The Theoretic Center of Computer Science[2]

Michael Kuhn and Roger Wattenhofer
Computer Engineering and Networks Laboratory
ETH Zurich
8092 Zurich, Switzerland
{kuhnmi,wattenhofer}@tik.ee.ethz.ch

Abstract

In this article we examine computer science in general, and theory and distributed computing in particular. We present a map of the computer science conferences, speculating about the center of computer science. In addition we present some trends and developments. Finally we revisit centrality, wondering about the central actors in computer science. This article is a printed version of a frivolous PODC 2007 business meeting talk, held by the second author. In an attempt to address a broader audience we have extended the content by data about theory conferences and computer science in general. Still, the character of the information remains the same, meaning our investigations should not be taken too seriously.

1 Introduction

People have always been fascinated by centrality. They have been afraid of the middle of the night, and estimated time based on the middle of the day. Most people also like to be the center of attention, and look at things in egocentric ways. Not surprisingly, the Babylonians placed themselves in the middle of their world map. And even today, almost any country claims to be the center of its continent. Noteworthy is surely also the question about the center of the universe. Among many others Ptolemy, Copernicus, and Galileo have thought about it, some of them at the risk of life.[3]

Recently, when presenting some material about conference similarities, we have been asked a question of similar nature by a colleague. He was curious about the center of computer science. In a declaredly navel-gazing attempt to address this question we will outline a map of computer science in the beginning of the article. We will then present different facets of the world of computer science. Being genuinely more risk-averse than Galileo et al.[4] we would like to stress that all our investigations should be taken with a grain of salt.

[2] ©Michael Kuhn and Roger Wattenhofer, 2007

[3] At the time the Catholic Church accepted Galileo's model, physicists have already claimed that there is no center of the universe. Contradicting the physicists, Fremont, WA declares itself the center of the universe. Other eccentric centers of the universe are the Toronto Maple Leaf hockey team, or a manhole cover in Wallace, Idaho.

[4] Not to mention Giordano Bruno!

2 Computer Science Cartography

Maps have always been an essential instrument in exploring space. It was, after all, the Ptolemy world map—and its mistakes—that triggered Columbus' monumental voyage and the discovery of America. Reason enough to draw a map of "our world", the world of computer science.

2.1 Method

We first set up a model for the world of computer science. Assuming that computer science conferences adequately represent computer science disciplines, we model the interrelations between the conferences as a graph.

To construct such a graph, we introduce a notion of "social conference similarity". Basically, two conferences are supposed to be closely related if they exhibit a large number of common authors. To avoid overestimating the similarity of massive events—they naturally have a large number of common authors—we improve on this idea by incorporating a normalization method: Consider two conferences, C_1 and C_2, that contain a total of s_1 and s_2 publications, respectively. Further, assume that there are k authors A_i ($i = 1, ..., k$) that have published in both of them and that author A_i has $p_{i,1}$ publications in conference C_1 and $p_{i,2}$ publications in conference C_2. We can now define the similarity $S(C_1, C_2)$ between C_1 and C_2 as follows[5]:

$$S(C_1, C_2) = \sum_{i=1}^{k} \min\left(\frac{p_{i,1}}{s_1}, \frac{p_{i,2}}{s_2}\right)$$

Applying this similarity measure to all pairs of conferences results in the desired graph, our model of the world of computer science. The required information for this graph was extracted from the DBLP bibliographic repository.

Luckily, the "cartography of graphs", better known as graph embedding, is a well explored topic. Simply speaking, the goal of a graph embedding algorithm is to assign Euclidean coordinates to all vertices of a graph, such that the resulting positions well reconstruct the graph distances between all pairs of nodes (also multi-hop).[6] Similarly as it is impossible to undistortedly draw the globe in 2 dimensions, it is in general also not possible to exactly represent a graph metric in 2 dimensions.

Out of the rich choice of embedding algorithms we have opted to apply the widely used multi-dimensional scaling method (MDS) to create our map. MDS minimizes the mean absolute squared error when comparing all the pairwise distances in the graph metric (i.e. shortest paths) and the resulting Euclidean metric.

It is important to note that there is no notion of orientation for such an embedding. It only represents pairwise distances, and would be equally valid after applying any congruence transformation.

2.2 Results

Figure 3 shows the map of computer science conferences, constructed as described in the previous section. To keep the map readable we have restricted the vertex set to only include top-tier conferences. The required

[5]There are surely a couple more normalization methods one could think of. Manual inspection of the options we tried, revealed this to be the most suited.

[6]More generally, graph embedding is the process of mapping a graph into any other space (not necessarily Euclidean), thereby pursuing some (not necessarily distance related) design goals.

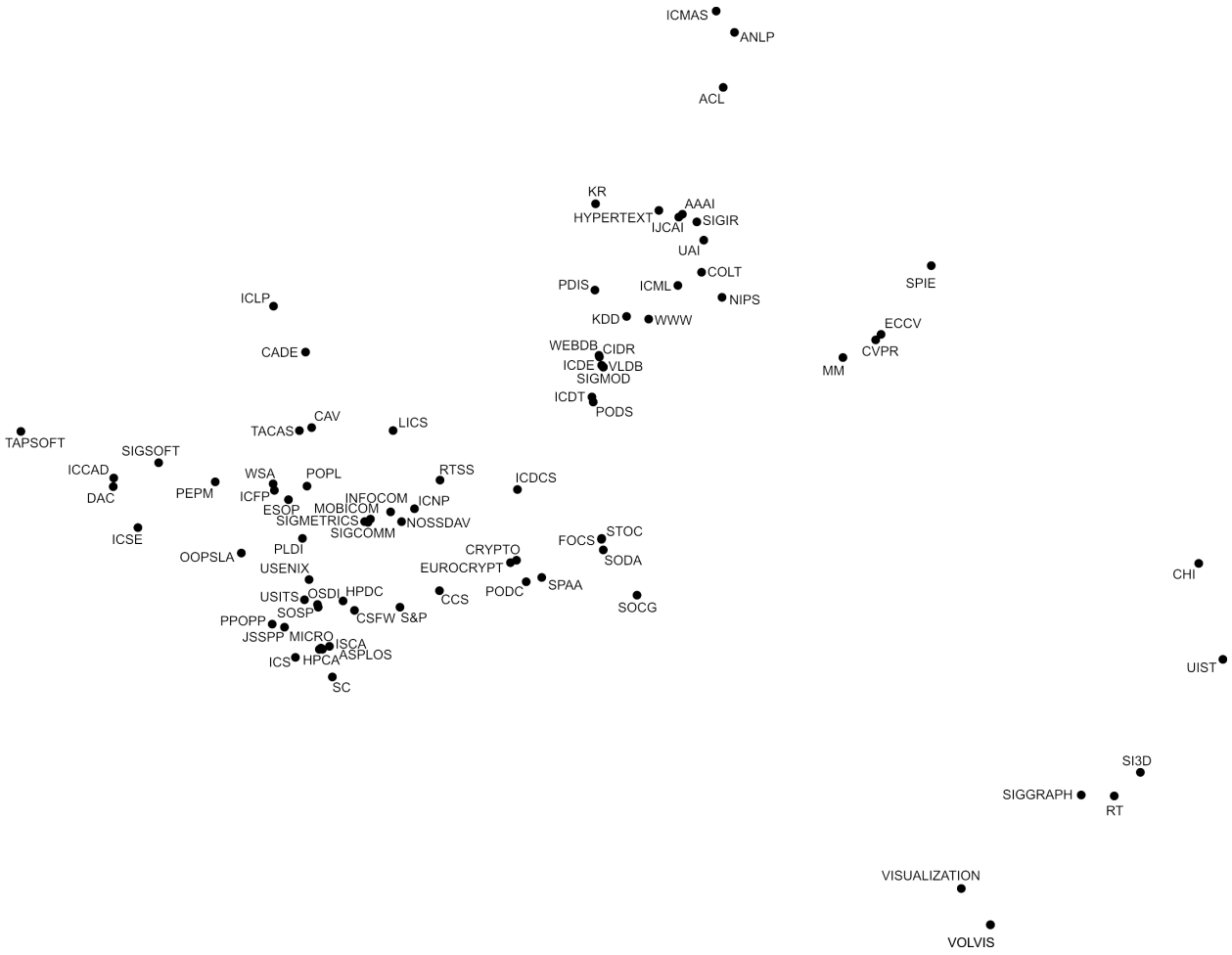


Figure 3: Our map of computer science: The map was constructed by embedding the conference graph into a 2-dimensional Euclidean space. Only top-tier conferences (according to Libra) are shown. Note that the map only represents pairwise distances, there is no notion of orientation, i.e. the axes can be chosen arbitrarily.

conference classification—with which the authors disagree to a certain extent—was taken from *Libra*[7].

Our map gives room for quips and questions. It can, first of all, be observed that like-minded conferences tend to gather together, they build families and tribes. It is thus possible to investigate the interconnections between entire "scientific cultures" rather than single conferences. Software engineers and design automation specialists, for example, live in the west of this world.[8] Their eastern neighbors are mainly into computer networks, and further south hardware architecture and operating systems experts are located.

In the very north, people mainly study natural languages and artificial intelligence. Also, they retrieve information from and mine the data provided by their direct southern neighbors (databases and the world wide web). The south-eastern population, finally, lives quite independently, and is mostly interested in user interfaces, graphics and visualization. It is interesting that these disciplines almost "drop" from the disk, and that we experience these large distances in the south-eastern part of our map. Are these gaps merely an artifact of our sloppy model, making our drawing as inaccurate as the disk-shaped Babylonian world map, or does this emptiness call for attention by the computer science community?!

We observe that the center of the graph is dominated by theory, and—slightly less significantly—also cryptography, distributed computing, networks, and databases. Interestingly, all these areas can be seen as service suppliers for other disciplines: Data can hardly be mined without databases, software design requires a great amount of (algorithmic) theory, and networks as well as cryptography and distributed systems play a crucial role in many real world systems. Hence, looking at our map, a theoretician could, with a healthy dose of self-esteem, derive that he (or at least his field of research) is in the center of computer science.[9]

Our theoretician would surely like to get a more detailed view on his self-declared center of computer science. Even though we question the center predicate, we do him—and hopefully also our theory-focused readers—a favor and zoom into this section. We have inserted the lower-tier conferences (again, according to Libra, and again, we do not fully agree) into the drawing by placing them into the weighted center of their closest top-tier neighbors (while keeping the layout of these top-tier conferences fixed). Figure 4 illustrates the theory close-up of our map.

In an attempt to interpret the figure, one might notice a slight separation of west and east. The west-side of the map seems to be mostly populated by the species of "practical theoreticians", while the east-side is rather inhabited by "pure theory". Interestingly, the cryptographers squeeze themselves into the territory of distributed computing. Clearly the two share common roots, but still the proximity is remarkable. Comparing the distances in the map to the distances in the original graph reveals that there is quite some discrepancy in this specific case. We speculate that distributed computing and cryptography get intermixed because they both lie between (pure) theory and systems/networking. Possibly, a 2-dimensional map can not accurately represent the ultimate truth. As detailed views usually provide deeper insights into a problem, the insight of this close-up is perhaps that our map should be treated with care.

3 Migration in Computer Science

As the history of mankind, the world of computer science is not static. Communities emerge, disappear, and migrate. In the following we want to have an eye on the movements in the proximity of PODC, as well as

[7] `http://libra.msra.cn/` For each discipline the site lists the major conferences grouped by tier (e.g. `http://libra.msra.cn/conf_category_1.htm`)

[8] Again, note that there are no axes defined, we just orient ourselves as the picture is aligned here.

[9] Clearly, the map leaves room for discussion. A closer look reveals many surprises, e.g. the location of design automation conferences. Also, other (equally reasonable) centers, such as databases or networking could doubtlessly be identified—please mind that we write this article for SIGACT and not SIGMOD or SIGCOMM. More fundamentally one might wonder whether there at all is a center of computer science; maybe we rather live in a "centerless" world, much like modern physics sees the universe.

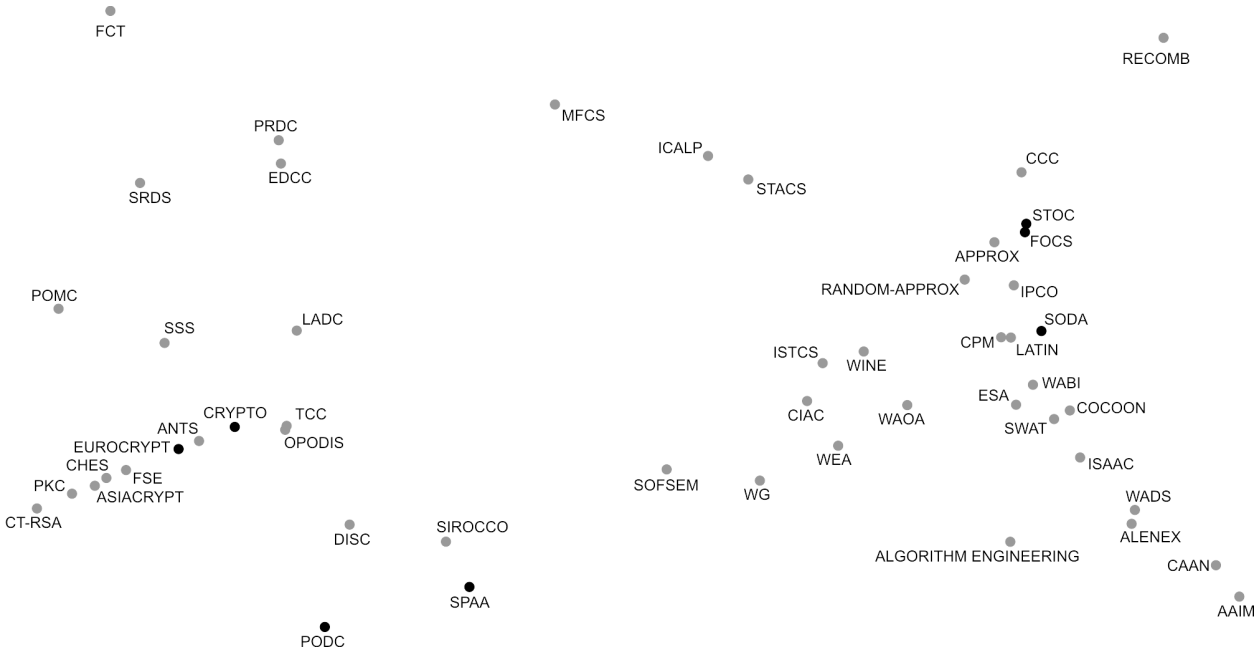


Figure 4: Close-up of the map around the theory conferences. Tier-1 conferences (according to Libra) are marked black, any other conferences gray.

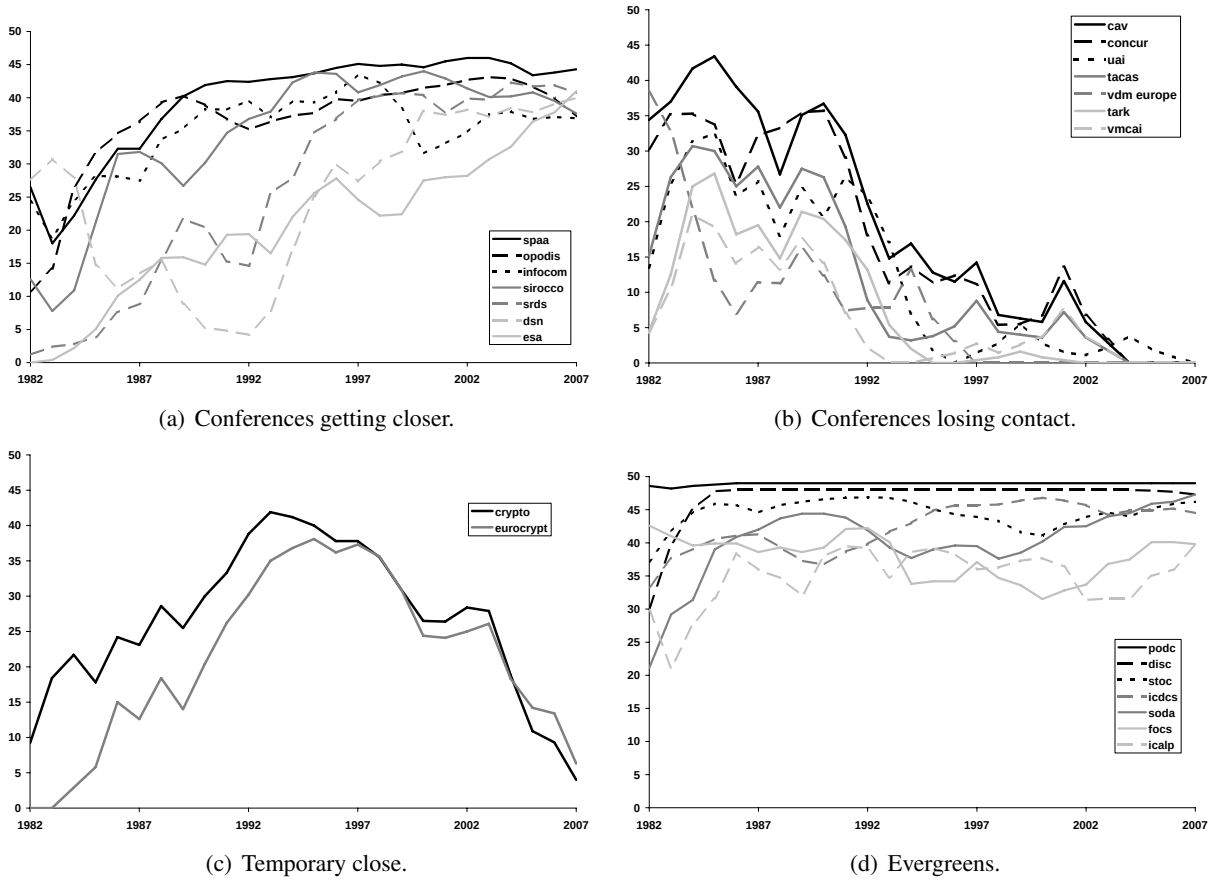
some major theory conferences, namely STOC, FOCS and SODA (which we in the following will treat as one).

3.1 Method

A conference is mainly defined by its participating authors. We thus assume that looking at the changes in authorship is a handy method to capture the changes of a conference over time. Consequently, we have applied the idea of our "social similarity measure" from Section 2.1 on a per year basis: For a particular year and conference we have examined where else the authors would typically publish (at all times). This gives, for this particular conference, an insight in what other conferences have been particularly close at a given point in time.

3.2 Results

The described "time dependent social similarity measure" allows to plot the development of a conference's proximity over time. Figures 5 and 6 show such plots for PODC and STOC/FOCS/SODA. An interesting observation is, maybe, the temporal closeness of cryptography to both, PODC and the theory conferences. Most likely, this does not mean that cryptography is in danger of extinction, but rather protocols an emancipation process; cryptographers have grown strong enough to form their own, independent community and thus drift away from other conferences.



(a) Conferences getting closer.

(b) Conferences losing contact.

(c) Temporary close.

(d) Evergreens.

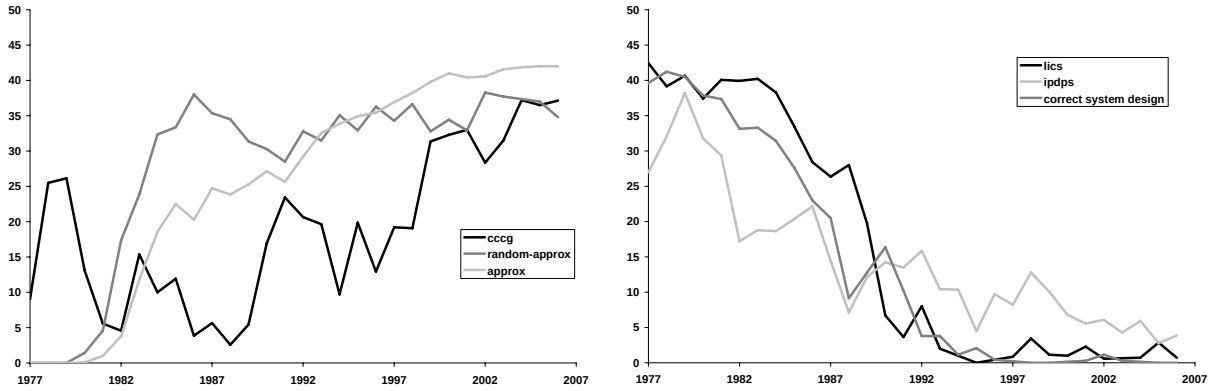
Figure 5: Related conference trends for PODC.

4 Keyword Trends: Predicting the Perfect Paper Title

After this brief excursion into the evolution of conference relationships, we want to come back to a more global view. How did computer science evolve over time? We believe that terminology is an meaningful witness for scientific history. We thus analyze the change of central keywords in computer science. Moreover, we will, in a tongue-in-cheek way, suggest the perfect titles for the next year's edition of PODC as well as STOC/FOCS/SODA.

4.1 Method

As usual when it comes to predicting future trends we rely on historic data. For this purpose, history has been divided into 6 time-slots, from 1977 to 2006 in 5-years chunks. Moreover, for each conference in question we have parsed all the titles appearing in its proceedings to extract the single words. Counting the number of occurrences for each word and time-slot allows to establish a "per-time-slot" ranking which can then be mined to extract trends. We have restricted the search space to keywords that made it into the top-50 in at least one time-slot. The actual trend analysis (i.e. selection of the keywords that best characterize the movements) was then mainly carried out manually.



(a) Conferences getting closer.

(b) Conferences losing contact.

(c) Temporary close.

(d) Evergreens.

Figure 6: Related conference trends for STOC/FOCS/SODA.

4.2 Results

How did computer science research evolve over time and where does it go? Our analysis shows that *wireless mobile agents based on neural learning that quickly adapt to image and video web services* are fashionable.[10] On the other hand, we are no longer interested in *computer experts specifying the semantics of parallel relational databases for VLSI in the prolog programming language*.[11] Also, *knowledge about object oriented simulation logic* does not seem to be required any longer, even though the topic was a hot in the beginning of the 90ties.

If you do not want to risk being labeled antediluvian after submitting to a major theory conference, better name your paper, say:

> *"Online Quantumn Algorithms to Approximate Directed Location Codes"* or *"On the Hardness of Scheduling using Random Sampling"*

Moreover, avoid titles like

[10] In other words, the keywords web, agent, wireless, neural, mobile, video, learning, image, adaptive, services are becoming increasingly popular.

[11] Again, we witness that the keywords computer, expert, etc. have seen their peak.

"Relational Logic to Separate Automata Isomorphism Classes" and *"Probabilistic Parallel Programs for VLSI Design"*

as these will immediately out you as being stuck in the 80ties. Similarly, the next year's blockbuster of PODC is rather going to be called

"Failure Detectors and Scalable Dynamic Quorum: A Free Mobile Ad-Hoc Game with Selfish Peers"

than, for example

"Recover from Committed Ring-Deadlocks using Message Passing Communication, Temporal Knowledge and Parallel Computation Processes"

5 Kevin Bacon of Computer Science

After temporarily drifting away from the centrality question, we want to come back to it again in this section. Maybe it was short-sighted to ask *what* the center of computer science is. Why not asking about the central actor of computer science, i.e. *who* the center of computer science is? Science is, after all, made by people, and not disciplines.

5.1 Method

The central actor? Well, it is widely known that this is Kevin Bacon—in the movie industry.[12] In math, the corresponding role is accredited to Paul Erdős. In a similar approach we attempt to nominate the central actor in computer science as well as STOC/FOCS/SODA and PODC. Analogously to the construction of the Erdős number, we base our method on the co-author-graph. We then create the induced subgraph for each region of interest (PODC, STOC/FOCS/SODA, and computer science). For PODC, for example, this graph would only contain authors that have at least one PODC paper, and so on.

Other than in the construction of the Erdős number, we do not rely on shortest paths, but rather on the PageRank idea: We start several short random walks at different nodes of the graph, and count how often each author gets visited.[13] This idea is then extended to time dependent centrality, by starting the random walks only at authors that have published in the last five years.

5.2 Results

Table 1 lists the central actor of computer science, as well as his major competitors. Oddly enough, our central actor, Alberto L. Sangiovanni-Vincentelli, acts quite at the edge of our computer science map.[14] But which is right, the map or the election? Critics might ask us to rethink our election process—as oppositions always do after elections. Maybe they are right. Sangiovanni-Vincentelli's toughest competitors, though, seem to live much more in the heartland of computer science—mostly in the area of databases. Maybe there is nonetheless a grain of truth in both, the map and the election?

[12] This is not perfectly true. Rod Steiger is said to be the best connected actor (source: Wikipedia article on the Bacon Number).

[13] More precisely, the number of walks is proportional to each author's number of Tier-1 papers.

[14] He has most frequently published in the area of design automation (DAG, ICCAD, DATE) and his research also covers electrical engineering. Note that both, the map and the author election base on the same set of publications, such that we cannot assume that electrical engineers get preferred by the election process.

All-time	Last 5 years
Alberto L. Sangiovanni-Vincentelli	Wei-Ying Ma
Noga Alon	Wei Wang
Hector Garcia-Molina	Hector Garcia-Molina
Michael Stonebraker	HongJiang Zhang
Michael J. Carey	Noga Alon
Yishay Mansour	Philip S. Yu
Moshe Y. Vardi	Christos Faloutsos
Christos Faloutsos	Gerhard Weikum
David Maier	Joseph M. Hellerstein
Philip S. Yu	Jiawei Han

Table 1: The most "central" authors in computer science. Note that name ambiguities might have distorted the results.

All-time	Last 5 years
Noga Alon	Noga Alon
Avi Wigderson	Avi Wigderson
Robert Endre Tarjan	Sanjeev Arora
Frank Thomson Leighton	Moses Charikar
Moni Naor	Anupam Gupta
Yishay Mansour	Madhu Sudan
Oded Goldreich	Erik D. Demaine
Richard M. Karp	Alan M. Frieze
Amos Fiat	Uriel Feige
Michael E. Saks	Yishay Mansour

Table 2: The most "central" authors in STOC/FOCS/SODA.

The critics would probably react even more sharply when looking at our central actors over the past 5 years. Indeed, there are some surprises in this list. Maybe this is a stunning proof of the often-quoted Asian research momentum, maybe it is caused by name ambiguities that even get reinforced by PageRank-like algorithms[15] (and most likely, it is a combination of both).

Clearly, the nomination of a single representative for the entire world of computer science is a delicate task, and the result inevitably controversial. Imagine, you had to elect a single person representing our planet in front of the universe—another unenviable task. In the following we thus focus on restricted areas of research. Elections within a single culture seem more realistic. Not surprisingly, Noga Alon, second in the list of computer science, is attributed the honor of the central player in theory (see Table 2).

For the PODC community, Nancy Lynch is found on top, as shown in Table 3. All in all, there seem to be less surprises in top-10 lists of the theory cluster and PODC. Most likely, name ambiguities play a less important role in these more restricted areas, as they contain a lower number of people. If you could not find your-

[15]Persons holding an ambiguous name do not only get more weight themselves, but also pass this weight to their collaborators, therefore the reinforcement. We want to stress that the (top-ranked) name Wei-Ying Ma does not seem to suffer from such ambiguities (as opposed to others). Wei-Ying Ma might or might not have profited from "passed on" ambiguities just as anybody else in the list.

All-time	Last 5 years
Nancy A. Lynch	Rachid Guerraoui
Danny Dolev	Nancy A. Lynch
Hagit Attiya	Roger Wattenhofer
Yehuda Afek	Danny Dolev
Maurice Herlihy	C. Pandu Rangan
Nir Shavit	Hagit Attiya
Rachid Guerraoui	Michel Raynal
Sam Toueg	Maurice Herlihy
Michel Raynal	Idit Keidar
Yishay Mansour	Sam Toueg

Table 3: The most "central" authors in PODC.

self on the lists, feel free to check the extended versions on `http://www.confsearch.org/ca.jsp`.

6 Conclusion

What is the center of computer science? A controversial question. Some might claim that computer science is all about building computing machines—possibly only theoretically. Others think it is all about the art of programming these machines, or about describing languages to talk to them. Yet other people associate computer science with algorithms, and believe P vs NP is at the heart of computer science. Finally, the tremendous impact of the Internet, or the huge data collections in service today might speak in favor of networking or database experts, respectively. In this article, we have tried to highlight this controversial nature by examining various aspects from different points of view. We have claimed persons to be in the center of computer science, and at the same time drawn a map placing them at the very border. We have looked into the evolution of scientific disciplines and observed that the same evidence can be interpreted as both, the birth as well as the funeral of a research area. Without doubt the future will teach our evaluations a lesson, ultimately revealing in which direction computer science evolves, and maybe even discover the most influential computer scientist. After all, research is not about how many papers we write, or how many citations they get, but rather, what the best contributions are.