The Distributed Computing Column

BY

Stefan Schmid

Aalborg University
Selma Lagerlöfs Vej 300, DK-9220 Aalborg, Denmark

This time, the Distributed Computing Column features two articles:

1. Gianlorenzo D'Angelo presents an interesting optimization perspective on graph centralities and surveys the state-of-the-art approximation bounds. Gianlorenzo D'Angelo also receives the Best Young Italian TCS Researcher Award for 2016 of the Italian Chapter of the EATCS. Congratulations!

2. In an effort to shed light on the consequences of Artificial Intelligence and Computerization on employment, Philipp Brandes and Roger Wattenhofer present an interesting refinement of a seminal study by the economists Frey and Osborne. In particular, Brandes and Wattenhofer's probabilistic model accounts for the unique tasks of each job, allowing us to look inside the Frey/Osborne blackbox, and giving rise to a number of interesting insights and discussions.

Approximation bounds for centrality maximization problems

Gianlorenzo D'Angelo
Gran Sasso Science Institute (GSSI).
Viale F. Crispi, 7, 67100, L'Aquila, Italy.
gianlorenzo.dangelo@gssi.infn.it

Abstract

Determining what are the most important nodes in a network is one of the main problems in the field of network analysis. Several so-called *centrality indices* have been defined in the literature to try to quantitatively capture the notion of importance (or centrality) of a node within a network. It has been experimentally observed that being central for a node, according to some centrality index, leads to several benefits to the node itself.

In this paper, we study the problem of maximizing the centrality index of a given node by adding a limited number of edges incident to it. We survey on some recent results on this problem by focusing on four well-known centrality indices, namely harmonic centrality, betweenness centrality, eccentricity, and page-rank.

1 Introduction

In the past decades, there has been an increasing interest in the analysis of real-world complex networks in diverse research areas from sociology to computer science, going through biology and economy. Relevant examples of networks are autonomous-systems networks within the Internet, the World Wide Web, networks deriving from transportation infrastructures like roads or public transport, networked energy systems, social networks, coauthorship networks, and financial systems. It is somewhat surprising to observe that several networks originating from different contexts exhibit similar structural properties.

One of the most studied network properties goes under the name of *centrality* of a node in a network. Informally speaking, a node is considered "central" if it is important within the network and it is believed that the importance that a node has within a network reflects, to some extent, the position of the node in the

network and, more in general, the network structure. However, researchers do not agree on a common definition of centrality, instead several *centrality indices* have been proposed in the literature to try to quantitatively capture this notion. Most of the centrality indices are based on distances between nodes (like the *closeness centrality* [2]), on the number of shortest paths passing thorough a node (like the *betweenness centrality* [17]), or on spectral properties (like the *page-rank* [8]). For more details on centrality indices, see [5, 30]. What is the right definition of centrality of a node is not clear and the choice depends on the application domain.

On the other hand, it has been experimentally observed that being central for a node, according to some centrality index, has several benefits for the node itself. For example, closeness centrality is significantly correlated with citation counts of an author in author-citation networks [36], betweenness centrality is correlated with the efficiency of an airport in transportation networks [28], and both closeness and betweenness are correlated with the efficiency of an individual to propagate the information in a social network [27]. Therefore, a lot of research effort has been done on the problems of computing the centrality indices of a given node or determining the most central nodes of a network, according to some index.

In this paper we look at centrality indices from a *proactive* point of view, that is we want to *modify* an existing network with the aim of improving the centrality of a given node. A network can be modified by adding or removing edges and nodes. By performing these operations the centrality of a node can increase, while the centrality of other nodes can decrease. For example by adding edges, the distances between nodes decreases and hence the closeness centrality of some node increases, while by removing edges the closeness centrality might decrease.

Which "strategy" should a node adopt in order to increase its own centrality value as much as possible? In this paper we formulate this question as an optimization problem which consists in finding a limited amount of edges to be added in a graph in order to maximize the centrality of a given node within a network.

Generally speaking, adding edges incident to a given node v reduces the distances between v and the other nodes and hence it increases the centrality of v in some centrality indices. Moreover, looking at social networks from a user (node) perspective, it is not difficult to imagine scenarios in which a node can only add edges incident to itself and hence it is reasonable to consider such constraint in our optimization problem. More specifically, we consider the problem of efficiently determining, for a given vertex v, the set of k edges incident to v that, when added to the original graph, maximizes the centrality of v, according to some index. We denote this optimization problem as Centrality Maximization problem (CM). In this paper we survey some recent results on the CM problem in which we use four relevant centrality indices to be maximized: harmonic centrality, betweenness centrality, eccentricity and page-rank. The results outlined in this paper are reported in Table 1.

Structure of the paper. In the next section, we give the notation used in the paper, define the centrality indices that we aim at maximizing, and give the problem statement. In Section 3, we survey on the known results on the CM problem. Finally, in Section 4, we outline some future research directions that deserve further investigation.

2 Preliminaries

Let $G = (V, E)$ be a directed or undirected graph. For each node v, if G is directed, N_v^i and N_v^o denote the set of in-neighbors and out-neighbors of v, respectively, i.e. $N_v^i = \{u \mid (u, v) \in E\}$ and $N_v^o = \{u \mid (v, u) \in E\}$. If G is undirected, N_v denotes the set of all neighbors of v, $N_v = \{u \mid \{u, v\} \in E\}$. Given two nodes s and t, we denote by d_{st}, σ_{st}, and σ_{stv} the distance from s to t in G, the number of shortest paths from s to t in G, and the number of shortest paths from s to t in G that contain v, respectively. When we discuss about page-rank, we will assume that the graph is strongly connected.

2.1 Centrality indices

A *centrality index* c (also called *centrality metrics* or *centrality measures*) is a function $c : V \to \mathbb{R}$ that associates a number to each node according to the importance of the node, that is if node v is at least as important as node u, then $c_v \geq c_u$. A centrality index induces a partial ordering of the nodes in V. The *ranking* of a node v according to some centrality index c is the placement of v in the ordering induced by c and it is defined as

$$r_v^c = |\{u \in V \mid c_u > c_v\}| + 1.$$

According to [4], centrality indices can be classified into three non-disjoint categories: *geometric* indices, *path-based* indices, and *spectral* indices. The first category includes all those measures that evaluate the importance of a node on the basis of a function of the distances from the node to any other node, more in details, a geometric index depends only on how many nodes exist at every distance from the given node. Examples of geometric indices are: node degree, closeness centrality [2], Lin's index [26], harmonic centrality [4], and eccentricity. Instead of considering distances to a node, path-based indices take into account all the shortest paths (or all simple paths) passing through a node. Examples in this category are stress centrality [34], betweenness centrality [6, 17] and its variants [7]. Spectral indices evaluate the importance of a node on the basis of the left dominant eigenvector of a matrix derived from the graph. Examples of spectral indices are: Katz' index [22], page-rank [8], and HITS [25].

In this paper we study the problem of augmenting a graph in order to maximize the centrality of a node according to some index. We focus on four relevant centrality indices that are representative of the above categories. In what follows we define such centrality indices.

- The *harmonic centrality* [4] of a node v is defined as the harmonic mean of the distances from all the other nodes to v, formally:

$$h_v = \sum_{\substack{s \in V \setminus \{v\} \\ d_{sv} < \infty}} \frac{1}{d_{sv}}.$$

- The *betweenness centrality* [6, 17] of a node v is defined as the sum over all pairs of nodes (s, t) of the ratio between the number of shortest path from s to t passing through v and all the shortest paths from s to t that is:

$$b_v = \sum_{\substack{s,t \in V \\ s \neq t; s,t \neq v \\ \sigma_{st} \neq 0}} \frac{\sigma_{stv}}{\sigma_{st}}.$$

- The *eccentricity* of a node v is the maximum distance between v and any other node, that is

$$e_v = \max_{u \in V} \{d_{uv}\}.$$

  Note that, in this case a node is central if its eccentricity is small.

- In a directed graph, the *page-rank* of a node v is the probability that a *random surfer walk* that starts at a random node in a graph is at v at a given point in time. A random surfer walk with parameter α, is a walk in the graph defined as follows: start at a random node in G, given by a starting probability distribution; with probability α, move to an edge chosen uniformly at random from those outgoing the current node; with probability $1 - \alpha$, move directly to another node that might be not connected to the current node. In this latter case, the next node node is chosen by according to the starting probability distribution.

  Formally, let us assume that G is a strongly connected directed graph. Let M be a $|V| \times |V|$ matrix where each element m_{uv} is defined as $m_{uv} = \frac{1}{|N_u^o|}$ if $(u, v) \in E$ and $m_{uv} = 0$ otherwise. For a given parameter α, the page-rank is the eigenvector $\bar{p}$ associated to the largest eigenvalue of the matrix

$$Q = \frac{1 - \alpha}{|V|} \mathbb{1} + \alpha M.$$

  The page rank of a node v is the element p_v in the position associated to v in $\bar{p}$.

2.2 Problem statement

Given a set S of edges not in E, we denote by $G(S)$ the graph augmented by adding the edges in S to G, i.e. $G(S) = (V, E \cup S)$. For a parameter x of G, we denote by $x(S)$ the same parameter in graph $G(S)$, e.g. the distance from s to t in $G(S)$ is denoted as $d_{st}(S)$. The centrality index of a node v clearly depend on the graph structure: if we augment a graph by adding a set of edges S incident to v, then the centrality of v might change. Generally speaking, adding edges incident to some node v can only increase the centrality of v. We are interested in finding a set S of edges incident to a particular node v that maximizes such an increment. Therefore, given a centrality index c, we define the following optimization problem.

Centrality Maximization (CM)	
Given:	A directed or undirected graph $G = (V, E)$; a node $v \in V$; and an integer $k \in \mathbb{N}$
Solution:	A set S of edges incident to v, $S = \{(u, v) \mid u \in V \setminus N_v^i\}$ ($S = \{\{u, v\} \mid u \in V \setminus N_v\}$, if G is undirected), such that $\lvert S \rvert \leq k$
Goal:	Maximize $c_v(S)$

We study the CM problem by using harmonic centrality, betweenness centrality, eccentricity, and page-rank as indices, obtaining problems CM-H, CM-B, CM-E, CM-P.

2.3 Maximizing monotone submodular functions

Some of the algorithms reported in this paper, exploit the results of Nemhauser et al. on the approximation of monotone submodular objective functions [29]. A function z defined on subsets of a ground set N, $z : 2^N \to \mathbb{R}$, is *submodular* if the following inequality holds for any pair of sets $S \subseteq T \subseteq N$ and for any element $e \in N \setminus T$

$$z(S \cup \{e\}) - z(S) \geq z(T \cup \{e\}) - z(T).$$

In other words, a submodular function exhibits decreasing marginal gains: the marginal value of adding a new element to a set decreases as the set increases. Let us consider the following optimization problem: given a finite set N, an integer k', and a real-valued function z defined on the set of subsets of N, find a set $S \subseteq N$ such that $\lvert S \rvert \leq k'$ and $z(S)$ is maximum. If z is *monotone and submodular*, then the following greedy algorithm exhibits an approximation of $1 - \frac{1}{e}$ [29]: start with the empty set, and, for k' iterations, add an element that gives the maximal marginal gain, that is if S is a partial solution, choose the element $j \in N \setminus S$ that maximizes $z(S \cup \{j\})$.

Theorem 1 ([29])**.** *For a non-negative, monotone submodular function z, let S be a set of size k obtained by selecting elements one at a time, each time choosing*

Algorithm 1: Greedy algorithm for CM on directed graphs.
Input : A directed graph $G = (V, E)$; a node $v \in V$; and an integer $k \in \mathbb{N}$
Output: Set of edges $S \subseteq \{(u, v) \mid u \in V \setminus N_v^i\}$ such that $|S| \leq k$

```
S := ∅;
for i = 1, 2, ..., k do
    foreach u ∈ V \ N_v^i(S) do
        Compute c_v(S ∪ {(u, v)})
    u_max := arg max{c_v(S ∪ {(u, v)}) | u ∈ V \ N_v^i(S)};
    S := S ∪ {(u_max, v)};
return S;
```

an element that provides the largest marginal increase in the value of z. Then S provides a $\left(1 - \frac{1}{e}\right)$-approximation.

In this paper, we exploit such results by showing that some centrality indices c are monotone and submodular with respect to the possible set of edges incident to a given node v. Hence, the greedy algorithm in Algorithm 1 provides a $\left(1 - \frac{1}{e}\right)$-approximation for CM.[1] Algorithm 1 iterates k times and, at each iteration, it adds to an initially empty solution S an edge (u, v) (or $\{u, v\}$ in the case of undirected graph) that, when added to $G(S)$, gives the largest marginal increase in the centrality of v, that is $c(S \cup \{(u, v)\})$ ($c(S \cup \{\{u, v\}\})$, respectively) is maximum among all the possible edges not in $E \cup S$ incident to v. This technique will be used for harmonic centrality, betweenness centrality, and page-rank.

3 Centrality maximization

In this section we study the CM problem for harmonic centrality, betweenness centrality, eccentricity, and page-rank. For each problem we will give both hardness of approximation results and approximation algorithms. In order to highlight the main ideas and techniques, we will give only proof sketches and references to the complete proofs.

3.1 Harmonic centrality

We now report the results for the CM-H problem, more details on these results can be found in [10]. We first show the hardness of approximation results for the

[1] Algorithm 1 can be easily modified to work in the case of undirected graphs.

undirected and directed graph cases and then give an approximation algorithm for both cases.

To derive an approximation hardness result for the undirected case, we make use of the *Minimum Dominating Set* (in short, MDS) problem, which is defined as follows: given an undirected graph $G = (V, E)$, find a *dominating set* of minimum cardinality, that is, a subset D of V such that $V = D \cup \bigcup_{u \in D} N_u$. It is known that, for any r with $0 < r < 1$, it cannot exist a $(r \ln |V|)$-approximation algorithm for the MDS problem, unless $P = NP$ [14]. We now use this result in order to show that the CM-H problem does not admit a polynomial-time approximation scheme. To this aim, we design an algorithm A' that, given an undirected graph $G = (V, E)$ and given the size k of the optimal dominating set of G, by using an approximation algorithm A for the CM-H problem returns a dominating set of G whose approximation ratio is at most $(r \ln |V|)$. Clearly, we do not know the value of k, but we know that this value must be at least 1 and at most $|V|$: hence, we run algorithm A' for each possible value of k, and return the smallest dominating set found. Algorithm A' runs the approximation algorithm A for the CM-H problem multiple times. Each time A finds k nodes $u \in V$ which are the "new" neighbours of the node whose centrality has to be increased: we then add these nodes to the dominating set and create a smaller instance of the CM-H problem (which contain, among the others, all the nodes in V not yet dominated). We continue until all nodes in V are dominated.

Algorithm A' is specified in Fig. 2, where k denotes a "guess" of the size of an optimal solution for MDS with input the graph G. In the following, ω denotes the number of times the while loop is executed. Since, at each iteration of the loop, we include in the dominating set at most k nodes, at the end of the execution of algorithm A' the set D includes at most $k \cdot \omega$ nodes. Hence, if k is the correct guess of the value of the optimal solution for the MDS instance, then D is a ω-approximate solution for the MDS problem (as we have already noticed, we do not know the correct value of k, but algorithm A' can be executed for any possible value of k, that is, for each $k = 1, 2, \ldots, |V|$).

The first instruction of the while loop of algorithm A' computes a transformed graph G' (to be used as part of the new instance for CM-H) starting from the current graph $G = (V, E_V)$, which is the subgraph of the original graph induced by the set $\{u_1, \ldots, u_n\}$, where $n = |V|$, of still not dominated nodes. This computation is done as follows. We add a new node z and two new nodes x_i and y_i, for each i with $1 \leq i \leq n$. Moreover, we add to E_V the edges $\{z, y_i\}$, $\{x_i, y_i\}$, and $\{x_i, u_i\}$, for each i with $1 \leq i \leq n$. As it is shown in the second line of the while loop, z is the node whose harmonic centrality h_z has to be increased by adding at most k edges: that is, the CM-H instance is formed by G', z, and k. We can assume that the solution S computed at the second line of the while loop of algorithm A' contains only edges connecting z to nodes in V (see [10] for details).

Algorithm 2: The approximation algorithm A' for the MDS problem, given a γ-approximation algorithm A for the CM-H problem and a "guess" k for the optimal value of MDS.

Input : an undirected graph $G = (V, E)$ and an integer k

Output: a dominating set D

```
D := ∅;
while V ≠ ∅ do
    Compute graph G' starting from G;
    S := A(G', z, k);
    D' := {u : {z, u} ∈ S}
    D := D ∪ D';
    V := V \ (D' ∪ ⋃_{u∈D'} N_u);
    G := subgraph of G induced by V;
return D;
```

First of all, note that, since k is (a guess of) the measure of an optimal solution D^* for MDS with input G, we have that the measure $h^*(G', z, k)$ of an optimal solution S^* for CM-H with input G' satisfies the following inequality:

$$h^*(G', z, k) \geq k + \frac{1}{2}(n - k) + \frac{3}{2}n = \frac{1}{2}k + 2n.$$

This is due to the fact that, by connecting z to all the k nodes in D^*, in the worst case we have that k nodes in G are at distance 1, $n - k$ nodes in G are at distance 2 (since D^* is a dominating set), the n nodes y_i are at distance 1, and the n nodes x_i are at distance 2 from z.

Given the solution S computed by the approximation algorithm A for CM-H, let a and b denote the number of nodes in G at distance 2 and 3, respectively, from z in $G'(S)$. Since all nodes in G' are at distance at most 3 from z, we have that $n = k + a + b$ (we can assume, without loss of generality, that $n \geq k$): hence, $a = n - b - k$. Since A is a γ-approximation algorithm for CM-H, we have that $h_z(S) \geq \gamma h^*(G', z, k)$. That is, $k + \frac{1}{2}a + \frac{1}{3}b + \frac{3}{2}n \geq \gamma\left(\frac{1}{2}k + 2n\right)$. From this inequality, by doing some algebraic computation that use the fact that $a = n - b - k$ and $k \leq n$, we obtain $b \leq 15n(1 - \gamma)$.

Assuming $\gamma > 1 - \frac{1}{15e} > \frac{14}{15}$ (which implies $15(1 - \gamma) < 1$), then after one iteration of the while loop of algorithm A', the number of nodes in G decreases by a factor $15(1 - \gamma)$. Hence, after $\omega - 1$ iterations, the number n of nodes in the graph G is at most a fraction $[15(1-\gamma)]^{\omega-1}$ of the number N of nodes in the original graph. Since we can stop as soon as $n < k$, we need to find the maximum value of ω such that $k \leq N[15(1 - \gamma)]^{\omega-1}$. By solving this inequality and by recalling that

$15(1-\gamma) < 1$, we obtain

$$\omega - 1 \leq \log_{15(1-\gamma)} \frac{k}{N} \leq \log_{15(1-\gamma)} \frac{1}{N} = \frac{\ln(N)}{\ln \frac{1}{15(1-\gamma)}}.$$

One more iteration might be necessary to trivially deal with the remaining nodes, which are less than k. Hence, the total number ω of iterations is at most $\frac{\ln(N)}{\ln \frac{1}{15(1-\gamma)}} + 1$. If $\gamma > 1 - \frac{1}{15e}$, then the solution reported by algorithm A' is an $(r' \ln N + 1)$-approximate solution, where $r' = \frac{1}{\ln \frac{1}{15(1-\gamma)}} < 1$. Clearly, for any r with $0 < r' < r < 1$, there exists a number $N^{(r)}$ sufficiently large, such that for any $N > N^{(r)}$, $r' \ln N + 1 \leq r \ln N$: hence, algorithm A' would be an $r \ln N$-approximation algorithm for MDS, and, because of the result of [14], P would be equal to NP. Thus, we have that, if $P \neq NP$, then γ has to be not greater than $1 - \frac{1}{15e}$. The next theorem follows.

Theorem 2 ([10])**.** *The* CM-H *problem on undirected graphs cannot be approximated within a factor greater than* $1 - \frac{1}{15e}$*, unless* $P = NP$*.*

We now focus on the directed case and show that also in this case the CM-H problem cannot be approximated within a certain constant upper bound, unless $P = NP$. We make use of the *Maximum Set Coverage* (in short, MSC) problem, which is defined as follows: given a set X, a collection $\mathcal{F}\}$ of subsets of X, and an integer k, find a sub-collection $\mathcal{F}' \subseteq \mathcal{F}$ such that $|\mathcal{F}'| \leq k$ and $s(\mathcal{F}') = |\cup_{S_j \in \mathcal{F}'} S_j|$ is maximized. It is known that the MSC problem cannot be approximated within a factor greater than $1 - \frac{1}{e}$, unless $P = NP$ [16].

In this case we follow the scheme of L-reductions [35, Chapter 16]. In detail, we will give a polynomial-time algorithm that transforms any instance I_{MSC} of MSC into an instance $I_{\text{CM-H}}$ of CM-H and a polynomial-time algorithm that transforms any solution S for $I_{\text{CM-H}}$ into a solution $\mathcal{F}'$ for I_{MSC} such that the following two conditions are satisfied for some constants a and b:

$$OPT(I_{\text{CM-H}}) \leq a OPT(I_{\text{MSC}}) \tag{1}$$

$$OPT(I_{\text{MSC}}) - s(\mathcal{F}') \leq b\,(OPT(I_{\text{CM-H}}) - h_v(S))\,. \tag{2}$$

where OPT denotes the optimal value of an instance of an optimization problem. If the above conditions are satisfied and there exists a α-approximation algorithm for CM-H, then there exists a $(1 - ab(1-\alpha))$-approximation algorithm for MSC [35, Chapter 16]. Since MSC is hard to approximate within a factor greater than $1 - \frac{1}{e}$, then $1 - ab(1-\alpha) < 1 - \frac{1}{e}$, unless $P = NP$. This implies that, if $P \neq NP$, $\alpha < 1 - \frac{1}{abe}$.

Given an instance $I_{\text{MSC}} = (X, \mathcal{F}, k)$ of MSC, we define an instance $I_{\text{CM}} = (G, v, k)$, where $G = (V, E)$, $V = \{v\} \cup \{v_{x_i} \mid x_i \in X\} \cup \{v_{S_j} \mid S_j \in \mathcal{F}\}$, and $E = \{(v_{x_i}, v_{S_j}) \mid x_i \in S_j\}$.

Without loss of generality, we can assume that any solution S of CM-H contains only edges (v_{S_j}, v) for some $S_j \in \mathcal{F}$ (see [10] for details). Given a solution S of CM-H, let $\mathcal{F}'$ be the solution of MSC such that $S_j \in \mathcal{F}'$ if and only if $(v_{S_j}, v) \in S$. We now show that $h_v(S) = \frac{1}{2}s(\mathcal{F}') + k$. To this aim, let us note that the distance from a vertex v_{x_i} to v is equal to 2 if an edge (x_{S_j}, v) such that $x_i \in S_j$ belongs to S, and it is ∞ otherwise. Similarly, the distance from a vertex v_{S_j} to v is equal to 1 if $(x_{S_j}, v) \in S$, and it is ∞ otherwise. Moreover, the set of elements x_i of X such that $d_{v_{x_i}v}(S) < \infty$ is equal to $\{x_i \mid x_i \in S_j \wedge (v_{S_j}vu) \in S\} = \bigcup_{S_j \in \mathcal{F}'} S_j$. Therefore,

$$\begin{aligned} h_v(S) = \sum_{\substack{u \in V \setminus \{v\} \\ d_{uv}(S) < \infty}} \frac{1}{d_{uv}(S)} &= \sum_{\substack{x_i \in X \\ d_{v x_i v}(S) < \infty}} \frac{1}{d_{v_{x_i} v}(S)} + \sum_{\substack{S_j \in \mathcal{F} \\ d_{v_{S_j} v}(S) < \infty}} \frac{1}{d_{v_{S_j} v}(S)} \\ &= \frac{1}{2} |\{x_i \in X \mid d_{v_{x_i} v}(S) < \infty\}| + |\{S_j \in \mathcal{F} \mid d_{v_{S_j} v}(S) < \infty\}| \\ &= \frac{1}{2} \left| \bigcup_{S_j \in \mathcal{F}'} S_j \right| + |\{S_j \mid (v_{S_j}, v) \in S\}| = \frac{1}{2} s(\mathcal{F}') + k. \end{aligned}$$

It follows that Conditions (1) and (2) are satisfied for $a = \frac{3}{2}$ and $b = 2$. Indeed, $OPT(I_{\text{CM-H}}) = \frac{1}{2}OPT(I_{\text{MSC}}) + k \leq \frac{3}{2}OPT(I_{\text{MSC}})$, where the inequality is due to the fact that $OPT(I_{\text{MSC}}) \geq k$, since otherwise the greedy algorithm would find an optimal solution for I_{MSC}. Moreover, $OPT(I_{\text{MSC}}) - s(\mathcal{F}') = 2\,(OPT(I_{\text{CM-H}}) - k) - 2\,(h_v(S) - k) = 2\,(OPT(I_{\text{CM-H}}) - h_v(S))$. The next theorem follows by plugging the values of a and b into $\alpha < 1 - \frac{1}{abe}$.

Theorem 3 ([10])**.** *The* CM-H *problem on directed graphs cannot be approximated within a factor greater than* $1 - \frac{1}{3e}$*, unless* $P = NP$*.*

In the following we show that h_u is monotone and submodular in the case of undirected graphs, the proof can be easily adapted to the case in which the graphs are directed. To simplify the notation, we assume that $\frac{1}{\infty} = 0$. To show that h_v is monotone increasing, it is enough to observe that, for each solution S to CM-H, for each edge $\{u, v\} \notin E \cup S$, and for each node $x \in V \setminus \{v\}$, $d_{vx}(S \cup \{\{u, v\}\}) \leq d_{vx}(S)$ (since adding an edge cannot increase the distance between two nodes) and, therefore, $\frac{1}{d_{vx}(S \cup \{\{u,v\}\})} \geq \frac{1}{d_{vx}(S)}$.

To prove that h_v is submodular, we show that, for each pair S and T of solutions for CM-H such that $S \subseteq T$, and for each edge $\{u, v\} \notin T \cup E$,

$$h_v(S \cup \{\{u, v\}\}) - h_v(S) \geq h_v(T \cup \{\{u, v\}\}) - c_u(T).$$

To this aim, we prove that each term of h_u is submodular, that is, for each vertex $x \in V \setminus \{v\}$,

$$\frac{1}{d_{vx}(S \cup \{\{u, v\}\})} - \frac{1}{d_{vx}(S)} \geq \frac{1}{d_{vx}(T \cup \{\{u, v\}\})} - \frac{1}{d_{vx}(T)}. \tag{3}$$

Let us consider the shortest paths from v to x in $G(T \cup \{\{u, v\}\})$, and let us distinguish the following two cases.

1. The first edge of a shortest path from v to x in $G(T \cup \{\{u, v\}\})$ is $\{u, v\}$ or belongs to $S \cup E$. In this case, such a path is a shortest path also in $G(S \cup \{\{u, v\}\})$, as it cannot contain edges in $T \setminus S$ (since these edges are all incident to v). Then, $d_{vx}(S \cup \{\{u, v\}\}) = d_{vx}(T \cup \{\{u, v\}\})$ and $\frac{1}{d_{vx}(S \cup \{\{u,v\}\})} = \frac{1}{d_{vx}(T \cup \{\{u,v\}\})}$. Moreover, $d_{vx}(S) \geq d_{vx}(T)$ (since $S \subseteq T$) and, therefore, $-\frac{1}{d_{vx}(S)} \geq -\frac{1}{d_{vx}(T)}$.

2. The first edge of all shortest paths from v to x in $G(T \cup \{\{u, v\}\})$ belongs to $T \setminus S$. In this case, $d_{vx}(T) = d_{vx}(T \cup \{\{u, v\}\})$ and, therefore, $\frac{1}{d_{vx}(T \cup \{\{u,v\}\})} - \frac{1}{d_{vx}(T)} = 0$. As $\frac{1}{d_{vx}(S)}$ is monotone increasing, then $\frac{1}{d_{vx}(S \cup \{\{u,v\}\})} - \frac{1}{d_{vx}(S)} \geq 0$.

In both cases, we have that the inequality (3) is satisfied and, hence, the next theorem follows.

Theorem 4 ([10])**.** *In both directed and undirected graphs, for each vertex u, function h_u is monotone and submodular with respect to any feasible solution for* CM-H.

Theorems 1 and 4 imply the next corollary.

Corollary 5. *The* CM-H *problem is approximable within a factor $1 - \frac{1}{e}$ in both directed and undirected graphs.*

3.2 Betweenness centrality

We now show that problem CM-B is hard to be approximated within a certain constant upper bound, that in the case of directed graphs the objective function is monotone and submodular, and that there are instances of the undirected case for which the greedy algorithm exhibits an arbitrarily small approximation ratio. We omit proof sketches for the first two results as the arguments are similar to those of Theorems 3 and 4, respectively. Full proofs of the results stated in this section can be found in [9, 12].

We observe that the next result for undirected graphs has been proven only for the case in which edges are weighted [12].

Theorem 6 ([9, 12])**.** *The* CM-B *problem on both directed and undirected graphs cannot be approximated within a factor greater than $1 - \frac{1}{2e}$, unless $P = NP$.*

Theorem 7 ([9])**.** *In directed graphs, for each vertex v, function b_v is monotone and submodular with respect to any feasible solution for* CM-B.

Corollary 8. *In directed graphs, the* CM-B *problem is approximable within a factor* $1 - \frac{1}{e}$.

We now prove that, differently from the directed case and from the case of harmonic centrality, the approximation ratio of a greedy solution for CM-B in the case of undirected graphs does not have a constant lower bound. To this aim, let us consider the following instance of CM-B.

- Graph $G = (V, E)$.
- $V = \{v, t, a, b, c, a', b', c'\} \cup A \cup B \cup C$, where $A = \{a_i\}_{i=1}^{y}$, $B = \{b_i\}_{i=1}^{x}$, $C = \{c_i\}_{i=1}^{y}$, and $y = x - 2$, for some $x > 2$;
- $E = \{\{v, t\}, \{a, b\}, \{b, c\}, \{a, a'\}, \{b, b'\}, \{c, c'\}, \{a', t\}, \{b', t\}, \{c', t\}\} \cup \{\{a_i, a\} \mid a_i \in A\} \cup \{\{b_i, b\} \mid b_i \in B\} \cup \{\{c_i, c\} \mid c_i \in C\}$;
- $k = 2$.

The initial value of b_v is zero. The greedy algorithm first chooses edge $\{b, v\}$ and then edge $\{a_i, v\}$, for some $a_i \in A$ (or equivalently $\{c_i, v\}$, for some $c_i \in A$). The value of $b_v(\{b, v\}, \{a_i, v\})$ is $2x + 3$. In fact, the following pairs have shortest paths passing through v in $G(\{b, v\}, \{a_i, v\})$: nodes in $B \cup \{b\}$ and t ($x + 1$ shortest paths), a_i and t (1 shortest path), a_i and nodes in $B \cup \{b\}$ ($\frac{x+1}{2}$ shortest paths), a_i and nodes in $C \cup \{c\}$ ($\frac{y+1}{2}$ shortest paths), and a_i and c' (1 shortest path). An optimal solution, instead, is made of edges $\{a, v\}$ and $\{c, v\}$ where $b_v(\{\{a, v\}, \{c, v\}\}) = \frac{x^2+3x-2}{2}$, where the quadratic term comes from the fact that there are $(y+1)^2$ paths passing through v between nodes in $A \cup \{a\}$ and nodes in $C \cup \{c\}$. Therefore, the approximation ratio of the greedy algorithm tends to be arbitrarily small as x increases. We observe that the bad approximation ratio of the greedy algorithm is due to the fact that it does not consider the shortest paths that pass through v by using both edges. The next proposition follows.

Proposition 9 ([12])**.** *In undirected graphs, the greedy algorithm exhibits an unbounded approximation ratio.*

3.3 Eccentricity

We now report the results on the CM-E problem, more details can be found in [13, 32]. Note that in this case a node is considered central if its eccentricity is small, therefore the CM-E problem is a minimization problem, that is we want to find the set of edges S that, when added to G, minimizes the value of $e_v(S)$, for some given node v. We first show that, unless $P = NP$, the problem cannot be approximated

within a certain constant lower bound, we then give an algorithm that guarantees a constant approximation ratio and an algorithm that guarantees an arbitrarily small approximation ratio if an higher number of edges is allowed.

To derive an approximation hardness result for the undirected case, we make use of the *Set Cover* (in short, sc) problem, which is defined as follows: given a set X, a collection $\mathcal{F}\}$ of subsets of X, and an integer B, find a sub-collection $\mathcal{F}' \subseteq \mathcal{F}$ such that $\cup_{S_j \in \mathcal{F}'} S_j = X$ and $|\mathcal{F}'| \leq B$. It is known that the set cover problem is NP-hard [18].

Given an instance $(X, \mathcal{F})$ of sc, we compute a graph $G = (V, E)$, where $V = \{v, v'\} \cup \{v_{x_i} \mid x_i \in X\} \cup \{v_{S_j} \mid S_j \in \mathcal{F}\}$ and $E = \{\{v, v'\}\} \cup \{\{v', v_{S_j}\} \mid S_j \in \mathcal{F}\} \cup \{\{v_{x_i}, v_{S_j}\} \mid x_i \in S_j\}$. Initially, the eccentricity of v is equal to 3. We prove that there exists a feasible solution for an instance $I_{\text{sc}} = (X, \mathcal{F})$ of sc if and only if there exists a solution S for the instance $I_{\text{CM-E}} = (G, v, k)$, where $k = B$, of CM-E such that $e_v(S) = 2$.

If I_{sc} admits a feasible solution $\mathcal{F}'$, then let us consider the solution $S = \{\{v, v_{S_j}\} \mid S_j \in \mathcal{F}'\}$ to $I_{\text{CM-E}}$. Since $|\mathcal{F}'| \leq B$, then $|S| \leq k$. Moreover, $\cup_{S_j \in \mathcal{F}'} S_j = X$ and then all the nodes v_{x_i} are at distance 2 to v. Therefore, $e_v(S) = 2$.

Let us now assume that $I_{\text{CM-E}}$ admits a solution S such that $e_v(S) = 2$, without loss of generality, we can assume that S contains only edges $\{v, v_{S_j}\}$ for some $S_j \in \mathcal{F}$ (see [13] for details). Let $\mathcal{F}'$ be the solution of sc such that $S_j \in \mathcal{F}'$ if and only if $\{v, v_{S_j}\} \in S$. Since $e_v(s) = 2$, the distance between v and all the nodes v_{x_i} is at most 2 and then for each v_{x_i} there exists an edge $\{v, v_{S_j}\} \in S$ such that $x_i \in S_j$. This implies that $\cup_{S_j \in \mathcal{F}'} S_j = X$. Moreover, since $|S| \leq k$, then $|\mathcal{F}'| \leq B$.

Let us assume that there exists an approximation algorithm A for CM-E that guarantees an approximation factor $\alpha < \frac{3}{2}$ and let S be the solution obtained by applying algorithm A to $I_{\text{CM-E}}$ derived from I_{sc}. We have that $e_v(S) < \frac{3}{2} OPT$. This implies that, if $(X, \mathcal{F})$ admits a feasible solution, then $e_v(S) < \frac{3}{2} \cdot 2 = 3$, that is $e_v(S) = 2$; otherwise, if $(X, \mathcal{F})$ does not admit a feasible solution, then $e_v(S) = 3$. Therefore, we can determine whether an instance of sc is feasible or not by means of algorithm A. The next theorem follows.

Theorem 10 ([13])**.** *The* CM-E *problem on undirected graphs cannot be approximated within a factor smaller than* $\frac{3}{2}$*, unless* $P = NP$*.*

In what follows we describe the algorithm given in [32] to solve the CM-E problem in undirected graphs. The algorithm is based on a former solution to the problem of minimizing the diameter of a graph by adding a limited number of edges [3].

The algorithm is reported in Algorithm 3 and works as follows: first node v is inserted into a set U, then, a for loop of k iteration is run. At each iteration $i = 1, 2, \ldots, k$, a node u_i that maximizes the minimum distance in G between u_i

Algorithm 3: Approximation algorithm for CM-E.

Input : An undirected graph $G = (V, E)$; a node $v \in V$; and an integer $k \in \mathbb{N}$

Output: Set of edges $S \subseteq \{\{u, v\} \mid u \in V \setminus N_v\}$ such that $|S| \leq k$

```
S := ∅;
U := {v};
for i = 1, 2, ..., k do
    u_i := arg max_{u∈V} min_{u_j∈U} d_{uu_j};
    U := U ∪ {u_i};
    S := S ∪ {{u_i, v}};
return S;
```

and a vertex in U is selected and inserted into U. The solution S returned is made of edges that connect nodes u_i in $U \setminus \{v\}$ to v, $S = \{\{u_i, v\} \mid u_i \in U \setminus \{v\}\}$.

To analyze the algorithm, we need some further notation. Let $IS(G)$ be the size of a *maximum independent set* of graph $G = (V, E)$, that is the size of a maximum subset of nodes $V' \subseteq V$ such that no two nodes in V' are joined by an edge in E. Given a subset of nodes $U \subseteq V$, the *radius* of U is defined as $r_U = \min_{x \in V} \max_{u \in U} d_{xu}$. Given a graph G and an integer $d \geq 0$, $G^d = (V, E^d)$ is the graph with the same nodes as G and an edge (x, y) if the distance in G between x and y is at most d.

Let S^* be an optimal solution for the instance of CM-E and let OPT denote $e_v(S^*)$. The diameter of $G(S^*)$ is at most $2OPT$ and therefore $IS((G(S^*))^{2OPT}) = 1$. The next lemma implies that $IS((G(S^*))^{2OPT}) \geq IS(G^{2OPT}) - |S^*|$.

Lemma 11 ([3])**.** *Let G be a graph and let $d \geq 0$. For each $e \in V \times V \setminus E$, $IS((G(\{e\}))^d) \geq IS(G^d) - 1$.*

It follows that $IS(G^{2OPT}) \leq k + 1$. Let $u_0 = v$. We partition the set of nodes V into $k + 1$ clusters $U_0, U_1, \ldots, U_k$ as follows: for each $i = 0, 1, \ldots, k$, a node u belongs to U_i if $d_{uu_i} \leq d_{uu_j}$, for each $j = 0, 1, \ldots, k$, ties are arbitrarily broken in order to form a partition. Sets $U_0, U_1, \ldots, U_k$ are called the *clusters* induced by Algorithm 3. The next lemma implies that, for each $i = 0, 1, \ldots, k$, $r_{U_i} \leq 2OPT$.

Lemma 12 ([3])**.** *Let G be a graph, let $d \geq 0$, and let $U_0, U_1, \ldots, U_k$ be the clusters induced by Algorithm 3 on G. If $IS(G^d) \leq k + 1$, then for each $i = 0, 1, \ldots, k$, $r_{U_i} \leq d$.*

Clearly $|S| \leq k$ and the distance between each node $u \in V$ and v in $G(S)$ is at most $2OPT + 1$ to v, therefore, $e_v(S) \leq 2OPT + 1$. The approximation factor guaranteed by Algorithm 3 is then $2 + \frac{1}{OPT}$.

Theorem 13 ([32]). *In undirected graphs, the* CM-E *problem is approximable within a factor* $2 + \frac{1}{OPT}$*, where* OPT *is the value of an optimal solution.*

The next theorem shows that if we allow a number of added edges that is higher than k, then we can obtain a solution that is at most $1 + \epsilon$ far from the optimal solution of the case in which only k additional edges are allowed.

Theorem 14 ([13]). *For any* $\epsilon > 0$*, there exists a polynomial-time algorithm that adds* $O(k \log |V|)$ *edges to reduce the eccentricity of* v *to at most* $1 + \epsilon$ *times the optimum eccentricity for the case in which* k *additional edges are allowed.*

3.4 Page-rank

We first show that the CM-P problem does not admit a polynomial-time approximation scheme, and then we show that a variant of the greedy algorithm guarantees a constant approximation ratio. More details on these results can be found in [1, 31].

The next theorem states that there exist no polynomial-time approximation scheme for the CM-P problem, unless $P = NP$. The proof is quite technical and hence it is omitted here, see [31] for details.

Theorem 15 ([31]). *The* CM-P *problem does not admit an FPTAS, unless* $P = NP$.

Let I denote the $|V| \times |V|$ identity matrix and let us consider matrix the matrix $Z = (I - \alpha M)^{-1}$. Then, the entry z_{uv} of Z is the expected number of visits to node v for a random surfer walk starting at node u [1]. The value $g_v = \frac{p_v}{z_{vv}}$ is the *overall reachability* of node v from all the other nodes, that is the probability that node v is reached by a random surfer walk that starts at some node u, for all $u \in V$ [31]. Let us consider a variant of the CM-P problem where the function to maximize is g_v and let us denote such problem as CM-G. The next theorem implies that problem CM-G can be approximated by the greedy algorithm with an approximation factor of $1 - \frac{1}{e}$.

Theorem 16 ([31]). *In directed graphs, for each vertex* v*, function* g_v *is monotone and submodular with respect to any feasible solution for* CM-G.

Let S be the solution of the greedy algorithm for problem CM-G and let $OPT = \frac{p_v^{OPT}}{z_{vv}^{OPT}}$ denote the value of an optimal solution for CM-G. The previous theorem implies that

$$\frac{p_v(S)}{z_{vv}(S)} \geq \left(1 - \frac{1}{e}\right) \frac{p_v^{OPT}}{z_{vv}^{OPT}}.$$

Finally, the next theorem follows by the observation that, for any solution S', $z_{vv}(S') \leq \sum_{i=0}^{\infty} \alpha^{2i} = \frac{1}{1-\alpha^2}$ and $z_{vv}(S') \geq 1$ [31].

Theorem 17 ([31]). *In directed graphs, the* CM-P *problem is approximable within a factor* $\left(1 - \alpha^2\right)\left(1 - \frac{1}{e}\right)$.

Centrality index	Graph type	Inapproximability Upper/Lower bound	Approximation algorithms
Harmonic	Undir.	$1 - \frac{1}{15e}$	$1 - \frac{1}{e}$
	Dir.	$1 - \frac{1}{3e}$	$1 - \frac{1}{e}$
Betweenness	Undir.	$1 - \frac{1}{2e}$	OPEN
	Dir.	$1 - \frac{1}{2e}$	$1 - \frac{1}{e}$
Eccentricity	Undir.	$\frac{3}{2}$	$2 + \frac{1}{OPT}$ $1 + \epsilon$, with $O(k \log \lvert V \rvert)$ edges
	Dir.	OPEN	OPEN
Page-rank	Undir.	OPEN	OPEN
	Dir.	NO FPTAS	$\left(1 - \alpha^2\right)\left(1 - \frac{1}{e}\right)$

Table 1: Summary of results.

4 Summary of results and open problems

In this paper we summarized some recent results on the CM problem which consist in finding a limited amount of edges to be added incident to a given node v in a graph in such a way that the centrality of v is maximized. In particular, we used harmonic centrality, betweenness centrality, eccentricity and page-rank as centrality indices. The results outlined in this paper are reported in Table 1. It is worth to note that for all the problems, except for CM-E, the approximation algorithm used is basically the same greedy algorithm.

In the following we list some research directions that deserve further investigation.

- First of all, it would be worth to close open cases pointed out in Table 1 and to close the gaps between approximation and inapproximability results. Moreover, it would be interesting to study the CM problem with other centrality indices. Note that not always the greedy algorithm given in this paper exhibits a bounded approximation ratio, see the case of CM-B for undirected graphs, and hence new algorithms could be required.

- A centrality index c induces a ranking of the nodes which is the placement of a node v according to c and it is denoted as r_v^c. It has been experimentally observed that increasing the centrality of a given node v has the consequence of increasing the ranking of v [9, 10, 12]. Therefore, maximizing the cen-

trality of v like in the CM problem decreases a lot r_v^c. However, it could be worth to directly study the problem of optimizing the position of v in the ranking, that is find a set S of edges incident to v that minimizes $r_v^c(S)$ or maximizes the possible increment in the ranking of v, $r_v^c - r_v^c(S)$.

- The notion of centrality index of a node can be extended to a set of nodes in the graph. The aim is to capture the centrality of a particular class of individuals within a large community (e.g. a specific department inside a large company). Informally, given a centrality index c and a subset of nodes $U \subseteq V$, the *group centrality index c* on U is the centrality of a "virtual" node u that collapses all the nodes in U. Relevant group centrality indices are degree, closeness, betweenness, and flow betweenness [15]. It could be interesting to extend the CM problem to group centrality indices, i.e. maximizing the group centrality of a given group of individuals in a network by adding a limited amount of edges incident to one or more nodes in the group.

- It is not difficult to figure out scenarios in which two or more nodes try to increase their centrality by adding new edges. In such a scenario the best strategy that each node should adopt might be different from the greedy one. It would then be interesting to study this scenario from a game theoretic perspective.

- In the field of complex networks, different models of information diffusion have been introduced in the literature in order to model the dynamics that regulate the diffusion of information in a network. Important examples are the *Linear Threshold Model* [21, 23, 33] and the *Independent Cascade Model* [19, 20, 23, 24]. In such models, we can distinguish between active, or infected, nodes which spread the information and inactive ones. At the beginning of the process a small percentage of nodes of the graph is set to active in order to let the information diffusion process start. Recursively, currently infected nodes can infect their neighbours with some probability. After a certain number of such cycles, a large number of nodes might become infected in the network. The *influence maximization problem* consists in finding a set A of k nodes such that if we initiate the spreading of information by activating the nodes in A, the number of nodes that is active at the end of the process is maximum. Nodes in A are called seeds. A possible extension of the work in this paper is to determine a limited number of edges to be added incident to the seeds in order to maximize the eventual number of active nodes. Preliminary results for the case of the independent cascade model have been presented in [11].

- Assuming that $k = o(|E|)$, the time complexity of the greedy algorithm is $O(k|V|f_c(|V|,|E|))$, where $f_c(|V|,|E|)$ is the time complexity of computing c for a node v in graph $G = (V, E \cup S)$ for some solution S such that $|S| \leq k$. In many cases, $f_c(|V|,|E|)$ is at least linear in $|E|$ and therefore, the time complexity of the greedy algorithm is at least quadratic. When graphs are huge, with billions of nodes and edges, the greedy algorithm requires to much time. However, we do not actually need to recompute the centrality of a node for each possible added edge and at each iteration but we can exploit the so-called *dynamic algorithms* for centrality measures that compute $c_v(S \cup \{e\})$ starting from $c_v(S)$ in a smaller computational time compared to $f_v(|V|, |E \cup S \cup \{e\}|)$. This approach has been adopted for the case of harmonic centrality [10].

- In many applications one wants to minimize the centrality of a given node rather then maximize it. Examples are applications in which one wants to reduce the traffic flow in nodes of a road or a communication networks or reduce the spread of disease in epidemic and social networks. In general this can be done by deleting edges from the graph. Therefore it would be interesting to study the "dual" problem of CM in which we want to minimize the centrality of a given node by deleting edges incident to that node.

References

[1] K. Avrachenkov and N. Litvak. The effect of new links on google pagerank. *Stochastic Models*, 22(2):319–331, 2006.

[2] A. Bavelas. Communication patterns in task-oriented groups. *The Journal of the Acoustical Society of America*, 22(6):725–730, 1950.

[3] D. Bilò, L. Gualà, and G. Proietti. Improved approximability and non-approximability results for graph diameter decreasing problems. *Theoretical Computer Science*, 417:12–22, 2012.

[4] P. Boldi and S. Vigna. Axioms for centrality. *Internet Mathematics*, 10(3–4):222–262, 2014.

[5] S. P. Borgatti. Centrality and network flow. *Social networks*, 27(1):55–71, 2005.

[6] U. Brandes. A faster algorithm for betweenness centrality. *Journal of Mathematical Sociology*, 25(2):163–177, 2001.

[7] U. Brandes. On variants of shortest-path betweenness centrality and their generic computation. *Social Networks*, 30(2):136–145, 2008.

[8] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. *Computer Networks and ISDN Systems*, 30(1-7):107–117, 1998.

[9] P. Crescenzi, G. D'Angelo, L. Severini, and Y. Velaj. Greedily improving our own centrality in a network. In *Proceedings of the 14th International Symposium on Experimental Algorithms (SEA 2015)*, volume 9125 of *Lecture Notes in Computer Science*, pages 43–55. Springer, 2015.

[10] P. Crescenzi, G. D'Angelo, L. Severini, and Y. Velaj. Greedily improving our own closeness centrality in a network. *ACM Transactions on Knowledge Discovery from Data*, 11(1):9:1–9:32, 2016.

[11] G. D'Angelo, L. Severini, and Y. Velaj. Influence maximization in the independent cascade model. In *Proceedings of the 17th Italian Conference on Theoretical Computer Science (ICTCS16)*, CEUR Workshop Proceedings. CEUR-WS.org, 2016. To appear.

[12] G. D'Angelo, L. Severini, and Y. Velaj. On the maximum betweenness improvement problem. *Electronic Notes in Theoretical Computer Science*, 322:153 – 168, 2016. Proceedings of the 16th Italian Conference on Theoretical Computer Science (ICTCS15).

[13] E. D. Demaine and M. Zadimoghaddam. Minimizing the diameter of a network using shortcut edges. In *Proceedings of the 12th Scandinavian Symposium and Workshops on Algorithm Theory (SWAT 2010)*, volume 6139 of *Lecture Notes in Computer Science*, pages 420–431. Springer, 2010.

[14] I. Dinur and D. Steurer. Analytical approach to parallel repetition. In *Proceedings of the 46th Annual ACM Symposium on Theory of Computing (STOC 2014)*, pages 624–633. ACM, 2014.

[15] M. G. Everett and S. P. Borgatti. The centrality of groups and classes. *The Journal of mathematical sociology*, 23(3):181–201, 1999.

[16] U. Feige. A threshold of ln n for approximating set cover. *Journal of the ACM*, 45(4), 1998.

[17] L. C. Freeman. A set of measures of centrality based on betweenness. *Sociometry*, 40(1):35–41, 1977.

[18] M. R. Garey and D. S. Johnson. *Computers and Intractability, A Guide to the Theory of NP-Completeness*. W.H. Freeman and Company, New York, 1979.

[19] J. Goldenberg, B. Libai, and E. Muller. Talk of the network: A complex systems look at the underlying process of word-of-mouth. *Marketing letters*, 12(3):211–223, 2001.

[20] J. Goldenberg, B. Libai, and E. Muller. Using complex systems analysis to advance marketing theory development: Modeling heterogeneity effects on new product growth through stochastic cellular automata. *Academy of Marketing Science Review*, 2001(9):1, 2001.

[21] M. Granovetter. Threshold models of collective behavior. *American journal of sociology*, 83(6):1420–1443, 1978.

[22] L. Katz. A new status index derived from sociometric analysis. *Psychometrika*, 18(1):39–43, 1953.

[23] D. Kempe, J. Kleinberg, and Éva Tardos. Maximizing the spread of influence through a social network. *Theory of Computing*, 11(4):105–147, 2015.

[24] D. Kempe, J. Kleinberg, and E. Tardos. Maximizing the spread of influence through a social network. In *Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2003)*, pages 137–146. ACM, 2003.

[25] J. M. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM*, 46(5):604–632, 1999.

[26] N. Lin. *Foundations of social research*. McGraw-Hill, New York, 1976.

[27] B. Macdonald, P. Shakarian, N. Howard, and G. Moores. Spreaders in the network SIR model: An empirical study. *CoRR*, abs/1208.4269, 2012.

[28] P. Malighetti, G. Martini, S. Paleari, and R. Redondi. The impacts of airport centrality in the EU network and inter-airport competition on airport efficiency. Technical Report MPRA-7673, 2009.

[29] G. Nemhauser, L. Wolsey, and M. Fisher. An analysis of approximations for maximizing submodular set functions–I. *Mathematical Programming*, 14(1):265–294, 1978.

[30] M. Newman. *Networks: an introduction*. Oxford university press, 2010.

[31] M. Olsen and A. Viglas. On the approximability of the link building problem. *Theoretical Computer Science*, 518:96–116, 2014.

[32] S. Perumal, P. Basu, and Z. Guan. Minimizing eccentricity in composite networks via constrained edge additions. In *Proceedings of the 32th IEEE Military Communications Conference (MILCOM)*, pages 1894–1899. IEEE, 2013.

[33] T. C. Schelling. *Micromotives and macrobehavior*. Norton & Company, 2006.

[34] A. Shimbel. Structural parameters of communication networks. *The bulletin of mathematical biophysics*, 15(4):501–507, 1953.

[35] D. Williamson and D. Shmoys. *The Design of Approximation Algorithms*. Cambridge University Press, 2011.

[36] E. Yan and Y. Ding. Applying centrality measures to impact analysis: A coauthorship network analysis. *Journal of the Association for Information Science and Technology*, 60(10):2107–2118, 2009.

Which Tasks of a Job are Susceptible to Computerization?

Philipp Brandes and Roger Wattenhofer

ETH Zurich, Switzerland

Abstract

In their paper, the two economists Frey and Osborne quantified the automation of jobs, by assigning each job in the O*NET database a probability to be automated. In this paper, we refine their results in the following way: Every O*NET job consists of a set of tasks, and these tasks can be related. We use a linear program to assign probabilities to tasks, such that related tasks have a similar probability and the tasks can explain the computerization probability of a job. Analyzing jobs on the level of tasks helps comprehending the results, as experts as well as laymen can more easily criticize and refine what parts of a job are susceptible to computerization.

1 Introduction

Computerization is considered to be one of the biggest socio-economic challenges. What is the foundation of the recent worries about many jobs being affected by automation [7, 8, 11, 12]?

Why did the last few years see dramatic technological progress regarding self-driving cars [14], board games [20], automatic language translation [1], or face recognition [16]? One reason is big data. While "intelligent algorithms" in the past were restricted to learning from data sets with a few thousand examples, we now have exabytes of data. Learning becomes even more powerful if you combine big data with a highly parallel hardware, stirred by the success of graphics processing units (GPUs). However, both of these technological advancements needed to be harvested, and they are with the advent of so-called deep learning algorithms, which have blown the competition away, starting with voice recognition [10]. As a consumer, you can already witness some of these advancements on your smartphone, but a lot more will come soon. We believe that these advancements will revolutionize white collar work and (with a little help from sensors and

robotics) also blue collar work. In contrast to previous waves of innovation, this time new emerging jobs might not be able to compensate jobs endangered by the new technology.

Computer scientists have been pondering the consequences of AI for a long time; starting with the possibly most famous paper in computer science "Computing Machinery and Intelligence" by Alan Turing, where he proposed the Turing test [21]. An early pioneer of AI, Marvin Minsky, claimed already 1967 that "within a generation ... the problem of creating 'artificial intelligence' will substantially be solved" [19]. More recent work in computer science focuses on the effects on the economy and the society [22–24].

In their paper, Frey and Osborne [12], two economists, quantitatively study job automation, predicting that 47% of US employment is at risk of automation. In order to calculate this number, Frey and Osborne labeled 70 of the 702 jobs from the O*NET OnLine job database[1] manually as either "automatable" or "not automatable". Then, for the remainder of the jobs in the O*NET database, they computed the automation probability as a function of the distance to the labeled jobs.

But the results of Frey and Osborne are opaque, one either believes their "magic" computerization percentages, or one has doubts. We want anybody to be able to easily understand and argue about our results, by incorporating the unique tasks of each job. This additional depth will help laymen as well as job experts to argue about potential flaws in our methodology.

If we know that a job is 100% automatable, we also know that every task of that job must be completely automatable. But what if a job is 87% automatable? Is every task 87% automatable? Or are 87% of the tasks completely automatable, and 13% not at all? We want to forecast which tasks of a job are safe and which tasks are automatable.

In order to calculate the automation probability for a task, we first need to determine its share of a job (Section 4). Based on this, we are able to assign each task a probability to be automated such that the weighted average of the probabilities is equal to the probability of the corresponding job (Section 6.1). During our evaluation (Section 6.2), we discover a few suspicious results in the probabilities by Frey and Osborne, e.g., a surprisingly high automation probability of 96% for the job *compensation and benefits managers*.

We analyze the correlation between various properties of a job and its probability to be automated (Section 7). E.g., we show that there is a strong negative correlation between the level of education required for a job and its probability to

[1] O*NET is an application that was created for the general public to provide broad access to the O*NET database of occupational information. The site is maintained by the National Center for O*NET Development, on behalf of the U.S. Department of Labor, Employment and Training Administration (USDOL/ETA); see https://www.onetonline.org/

be automated. In Section 8, we try to determine whether the effects of automation can already be seen in our data.

Our complete results can be found at `http://jobs-study.ethz.ch`.

2 Related Work

The current effects of automation have been studied intensively in economics. Most studies agree that some routine tasks have already fallen victim to automation [2, 4, 13]. A task is routine if "it can be accomplished by machines following explicit programmed rules" [4]. With computers being able to do routine tasks, the demand for human labor performing these tasks has decreased. But on the other hand, the demand for college educated labor has increased over the last decades [6, 25, 26]. The effect is more pronounced in industries that are computer-intensive [3]. As a consequence of this, the employment share of the highest skill quartile has increased. In addition to more people being employed in the highest skill quartile, the real wage for this quartile has increased faster than the average real wage. Service occupations, which are non-routine, but also not well paid, have also seen an increase in employment share and in real hourly wage. Thus, both, employment share and real wage, are U-shaped with respect to the skill level [2]. This employment pattern is a phenomenon that is called polarization. This is not unique to the US, but can also be observed, e.g., in the UK [13]. These papers make important observations about the effects that automation already has. Until now, routine tasks are the ones most affected, but more and more tasks can nowadays be performed by a computer. We focus on the future and try to predict which tasks will be automated next.

John Keynes predicted already in 1933 that there will be widespread technological unemployment "due to the means of economising the use of labour outrunning the pace at which we can find new uses for labour" [15]. Automation might be the technology, where this becomes true [7, 8, 11, 12]. "Automation of knowledge work", "Advanced robotics", and "Autonomous and near-autonomous vehicles" are considered to be 3 out of 12 potentially economically disruptive technologies [17]. Computer labor and human labor may no longer be complements, but competitors. Automation might be the cause for the current stagnation [7]. There might be too much technological progress, which causes high unemployment. A trend that could be going on for years, but was hidden by the housing boom [9].

The paper by Frey and Osborne is the first to make quantitative claims about the future of jobs [12]. Together with 70 machine learning experts, Frey and Osborne first manually labeled 70 out of 702 jobs from the O*NET database as either "automatable" or "non automatable". This labeling was, as the authors admit, a subjective assignment based on "eye balling" the job descriptions from

O*NET. Labels were only assigned to jobs where the whole job was considered to be (non) automatable, and to jobs where the participants of the workshop were most confident. To calculate the probability for non-labeled jobs, Frey and Osborne used a probabilistic classification algorithm. They chose 9 properties from O*NET as features for their classifier, namely "Finger Dexterity", "Manual Dexterity", "Cramped Work Space, Awkward Positions", "Originality", "Fine Arts", "Social Perceptiveness", "Negotiation", "Persuasion", and "Assisting and Caring for Others".

The results from Frey and Osborne for the US job market were adopted to other countries, e.g., Finland, Norway, and Germany [5, 18]. This was done by matching each job from O*NET to the locally used standardized name. Due to differences in the economies, a different percentage of people will be affected by this change, e.g., only one third in Finland and Norway are at risk compared to 47% in the US.

3 Model

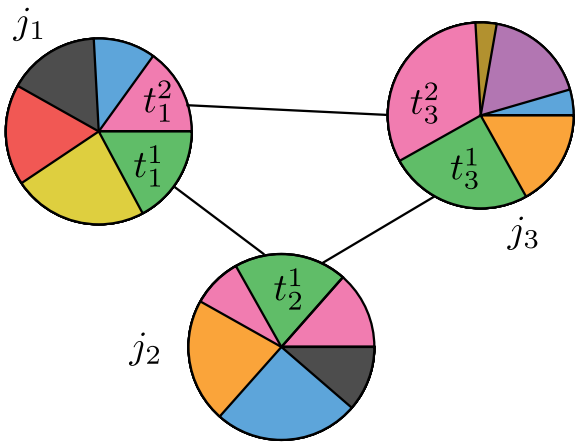
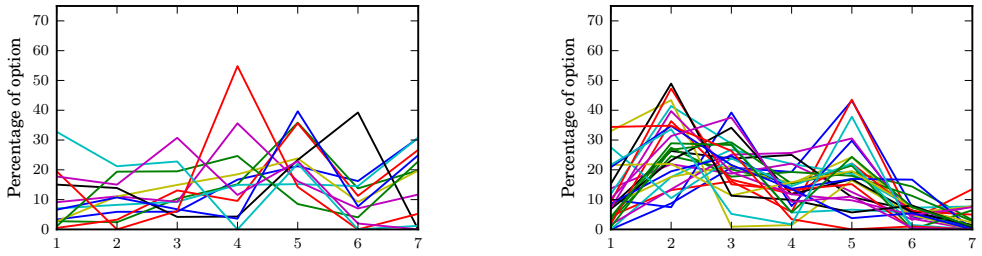


Figure 1: This figure shows a small example consisting of three jobs represented by circles. The share of each task of a job is shown by its sector. Two tasks are connected by a line if and only if they are related, i.e., similar according to O*NET. Task t_1^1 from job j_1 and task t_2^1 from job j_2 are related as indicated by the line connecting them. Note that this relationship is not transitive. Thus, tasks t_1^1 and t_3^1 do not need to be related.

We are given a set of jobs $J = \{j_1, \ldots, j_n\}$. Each job j_i consists of a set of tasks $T_i = \{t_i^1, \ldots, t_i^m\}$, where every task belongs to exactly one job. We call two tasks t_i^k and $t_{i'}^{k'}$ related if and only if these tasks are similar according to O*NET. Two jobs



(a) Electro-Mechanical Technicians

(b) Environmental Engineering Technicians

Figure 2: The number of tasks and the frequencies assigned to them can differ significantly even for related jobs.

with related tasks are also called related. An example with 3 jobs is depicted in Figure 1.

O*NET provides us for each task t_i^k with the information how often it is performed. This information was gathered by asking job incumbents and occupational experts. The options are "yearly or less", "more than yearly", "more than monthly", "more than weekly", "daily", "several times daily", and "hourly or more". O*NET provides a percentage for each of the 7 options. We denote these frequencies of task t_i^k with $f_1(t_i^k), \ldots, f_7(t_i^k)$. Since these values are percentages, for every task t_i^k they sum up to 100%, i.e., $\sum_{\ell=1}^{7} f_\ell(t_i^k) = 1$.

Each job j_i has a given probability $p(j_i)$ to be automated. We want to use $p(j_i)$ to calculate a probability to be automated for each task of this job.

4 From Task Frequencies to Task Shares

We use the frequencies with which a task is performed to assign each task t_i^k its share $s(t_i^k)$. For every task t_i^k, the share denotes how much time is spent doing this task, such that $\sum_{t_i^k \in T_i} s(t_i^k) = 1$. The frequency values from O*NET do not fulfill this property; their values are very consistent for one job, but they can vary a lot between different jobs and might even seem to contradict each other. An extreme example can be seen in Figure 2. The seven frequency options provided by O*NET are on the x-axis and on the y-axis is the corresponding value of each option.

To make use of the high consistency within a job, we decided that the share of a task is a weighted average of its frequencies, i.e., $s(t_i^k) := \sum_{\ell=1}^{7} x_i^\ell f_\ell(t_i^k)$. We want to calculate the job specific coefficients x_i^ℓ. Let us illustrate these coefficients with a simple example. If $x_i^7 = 0.1$, then a task t_i^k that is done exclusively "hourly or more" (i.e., $f_7(t_i^k) = 1$) makes up 10% of job j_i.

We want these coefficients to satisfy a few assumptions. If O*NET states that

a task is done "hourly or more", then the share of this task should be higher than the share of a task that is done "several times daily". This translates to $x_i^\ell \leq x_i^{\ell+1}$ $\forall \ell \in \{1, \ldots, 6\}$ and $0 \leq x_i^1$.

These constraints neither use that jobs are related nor do they define the coefficients uniquely. Both issues are solved if we require the coefficients x_i^ℓ and $x_{i'}^\ell$ for two related jobs j_i and $j_{i'}$ to be similar. The intuition behind this is that the frequencies of O*NET for related jobs are not independent of each other either, but rather should be similar as well. Occupational experts who have rated the frequency in which a task is done for one job, are likely to have rated the frequencies of related jobs.

The coefficients cannot be identical without violating the other constraints. Jobs have a different number of tasks and the frequencies are task specific. The example in Figure 2 highlights this. It is therefore easy to see that we cannot have the same coefficients for two related jobs and fulfill the equality $\sum_{t_i^k \in T_i} s(t_i^k) = 1$ for both jobs simply because the number of tasks can differ a lot.

Thus, we allow a bit of slack in the coefficients of related jobs. We use the variable $x_{i,i'}^\ell$ to express the difference between the coefficients x_i^ℓ and $x_{i'}^\ell$ for two related jobs $j_i, j_{i'}$. Formally, we define it as $x_{i,i'}^\ell := \max\{x_i^\ell - x_{i'}^\ell, x_{i'}^\ell - x_i^\ell\}$. This yields the following linear program, which minimizes the overall slack:

$$
\begin{aligned}
\text{minimize} \quad & \sum x_{i,i'}^\ell & & \\
\text{s.t.} \quad & & & \\
x_{i,i'}^\ell \geq \quad & x_i^\ell - x_{i'}^\ell & \forall \ell & \\
& & \forall j_i, j_{i'} \in J \text{ that are related} & \\
x_{i,i'}^\ell \geq \quad & x_{i'}^\ell - x_i^\ell & \forall \ell & \\
& & \forall j_i, j_{i'} \in J \text{ that are related} & \\
\sum_{t_i^k \in T_i} \sum_{\ell=1}^{7} x_i^\ell f_\ell(t_i^k) \leq \quad & 1 + \varepsilon & \forall j_i \in J & \\
\sum_{t_i^k \in T_i} \sum_{\ell=1}^{7} x_i^\ell f_\ell(t_i^k) \geq \quad & 1 - \varepsilon & \forall j_i \in J & \\
x_i^1 \geq \quad & 0 & \forall j_i \in J & \\
x_i^\ell \geq \quad & x_i^{\ell-1} & \forall j_i \in J \qquad \ell \in \{2, \ldots, 7\} &
\end{aligned}
$$

We set ε to 0.01. The resulting LP has 169,372 variables in its objective function. Since there are 735 jobs,[2] this means that a job is related to approximately 32.9

[2]We consider slightly more jobs than Frey and Osborne, since we use the finest granularity available from O*NET.

other jobs on average. The value of the objective function is 24.6, i.e., for two related jobs the coefficients differ only by 0.000145 on average. For comparison, the average value of a coefficient is 0.060. Our complete results can be found online at `http://jobs-study.ethz.ch`.

5 From Jobs to Tasks

Knowing the shares of the tasks enables us to set up a linear program that calculates for each task the probability to be automated. We want that the weighted average of the automation probabilities $p(t_i^k)$ of the tasks of a job j_i can explain the automation probability $p(j_i)$ of the job, i.e., $\sum_{t_i^k \in T_i} p\left(t_i^k\right) \cdot s\left(t_i^k\right) \approx p\left(j_i\right)$. Furthermore, we want to assign related tasks similar automation probabilities. To do this, we define a variable $t_{i,i'}^{k,k'}$ for each pair of related tasks t_i^k and $t_{i'}^{k'}$. It denotes the probability difference that we assign to the two tasks. Formally, it is defined as $t_{i,i'}^{k,k'} := \max\{p(t_i^k) - p(t_{i'}^{k'}), p(t_i^k) - p(t_{i'}^{k'})\}$. We want to minimize the sum of these variables, i.e., the sum of the probability difference of all related tasks.

Combining these requirements with necessary conditions to have meaningful probabilities, i.e., $0 \leq p(t_i^k) \leq 1$, yields the following linear program:

$$
\begin{aligned}
\text{minimize} \quad & \sum t_{i,i'}^{k,k'} & & \\
\text{s.t.} \quad & & & \\
p(t_i^k) - p(t_{i'}^{k'}) \leq \quad & t_{i,i'}^{k,k'} & & \forall t_i^k, t_{i'}^{k'} \text{ that are related} \\
p(t_{i'}^{k'}) - p(t_i^k) \leq \quad & t_{i,i'}^{k,k'} & & \forall t_i^k, t_{i'}^{k'} \text{ that are related} \\
\sum_k p(t_i^k) \cdot s\left(t_i^k\right) \leq \quad & p(j_i)\,(1+\varepsilon) & & \forall j_i \in J \\
\sum_k p(t_i^k) \cdot s\left(t_i^k\right) \geq \quad & p(j_i)\,(1-\varepsilon) & & \forall j_i \in J \\
\\
p(t_i^k) \geq \quad & 0 & & \forall j_i \in J, t_i^k \in T_i \\
p(t_i^k) \leq \quad & 1 & & \forall j_i \in J, t_i^k \in T_i \\
t_{i,i'}^{k,k'} \geq \quad & 0 & & \forall t_{i,i'}^{k,k'} \\
t_{i,i'}^{k,k'} \leq \quad & 1 & & \forall t_{i,i'}^{k,k'}
\end{aligned}
$$

We set ε to 0.01.

6 Linear Program Results

We now analyze the results of the linear program as described above. Later on, we will look at a small refinement to automatically detect outliers in our results.

6.1 Task Probabilities

The linear program as described in Section 5 has 105,748 variables in its objective function and it has a minimal value of 9,846. This means that two related tasks differ, on average, with regard to their probability by 9.3%. The complete results can be found online at `http://jobs-study.ethz.ch`. The histogram of the probability difference between two related tasks is shown in Figure 3. A majority of related tasks is assigned a similar probability. A small fraction of related tasks is assigned diametrically opposed probabilities, which seems startling. It can be reconciled by considering that neither the classification by Frey and Osborne nor the classification of tasks being related by O*NET are perfect.

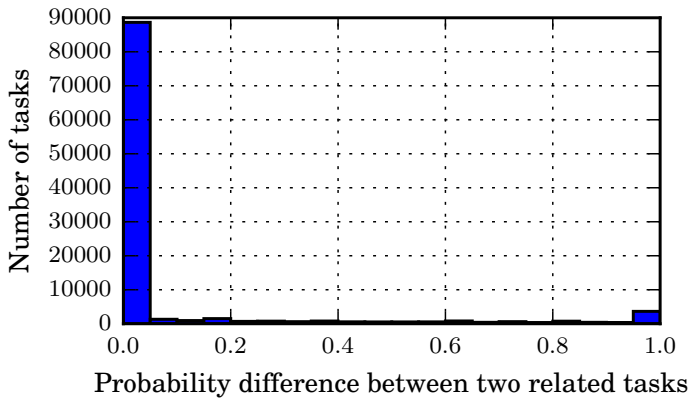


Figure 3: A histogram of the probability with which two related tasks differ.

One example that highlights this are the two jobs *computer programmer* and *software developers, applications*. These two jobs have many related tasks, but the probabilities of these jobs differ a lot (4% for *software developers, applications*, 48% for *computer programmers*). Hence, the diametrically opposed probabilities are necessary to meet the constraints of the linear program.

In the following, we present a few selected jobs to illustrate our results. The first example is *chemists*. This job has an automation probability of 10% according to Frey and Osborne. Only one task has, according to our linear program, a high probability of being automated: “Induce changes in composition of substances by introducing heat, light, energy, or chemical catalysts for quantitative or qualitative

Task Description	p	Share
Write decisions on cases.	1	5.1
Instruct juries on applicable laws, direct juries to deduce the facts from the evidence presented, and hear their verdicts.	1	3.4
Monitor proceedings to ensure that all applicable rules and procedures are followed.	1	8.0
Advise attorneys, juries, litigants, and court personnel regarding conduct, issues, and proceedings.	1	6.2
Interpret and enforce rules of procedure or establish new rules in situations where there are no procedures already established by law.	1	5.4
Conduct preliminary hearings to decide issues such as whether there is reasonable and probable cause to hold defendants in felony cases.	1	3.9
Rule on admissibility of evidence and methods of conducting testimony.	0.94	5.3
Preside over hearings and listen to allegations made by plaintiffs to determine whether the evidence supports the charges.	0.46	5.9
Perform wedding ceremonies.	0.39	2.7
Read documents on pleadings and motions to ascertain facts and issues.	0	10.1
Research legal issues and write opinions on the issues.	0	6.5
Settle disputes between opposing attorneys.	0	4.6
Participate in judicial tribunals to help resolve disputes.	0	6.6
Rule on custody and access disputes, and enforce court orders regarding custody and support of children.	0	6.3
Sentence defendants in criminal cases, on conviction by jury, according to applicable government statutes.	0	4.0
Grant divorces and divide assets between spouses.	0	4.7
Award compensation for damages to litigants in civil cases in relation to findings by juries or by the court.	0	3.8
Supervise other judges, court officers, and the court's administrative staff.	0	8.5

Table 4: The automation probability and the share of each task of *Judges, Magistrate Judges, and Magistrates"*. The automation probability of this job is 40%.

analysis." Other simple mechanical tasks have been assigned low automation probabilities. We will revisit this job in Section 6.2.

Next up: *judges*. Their automation probability is 40%. The tasks, their probabilities, and their shares are shown in Table 4. The tasks that can be automated can be grouped in two sets: preliminary hearings which includes making first assessments, and ensuring that the procedures in court are followed. The tasks that involve sentencing (or the preparation thereof) have been assigned low automation probabilities.

Figure 5 shows the histogram of the probabilities from our linear program. The probabilities for most tasks are either very high or very low and only a few tasks have a probability in-between. This desired side effect of our linear program helps us to achieve our goal of allowing job experts (and laymen) to argue about the validity of our results. We invite the reader to have a look at other jobs at `http://jobs-study.ethz.ch`.

6.2 Outlier Detection

To evaluate our approach and check for outliers, we use a variant of cross-validation. For every job j_i, we create a linear program without job j_i. This yields a probability

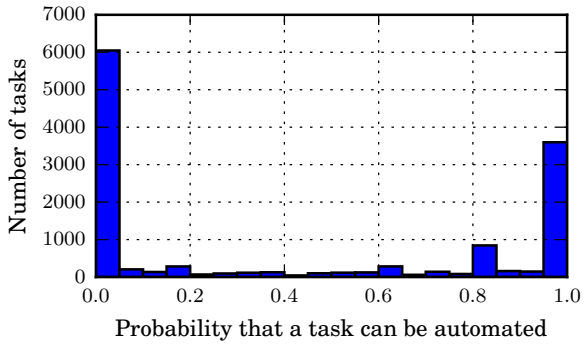


Figure 5: A histogram of the probabilities that tasks can be automated. Nearly all tasks are assigned either 0 or 1. This simplifies arguing whether our classification is correct.

$p_i(\cdot)$ for every task but the tasks from j_i. Afterward, we calculate the new probability $p'(j_i)$ that job j_i can be automated. We do this by setting the probability $p'(t_i^k)$ of each task t_i^k to the average of all tasks that are related to it. We denote the set of related tasks by $N(t_i^k)$. Formally, we set $p'(t_i^k) := \frac{1}{|N(t_i^k)|} \sum_{t_{i'}^{k'} \in N(t_i^k)} p_i(t_{i'}^{k'})$.

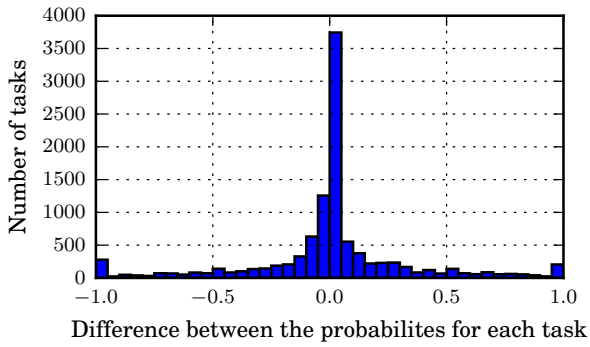


Figure 6: A histogram of $p(t_i^k) - p'(t_i^k)$ for all tasks. This plot shows that most tasks have a similar probability in both approaches. The values are sharply distributed around 0.

We first compare $p'(t_i^k)$ with $p(t_i^k)$. The difference between these two probabilities should be small for the majority of the tasks. This is indeed what can be seen in Figure 6. The histogram of $p'(t_i^k) - p(t_i^k)$ shows that nearly all tasks have similar probabilities in both approaches. The average absolute difference is less than 20%. The distribution is centered around 0. Its mean is less than 0.05%.

By combining the new probability of each task with its share, we can calculate

the new probability of job j_i by using a weighted average. This allows us to compare $p'(j_i)$ with $p(j_i)$.

We have plotted this difference, i.e., $p(j_i) - p'(j_i)$, in Figure 7. We can see that the difference is centered around 0%; with the average absolute difference being less than 20%. For more than half of the jobs our probability differs by less than 20% from Frey and Osborne [12]. Most interesting are the jobs whose probability differs significantly. We now have a look at a few of them.

There are jobs where our probability is more than 80% smaller than the one by Frey and Osborne. One job is *compensation and benefits managers*. We assigned it a probability $p'(j)$ to be automated of 9.1%; compared to $p(j) = 96\%$ by Frey and Osborne. We do not claim to know the true value, but we can look at the job and compare it to the probabilities of jobs we consider similar. Notice that this is conceptually similar to what our linear program does and thus might be biased. The tasks of this job are shown in Table 8. If we manually compare them to related tasks, we conclude that they do not seem to be automatable in the next few decades. We do favor our result over the result of Frey and Osborne.

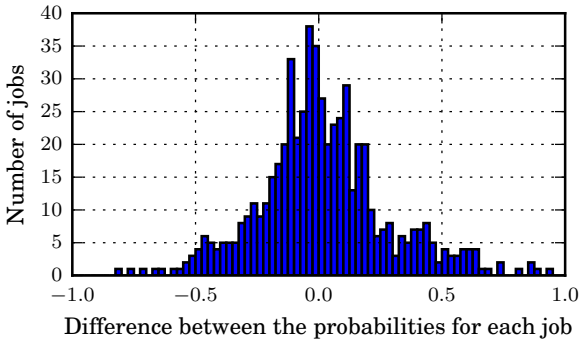


Figure 7: A histogram of the difference between the probability by Frey and Osborne with our probability. The distribution is centered around 0 and a majority of the jobs differs by less than 20%.

There is only one job that we assign a much higher probability than Frey and Osborne. The job *First-Line supervisors of production and operating workers* has been assigned a 83% automation probability by us and only 1.6% by Frey and Osborne. A close inspection of the tasks makes us believe that the true value is between these extremes. Quite a few of the tasks are clearly automatable, e.g., "Keep records of employees' attendance and hours worked." and "Observe work and monitor gauges, dials, and other indicators to ensure that operators conform to production or processing standards." Others, e.g., "Read and analyze charts, work orders, production schedules, and other records and reports to determine

Task Description	p	p'
Advise management on such matters as equal employment opportunity, sexual harassment and discrimination.	1	0.15
Study legislation, arbitration decisions, and collective bargaining contracts to assess industry trends.	1	0
Fulfill all reporting requirements of all relevant government rules and regulations, including the Employee Retirement Income Security Act (ERISA).	1	0.20
Investigate and report on industrial accidents for insurance carriers.	1	0.12
Represent organization at personnel-related hearings and investigations.	1	0
Analyze compensation policies, government regulations, and prevailing wage rates to develop competitive compensation plan.	1	0.5
Mediate between benefits providers and employees, such as by assisting in handling employees' benefits-related questions or taking suggestions.	1	0.42
Prepare detailed job descriptions and classification systems and define job levels and families, in partnership with other managers.	1	0
Prepare personnel forecasts to project employment needs.	1	0
Direct preparation and distribution of written and verbal information to inform employees of benefits, compensation, and personnel policies.	1	0
Manage the design and development of tools to assist employees in benefits selection, and to guide managers through compensation decisions.	1	0
Design, evaluate and modify benefits policies to ensure that programs are current, competitive and in compliance with legal requirements.	1	0
Administer, direct, and review employee benefit programs, including the integration of benefit programs following mergers and acquisitions.	1	0
Prepare budgets for personnel operations.	1	0.03
Maintain records and compile statistical reports concerning personnel-related data such as hires, transfers, performance appraisals, and absenteeism rates.	1	0
Contract with vendors to provide employee services, such as food services, transportation, or relocation service.	1	0.38
Identify and implement benefits to increase the quality of life for employees, by working with brokers and researching benefits issues.	1	0
Plan, direct, supervise, and coordinate work activities of subordinates and staff relating to employment, compensation, labor relations, and employee relations.	1	0
Negotiate bargaining agreements.	1	0.67
Plan and conduct new employee orientations to foster positive attitude toward organizational objectives.	1	0
Conduct exit interviews to identify reasons for employee termination.	1	0
Develop methods to improve employment policies, processes, and practices, and recommend changes to management.	0.51	0
Formulate policies, procedures and programs for recruitment, testing, placement, classification, orientation, benefits and compensation, and labor and industrial relations.	0.23	0.01

Table 8: The tasks and their corresponding probability that they will be automated for *Compensation and Benefits Managers* according to our original linear program and the cross-validation.

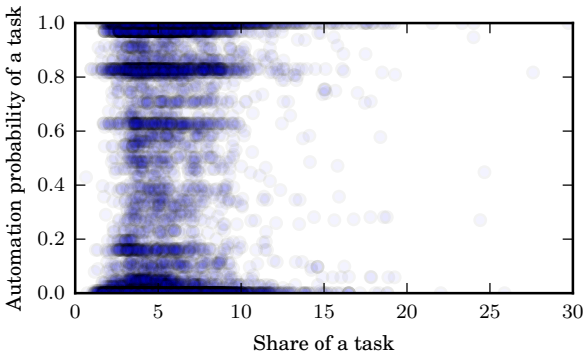


Figure 9: Our results show almost no correlation between the share of a task and its probability that it will be automated.

production requirements and to evaluate current production estimates and outputs." seem difficult to automate. The complete results for this can job be found at `http://jobs-study.ethz.ch`.

We continue by comparing the previous results with the approach described in this section. To do this, we return to the jobs that we have looked at previously. First off is the job *chemists*. The automation probability of most tasks has increased. Consequently, the automation probability of this job has increased from 10% to 42%. Due to the large difference, this job should be analyzed in-depth by job experts.

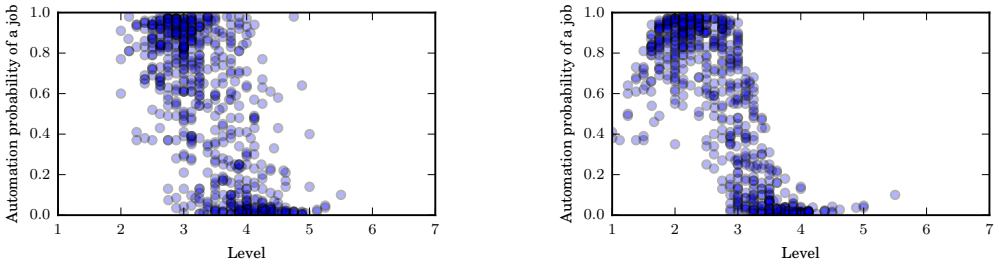
The changes in the automation probability of the tasks of *judges* are much smaller. Most tasks have a similar automation probability as before and the overall probability of this job has changed marginally, i.e., increased only from 40% to 50%. Therefore, we are confident that the classification by Frey and Osborne is correct.

We conclude that our approach can also be used to detect outliers in the results of Frey and Osborne. We can then manually inspect the automation probabilities of the tasks of such an outlier to determine the truth. We think our results allow us to fine tune the results from Frey and Osborne, but not replace it, as we need their results to bootstrap our linear program.

7 Further Analysis

In addition to inspecting every task of every job, we consider a broader picture. We do this by looking at general properties of a job that correlate with the probability that it can be automated.

7.1 Tasks



(a) Deductive Reasoning

(b) Originality

Figure 10: The probability that a job can be automated over the level of the abilities "deductive reasoning" and "originality" used in this job. These levels are defined by O*NET and for each they provide an anchor point. Level 2 of "deductive reasoning" means "knowing that a stalled car can coast downhill" and level 5 "deciding what factors to consider in selecting stocks". Level 2 of "originality" means "using a credit card to open a locked door" and level 6 "inventing a new type of man-made fiber". Every point represents one job. A higher level of either of these two abilities correlates, as expected, with a lower probability to be automated.

We first analyze the share of a task. The higher the share, the more often a task is performed. Hence, from a machine learning perspective this means that much more training data is available. This might lead to the conclusion that such a task is easier to automate. To disprove this claim, we plotted the share of a task over the probability that a task can be automated according to our linear program. The resulting graph is shown in Figure 9. Every dot represents one task, with its share on the x-axis and its probability on the y-axis. We see that there is barely any correlation between these two. We conclude that tasks that are done more frequently are not more likely to be automated.

7.2 Jobs

We continue our analysis by looking at the correlation between the properties that a job has, e.g., what kind of degree is necessary to do a job, and the probability that this job can be automated. Correlation does not imply causation, but nevertheless, these results reveal some interesting nuggets.

O*NET provides the level that the ability "deductive reasoning" is used in a job. The level ranges from 1 to 7, where for example level 2 means "knowing that a stalled car can coast downhill" and level 5 "deciding what factors to consider

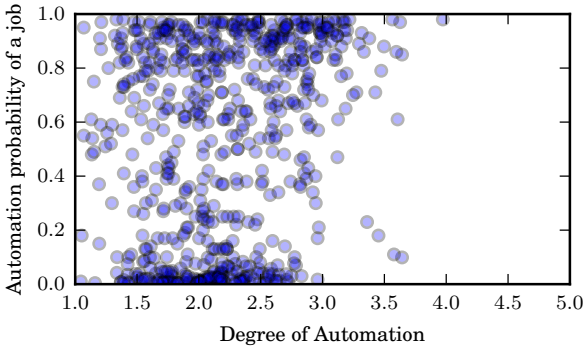
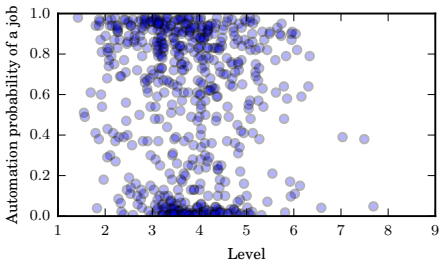
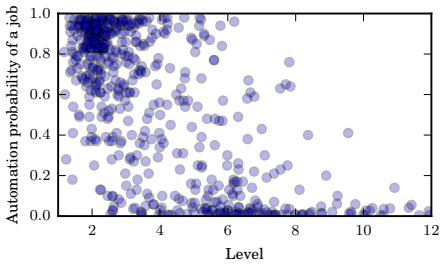


Figure 11: The probability that a job can be automated over the level of the current "Degree of Automation". This level ranges from 1 (Not at all automated) to 5 (Completely automated). The rather small correlation of 0.23 implies that different jobs will soon be affected.

in selecting stocks". For every job, we have one value between 1 and 7. The resulting graph can be seen in Figure 10. Every job is represented by one dot; its x-coordinate being its level and the y-coordinate its probability. We can see that jobs that require a high level of deductive reasoning tend to have a lower probability of being automated. A similar result can be seen for "originality" (see Figure 10).



(a) On the Job Training



(b) Required Level of Education

Figure 12: The probability that a job can be automated over the amount of "On the Job Training", which ranges from 1 (none or short demonstration) to 9 (over 10 years) and the "Required of Level of Education", which ranges from 1 (less than a high school diploma) to 12 (post-doctoral training).

Level 2 of "originality" means "using a credit card to open a locked door" and level 6 means "inventing a new type of man-made fiber". This confirms our expectation that these abilities will remain difficult for a computer.

O*NET even has an explicit value for the current level of the "degree of

automation" for each job. This level ranges from 1 (not at all automated) to 5 (completely automated). As depicted in Figure 11, the already existing level of automation barely correlates with the probability that this job will be automated. This indicates that not only jobs that are already affected by automation are in danger, but also a whole new set of jobs. This is aligned with the recent worries about many new jobs soon being affected by computerization.

We conclude this section by looking at the effect that the level of required education for a job has on the probability to be automated. Jobs that require only very little education (level 1, i.e., less than a high school diploma) tend to have a higher probability than jobs that require an associate degree (level 5) which in turn have a higher probability than jobs that require post-doctoral training (level 12). Most jobs that require little education are in danger. It is noteworthy that the effect of training before the job is much stronger than the effect of on the job training. Jobs that require more on the job training only have a marginally smaller probability to be automated. Both plots are shown in Figure 12.

8 Existing Trends

In this section we analyze to what extent automation is already happening. To investigate this, we used historical job data from 2001 until 2015. For every year we know for every job in the O*NET database how many people were employed. Instead of using the absolute values (which tend to increase for most jobs since more and more people live and work in the US), we consider the relative values and look at the percentage of how many people are employed in this job. We consider the fraction a job j_i has in 2001 and focus on the factor c_i by which this fraction has changed in 2015. Note that due to the financial crisis, where a lot of jobs have been lost, the data shows to some extend how recession-proof a job is.

We plot the factor c_i against the automation probability of this job for every job j_i. Every job is represented by a dot; its x-coordinate being the factor c_i and the y-coordinate being its automation probability. We expect that jobs with a high automation probability tend to have a factor smaller than one, i.e., its fraction of the labor market has decreased and vice versa. The result can be seen in Figure 13. Even though there are quite a few outliers, the overall trend follows our expectation. The most extreme outlier is the job *service unit operators, oil, gas, and mining* whose fraction has increased by more than a factor of 4, but is deemed 93% automatable. This can easily be explained due to the rise of fracking. A job that is considered non-automatable, but whose fraction has been reduced to about a third of its initial value, is *advertising and promotions managers*. We suspect that this job is highly susceptible to recessions, i.e., the financial crisis in 2008/2009.

The above analysis shows that the demand for jobs that are highly automatable

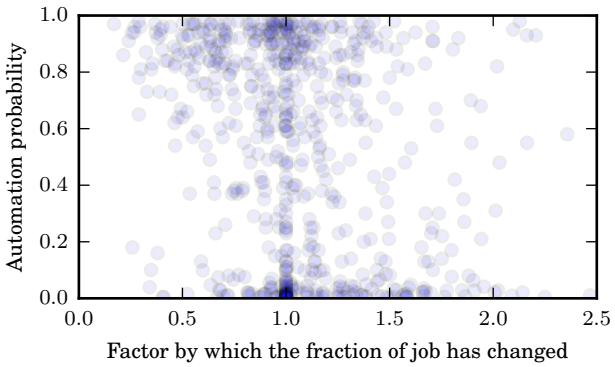
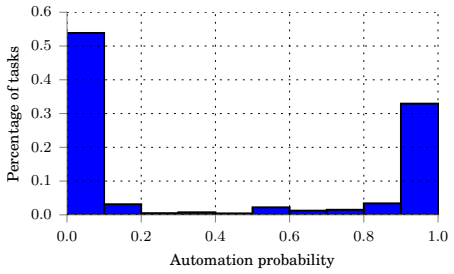


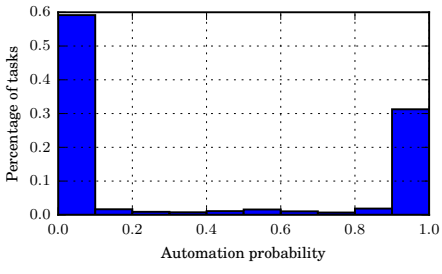
Figure 13: The factor c_i by which the fraction of a job has changed over the automation probability. We can see that jobs that have a high automation probability.

has already started to decrease. We once again look at the problem from a task level perspective. We assume that the demand for a job decreases because the demand for highly automatable tasks decreases since some of these tasks are already being automated.

Our analysis works as follows: We initially set the share $s\left(t_i^k\right)$ of a task t_i^k as described in Section 4. We allow the share $s(t_i^k)$ of a task t_i^k to change by a factor which we denote by c_i^k. The weighted average of the adjusted share should be equal to c_i, i.e., $\sum_{k \in T_i} c_i^k \cdot s\left(t_i^k\right) \approx c_i$ to reflect the change in demand for a job. We assume that if one task can already be automated, i.e., has a factor smaller than one, then related tasks also tend to decrease and vice versa. We want to minimize the overall difference $s_{i,i'}^{k,k'} := |c_i^k - c_{i'}^{k'}|$ between two related tasks $t_i^k, t_{i'}^{k'}$.



(a) Tasks whose share decreases by at least a factor of 0.5



(b) Tasks whose share increases by at least a factor of 2

Figure 14: The histogram of the automation probability of all tasks whose share either decreases by at least factor of 0.5 or increases by at least a factor of 2.

This is modeled in the following linear program:

$$
\begin{aligned}
\text{minimize} \quad & \sum s_{i,i'}^{k,k'} & & \\
\text{s.t.} \quad & & & \\
c_i^k - c_{i'}^{k'} \leq \quad & s_{i,i'}^{k,k'} & & \forall t_i^k, t_{i'}^{k'} \text{ that are related} \\
c_{i'}^{k'} - c_i^k \leq \quad & s_{i,i'}^{k,k'} & & \forall t_i^k, t_{i'}^{k'} \text{ that are related} \\
\sum_k c_i^k \cdot s\left(t_i^k\right) \leq \quad & c_i\,(1+\varepsilon) & & \forall j_i \in J \\
\sum_k c_i^k \cdot s\left(t_i^k\right) \geq \quad & c_i\,(1-\varepsilon) & & \forall j_i \in J \\
c_i^k \geq \quad & 0 & & \forall j_i \in J, t_i^k \in T_i
\end{aligned}
$$

To verify our claim, we first look at tasks whose factor c_i^k is less than 0.5, i.e., whose share has more than halved. We expect that these tasks to have a high automation probability. The histogram is depicted in Figure 14. Similarly, tasks whose share has increased by more than a factor of 2 should have a low automation probability. The results are also shown in Figure 14. Despite various other economic factors that come into play like economic cycles or major political events, we can see that tasks whose share has more than doubled tend to have a smaller automation probability.

9 Conclusion

We believe that automation is one of the main challenges for society. In our opinion, the seminal work of Frey and Osborne did an excellent job of getting the discussion going. In this paper we dug a bit deeper, by looking not only at jobs – but at the tasks that make up a job. We hope that opening the Frey/Osborne black box will contribute to the discussion. The professionals that are actually doing a job are the main experts to decide what parts of their job can or cannot be computerized. The Frey/Osborne work only tells these experts that their job is 87% automatable, but what does it actually mean? With our work, job experts can look inside the box, and understand which tasks of their job are at risk. Our hope is that the job experts have a discussion which results are believable and which are not, and why. To facilitate this discussion, we have created a web page (`http://jobs-study.ethz.ch`) that allows users to comment upon our results.

References

[1] Michael Auli, Michel Galley, Chris Quirk, and Geoffrey Zweig. Joint language and translation modeling with recurrent neural networks. In *EMNLP*, volume 3, page 0, 2013.

[2] David H. Autor and David Dorn. The growth of low skill service jobs and the polarization of the us labor market. Technical report, National Bureau of Economic Research, 2009.

[3] David H. Autor, Lawrence F Katz, and Alan B Krueger. Computing inequality: have computers changed the labor market? Technical report, National Bureau of Economic Research, 1997.

[4] David H Autor, Frank Levy, and Richard J Murnane. The skill content of recent technological change: An empirical exploration. Technical report, National Bureau of Economic Research, 2001.

[5] Holger Bonin, Terry Gregory, and Ulrich Zierahn. Übertragung der Studie von Frey/Osborne (2013) auf Deutschland. Technical report, ZEW Kurzexpertise, 2015.

[6] John Bound and George E Johnson. Changes in the structure of wages during the 1980's: An evaluation of alternative explanations. Technical report, National Bureau of Economic Research, 1989.

[7] Erik Brynjolfsson and Andrew McAfee. *Race against the machine: How the digital revolution is accelerating innovation, driving productivity, and irreversibly transforming employment and the economy*. Brynjolfsson and McAfee, 2012.

[8] Erik Brynjolfsson and Andrew McAfee. *The second machine age: work, progress, and prosperity in a time of brilliant technologies*. WW Norton & Company, 2014.

[9] Kerwin Kofi Charles, Erik Hurst, and Matthew J Notowidigdo. Manufacturing decline, housing booms, and non-employment. Technical report, National Bureau of Economic Research, 2013.

[10] L Deng, D Yu, and G Hinton. Deep learning for speech recognition and related applications. In *NIPS Workshop*, 2009.

[11] Martin Ford. *Rise of the Robots: Technology and the Threat of a Jobless Future*. Basic Books, 2015.

[12] Carl Benedikt Frey and Michael A. Osborne. The future of employment: How susceptible are jobs to computerisation?, 2013.

[13] Maarten Goos and Alan Manning. Lousy and lovely jobs: The rising polarization of work in britain. *The review of economics and statistics*, 89(1):118–133, 2007.

[14] Erico Guizzo. How google's self-driving car works. *IEEE Spectrum Online, October*, 18, 2011.

[15] John Maynard Keynes. Economic possibilities for our grandchildren (1930). *Essays in persuasion*, pages 358–73, 1933.

[16] Quoc V Le. Building high-level features using large scale unsupervised learning. In *ICASSP*, pages 8595–8598. IEEE, 2013.

[17] James Manyika, Michael Chui, Jacques Bughin, Richard Dobbs, Peter Bisson, and Alex Marrs. *Disruptive technologies: Advances that will transform life, business, and the global economy*, volume 12. McKinsey Global Institute New York, 2013.

[18] Pajarinen Mika and Anders Ekeland. Computerization and the future of jobs in norway. 2015.

[19] Marvin L Minsky. *Computation: finite and infinite machines*. Prentice-Hall, Inc., 1967.

[20] David Silver, Aja Huang, Christopher J. Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel, and Demis Hassabis. Mastering the game of go with deep neural networks and tree search. *Nature*, 529:484–503, 2016.

[21] Alan M Turing. Computing machinery and intelligence. *Mind*, 59(236):433–460, 1950.

[22] Moshe Y. Vardi. The great robotics debate. *Commun. ACM*, 56(7), July 2013.

[23] Moshe Y Vardi. Is information technology destroying the middle class? *Commun. ACM*, 58(2):5, 2015.

[24] Moshe Y. Vardi. The moral imperative of artificial intelligence. *Commun. ACM*, 59(5), April 2016.

[25] Adrian Wood. North-south trade, employment and inequality. *New York*, 1994.

[26] Adrian Wood. Globalisation and the rise in labour market inequalities. *The economic journal*, 108(450):1463–1482, 1998.