# Transitive Identity Mapping using Force-Based Clustering

H. Van Dyke Parunak, Sven Brueckner

Soar Technology, 3600 Green Court, Suite 600, Ann Arbor, MI 48105
{van.parunak, sven.brueckner}@soartech.com

**Abstract.** In most information retrieval systems, software processes (whether agent-based or not) reason about passive items of data. An alternative approach instantiates each record as an agent that actively self-organizes with other agents (including queries). Imitating the movement of bodies under physical forces, we describe a distributed algorithm ("force-based clustering," or FBC) for dynamically clustering and querying large, heterogeneous, dynamic collections of entities. The algorithm moves entities in a virtual space in a way that estimates the transitive closure of the pairwise comparisons. We demonstrate this algorithm on a large, heterogeneous collection of records, each representing a person. We have some information about a person of interest, but no record in the collection directly matches this information. Application of FBC identifies a small subset of records that are good candidates for describing the person of interest, for further manual investigation and verification.

**Keywords**: clustering; transitive mapping; self-organizing information; active data

## 1 Introduction

Many real-world search problems require inexact matches against heterogeneous data sources, where no single data source can answer the query. For example:

> An unidentified male visits a clinic, signing in with an illegible signature and partially illegible phone number, then leaves the clinic before being seen. Later, staff discovers that another patient has symptoms of an influenza-like illness consistent with a potentially deadly and highly contagious virus. Staff initiates quarantine to limit close contact until the laboratory confirms diagnosis. In reviewing the sign-in log, staff discover the visitor's entry. Patients and staff cannot recall any supporting information about the unidentified individual, who is at risk and must be identified as quickly as possible.

An anonymous sponsor provided us with eight databases (DBs) (Table 1), containing varying combinations of name, address, phone number, and DB-specific record identifiers (Doc-IDs) for fictitious individuals, but concealing the identity of the POI. SRLU has 130 records, and the others have on the order of 50,000 each. Cursory examination shows that some keys are shared both within and across databases, and

Table 1: Databases.—A '?' indicates that the field is present in only some records of the DB.

| DB name | Available Fields | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | Last name | Middle name | Middle initial | First name | Street # | Street | City | State | Zip | Phone | Doc ID |
| CCCR | X | X | | X | X | X | X | X | X | | X |
| CCTR | | | | | X | X | X | X | X | | X |
| HPA | X | | | X | | X | X | X | X | | |
| HR | | | | | X | X | X | X | X | X | |
| ID | X | X | | X | X | X | X | X | X | | X |
| SRLU | X | | X | X | X | X | X | X | X | X | |
| TR | X | | ? | X | | | | | | X | X |
| WP | X | | | X | X | X | X | X | X | X | |

some names appear to be variant spellings (e.g., "Tom F. Tuk" and "Tolman Fredegar Took" share other information). Only the last two digits of the phone number are illegible, but 104 records have phone numbers that could match the available number, some associated with different names or addresses, suggesting errors in the data. These phone numbers are in states different than the clinic. Our task is to develop a prioritized list of people with contact information whom authorities should contact in order to reach the mystery patient as quickly as possible.

Constructing and reasoning over such scenarios is combinatorially prohibitive, and too slow for emergencies (such as tracking an epidemic or disrupting a terrorist attack. Force-Based Clustering (FBC) instantiates each record as a software agent in an abstract low-dimensional space (a three-dimensional torus wrapped in four dimensions). The agents compute virtual "forces" among themselves, and move in response to those forces. The transitivity of these forces brings together agents whose similarity may not be documented directly, but that are linked by a chain of similar agents.

Section 2 defines our algorithm. Section 3 relates it to other techniques. Section 4 reports its performance. Section 5 concludes.

## 2 Technical Approach

We summarize the motivation for and implementation of FBC.

### 2.1 Motivation

FBC is motivated by physical forces, which show several characteristics:

- If entities are close, they repel one another (mutual exclusion).
- If they are far apart, their interaction rapidly decreases (depending on the physical force involved, as the square or even higher powers of the separation).
- Multiple types of forces can contribute concurrently to interactions.

Each record in our database is an agent. We distribute them in an abstract space, define virtual forces among them, and let them move. Similar records move closer to one another, pulling their neighbors with them (and thus providing transitive closure). We query the system by inserting a query record that contains what we know about

the POI, letting the system converge, and retrieving records that end up close to the query. The closer a record is to the query, the higher we rank it in our list of persons to contact. This approach emulates physical movement:

- Extremely close agents repel one another, keeping similar records from collapsing to the same location, in spite of attractive forces among them.
- The decrease of interaction strength with distance means that most interactions are among nearby agents. This locality of interaction facilitates convergence of any digital algorithm for computing agent movement. Physical forces act all at once, but an algorithm must manipulate a subset of agents at each time step. Local interactions reduce the set of agents with which a given agent effectively interacts, allowing their influence to be felt in fewer steps.
- The concept of multiple forces lets us handle heterogeneous records with varying field contents, by defining a "last-name force," an "address force," and a "phone-number-force." Integration of these forces through agent movement allows transitive interactions among records that do not directly overlap. For example, imagine that record A has only address and DB key information (database "CCTR"), B has address and name ("CCCR"), and C has name and phone number ("TR"). The "address force" will bring A and B together, the "name" force will bring B and C together, and as a result, A and C come close together, suggesting a link between the phone number in C with the DB key in A.

## 2.2 Implementation

Our implementation includes similarity computation, force definition, distributed execution, and convergence detection.

**Similarity Computation**

From the union of the fields in the databases, we define nine complex attributes in [0, 1], each derived from one or more raw attribute fields. These rules take care of missing simple attributes (e.g., a full-name complex attribute with no middle name) and may also employ external data sources in the similarity calculation. The overall similarity between two records is a weighted sum of the component similarities.

The similarity computations and weights we assign to each complex attribute reflect our understanding about the contribution each can make to the challenge.

*Edge*.—The data set provides not only ~350k records, but also a pre-computed set of ~58k similarity-weighted edges based on the record attributes. The "Edge" complex attribute is the total composite score of two records as recorded in this table. If the table does not specify an edge between two records, their "Edge" similarity is zero.

*Source*.—Though we do not know the meaning of the various databases, we must combine them to achieve transitive closure. Differences in record structure may reduce the similarity score based on substantive fields. To encourage exploration of cross-databases similarities, we assign a similarity component of 1 between two records if their sources are different, and 0 if they are the same.

*Full Name*.—A person's name is the most specific and semantically meaningful identifier. Between two "Full Name" attributes we compute the similarity based on the presence and match of the three component strings (0.5 for last name match, an additional 0.3 for first name match, and a final 0.2 for a middle name match). Our current implementation determines a name component match solely on (ignore-case) string identity; applying Soundex [12] is a natural extension.

*US State*.—The "State" attribute of a record offers a rough indicator of its geographic location. We define a static similarity measure between all states based on the normalized geographic distance of the latitude/longitude of their capitals. Records with identical state identifiers have a similarity of 1. Records for states with the largest geographic distance of capital cities, or records that do not identify a (known) state, all have a similarity value of 0. Other similarity values are proportional in-between.

*Zip*.—The US postal code system offers a finer approximation of the geographic location of this record than the US-State attribute. Similarity between two Zip attribute values is first established by identity (similarity=1). If the Zip codes are not identical but both populated, their similarity is based on the [0, 1] normalized geographic distance of the latitude/longitude coordinates of their respective geodesic centers. Unpopulated records result in a similarity of zero.

*Full Address*.—The "Full Address" complex attribute combines our various geographic estimates of the location a record references. Complete or partial matches of the components of the Full Address accumulate similarity contributions. Matches for State and Zip entries are provided by their respective complex attribute wrappers. City and street name strings are either identical or not. If street names are identical, then highly similar house numbers provide a small similarity bonus.

*Phone*.—Our only concrete identifier for the POI is a phone number, but the database contains some duplicate numbers, suggesting either shared phones or erroneous data. We accumulate units of similarity by first assessing the similarity of the area codes, then the prefix similarity, and finally the line code similarity. In this sequence, whenever there is not an identical match, further similarity accumulation will stop. For instance, there is no reason to compare line codes with different prefixes. If the area codes of two records already do not match, we use an external database of latitude/longitude coordinates of major cities or towns in the geographic coverage of the area code to provide a partial area-code similarity measure.

*Gender*.—The POI is male. While there is no explicit "Gender" field in the databases, the first name of a person, if provided, does provide an estimate of the likely gender. We extracted a database of 48k international first names that estimates likely gender as one of 5 values: Strongly Female, Possibly Female, Unknown, Possibly Male, Strongly Male. Using string matching with the provided First Name, we assign each record one of these five genders. If the first name is missing, we assign Unknown. This name database gives substantial coverage of the records in the challenge data set, once we add the genders of the main characters of J.R.R. Tolkin's *Lord of the Rings* trilogy. We manually defined a 5x5 similarity matrix between the gender identities, assigning a similarity of 1 for matching Strongly Female or Male records and decreasing similarity for more distant fields in the gender identity ordering.

*ID-DOC*.—We do not know the meaning of the ID-DOC keys, but a cursory survey of the data shows that some of these are shared, both within and across databases. Binary string similarity (0 or 1) is assessed.

**Force Definition**

The force between two agents has two components: one repulsive and one attractive. Each force is computed using the convex distance scaling function

$$g(d,m,s) = \frac{e^{s\frac{m-d}{m}} - 1}{e^s - 1}$$

where

- $d$ is the shortest distance on the torus between the two agents,
- $m$ is the maximum distance on the torus,
- $s$ is a shaping factor; as $s$ increases, force drops off more rapidly..

For similarity $\varphi$ between two agents, the force is

$$f(d,\varphi) = w_a\, d\, g(d,m,s_a)\varphi - w_b\, m\, g(d,m,s_b)\,(1-\varphi)$$

Figure 1 plots this force for $s_r$ = 5, $s_a$ = 6, $w_a$ = $w_r$ = 1, $\varphi$ = .97, with repulsion at low separations, low force at large separations, and attraction in between. The force is multiplied by a maximum step length to give the distance that the agent moves in the direction of the agent with which it is being compared. The larger the step length, the more agents move on each iteration, but the more danger there is of thrashing.

**Distributed Execution**

We apply FBC repeatedly to pairs of points. Convergence is smoothest if step length is modest, which in turn requires each pair of points to be evaluated repeatedly. Application of this algorithm to a large number $N$ of points thus involves $O(N^2)$ operations, which can be prohibitive for very large datasets.

The processes in our experiments execute asynchronously against a centralized DB. Each time a process is invoked, it

- Retrieves $C$ points and their locations from the database (in our experiments, $C$ = 35; all except queries are randomly chosen);
- Applies the FBC algorithm to all pairs of points in the sample, and computes new locations for them;
- Writes the points back into the database.

A process remembers the $k$ closest agents to a query agent that it has seen. The results of the clustering search can be retrieved by merging these lists across processes.

FBC scales linearly in both space and time in the number of processors, and offers a

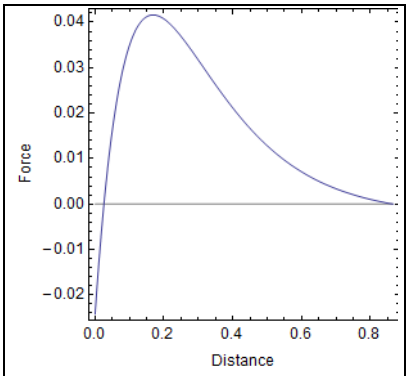


Figure 1: Example of computed force

quadratic benefit over a naïve algorithm, regardless of the degree of distribution. Let

- $N$ = the number of agents;
- $m$ = the number of processors;
- $C$ = the number of points clustered by each processor at one time;
- $d$ = the maximum step length an agent can take in a processing cycle, expressed as a fraction of the maximum distance $D$ on the torus.

Consider the quadratic benefit first. A naïve approach computes the similarity of each agent to all other agents, for complexity $O(N^2)$. In FBC, an agent is closely attracted to a group of $g$ other agents, and each interaction between two agents approximates $g^2$ interactions, reducing the complexity by a factor of $(N/g)^2$.

Massive data invites distribution. We expect time complexity to scale linearly with the number of processors. On average, assume that each agent starts out $D/2$ from its optimal location. Then an agent needs $O(1/(2d))$ interactions to reach its destination. For simplicity of analysis, assume that processes run synchronously. The probability of an agent being selected in a processing cohort is $mC/N$. Each selection yields $C$ interactions, so an agent can anticipate $mC^2/N$ interactions per processing round, thus requiring $N/(2dmc^2)$ processing rounds to move to its optimal location. Each round takes $O(C^2)$ time, so the total processing time is $N/(2dm)$, independent of $C$.

FBC processes run asynchronously, so two processes may simultaneously move the same point, and only the last one to write to the database will be preserved. The chance of a record being in a given clustering process is $C/N$. The chance that we get some record—any record—in two processes concurrently is $N*(C/N)^2 = C^2/N$. The number of possible pairs of processes is ${}_mC_2$, so the probability of collision is $\binom{m}{2} C^2/N$, which for $C$ = 35, $N$ = 350,000, $m$ = 4 is about 4%. While $C$ does not affect time complexity directly, it does affect collision probability quadratically, commending a choice of small $C$. We do not attempt to detect these collisions, but rely on the incremental any-time nature of the algorithm to correct them over time.

Space complexity is also linear, since our clustering process needs hold in memory only the set of records being clustered. Empirically, we find that 350,000 records is at the limit of a single processor, while smaller cohorts are easily processed.

**Convergence Detection**

If steps are small enough, incremental distributed processes like FBC converge roughly exponentially [13]. To monitor convergence, we compare the pairwise separation of agents on the torus with their pairwise similarity, similar to Kruskal stress [8] in multi-dimensional scaling. If we were able to capture all the similarity information in the spatial distribution, the rank ordering of distances between agents in space would be the same as their rank ordering in similarity. Sets of two pairs $(x_i, y_i)$, $(x_j, y_j)$ of joint observations from random variables $X, Y$ define four values:

- $P$ is the number of such pairs in which the rank ordering of $x$**.** and $y$**.** is the same.
- $Q$ is the number of pairs in which the rank ordering is different.
- $T$ is the number of pairs in which the $x$ values are tied.
- $U$ is the number of pairs in which the $y$ values are tied.

Given these values, the Kendall τ is

$$\tau = \frac{P - Q}{\sqrt{(P + Q + T) * (P + Q + U)}}$$

By construction, τ ranges from -1 if the variables are anti-correlated to +1 if they are perfectly correlated. We consider only pairs of points one of whose members is a query. Since retrieval in KBC consists of selecting those points that are nearest to the query, this measure accurately reflects the aspects of convergence that are important to us, ignoring dynamics far from the query that have no impact on retrieval.

## 3 Comparison with Other Technologies

The FBC algorithm merits comparison with a number of other technologies.

### 3.1 Semantic Analysis

A semantic approach to identity matching reasons explicitly about the semantics of each field. For example, it might represent the insight that if two people have the same address, they probably know one another. Such an approach is standard in classical artificial intelligence, and can bring a great deal of domain knowledge to bear on the problem. However, it is computationally very costly, and thus inappropriate for extremely large datasets. It also requires extensive knowledge engineering, slowing its application to problems that must be solved quickly.

FBC uses domain knowledge, in defining similarity metrics for complex attributes. The definitions discussed in Section 3.2 all incorporate our intuitions and semantic understanding of the problem., FBC translates these intuitions into numbers, permitting much faster computation than symbolic manipulation allows.

### 3.2 Cluster Analysis

Cluster analysis [7] seeks to identify entities that are near to one another by some measure. Centralized methods use a distance matrix of pairwise separation of entities, and many of them require updating this matrix repeatedly. Their complexity is thus at least $O(n^2)$, and in practice they reach their limit with datasets on the order of $10^5$ elements (e.g., 202,000 galaxies in [6]). Decentralized approaches [14] typically partition the data, cluster each subset separately, then exchange either cluster parameters (such as centroids) or representative points to estimate a merged clustering.

Cluster analysis differs from FBC in three important ways.

- Constructing and maintaining a distance matrix is difficult with dynamic data. Typically, one cannot add data during clustering. FBC is any-time: it can accept new data while running (though this project does not draw on this feature).
- Cluster analysis views entities as fixed in attribute space, and applies distance measurement to them as passive objects. FBC allows them to move in an abstract low-dimensional space, actively participating in their own organization.

- A consequence of the centralized distance matrix is that all attributes participate equally in global clustering decisions, hindering the analysis of heterogeneous data. FBC allows entities to interact pairwise, drawing only on those attributes that both entities possess. Integration across heterogeneous attribute sets happens by transitive interactions, in which an entity shares some attributes with one entity, other with another, and thus intermediates their interaction.

### 3.3 Dimensionality-Reduction Algorithms

The movement of FBC entities is reminiscent of some iterative-update algorithms for multi-dimensional scaling (MDS), an instance of methods that reduce the dimensionality of a set of records. In general, dimensionality-reduction algorithms do not handle data that is massive, high-dimensional, dynamic, and heterogeneous. Algorithms for dimensionality reduction of distributed sensor data [4,16] rely on the homogeneous or near-homogeneous feature spaces of sensory data. Conventional schemes for dimension reduction, such as the linear FastMap algorithm [3] or nonlinear algorithms such as IsoMap [17] or Locally Linear Embedding [15], do not handle diverse or distributed data. Some of these schemes have been adapted to a distributed environment [1,9-11], but presume homogeneous data and a non-dynamic environment that allows iteration over static data collections on each processor. They commonly make local estimates of globally relevant data items globally, and exchange these estimates iteratively across the network of processors. In high-dimensional data, not all dimensions are relevant to every interaction. Structure among subsets of the data lies on a much lower-dimensional manifold, whose dimensions depend on the query. Systems that exploit this insight [2,5] require access to all the data in a single process, and so do not support the distribution needed for timely processing of massive data.

## 4 Evaluation

We discuss preliminary results from experiments with the data set, using only the "Edge" similarity coefficient, and provide an initial assessment of the convergence characteristics of the information matching process with a small artificial data set.

### 4.1 Results

The information matching process is inherently parallel and can be distributed wih gains essentially linear in the number of processors, e.g., in a MapReduce/Hadoop cloud-computing environment. Our experiment used three standard WinTel PCs to execute 4 clustering processes each and an additional PC to run the MySQL database with the 350k records and their clustering coordinates. In this small setup, we arrived at the results reported here in less than two days execution even though one PC (4 processes) failed due to network problems after less than 8 hours.

The clustering space is a unit (1x1x1) box with all its 6 faces wrapped. Thus, we can operate in a finite volume without having to address edge-effects. The result is a

small island of records, close to the query records, and separated by a distinct "moat" from the mass of irrelevant records.

Figure 2 shows the three queries (phone, address, phone+address) in the upper right, and also records that have edges in the challenge data edge table *and* either set "NY" as their State, "Bethesda" as their city, or "212" as their area code. Adjacent to the queries, a "moat" separates a relatively small set of relevant neighbors from the rest of the data.

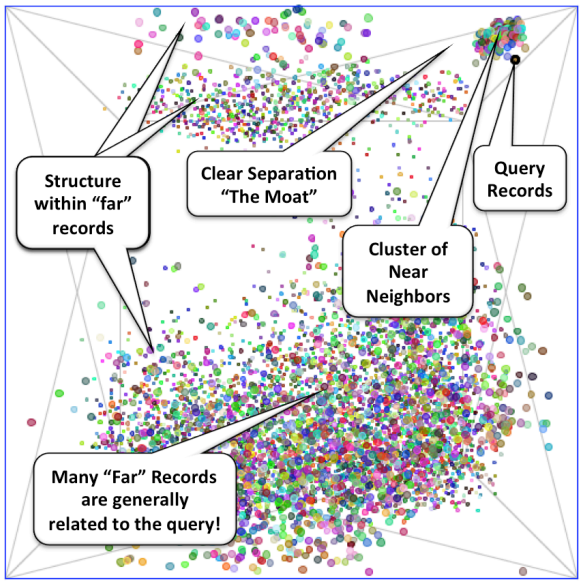


Figure 2: 7.2k Converged Records in Clustering Space

This plot illustrates two additional features of FBC.

1. All of the 7,195 records in this figure are generally relevant to the problem, yet FBC locates many of them far from the query, showing its selectivity.
2. There is considerable structure within the "irrelevant" records, suggesting that FBC can process multiple queries at the same time.

Figure 3 explores our "transitivity" claim, plotting for all pairs of (record, query) pairs the explicit edge similarity vs. their distance in clustering space. The gross structure of the arrangement shows many records far from the queries (all without direct similarity), the empty space in the mid-region, and the cluster of relevant records near the queries. In the fine structure of the query neighbors, data

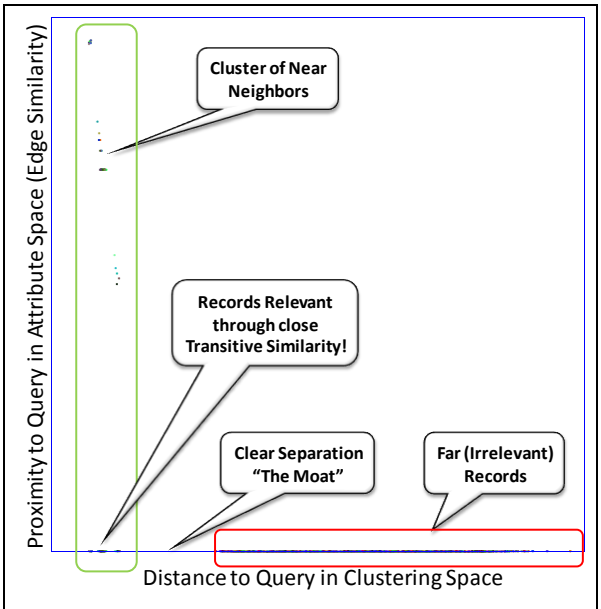


Figure 3: Correlation between Clustering Distance and Pair-Wise Edge Similarity of Data Records to Query Records

records with non-zero direct similarity are ordered in clustering distance according to their similarity. Inn confirmation of transitivity, many data records near the query records have no direct similarity to those queries, but have been pulled in by the transitive nature of FBC.

How well do we solve the initial challenge? In our interface, we select one query record, which triggers the collection of a few of the nearest neighbors to the query agents in clustering space. We order these by their cluster-space distance and estimate their match strength from their normalized (by maximum possible distance in space) distance to the selected query record. Selecting any neighbor from that list triggers an excursion into the underlying attribute-similarity space. In an exhaustive recursive process starting at the query record, we are looking for non-trivial (more than one step) transitive paths that lead to the selected data record. A data record with more than 92% similarity to the query record based on distance in clustering space and ranked #6 among all neighbors of this query, is connected to the query through only two intermediate records (note that these records themselves do not have to be close to the query in clustering space). We discover this path of records from the (incomplete phone number) query to the $6^{th}$-nearest neighbor:

- The unidentified male "Mr. X" lives in New York with his wife, Gloriana D. Brandybuck. Their land-line phone number at 3306 Rosewood Lane, New York, NY 10003 is consistent with the clinic records.
- Since they have just married, Gloriana Donnamira Brandybuck's driver's license or credit card (a Doc-ID) is still registered with her old address at 2719 Pin Oak Drive, Manhattan, NY 10018.
- They traveled to Maryland and checked them into a hotel at 18 Wayback Road, Bethesda, MD 20014, using the same Doc-ID.
- The day after their arrival, Mr. X fell ill and decided to visit the medical clinic nearby at 4408 East Madison Ave., Bethesda, MD, 20014.

The sponsor verified the correctness of our solution.

### 4.2 Convergence Assessment

We claim that FBC can be distributed over many processing units by aggressively sub-sampling the number of clustering interactions that have to be computed. We also claim that this distributed process has exponential convergence characteristics that provide a good answer fast and improved answer if more time is available (any-time characteristic). To assess these claims, we performed an initial experiment with an artificial data set of 350 color (RGB) data records, groups into seven clusters, and one query record. We start the experiment with a random arrangement of the records' agents in cluster space and run to (manually determined) convergence. We repeat the experiment varying $C$, emulating distributed execution with $C$/351 parallel processes. Our sequential execution of the random sampling ignores the possibility for collisions (the same record moved by more than one process at the same time), which will introduce additional noise.

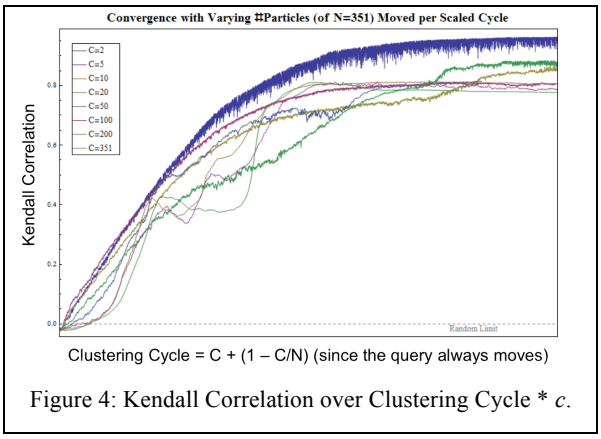


Figure 4: Kendall Correlation over Clustering Cycle * *c*.

Figure 4 assesses the impact of parallelization by scaling the x-axis for each data series by *C* and correcting for the movement of the query record. Thus scaled, the convergence curves trend very closely to each other, suggesting that a nearly linear speed-up with the number of processors may be accomplished in a distributed setting. The additional noise introduced by the sub-sampling with small *c* actually improves the final quality of the converged result. We hypothesize that, similar to simulated annealing processes for instance, the noise prevents the clustering from falling into sub-optimal stable states and instead drives it closer to the global optimum.

## 5 Conclusion

Many problems in epidemiology and domestic security require the ability to discover transitive linkages across heterogeneous databases rapidly, without reasoning explicitly about possible scenarios. Instead of reasoning about the various records, Force-Based Clustering (FBC) turns each record into a software agent that moves in an abstract information space in response to the net "force" it feels from other agents. These forces in turn are defined by generic similarity measures over commonly occurring fields, measures that can readily be defined in advance and applied quickly to available information. The agent interactions can be distributed over many processors to speed the clustering process. Application of this approach to a synthetic data set (provided by an anonymous sponsor external to our research group) allows us to identify the person of interest. Potential extensions include tuning the weights of different features, providing for human direction of the processing, distributing data as well as processing via Hadoop, and exploring dynamically changing data.

## References

[1] Abu-Khzam, F.N., Samatovaz, N., Ostrouchov, G., Langston, M.A., Geist, A.: Distributed Dimension Reduction Algorithms for Widely Dispersed Data. the Fourteenth IASTED International Conference on Parallel and Distributed Computing and Systems (IASTED PDCS 2002), pages 167-174, ACTA Press, 2002)

[2] Aggarwal, C.C., Yu, P.S.: Finding Generalized Projected Clusters In High Dimensional Spaces. SIGMOD Conference, pages 70-81, 2000)

[3] Faloutsos, C., Lin, K.-I.D.: FastMap: A Fast Algorithm for Indexing, Data-Mining and Visualization of Traditional and Multimedia Datasets. ACM SIGMOD, pages 163-174, San Jose, CA (1995)

[4] Fang, J., Li, H.: Optimal/Near-Optimal Dimensionality Reduction for Distributed Estimation in Homogeneous and Certain Inhomogeneous Scenarios. IEEE Transactions on Signal Processing, 58(8):4339-4353 (2010)

[5] Hinneburg, A., Aggarwal, C., D.A., K.: What is the nearest neighbor in high dimensional spaces? , 26th Int. Conf. on Very Large Data Bases (VLDB 2000), pages 506-515, Morgan Kaufmann, Cairo, Egypt (2000)

[6] Jang, W., Hendry, M.: Cluster analysis of massive datasets in astronomy. Statistics and Computing, 17(3):253-262 (2007)

[7] Kaufman, L., Rousseeuw, P.J.: Finding Groups in Data: An Introduction to Cluster Analysis. New York, John Wiley & Sons (1990)

[8] Kruskal, J.B.: Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. Psychometrika, 29:1-27 (1964)

[9] Magdalinos, P., Doulkeridis, C., Vazirgiannis, M.: K-Landmarks: Distributed Dimensionality Reduction for Clustering Quality Maintenance. 10th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD'06), Berlin, Germany (2006)

[10] Magdalinos, P., Doulkeridis, C., Vazirgiannis, M.: A Novel Effective Distributed Dimensionality Reduction Algorithm SIAM Feature Selection for Data Mining Workshop (SIAM-FSDM'06), Bethesda, MD (2006)

[11] Magdalinos, P., Vazirgiannis, M., Valsamou, D.: Distributed Knowledge Discovery with Non Linear Dimensionality Reduction. the 14th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD'10), Hyderabad, India (2010)

[12] NARA: The Soundex Indexing System. National Archives and Records Administration, (2007). http://www.archives.gov/research/census/soundex.html

[13] Parunak, H.V.D., Brueckner, S.A., Sauter, J.A., Matthews, R.: Global Convergence of Local Agent Behaviors. In Proceedings of Fourth International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS05), pp. 305-312, ACM (2005)

[14] Rajaraman, A., Ullman, J.D.: Mining of Massive Datasets. Cambridge, UK, Cambridge University Press (2011)

[15] Roweis, S., Saul, L.: Nonlinear dimensionality reduction by locally linear embedding. Science 290(5500):2323--2326 (2000)

[16] Roy, O., Vetterli, M.: Dimensionality Reduction for Distributed Estimation in the Infinite Dimensional Regime. IEEE Trans. Information Theory, 54(4):1655-1669 (2008 )

[17] Tenenbaum, J.B., Silva, V.d., Langford, J.C.: A Global Geometric Framework for Nonlinear Dimensionality Reduction. Science, 290(December 22):2319-2323 (2000)