Cross-correlation based clustering and dimension reduction of multivariate time series

Attila Egri*, Illés Horváth†, Ferenc Kovács‡, Roland Molontay§ and Krisztián Varga‡
*Budapest University of Technology and Economics
Department of Differential Equations
†MTA-BME Information Systems Research Group, Hungary
‡Nokia, Bell Labs
§Budapest University of Technology and Economics
Department of Stochastics

***Abstract*—In this paper, we investigate dimension reduction possibilities of multidimensional time series data and we introduce a graph based clustering approach using the cross-correlation between time series. The proposed solution consists of two main steps: introducing a novel similarity measure for measuring cross-correlations and a graph-based clustering technique. These two parts are both compared to existing techniques, including noise tolerance and our solution performs better in a noisy environment. The proposed solution is applied to performance metrics of a specific data processing system in order to identify and efficiently visualize connections among the collected metrics. The introduced method provides a more balanced clustering than classic ones, and it is suitable to reveal dependencies and connections among performance metrics time series data.**

I. Introduction

Application fingerprinting with respect to resource consumption is a methodology to characterize the behavior and performance of a software system. The main goal of the technique is to capture the timely performance characteristics of a software system.

The basis of a performance modeling is to collect performance metrics of a system. These metrics can describe not only available hardware resources such as CPU usage, available physical memory or transmitted bytes but also application level resources such as queue lengths, number of received messages etc. These metrics are sampled at specific time intervals to obtain a set of time series data. Even a small distributed system can provide hundreds of metrics therefore to identify the relevant ones are important.

There are two main techniques to reduce the complexity of high dimensional multivariate time series:

- *dimension reduction:* determining the most important attributes,
- *clustering:* grouping the similar attributes together.

These topics are highly investigated and several paper were published during the past years e.g., [7], [12], [27], [22], [21], [10], [32], just to mention a few. The similarity or dissimilarity measure is basis of the clustering and various methods exist to define them [7], [12] and [27].

The main objective of this work is to identify connections among various performance metric and to identify central elements of the found clusters in order to reduce the number of performance metrics to track. We introduce a novel similarity measure of time-series based on cross-correlations and associate a weighted graph to the similarity matrix. Time series clustering based on this similarity measures produce more stable and reliable clustering results even in a noisy environment. This noise tolerance is investigated with different noise model and is introduced in Section IV.

Although the introduced solution is evaluated by monitoring and optimizing the necessary performance metrics of a stream processing application, which calculates high level network KPIs, the described method can be applicable in other scenarios as our method keeps background assumptions as minimal as possible.

The rest of the paper is organized as follows. First, an overview of existing dissimilarity measures and dimension reduction methods are given in Section II. Then we propose a cross-correlation based dissimilarity measure and clustering in Section III. Evaluation of the proposed methods and comparison to some traditional techniques is provided in Section IV. Finally, Section V concludes the work.

II. Overview of clustering and dimension reduction for multivariate time series

Data in the form of time series infuses several scientific fields, including medicine, finance, economics, hydrology, engineering and the analysis of time series has become a well-established topic [13], [16]. There has been an increased interest recently in multidimensional time series [25], [29] as the result of the growing number of available multidimensional time series data due to the development of data collection technology [30].

A time series is a series of successive measurements, $x_i(t)$; $[i = 1, \ldots, d;\ t = 1, \ldots n]$ made sequentially over a time interval where i indexes the variables observed at each time point t. A time series is univariate if $d = 1$ and multivariate if $d \geq 2$. A natural representation of a multivariate time series is a data matrix $A \in \mathbb{R}^{n \times d}$, where n (the number of rows) denotes the cardinality of the index set, i.e., the number of time stamps and d (the number of columns) refers to the number of variables (also known as attributes, features or sensors),

i.e., the dimension. Each column $A_{.,i}$ of the matrix can be considered as a one-dimensional time series.

Some key issues when dealing with multidimensional (occasionally extremely high dimensional) time series are determining important attributes (dimension reduction)and grouping the similar features together (clustering). As for the latter, the usual way of determining similar attributes of a multidimensional time series is using distance based clustering approaches. The data points are one dimensional time series objects and the task is to perform time series clustering. Undoubtedly, the dissimilarity measure (or distance) is the most essential ingredient of time series clustering. Throughout the years, several time series similarity measures have been proposed; for extensive reviews, comparative studies with rigorous evaluation we refer the reader to [7], [12] and [27]. From among the many available (dis-)similarity measures, we specifically mention auto-correlation function (ACF) and Euclidean distance (EUCL) [4].

After choosing the dissimilarity measure any properly chosen conventional clustering algorithm can be employed; an overview of various general-purpose clustering procedures that have been used in recent time series clustering studies can be found in [18]. A time series oriented approach is k-Shape a robust partional clustering algorithm that preserves the shapes of time series [36].

Reducing the dimensonality of a high-dimensional time series is an important issue from many respects: relieving the curse of dimensionality, interpreting the variability, reducing the computational and storage capacity by monitoring only a few common factors/subset of features. There are two general approaches to address dimensionality reduction: feature extraction and feature selection. Feature extraction refers to mapping the existing features into a lower dimensional space while feature selection means identifying a subset of original features without a transformation.

There are several standard tools available to perform feature extraction on multivariate time series, including methods based on canonical correlation analysis [6], factor modeling [24], independent component analysis [5], principal component analysis [28]; a literature review can be found in [14]. Several more recent works have proposed improved methods such as distance covariance based independent component analysis [22], dynamic orthogonal components [21], time series central subspace analysis [23].

Feature selection techniques for multivariate time series have also received a lot of research interest recently. Feature selection methods are usually preferred by domain experts since the connection of the selected subset of features to the originally acquired ones is preserved. The widely used methods include techniques based on common principal components (CLeVer) [33], recursive feature elimination and support vector machine [34], correlation analysis [32], mutual information and class separability [10]. A comparative study on the applicability of a range of feature selection methods for domain specific data can be found in [9].

The main motivation behind the proposed method presented in Section III is to provide a relatively balanced clustering with keeping background assumptions as minimal as possible. Other existing time series models can be fitted to the data, but they require certain assumptions that may or may not be valid, and various models may exhibit certain types of behaviour. Our only assumption for the proposed method is weak (wide sense) stationarity of the time series, which allows for correlation and cross-correlation based calculations. Apart from this, no background model is fitted. Weak stationarity is an essential prerequisite for meaningful performance evaluation of the system [35].

III. Description of the proposed method

A. Cross-correlation based similarity measure

We are given a multi-dimensional time series data set, represented by a data matrix $A \in \mathbb{R}^{n\times d}$. Each column $A_{.,i}$ of the matrix can be considered as a one-dimensional time series. First, we calculate the sample cross-correlation function of each pair $(A_{.,i}, A_{.,j})$, $1 \le i, j \le d$.

Strictly speaking, the cross-correlation of a pair of jointly wide sense stationary (second order stationarity, see Section 2.4 in [8]) stochastic processes $(X(t), Y(t))$ is given by

$$\rho_{X,Y}(\tau) := \frac{\mathbb{E}\left[(X(t)-\mu_X)\,(Y(t+\tau)-\mu_Y)\right]}{\sigma_X \sigma_Y},$$

where μ_X, μ_Y and σ_X, σ_Y are the mean and standard deviation of the processes $(X(t))$, $(Y(t))$ respectively. (The cross-correlation is independent of t due to the requirement that $(X(t), Y(t))$ are jointly wide-sense stationary.) In particular, the sample cross-correlations of two times series can be estimated by averaging the product of samples measured from one process and samples measured from the other [31]:

$$\rho_{X,Y}(\tau) = \frac{\sum_{t=1}^{n}(X(t)-\overline{X})(Y(t+\tau)-\overline{Y})}{\sqrt{\sum_{t=1}^{n}(X(t)-\overline{X})^2}\sqrt{\sum_{t=1}^{n}(Y(t+\tau)-\overline{Y})^2}},$$

where $\overline{X} = \sum_{t=1}^{n} X(t)/n$ and $\overline{Y} = \sum_{t=1}^{n} Y(t)/n$.

After calculating the sample cross-correlation function of the columns (i.e., determining $\rho_{A_{.,i},A_{.,j}}(\tau)$ for all $1 \le i, j \le d$ and for all possible values of $\tau = 0, \pm 1, \pm 2, \dots$) we define the similarity of two attributes as the maximum of the absolute values of the sample cross-correlations of the corresponding one-dimensional time series. More precisely, the similarity of ith and jth attribute is defined as:

$$\mathrm{sim}(A_{.,i}, A_{.,j}) := \max_{\tau} \left|\rho_{A_{.,i},A_{.,j}}(\tau)\right|.$$

We note that this similarity measure is a symmetric measure, thus $\mathrm{sim}(A_{.,i}, A_{.,j}) = \mathrm{sim}(A_{.,j}, A_{.,i})$ and its values ranges from 0 (no relation) to 1 (strong relation). We also note that in practice, there is a limit on parameter τ; in the present study, we use $|\tau| \le 20$ (which corresponds to 40 seconds in this case). The value was settled upon after investigation of several cross-correlation functions; at that point, all relevant correlations are already captured.

B. Graph based clustering method

For any (dis)similarity matrix A, a weighted undirected graph G can be constructed: every vertex $i \in V(G)$ corresponds to an attribute (i.e. the $A_{.,i}$ column of matrix A) and two vertices i and j are connected with an edge $e_{i,j} \in E(G)$ of weight $w(e_{i,j}) := \text{sim}(A_{.,i}, A_{.,j})$. In order to uncover the connections in the multi-dimensional time series A, in other words, to find attributes that are similar (i.e., the clusters) we have to reveal the community structure of the graph G. There are several methods to do so; one of the easiest is to construct a graph G' on the same vertex set $V(G) = V(G')$ but we only keep the edges with weight above a certain threshold δ (e.g. we remove an $e_{i,j}$ if $w(e_{i,j}) \leq \delta$, then we can identify the communities (or clusters) as the connected components of G' (similarly to the concept of clique cluster from [19]). Clearly, the number of clusters obtained is monotone increasing in δ; in practice, the value of δ offers great scalability in how strong connections we intend to measure. Typically, $\delta > 0.7$ shows a very strong connection.

There are several other sophisticated algorithms for community detection (or graph partition) of weighted graphs, for a survey we refer the reader to [11].

C. Dimension reduction through finding "central" nodes

Next, central nodes are selected within each cluster according to the sum of the weight of the edges from each node, taking the node with maximal value as the center. This centrality measure corresponds to the weighted degree of vertices in G', i.e., only edges with weight above a certain threshold are taken into consideration. Dimension reduction is then carried out by selecting cluster centers from among all attributes, thus the reduced number of attributes is equal to the number of clusters.

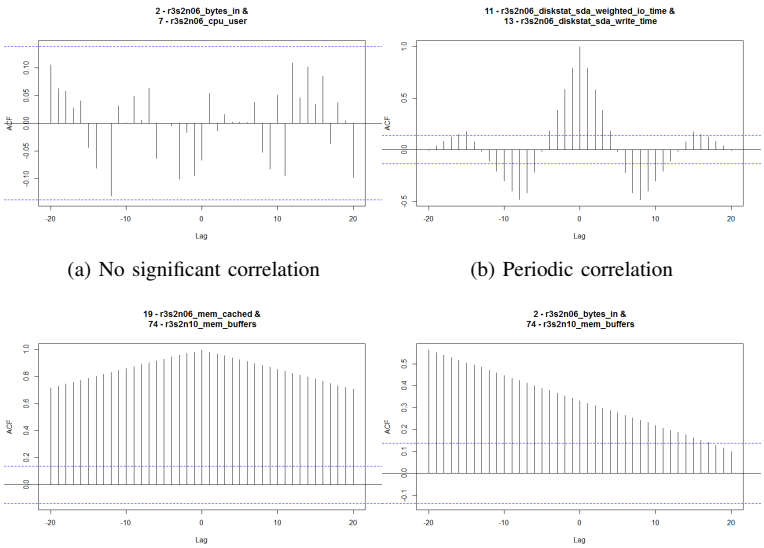
IV. Validation of the proposed method

In this section we evaluate the proposed methods using performance metrics from a stream processing application. First the experimentation environment is described than some motivation for cross-correlation based similarity metrics is provided. Then the validation is accomplished in two steps: first, we provide a sensitivity analysis for the dissimilarity measures, then the result of graph based clustering are evaluated.

A. Experimentation environment

The basis of the experimentation is a storm-based data processing system, which calculates higher level network performance KPIs (key performance indicators). The investigated system is a simplified version of a real one and works as follows:

- The network elements (e.g. base stations) send compressed XML status reports periodically (e.g. every five minutes). The network elements send their reports at different points in time to avoid overloading the data processing system.



(a) No significant correlation

(b) Periodic correlation

(c) Highly correlated variables

(d) Decreasing correlation

Fig. 1: Various cross-correlation profiles

- The data processing system receives the reports and calculates KPIs. During this experimentation the calculation was done by a four stage pipeline:
  1) *Parser:* It parses the XML documents and retrieves the measurements from them.
  2) *Enrich:* It extends the measurements with additional information, coming from different sources.
  3) *Filter:* It is possible to filter out measurements from the stream that allows to refresh only the selected part of the KPIs.
  4) *Aggregate:* It calculates the new value of the KPIs.

The experimentation took place in our lab environment, with status reports stored in an HDFS [2] storage and played back with real traffic timing or accelerated playback. Monitoring was done using the Ganglia monitoring system [1]. The monitoring assist server gathers second-precision status data from the other computers in real time. Mainly CPU, memory, disk and network usage is being monitored on all nodes among some other metrics.

B. Cross-correlation profiles

We include a few different cross-correlations between certain pairs of performance metrics just to illustrate the very different types of behavior the cross-correlations may exhibit. In Figure 1, lag between the two performance metrics is shown on the x axis, and cross-correlation with the given lag is displayed on the y axis. Note that the various behaviors may be due to very distinct computer engineering reasons. For example, periodicity may be due to read and write operations alternating on the file server. In the present study, we only want to identify high correlation between performance metrics with any lag without giving a sophisticated background model of the system.

C. Sensitivity analysis of similarity measures

Here we compare the cross-correlation based similarity measure (CRC) against some established (dis)similarity measures such as auto-correlation function (ACF), conditionally independent dyads (CID), partial auto-correlation function (PACF), periodogram based dissimilarity (PER), correlation-based dissimilarity (COR) and euclidean distance (EUCL); using the TSclust package for R [4]. The performance of the various methods is difficult to compare directly; we conduct a sensitivity analysis of time series similarity measures adopted from [17].

For cross-correlation, the lag was considered up to $|\tau| \leq 20$, which corresponds to 40 seconds.

Let k denote the number of objects (performance metrics in our case) represented by a time series. Let $D_* = (a_{i,j,*})$ denote the (dis)similarity matrix of size $k \times k$ where $*$ refers to the type of the measure. Now we perturbate the ith object according to a certain effect that determines the different aspects of sensitivity detailed in the next paragraph. Let $P_*^{**} = p_{i,j,*}^{**}$ be a matrix of size $k \times k$, where $**$ stands for the perturbation type and $*$ for the measure, and an element of P: $p_{i,j}$ the distance between the perturbated object i and unaffected object j. We consider the average effect of perturbation $**$ as an averaged ratio of the distances:

$$p_*^{**} = \sum_{\substack{i,j=1 \\ i \neq j}}^{k} \frac{1}{k(k-1)} \frac{p_{i,j,*}^{**}}{a_{i,j,*}}. \tag{1}$$

Since the perturbation types have stochastic components, the p_*^{**} values should be averaged for more simulations.

The different perturbation types are

- Amplitude scaling: $x^{\text{AS}}(t) = x(t) \cdot B$ where $B \sim \mathcal{N}(1, \sigma^2)$.
- Amplitude translation: $x^{\text{AT}}(t) = x(t) + B$ where $B \sim \mathcal{N}(0, \sigma^2)$.

The amount of variability introduced in the perturbation is driven by the σ parameters. We conducted experiments with a range of levels of introduced variability to understand the effect of levels on performance. For both types of perturbations, the range is $\sigma = 0.04, 0.08, \ldots, 0.4$.

Figures 2–3 display the results of the sensitivity analysis for the different perturbation types. The figures were obtained from the average of several perturbations and averaged out among all components of the pipeline.

The most information comes from Figure 2; for amplitude scaling perturbations, CRC is less sensitive than CID, PER and EUCL while more sensitive than ACF, COR or PACF. Figure 3 shows that CID and EUCL are more sensitive than the others.

D. Validation of the graph based clustering method

The graph based clustering approach (GBC) is compared to two algorithms that are commonly used in time series clustering studies: partitioning around medoids (PAM) that is a realization of k-medoids algorithm [15], and a p-value based

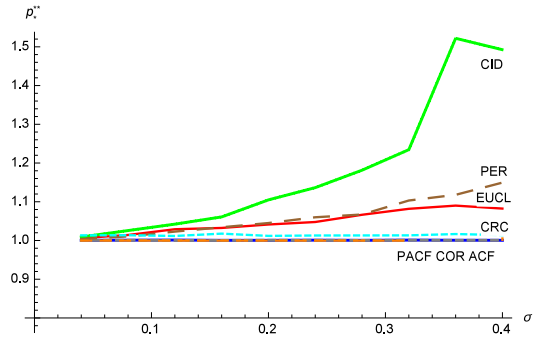


Fig. 2: Sensitivity for amplitude scaling perturbations

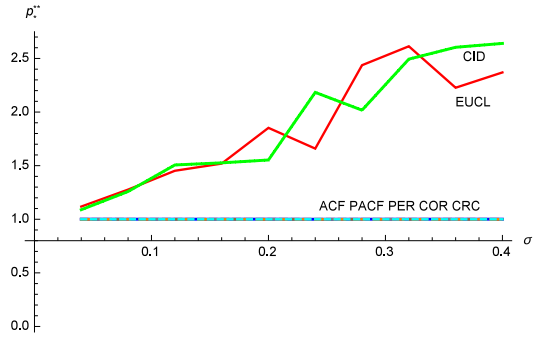


Fig. 3: Sensitivity for amplitude translation perturbations

clustering algorithm (PVAL). The p-value of a pair of time series (i, j) corresponds to the p-value obtained from checking whether the i-th and j-th series come from the same generating model; the cluster solution is formed by series with associated p-values greater than a pre-specified level of significance, for more details see [20].

Many of these clustering methods can be parameterized in order to either produce a given number of clusters directly, or the number of clusters produced can be changed through other parameters indirectly. We parameterized all methods to give roughly the same number of clusters. Apart from the number of clusters, we are also interested in providing a relatively balanced and robust clustering, with no giant cluster but a relatively high number of medium-sized clusters. The main motivation for this is that neither clusters of size 1 nor huge clusters carry too much information about the connections between the various performance metrics.

Comparison is done regarding the number and size of the clusters for each combination of the GBC/PAM/PVAL clustering methods using any of the similarity measures ACF/EUCL/CRC. Table I includes a summary of the total number of clusters, the number of clusters of size at least 4, and the size of the largest cluster for each combination of the ACL, EUCL, CRC similarity measures and the GBC, PAM and PVAL clustering methods.

Figure 4 displays a CRC/GBC cluster with the red, respectively, blue edges corresponding to the correlation between the performance metrics in the aggregate, respectively, filter component. It should be noted that the correlation values are

	GBC	PAM	PVAL
ACF	59/4/28	60/4/18	55/4/25
EUCL	62/2/47	60/3/36	59/2/35
CRC	58/7/11	60/5/8	64/2/4

TABLE I: No. of clusters / no. of clusters of size ≥ 4 / size of largest cluster

almost identical for the two processing units. This cluster means that these 8 performance metrics are redundant, and it would make sense to measure and log only 1 out of the 8.

Next, we present a confusion matrix between GBC and either PAM or PVAL in order to compare the actual clusterings. For both GBC/PAM and GBC/PVAL, we count the number of pairs of performance metrics that are

- in the same cluster for both methods,
- in the same cluster for the first method but not the second,
- in the same cluster for the second method but not the first,
- in different clusters for both methods.

In general, two clusterings are similar if the diagonal elements are high compared to off-diagonal elements.

Confusion matrices in Table II and III are calculated for EUCL, ACF and CRC similarity measures. A confusion matrix compares two setups of clustering method plus dissimilarity measure by counting the number of pairs of performance metrics that ended up in the same cluster or different clusters with either setup. Note that the sum of the elements in each 2×2 confusion matrix is 115^2 (the total number of pairs of performance metrics). In general, high diagonal values mean the two setups provide similar clustering.

EUCL	diff.	same
diff.	1485	14
same	820	10906

ACF	diff.	same
diff.	583	6
same	402	12234

CRC	diff.	same
diff.	367	4
same	102	12752

TABLE II: PAM vs GBC confusion matrices

EUCL	diff.	same
diff.	115	1310
same	2190	9610

ACF	diff.	same
diff.	115	994
same	870	11246

CRC	diff.	same
diff.	117	128
same	352	12628

TABLE III: PVAL vs GBC confusion matrices

E. Central nodes

Just as an illustration, we include the cluster centers of clusters of size at least 4 according to CRC/GBC clustering (our proposed process). Further details about the actual clustering are omitted due to lack of space.

- `r3s2n06_cpu_sintr` - a CPU metric related to exception handling
- `r3s2n06_mem_cached` - corresponds to the number of cached files (before sending them through the network) in the file server
- `r3s2n06_multicpu_steal1` - corresponds to idle time of CPU core 1 on the file server (due to waiting for the file server)
- `r3s2n10_pkts_out` - corresponds to the amount of packets sent through the network interface by the test server – including overhead
- `r3s2n10_cpu_idle` - corresponds to idle time percentage of the CPU of the test server
- `r3s2n10_multicpu_steal1` - corresponds to idle time of CPU core 1 on the test server (due to reading files from the hard drive)
- `r3s2n10_tx_pkts_eth0` - corresponds to the amount of packets sent by the test server

We note that the clusters generally correspond to memory usage, CPU usage and I/O load of the file server and the test server. One should keep in mind that large clusters typically have several performance metrics "close" to the center, which may be just as suitable for cluster center.

By using only the central nodes, it is possible to log only 60 of the 280 performance metrics in total, which is a considerable reduction in numbers.

V. Conclusions and outlook

We have presented a novel method to automatically find connections between variables in multivariate time series. The proposed approach consists of two major phases: a novel similarity measure (for measuring cross-correlations) and a graph based clustering algorithm (using the correlations). It was applied to performance metrics in a stream processing system. We also compared our findings to the results of other (dis)similarity measures and techniques of multivariate time series clustering. The noise tolerance of the provided methods is also investigated and it performs better then the traditional ones.

The method may be refined in certain ways. One possible generalization of the graph based clustering is to consider the k-edge-connected components in order to identify stronger connections or using community detection techniques. Also, in the application, many of the clusters contained only a single performance metric. Identifying the importance of such performance metrics and modeling them with classic clustering methods is difficult. A better model that naturally includes singletons may be a so-called hybrid model, where large clusters and singletons are treated separately. Such a model may be versatile enough to warrant further research.

Acknowledgment

The research of R. Molontay was partially supported by MTA-BME Stochastics Research Group and by NKFI K10475 research grant.

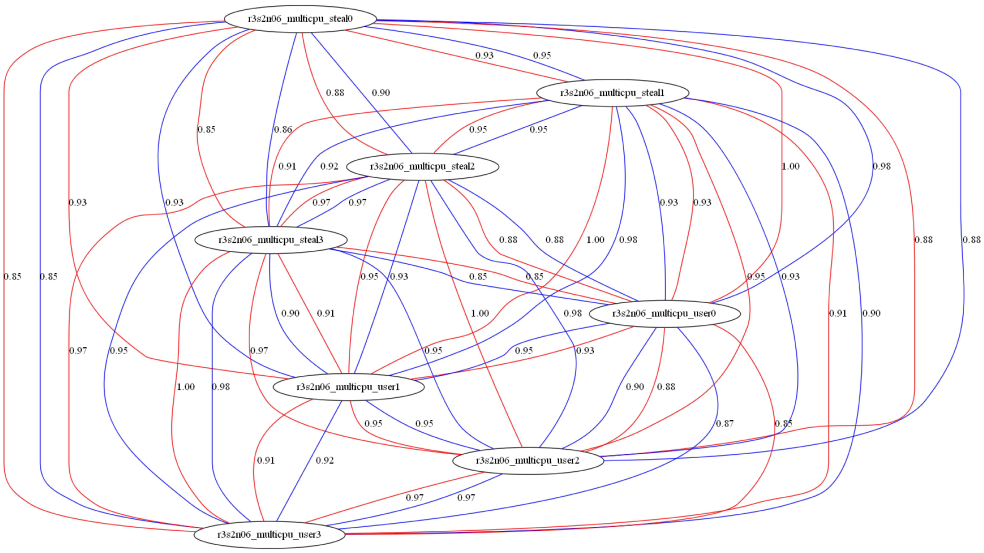


Fig. 4: Cluster with aggregate and filter connections

REFERENCES

[1] Ganglia monitoring system. `http://ganglia.sourceforge.net/` Accessed: 2017-06-10.

[2] Hadoop distributed file system. `https://hadoop.apache.org/docs/r1.2.1/hdfs_design.html` Accessed: 2017-06-10.

[3] NbClust package for R: Determining the Best Number of Clusters in a Data Set. `https://cran.r-project.org/web/packages/NbClust/` Accessed: 2017-06-10.

[4] TSclust package for R: Time Series Clustering Utilities. `https://cran.r-project.org/web/packages/TSclust/` Accessed: 2017-06-10.

[5] A. D. Back and A. S. Weigend. A first application of independent component analysis to extracting structure from stock returns. *International journal of neural systems*, 8(04):473–484, 1997.

[6] G. E. Box and G. C. Tiao. A canonical analysis of multiple time series. *Biometrika*, 64(2):355–365, 1977.

[7] K. Buza, A. Nanopoulos, and L. Schmidt-Thieme. Fusion of similarity measures for time series classification. In *Hybrid Artificial Intelligent Systems*, pages 253–261. Springer, 2011.

[8] C. Chatfield. *Time-series forecasting*. CRC Press, 2000.

[9] S. Cheema, T. Henne, U. Koeckemann, and E. Prassler. Applicability of feature selection on multivariate time series data for robotic discovery. In *Advanced Computer Theory and Engineering (ICACTE), 2010 3rd International Conference on*, volume 2, pages V2–592. IEEE, 2010.

[10] L. Fang, H. Zhao, P. Wang, M. Yu, J. Yan, W. Cheng, and P. Chen. Feature selection method based on mutual information and class separability for dimension reduction in multidimensional time series for clinical data. *Biomedical Signal Processing and Control*, 21:82–89, 2015.

[11] S. Fortunato. Community detection in graphs. *Physics reports*, 486(3):75–174, 2010.

[12] R. Giusti, G. E. Batista, et al. An empirical comparison of dissimilarity measures for time series classification. In *Intelligent Systems (BRACIS), 2013 Brazilian Conference on*, pages 82–88. IEEE, 2013.

[13] D. Hamilton. *Time Series Analysis*. Princeton University Press, 1994.

[14] X. Huang. Topics in multivariate time series analysis: Statistical control, dimension reduction visualization and their business applications. 2010.

[15] L. Kaufman, P. J. Rousseeuw. *Finding groups in data: an introduction to cluster analysis.* John Wiley & Sons, 2009.

[16] M. Last, A. Kandel, and H. Bunke. *Data mining in time series databases*, volume 57. World scientific, 2004.

[17] S. Lhermitte, J. Verbesselt, W. W. Verstraeten, P. Coppin. A comparison of time series similarity measures for classification and change detection of ecosystem dynamics. *Remote Sensing of Environment*, Vol. 115, 12:3129–3152, 2011.

[18] T. W. Liao. Clustering of time series data - a survey. *Pattern recognition*, 38(11):1857–1874, 2005.

[19] J. Liu, Q. Zhang, W. Wang, L. McMillan, and J. Prins. Clustering pairwise dissimilarity data into partially ordered sets. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 637–642. ACM, 2006.

[20] E. A. Maharaj. Cluster of time series. In *Journal of Classification*, 17(2):297–314, 2000.

[21] D. S. Matteson and R. S. Tsay. Dynamic orthogonal components for multivariate time series. *Journal of the American Statistical Association*, 2012.

[22] D. S. Matteson and R. S. Tsay. Independent component analysis via distance covariance. *arXiv preprint arXiv:1306.4911*, 2013.

[23] J.-H. Park, T. Sriram, and X. Yin. Dimension reduction in time series. *Statistica Sinica*, pages 747–770, 2010.

[24] D. Pena and G. E. Box. Identifying a simplifying structure in time series. *Journal of the American statistical Association*, 82(399):836–843, 1987.

[25] G. C. Reinsel. *Elements of multivariate time series analysis*. Springer Science & Business Media, 2003.

[26] P. J. Rousseeuw. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, 20:53–65, 1987, Elsevier

[27] J. Serrí and J. L. Arcos. An empirical evaluation of similarity measures for time series classification. *Know.-Based Syst.*, 67:305–314, 2014.

[28] J. H. Stock and M. W. Watson. Forecasting using principal components from a large number of predictors. *Journal of the American statistical association*, 97(460):1167–1179, 2002.

[29] R. S. Tsay. *Multivariate Time Series Analysis: With R and Financial Applications*. John Wiley & Sons, 2013.

[30] V. Turner, J. F. Gantz, D. Reinsel, and S. Minton. The digital universe of opportunities: Rich data and the increasing value of the internet of things. *DC Analyze the Future*, 2014.

[31] W. N. Venables and B. D. Ripley. *Modern applied statistics with S-PLUS*. Springer Science & Business Media, 2013.

[32] Q. G. Wang, X. Li, and Q. Qin. Feature selection for time series modeling. *Journal of Intelligent Learning Systems and Applications*, 5(03), 2013.

[33] K. Yang, H. Yoon, and C. Shahabi. Clever: A feature subset selection technique for multivariate time series. In *Advances in Knowledge Discovery and Data Mining*, pages 516–522. Springer, 2005.

[34] H. Yoon and C. Shahabi. Feature subset selection on multivariate time series with extremely large spatial features. In *Data Mining Workshops, 2006. ICDM Workshops 2006. Sixth IEEE International Conference on*, pages 337–342. IEEE, 2006.

[35] D.G. Feitelson *Workload modeling for computer systems performance evaluation*. Cambridge University Press, 2015.

[36] J. Paparrizos, and L. Gravano k-shape: Efficient and accurate clustering of time series. In *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*, pages 1855–1870. ACM, 2015.