Who is Hiring Whom: A New Method in Measuring Graduate Programs

Mr. BOLUN HUANG, Microsoft Corp.

Bolun Huang is currently a software engineer in Microsoft Corp., Redmond. Before that, he was a master of science student in the Department of Electrical and Computer Engineering at Texas A&M University. He completed dual bachelors from a joint program between Queen Mary University of London and Beijing University of Posts and Telecommunications. His research interests include: Data Mining, Social Network Analysis, Machine Learning and Computer&Network Security.

Ms. Samantha Wang

Samantha Wang is an undergraduate student studying Electrical and Computer Engineering at Carnegie Mellon University.

Prof. Narasimha Reddy

Who is Hiring Whom: A New Method in Measuring Graduate Programs

Abstract

In this paper, based on the assumption that "schools tend to hire Ph.D.s from peer or better schools", we propose a statistical and mathematical approach to rank graduate programs using algorithms deployed on a mutual "*hiring graph*" among universities. In order to validate our approach, we collect faculty data from the top 50 Computer Science (CS) departments, the top 50 Mechanical Engineering (ME) departments and the top 50 Electrical Engineering (EE) departments across the United States according to U.S. News so as to construct our *hiring graph*. We refine the PageRank (PR) algorithm and the Hyperlink-Induced Topic Search (HITS) algorithm in order to rank the graduate programs from the *hiring graph*. Our new rankings are generally consistent with U.S. News rankings, while exposing some new observations about some particular programs. By conducting extensive data analysis, we discover many interesting patterns and insights from our data. Finally, we propose a cross-domain model for graduate program ranking and introduce weight differentiation adjustment and tiles into our rankings.

Introduction

Academic programs are ranked using different objective and subjective metrics, providing different perspectives on the quality, productivity and affordability of the programs. Program rankings are closely followed by aspiring students, universities and employed in hiring and funding decisions. Among the many rankings of programs, U.S. News rankings have a wide following. U.S. News updates the ranking of graduate programs in multiple fields annually. According to the statement from U.S. News' website[1], they rank the graduate programs based on both statistical data and expert assessment data. The statistical data includes both input and output measures, reflecting the quality of resources into the programs and educational outcomes from the programs. The expert assessment data is collected from the input of program deans. Each dean is asked to rank a program from 1 to 5 and the average rating is used as the ranking score. Finally these two types of measurements are normalized, weighted and totaled into a ranking score for each program.

Related research on university program ranking has been done. For example, a comprehensive study on university ranking is provided in book[2], revealing the theoretical basis of traditional university ranking system. Besides this, various ranking metrics such as citation counts, hiring

preference and some other indicators have been explored. For example, some researchers proposed using citation as a measurement, such as relative citation counts among universities[3] and h-index[4], to evaluate the research quality of graduate programs in a particular field. Barnett et al[5] proposed the use of faculty hiring networks as an indicator in the university program ranking. Lopes et al[6] proposed a social network analysis on university ranking based on the internal collaborations among universities.

We propose a methodology to generate rankings of university programs from what we call a "*hiring graph*". The *hiring graph* is basically a directed social graph revealing the employment relationships of Ph.D.s among universities. The *hiring graph* consists of different university programs as nodes, and edges corresponding to the hiring of one program's graduates by another program. In the *hiring graph*, a directed edge from program A to program B indicates that A hires at least one Ph.D. from B as its faculty member. Our hypothesis is that "schools tend to hire Ph.D.s from peer or better schools". We note that a lot of resources are placed in the hiring activity, including assessment from domain experts, academic review, salaries and so on, and therefore the hiring decision reflects the academic quality of the faculty member in a comprehensive way.

Our rationale for employing the *hiring graph* has several reasons. First, this is based on our hypothesis that "universities tend to hire Ph.D.s from peer or better programs". This is anecdotally validated by many school hiring practices. Second, we did not want a school's ranking to be impacted by its own decisions; the ranking of a program has to be validated by decisions of others. Hence, we only consider the incoming edges of our *hiring graph*, i.e., only the Ph.D.s hired by other programs impact its ranking. Third, the *hiring graph* is somewhat self-consistent in the sense that we don't need any external input in the process of ranking. For example, if we were to consider hiring by industry, we would need to somehow have a notion of the relative value of a "Google hire" versus an "IBM hire". Fourth, since the hiring slots tend to be few and expensive in resources, we postulate rankings based on hiring decisions are harder to "game".

There are several issues with employing the university *hiring graph*. First, a very small percentage of graduates actually get hired by universities and hence this is a small sample of the total population. Second, a university professor's tenure system biases the *hiring graph* towards a "survival bias". Given that tenure decision is made within 5-7 years and a typical professor's career may span 30 years, most of the information in the *hiring graph* tends to reflect professors who get through the tenure process.

Third, the longevity of a typical professor's career makes a hiring decision that reflects on that program for a long period of time. Our analysis reflects this as explained later. Fourth, most departments tend to be small with a faculty size between 20 and 50, and hence the amount of sample data cannot be increased. Fifth, a department faculty may support multiple graduate programs and separating faculty into these programs requires more work. For example, several departments support both Electrical Engineering and Computer Science programs. When this data is available, it can be factored into our approach to rank individual programs.

We show that the proposed methodology that depends on hiring decisions provides valuable insight into ranking of graduate programs. We don't claim that this is superior to other methods of ranking, but that it provides a new way to rank graduate programs.

2 Data Set

2.1 Data Description

In order to construct the *hiring graph*, we collected two faculty profile data sets, from the top 50 Computer Science (CS) Departments[7] and the top 50 Mechanical Engineering (ME) Departments[8] across the USA respectively, both retrieved from U.S. News' latest released rankings. We did not combine these two data sets even though we have found a few ME professors graduated with CS Ph.D.s. For each faculty member in our data set, we collected two pieces of information, where and when the faculty member got his/her Ph.D.. Table 1 shows some sample data that we have collected. In Table 1, columns 2 to 5 represent all 4 entries for each faculty member: *1) Dept.*: Department that the faculty member works in; *2) Univ.*: University that the faculty member works in; *3) Ph.D. From*: University from which the faculty member got Ph.D.; *4) Year Grad.*: Year the faculty member got his/her Ph.D..

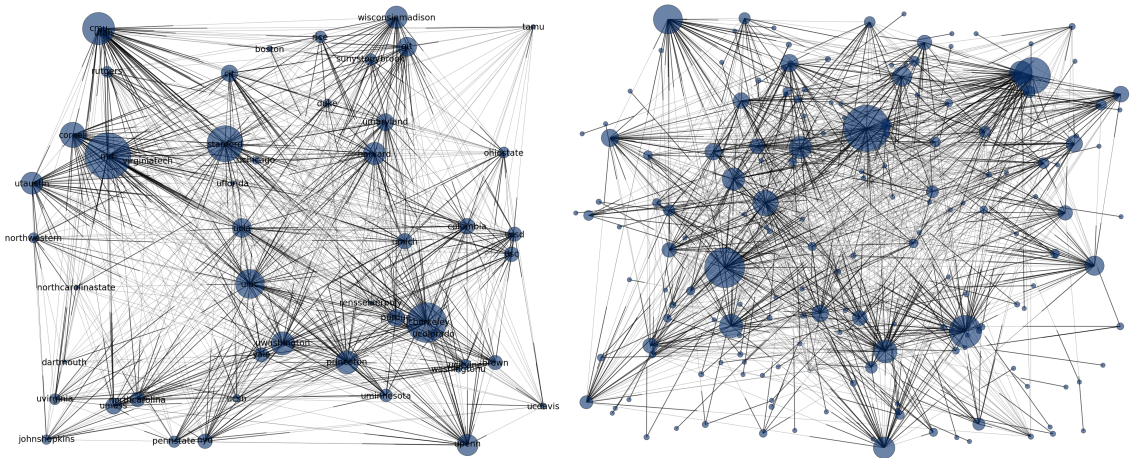
There are several things that we have to point out in our data set. First, some professors do not post their educational information on the web at all. Luckily, most of the faculty members disclose their resume or educational information on their department page or personal page, making it possible for us to collect a large enough sample for the *hiring graph*. Second, all the faculty data we collected reflects the current status of each program. This is to say that, the graph only reflects current employment and does not reflect historical employment. Third, the graph also does not reflect the the hiring decisions that may have been terminated without tenure.

Since we cannot find any organization that can provide such data, our data is collected from the website of each graduate program. This data was collected manually from March 2014 to April 2014. For the top 50 CS programs data set, we collected data from 2,018 faculty members currently in those programs. Out of these, 1,793 (88.9%) faculty members have their Ph.D. graduation year information on their web page. For the top 50 ME programs data set, we collected data from 1,941 faculty members currently in those programs, of which 1,709 (88.0%) faculty members have educational year information on their web page.

Our data reflects that the faculty Ph.D. graduation years range from 1949 to 2014 in the CS data set and from 1946 to 2013 in the ME data set. This enables us to bin the data based on the year to obtain a historical progression of school hirings. While our methodology can be applied to the entire *hiring graph* of all CS or ME programs, we restrict ourselves to the top 50 programs due to difficulties in collecting the data manually. The data used in this paper is publicly accessible at our shared data set oursharedata.

2.2 "*Hiring Graph*"

Mathematically, the *hiring graph* could be denoted as a directed graph $G = (V, E)$, comprising a set V of nodes (programs) and a set E of edges. An edge $E(x, y)$ means there is at least one Ph.D. from university y hired by university x as a faculty member. In the *hiring graph*, one university might hire several Ph.D.s from another university as faculty members. In this case, we set the weight of each edge to be the number of Ph.D.s hired by that university. For example, assuming



(a) Subtracted Hiring Graph (b) Extended Hiring Graph

Figure 1: Hiring Graph in CS Data Set

university A hires 9 Ph.D.s from university B, regardless of the year in which the Ph.D.s were graduated, the weight of edge $E(A, B)$ would be 9.

There are *self-edges* in the *hiring graph*, given that some programs hire their own Ph.D.s as faculty members. We also have faculty members from many other universities outside the Top 50. For example, in the CS data set, many faculty members come from universities like Hebrew University and University of Toronto. In our CS data set, we have 182 universities in our graph in total, among which are the top 50 CS Schools. In our experiments, we might or might not consider those universities outside the top 50 while running our algorithms. Similarly, there are 211 universities in our ME data set. Figure 1a is the subtracted *hiring graph* exclusively for the top 50 CS schools; figure 1b is the extended *hiring graph* when including those CS schools outside the top 50. We will discuss both the cases in our results. No matter in which case, we only consider the top 50 U.S. News programs for ranking.

While we limit our attention to this smaller set (due to difficulties in collecting the data), our methodology can be applied to all the schools when the data is available. Even with this data, we find insights about other schools that are not part of this set.

Table 1: Sample Data Format

Faculty	Dept.	Univ.	Ph.D. From	Year Grad.
F1	CS	CMU	MIT	2005
F2	CS	Princeton	UTAustin	2009
F3	CS	TAMU	UIUC	1997
⋮		⋮		
F4	ME	Cornell	Caltech	1987
F5	ME	UCLA	UCBerkeley	1991
F6	ME	Purdue	Stanford	2012
⋮		⋮		

3 Our Approach

The *hiring graph* and ranking of graduate programs have similarities to web graph and rankings of web pages. In page-ranking methodologies[10,11,12,13], links pointing to a web page are seen to increase the authority or importance of that page. Similarly, we postulate that hiring links provide similar information in our *hiring graph* about the quality of graduate programs. Hence, we propose to employ page ranking algorithms on the *hiring graph* to ascertain the quality of the programs.

The simplest way is to rank graduate programs according to their *in-degree*, which represents "the number of Ph.D.s who got hired by other schools" in the *hiring graph*. We also apply various link-based algorithms based on the PageRank (PR) algorithm[10] and the Hyperlink-induced Topic Search (HITS) algorithm[11] to generate our rankings.

3.1 PR-based Algorithms

3.1.1 PR Algorithm

The PR algorithm was originally invented to rank web pages according to their relative importance. It is based on a model called "*random surf model*", in which a random surfer is assumed to periodically jump to any random web page in the Web[10]. According to our assumptions described before, an incoming edge of a program would increase the importance of that program, which is consistent with idea of PageRank. Thus we believe that PageRank(PR)-like algorithms could be applied to our problem.

Here we describe an iterative manner of computing the PR score of every node in a graph. Let $G = (V, E)$ be the directed graph with a set V of vertices or nodes and a set E of edges. At the beginning, the PR scores of all nodes are initialized as $\frac{1}{N}$ where N is the total number of nodes in the graph. In each iteration, the PR score $r(p_i)$ of node p_i is defined as:

$$r(p_i) = \frac{(1-\alpha)}{N} + \alpha \cdot \sum_{p_j \in M(p_i)} \frac{r(p_j)}{L(p_j)} \tag{1}$$

where N is the total number of nodes, $p_0, p_1, \ldots, p_{N-1} \in V$, $M(p_i)$ is the set of programs that link to p_i, $L(p_j)$ is the number of outgoing links from p_j, and α is the damping factor. Letting the damping factor $\alpha = 0.85$ is a democratic choice[10]. Hence, in our PR-based approach we also use the same value, 0.85, as our damping factor. The algorithm stops when the PR scores converge, or in other words, remain unchanged or change little between two consecutive iterations.

As we can see in Equation 1, the sum of PR scores of $p_j \in M(p_i)$ brings a normalization effect to node p_i since the PR score of p_j is divided by the number of outgoing links of p_j.

One significant difference between the *hiring graph* and the web graph is that in the *hiring graph*, every edge has a weight. Therefore, we refine the original PR algorithm by considering edge weights. We consider two models for taking edge weights into consideration. The first model

gives a weight of 1 for each hire. The second model gives a total weight of 1 for all the hires of a program. If a program has N_i faculty, the outgoing edges from that program add up to N_i and in the second model, each hire of that program is given a weight of $\frac{1}{N_i}$ for a total weight of 1 for all the outgoing edges. These two models roughly correspond to the "House of Representatives" and "Senate" models of representation. Accordingly, we expect the first model to favor large programs and the second model to favor smaller programs.

3.1.2 Weighted PR Algorithm with Weights Normalized

When still considering the normalization effect, the new formula of the PR score $r(p_i)$ of node p_i would become

$$r(p_i) = \frac{(1-\alpha)}{N} + \alpha \cdot \sum_{p_j \in M(p_i)} \frac{r(p_j) \cdot w(\varepsilon(p_j, p_i))}{W(p_j)}, \tag{2}$$

where $w(\varepsilon(p_j, p_i))$ denotes the weight of $\varepsilon(p_j, p_i)$, $M(p_i)$ is the set of programs that link to p_i, α is the damping factor, and $W(p_j)$ is the sum of the weights of outgoing links from p_j, whose formula is:

$$W(p_j) = \sum_{\varepsilon(p_j, p_k) \in E} w(\varepsilon(p_j, p_k)). \tag{3}$$

3.1.3 Weighted PR Algorithm with Weights Unnormalized

We also test another refinement of the PR algorithm, in which the incoming link effect is not normalized by the sum of link's outgoing weights, but the total number of nodes in the graph. The formula of this refined PR algorithm is defined as follows:

$$r(p_i) = \frac{(1-\alpha)}{N} + \alpha \cdot \sum_{p_j \in M(p_i)} \frac{r(p_j) \cdot w(\varepsilon(p_j, p_i))}{N}. \tag{4}$$

As we can see in Formula 4, since the normalization factor is a fixed value, the normalization is not taken into effect. In this case, the actual number of edges and the actual value of edge weight matter.

3.2 HITS-based Algorithms

3.2.1 HITS Algorithm

The HITS algorithm was designed to discover the "authoritative" sources of a particular topic in the World Wide Web (WWW). It defines two types of pages in the Web: *hubs* and *authorities*. A *hub* is a page that links to other pages; an *authority* is a page that is linked to by other pages. The ranking philosophy behind HITS is a *mutually reinforcing relationship*: "a good *hub* is a page that points to many good authorities; a good *authority* is a page that is pointed to by many good

hubs"[11]. HITS is usually implemented in an iterative manner. In each iteration, the updating rules for the authority value $Auth(p)$ and hub value $Hub(p)$ of page p are formulated as

$$Auth(p) \leftarrow \sum_{\varepsilon(q,p)\in E} Hub(q) \tag{5}$$

and

$$Hub(p) \leftarrow \sum_{\varepsilon(p,q)\in E} Auth(q). \tag{6}$$

In each iteration, the new values are updated from the old values from last iteration. The hub scores and authority scores are normalized every time before the next iteration. The algorithm stops when the hub scores or authority scores converge. Finally, we look at the authority score of each program for ranking.

Unlike the PR algorithm, the HITS algorithm considers the effect of hubs. In the HITS algorithm, the effect of hubs and authorities will reinforce each other, and the authorities pointed to by strong hubs will stand out from the authorities pointed to by weak hubs. In the *hiring graph* especially, to UCBerkeley for example, we expect that a link from MIT would be more important than say, a link from TAMU, because MIT has more credits to support UCBerkeley being a better school. Under this assumption, we develop several variations of the HITS algorithm on our *hiring graph*. We employ the two weighting models described above.

3.2.2 Weighted HITS Algorithm

The updating rules are defined in Equation 7 and 8 for the weighted HITS algorithm. The only difference in the following updating rules from the formula of HITS is that we multiply the weight of the incoming/outgoing edges when calculating the authority/hub of a given node.

$$Auth(p) \leftarrow \sum_{\varepsilon(q,p)\in E} Hub(q) \cdot w(\varepsilon(q,p)) \tag{7}$$

and

$$Hub(p) \leftarrow \sum_{\varepsilon(p,q)\in E} Auth(q) \cdot w(\varepsilon(p,q)), \tag{8}$$

where $w(\varepsilon(p,q))$ is the weight of the edge from node p to node q.

3.2.3 Weighted HubAvg Algorithm

To overcome the shortcoming of the HITS algorithm that a hub might get a high weight when it points to a large number of low quality authorities, we suggest the following refinement accordingly[14]. While the updating rule for authority remains the same as Equation 7, the hub score is normalized by the number of outgoing edges of the node:

$$Hub(p) \leftarrow \frac{1}{M(p)} \sum_{\varepsilon(p,q)\in E} Auth(q) \cdot w(\varepsilon(p,q)), \tag{9}$$

where $M(p)$ is the sum of the weights of outgoing edges of node p.

4 Evaluation Methodology

In order to evaluate the performance of the above link-based algorithms, we use the U.S. News ranking as a baseline. However, this is not to say that U.S. News' rankings are the "ground truth", since they are a subjective point of view. We only use it as a reference to analyze our own ranking method so that we can discuss and reach conclusions based on what we have observed.

4.1 RankDistance

In order to measure the distance between two rankings, we employ a measure called "RankDistance". The computation of "RankDistance" is described as follows. Suppose R_1 and R_2 are two rankings from a set of samples $S = (a_0, a_1, \ldots, a_{N-1})$. Defining the rank of a_i in R_j as $P_{R_j}(a_i)$, the RankDistance $RankDist(R_1, R_2)$ between R_1 and R_2 is:

$$RankDist(R_1, R_2) = \frac{\sum_{a_i \in S} |P_{R_1}(a_i) - P_{R_2}(a_i)|}{N}, \tag{10}$$

where N is the total number of samples.

From Equation 10 we can see that the smaller the $RankDist(R_1, R_2)$ is, the closer R_1 and R_2 are. In our experiments, we compare our method with U.S. News' results using $RankDist$. As we said before, we are not taking U.S. News as the ground truth with which our results have to perfectly match.

4.2 Sensitivity Analysis

Apart from $RankDist$, which measures how close our rankings are to those of U.S. News, we also employ another measurement called "sensitivity analysis", which measures how robust the algorithm is to small changes in data. The intuition of sensitivity analysis is that universities keep hiring and professors retire or leave universities every year for various reasons. We do not expect significant movements in ranks due to minor changes in the *hiring graph*. The sensitivity analysis looks at this issue.

Our methodology to measure the sensitivity is as follows. For each ranked program, we carry out two hypothetical changes separately in the graph regarding this program: *1)* add a non-existing edge from one top ranked program to this program; *2)* delete one existing edge from the best program that links to this program; if not available, delete one existing edge from the best program that is linked to by this program. The first change will boost the rank of the program and the second change will lower the rank of the program. Thus by running a specific algorithm, we will have both an upper bound and a lower bound for each program.

5 Results on Top50 CS Data Set

In this section we present our experimental results on the Top50 CS data set. We deploy five methods in our experiments: in-degree ranking, weighted PageRank algorithm with weights normalized, weighted PageRank algorithm with weights unnormalized, weighted HITS algorithm and weighted Hubavg algorithm. Table 2 provides a mapping between each algorithm and its abbreviation, which will be used frequently in the following discussions. A reference between programs mentioned in this paper and their full names can be found in our shared data[9].

5.1 Graph extended or subtracted? Self-edges removed or retained?

Considering the entire Top50 CS data set, we have 182 schools and 1,106 edges. The total weight is 2018 in the extended CS *hiring graph*. We generate a subtracted graph that only contains the top 50 schools. In the subtracted graph, we have 50 schools and 842 edges; the total weight is 1740. In addition, as we know that there are self-edges in the graph, we also compared the differences between the one with self-edges and the one without self-edges.

Table 3 shows the results of our algorithms compared with U.S. News Ranking using *RankDist* measurement. According to the definition of $RankDist$, given a set of 50 samples, the maximum $RankDist$ between two rankings we can get is 25, which occurs when one is exactly the reverse of the other one. Another common case is that when we randomly shuffle the ranking, we get a $RankDist$ 16.63, averaged by 1000 trials of random shuffles. In Table 3, column 1 represents the algorithms that we use; column 2 shows the *RankDist* to U.S. News ranking when we employ the algorithm on the *subtracted graph with self-edges retained*; column 3 shows the *RankDist* to U.S. News ranking when we employ the algorithm on the *subtracted graph with self-edges removed*; column 4 shows the *RankDist* to U.S. News ranking when we employ the algorithm on the *extended graph with self-edges retained*; column 5 shows the *RankDist* to U.S. News ranking when we employ the algorithm on the *extended graph with self-edges removed*. The last column and the last row show the average of each row and each column respectively.

We can see from Table 3 that the *RankDist* values are not much different for HITS-based algorithms and IndeRank whether we consider extended or subtracted graph. However, the PageRank-based algorithms have smaller *RankDist* values with extended graphs.

By comparing column 2 and column 3, we can see that, except WeightedPR_w_n, $RankDist$ values in column 3 are all smaller than those in column 2, indicating that results obtained from the graph without self-edges are generally closer to the U.S. News ranking compared with the graph

Table 2: Algorithms and their Abbreviations

Algorithm	Abbreviation
In-degree Ranking Algorithm	IndeRank
Weighted PR Algorithm with weights normalized	WeightedPR_w_n
Weighted PR Algorithm with weights unnormalized	WeightedPR_wo_n
Weighted HITS Algorithm	HITS_Weighted
Weighted Hubavg Algorithm	HITS_Hubavg

Table 3: Results on Top50 CS Data Set

	RankDist to the U.S. News Ranking				
Algorithm	Subtracted graph with self-edges	Subtracted graph w/o self-edges	Extended graph with self-edges	Extended graph w/o self-edges	**Average**
Max.	25.0	25.0	25.0	25.0	25.0
RandomShuffle	16.63	16.63	16.63	16.63	16.63
IndeRank	5.04	3.92	5.00	4.08	**4.51**
WeightedPR_w_n	5.28	5.04	5.08	4.72	**5.05**
WeightedPR_wo_n	5.00	4.44	4.76	3.92	**4.53**
HITS_Weighted	4.72	4.20	4.72	4.16	**4.45**
HITS_Hubavg	4.44	3.92	4.40	3.88	**4.16**
Average	**4.896**	**4.304**	**4.792**	**4.152**	—

with self-edges. By comparing column 4 and column 5, we can observe a similar fact like this. The above observations indicate that removing self-edges in the CS *hiring graph* probably helps improve the performance of our algorithms. In the case of the *extended graph with self-edges removed*, HITS_Hubavg is performing the best with a $RankDist$ of 3.88, then comes WeightedPR_wo_n (3.92), IndeRank (4.08) and HITS_Weighted (4.16). In addition, HITS-based algorithms generally produce closer rankings to U.S. News than PR-based algorithms, probably because they consider the mutual reinforced effect from both hubs and authorities. What's more, WeightedPR_wo_n consistently gives smaller $RankDist$s than WeightedPR_w_n.

Given that we would like a program's rank to be not impacted directly by its own hiring decisions, we will consider *hiring graph* with self-edges *removed* from now on.

5.2 Original Rankings

Table 4 shows the resulting rankings obtained from all five proposed algorithms along with the U.S. News ranking. We note that all the results in Table 4 are retrieved from the experiments on the ***extended*** *graph with self-edges* ***removed*** from the entire CS data set. We also note that the U.S. News gives the same rank for some programs, which could be ranked in any order. In addition, programs with the same *in-degree* could be ranked in any order in IndeRank ranking.

A number of observations can be made from the results. MIT, CMU, Stanford and UCBerkeley always occupy the top 4 schools in the rankings. What's more, CMU seems to be a little bit over-ranked by U.S. News and MIT stands out in all our five algorithms.

By comparing the results from WeightedPR_w_n and WeightedPR_wo_n, the rankings of some schools are dramatically different. UIUC is ranked No. 5 in WeightedPR_wo_n while No. 8 in WeightedPR_w_n; Harvard is ranked No. 5 in WeightedPR_w_n while No. 8 in WeightedPR_wo_n. UCLA and Caltech are also ranked differently by these two algorithms. UCLA is ranked No. 13 in WeightedPR_wo_n while No. 18 in WeightedPR_w_n; Caltech is ranked No 11 in WeightedPR_w_n while No. 15 in WeightedPR_wo_n. This is to say that large programs, like UIUC and UCLA, are ranked higher in WeightedPR_wo_n, while smaller

programs like Harvard and Caltech are ranked higher in WeightedPR_w_n. As we have discussed in Section 3.1, it is the actual number of incoming edges and the actual value of corresponding weights that matter in WeightedPR_wo_n, in which case those large programs which place lots of Ph.D.s might have an advantage. In short, for PR-based algorithms, normalization favors smaller programs while unnormalization favors bigger programs, as expected. Considering WeightedPR_wo_n yields a closer result to U.S. News' ranking, we can conclude that the U.S. News probably favors bigger programs as well.

What's more, the HITS-based algorithms, HITS_Weighted and HITS_Hubavg seem to give very similar rankings according to Table 4. This is probably because HITS-based algorithms are more stable than PR-based algorithms since HITS-based algorithms take the effects from both hubs and authorities into consideration.

From Table 4 we can also observe that there are some programs with huge divergences between our rankings and U.S. News ranking, such as Harvard, Duke, StonyBrook, UMass and Utah. We will discuss some of these cases in detail later.

5.3 Recent Years *vs* Earlier Years

Here we compare the results obtained from recent data with results from earlier years data. As we described in Section 2.1, more than 80 percent of the entries have *Year Grad.* information in our data. By generating the Cumulative Distribution Function (CDF) of year distribution, the CDF curve crosses 50 percent between calendar year 1994 and 1995. In fact, before 1994 inclusively, there are 875 data points; after 1994 exclusively, there are 918 data points. The numbers are roughly equal and it would be fair to divide the data set by year 1994 into two equally large subsets to analyze the effect of year of graduation.

Table 5 shows the comparison between the results of recent years and earlier years. In Table 5, column 2 shows the resulting $RankDist$s obtained on the entire data set; column 3 shows the resulting $RankDist$s obtained on the data set from 1949 to 1994; column 4 shows the resulting $RankDist$s applied on the data set from 1995 to 2014.

The $RankDist$ values in column 4 are all smaller than those in column 3, indicating that recent year data reflects the U.S. News ranking better than earlier year data. In the future, we plan to employ a weight differential model based on the year of hiring to make the rankings more sensitive to recent year data.

Figure 2 shows the ranking divergence of all CS programs obtained from recent data. Compared with Table 4, in recent data ranking, Harvard is no longer ranked higher than U.S. News and Yale drops down greatly in our recent data ranking.

5.4 Observations

In order to know where the differences between U.S. News ranking and our rankings come from, we investigate into the actual rank difference for each program in our results. We use

Table 4: Resulting Rankings Obtained from Top50 CS Data Set

	The rankings are retrieved from experiments on the entire ***extended graph with self-edges removed***					
Rank	USNews	IndeRank	WeightedPR_w_n	WeightedPR_wo_n	HITS_Weighted	HITS_Hubavg
1	cmu	mit	mit	mit	mit	mit
2	mit	ucberkeley	stanford	ucberkeley	ucberkeley	ucberkeley
3	stanford	stanford	ucberkeley	stanford	stanford	stanford
4	ucberkeley	cmu	cmu	cmu	cmu	cmu
5	uiuc	uiuc	**harvard**	**uiuc**	uiuc	uiuc
6	cornell	cornell	cornell	cornell	washington	cornell
7	washington	princeton	washington	washington	cornell	washington
8	princeton	washington	**uiuc**	**harvard**	princeton	princeton
9	gatech	utaustin	princeton	princeton	**harvard**	**harvard**
10	utaustin	**harvard**	wisconsin	utaustin	ucla	utaustin
11	**caltech**	upenn	**caltech**	wisconsin	upenn	upenn
12	wisconsin	wisconsin	utaustin	upenn	wisconsin	wisconsin
13	ucla	ucla	upenn	**ucla**	utaustin	ucla
14	umich	gatech	**umass**	gatech	**caltech**	**caltech**
15	columbia	umaryland	gatech	**caltech**	**umass**	gatech
16	ucsd	**purdue**	umich	umaryland	gatech	**umass**
17	umaryland	**caltech**	yale	**purdue**	umich	umaryland
18	**harvard**	**umass**	**ucla**	**umass**	umaryland	umich
19	upenn	umich	ucsd	umich	ucsd	columbia
20	brown	columbia	columbia	columbia	columbia	**purdue**
21	**purdue**	usc	**purdue**	ucsd	yale	ucsd
22	rice	ucsd	umaryland	yale	**purdue**	yale
23	usc	northcarolina	northcarolina	usc	usc	nyu
24	yale	yale	**stonybrook**	northcarolina	nyu	usc
25	**duke**	nyu	uminnesota	nyu	northcarolina	northcarolina
26	**umass**	brown	nyu	brown	**stonybrook**	**stonybrook**
27	northcarolina	**stonybrook**	brown	**stonybrook**	brown	brown
28	johnshopkins	uminnesota	**utah**	uminnesota	pennstate	rice
29	nyu	rice	usc	rice	uminnesota	uminnesota
30	pennstate	pennstate	uvirginia	ohiostate	ohiostate	ohiostate
31	ucirvine	ohiostate	rice	pennstate	ucirvine	pennstate
32	uminnesota	**utah**	ohiostate	**utah**	**utah**	**utah**
33	uvirginia	northwestern	pennstate	uvirginia	northwestern	uvirginia
34	northwestern	ucirvine	johnshopkins	ucirvine	uvirginia	ucirvine
35	ohiostate	uvirginia	ucirvine	northwestern	rutgers	northwestern
36	rutgers	johnshopkins	northwestern	johnshopkins	rice	johnshopkins
37	ucdavis	rutgers	uchicago	rutgers	johnshopkins	rutgers
38	ucsb	uarizona	ucolorado	uarizona	ucolorado	uarizona
39	uchicago	ucolorado	dartmouth	ucolorado	uarizona	ucolorado
40	dartmouth	uchicago	rutgers	uchicago	uchicago	uchicago
41	**stonybrook**	**duke**	**duke**	**duke**	ucdavis	**duke**
42	tamu	ucsb	uarizona	ucsb	**duke**	ucdavis
43	uarizona	ucdavis	boston	ucdavis	ucsb	wustl
44	ucolorado	wustl	wustl	boston	wustl	dartmouth
45	**utah**	boston	ucsb	wustl	dartmouth	ucsb
46	vatech	dartmouth	ucdavis	dartmouth	boston	boston
47	wustl	ncstate	tamu	tamu	ncstate	ncstate
48	arizonastate	tamu	ncstate	ncstate	tamu	tamu
49	boston	arizonastate	arizonastate	arizonastate	arizonastate	arizonastate
50	ncstate	vatech	vatech	vatech	vatech	vatech

Table 5: Results between Recent Years and Earlier Years on CS Data Set

	RankDist to the U.S. News Ranking on extended graph w/o self-edges		
Algorithm	Entire Data	1949∼1994	1995∼2014
IndeRank	4.08	6.28	4.28
WeightedPR_w_n	4.72	6.44	4.68
WeightedPR_wo_n	3.92	6.04	4.20
HITS_Weighted	4.16	6.42	5.00
HITS_Hubavg	3.88	6.12	4.64

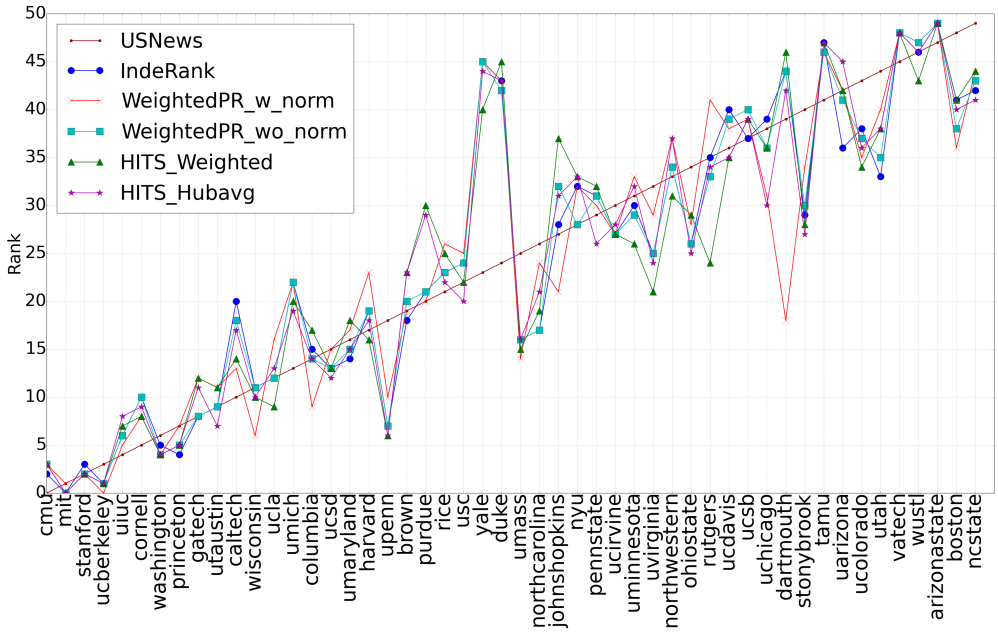


Figure 2: Ranking Divergence of CS Programs Compared to U.S. News (1995-2014)

WeightedPR_wo_n and HITS_Hubavg for analysis because these two algorithms seem to be doing better than other algorithms according to our previous discussion.

Table 6 shows the exact difference for some of the programs in WeightedPR_wo_n ranking and HITS_Hubavg ranking compared with U.S. News. The positive value means our rank is higher than the rank in U.S. News; the negative value means our rank is lower than the rank in U.S. News. The $AbsDif$ value is the absolute difference between the value in '49 $\sim$ '94 and the value in '95 $\sim$ '14.

The first block, consisting of Yale, Purdue, Harvard and NYU, contains the programs that were doing much better before 1994 than they did after 1994. Part of the reason could be that they are old programs, who have established their academic strengths in the earlier days. Another reason could be that they have fallen behind in recent years.

Harvard and Yale seem to be two representative examples of such programs. Table 7 shows the incoming edges of Harvard with year. Before 1994, 33 Ph.D.s from Harvard were hired widely at MIT, UCBerkeley, Cornell, Purdue and many other programs; After 1994, only 11 Ph.D.s from

Table 6: Rank Difference Comparison on CS Data Set

	WeightedPR_wo_n				*HITS_Hubavg*			
Univ	Entire	'49∼'94	'95∼'14	AbsDif	Entire	'49∼'94	'95∼'14	AbsDif
Yale	+2	+12	-22	34	+2	+12	-21	33
NYU	+4	+12	0	12	+6	+16	-5	21
Purdue	+4	+8	-1	9	+1	+5	-9	14
Harvard	+10	+11	-2	13	+9	+12	-1	13
UCSD	-5	-19	+2	21	-5	-19	+3	22
Gatech	-5	-20	0	20	-6	-24	-3	21
Rice	-7	-16	-2	14	-6	-16	-1	15
Columbia	-5	-9	0	9	-4	-10	0	10
Utah	+13	+17	+9	8	+13	+18	+6	12
Duke	-16	-8	-18	10	-16	-9	-19	10
StonyBrook	+14	+20	+10	10	+15	+21	+13	8
Caltech	-4	-7	-8	1	-3	-6	-7	1
UIUC	0	0	-2	2	0	0	-4	4
Stanford	0	0	0	0	0	+1	0	1
UTAustin	0	+1	0	1	0	+1	+2	1
MIT	+1	+1	+1	0	+1	+1	+1	0

Harvard got hired. For another example, Table 8 shows the incoming edges of Yale with year. We can see that Yale's Ph.D.s were hired widely among Princeton, UCLA and some other schools before 1994, while after 1994, there was only one Ph.D. from Yale currently at Cornell. Since these programs did not place as many of their graduates into other programs in recent years, their ranks fall down substantially when we look at recent data.

The second block, including Gatech, UCSD, Rice and Columbia, contains the programs that are ranked lower before 1994 but have improved in standings recently. This is probably because they are young programs and grew fast in the recent years.

The third block includes those programs that are either "under-estimated" or "over-estimated" by U.S. News. For example, in our case, StonyBrook and Utah are under ranked by U.S. News while Duke and Caltech are over ranked by U.S. News.

The fourth block consists of those programs that are ranked similarly in both our rankings and U.S. News ranking, such as UIUC, Stanford, UTAustin and MIT.

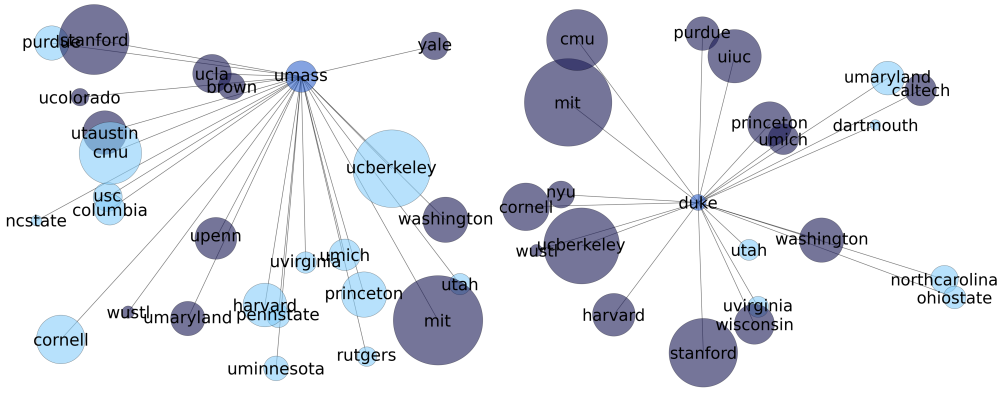
These observations are not coincidental but all reflected from the *hiring graph*. Here is an example. Duke and UMass are both ranked No. 25 in U.S. News ranking. However, they are performing differently in placing their Ph.D.s in the academia. Figure 3a shows the neighbours of UMass in the *hiring graph*; Figure 3b shows the neighbors of Duke in the *hiring graph*. We can see in Figure 3a that CMU, UCBerkeley, Princeton, Cornell, Harvard and some other schools (Light Nodes) have hired Ph.D.s from UMass. On the other hand, in Figure 3b, Utah, UVirginia, UMaryland, Dartmouth, NorthCarolina and OhioState (Light Nodes) have hired Ph.D.s from Duke. Since these programs are not as highly ranked as the programs that hired UMass Ph.D.s, UMass gets ranked higher in our approach.

Table 7: Incoming Neighbours of Harvard in CS Data Set

Harvard's Incoming Nodes							
Univ.	Year	Univ.	Year	Univ.	Year	Univ.	Year
NYU	1950	UMaryland	1970	Caltech	1980	Yale	1968
NorthCarolina	1956	Duke	1970	Cornell	1981	USC	1969
UCBerkeley	1959	NYU	1970	MIT	1984	Wustl	1978
UCLA	1963	MIT	1972	UMaryland	1985	Stanford	1980
Purdue	1963	Princeton	1973	Dartmouth	1986	Columbia	1993
Yale	1965	Harvard	1974	Gatech	1989	UPenn	1993
UMass	1966	UArizona	1974	UPenn	1989		
NorthCarolina	1967	StonyBrook	1976	Boston	1992		
UCDavis	1967	Duke	1977	CMU	1993		
UIUC	1995	StonyBrook	2003	Duke	2008	StonyBrook	1998
UCLA	1996	Boston	2003	Northwestern	2012	Harvard	2007
Cornell	1997	ArizonaState	2005	ArizonaState	2012		

Table 8: Incoming Neighbours of Yale in CS Data Set

Yale's Incoming Nodes							
Univ.	Year	Univ.	Year	Univ.	Year	Univ.	Year
Dartmouth	1975	UCLA	1982	Northwestern	1986	NYU	1980
UMass	1977	UMaryland	1982	USC	1987	Northwestern	1986
CMU	1979	Princeton	1986	NYU	1988	Rutgers	1994
Princeton	1980	Northwestern	1986	UPenn	1994		
Cornell	2005						



(a) UMass (b) Duke

Figure 3: One-level Neighbouring Graphs in CS Data Set

5.5 Sensitivity Analysis

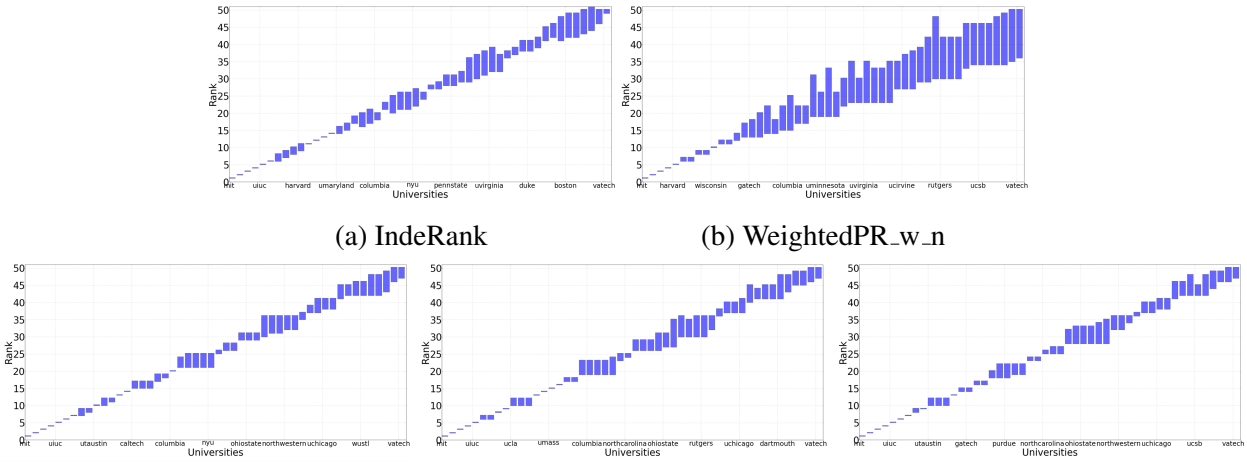
Our expectation is that, one or two faculty members coming or leaving the department should not affect the rank of the department dramatically. Any change in rankings from such small changes in the *hiring graph* is considered to provide an idea of fidelity of rankings. To measure the sensitivity of program's rank, we add a "*virtual*" edge from the # 1 program (e.g., *MIT* in CS data) to that program, which means that MIT just hired a Ph.D. from that program; if there is already an edge from MIT to that program, we increase the edge weight by 1. In a second experiment, we delete an existing edge from the highest ranked program to that program. If the target edge has a weight more than 1, we decrease the edge weight by 1; if the target edge has a weight exactly as 1, we remove the edge from the graph; if the program doesn't have any incoming edge at all, we delete one outgoing edge pointing to the best program from that program. The reason we perform these two manipulations is that we have already seen that the quality and quantity of incoming edges play an essential role in the ranking of programs.

Figure 4 shows the sensitivity bound of each program by all our five algorithms. In Figure 4, the x axis represents the programs ordered by the rank from 1 to 50; the y axis represents the ranks. In these figures, each program has a bar that represents its sensitivity variation bound. The bottom of the bar represents the upper bound, or how high it could be ranked when adding a virtual significant edge; the top of the bar represents the lower bound, or how low it could be ranked when deleting a significant edge to that program. Thus, the narrower the variation bound is, the less sensitive that program's ranking is to minor changes in the *hiring graph*.

In Figure 4a, IndeRank generally has a small variation bound for each program. However, the greatest disadvantage of IndeRank is that IndeRank is not able to rank those programs with the same number of incoming edges. This is because IndeRank only considers the quantity of edges regardless of the quality of edges. As an example, the in-degree of Caltech is 26 and in-degree of Purdue is 30 in our extended *hiring graph* without self-edges. For IndeRank, Purdue is ranked higher than Caltech. However, in the *hiring graph*, MIT, UCBerkeley, Stanford and many other highly-ranked programs hire Ph.D.s from Caltech, while this is not the case for Purdue. The edge quality of Caltech is better than Purdue. As a result, all other four algorithms in our approach rank Caltech higher than Purdue (For example, in HITS_Weighted, Caltech is ranked #14 while Purdue is ranked #22).

In Figure 4b, WeightedPR_w_n looks very sensitive to graph changes because the upper bounds for the lower ranked programs are extremely wide. This happens for two reasons. First, PageRank only cares about the *authority*, which brings up the *authority* of that program instantly when adding an edge to that program pointed by another well established *authority*. Second, adding a high quality incoming edge provides a major contribution to that program because of normalization.

The performances of the other three algorithms are similar in terms of the sensitivity analysis. They all have small variation bounds, indicating that they are less sensitive. In addition, we can observe a "step-like" shape from Figure 4c to Figure 4e, indicating that some programs share either upper bound, lower bound or both. It is a good indicator that these programs could probably be ranked together.



(a) IndeRank (b) WeightedPR_w_n

(c) WeightedPR_wo_n (d) HITS_Weighted (e) HITS_Hubavg

Figure 4: Sensitivity Graphs on CS Data Set

Table 9: Average Sensitivity Bounds of all Algorithms on CS Data Set

Algorithm	UpperBound	LowerBound	Abs.Range
IndeRank	+1.54	-1.54	3.08
WeightedPR_w_n	+5.76	-1.98	7.74
WeightedPR_wo_n	+1.54	-0.92	2.46
HITS_Weighted	+1.60	-1.24	2.84
HITS_Hubavg	+1.40	-1.16	2.56

Table 9 summarizes the average sensitivity bounds for all the algorithms. The *UpperBound* indicates the average boost-up of all programs; the *LowerBound* indicates the average degradation of all programs; the *Abs.Range* is the absolute difference between *UpperBound* and *LowerBound*. The *UpperBound* of WeightedPR_w_n (5.76) is extremely high, which is consistent with our analysis on the sensitivity graph. According to Table 9, Weighted_wo_norm, HITS_Weighted and HITS_Hubavg seem to offer a better distinction of programs.

6 Results on Top50 ME Data Set

Our proposed approach is seen to work well on our Top50 CS data set. We employ another data set of CS hirings collected independently [15]. The results on that data set are consistent with the rankings obtained based on our data collection, showing that there are no significant gaps in our data collection. WeightedPR_wo_n, HITS_Weighted and HITS_Hubavg are doing well in terms of both $RankDist$ to U.S. News and sensitivity analysis. We also discover interesting patterns from our data. In order to validate our approach, we re-run the same experiments on a completely different data set—the top 50 ME data set. If our approach is robust, we should expect similar results from the ME data set.

We examine how our approach performs in the four cases discussed in Section 5.1 on our ME data set. Table 10 shows the comparisons among these cases.

Table 10: Results on Top50 ME Data Set

	RankDist to the U.S. News Ranking			
Algorithm	Subtracted graph with self-edges	Subtracted graph w/o self-edges	Extended graph with self-edges	Extended graph w/o self-edges
IndeRank	5.36	4.88	5.52	4.96
WeightedPR_w_n	6.84	6.80	6.60	6.04
WeightedPR_wo_n	6.08	5.52	5.08	5.00
HITS_Weighted	4.48	5.12	4.48	5.12
HITS_Hubavg	4.84	5.04	4.80	4.84

Table 11: Results between Recent Years and Earlier Years on ME Data Set

	RankDist to the U.S. News Ranking on extended graph w/o self-edges		
Algorithm	Entire Data	1946∼1990	1991∼2013
IndeRank	4.96	7.12	5.60
WeightedPR_w_n	6.04	7.84	5.60
WeightedPR_wo_n	5.00	7.4	5.52
HITS_Weighted	5.12	7.76	6.00
HITS_Hubavg	4.84	7.36	5.68

As we can see in Table 10, results obtained from the *extended graph without self-edges* are the best among the four, which is consistent with the CS results. The best $RankDist$ we achieved is from HITS_Weighted, which is 4.48. HITS_Hubavg (4.8), IndeRank (4.88) and WeightedPR_wo_n (5.0) also yield rankings close to U.S. News ranking. On average, results from the *extended graph without self-edges* are the smallest. One thing we notice is that the $RankDist$ in the ME data set is slightly larger than that in the CS data set.

For Top50 ME data set, we also compare the cases between earlier years and recent years. In ME data set, the earliest year is 1946 and the latest year is 2013. the CDF curve of year distribution in ME data set crosses 50 percent between calendar year 1990 and 1991. Table 11 shows the comparison between the results from the recent and earlier years data.

We can see clearly from Table 11 that the result is consistent with the result obtained from the CS data set. The rankings obtained from the years between 1991 and 2013 are generally closer to the U.S. News than the rankings obtained from the years before 1991. It again proves that recent data reflects the U.S. News ranking better.

Table 12 summarizes the average sensitivity variation bound for each algorithm deployed on the ME data set. As we can see, HITS_Weighted has the smallest variation bound as $Abs.Range = 1.88$, then follow HITS_Hubavg $(Abs.Range = 2.12)$ and WeightedPR_wo_n $(Abs.Range = 3.28)$.

Table 12: Average Sensitivity Bounds of all Algorithms on ME Data Set

Algorithm	UpperBound	LowerBound	Abs.Range
IndeRank	+2.44	-2.42	4.86
WeightedPR_w_n	+7.50	-2.34	9.84
WeightedPR_wo_n	+1.86	-1.42	3.28
HITS_Weighted	+0.46	-1.42	1.88
HITS_Hubavg	+0.80	-1.32	2.12

7 Cross-Domain University Graduate Program Ranking Model

A CS program may not always hire Ph.D.s from other CS programs. It is possible that a CS program may hire Ph.D.s from other programs such as Electrical Engineering and Math, for example. In order to take this into account, we can carry out something like "cross-domain" ranking, consolidating different *hiring graphs* of different programs into one. We will show how this can be carried out here. More extensive data collection of several programs would be needed to carry out this comprehensively. Given the fact that, though not the majority, cross-domain hiring exists in the *hiring graph*, we propose our cross-domain university graduate program ranking model based on our previous model. When considering only the cross-field effect, for every school p in *hiring graph* $G = (V, E)$, the ranking of school p is actually a set of ranking scores of p in multiple fields:

$$r_p = \langle r_p(f_1), r_p(f_2), \cdots, r_p(f_m) \rangle, \tag{11}$$

where $f_1, f_2, \cdots, f_m$ are all the programs in school p. Taking HITS_Weighted as an example, the new updating rules become:

$$Auth_p(f_i) \leftarrow \sum_{q(f_j) \in M(p(f_i))} Hub_q(f_j) \cdot w(\varepsilon(q(f_j), p(f_i))), \tag{12}$$

where $M(p(f_i))$ is the set of incoming neighbors of $p(f_i)$; and

$$Hub_p(f_i) \leftarrow \sum_{q(f_j) \in O(p(f_i))} Auth_q(f_j) \cdot w(\varepsilon(p(f_i), q(f_j))), \tag{13}$$

where $O(p(f_i))$ is the set of outgoing neighbors of $p(f_i)$.

When different domains are treated equally, in each iteration, $r_p(f_i)$ should be normalized such that $\sum_{p \in \mathbb{N}(f_i)} r_p(f_i) = N(f_i)$, where $N(f_i)$ is the total number of nodes in f_i and $\mathbb{N}(f_i)$ denotes all the programs in f_i. When different domains are treated differently, the normalization should be adjusted accordingly for each f_i.

Here is an example showing how this works. We collect faculty data from top 50 Electrical Engineering(EE) programs[16] in U.S. News, along with their Ph.D. degrees. In our EE data set, 219 out of 4,484 EE faculty members were CS Ph.D.s. Figure 5 shows the difference between the original CS ranking and the new CS ranking when considering the EE effect. As we can see, Vatech boosts 11 ranks higher after considering the EE hirings. This is because EE programs in

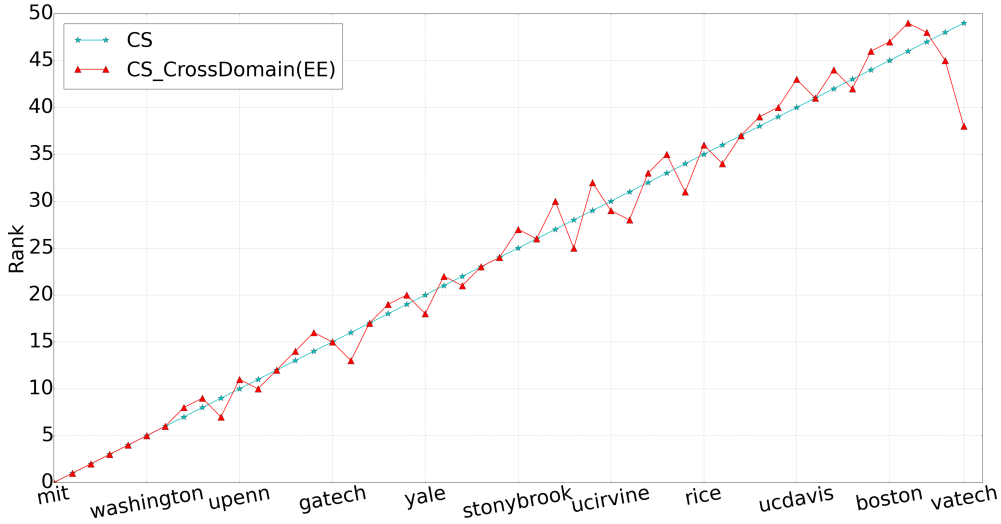


Figure 5: Cross-Field Effect from EE on CS Ranking given by HITS_Weighted

UIUC and CMU hire 3 CS Ph.D.s from Vatech in our new cross-domain data, making Vatech ranked higher in the cross-domain model. However, due to the lack of data, we are not able to comprehensively analyze and validate the cross-domain graduate program ranking model.

8 Rankings with Tiles

In order to let our ranking more time-sensitive, we propose assigning larger weights to recent hires while assigning smaller weights to older hires. The intuition behind *weight differentiation adjustment* is to let our ranking model more sensitive to recent changes in the *hiring graph*. For example, if the hiring decision is made after 2000, the adjusted weight $W_{new} = 1.0 \times W_{old}$; if the hiring decision is made between 1991 and 2000, the adjusted weight $W_{new} = 0.8 \times W_{old}$; if the hiring decision is made between 1981 and 1990, the adjusted weight $W_{new} = 0.6 \times W_{old}$; ...; etc. We note that the weight adjustment scheme is a subjective point of view. We also note that the granularity of weight differentiation could be adjusted accordingly. For example, if we would like the ranking to measure the program quality recently, we should decrease the weight more drastically over years; If we would like the ranking to reflect the historical momentum of programs in a few decades, we probably should have less drastic weighting differentiation scheme. Weight differentiation scheme makes our ranking model more robust given the fact that *hiring graph* keeps changing over years. The result discussed in this section is based on a "less drastic" weight differentiation model, which is described as follows: if the hiring is made after 2000, the adjusted weight $W_{new} = 1.0 \times W_{old}$; if the hiring is made before 2000, the adjusted weight $W_{new} = 0.8 \times W_{old}$.

In addition, we consider the cross domain effects among our three data sets: Top 50 CS, Top 50 ME and Top 50 EE in rendering our proposed ranking. In our cross-domain data set, 180 EE Ph.D.s and 21 ME Ph.D.s joined CS programs; 219 CS Ph.D.s joined EE programs; 44 CS Ph.D.s and 96 EE Ph.D.s joined ME programs. We note that, in our evaluation, we ignore the

cross-domain from other fields, Math and Physics for example, even though there are a few such links present in our data.

Finally, like what U.S. News does, we introduce tiles in our rankings. We assign some programs with the same rank according to our sensitivity analysis. The observation that some programs have the same upper bound or lower bound or both is a good indicator of the tiles of programs. Our ordering strategy is described in Algorithm 1.

Algorithm 1 Tiles Ordering Strategy

p_i: Program ranked as i
p_{i+1}: Program ranked as $i+1$
$R(p)$: The rank of program p
$UB(p)$: Upper bound of program p
$LB(p)$: Lower bound of program p
if $UB(p_i) \geq UB(p_{i+1})$ **then**
  if $LB(p_i \geq LB(p_{i+1}))$ **then**
    $R(p_{i+1}) = R(p_i) = i$;
  else
    the order remains;
  end if
else
  the order remains;
end if

Table 13 combined with Table 14 shows our final rankings with tiles. In Table 13 and Table 14, we only present the results obtained from *HITS_Hubavg* and *WeightedPR_won* because they seem doing better in the previous analysis.

Table 13: Ranking Results with Tiles (Part I)

The rankings are retrieved from the entire ***extended graph with self-edges removed*** with ***weight differentiation and cross-domain effects applied***											
CS				ME				EE			
HITS_Hubavg		WeightedPR_won		HITS_Hubavg		WeightedPR_won		HITS_Hubavg		WeightedPR_won	
RankDist = 3.7		RankDist = 4.12		RankDist = 4.86		RankDist = 4.94		RankDist = 5.3		RankDist = 4.9	
univ.	rank	univ.	rank	univ.	rank	univ.	rank	univ.	rank	univ.	rank
mit	1	mit	1	ucberkeley	1	ucberkeley	1	ucberkeley	1	mit	1
ucberkeley	2	ucberkeley	2	stanford	2	stanford	2	stanford	2	ucberkeley	2
stanford	3	stanford	3	mit	3	mit	3	mit	2	stanford	3
cmu	4	cmu	4	caltech	4	caltech	4	uiuc	4	uiuc	4
uiuc	5	uiuc	5	uiuc	5	uiuc	5	caltech	5	umich	5
cornell	6	cornell	6	umich	6	umich	6	umich	6	caltech	6
washington	7	washington	7	cornell	7	cornell	7	princeton	7	princeton	7
princeton	7	princeton	7	purdue	8	princeton	8	cornell	8	cornell	8
utaustin	9	utaustin	9	princeton	8	purdue	8	ucla	9	ucla	9
wisconsin	9	wisconsin	10	harvard	10	gatech	10	cmu	10	cmu	10
harvard	11	harvard	11	gatech	11	harvard	10	purdue	11	purdue	11
upenn	12	upenn	11	ucla	12	ucla	12	usc	12	gatech	12
ucla	13	ucla	13	wisconsin	13	utaustin	13	gatech	12	usc	13
gatech	14	gatech	13	uminnesota	14	wisconsin	13	harvard	12	harvard	13
umaryland	15	umaryland	15	utaustin	14	uminnesota	15	ucsb	15	wisconsin	15
caltech	15	columbia	16	johnshopkins	16	pennstate	15	wisconsin	16	ucsb	15
columbia	17	umass	16	pennstate	16	upenn	15	umaryland	16	umaryland	15
umass	17	purdue	16	upenn	16	johnshopkins	18	ucsd	16	utaustin	18
ucsd	17	caltech	16	northwestern	19	northwestern	18	utaustin	19	rice	19
umich	20	ucsd	20	cmu	19	cmu	18	rice	19	ucsd	19
purdue	20	umich	21	ucsd	21	ucsd	21	columbia	21	columbia	21
usc	22	northcarolina	21	vatech	21	vatech	22	uminnesota	22	uminnesota	22
nyu	22	usc	23	washington	23	washington	23	northwestern	22	ucolorado	23
northcarolina	22	nyu	24	ohiostate	23	columbia	23	upenn	22	northwestern	23
yale	25	uminnesota	25	michstate	25	ohiostate	23	ucolorado	25	ohiostate	23

Table 14: Ranking Results with Tiles (Part II)

The rankings are retrieved from the entire ***extended graph with self-edges removed*** with ***weight differentiation and cross-domain effects applied***											
CS				ME				EE			
HITS_Hubavg		WeightedPR_won		HITS_Hubavg		WeightedPR_won		HITS_Hubavg		WeightedPR_won	
RankDist = 3.7		RankDist = 4.12		RankDist = 4.86		RankDist = 4.94		RankDist = 5.3		RankDist = 4.9	
univ.	rank	univ.	rank	univ.	rank	univ.	rank	univ.	rank	univ.	rank
stonybrook	26	yale	25	columbia	25	tamu	26	washington	26	upenn	23
uminnesota	27	uvirginia	25	ucsb	25	michstate	27	umass	26	duke	27
pennstate	27	stonybrook	25	rensselaer	25	ucsb	28	wustl	26	wustl	27
uvirginia	27	brown	25	yale	29	rensselaer	29	ohiostate	26	washington	29
rice	27	rice	30	tamu	29	duke	30	duke	30	umass	29
ucirvine	27	pennstate	31	duke	31	uvirginia	30	johnshopkins	31	johnshopkins	31
brown	27	ucirvine	32	ucdavis	32	yale	30	uflorida	32	tamu	31
johnshopkins	33	johnshopkins	33	uvirginia	33	iowastate	33	rutgers	32	rensselaer	33
northwestern	34	ohiostate	33	ucirvine	34	ucirvine	34	vatech	32	uflorida	34
utah	34	northwestern	35	rice	34	rice	34	rensselaer	32	vatech	34
ohiostate	36	utah	35	iowastate	34	ucdavis	34	northeastern	36	northeastern	36
rutgers	36	rutgers	35	umaryland	37	umaryland	37	tamu	36	rutgers	36
uchicago	38	duke	38	ncstate	38	ncstate	38	notredame	38	notredame	38
duke	39	uarizona	38	lehigh	39	lehigh	39	uvirginia	39	ncstate	39
uarizona	39	ucolorado	38	rutgers	40	uflorida	40	arizonastate	39	uvirginia	40
ucolorado	39	uchicago	38	uflorida	41	wustl	40	ncstate	39	arizonastate	40
ucdavis	39	ucdavis	42	wustl	41	arizonastate	40	pennstate	42	boston	42
wustl	43	wustl	43	usc	41	case	43	ucdavis	43	brown	42
ucsb	44	boston	43	arizonastate	41	ucolorado	43	boston	43	pennstate	44
boston	45	ucsb	43	drexel	41	rutgers	43	uarizona	45	ucdavis	44
dartmouth	46	dartmouth	46	case	41	usc	46	brown	45	uarizona	46
ncstate	46	ncstate	46	ucolorado	41	notredame	46	case	45	ucirvine	46
tamu	48	tamu	48	notredame	48	drexel	46	iowastate	48	case	46
arizonastate	48	arizonastate	49	vanderbilt	49	vanderbilt	49	ucirvine	49	iowastate	46
vatech	48	vatech	50	dartmouth	50	dartmouth	50	utah	50	utah	50

9 Discussion

As we have shown, our algorithmic approach produces objective and reliable rankings for graduate programs across multiple fields. We also introduce cross-domain adjustment and ranking with tiles into our model, making our model more versatile and practical. We note that different algorithms produce different rankings while using the same data. *HITS_Hubavg* and *WeightedPR_won* are the two algorithms that not only give accurate ranks of programs but also stably produce reliable rankings of programs. Our intent is not to provide a new ranking of the programs, but to provide a new methodology for ranking the programs. We leave the choice of algorithm and the choice of weighting of recent hires over older hires to those interested in producing a ranking of the programs.

Program rankings provide information for aspiring students, universities, hiring and funding agencies about the relative quality, productivity and affordability of different programs. Our methodology here adds another perspective for ranking graduate programs. Our methodology provides insights about different programs' progression over time and the impact of the program's size on rankings.

10 Conclusion and Future Work

We proposed a new and alternative way to rank graduate programs using the hiring data of these programs. We have shown that our approach produces reasonable and reliable rankings for graduate programs. In addition, we have seen that our approach works across different fields. Moreover, by extensive data analysis, we not only discovered what is behind the *hiring graph* but also revealed valuable knowledge beyond U.S. News ranking.

The future work is to refine and improve the "cross-domain" graduate program ranking model with more data collection. We believe that the "cross-domain" effect across fields and countries matters in the *hiring graph*. We also plan to extend our work into industry hires, making our model more comprehensive and general. We plan to continue this work with more detailed data collection in the future.

References

[1] Sam Flanigan and Robert Morse. How u.s. news calculated the 2015 best graduate schools rankings @U.S. NEWS, March 2014. `http://www.usnews.com/education/best-graduate-schools/articles/2014/03/10/how-us-news-calculated-the-2015-best-graduate-schools-rankings`.

[2] Jung Cheol Shin, Robert K. Toutkoushian, and Ulrich Teichler. *University Rankings: Theoretical Basis, Methodology and Impacts on Global Higher Education*. Springer Verlag, 2011.

[3] Loet Leydesdorff and Jung C. Shin. How to evaluate universities in terms of their relative citation impacts: Fractional counting of citations and the normalization of differences among disciplines. *The American Society for Information Science and Technology*, 62(6):1146–1155, 2011.

[4] Francisco A Ponce and Andres M Lozano. Academic impact and rankings of american and canadian neurosurgical departments as assessed using the h index. *Neurosurgery*, 113(3):447–457, 2010.

[5] George Barnett and Thomas Hugh Feeley. Comparing the nrc and the faculty hiring network methods of ranking doctoral programs in communication. *Communication Education*, 60(3):362–370, 2011.

[6] Giseli Rabello Lopes, Mirella M. Moro, Roberto da Silva, Eduardo M. Barbosa, , and Jose Palazzo Moreira de Oliveira. Ranking strategy for graduate programs evaluation. In *IADIS International Conference WWW/Internet*, pages 253–260, 2011.

[7] Computer science graduate program ranking @U.S. NEWS, March 2014. `http://grad-schools.usnews.rankingsandreviews.com/best-graduate-schools/top-science-schools/computer-science-rankings`.

[8] Mechanical engineering graduate program ranking @U.S. NEWS, April 2014. `http://grad-schools.usnews.rankingsandreviews.com/best-graduate-schools/top-engineering-schools/mechanical-engineering-rankings?int=997808`.

[9] Shared data used in this paper @Google Drive, April 2014. `https://47628fb0e83f53b49f67da074251d0bc0fbbf061.googledrive.com/host/0Bwc8jMFBkXf4VzVkM08wQ1VPNnc/`.

[10] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web, 1999.

[11] Jon M. Kleinberg. Authoritative sources in a hyperlinked environment. *ACM*, 46(5):604–632, 1999.

[12] Taher H. Haveliwala. Topic-sensitive pagerank. In *In proceedings of Eleventh International Conference on World Wide Web*, 2002.

[13] Hema Dubey and Prof. B. N. Roy. An improved page rank algorithm based on optimized normalization technique. In *International Journal of Computer Science and Information technologies(IJCSIT)*, pages 2183–2188, 2011.

[14] Allan Borodin, Gareth O. Roberts, Jeffrey S. Rosenthal, and Panayiotis Tsaparas. Link analysis ranking: Algorithms, theory, and experiments. *ACM Transactions on Internet Technology (TOIT)*, 5(1):231–297, 2005.

[15] Publically available data collected by another cs researcher @blog.cs.brown.edu, April 2014. `http://blog.cs.brown.edu/2014/04/30/professor-jeff-huang-and-19-students-crowdsource-dataset-2200-computer-science-faculty/`.

[16] Electrical engineering graduate program ranking @U.S. NEWS, May 2014. `http://grad-schools.usnews.rankingsandreviews.com/best-graduate-schools/top-engineering-schools/electrical-engineering-rankings?int=9a0108`.