simularity

The Simularity High Performance Correlation Engine

Simularity's High Performance Correlation Engine (HPCE) is a purpose-built analytics engine for similarity and correlation analytics optimized to do real-time, faceted analysis and discovery over vast amounts of data on small machines, such as laptops and commodity servers.

This paper covers an overview of the Simularity HPCE, including loading your data, the basic internal structure, how the HPCE is used, and the high level steps for an integration. It is not detailed integration, installation, or API guide.

Why Use the Simularity HPCE?

The Simularity HPCE an efficient, easy to implement, and cost-effective way to use similarity analytics across all available data, not just a sample. Similarity analytics can be used for product recommendations, marketing personalization, fraud detection, identifying factors correlated with negative outcomes, to discover unexpected correlations in broad populations, and much more.

Why similarity analytics? Similarity analytics are the best analysis tool for discovery of insights from big data. The value is in getting the data to tell you things you didn't know. This is a challenge best solved by looking for connections in the data. You just can't do this discovery with the standard analytics that come with a data warehouse. And doing this type of discovery over large datasets is cost-prohibitive with standard analytics packages.

Without similarity analytics, you have to make assumptions about the answers you might get before you even start asking questions – you have to know what you're looking for. Standard analytics are not built to look for connections, whereas that's all we do.

Technology Overview

We use a highly compact, in-memory representation of data specifically designed to do similarity analytics. We couple this with a flexible logic-programming layer that enables completely customized business rules, and a web services layer on top for easy integration with any website or browser-based tool. This unique data representation allows real-time faceting of the data, and the web services layer (API) makes including correlations in your systems, technology, or applications easy.

Our technology is a best-in-class solution for situations where reductionist approaches such as data sampling are currently used for correlation and similarity analysis because

using all the data has been cost and time prohibitive. In many cases the HPCE can do the analysis of all the data on the very servers that are currently being used to do analysis on just a sample.

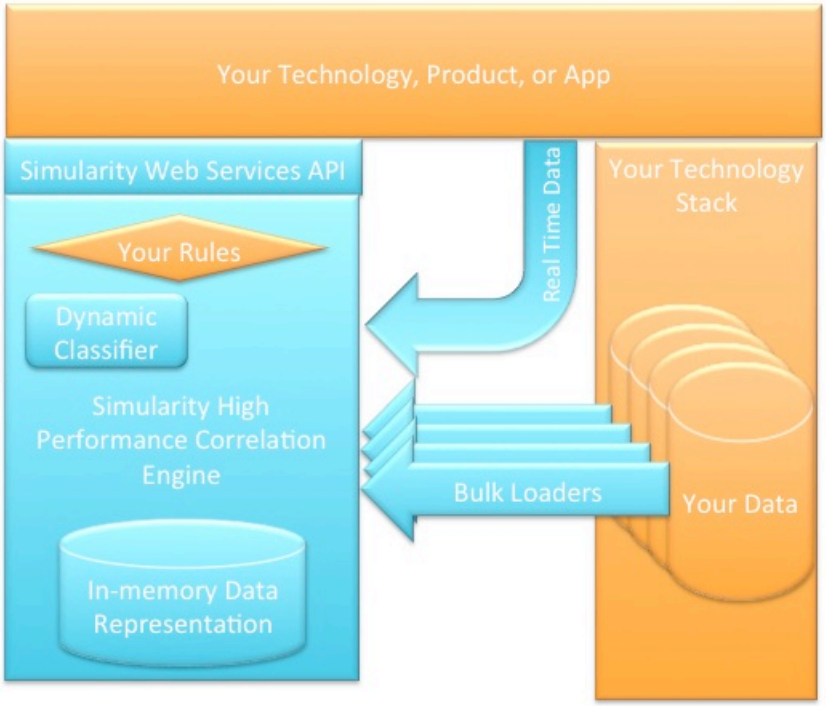

Figure 1. Platform Overview

Correlation Requests

The main way programs and systems interact and derive value from the HPCE is via Correlation Requests. A Correlation Request specifies a subset of the data to be examined or a key data element, the data type for which to calculate correlations, and the correlation metric to be used.

For example, let's say we want to find products correlated with a key product for in order to generate recommendations of the form "people who bought this also bought these other things." In this Correlation Request, the key would be the key product, the data type for which to calculate correlations is products, and we have our choice of metrics (log-likelihood works well as a metric for product recommendations). The results will be a list of products correlated with the key product, and their corresponding log-likelihood value, ordered by strength of correlation, strongest first.

In a different scenario, let's say we want to examine whether there is any seasonality to a particular event type, such as customers terminating their subscriptions to a service. In this Correlation Request, the subset of the data to be examined is the set of customers who have terminated their subscriptions, the data type for which to calculate

correlations would be the month of their termination, and again we have our choice of correlation metrics (p-value is an indication of the probability of a correlation, and works well in this case). The results will be a list of months, and their corresponding p-value, ordered by strength of correlation, strongest first.

That's the basic concept, but in fact there are a lot of ways you can fine tune correlation requests.

Faceting

Wikipedia defines faceted search thusly:

Faceted search is a technique for accessing information organized according to a faceted classification system, allowing users to explore a collection of information by applying multiple filters.

Similar to faceted search, faceting in correlations allows multiple ways to specify a subset of the data to consider. The HPCE has several mechanisms that can be used in combination to create faceted correlations: complex expressions, type subsets, and action subsets.

Complex Expressions

If you want to use a subset of data to be examined, the HPCE supports complex expressions to identify the subset. For example, to examine factors correlated with people who have been diagnosed with both diabetes and also hypertension, you'd want to use a complex expression such as this:

(people with diabetes diagnosis codes 1 or 2 or 3 or 4) and (people with hypertension diagnosis codes 5 or 6 or 7 or 8 or 9)

If you want to look at this same group but exclude people who are over 65, you could use a complex expression like this to select your subjects:

(people with diabetes diagnosis codes 1 or 2 or 3 or 4) and (people with hypertension diagnosis codes 5 or 6 or 7 or 8 or 9) and not (age greater than 65)

There are details about how to specify complex expressions in the API section of this paper.

Type Subsets

You can indicate the types of objects or subjects to be considered when determining the correlation metric. For example, if you're creating recommendations for a particular product type, i.e. a food item, you might want to specify that only products of particular types (such as other food items) be used to determine correlated products, even if people who liked this food item might also have liked movies and books.

You can also specify which types of objects that you want in the results, i.e. if you have a key product that is a health and beauty item, such as a lipstick, you might want to only get correlated items that are also health and beauty items, even if there are products

that aren't health and beauty items that were also purchased by customers who purchased this product.

Action Subsets

You may want to specify a subset of actions that are considered in determining correlations. For example, you might want to include all positive actions (such as liked, loved, bought, 4 star review, 5 star review, added to cart, added to wishlist) when creating product recommendations, and exclude negative actions (such as disliked, one or two star reviews, returned, complained). You might want to create different sets of recommendations such as "people who viewed this item also viewed" and "people who bought this time also bought." You can do this by specifying which actions you want to consider in your Correlation Request.

Simularity Triples

The data representation used by the HPCE is designed to be a general-purpose methodology for representing information, as well as an efficient model for computing correlations. Virtually any structured and semi-structured data can be represented in the Simularity Data Representation, and it can be loaded from any data source.

Data in relational databases, CSV files, NoSQL systems, HDFS, or nearly any other representation can be loaded into the Simularity Data Representation via loader programs. The loading of data happens externally to the Simularity HPCE over a socket, so users can create their own data loaders in any programming language.

The loader will take the data in its existing form (for example a relational table in an RDBMS) and turn it into triples that can be used by the HPCE. Data in the Simularity HPCE is stored in triples, which are of the form Subject/Action/Object. For example "Liz likes *Stranger in a Strange Land*" is a triple, where "Liz" is the subject, "likes" is the action, and "*Stranger in a Strange Land*" is the object.

Because the internal data representation is very compact, many data points can be simultaneously loaded into memory or cached, which helps the HPCE achieve is high performance. Although virtual memory will automatically be used when the amount of data exceeds the available RAM, and the internal data representation can be serialized to disk very quickly, for best results we recommend that enough RAM be used to have all of your data loaded simultaneously in RAM.

Typed Subjects and Objects

Simularity adds types to objects and subjects to allow faceting, so the above example might be better expressed as Customer:Liz likes Book:*Stranger In a Strange Land*. Types and actions are used to include or exclude results from a correlation request, and to change how items in the database are considered when a correlation request is executed. By using types and actions, many kinds of data can be represented, and correlations computed using many different models for selecting what is considered.

Subject types are the types associated with subjects, object types are likewise the types associated with objects; each subject and object must have a type. Subject types and

Object types are inherently different, so while the names that these types have must be different, they may use the same underlying numerical representation. Subject types distinguish between the different agents of the system.

In deciding what components of your data are subjects, remember that correlations are typically computed between Subjects or between Objects, i.e. you may compute a correlation between a book and a movie (both objects) or between two customers (both subjects). It is possible to compute a correlation between a customer and book (Subject/Object correlation), this is a different operation than the discovery of basic correlations. In general, the attributes of objects are subjects, and the attributes of subjects are objects, thus correlations can be easily computed between attributes.

Actions

Actions can also be thought of as relationships, or attributes. Examples of actions include "likes," "added to wishlist," "is a friend of," "has a". Actions are specific to an HPCE installation, and can be completely defined by the implementation. Actions have reciprocal relationships, such as "likes" and "is liked by", although both are generally referred to by the same name. Actions can be used to filter the operations which are considered when a correlation is computed, for example, when calculating product recommendations, you might want to consider all of these actions: bought, likes, loves, added to wishlist, and added to cart.

Actions may be forward, reverse, or both. A forward action is a subject acting on an object; likewise, a reverse action is an object acting on a subject. When specifying actions in a correlation request, the default is to consider both, however, it is possible to consider only a forward or reverse action.

Binning for Continuous Values

Objects in the Simularity triples are Boolean, that is they either exist or do not; they do not contain any other values. Continuous values, such as dates or ages, can be represented by binning, where the object represents a range of values. For instance, a type could exist to describe a customer subject called CUSTOMER_AGE; objects of that type would be age ranges, such as less than 12, 12-17, 18-24, 25-30, 30-39, 40-49, 50-59, and 60 or greater.

Another way to construct bins from continuous values is via mean and standard deviation. For example, if you have a Body Mass Index value for each patient in your data, i.e. 26.7, you may want to create bins to identify how far away each patient is from the average. That way you can easily do analysis on patients that are significantly above or below average. You'd do this by calculating the mean and standard deviation for this value across your data set. Then you'd create objects for the standard deviations that are positive and negative integers. Then, for each patient, add an object that indicates how many standard deviations their BMI is from the mean (rounded to an integer).

Requests can be constructed that use multiple objects, so we could get correlations corresponding, for instance, to everyone over 18. You can similarly structure time this way to associate timestamps with objects or subjects; the granularity could range from

seconds to years. For example, a timestamp representing a month/year combination would have a granularity of 1 month.

It is possible to have a bin granularity so small that each object only corresponds to one subject; such objects would not be useful for computing correlations, however, they could be used in rules to include or exclude results from a correlation request. If correlation requests are desired for timestamps, the bins must be wide enough to include multiple objects; for the most part, the more objects (or subjects) in a bin, the more effective it is for computing correlations.

The Simularity Fold

Here is a brief example of how the HPCE uses the Simularity triples to determine similarities and correlations. We call this process a "fold," in that we "fold" through the objects our subject (or set of subjects) has acted on to get the subjects that have also acted on those objects and are thus correlated. We can also fold from an object through subjects to get correlated objects. Consider the following diagram, which shows the Simularity Data Representation with subjects as purple circles, actions as lines, and objects as blue rectangles.

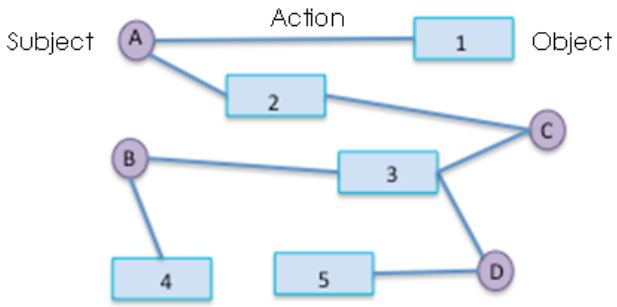

Figure 2 Example of Simularity Triples

From this diagram we can see that subject A has acted (whatever the action might be) on objects 1 and 2, subject B has acted on objects 3 and 4, etc.

To get all the subjects that are similar to (or correlated with) subject A, the HPCE gets the objects that A has acted on, 1 and 2, and then finds the subject(s) that have also acted on those objects, in this example just subject C, to which the correlation metrics will be applied.

Likewise, to find all the objects that are similar to (or correlated with) object 2, the HPCE gets all the subjects that have acted on object 2, A and C, and then finds the object(s) that they have also acted on, 1 and 3 in this case, to which the correlation metrics will be applied.

Logic Layer

The Simularity HPCE comes with a logic layer that is implemented in Prolog. It provides a framework for business rules, as well as a means of loading and maintaining those rules.

Developers have access to the logic layer and can write and load their own rules into the HPCE. This feature is entirely optional; the HPCE can be fully functional and useful without additional business rules.

Web Services Layer

The Simularity HPCE presents a RESTful web services layer to clients. Requests are represented as a URI, and responses are in JSON.

Types of API Calls

There are a number of different Correlation Requests that can be specified via web services, for example:

- Get the correlation value between two specified subjects or objects. Example: determine how similar two users are to one another.
- Get the set of *N* objects correlated with a key object, and the correlation values. Example: find the N most correlated products to a key product.
- Get the set of *N* subjects correlated with a key subject and the correlation value. Example: find the N most similar users to a key user.
- Get the set of N objects correlated with a key subject. Example: get N products that are recommended for a specific user.

Sample API Call

For this example, we're referring to the MovieLens data. We've created actions for each of the possible star ratings for movies, i.e. the "rated5" action means the user rated the movie 5 stars.

The basic web service call to the HPCE is /expression, which gets correlations to a set of items specified by an expression. It is done via an HTTP Post, where the contents of the post data define the expression. Below is a sample URL:

http://localhost:3000/expression?action=rated4&action=rated5&otype=movie&stype=user&metric=log_likelihood&legit=5&count=10&use_legit=true

with a post data of "object(movie, 260)"

This sample call would retrieve the top 10 (count=10) correlated movies (otype=movie) for movie number 260 (object(movie, 260)) using users as the inner fold (stype=user) where each result has at least 5 ratings (legit=5&use_legit=true), considering only actions that are 4 or 5 star ratings (action=rated4&action=rated5), using log_likelihood as the

correlation metric (metric=log_likelihood).

Description of Parameters

The complete parameter list for /expression is:

stype=X
This is the Subject Type to match in a fold. X is either a type number, or a symbol defining a type. Any number of stype parameters can be specified.

otype=X
This is the Object Type to match in a fold. X is either a type number or a symbol. Any number of otype parameters can be specified.

action=X
This is the Action to consider in the fold. When "action" is specified (rather than "faction" or "raction"), X is a both a forward and reverse action. X is a string or action number which specifies an action in both directions (subject to object and object to subject). Any number of action parameters can be specified.

faction=X
This is the Action to consider in the fold. X is a string or action number which specifies a forward action (subject to object). Any number of faction parameters can be specified.

raction=X
This is the Action to consider in the fold. X is a string or action number which specifies a reverse action (object to subject). Any number of raction parameters can be specified.

use_legit=Bool
This indicates whether or not to use the legit parameter. If the string is "true" then the legit parameter will be used instead of the hard limit of 10 result actions (see legit). Absence, or any value other than true means a minimum of 10 will be applied to matching result actions.

count=X
Count indicates the number of results to return and is a required parameter. No more than X results (where X is an integer) will be returned.

metric=X
Metric specifies which metric to use. X is a string. If not specified, then the default is log_likelihood.

legit=X
Legit indicates the minimum legitimate matching action count for a result to be used. This value is always enforced. If use_legit is not set to true, an additional minimum enforcement is done which requires at least ten expression results, at least 10 actions on a result, and at least 10 items in common between the 2.

Actions
At least 1 action must be specified in both directions to get results. Thus, either action=X OR both faction=X raction=Y must be specified.

The Expression
In addition to the URL parameters, an expression string must be posted via HTTP_POST.

This expression is a boolean expression comprising of terminals and operators. Parentheses may be used to enforce precedence. The operators are * (intersection) + (union) - (set difference) and / (set symmetric difference). Precedence is that of standard math operators.

Thus, in the MovieLens data where movies are objects the expression object('movie', 260) + object('movie', 1196) selects as an inner set all subjects which have a type specified by an stype parameter who had a matching reverse action on Star Wars Episode IV or Star Wars Epsiode V, counting each of them only once.

Types
The expression that gets passed into the POST contains either all subject(X, Y) terminals or all object(X, Y) terminals. Whether the expression is Subjective (subject terminals) or Objective (object terminals) determines the meaning of stype and otype. If the expression is Subjective, stype determines the type or types of results and otype determines the object types used to fold through. Conversely, if the expression is Objective, then otype determines the type or types of results and stype determines the type or types to fold through.

Note that strings can be used to specify type names and object names if they have been either dynamically specified by the loader, or specified by the appropriate declarations in the config file.

What is Legit?
We have the legit parameter so that you can exclude results that don't have enough data, whatever that might mean for your particular data set. If you have a small data set, you probably want a small value for legit, and if you have a large data set you might want a larger one. 10 is probably the highest value you'll want to use for legit. For example, in recommendations, you probably don't want to include items with only one action on them in your recommended results, since a single action is not likely to be significant enough to produce a useful recommendation.

Similarity and Correlation Metrics

The HPCE has the option of using several different similarity or correlation metrics. The metric to be used is specified in the Correlation Request. The following metrics are currently supported, but it's easy to add new metrics, so nearly any measurement of the correlation between things can be integrated with the HPCE. As in any correlation request in the HPCE, the set of types and actions to be considered can be fully specified. Metrics that are symmetric will give you the same number, regardless of the order of the items (i.e. the similarity of A to B is the same as the similarity of B to A in symmetric metrics).

Sim Scores

A Sim Score is an expression of the correlation between two objects based on how many events they have in common. It is based on the log-likelihood ratio, but is scaled to show a more linear distribution of the differences (log-likelihood values tend to be very close to 1). Sim Scores range between 0 and 1, are symmetric, and a higher number indicates a stronger similarity.

Upper P-value

The upper p-value, frequently referred to as "statistical significance", is the probability of observing a value as high or higher than the one that was actually observed, assuming that the null hypothesis is true. In this case, the null hypothesis we are testing is that there is no relationship between the observed phenomena. Upper P-values are thus an indication of how likely it is that the observed relationship between objects could occur by random chance. Very small upper p-values indicate a very strong probability of a correlation that cannot be ascribed to chance. Upper P-values range from 0 to 1 and are not symmetric. So, the p-value of A given B is not the same as the p-value of B given A.

Lower P-value

The lower p-value is the probability of observing a value as low or lower as the one that was actually observed, assuming that the null hypothesis is true. In this case, the null hypothesis we are testing is that there is no relationship between the observed phenomena. Lower P-values are thus an indication of how likely it is that the observed relationship between items could occur by random chance. Lower P-values indicate a negative correlation, that is, the presence of one object implies the absence of the other. Very small lower p-values indicate a very strong probability of a negative correlation that cannot be ascribed to chance. Lower P-values range from 0 to 1. Lower P-values are not symmetric.

Cosine Similarity

Cosine similarity is the measure of distance between two vectors. For example, in object to object similarity, the vector is derived from the number of times the two objects occur in the same event. The cosine of the angle between the vectors indicates whether the two vectors are pointing roughly in the same direction. The values range from -1 to 1. A value of -1 means they are pointing in opposite directions, and a value of 1 means they are pointing in the same direction. Cosine similarity is symmetric.

Sorensen Similarity Index

The Sorensen Similarity Index is two times the number of events where both objects occur, divided by the total of the number of events in the universe for each object. Also known as the Dice Coefficient, it is similar to the Jaccard index, but gives less weight to outliers. Values range from 0 to 1, and they are symmetric.

Jaccard Index

The Jaccard index is the cardinality of the intersection of the objects divided by the cardinality of their union. Roughly, this is the percentage of events the two objects have in common versus the total number of events for each object. The values of the Jaccard index range from 0 to 1, and they are symmetric.

Log-likelihood Ratio

The log-likelihood ratio is a metric that is derived from the rate of occurrence of the object in the events of the filter set versus the rate of occurrence of the object in the universe of events. Low values of the likelihood ratio mean that the observed result was likely to occur under the null hypothesis (by random chance) indicating a lack of a relationship between the objects. High values of the log-likelihood ratio indicate that the observed outcome was unlikely to occur under the null hypothesis (random chance) and therefore indicates a likely correlation between the objects. The log-likelihood ratio is proportional to the 2 log lambda of the binomial likelihood ratio. This value is scaled so as to range from 0 to 1, and orders items in the same order as the 2 log lambda statistic. Log-likelihood is symmetric.

Sample Use Case: Simularity HPCE as a Recommendation Engine

The Simularity HPCE can be used as an effective product recommendation system. Products can be recommended by either Object/Object recommendations (People who bought/viewed etc. this thing also bought these things) or Subject/Object recommendations (people similar to you bought these things). Subject/Subject and even Object/Subject recommendations are possible, although less useful in most circumstances.

Recommendations, like all correlations in Simularity, may be faceted, that is, they may consider only a subset of the types and actions in the Simularity Data Representation. For example, a recommendation system that had types of books, movies, and video games, and actions of viewed, wish-listed, and purchased could be structured in such a way that a correlation request will find all of the purchased video games correlated with views of Robert Heinlein's book *Starship Troopers*. In this example, the only data that is considered is views of *Starship Troopers* and purchases of video games (there is also a consideration of the type of subjects that do the viewing and purchasing – in the case of most recommendation systems, there is only one kind of customer).

In addition, user specified rules in the logic layer can be implemented which limit the items that can be recommended according to arbitrarily complex business rules.

Anatomy of an Integration

Using the Simularity HPCE is easy. You can install the software on your machines, our use our hosted service. You own the data, and you are in control of the access to your instance of the engine. Here are the typical steps of an integration with the Simularity HPCE.

Step 1: Data Mapping

The first step is to analyze how you want your data to be represented as triples in the HPCE. This means separating the data into subjects and objects, and separating those subjects and objects into appropriate types.

Sometimes this partitioning is obvious: in-store and internet customers are subjects of two types and products are objects, with varying types. The partitioning may not always be this obvious, however.

If you want to include an internet user's zip code, that zip code would be an object (it's an attribute of a subject, probably of type ZIP_CODE). If you wanted to include the warehouses where products are located, the warehouse would be a subject (it's an attribute of an object). You could do a subject/subject correlation request then between an internet user and warehouses, to find the warehouses most correlated to (used by) a particular user, or, perhaps more interestingly, the other way.

Step 2: Data Loading

Batch Load

The data in the Simularity HPCE comes from an external loader program. The loader reads from some data store, such as a relational database, text files, or any other data source, and transforms it into the Simularity triples. These triples are then added to the Simularity HPCE by calling a web service, or by connecting to a TCP socket.

Default loaders are provided with the HPCE. It should be noted that any programming language can be used to write a loader; if specialized libraries are required to read a data source, the programming language need only be able to write to a socket in order to load the data – no specialized Simularity library is required.

Streaming Load

Once operational, data may be added to the Simularity HPCE at any time. Simply connect to the HPCE's loader socket or call the web service and new data can be written (or deleted) in real time. The data can be updated continuously without interfering with ongoing correlation requests. Each new correlation request will use the latest data.

Step 3: Business Rules

Determine which, if any, business rules you'd like to apply to filter the results of your correlation requests. There may be arbitrarily many rules, and these rules act as filters or modifiers to correlation results that have already been determined.

These business rules could include results that should be excluded, for example perhaps you don't want your recommendations to include self-help books or text books when providing personalized recommendations for a user. These rules might include an optional set of strategies for filling out result sets when there aren't enough correlated items, such as using best sellers in the same genre as the key item.

Step 4: Results

The Simularity HPCE can provide results in one of two ways: dynamically, as part of a response to a web service request (JSON), or in batch operation, where data is output to a CSV file.

Batch operation is typically run over a large set of the data, which is then processed by rules, one of which specifies how to output the data. In batch operation, correlations can be generated for some or all of the objects, subjects or both in the system, and stored in a file for loading into a relational database, spreadsheet, or other means of processing.

Dynamic results are returned in real time, via the web services layer, and are represented in JSON. Using the web services layer, results can be incorporated into any website.