Int J Software Informatics, Volume 5, Issue 4 (2011), pp. 0–0
International Journal of Software and Informatics, ISSN 1673-7288

E-mail: ijsi@iscas.ac.cn
http://www.ijsi.org
Tel: +86-10-62661040

The Future of Computer Science

John E. Hopcroft, Sucheta Soundarajan, and Liaoruo Wang

(Cornell University, Ithaca NY 14853, USA)

Abstract Computer science is undergoing a fundamental change and is reshaping our understanding of the world. An important aspect of this change is the theory and applications dealing with the gathering and analyzing of large real-world data sets. In this paper, we introduce four research projects in which processing and interpreting large data sets is a central focus. Innovative ways of analyzing such data sets allow us to extract useful information that we would never have obtained from small or synthetic data sets, thus providing us with new insights into the real world.

Key words:

Hopcroft JE, Soundarajan S, Wang L. The future of computer science. *Int J Software Informatics*, **Vol.5, No.4 (2011): 000–000. http://www.ijsi.org/1673-7288/5/i110.htm**

1 Introduction

Modern computer science is undergoing a fundamental change. In the early years of the field, computer scientists were primarily concerned with the size, efficiency and reliability of computers. They attempted to increase the computational speed as well as reduce the physical size of computers, to make them more practical and useful. The research mainly dealt with hardware, programming languages, compilers, operating systems and data bases. Meanwhile, theoretical computer science developed an underlying mathematical foundation to support this research which in turn, led to the creation of automata theory, formal languages, computability and algorithm analysis. Through the efforts of these researchers, computers have shrunk from the size of a room to that of a dime, nearly every modern household has access to the internet and communications across the globe are virtually instantaneous.

Computers can be found everywhere, from satellites hundreds of miles above us to pacemakers inside beating human hearts. The prevalence of computers, together with communication devices and data storage devices, has made vast quantities of data accessible. This data incorporates important information that reveals a closer approximation of the real world and is fundamentally different from what can be extracted from individual entities. Rather than analyzing and interpreting individual messages, we are more interested in understanding the complete set of information from a collective perspective. However, these large-scale data sets are usually far greater than can be processed by traditional means. Thus, future computer science

This research was partially supported by the U.S. Air Force Office of Scientific Research under Grant FA9550-09-1-0675.
Corresponding author: John E. Hopcroft, Email: jeh@cs.cornell.edu
Received 2010-12-05; Revised 2011-05-05; Accepted 2011-05-12.

research and applications will be less concerned with how to make computers work and more focused on the processing and analysis of such large amounts of data.

Consider the following example of internet search. At the beginning of the internet era, users were required to know the IP address of the site to which they wished to connect. No form of search was available. As websites proliferated, online search services became necessary in order to make the internet navigable. The first internet search tool was developed in 1993, and dozens more were created over the next several years. Ask Jeeves, founded in 1996, relied partially on human editors who manually selected the best websites to return for various search queries. Given the huge number of websites available today, such a strategy is clearly no longer feasible. Google, founded in 1998, is a leader among today's search engines. It relies on a search algorithm that uses the structure of the internet to determine the most popular and thus, perhaps, the most reputable websites.

However, while Google's search engine was a major advance in search technology, there will be more significant advances in the future. Consider a user who asks the question, "When was Einstein born?" Instead of returning hundreds of webpages to such a search, one might expect the answer "Einstein was born at Ulm, in Wurttemberg Germany, on March 14, 1879", along with pointers to the source from which the answer was extracted. Other similar searches might be:

- Construct an annotated bibliography on graph theory
- Which are the key papers in theoretical computer science?
- Which car should I buy?
- Where should I go to college?
- How did the field of computer science develop?

Search engine companies have saved billions of search records along with a whole archive of information. When we search for the answer to the question "Which car should I buy?", they can examine pages that other individuals who did similar searches have looked at, extract a list of factors that might be important (e.g. fuel economy, price, crash safety), and prompt the user to rank them. Given these priorities, the search engine will provide a list of automobiles ranked according to the preferences, as well as key specifications and hyperlinks to related articles about the recommended models.

Another interesting question is, "Which are the key papers in theoretical computer science?" One would expect a list such as:

- Juris Hartmanis and Richard Stearns, "On the computational complexity of algorithms"
- Manuel Blum, "A machine-independent theory of the complexity of recursive functions"
- Stephen Cook, "The complexity of theorem proving procedures"
- Richard Karp, "Reducibility among combinatorial problems"

– Andrew Yao, "Theory and applications of trapdoor functions"

– Shafi Goldwasser, Silvio Micali and Charles Rackoff, "The knowledge complexity of interactive proof systems"

– Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan and Mario Szegedy, "Proof verification and the hardness of approximation problems"

With thousands of scientific papers published every year, information on which research areas are growing or declining would be of great help to rank the popularity and predict the evolutionary trend of various research topics. For example, Shaparenko et al. used sophisticated artificial intelligence techniques to cluster papers from the Neural Information Processing Systems (NIPS) conference held between 1987 and 2000 into several groups, as shown in Fig.1. Since all papers presented at the NIPS conference are in digital format, one can use this information to plot the sizes of the clusters over time. Clusters 10 and 11 clearly show the two growing research areas in NIPS, namely "Bayesian methods" and "Kernel methods". The graph correctly indicates that the "Bayesian methods" cluster emerged before the "Kernel methods" cluster, with both topics starting to dominate the NIPS conference by 2000. In addition, Cluster 1 on neural networks, Cluster 4 on supervised neural network training, and Cluster 8 on biologically-inspired neural memories were popular in the early years of NIPS, but almost disappeared from the conference by 2000. With the help of advanced techniques, we should be able to accurately predict how important a paper will be when it is first published, as well as how a research area will evolve and who will be the key players.

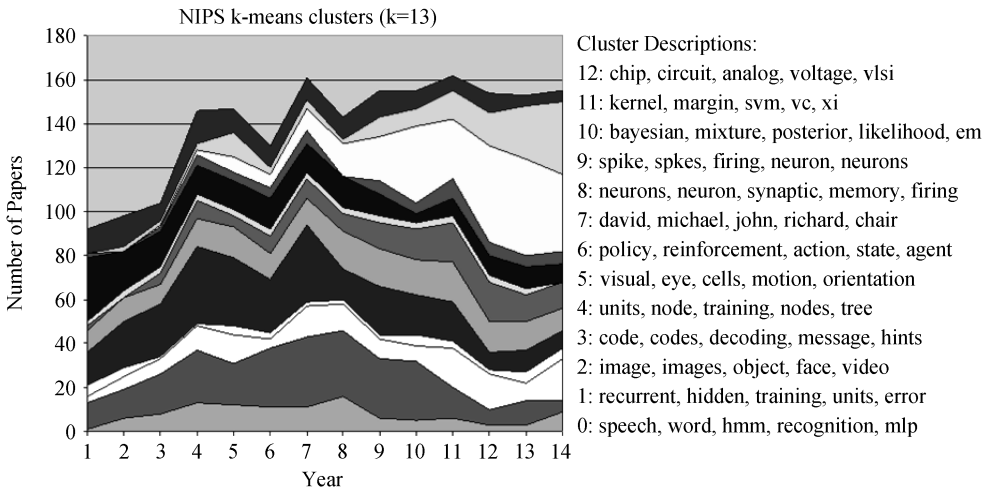

Figure 1. The distribution of $k = 13$ clusters of NIPS papers[1]. The histograms of each cluster are stacked on top of each other to show the influence of cluster popularity over time.

In the beginning years of computer science, researchers established a mathematical foundation consisting of areas such as automata theory and algorithm analysis to support the applications of that time. As applications develop over time, so must the underlying theory. Surprisingly, the intuition and mathematics behind the theory

of large or high-dimensional data are completely different from that of small or low-dimensional data. Heuristics and methods that were effective merely a decade ago may already be outdated.

In this paper, we begin in Section 2 by giving an overview of several ongoing projects dealing with the analysis of large data sets which represent the work currently being done in a wide variety of research areas. In Section 3, we discuss some examples of the theoretical foundation required to rigorously conduct research in these areas, such as large graphs, high-dimensional data, sparse vectors, and so on. We conclude in Section 4 with comments on the problems considered.

2 Innovative Research Projects

Traditional research in theoretical computer science has focused primarily on problems with small input sets. For instance, in the past, stores tracked the items purchased by each individual customer and gave that customer discounts for future purchases of those items. However, with the help of modern algorithms, service providers such as Netflix are now able to, not only make predictions based on a customer's past preferences, but amalgamate preferences from millions of customers to make accurate and intelligent suggestions and effectively increase sales revenue. The following subsections describe four ongoing research projects involving the analysis and interpretation of large data sets. Each represents a promising direction for rediscovering fundamental properties of large-scale networks that will reshape our understanding of the world.

2.1 Tracking communities in social networks

A social network is usually modeled as a graph in which vertices represent entities and edges represent interactions between pairs of entities. In previous studies, a community was often defined to be a subset of vertices that are densely connected internally but sparsely connected to the rest of the network[2−4]. Accordingly, the best community of the graph was typically a peripheral set of vertices barely connected to the rest of the network by a small number of edges. However, it is our view that for large-scale real-world societies, communities, though better connected internally than expected solely by chance, may also be well connected to the rest of the network[5]. It is hard to imagine a small close-knit community with only a few edges connecting it to the outside world. Rather, members of a community, such as a computer science department, are likely to have many connections outside the community, such as family, religious groups, other academic departments and so on, as shown in Fig.2. Empirically, a community displays a higher than average edge to vertex squared ratio which reflects the probability of an edge between two randomly-picked vertices, but can also be connected to the rest of the network by a significant number of edges, which may even be larger than the number of its internal edges.

With this intuitive notion of community, two types of structures are defined: the "whiskers" and the "core". Whiskers are peripheral subsets of vertices that are barely connected to the rest of the network, while the core is the central piece that exclusively contains the type of community we are interested in. Then, the algorithm for finding a community can be reduced to two steps: 1) identifying the core in which no whiskers exist, and 2) identifying communities in the core. Further, extracting

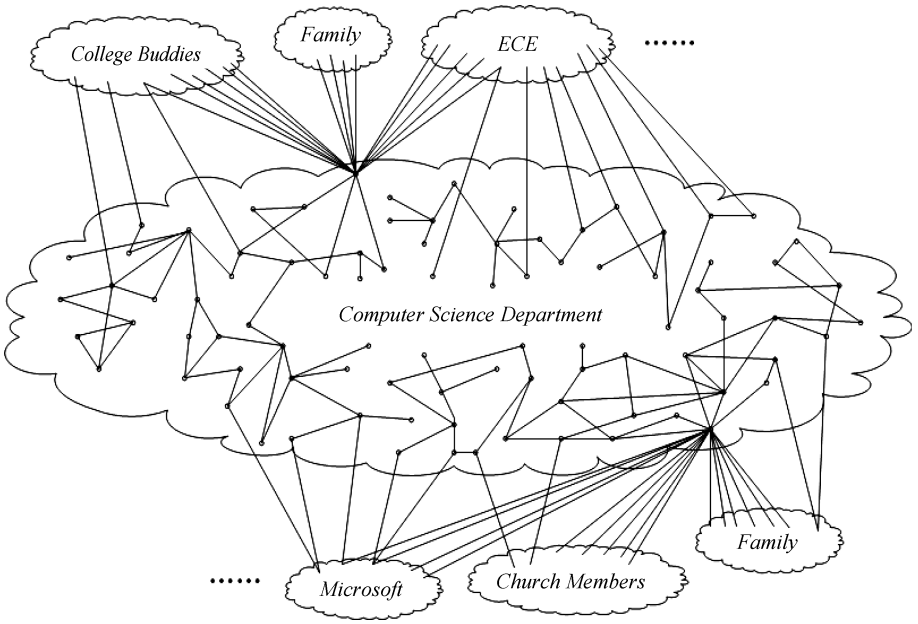

Figure 2. A sample friendship network. Vertices typically have a significant number of cut edges

the exact core from both weighted and unweighted graphs has been proved to be NP-complete. Alternative heuristic algorithms have been developed, all of which are capable of finding an approximate core, and their performance can be justified by the experimental results based on various large-scale social graphs[5]. In this way, one can obtain communities that are not only more densely connected than expected by chance alone, but also well connected to the rest of the network.

Much of the early work on finding communities in social networks focused on partitioning the corresponding graph into disjoint subcomponents[3,4,6−11]. Algorithms often required dense graphs and conductance was widely taken as the measure of the quality of a community[2−3,12−13]. Given a graph $G = (V, E)$, the *conductance* of a subset of vertices $S \subseteq V$ is defined as:

$$\varphi(S) = \frac{\sum_{i \in S, j \notin S} A_{ij}}{\min\left\{\sum_{i \in S, j \in V} A_{ij},\ \sum_{i \notin S, j \in V} A_{ij}\right\}},$$

where $\{A_{ij} | i, j \in V\}$ are the entries of the adjacency matrix for the graph. A subset was considered to be community-like when it had low conductance value. However, as discussed earlier, an individual may belong to multiple communities at the same time and will likely have more connections to individuals outside of his/her community than inside. One approach to finding such well-defined overlapping communities is that of Mishra et al.[14] where the concept of an (α, β)-community was introduced and several algorithms were given for finding an (α, β)-community in a dense graph under certain conditions. Given a graph $G = (V, E)$ with self-loops added to all vertices, a subset $C \subseteq V$ is called an (α, β)-*community* when each vertex in C is connected to at least β vertices of C (self-loop counted) and each vertex outside of C is connected to at most α vertices of C ($\alpha < \beta$).

Among the interesting questions being explored are why (α, β)-communities correspond to well-defined clusters and why there is no sequence of intermediate (α, β)-communities connecting one cluster to another. Other intriguing questions include whether different types of social networks incorporate fundamentally different social structures; what it is about the structure of real-world social networks that leads to the structure of cores, as in the Twitter graph, and why some synthetic networks do not display this structure.

By taking the intersection of a group of massively overlapping (α, β)-communities obtained from repeated experiments, one can eliminate random factors and extract the underlying structure. In social graphs, for large community size k, the (α, β)-communities are well clustered into a small number of disjoint cores, and there are no isolated (α, β)-communities scattered between these densely-clustered cores. The number of cores decreases as k increases and becomes relatively small for large k. The cores obtained for a smaller k either disappear or merge into the cores obtained for a larger k. Moreover, the cores correspond to dense regions of the graph and there are no bridges of intermediate (α, β)-communities connecting one core to another. In contrast, the cores found in several random graph models usually have significant overlap among them, and the number of cores does not necessarily decrease as k increases. Extensive experiments demonstrate that the core structure displayed in various large-scale social graphs is, indeed, due to the underlying social structure of the networks, rather than due to high-degree vertices or to a particular degree distribution[5,15].

This work opens up several new questions about the structure of large-scale social networks, and it demonstrates the successful use of the (α, β)-community algorithm on real-world networks for identifying their underlying social structure. Further, this work inspires an effective way of finding overlapping communities and discovering the underlying core structure from random perturbations. In social graphs, one would not expect a community to have an exact boundary; thus, the vertices inside an (α, β)-community but outside the corresponding core are actually located in the rough boundary regions. Other open questions include how the core structure will evolve, whether the cores correspond to the stable backbones of the network, and whether the vertices that belong to multiple communities at the same time constitute the unstable boundary regions of the network.

2.2 Tracking flow of ideas in scientific literature

Remarkable development in data storage has facilitated the creation of gigantic digital document collections available for searching and downloading. When navigating and seeking information in a digital document collection, the ability to identify topics with their time of appearance and predict their evolution over time, would be of significant help. Before starting research in a specific area, a researcher might quickly survey the area, determine how topics in the area have evolved, locate important ideas, and the papers that introduced those ideas. Knowing a specific topic, a researcher might find out whether it has been discussed in previous papers, or is a fairly new concept. As another example, a funding agency that administers a digital document collection might be interested in visualizing the landscape of topics in the collection to show the emergence and evolution of topics, the bursts of topics, and the

interactions between different topics that change over time.

Such information-seeking activities often require the ability to identify topics with their time of appearance and to follow their evolution. Recently, in their unpublished work, Jo et al. have developed a unique approach to achieving this goal in a time-stamped document collection with an underlying document network which represents a wide range of digital texts available over the internet. Examples are scientific paper collections, text collections associated with social networks such as blogs and Twitter, and more generally, web documents with hyperlinks. A document collection without an explicit network can be converted into this format by connecting textually similar documents to generate a document network.

The approach emphasizes discovering the topology of topic evolution inherent in a corpus. As demonstrated in the above work, the topology inherent in the corpus carries surprisingly rich information about the evolution of topics. Topics, along with the time that they start to appear in the corpus, can be identified by visiting each document in the corpus chronologically and determining if it initiates a topic. A document is considered to initiate a topic if it has a textual content that is not explained by previously discovered topics and persists in a significant number of later documents. After topics are obtained by the chronological scan, an associated graph can be built whose vertices are topics and whose edges reflect cross-citation relations between topics. Globally, this generates a rich topological map showing the landscape of topics over time. Figure 3 shows the results of the work by Jo et al. applying this approach to the ACM corpus. Topics in the network research area emerged in the 1980s without significant ancestors, while the areas of compiler and graphics research exhibit steady evolution with an appreciable number of topics in the early years of the ACM corpus. We can also construct an individual topic evolution graph for a given seed topic, and such a graph may contain multiple threads indicating that the seed topic has been influenced by multiple fields. The relationship between these threads may change over time as well.

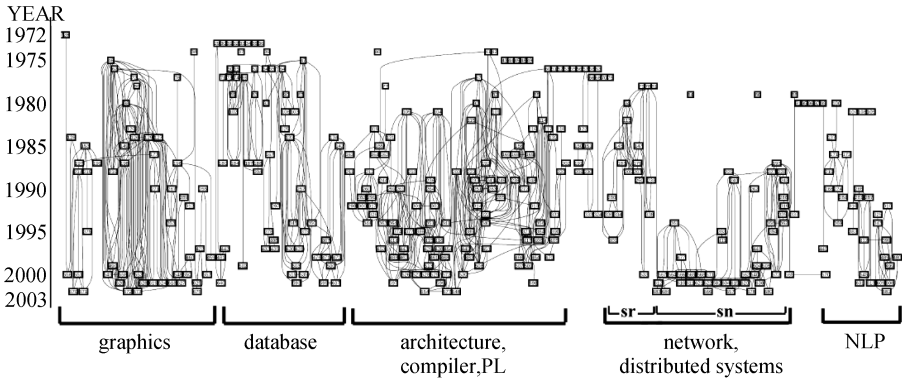

Figure 3. Topic evolution map of the ACM corpus

Related to this research, content-based inferring and learning has been extensively studied recently. Various methods to improve question-answer services in social networks have been proposed[16−19]. In addition, tracking popular events in social communities can be achieved using a statistical model[20].

2.3 Reconstructing networks

The study of large networks has brought about many interesting questions, such as how to determine which members of a population to vaccinate in order to slow the spread of an infection, or where to place a limited number of sensors to detect the flow of a toxin through a water network. Most algorithms for solving such questions make the assumption that the structure of the underlying network is known. For example, detectives may want to use such algorithms to identify the leaders of a criminal network, and to decide which members to turn into informants. Unfortunately, the exact structure of the criminal network cannot be easily determined. However, it is possible that the police department has some information about the spread of a certain property through the network; for instance, some new drug may have first appeared in one neighborhood, and then in two other neighborhoods, and so on. The work by Soundarajan et al.[21] attempts to create algorithms to recover the structure of a network given information about how some property, such as disease or crime, has spread through the network.

This work begins by defining a model of contagion describing how some property has spread through a network. The model of contagion for information spread may be: "a vertex learns a piece of information in the time interval after one of its neighbors learns that information." A more complex model of contagion corresponding to the spread of belief may be: "a vertex adopts a new belief in the time interval after a proportion p of its neighbors adopts that belief." For example, a person will probably not join a political party as soon as one of his friends joins that party, but he may join it after two-thirds of his friends have joined it.

Next, the network recovery algorithm assumes that vertices are partitioned into discrete time intervals, corresponding to the time when they adopt the property. For a given model of contagion, the algorithm attempts to find a network over the set of vertices such that when the property in question (e.g. information, belief) is introduced to some vertices in the first time interval, and then spreads to other vertices in accordance with the model of contagion, every vertex adopts the property at an appropriate time. Initial work has created such algorithms for two models of contagion: the model corresponding to the spread of information, where a vertex adopts a property in the time interval after one of its neighbors has adopted that property, and the model corresponding to the spread of belief, where a vertex adopts a property in the time interval after at least half of its neighbors have adopted that property.

Future work will focus on finding algorithms for other models of contagion, especially the models in which a vertex adopts the property after a proportion p of its neighbors has adopted that property, for arbitrary values of p. Other directions include finding algorithms for networks in which there are two or more properties spreading through the network. This work also opens up questions about the types of graphs produced by these algorithms. For instance, do all possible graphs have some edges in common? Are there any edges that do not appear in any of the solution graphs? Which edges are the most or least likely? Related research in this area includes work by Gomez-Rodriguez et al.[22], which studied information flow and cascades in online blogs and news stories. Work by Leskovec et al.[23] studied the question of how to detect outbreaks or infection in a network. In addition, a more

general problem of link prediction was studied by Clauset et al. in Ref. [24].

2.4 Tracking bird migration in north America

Hidden Markov models (HMMs) assume a generative process for sequential data whereby a sequence of states (i.e. a sample path) is drawn from a Markov chain in a hidden experiment. Each state generates an output symbol from a given alphabet, and these output symbols constitute the sequential data (i.e. observations). The classic single path problem, solved by the Viterbi algorithm, is to find the most probable sample path given certain observations for a given Markov model[25].

Two generalizations of the single path problem for performing collective inference on Markov models are introduced in Ref. [25], motivated by an effort to model bird migration patterns using a large database of static observations. The eBird database maintained by the Cornell Lab of Ornithology contains millions of bird observations from throughout North America reported by the general public using the eBird web application. Recorded observations include location, date, species and number of birds observed. The eBird data set is very rich, and the human eye can easily discern migration patterns from animations showing the observations as they unfold over time on a map of North America. However, the eBird data entries are static and movement is not explicitly recorded, only the distributions at different points in time. Conclusions about migration patterns are made by the human observer, and the goal is to build a mathematical framework to infer dynamic migration models from the static eBird data. Quantitative migration models are of great scientific and practical importance. For example, this problem comes from an interdisciplinary project at Cornell University to model the possible spread of avian influenza in North America through wild bird migration.

The migratory behavior of a species of birds can be modeled by a single generative process that independently governs how individual birds fly between locations. This gives rise to the following inference problem: a hidden experiment draws many independent sample paths simultaneously from a Markov chain, and the observations reveal collective information about the set of sample paths at each time step, from which the observer attempts to reconstruct the paths.

Figure 4 displays the pattern of ruby-throated hummingbird migration inferred by this model for the four weeks starting on the dates indicated. The top row shows the distributions and migrating paths inferred by the model: grid cells colored in lighter shades represent more birds; arrows indicate flight paths between the week shown and the following week, with line width proportional to bird flow. The bottom row shows the raw data for comparison: white dots indicate negative observations; black squares indicate positive observations, with size proportional to bird count; locations with both positive and negative observations appear a charcoal color. This leads to a somewhat surprising prediction that when migrating north, some hummingbirds will fly across the Gulf of Mexico while others follow the coastline, but when flying south, they generally stay above land. This prediction has been confirmed by work performed by ornithologists. For example, in the summary paragraph on migration from the Archilochus colubris species account[26], Robinson et al. write “Many fly across Gulf of Mexico, but many also follow coastal route. Routes may differ for north- and southbound birds.” The inferred distributions and paths are consistent

with both seasonal ranges and written accounts of migration routes.

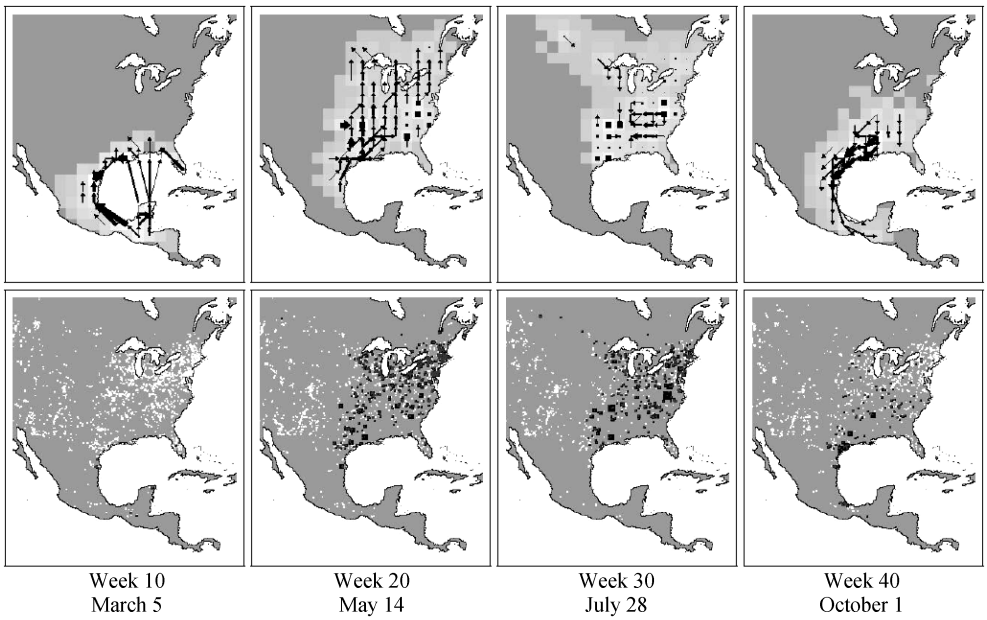

Figure 4. Ruby-Throated hummingbird (Archilochus colubris) migration

3 Theoretical Foundation

As demonstrated in the previous section, the focus of modern computer science research is shifting to problems concerning large data sets. Thus, a theoretical foundation and science base is required for rigorously conducting studies in many related areas. The theory of large data sets is quite different from that of smaller data sets; when dealing with smaller data sets, discrete mathematics is widely used, but for large data sets, asymptotic analysis and probabilistic methods must be applied. Additionally, this change in the theoretical foundation requires a completely different kind of mathematical intuition.

We will describe three examples of the science base in this section. Section 3.1 briefly introduces some features of large graphs and two types of random graph models. Section 3.2 explores the properties and applications of high-dimensional data. Finally, Section 3.3 discusses some scientific and practical problems involving sparse vectors.

3.1 Large-Scale graphs

Large graphs have become an increasingly important tool for representing real-world data in modern computer science research. Many empirical experiments have been performed on large-scale graphs to reveal interesting findings[3,5,15,21−27]. A computer network may have consisted of only a few hundred nodes in previous years, but now we must be able to deal with large-scale networks containing millions or even billions of nodes. Many important features of such large graphs remain constant when small changes are made to the network. Since the exact structure of large

graphs is often unknown, one way to study these networks is to consider generative graph models instead, where a graph is constructed by adding vertices and edges in each time interval. Although such graphs typically differ from real-world networks in many important ways, researchers can use the similarities between the two types of networks to gain insight into real-world data sets.

A simple but commonly used model for creating random graphs is the Erdös-Renyi model, in which an edge exists between each pair of vertices with equal probability, independent of the other edges. A more realistic model is known as the "preferential attachment" model, in which the probability that an edge is adjacent to a particular vertex is proportional to the number of edges already adjacent to that vertex. In other words, a high degree vertex is likely to gain more edges than a low degree vertex. The preferential attachment model gives rise to the power-law degree distribution observed in many real-world graphs.

Another interesting feature of real-world networks is the existence of the "giant component". The following table describes the number of components of each size in a protein database containing 2,730 proteins and 3,602 interactions between proteins.

Component Size	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	···	1000
# Components	48	179	50	25	14	6	4	6	1	1	1	0	0	0	0	1	···	0

As the component size increases, the number of components of that size decreases. Thus, there are many fewer components of size four or more in this protein graph than components of size one, two, or three. However, those smaller components contain only 899 proteins, while the other 1,851 proteins are all contained within one giant component of size 1,851, as shown in the following table.

Component Size	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	···	1851
# Components	48	179	50	25	14	6	4	6	1	1	1	0	0	0	0	1	···	1

Consider the Erdös-Renyi random graph model in which each edge is added independently with equal probability. Suppose that we start with 1,000 vertices and zero edges. Then, there are clearly 1,000 components of size one. If we add one edge, we will have 998 components of size one and one component of size two. However, a giant component begins to emerge as more edges are added, as shown in the following table. The graph contains many components of small size and a giant component of size 101. This occurs because a component is more likely to attract additional vertices as its size increases.

Size of Component	1	2	3	4	8	10	14	20	55	101
Number of Components	367	70	24	12	2	2	2	1	1	1

Since random graph models mimic some vital features of real-world networks, it is often helpful to study the processes that generate these features in random graph models. An understanding of these processes can provide valuable insights for analyzing real-world data sets.

3.2 High-Dimensional data

High-Dimensional data is fundamentally different from low-dimensional data, and it is particularly useful in many real-world applications such as clustering[28−30]. For example, if we randomly generate points in a 2-dimensional plane, the distance between pairs of points will be quite varied. However, if we randomly generate points in a 100-dimensional space, then all pairs of points will be essentially the same distance apart. The reason that randomly generated points in high dimensions are the same distance apart is the following. Given two random points x_1 and x_2 in a d-dimensional space, the square of the distance between them is given by:

$$D^2(x_1, x_2) = \sum_{i=1}^{d} (x_{1i} - x_{2i})^2 \ .$$

By the Law of Large Numbers, the value of $D^2(x_1, x_2)$, which is the sum of a large number of random variables, is tightly concentrated about its expected value. Thus, high-dimensional data is inherently unstable for conventional clustering algorithms.

Another interesting phenomenon in high dimensions is the volume of a unit radius sphere. As the dimension d increases, the volume of a unit radius sphere goes to zero. This is easily seen by the following integration:

$$\begin{aligned} V(d) &= \int_{x_1=-1}^{1} \int_{x_2=-\sqrt{1-x_1^2}}^{\sqrt{1-x_1^2}} \cdots \int_{x_d=-\sqrt{1-x_1^2-x_2^2-\cdots-x_{d-1}^2}}^{\sqrt{1-x_1^2-x_2^2-\cdots-x_{d-1}^2}} \mathrm{d}x_d \cdots \mathrm{d}x_1 \\ &= \int_{S_d} \int_{r=0}^{1} r^{d-1} \, \mathrm{d}r\mathrm{d}\Omega = \frac{A(d)}{d} \ . \end{aligned}$$

where $A(d) = \int_{S_d} \mathrm{d}\Omega$ is the surface area of a unit radius sphere. Consider the Gaussian integration:

$$I(d) = \int_{x_1=-\infty}^{\infty} \int_{x_2=-\infty}^{\infty} \cdots \int_{x_d=-\infty}^{\infty} e^{-\left(x_1^2+\cdots+x_d^2\right)} \mathrm{d}x_d \cdots \mathrm{d}x_1 = \left(\int_{-\infty}^{\infty} e^{-x_1^2} \mathrm{d}x_1 \right)^d = \pi^{d/2}.$$

We can also write $I(d)$ as follows:

$$I(d) = \int_{S_d} \int_{r=0}^{\infty} r^{d-1} e^{-r^2} \ \mathrm{d}r\mathrm{d}\Omega = \frac{A(d)}{2} \int_{t=0}^{\infty} t^{d/2-1} e^{-t} \ \mathrm{d}t \quad \left(t = r^2\right) = \frac{A(d)}{2} \Gamma\left(d/2\right) \ .$$

Therefore,

$$V(d) = \frac{2\pi^{d/2}}{d\Gamma\left(d/2\right)} \qquad \text{and} \qquad \lim_{d\to\infty} V(d) \approx \lim_{d\to\infty} \frac{3^d}{d!} = 0 \ .$$

Consider a standard multivariate Gaussian distribution in one dimension ($\mu = 0, \sigma^2 = 1$). The maximum probability density is at the origin and essentially all of the probability mass is concentrated within a distance of 3σ of the origin. Now, consider a standard multivariate Gaussian distribution in high dimensions. If we integrate the probability mass within distance one of the origin, we get zero since the unit radius sphere has no volume. In fact, we will not discover any probability mass until we move away from the origin by a distance such that a sphere of that radius has

non-zero volume. This occurs when the radius of the sphere reaches $\sqrt{d}$, where d is the dimension of the space. As we continue to increase the volume of the sphere we are integrating over, the probability mass will soon stop increasing since the density function decreases exponentially fast. Therefore, even though the probability density is maximum at the origin, all of the probability mass is concentrated in a narrow annulus of radius $\sqrt{d}$.

Given two standard Gaussian processes whose centers are extremely close, each generating random data points, we should be able to tell which Gaussian process generated which point. In high dimensions, even though the centers are close to each other, the two annuli will not overlap by much. An algorithm for determining which Gaussian process generated which point is to calculate the distance between each pair of points. Two points generated by the same Gaussian process would be a distance $\sqrt{2d}$ apart, and the distance between two points generated by different Gaussian processes would be $\sqrt{\delta^2 + 2d}$, where δ is the distance between the two centers. To see this, first consider two random points generated by the same Gaussian process. Generate the first point and then rotate the coordinate axes such that the point is at the North Pole. Generate the second point and this point will lie on, or very near, the equator, since the surface area of a high-dimensional hyper-sphere is close to the equator. Thus, the two points and the origin form a right triangle. Having approximated the annulus by a hyper-sphere of radius $\sqrt{d}$, the distance between the two points generated by the same Gaussian process is $\sqrt{2d}$.

To calculate the distance between two random points generated by different Gaussian processes, again generate the first point and rotate the coordinate axes such that the point is at the North Pole. Then, generate the second point and it will lie on, or very near, the equator of the second hyper-sphere. Thus, the distance between the two points is given by $\sqrt{\delta^2 + 2d}$ as illustrated in Fig. 5.

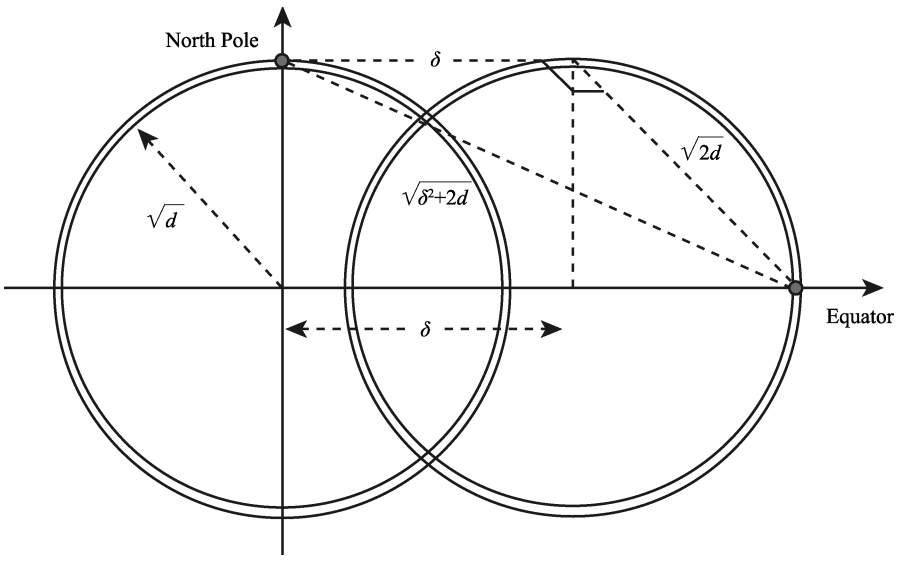

Figure 5. Two distinct Gaussian processes

Note that δ should be large enough such that $\sqrt{\delta^2 + 2d} > \sqrt{2d} + \gamma$, where γ includes the approximation error of the annulus associated with a hyper-sphere and the fact

that the second point is not exactly on the equator.

If the centers of the Gaussian processes are even closer, we can still separate the points by projecting the data onto a lower dimensional space that contains the centers of the Gaussian processes. When projecting the data points onto the hyper-plane containing the centers of the Gaussian processes, the distances between the centers are preserved proportionally. The perpendicular distance of each point to the hyper-plane is considered to be noise. This projection eliminates some of the noise and increases the signal-to-noise ratio. Apply the above approach to the projected data, and the dimension of the space has been reduced from d to k, where k is the number of Gaussian processes we attempt to distinguish among. Now, the Gaussian processes are required to be some smaller distance apart that depends only on k and γ.

The question remains of how one determines the hyper-plane through the centers of the Gaussian processes. This involves singular value decomposition (SVD). Consider the rows of an $n \times d$ matrix A as being n points in a d-dimensional space. The first singular vector of A is the direction of a line through the origin which minimizes the perpendicular distance of the n points to the line. Further, the best k-dimensional subspace that minimizes the perpendicular distance of the n points to the subspace contains the line defined by the first singular vector. This follows from symmetry assuming that the Gaussian processes are spherical. Given k spherical Gaussian processes, the first k singular vectors define a k-dimensional subspace that contains the k lines through the centers of the k Gaussian processes, and hence, contains the centers of the Gaussian processes.

A science base for high dimensions would also deal with projections. One can project n points in a d-dimensional space to a lower dimensional space while approximately preserving pair-wise distances proportionally, provided the dimension of the target space is not too low. Clearly, one could not hope to preserve all pair-wise distances proportionally while projecting n points onto a one-dimensional line. However, with high probability, a random projection to a $\log n$-dimensional space will approximately preserve all pair-wise distances proportionally[31].

3.3 Sparse vectors

Having sketched an outline of a science base for high-dimensional data, we now focus on studying sparse vectors as a science base for a number of application areas. Sparse vectors are useful to reduce the time required to find an optimal solution and to facilitate reconstructions and comparisons[21,32]. For example, plant geneticists are interested in determining the genes responsible for certain observable phenomena. The internal genetic code is called the genotype and the observable phenomena or outward manifestation is called the phenotype. Given the genotype for a number of plants and some output parameter, one would attempt to determine the genes responsible for that particular phenotype, as illustrated in Fig. 6.

Solutions to this type of problem are generally very sparse. Intuitively, one would expect this since only a small set of genes is probably responsible for the observed manifestation. This situation arises in a number of areas and suggests the following underlying problem: given a matrix A, a sparse vector x and a vector b where $Ax = b$, how do we find the sparse vector x knowing A and b? This problem can be formally

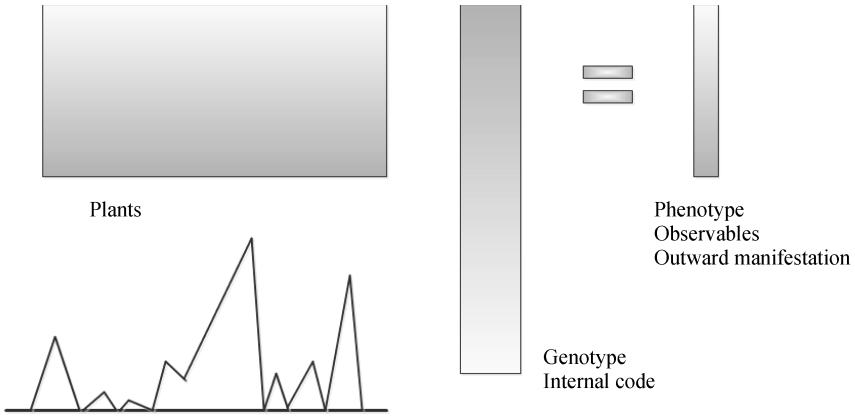

Figure 6. The genes responsible for a given phenotype

written as:

$$\text{minimize } ||x||_0 \quad \text{subject to} \quad Ax = b$$

where the 0-norm is the number of non-zero elements of the sparse vector x. However, the 0-norm is non-convex and optimization problems of this nature are often NP-complete. Thus, the question remains of when the solution to the convex optimization problem:

$$\text{minimize } ||x||_1 \quad \text{subject to} \quad Ax = b$$

correctly recovers the vector x. Note that the above minimization problem can be solved by linear programming.

4 Conclusion

Future computer science research is believed to employ, analyze, and interpret large data sets. In this paper, we have discussed several examples of current projects that represent modern research directions in computer science, ranging from identifying communities in large-scale social networks to tracing bird migration routes in North America. As computing pervades every facet of our lives and data collection becomes increasingly ubiquitous, feasible algorithms for solving these problems are becoming more and more necessary to analyze and understand the vast quantities of available information. In order to rigorously develop these algorithms, a mathematical foundation must be established for large data sets, including the theory of large graphs, high-dimensional data, sparse vectors and so on. These innovative studies discover striking results that reveal a fundamental change in computer science that will reshape our knowledge of the world.

References

[1] Shaparenko B, Caruana R, Gehrke J, Joachims T. Identifying temporal patterns and key players in document collections. Proc. of IEEE ICDM Workshop on Temporal Data Mining: Algorithms, Theory and Applications (TDM-05). 2005. 165–174.

[2] Gaertler M. Clustering. Network Analysis: Methodological Foundations, 2005, 3418: 178–215.

[3] Leskovec J, Lang K, Dasgupta A, Mahoney M. Statistical properties of community structure in large social and information networks. Proc. of 18th International World Wide Web Conference

(WWW). 2008.
[4] Newman MEJ, Girvan M. Finding and evaluating community structure in networks. Phys. Rev. E, 2004, 69(026113).
[5] Wang L, Hopcroft JE. Community structure in large complex networks. Proc. of 7th Annual Conference on Theory and Applications of Models of Computation (TAMC). 2010.
[6] Clauset A, Newman MEJ, Moore C. Finding community structure in very large networks. Phys. Rev. E, 2004, 70(06111).
[7] Girvan M, Newman MEJ. Community structure in social and biological networks. Proc. of Natl. Acad. Sci. USA. 2002, 99(12): 7821–7826.
[8] Newman MEJ. Detecting community structure in networks. The European Physical J. B, 2004, 38: 321–330.
[9] Newman MEJ. Fast algorithm for detecting community structure in networks. Phys. Rev. E, 2004, 69(066133).
[10] Newman MEJ. Finding community structure in networks using the eigenvectors of matrices. Phys. Rev. E, 2006, 74(036104).
[11] Newman MEJ. Modularity and community structure in networks. Proc. of Natl. Acad. Sci. USA. 2006, 103(23): 8577–8582.
[12] Lang K, Rao S. A flow-based method for improving the expansion or conductance of graph cuts. Proc. of 10th International Conference Integer Programming and Combinatorial Optimization (IPCO). 2004.
[13] Schaeffer SE. Graph clustering. Computer Science Review, 2007, 1(1): 27–64.
[14] Mishra N, Schreiber R, Stanton I, Tarjan RE. Finding strongly-knit clusters in social networks. Internet Mathematics, 2009, 5(1–2): 155–174.
[15] He J, Hopcroft JE, Liang H, Supasorn S, Wang L. Detecting the structure of social networks using (α, β)-communities. Proc. of 8th Workshop on Algorithms and Models for the Web Graph (WAW). 2011.
[16] Liu Y, Bian J, Agichtein E. Predicting information seeker satisfaction in community question answering. SIGIR, 2008.
[17] Wang K, Ming Z, Chua T. A syntactic tree matching approach to finding similar questions in community-based qa services. SIGIR, 2009.
[18] Wang XJ, Tu X, Feng D, Zhang L. Ranking community answers by modeling question-answer relationships via analogical reasoning. SIGIR, 2009.
[19] Yang T, Jin R, Chi Y, Zhu S. Combining link and content for community detection: a discriminative approach. KDD, 2009.
[20] Lin CX, Zhao B, Mei Q, Han J. Pet: a statistical model for popular events tracking in social communities. KDD, 2010.
[21] Soundarajan S, Hopcroft JE. Recovering social networks from contagion information. Proc. of 7th Annual Conference on Theory and Applications of Models of Computation (TAMC). 2010.
[22] Gomez-Rodriguez M, Leskovec J, Krause A. Inferring networks of diffusion and influence. Proc. of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD). 2010.
[23] Leskovec J, Krause A, Guestrin C, Faloutsos C, VanBriesen J, Glance N. Cost-effective outbreak detection in networks. Proc. of 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD), 2007.
[24] Clauset A, Moore C, Newman MEJ. Hierarchical structure and the prediction of missing links in networks. Nature, May 2008.
[25] Sheldon DR, Saleh ElmohamedM A, Kozen D. Collective inference on markov models for modeling bird migration. Neural Information Processing Systems (NIPS), 2007.
[26] Robinson TR, Sargent RR, Sargent MB. Ruby-throated hummingbird (archilochus colubris). In: Poole A, Gill F, eds. The Birds of North America, number 204. The Academy of Natural Sciences, Philadelphia, and The American Ornithologists' Union, Washington, D.C., 1996.
[27] Gehrke J, Ginsparg P, Kleinberg J. Overview of the 2003 KDD cup. SIGKDD Explorations, 2003, 5(2): 149–151.
[28] Aggarwal CC, Wolf JL, Yu PS, Procopiuc C, Park JS. Fast algorithms for projected clustering.

ACM SIGMOD, 1999, 28(2): 61–72.

[29] Kailing K, Kriegel HP, Kr**H**oger P. Density-connected subspace clustering for high-dimensional data. Proc. of 4th SIAM International Conference on Data Mining (SIAM). 2004. 246–257.

[30] Kriegel HP, Kr**H**oger P, Zimek A. Clustering high-dimensional data: A survey on subspace clustering, pattern-based clustering, and correlation clustering. ACM Trans. on Knowledge Discovery from Data, 2009, 3(1): 1–58.

[31] Garey MR, Johnson DS. Computers and intractability: a guide to the theory of NP-completeness. Freeman WH, 1979.

[32] Gibbs NE, Poole Jr. WG, Stockmeyer PK. A comparison of several bandwidth and profile reduction algorithms. ACM Trans. on Mathematical Software, 1976, 2(4): 322–330.