Fast Consensus Clustering via local optimization with dynamic similarity updates

Thomas Glassen
Universität der Bundeswehr München

November 11, 2019

Address correspondence regarding this manuscript to Thomas Glassen (thomas.glassen@ymail.com).

Abstract

The selection of the right method for Consensus Clustering can be a tricky task because it often requires a compromise between speed and precision. For example a group of procedures represent the so-called Local Search (LS) methods, which gradually optimize a consensus candidate until no further improvement is gained. This group of methods often achieve a more accurate consensus than other methods, but nearly always have a quadratic factor of the partition size N in their runtime complexities. Because this is widely considered as impracticable for realistic datasets, I present here a general strategy to accelerate those procedures significantly. Concretely I will show how to build LS algorithms that use the Adjusted Rand Index, the Jaccard Index, the Mirkin Metric, the Normalized Mutual Information, the Variation of Information or other similarity / distance measures to achieve a per cycle runtime complexity of $O(KNC)$ each. There are no heuristics needed like the sampling / unsampling approach proposed in the literature to relax the typical high calculation effort. To the best of the author's knowledge the presented strategy results in the currently fastest LS procedures in the Literature. The algorithm is available as Java code on GitHub (Glassen, 2018a).

Keywords: Consensus Clustering, Median Partition, Adjusted Rand Index, Normalized Mutual Information

Introduction

Consensus Clustering (CC) as a method has evolved since the advent of its popularity by e. g. Strehl and Ghosh (2002) or Monti, Tamayo, Mesirov, and Golub (2003) to an extensive field of research. The term CC comprises different approaches to calculate a consensus from a partition distribution or a clustering ensemble, which reflects the available clusterings as much as possible. Despite the enormous variety of different methods that have been developed so far, almost all procedures can be assigned to one of two classes: methods that develop the consensus based on the co-currence of the clustered individuals and methods which construct a median partition as the solution of an optimization process (Vega-Pons & Ruiz-Shulcloper, 2011). In principle, both approaches are considered to be equal (Topchy, Law, Jain, & Fred, 2004), although the latter class is regarded as theoretically better founded (Vega-Pons & Ruiz-Shulcloper, 2011). Thus, based on a predefined similarity measure S, the median partition P^* is defined as the clustering, which has the highest sum of similarites to all partitions P_i in the ensemble $E = \{P_1, ..., P_K\}$:

$$P^* = \arg\max_P \sum_{i=1}^{K} S(P_i, P) \tag{1}$$

Analogously, a distance measure can be used instead of a similarity measure, whereby the searched $P*$ then represents the partition with the minimum sum of distances (Barthlemy & Leclerc, 1993).

The median partition problem, which goes back to Régnier (1983), is considered NP-hard at least for the simple similarity measure of the Mirkin-Metric for cluster ensembles of variable size K (Filkov & Skiena, 2004). Accordingly, procedures within the class of the median partition paradigm typically have a heuristic approach to the problem (Vega-Pons & Ruiz-Shulcloper, 2011). One group of procedures within this paradigm are based on counting-pairs measures, such as the Mirkin-Metric, and can be further divided in clustering-based heuristics and LS heuristics (Vega-Pons & Ruiz-Shulcloper, 2011). Methods of the first group merge iteratively objects and clusters whose similarity, based on a predefined counting-pair measure, does not fall below a treshhold. LS methods instead start with a median partition candidate in which all individuals successively are moved to another or a new cluster until either no further similarity improvement is achievable or another termination criterion is reached.

Goder and Filkov (2008) compared methods of these two subgroups and found that the clustering-based methods are usually faster but often generate solutions of lower quality than LS methods. Also Gionis, Mannila, and Tsaparas (2007) came to a similar result, in which their LS method consistently had the lowest distance errors of the tested procedures. Both author groups state a quadratic factor of the partition size N within the runtime complexities of their LS methods, which is generally considered as not practical for realistic datasets (Bertolacci & Wirth, 2007; Vega-Pons & Ruiz-Shulcloper, 2011). As Goder and Filkov (2008) have already noted, the runtime essentially depends on an efficient calculation of the counting-pairs measure. For this reason, the authors confined themselves to the easily computable Mirkin Metric, which, according to their knowledge, can be updated very quickly unlike the Adjusted Rand Index or the Jaccard Index. In fact, Goder and Filkov (2008) achieve an update complexity of the Merkin-Metric of $O(|B_s| + |B_d|) = O(N)$

Table 1: Intersection Size Matrix of P_i and P

Cluster	B_1	...	B_C
B_{i1}	$m_{i11} = \lvert B_{i1} \cap B_1 \rvert$	...	$m_{i1C} = \lvert B_{i1} \cap B_C \rvert$
...	...	...	...
B_{iC_i}	$m_{iC_i1} = \lvert B_{iC_i} \cap B_1 \rvert$	...	$m_{iC_iC} = \lvert B_{iC_i} \cap B_C \rvert$

instead of $O(K(N + C^2))$ for the new overall similarity of the candidate to the ensemble, after an object of Cluster B_s has been moved to another or new cluster B_d. However, their method requires initial preprocessing costs of $O(N^2K)$ for building a bookkeeping matrix, which makes the entire process still expensive.

In the following a general strategy is presented, which significantly surpasses this runtime complexity. Concretely the presented strategy updates the overall similarity after a movement of an individual in $O(K)$ and requires only $O(K(N + C^2))$ pre-processing costs. Additionally this strategy does not only work when using the Mirkin-Metric as comparison measure, but also when using the Rand Index, Adjusted Rand Index, Jaccard Index, Folwkes-Mallows Index, Wallace Coefficient, the Variation of Information, Mutual Information or Normalized Mutual Information. No matter which of these measures is used, the resulting LS method will have a per cycle runtime complexity of $O(KNC)$, which is to the best of the authors knowledge currently the fastest runtime complexity of a LS algorithm in the literature.

The structure of this article is as follows. First the building blocks of the algorithm are presented, the preprocessing part, the similarity calculation between partitions, the dynamic measure update and the LS procedure itself. Afterwards, I verify the determined runtime complexity of the new method on the basis of a selected dataset by varying the number of clusters and the partition size N by adding new individuals. In addition, I investigate the impact of the choice of the similarity measure on the accuracy of the consensus and, in anticipation of a replication of Goder and Filkov (2008) and Gionis et al. (2007), the accuracy of the new method in general to a known representative of the clustering-based procedures.

Methods

Preprocessing

Let $I = \{x_1, ..., x_N\}$ be the individuals of the dataset to be clustered, $E = \{P_1, ..., P_K\}$ the ensemble with K partitions, where each partition $P_i = \{B_{i1}, ..., B_{iC_i}\}$ contains C_i clusters B with $B_{ij} \cap B_{ik} = \emptyset$ for arbitrary $j \neq k$ and $\bigcup_{j=1}^{C_i} B_{ij} = I$. $P = \{B_1, ..., B_C\}$ is the consensus candidate.

In the first preprocessing step, we generate K so-called intersection-size matrices of the form as shown in table 1.

Then we calculate K so-called mismatch matrices (shown in table 2) using the fol-

Table 2: Counts of pairs that are / are not in the same cluster

Cluster of x_j, x_k in P_i / P	same	different
same	a_i	c_i
different	d_i	b_i

lowing formulas from Hubert and Arabie (1985):

$$a_i = \frac{1}{2}\sum_{j=1}^{C_i}\sum_{k=1}^{C} m_{ijk}(m_{ijk}-1) \tag{2}$$

$$b_i = \frac{1}{2}(N^2 + \sum_{j=1}^{C_i}\sum_{k=1}^{C} m_{ijk}^2 - (\sum_{j=1}^{C_i} m_{ij.}^2 + \sum_{k=1}^{C} m_{i.k}^2)) \tag{3}$$

$$c_i = \frac{1}{2}(\sum_{j=1}^{C_i} m_{ij.}^2 - \sum_{j=1}^{C_i}\sum_{k=1}^{C} m_{ijk}^2) \tag{4}$$

$$d_i = \frac{1}{2}(\sum_{k=1}^{C} m_{i.k}^2 - \sum_{j=1}^{C_i}\sum_{k=1}^{C} m_{ijk}^2) \tag{5}$$

The variables $m_{ij.} = \sum_{k=1}^{C} |B_{ij} \cap B_k| = |B_{ij}|$ and $m_{i.k} = \sum_{j=1}^{C_i} |B_{ij} \cap B_k| = |B_k|$ correspond to the respective cluster sizes in both partitions. Finally, we calculate the entropy of each partition P_i and candidate P as well as the joint entropy of each pair (P_i, P) for $i \in \{1, ..., K\}$ according to Ghosh and Acharya (2011). The entropy of a partition P_i in the ensemble is

$$H(P_i) = -\sum_{j=1}^{C_i} \frac{m_{ij.}}{N} \log(\frac{m_{ij.}}{N}) \tag{6}$$

and that of the candidate

$$(P) = -\sum_{k=1}^{C_k} \frac{m_{i.k}}{N} \log(\frac{m_{i.k}}{N}) \tag{7}$$

The respective joint entropy is

$$H(P_i, P) = -\sum_{j=1}^{C_i}\sum_{k=1}^{C_k} \frac{m_{ijk}}{N} \log(\frac{m_{ijk}}{N}) \tag{8}$$

Generating the K intersection-size matrices costs $O(KN)$. K mismatch matrices can be created in $O(KC^2)$. The calculation of entropy and joint entropy for each partition pair requires $O(KC^2)$ steps. The total preprocessing costs are therefore $O(K(N + C^2))$.

Similarity calculation

After the intersection-size and mismatch matrices are constructed, the similarity measures listed in the introduction can be calculated according to table 3 (Wallace, 1983; Fred & Jain, 2005; Wagner & Wagner, 2007; Santos & Embrechts, 2009; Ghosh & Acharya,

Table 3: Calculation of the various similarity measures

Measure	Formula
Rand Index	$\frac{a_i+b_i}{a_i+b_i+c_i+d_i}$
Adjusted Rand Index	$\frac{\frac{N(N-1)}{2}(a_i+b_i)-[(a_i+c_i)(a_i+d_i)+(d_i+b_i)(c_i+b_i)]}{\frac{N(N-1)}{2}-[(a_i+c_i)(a_i+d_i)+(d_i+b_i)(c_i+b_i)]}$
Fowlkes-Mallows Index	$\frac{a_i}{\sqrt{(a_i+c_i)(a_i+d_i)}}$
Jaccard Index	$\frac{a_i}{a_i+c_i+d_i}$
Mirkin Metric	$2(c_i+d_i)$
Wallace Coefficient 1	$\frac{a_i}{a_i+c_i}$
Wallace Coefficient 2	$\frac{a_i}{a_i+d_i}$
Variation of Information	$2H(P_i,P)-H(P_i)-H(P)$
Mutual Information	$H(P_i)+H(P)-H(P_i,P)$
Normalized Mutual Information 1	$\frac{H(P_i)+H(P)-H(P_i,P)}{\sqrt{H(P_i)H(P)}}$
Normalized Mutual Information 2	$\frac{2(H(P_i)+H(P)-H(P_i,P))}{H(P_i)+H(P)}$

2011). As can be seen, the calculation of each of these measures costs $O(1)$ per partition pair or $O(K)$ for the overall similarity of P to E, if (2-8) is given. We therefore are interested in maintaining (2-8) after changes to the partitions were applied. The next subsection describes how to update these as well as further bookkeeping variables on a minimum cost.

Measure update

When looking at the intersection-size matrices, one recognizes that if an individual x_j from the candidates' cluster B_{s_2} is to be moved into a cluster B_d, only the cells $m_{is_1s_2}$ and m_{is_1d} are affected, where B_is_1 is the cluster of x_j in $P_i \in E$. Specifically, $m_{is_1s_2}$ is decremented and m_{is_1d} is incremented. Furthermore, we see in (2-5) that for a corresponding update of the mismatch matrices of the (P_i, P) we only have to replace the portion of $m_{is_1s_2}$ and m_{is_1d} in each of the sums (2-5) with the new portion of $m_{is_1s_2}-1$ and $m_{is_1d}+1$. This principle of summand replacement is also applicable if an update of the entropies $H(P_i)$ and $H(P_i, P)$ is needed. In the following I give an overview of how each variable exactly needs to be changed. The individual derivations for each variable can be found in the Appendix.

$$\Delta a_i = m_{is_1d} - (m_{is_1s_2} - 1) \tag{9}$$

$$\Delta b_i = (m_{i.s_2} - m_{is_1s_2}) - (m_{i.d} - m_{is_1d}) \tag{10}$$

$$\Delta c_i = (m_{is_1s_2} - 1) - m_{is_1d} \tag{11}$$

$$\Delta d_i = (m_{i.d} - m_{is_1d}) - (m_{i.s_2} - m_{is_1s_2}) \tag{12}$$

$$\Delta H(P) = \sum_{z=0}^{1} \frac{(-1)^z}{N} \left((m_{i.s_2} - z) \log(\frac{m_{i.s_2} - z}{N}) + (m_{i.d} + z) \log(\frac{m_{i.d} + z}{N}) \right) \tag{13}$$

$$\Delta H(P_i, P) = \sum_{z=0}^{1} \frac{(-1)^z}{N} \left((m_{is_1s_2} - z) \log(\frac{(m_{is_1s_2} - z}{N}) + (m_{is_1d} + z) \log(\frac{m_{is_1d} + z}{N}) \right) \tag{14}$$

$$\Delta m_{is_1s_2} = -1 \tag{15}$$

$$\Delta m_{is_1d} = 1 \tag{16}$$

Thus, we apply the updates (9-16) for each pending movement of x_j and change the cluster membership of x_j afterwards in only $O(1)$ steps per partition pair or $O(K)$ for all pairs (P_i, P) with $i \in \{1, ..., K\}$.

Local Search procedure

The LS algorithm now works like this. We first select an arbitrary consensus partition P and start the preprocessing steps. Afterwards we improve the consensus by moving individuals in P systematically to other existing or new clusters in P. Each time we move an individual x_j, we apply the measure update to ensure that the overall similarity of P to E can be quickly re-calculated. If this overall similarity decreases we undo the movement, otherwise we stay with the new consensus. If all individuals have been considered in this way the optimization process restarts with the first individual. This procedure is continued as long as a better consensus was found in one cycle over all individuals. The detailed sequence of those steps are shown in algorithm 1.

I have now presented the entire procedure and want to take a closer look at its performance in the next section.

Results

Consensus Clustering is not only used to calculate a consensus between the results of different clustering methods, but also to determine the median or mean partition of a partition distribution. For example, Huelsenbeck and Andolfatto (2007) used a LS procedure with the tranfer distance as a measure of similarity to determine a representative partition from the samples of a Dirichlet process mixing model (DPMM). The transfer distance is a measure that is defined by the minimum number of movements of individuals between the clusters until both partitions match (Gusfield, 2002; Denœud, 2008).

In order to achieve a polynomial runtime complexity with the procedure of Huelsenbeck and Andolfatto (2007), the calculation of the transfer distance is usually reduced to the solution of a Linear Sum Assignment Problem (LSAP) using the approach of

Algorithm 1 Local Search procedure for constructing the consensus P^*

Input: Clustering Ensemble E
Output: partition with local maximum similarity to all partitions in E

```
P = any, flag = true, highestSimilaritySum = 0
for all P_i ∈ E do
  highestSimilaritySum += S(P_i, P)
end for
while flag do
  flag = false
  for all x_j ∈ I do
    bestB = current cluster of x_j in P
    for all B_d ∈ P ∪ { empty cluster} do
      similaritySum = 0
      for all P_i ∈ E do
        move(P_i, P, x_j, B_d)
        similaritySum += S(P_i, P)
      end for
      if similaritySum > highestSimilaritySum then
        highestSimilaritySum = similaritySum
        bestB = B_d
        flag = true
      end if
    end for
    for all P_i ∈ E do
      move(P_i, P, x_j, bestB)
    end for
  end for
end while
return P
```

Konovalov, Litow, and Bajema (2005) (e.g., Onogi, Nurimoto, & Morita, 2011). The typical implementation of this reduction in the literature costs $O(NC^2)$ (e.g. in, Onogi et al., 2011; Konovalov et al., 2005), while solving the LSAP costs $O(C^3)$. The entire procedure of Huelsenbeck and Andolfatto (2007) then has a runtime of $O(KN^2C^3)$.

However, it is possible to accelerate this procedure by using the reduction algorithm of Glassen (2018b) instead of the typical implementation. The reduction of Glassen (2018b) costs only $O(N + C^2)$ by which the runtime of the entire procedure of Huelsenbeck and Andolfatto (2007) reduces to $O(KNC^4 + KN^2C)$. Note that in practice $N >> C$ is usually present and thus the second term dominates the new runtime complexity. If this is the case, we still have a runtime with a quadratic factor N.

In the following I will call this improved procedure LSHA and compare it with the LS procedure developed in this article (named LSNEW from now on) as well as with the heuristic CC-Pivot + Pick-A-Cluster (CC+PAC) (Ailon, Charikar, & Newman, 2008). The latter is a clustering-based procedure with a guaranteed total runtime of $O(KNC)$ and the

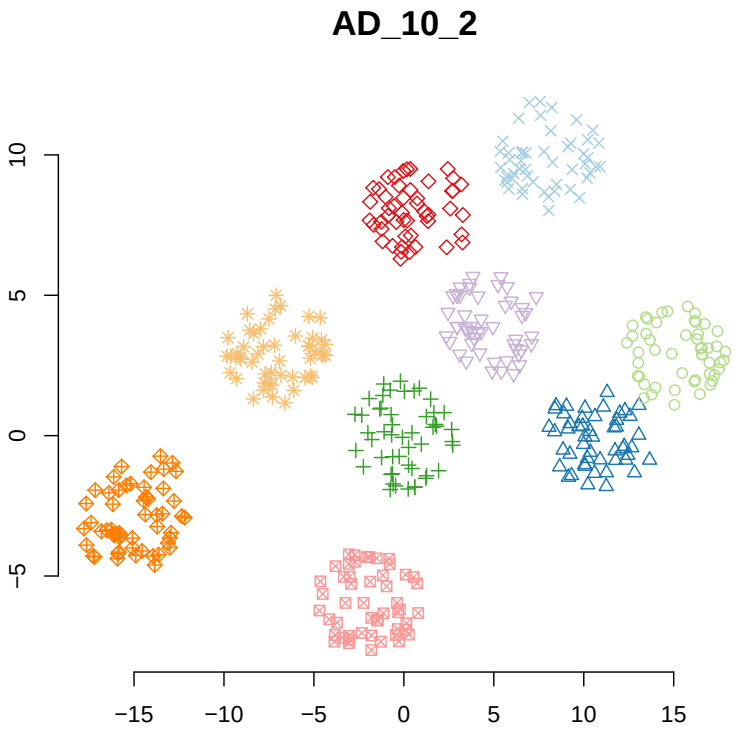

Figure 1. The used dataset from Bandyopadhyay & Maulik (2001)

second best known approximation factor of $\frac{11}{7}$ (Bertolacci & Wirth, 2007; Xanthopoulos, 2014). CC+PAC was preferred over the method CCLP-Pivot, which has the best known approximation factor of $\frac{4}{3}$. This is because the latter has a runtime complexity of $O(N^8)$ (Xanthopoulos, 2014) and is therefore completely impractical for realistic datasets. We assume that analogous to Goder and Filkov (2008) and Gionis et al. (2007), we observe a superiority of LSNEW with regard to the quality of the determined consensus compared to CC+PAC. At the same time, we expect the improved runtime of LSNEW to be close to the runtime of the extremely fast CC+PAC. For the benchmarks I used the dataset'AD_10_2' of Bandyopadhyay and Maulik (2001), shown in figure 1. The partition ensembles are created via the DPMM described in figure 2. All calculations were done on a PC with i7-4870HQ CPU.

Benchmark 1

In the first benchmark, the algorithms' performance is tested for a different number of clusters in the partitions. Concretely, I varied the concentration parameter $\alpha = \{2, 4, 6, ..., 20\}$ of the DPMM and generated for each parameter setting 100 sample partitions via Gibbs Sampling using the dataset already mentioned. Afterwards a consensus was determined via CC+PAC and every variant of LSNEW with each of the similarity mea-

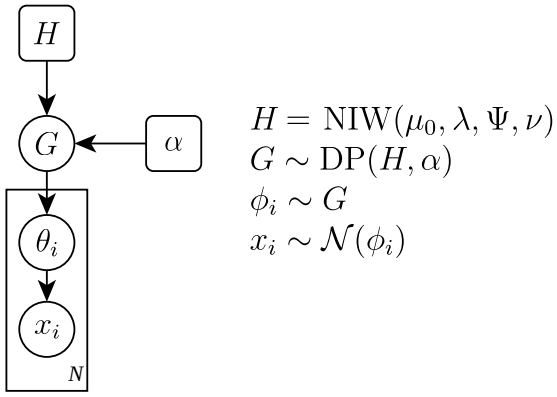

Figure 2. The applied DPMM. NIW is the Normal-inverse-Wishart distribution.

sures Transfer-Distance, Normalized Mutual Information 1, Adjusted Rand Index, Fowlkes-Mallows Index, Jaccard Index and Mirkin Metric. For every value of α, the consensus was calculated 400 times and an average similarity of the consensuses to their ensembles was determined based on each of the similarity measures used.

The result is shown in Figure 3. For better comparability the transfer- and mirkin distances were plotted standardized. The latter according to Meilă (2005) with $\mathrm{MM}_{\mathrm{st}} = 1 - \frac{\mathrm{MM}}{N^2}$ and the transfer distance with $\mathrm{TD}_{\mathrm{st}} = 1 - \frac{\mathrm{TD}}{N-1}$. On the x-axis of the plots, the average number of clusters $\bar{C}$ of the partitions in the ensemble E is plotted instead of α.

As can be seen, LSNEW combined with every similarity measure leads to a more accurate consensus on each of the external measures of validity compared to the clustering-based heuristic CC+PAC. Concurrently, the runtimes of the LSNEW variants show, as expected, only a shift by a constant from the runtime of CC+PAC with a linear increase over $\bar{C}$. However, in comparison to the procedure of Glassen (2018b) TSNEW leads to a clearly growing time advantage as $\bar{C}$ increases.

Although LS procedures are heuristics of unknown runtime complexity (Vega-Pons & Ruiz-Shulcloper, 2011), there seems to be no linear relationship between the average number of iterations and $\bar{C}$ at least for this dataset. More specifically, the former did not exceed a constant (here = 6) for the tested α. If this pattern would apply to every interval of α and every dataset, the runtime complexity of an iteration of LS procedures would correspond to that of their total runtime. Thus CC+PAC and the LSNEW variants would be consistent in this respect. I will pick this up in the discussions section.

Benchmark 2

In the second benchmark, the number of individuals in the partitions was varied instead of α. For this purpose, new individuals were added uniformly clusterwise to the original dataset in Figure 1 by assuming a multivariate normal distribution of the cluster members. In addition to the original dataset with an $N = 500$, $N = 550, 600, 650, ..., 1000$ were tested and a fixed $\alpha = 2$ was used. The result is shown in Figure 4.

In this benchmark, CC+PAC also consistently proves to be the procedure with the

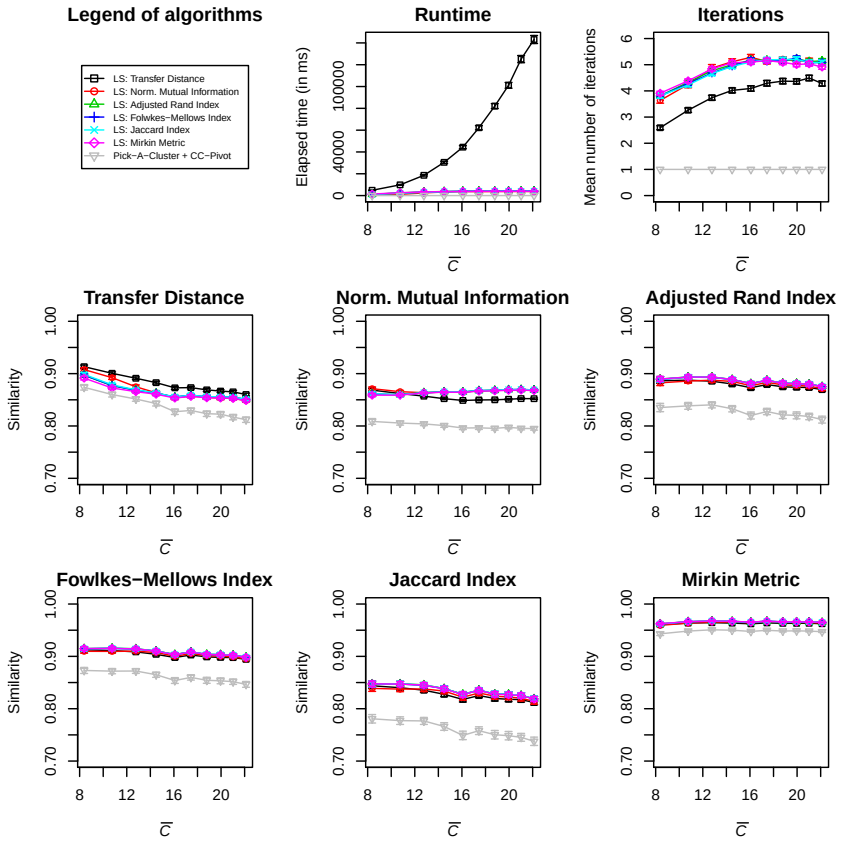

Figure 3. The results of the simulation with varying α. Error bars are 95% confidence intervals with the assumption of approximately normal distributed means. LSHA is labeled in this figure as LS: Transfer Distance, all other LS algorithms are variants of LSNEW.

lowest accuracy, but again also as the procedure with the lowest runtime. It is followed by the LSNEW variant with the Normalized Mutual Information as similarity measure, which also clearly differs from the other alternatives. It seems that at this combination, individuals are not moved so often between clusters, which is also reflected by the average number of iterations. In the latter, the mentioned combination tends to have the lower mean values compared to the other LSNEW variants.

Similar to benchmark 1, there seems to be no linear relationship between the number of iterations and the partition size in benchmark 2. As can be seen, the average number of iterations remains almost constant for all LS procedures and partition sizes.

Overall in benchmark 2, it can be concluded that the runtimes of the individual algorithms also give an expected picture. The average runtimes of the LSNEW variants stay relatively close to those of CC+PAC for increasing N, while the difference to the procedure of Huelsenbeck and Andolfatto (2007) becomes increasingly larger. Both benchmarks thus confirm the complexity analyses.

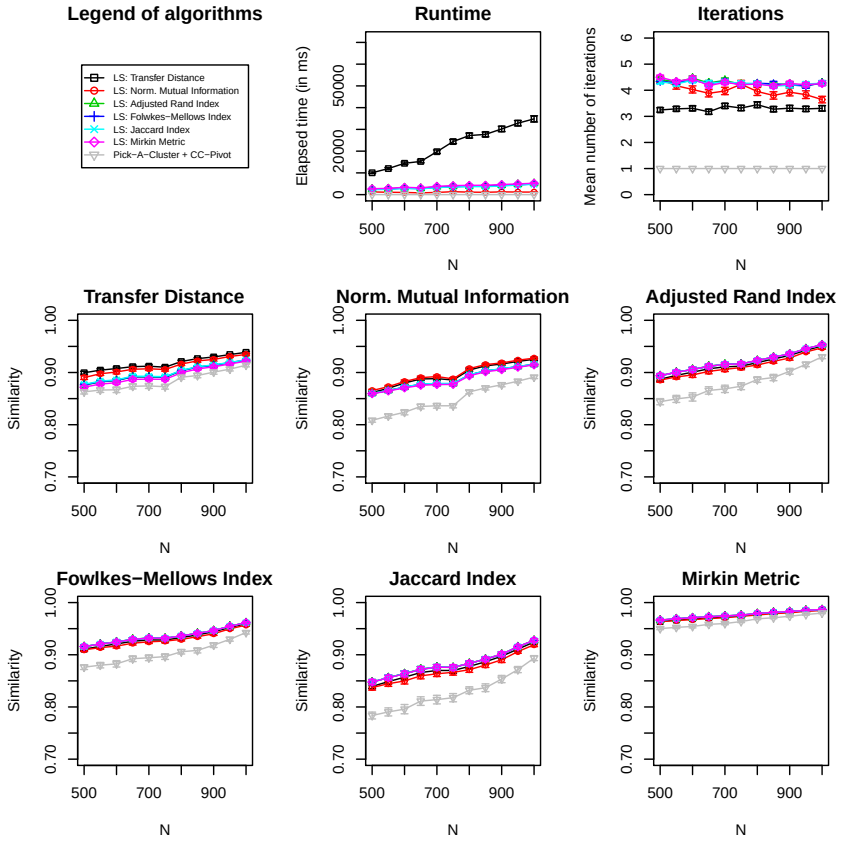

Figure 4. The results of the simulation with varying N. Error bars are 95% confidence intervals with the assumption of approximately normal distributed means. LSHA is labeled in this figure as LS: Transfer Distance, all other LS algorithms are variants of LSNEW.

Discussion and conclusion

To the best of the authors knowledge, the improved LS variants in this article have the currently lowest runtime complexity of all LS procedures in the literature. Thus, they allow the calculation of consensuses even for realistic datasets in a reasonable time. Simultaneously the presented variants have a practically provable higher accuracy than the very fast method with the currently second-best approximation factor CC+PAC. First explorative benchmarks also suggest the possibility that the runtime complexity of one iteration of LS procedures could also correspond to that of their total runtime. If this is true, the LS variants presented here and CC+PAC could possibly have the same overall runtime complexity. Further evaluations are necessary in order to obtain more certainty here, especially with regard to the question of the influence of specific properties of a dataset on the number of iterations of the procedures.

Regardless of this, practitioners now have access to LS procedures, which are not only faster than previously published in the literature, but also provide alternative similarity measures. Previous LS methods were essentially limited on the Mirkin Metric (Filkov & Skiena, 2004; Goder & Filkov, 2008; Gionis et al., 2007), the Transfer Distance (Huelsenbeck

& Andolfatto, 2007) or the Adjusted Rand Index (Krieger & Green, 1999).

Note that the presented strategy in this paper does not support LS methods that use the Transfer Distance as comparison measure between partitions. If one needs a faster LS method with this measure, I refer to the meanwhile published paper of Glassen, Oertzen, and Konovalov (2018). A detailed comparison between that algorithm and those presented in this article needs to be done in the future. Nevertheless, the method of Glassen et al. (2018) needs $O(KNC^3)$ steps per cycle while the LS procedures in this article need only $O(KNC)$.

References

Ailon, N., Charikar, M., & Newman, A. (2008). Aggregating Inconsistent Information: Ranking and Clustering. *J. ACM*, *55*(5), 23:1–23:27. doi: 10.1145/1411509.1411513

Bandyopadhyay, S., & Maulik, U. (2001). Nonparametric genetic clustering: comparison of validity indices. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, *31*(1), 120–125. doi: 10.1109/5326.923275

Barthlemy, J.-P., & Leclerc, B. (1993). The Median Procedure for Partitions. In *Partitioning Data Sets, Proceedings of a DIMACS Workshop, Princeton, New Jersey, USA, April 19-21, 1993* (pp. 3–34).

Bertolacci, M., & Wirth, A. (2007). Are approximation algorithms for consensus clustering worthwhile? In *Proceedings of the 2007 SIAM International Conference on Data Mining* (pp. 437–442). Society for Industrial and Applied Mathematics. (DOI: 10.1137/1.9781611972771.41)

Denœud, L. (2008). Transfer distance between partitions. *Advances in Data Analysis and Classification*, *2*(3), 279–294. doi: 10.1007/s11634-008-0029-0

Filkov, V., & Skiena, S. (2004). Integrating microarray data by consensus clustering. *International Journal on Artificial Intelligence Tools*, *13*(04), 863–880. doi: 10.1142/S0218213004001867

Fred, A. L. N., & Jain, A. K. (2005). Combining multiple clusterings using evidence accumulation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *27*(6), 835–850. doi: 10.1109/TPAMI.2005.113

Ghosh, J., & Acharya, A. (2011). Cluster ensembles. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, *1*(4), 305–315. doi: 10.1002/widm.32

Gionis, A., Mannila, H., & Tsaparas, P. (2007). Clustering Aggregation. *ACM Trans. Knowl. Discov. Data*, *1*(1). doi: 10.1145/1217299.1217303

Glassen, T. (2018a). *Local Search with dynamic similarity updates.* `https://github.com/t-glassen/dynamic_updates`. Accessed 1 Jan 2018.

Glassen, T. (2018b). *Psychologisch orientierte Kategorisierung in der kognitiven Robotik mit dem Hierarchischen Dirichlet Prozess.* dissertation.

Glassen, T., Oertzen, T. v., & Konovalov, D. A. (2018). Finding the mean in a partition distribution. *BMC Bioinformatics*, *19*(1), 375. Retrieved 2018-11-03, from `https://doi.org/10.1186/s12859-018-2359-z` doi: 10.1186/s12859-018-2359-z

Goder, A., & Filkov, V. (2008). Consensus Clustering Algorithms: Comparison and Refinement. In *Proceedings of the Meeting on Algorithm Engineering & Expermiments* (pp. 109–117). Philadelphia, PA, USA: Society for Industrial and Applied Mathematics.

Gusfield, D. (2002). Partition-distance: A problem and class of perfect graphs arising in clustering. *Information Processing Letters*, *82*(3), 159–164. doi: 10.1016/S0020-0190(01)00263-0

Hubert, L., & Arabie, P. (1985). Comparing partitions. *Journal of Classification*, *2*(1), 193–218. doi: 10.1007/BF01908075

Huelsenbeck, J. P., & Andolfatto, P. (2007). Inference of Population Structure Under a Dirichlet Process Model. *Genetics*, *175*(4), 1787. doi: 10.1534/genetics.106.061317

Konovalov, D. A., Litow, B., & Bajema, N. (2005). Partition-distance via the assignment problem. *Bioinformatics*, *21*(10), 2463–2468. doi: 10.1093/bioinformatics/bti373

Krieger, A. M., & Green, P. E. (1999). A Generalized Rand-Index Method for Consensus Clustering of Separate Partitions of the Same Data Base. *Journal of Classification*, *16*(1), 63–89. doi: 10.1007/s003579900043

Meilă, M. (2005). Comparing Clusterings: An Axiomatic View. In *Proceedings of the 22nd International Conference on Machine Learning* (pp. 577–584). New York, NY, USA: ACM. doi: 10.1145/1102351.1102424

Monti, S., Tamayo, P., Mesirov, J., & Golub, T. (2003). Consensus Clustering: A Resampling-Based Method for Class Discovery and Visualization of Gene Expression Microarray Data. *Machine Learning*, *52*(1-2), 91–118. doi: 10.1023/A:1023949509487

Onogi, A., Nurimoto, M., & Morita, M. (2011). Characterization of a Bayesian genetic clustering algorithm based on a Dirichlet process prior and comparison among Bayesian clustering methods. *BMC Bioinformatics*, *12*, 263. doi: 10.1186/1471-2105-12-263

Régnier, S. (1983). Sur quelques aspects mathmatiques des problmes de classification automatique. *Mathmatiques et Sciences Humaines*, *82*, 13–29.

Santos, J. M., & Embrechts, M. (2009). On the Use of the Adjusted Rand Index as a Metric for Evaluating Supervised Classification. In *Artificial Neural Networks ICANN 2009* (pp. 175–184). Springer, Berlin, Heidelberg. doi: 10.1007/978-3-642-04277-5_18

Strehl, A., & Ghosh, J. (2002). Cluster Ensembles — A Knowledge Reuse Framework for Combining Multiple Partitions. *Journal of Machine Learning Research*, *3*(Dec), 583–617.

Topchy, A. P., Law, M. H. C., Jain, A. K., & Fred, A. L. (2004). Analysis of consensus partition in cluster ensemble. In *Fourth IEEE International Conference on Data Mining, 2004. ICDM '04* (pp. 225–232). doi: 10.1109/ICDM.2004.10100

Vega-Pons, S., & Ruiz-Shulcloper, J. (2011). A survey of clustering ensemble algorithms. *International Journal of Pattern Recognition and Artificial Intelligence*, *25*(03), 337–372. doi: 10.1142/S0218001411008683

Wagner, S., & Wagner, D. (2007). Comparing Clusterings - An Overview.

Wallace, D. L. (1983). Comment. *Journal of the American Statistical Association*, *78*(383), 569–576. doi: 10.1080/01621459.1983.10478009

Xanthopoulos, P. (2014). A Review on Consensus Clustering Methods. In *Optimization in Science and Engineering* (pp. 553–566). Springer, New York, NY. (DOI: 10.1007/978-1-4939-0808-0_26)

Appendix

Update of a_i:

$$\begin{aligned} t_1 &= \frac{1}{2}\left(-(m_{is_1s_2}(m_{is_1s_2}-1)) + ((m_{is_1s_2}-1)((m_{is_1s_2}-1)-1))\right) \\ &= \frac{1}{2}\left(-m_{is_1s_2}^2 + m_{is_1s_2} + m_{is_1s_2}^2 - 2m_{is_1s_2} + 1 - m_{is_1s_2} + 1\right) = -(m_{is_1s_2}-1) \end{aligned} \tag{17}$$

$$\begin{aligned} t_2 &= \frac{1}{2}\left(-(m_{is_1d}(m_{is_1d}-1)) + ((m_{is_1d}+1)((m_{is_1d}+1)-1))\right) \\ &= \frac{1}{2}\left(-m_{is_1d}^2 + m_{is_1d} + m_{is_1d}^2 + 2m_{is_1d} + 1 - m_{is_1d} - 1\right) = m_{is_1d} \end{aligned} \tag{18}$$

$$\Delta a_i = t_1 + t_2 = m_{is_1d} - (m_{is_1s_2} - 1)$$

Update of b_i:

$$\begin{aligned} t_3 &= \frac{1}{2}\left(-m_{is_1s_2}^2 + (m_{is_1s_2}-1)^2 + m_{i.s_2}^2 - (m_{i.s_2}-1)^2\right) \\ &= \frac{1}{2}\left(-m_{is_1s_2}^2 + m_{is_1s_2}^2 - 2m_{is_1s_2} + 1 + m_{i.s_2}^2 - m_{i.s_2}^2 + 2m_{i.s_2} - 1\right) = m_{i.s_2} - m_{is_1s_2} \end{aligned} \tag{19}$$

$$\begin{aligned} t_4 &= \frac{1}{2}\left(-m_{is_1d}^2 + (m_{is_1d}+1)^2 + m_{i.d}^2 - (m_{i.d}+1)^2\right) \\ &= \frac{1}{2}\left(-m_{is_1d}^2 + m_{is_1d}^2 + 2m_{is_1d} + 1 + m_{i.d}^2 - m_{i.d}^2 - 2m_{i.d} - 1\right) = m_{is_1d} - m_{i.d} \end{aligned} \tag{20}$$

$$\Delta b_i = t_3 + t_4 = (m_{i.s_2} - m_{is_1s_2}) + (m_{is_1d} - m_{i.d})$$

Update of c_i:

$$\begin{aligned} t_5 &= \frac{1}{2}\left(m_{is_1s_2}^2 - (m_{is_1s_2}-1)^2\right) \\ &= \frac{1}{2}\left(m_{is_1s_2}^2 - m_{is_1s_2}^2 + 2m_{is_1s_2} - 1\right) = m_{is_1s_2} - \frac{1}{2} \end{aligned} \tag{21}$$

$$\begin{aligned} t_6 &= \frac{1}{2}\left(m_{is_1d}^2 - (m_{is_1d}+1)^2\right) \\ &= \frac{1}{2}\left(m_{is_1d}^2 - m_{is_1d}^2 - 2m_{is_1d} - 1\right) = -m_{is_1d} - \frac{1}{2} \end{aligned} \tag{22}$$

$$\Delta c_i = t_5 + t_6 = (m_{is_1s_2} - 1) - m_{is_1d}$$

Update of d_i:

$$\begin{aligned} t_7 &= \frac{1}{2}\left(-m_{i.s_2}^2 + (m_{i.s_2}-1)^2 + m_{is_1s_2}^2 - (m_{is_1s_2}-1)^2\right) \\ &= \frac{1}{2}\left(-m_{i.s_2}^2 + m_{i.s_2}^2 - 2m_{i.s_2} + 1 + m_{is_1s_2}^2 - m_{is_1s_2}^2 + 2m_{is_1s_2} - 1\right) = -(m_{i.s_2} - m_{is_1s_2}) \end{aligned} \tag{23}$$

$$\begin{aligned} t_8 &= \frac{1}{2}\left(-m_{i.d}^2 + (m_{i.d}+1)^2 + m_{is_1d}^2 - (m_{is_1d}+1)^2\right) \\ &= \frac{1}{2}\left(-m_{i.d}^2 + m_{i.d}^2 + 2m_{i.d} + 1 + m_{is_1d}^2 - m_{is_1d}^2 - 2m_{is_1d} - 1\right) = m_{i.d} - m_{is_1d} \end{aligned} \tag{24}$$

$$\Delta d_i = t_7 + t_8 = (m_{i.d} - m_{is_1d}) - (m_{i.s_2} - m_{is_1s_2})$$