherent. In a recent breakthrough, by Václav Rozhoň and Mohsen Ghaffari (reported in Shonan Meeting 162 on Distributed Graph Algorithms), a polylogarithmic round deterministic distributed algorithm was presented for the Maximal Indepedent Set Problem (MIS). This breakthrough almost closes the gap between random and deterministic distributed algorithms for MIS. However, the power of randomness remains an open problem for many problems in different models of local computation. One intriguing example in data structures is whether randomness is crucial for constant time space efficient dictionaries. All designs to date rely on random hash functions. Understanding the power of randomization in the design of local algorithms is of fundamental importance. Any progress in this direction will have significant implications for the field.

Meeting Schedule

Check-in day (March 19th)

- Welcome banquet

Day 1 (March 20th)

- Invited talk by Mikkel Thurop
- Lecture session 1
- Group photo shooting
- Lecture session 2
- Free discussion

Day 2 (March 21st)

- Invited talk by Alkida Balliu
- Lecture session 1
- Lecture session 2
- Free discussion

Day 3 (March 22nd)

- Invited talk by Moni Naor
- Lecture session
- Excursion
- Main banquet

Day 4 (March 23rd)

- Invited talk by Francois Le Gall
- Lecture session
- Wrap up

Overview of Talks

Reconstructing the Tree of Life (Fitting Distances by Tree Metrics)

Mikkel Thurop

We consider the numerical taxonomy problem of fitting an SxS distance matrix D with a tree metric T. Here T is a weighted tree spanning S where the path lengths in T induce a metric on S. If there is a tree metric matching D exactly, then it is easily found. If there is no exact match, then for some k, we

want to minimize the L_k norm of the errors, that is, pick T so as to minimize $|D-T|_k = (\sum_{i,j \in S} |D(i,j) - T(i,j)|^k)^{1/k}$.

This problem was raised in biology in the 1960s for k = 1,2. The biological interpretation is that T represents a possible evolution behind the species in S matching some measured distances in D. Sometimes, it is required that T is an ultrametric, meaning that all species are at the same distance from the root.

An evolutionary tree induces a hierarchical classification of species and this is not just tied to biology. Medicine, ecology and linguistics are just some of the fields where this concept appears, and it is even an integral part of machine learning and data science. Fundamentally, if we can approximate distances with a tree, then they are much easier to reason about: many questions that are NP-hard for general metrics can be answered in linear time on tree metrics. In fact, humans have appreciated hierarchical classifications at least since Plato and Aristotle (350 BC).

The numerical taxonomy problem is important in practice and many heuristics have been proposed. In this talk we will review the basic algorithmic theory, results and techniques, for the problem, including the most recent result from FOCS'21 [Vincent Cohen-Addad et al., 2021]. They paint a varied landscape with big differences between different moments, and with some very nice open problems remaining.

- At STOC'93, Farach, Kannan, and Warnow [Martin Farach et al., 1995] proved that under L_∞, we can find the optimal ultrametric. Almost all other variants of the problem are APX-hard.

- At SODA'96, Agarwala, Bafna, Farach, Paterson, and Thorup [Richa Agarwala et al., 1999] showed that for any norm $L_k, k >= 1$, if the best ultrametric can be α-approximated, then the best tree metric can be 3α-approximated. In particular, this implied a 3-approximation for tree metrics under L_∞.

- At FOCS'05, Ailon and Charikar [Nir Ailon and Moses Charikar, 2011] showed that for any $L_k, k >= 1$, we can get an approximation factor of $O(((\log n)(\log\log n))^{1/k})$ for both tree and ultrametrics. Their paper was focused on the L_1 norm, and they wrote "Determining whether an O(1) approximation can be obtained is a fascinating question".

- At FOCS'21, Cohen-Addad, Das, Kipouridis, Parotsidis, and Thorup [Vincent Cohen-Addad et al., 2021] showed that indeed a constant factor is possible for L_1 for both tree and ultrametrics. This uses the special structure of L_1 in relation to hierarchies.

- The status of L_k is wide open for $1 < k < \infty$. All we know is that the approximation factor is between $\Omega(1)$ and $O((\log n)(\log\log n))$.

Massively Parallel Computation on Embedded Planar Graphs

Jakub Tětek

Many of the classic graph problems cannot be solved in the Massively Parallel Computation setting (MPC) with strongly sublinear space per machine and o(log n) rounds, unless the 1-vs-2 cycles conjecture is false. This is true even on planar graphs. Such problems include, for example, counting connected components, bipartition, minimum spanning tree problem, (approximate) shortest paths, and (approximate) diameter/radius.In this paper, we show a way to get

around this limitation. Specifically, we show that if we have a "nice' (for example, straight-line) embedding of the input graph, all the mentioned problems can be solved with $O(n^{(2/3+\epsilon)})$ space per machine in O(1) rounds. In conjunction with existing algorithms for computing the Delaunay triangulation, our results imply an MPC algorithm for exact Euclidean minimum spanning tree (EMST) that uses $O(n^{(2/3+\epsilon)})$ space per machine and finishes in O(1) rounds. This is the first improvement over a straightforward use of the standard Boruvka's algorithm with the Dauleanay triangulation algorithm of Goodrich [SODA 1997] which results in $\Theta(logn)$ rounds. This also partially negatively answers a question of Andoni, Nikolov, Onak, and Yaroslavtsev [STOC 2014], asking for lower bounds for exact EMST.

Dictionaries et al.

Ioana Bercea

We will briefly discuss recent advancements in dictionary and filter design. A dynamic dictionary is a data structure that maintains sets under insertions and deletions and supports membership queries of the form "is an element x in the set?". A filter performs approximate membership in that it always answers "yes" if the element is in the set and otherwise, it makes an error with probability at most some epsilon. Both dictionaries and filters are fundamental data structures that are employed in data management projects which require a space-efficient representation of and fast access to large datasets. We will focus on how state-of-the-art techniques exploit locality in the word RAM model to obtain space and time gains, and end with some future exciting directions.

Competitive Vertex Recoloring

Boaz Patt-Shamir

Motivated by placement of jobs in physical machines, we introduce and analyze the problem of online recoloring, or online disengagement. In this problem, we are given a set of n weighted vertices and ak-coloring of the vertices (vertices represent jobs, and colors represent physical machines). Edges, representing conflicts between jobs, are inserted in an online fashion. After every edge insertion,the algorithm must output a proper k-coloring of the vertices. The cost of recoloring a vertex is the vertex's weight. Our aim is to minimize the competitive ratio of the algorithm, i.e., the ratio between the cost paid by the online algorithm and the cost paid by an optimal, offline algorithm.We consider a couple of polynomially-solvable coloring variants.Specifically, for 2-coloring bipartite graphs we present an$O(\log n)$-competitive deterministic algorithm and an $\Omega(\log n)$lower bound on the competitive ratio of randomized algorithms. For$(\Delta+1)$-coloring, where Δ is the maximal node degree, wepresent tight bounds of $\Theta(\Delta)$ and $\Theta(\log\Delta)$ on the competitive ratios of deterministic and randomized algorithms,respectively (where Δ denotes the maximum degree). We also consider the fully dynamic case which allows edge deletions as well as insertions. All our algorithms are applicable to the case where vertices are arbitrarily weighted, and all our lower bounds hold even in the uniform weights (unweighted) case.

Fault Tolerant Algorithms on Networks

Mikaël Rabie

In the paper Fault Tolerant Coloring of the Asynchronous Cycle, we introduced a coloring algorithm on rings, bringing Fault and asynchrony to the usual problems we observe. In particular, some usual problems like MIS cannot be solved in this setting, nor 3-coloring. However, 5 colors are enough. The question arising is the following: what problems can be still solved with this new restrictions on our usual models?

Distributed Edge Coloring in Time Polylogarithmic in Δ

Dennis Olivetti

In this talk we consider the distributed $O(\Delta)$-edge coloring problem. We will see how a simple subroutine allowed us to obtain an algorithm that requires just $O(poly \log \Delta + log^* n)$ rounds. This subroutine is about a different problem called semi-matching. We will discuss some open questions about this problem.

Automatic round elimination 2.0: the case of bipartite maximal matching from A to Z

Alkida Balliu

Given a non-trivial problem P, the round elimination technique provides an automatic way to compute a problem P' that is exactly one round easier than P. This technique has been very useful for proving lower bounds in the LOCAL model of distributed computation. However, computing P' is a tricky process, and this has contributed in making the round elimination technique difficult to approach. In this tutorial-like talk we will see a new and much easier procedure for computing P'. As an example, we will show a simple and self-contained proof for the hardness of the bipartite maximal matching problem.

Efficient Asynchronous Unison

Yuval Emek

Introduced in the early 1990s, asynchronous unison (AU) is a fundamental distributed task that serves as the key component in the synchronization of self-stabilizing algorithms, namely, distributed algorithms which are guaranteed to recover from any combination of transient faults. (A nice feature of the AU formulation is that it actually abstracts away much of the "dirty side" of asynchronous self-stabilizing systems.) Specifically, an AU algorithm with runtime T_{AU} and state space S_{AU} provides the means to transform any self-stabilizing synchronous algorithm ALG with runtime T_{ALG} and state space S_{ALG} into an asynchronous self-stabilizing algorithm that simulates ALG whose runtime is$O(T_{ALG} + T_{AU} + D)$and whose state space is$O(S_{AU} \cdot f(S_{ALG}))$,where D is the graph diameter and f is a (polynomial) function that depends on the communication model. In this talk, we will discuss attempts to develop efficient AU algorithms, with the "holy grail" being an AU algorithm whose runtime and state space are $O(D)$.

The sqrt(log n) barrier in MPC

Slobodan Mitrovic

The Massively Parallel Computation (MPC) model is now a standard for abstracting computational capabilities of MapReduce, Hadoop, Spark and similar modern large-scale frameworks. Many important algorithms of locality radius T can be relatively easily executed in O(T) MPC rounds; in particular, a T-round PRAM algorithm can essentially be executed in O(T) MPC rounds in a black-box manner. The central question of MPC investigation is how to solve the corresponding problems in much fewer than O(T) rounds.For a variety of fundamental problems we now have algorithms that require o(T) MPC rounds, even though state-of-the-art distributed and LOCAL algorithms have locality radius T. Depending on the problem of interest and also on the MPC regime, sometimes the corresponding MPC algorithms require O(poly log T) or even only O(1) rounds. In the most challenging MPC regime, so-called sublinear, state-of-the-art results enable solving certain problems in O(sqrt(T)) MPC rounds. On the other hand, for the same problems there are O(log T) MPC round algorithms for the less restrictive MPC regime, the one called linear. These problems include approximate versions of maximum matching, vertex cover, densest subgraph and out-degree orientation, and maximal independent set, all for which holds T = log n. So far, there is no clear evidence whether this disparity between O(log T) and O(sqrt(T)) is due to the nature of problems or due to lack of our ability to develop better tools.In this talk, I will briefly survey state-of-the-art results and provide a high-level overview of the main idea behind the O(sqrt(log n)) algorithms.

Locality-Sensitive Orderings

Arnold Filtser

Chan, Har-Peled, and Jones [2020] recently developed locality-sensitive ordering (LSO), a new tool that allows one to reduce problems in the Euclidean space R^d to the line. LSO is a collection of orderings over the points of a metric space, such that every two points are guaranteed to be "close" in one of the orderings. Later, with Hung Le, we constructed LSO for doubling metrics. Furthermore, we defined two new types of LSO (triangle, left-sided), which allows constructing LSO for many different spaces.In this talk we will discuss the various LSO types, and some applications.

Average Sensitivity of Graph Algorithms

Yuichi Yoshida

Graph algorithms are widely used for decision making and knowledge discovery. To ensure their effectiveness, it is essential that their output remains stable even when subjected to minor perturbations to the input because frequent output changes can result in costly decisions, reduced user trust, and potential safety concerns. We formalize this feature by introducing the notion of average sensitivity of graph algorithms, which is the average earth mover's distance between the output distributions of an algorithm on a graph and its

subgraph obtained by removing an edge, where the average is over the edges removed and the distance between two outputs is the Hamming distance. In this talk, I will survey recent results on average sensitivity of graph algorithms and other related topics.

Graph Exploration with a Clumsy Agent

Yuichi Sudo

Bojko, Gotfryd, Kowalski, and Pajak [MCFS 2022] recently focused on the graph exploration problem by a weak mobile agent, which we call a clumsy agent. Here we use the word "clumsy" in the sense that after the agent moves from a node u to a node v, it does not recognize the edge u,v at v. A clumsy agent cannot easily backtrack from v to u after moving u to v, while a standard or normal agent can easily make backtrack whenever it wants. Thus, in general, it is not easier to design a time- and/or space-efficient algorithm with clumsy agents than with normal agents. Bojko et al. gave a time-optimal graph exploration algorithm for trees by which a single clumsy agent visits all nodes and terminates thereafter. Some of my students and I gave a time-optimal graph exploration algorithm for general graphs very recently. In this talk, I will briefly explain this result and give some open problems.

Data vs Dimension Reduction in High-Dimensional Streams

Robert Krauthgamer

Many streaming algorithms for high-dimensional data rely on metric embeddings to effectively "compress" the input. Assuming for concreteness that the input is a set of n points in R^d, one popular approach, called dimension reduction, is to map the input points to low dimension d' and solve the resulting low-dimensional instance. A second approach, called data reduction, replaces the n input points with a small number n' of points (perhaps with multiplicities), still in R^d. These methods reduce the overall size of the instance from O(nd) to O(nd') or O(n'd). Yet another approach is to map the input into a tree metric, which distorts the distances significantly, but produces an instance that is easy to solve.

Techniques for Dynamic Sparse Graphs

Jacob Holm

In dynamic graph algorithms the general goal is to maintain some structure S related to a graph G under local updates, e.g. edge insertions or deletions. Usually, we want to minimize the time and/or space used, but in some cases our goal is to minimize the number of local changes to S, also known as the recourse. Often, the graphs in question are known to be from a "sparse" family, e.g. planar graphs, H-minor-free graphs, bounded arboricity graphs or bounded treewidth graphs. In such graph classes we can often do significantly better than for general graphs, using techniques such as r-divisions or bounded outdegree

orientations that exploit local properties of these graphs.I'll briefly cover some of these techniques and their limitations, and some ideas for future work.

Challenges in Quantum Distributed Computing

Francois Le Gall

This talk will describe recent results and challenges in quantum distributed computing, i.e., distributed computing where the processors of the network can exchange quantum messages. After explaining the basics of quantum computing, I will survey some recent results, focusing on quantum distributed graph algorithms. I will then describe several important open questions in quantum distributed computing.

Resilience against Mobile Adversaries in CONGEST

Orr Fischer

In this talk, we aim to answer the question of what is the cost of providing resilience versus mobile adversaries in distributed CONGEST networks. We will introduce several tools and techniques from interactive coding, error correction codes, and sketching, and show how to utilize them to obtain resiliency compilers for CONGEST. Based on a joint work with Merav Parter.

List of Participants

- Prof. Guy Even, Tel Aviv University
- Prof. Pierre Fraigniaud, IRIF
- Prof. Gregory Schwartzman, JAIST
- Prof. Artur Czumaj, University of Warwick
- Prof. Bernhard Haeupler, ETH Zurich and CMU
- Prof. Taisuke Izumi, Osaka University
- Prof. Robert Krauthgamer, Weizmann Institute of Science
- Prof. Fabian Kuhn, University of Freiburg
- Prof. Francois Le Gall, Nagoya University
- Prof. Boaz Patt-Shamir, Tel Aviv University
- Prof. Yuichi Yoshida, National Institute of Informatics
- Prof. Yuichi Sudo, Hosei University
- Prof. Yannic Maus, TU Graz
- Mr. Uri Meir, Tel-Aviv University

- Prof. Sebastian Brandt, CISPA Helmholtz Center for Information Security
- Prof. Alkida Balliu, Gran Sasso Science Institute
- Prof. Dennis Olivetti, Gran Sasso Science Institute
- Prof. Yuval Emek, Technion
- Prof. Moni Naor, Weizmann Institute of science
- Dr. Ioana-Oriana Bercea, IT University of Copenhagen
- Prof. Slobodan Mitrovic, UC Davis
- Prof. Ansis Rosmanis, Nagoya University
- Prof. Arnold Filtser, Bar Ilan University
- Dr. Orr Fischer, Weizmann Institute of Science
- Dr. Mikaël Rabie, IRIF - University Paris Cite
- Mr. Jakub Tetek, University of Copenhagen
- Mr. Avinandan Das, IRIF, CNRS
- Prof. Artur Czumaj, University of Warwick
- Prof. Jacob Holm, University of Copenhagen
- Prof. Mikkel Thorup, University of Copenhagen