Erschienen in: Advances in Intelligent Data Analysis XII / Tucker, Allan; Höppner, Frank; Siebes, Arno; Swift, Stephen (Hrsg.). - Berlin, Heidelberg : Springer Berlin Heidelberg, 2013. - (Lecture Notes in Computer Science ; 8207). - S. 223-236. - ISBN 978-3-642-41397-1
https://dx.doi.org/10.1007/978-3-642-41398-8_20

Diversity-Driven Widening

Violeta N. Ivanova and Michael R. Berthold

Dept. of CIS and Graduate School Chemical Biology (KoRS-CB)
University of Konstanz, 78457 Konstanz, Germany
`firstname.lastname@uni-konstanz.de`

Abstract. This paper follows our earlier publication [1], where we introduced the idea of tuned data mining which draws on parallel resources to improve model accuracy rather than the usual focus on speed-up. In this paper we present a more in-depth analysis of the concept of Widened Data Mining, which aims at reducing the impact of greedy heuristics by exploring more than just one suitable solution at each step. In particular we focus on how diversity considerations can substantially improve results. We again use the greedy algorithm for the set cover problem to demonstrate these effects in practice.

1 Introduction

In [1], we claim that utilizing parallel compute resources to improve the accuracy of data mining algorithms and to obtain better models is of merit and is an important, emerging area of research in the the field of (parallel) data mining. The main reason for data mining algorithms not finding optimal solutions, is the usually enormous solution space, which requires the use of a – often greedy – heuristic. While this helps in finding a solution in reasonable time, it limits the exploration of the solution space and often leads to suboptimal solutions. In [1] we presented two generic strategies for using parallel resources to improve a greedy data mining heuristic, namely *Deepening* and *Widening*. Deepening focuses on smarter strategies to pick temporary solutions by looking several steps ahead and selecting a temporary solution, which has shown potential to perform better further down the search. The goal of Widening, in contrast, lies on achieving better accuracy by exploring more solutions simultaneously, not just the locally optimal one. We then demonstrated that both of these tuning methods offer potential for improvement using a widened versions of the greedy base algorithm for the set cover problem and a deepened decision tree learner.

In this paper we again focus on our key goal: the development of algorithm and architecture-independent generic strategies, which can be applied to a broad spectrum of data mining heuristics in order to improve their accuracy, while keeping the runtime constant. At the same time we wish to abstract away from implementation details, such as parallelization models. We will focus primarily on Widening in this paper and mechanisms to further improve the search.

The ideal goal of Widening is, given a sufficient (and hence usually enormous) number of parallel resources, to enable full exploration of the search space and guarantee the discovery of the optimal solution. In practice this is, of course, usually not feasible. Instead we need to make sure we make best use of every parallel resource available towards the goal of improvement of solution quality, while still keeping the runtime of the widened heuristic the same as the runtime of the original heuristic. The goal of cost-effective investment of parallel resources is closely related to the concept of *diverse exploration* of the search space, which is the main focus of this paper. The main goal of *Diversity-driven Widening* is to force different workers to investigate substantially different models, hence resulting in a diverse set of final solutions. Ideally this increases our chances to find an even better model than the standard heuristic. We describe simple ways to achieve diversity and illustrate how enforcing diversity in widening techniques helps to further improve the accuracy of the data mining heuristic. However, ensuring diversity often adds computational overhead which contradicts the second goal of Widening: keeping the runtime constant. We therefore investigate and compare Widening techniques with and without communication between the parallel workers. We demonstrate these different practical techniques on the greedy algorithm for the set cover problem.

2 Related Work

Trading quantity (of computational resources) for quality (of discovered solutions) has already been published before. In [2] the authors focus on a broad range of applications, ranging from cryptography to game playing. We, instead, focus on data mining algorithms which allows us to formalize a number of constraints based on the underlying model search space.

Plenty of related work exists in others areas, e.g. parallel data mining. We do not have the space to discuss all of this in detail and only briefly summarize the main trends and mainly focus on improvements of search heuristics through the use of diversity, as this is most relevant for the focus of this paper.

Speed-Up through Parallelization. For the vast majority of parallelizations of data mining algorithms, the aim is to improve efficiency. Comprehensive surveys are found in [3,4,5,6]. A large amount of work focuses on the parallelization of decision tree learning. One of the earliest distributed decision tree algorithms, SPRINT [7], has served as the basis for many subsequent parallel decision tree approaches. Some noteworthy examples include [8] (employing data parallelism), [9] (using task parallelism), and [10,11] (presenting hybrid approaches). Probably the second most researched area is parallel association rule mining algorithms. Extensive surveys exist in this area as well [12]. Parallelism in clustering algorithms has been used for both efficient cluster discovery and more efficient distance computations. Partitioning clustering algorithms are parallelized mostly using message-passing models, examples are presented in [13,14]. Examples for hierarchical clustering, which is more costly, include [15,16]. However, as discussed above, speed-up is not the primary goal of Widening.

In recent years specialized frameworks have also emerged, allowing data mining algorithms to be implemented in a distributed fashion and/or operating on distributed data. MapReduce [17] is the most prominent paradigm for processing parallel tasks across very large datasets and allows for massive scalability across thousands of servers in a cluster or a grid. Due to its inherent structure, MapReduce requires specialized versions of the data mining algorithms to be developed. Chu et al. [18] present a general approach by using a summation representation of the algorithms. While offering amazing scalability, MapReduce has inherent flaws – it is not designed to deal well with moderate-size data with complex dependencies; it is not suitable for algorithms that require communication between the parallel workers or impose other dependencies across iterations (see [19] for more details). So whereas MapReduce offers the potential for creating better models based on processing more data, Widening focuses on improving model quality using normal amounts of data. This does not exclude Widened MapReduce style implementations, of course, but it is not at the core of this paper.

Model Quality Improvement. A number of papers also concentrate on improving the accuracy of the models. Some approaches learn more models to be used in concert (ensembles) or in a randomized fashion (meta heuristics).

Ensembles use multiple models to obtain better predictive performance than could be obtained from any of the constituent models. The most notable examples are bootstrap aggregating or bagging [20], boosting [21], and random forests [22]. However, a high degree of accuracy comes at the price of interpretability, as these methods do not result in a single interpretable model, which is contrary to the goal of Widened Data Mining.

Learners based on stochastic learning algorithms, such as genetic algorithms are naturally parallelizable. Parallelization can be achieved by way of independent parallel execution of independent copies of a genetic algorithm, followed by selecting the best of the obtained results. This results in improved accuracy [23]. This is similar to Widening, however, Widening aims at exploring the search space in a structured way as opposed to the randomized nature of these other methods.

Using Diversity to Improve Search Heuristics. There is a wealth of literature focusing on the improvement of (greedy) search algorithms in general. In [24], an approach is presented for incorporating diversity within the cost function, which is used to select the intermediate solution. In [25], the authors use the observation that, in most cases, failing to find the optimal solution can be explained by a small number of erroneous decisions along the current path. Therefore, their improved search first explores the left-most child at a given depth as suggested by the current heuristic function. If no solution is found, all leaf nodes are tried that differ in just one decision point from the heuristic function choice. The algorithm iteratively increases the deviation from the greedy heuristic. The Widening proposed here performs a similar search for alternatives, but in parallel. In [26] the idea of adding diversity by a simple K-best first search was explored and shown empirically to be superior to the greedy (best-first) search heuristic.

Improving Set Covering. And finally, we should not forget to mention that a vast amount of literature addresses improving the greedy algorithm for the set cover problem [27,28]. Since we use this to illustrate the effects of Widening, we hasten to add that we do not intend to compete with these approaches! However, the greedy base set covering algorithm allows the benefits of Widening to be demonstrated well and intuitively.

3 General Widening of a Greedy Heuristic

We can view many of the data mining algorithms as a *greedy search* through a space of potential solutions, the model *search space*. This search space consists of model candidates, from which the greedy algorithm chooses a locally optimal solution at each step, until a sufficiently good solution is found, based on some stopping criteria. The greedy search can, therefore, be schematically presented as an iterative application of two operators: *refinement* r and *selection* s.

During the refinement operation, a temporary model m is made more specific to generate new, potentially better, models (which we refer to as *refinements*). The *selection* operator chooses the locally best model from all possible refinements.

For the purpose of this paper it is sufficient to assume the existence of a family of models $\mathcal{M}$, that constitutes the domain of the two operators. The refinement operator is model and algorithm specific and the selection operator is usually driven by the training data. We will investigate the selection operator in more detail, as it will be the tool we use to widen a greedy heuristic. It is usually based on a given quality measure ψ, which evaluates the quality of a model m from a family of models $\mathcal{M}$ (and therefore also its refinements):

$$\psi : \mathcal{M} \to \mathbb{R}.$$

Employing this notation, we can present one iterative step of the greedy search as follows:

$$m' = s_{\text{best}}(r(m)),$$

where

$$s_{\text{best}}(M) = \underset{m'' \in M}{\arg\max} \{\psi(m'')\},$$

that is, the model from the subset $M \subseteq \mathcal{M}$ which is ranked highest by the quality measure is chosen at each step. Figure 1 depicts this view of a greedy model searching algorithm.

We can now also specify how we got to a certain model and define the concept of *selection path*, which defines how a specific model is reached:

$$\mathcal{P}_s(m) = \{m^{(1)}, m^{(2)}, \ldots, m^{(n)}\},$$

where the order is specified via the refinement/selection steps, that is

$$\forall i = 1, \ldots, n-1 \,:\, m^{(i+1)} = s(r(m^{(i)})).$$

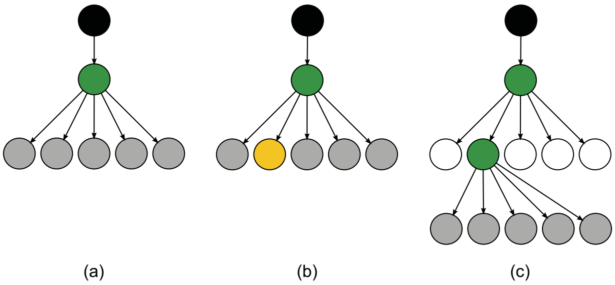

Fig. 1. The classic heuristic (often greedy) search algorithm. On the left (a), the current model m is depicted in green, the refinement options $r(m)$ are shown gray. The selection operator s picks the yellow refinement (b) and the next level then continues the search based on this choice.

and $m^{(n)} = m$ and $m^{(1)}$ is a root model for which no other model exists that it is a refinement of. Note that the selection path depends heavily on the chosen selection operator s, which will come in handy later.

3.1 Widening of a Greedy Heuristic

In order to improve the accuracy of the greedy algorithm one has to deal with its inherent flaw – the fact that a locally optimal choice may in fact not lead us towards the globally optimal solution. To address this issue, we can explore several options in parallel – which is precisely what Widening is all about. How those parallel solution candidates are picked is the interesting question, which we will address later, but let us first look into widening itself in a bit more detail. Using the notation introduced above, one iteration of Widening can be represented as follows:

$$M' = \{m'_1, \ldots, m'_k\} = s_{\text{widened}} \left(\bigcup_{m \in M} r(m) \right).$$

That is, at each step, the *widened* selection operator s_{widened} considers the refinements of a set M of original models and returns a new set M' of k refined models for further investigation. We will refer to parameter k as the *width* of the widened search. Intuitively, it is clear that the larger the width, i.e. the more models (and hence selection paths) are explored in the solution space, the higher our chances are of finding a better model in comparison to the normal greedy search. Figure 2 illustrates this process.

An easy implementation of the above (what we will later refer to as top-k Widening) is a beam search. Instead of following one greedy path, the path of k best solution candidates is explored. However, this does not guarantee that we are indeed exploring alternative models – on the contrary, it is highly likely that we are exploring only closely related variations of the locally best model.

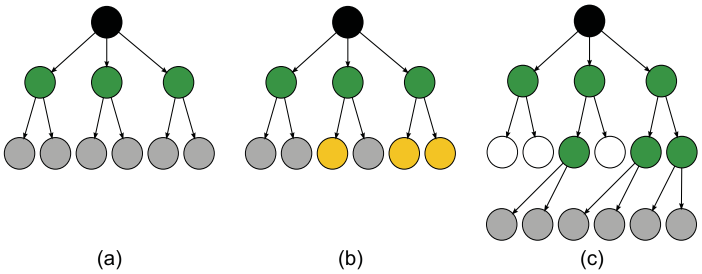

Fig. 2. Widening. From a set of models M (green circles), the refinement operator creates several sets of models (gray), shown on the left (a). The selection now picks a subset of the refined models (yellow circles in (b)) and the search continues from these on the right (c).

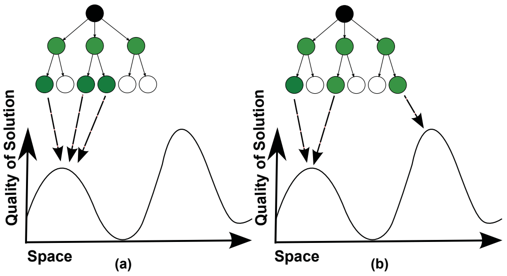

Fig. 3. Normal Widening may lead to local exploitation only (a). Adding diversity constraints encourages broader exploration of the model space (b).

In the area of genetic algorithms this effect is known as *exploitation*, that is, we are essentially fine tuning a model in the vicinity of an (often local) optimum.

3.2 Diversity-Driven Widening

In order to avoid the local exploitation, as discussed above, we can add diversity constraints which enforces the search to more broadly investigate our search space. In genetic algorithms this is often called *exploration*. Figure 3 illustrates the difference between local exploitation and global exploration. This effect has also been shown to improve results quite considerably for other beam search type problems, as we briefly discussed in Section 2. By forcing the parallel workers to consider not only multiple selection paths, but also *diverse* ones simultaneously, we aim to obtain better exploration of the search space and escape entrapment by local optima. Techniques for this type of Diversity-driven Widening are the main focus of this paper.

4 Techniques for (Diversity-Driven) Widening

In this section we will describe several specific techniques for Widening. We start by establishing the base *top-k Widening*, describe the diversity-driven version, and then focus on approaches that require less or no communication effort.

Top-*k* Widening. In [1] we already described this approach to Widening. It is the most obvious approach and identical to a classic beam search. In each iteration of top-k Widening each parallel worker selects the top k choices for the refinements of its model and from the resulting k^2 choices, the top k are chosen:

$$\{m'_1, \ldots, m'_k\} = s_{\text{top-}k}\left(\bigcup_{i=1,\ldots,k} s_{\text{top-}k}\left(r(m_i)\right)\right)$$

where $s_{\text{top-}k}$ selects the top k models from a set of models according to the given quality measure ψ. Obviously, $s_{\text{top-1}} = s_{\text{best}}$.

In [1] we demonstrate that top-k Widening leads to an improved quality, with larger *width k* leading to better accuracy. However, two main flaws exist. The first problem, as mentioned above already, is that possibly only a small neighborhood of the best solutions is explored. Secondly, continuous communication is required between threads which contradicts our goal of wanting to keep the time constant.

Diverse Top-*k* Widening. As discussed above we can tackle the first flaw of Top-k Widening by enforcing diversity. One simple way to add diversity can be achieved by using a fixed diversity threshold θ, a distance function δ, and by modifying the selection operator $s_{\text{top-}k,\delta}$ to iteratively pick the best k refinements, that satisfy the given diversity threshold. This can be summarized as follows:

```
$M_{\text{all}} = \cup_{i=1,\ldots,k} r(m_i)$                 create set of all possible refinements
$m_1 = \arg\max_{m \in M_{\text{all}}} \{\psi(m)\}$          pick the locally optimal model as first model
$M_1 = \{m_1\}$                                             add as initial model to solution
for $i = 2, \ldots, k$:                                     iteratively pick next, sufficiently diverse model:
  $m_i = \arg\max_{m \in M_{\text{all}}} \{\psi(m) \mid \neg\exists m' \in M_{i-1} : \delta(m, m') < \theta\}$
  $M_i = M_{i-1} \cup \{m_i\}$
endfor
return $M_k$
```

This is a known approach for diverse subset picking, however, our second problem persists: we still require frequent communication among our parallel workers to make sure we pick a diverse solution subset among all intermediate solutions at each iteration.

Communication-Free Widening. In order to achieve communication-free Widening, we must force each worker to focus on its own subset of models without continuously synchronizing this with the other workers. Ideally, communication-free Widening is achieved via partitioning the model search space between the parallel workers in such a way, that the full search space is covered by the partitioning,

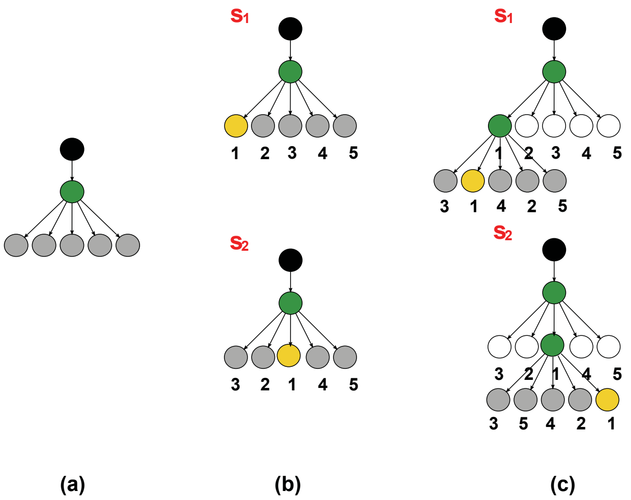

Fig. 4. Communication-free Widening: two different selection paths generated by two different selection operators s_1 and s_2.

every model is reachable in at least one partition, and there is no overlap between different partitions. Even more ideally, those partitions contain sufficiently diverse solutions. However, those objectives are difficult to meet in practice. Our current goal is more modest: we aim to enforce diverse selection paths to be explored by the different parallel workers, which will hopefully lead to diverse final solutions. We approach this goal indirectly, by assigning a modified quality measure ψ_i to each selection operator s_i (of worker i), which, when given a choice, has a personalized preference for a (different) subset of models. If this individualized assignment is properly implemented, each parallel worker i will explore a different selection path $\mathcal{P}_{s_i}$ if refinements with sufficiently similarly high quality exist.

Figure 4 illustrates two different selection paths generated by two different selection operators s_1 and s_2. Our goal that each individualized selection operator explores a different and diverse path through the search space can be achieved by a modification of the selection operators which we describe below.

Diverse Communication-Free Widening. As described above, we need to ensure that the explored selection paths by the parallel workers $i = 1, \ldots, k$ are *sufficiently diverse*. To increase the chances for different parallel workers to explore diverse paths, we modify the k quality measures ψ_i of operator s_i so that each ψ_i assigns different preferences for the models in the search space. In the most subtle case, this will only break ties between models differently when we have more than one refinement with the (locally) optimal quality. However, to achieve real exploration, we will also want to lift slightly worse models above better ones *for some* of our workers but we need to ensure that at least one worker still investigates the locally optimal choice. It is important to note that while we want to explore different and diverse selection paths, we also do not want to focus solely on diversity. Random diverse exploration may lead to investing resources in the discovery of many degenerate solutions. We will need to balance between the two notions "diversity" and "quality". This trade-off was already visible in the case of the diverse top-k approach discussed above, where the threshold θ determines how much the selection operator $s_{\text{top-}k,\delta}$ is influenced by diversity and how much by the quality of the remaining solutions.

Enforcing diversity by modifying the underlying quality measures tends to be fairly algorithmic specific. Generally, the quality measures can assign diverse preferences directly to models (or parts of models), which we call *model-driven diversity* or by assigning preferences based on data points, which we term *data-driven diversity.* We will show examples for these approaches in the following section.

5 Diversity-Driven Widening of Set Covering

Various data mining problems employ strategies that are similar to the set cover problem. We have already used this algorithm in [1] to illustrate the benefits of Widening and will use it here as well.

5.1 The Set Cover Problem

We consider the standard (unweighted) set cover problem. Given a universe X of n items and a collection $\mathcal{S}$ of m subsets of $X : \mathcal{S} = \{S_1, S_2, \ldots, S_m\}$. We assume that the union of all of the sets in $\mathcal{S}$ is X: $\bigcup_{S_i \in \mathcal{S}} S_i = X$. The aim is to find a sub-collection of sets in $\mathcal{S}$, of minimum size, that contain ("cover") all elements of X.

The standard iterative algorithm [29] follows a greedy strategy, which, at each step, selects the subset with the largest number of remaining uncovered elements. Using the formalizations introduced above, a single iterative step of the algorithm operates as follows: if m is the temporary cover, a refinement generated by $r_{\text{greedySCP}}(m)$ represents the addition of a single subset, not yet part of m, to m. From all of the possible refinements, generated by $r_{\text{greedySCP}}(m)$, $s_{\text{greedySCP}}$ picks the one with the largest number of covered elements as the new intermediate cover. The quality measure ψ, used by the selection operator, $s_{greedySCP}$, therefore simply ranks the models based on the number of elements they cover.

5.2 Diversity-Driven Widening of Set Covering

In the following we will discuss how we can use the widening strategies described above for the greedy algorithm for the set cover problem. Note that our goal here is not to outperform other algorithmic improvements of the standard greedy set covering algorithm but instead use this to illustrate the benefits of Widening itself.

Top-k Widening and Diversity. Instead of selecting one locally best intermediate cover, the top-k Widening of the greedy SCP algorithm selects k best covers at each given step. To implement diversity, we can use a a simple threshold based on the Jaccard distance and enforce that the chosen k intermediate covers chosen by the selection operator $s_{\text{top-}k,\delta}$ at each step have a minimum distance:

$$\delta(m_i, m_j) = 1 - \frac{|m_i \cup m_j|}{|m_i \cap m_j|}.$$

(Each model m covers a set of elements, so we are interested in picking intermediate models that are sufficiently different.)

Communication-Free Widening: Model-Driven Diversity. Enforcing diversity without continuously comparing intermediate models is more difficult. We can define individual quality measures ψ_i, by enforcing different preferences for different subsets. Given an intermediate cover m, ψ_i evaluates the refinement $m' = m \cup S_j$ for an additional subset S_j based on the original quality measure and an individual preference weight $w_i \in (0, 1)$ for the subset S_j:

$$\psi_i(m \cup S_j) = \psi(m \cup S_j) + t * w_i(S_j),$$

The set of weights $w_i(\cdot)$ for a given ψ_i defines an order π_i on the set of subsets $\mathcal{S}$ for a particular parallel worker i:

$$w_i(S_{\pi_i(1)}) > \cdots > w_i(S_{\pi_i(|\mathcal{S}|)})$$

Our goal is to have k diverse orders $\pi_1, \ldots, \pi_k$ of the subsets by ensuring that the *inversion distances* between different orders are maximized. The inversion distance between two ordered sets calculates how many pairs of elements are present in a different order in the two orders π_p and π_q:

$$d_{\text{inv}}(\pi_p, \pi_q) = \sum_{k \neq l} \begin{cases} 1 & \text{if } (\pi_p(k) - \pi_p(l)) \cdot (\pi_q(k) - \pi_q(l)) < 0 \\ 0 & \text{else} \end{cases}.$$

Assigning preferences in this fashion will steer the selection operators based on characteristics of the models (or model fragments), hence the term *model-driven diversity.*

By varying parameter t, we can control how much the selection paths of the parallel workers deviate from the selection paths explored by the greedy SCP algorithm. The parameter t controls the relative importance of the factors quality and diversity. For parameter $t \leq 1$, the parallel workers explore different selection paths of the greedy algorithm, only considering different paths that have equally good, local quality. Here the different orders π_i only serve to differentiate the tie breaking. For parameter values $t > 1$, the selection paths of the parallel workers also include locally sub-optimal solutions. Large values of t ($\gg 1$) will lead to random exploration of the search space.

Communication-Free Widening: Data-Driven Diversity. In contrast to the model (fragment) driven diversity described above, we can also ensure diversity by weighting data elements. To accomplish this we enforce diverse preferences for the elements from X for the different selection operators s_i:

$$\psi_i(m \cup S_j) = \psi(m \cup S_j) + t \cdot \frac{1}{|\{e \in S_j \wedge e \notin m\}|} \sum_{e \in S_j \wedge e \notin m} w_i(e),$$

the preference for different elements is again defined via weights $w_i(e)$ and the weights define an ordering on the elements where we again aim for k different orderings via sufficient inversion distance. Note that this approach bears some similarities to boosting because we weight the impact of data elements on the model quality measure differently.

It must be noted, that, while using diverse quality measures can help steer the parallel workers into diverse selection paths, it by no means guarantees it. Choosing different models at each step can still lead to having the same final solution, just generated along a different paths. Implementing selection in such a way that diversity of the obtained final solutions is guaranteed is an interesting focus of future work. In the following section, however, we will demonstrate that regardless of the lack of theoretical guarantees, our simple approaches to Diversity-driven Widening are beneficial.

6 Experimental Evaluation and Discussion

In this section we demonstrate the impact of the Widening techniques discussed above using three benchmark data sets: $rail507$, $rail516$, and $rail582$ [30]. We aim to demonstrate how different widths of the search affect the quality of the solution and the additional benefit of enforcing diversity on the widened searches. Each experiment was repeated 50 times. For diverse top-k Widening, a Jaccard distance threshold of 0.01 was used in all experiments.

Figure 5 shows the results for top-k Widening with and without diversity. Figure 6 shows the results for communication-free Widening without diversity

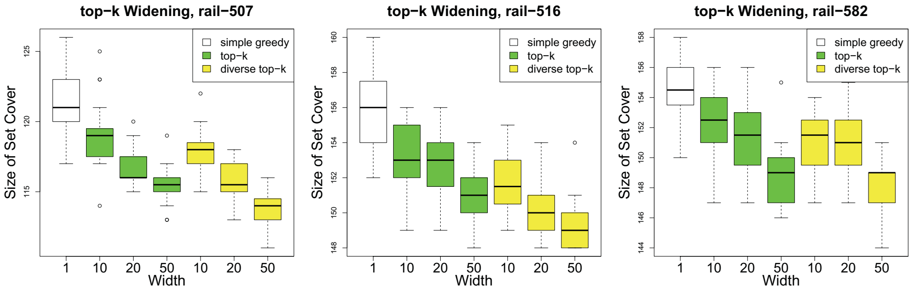

Fig. 5. Results from the evaluation top-k Widening with and without diversity

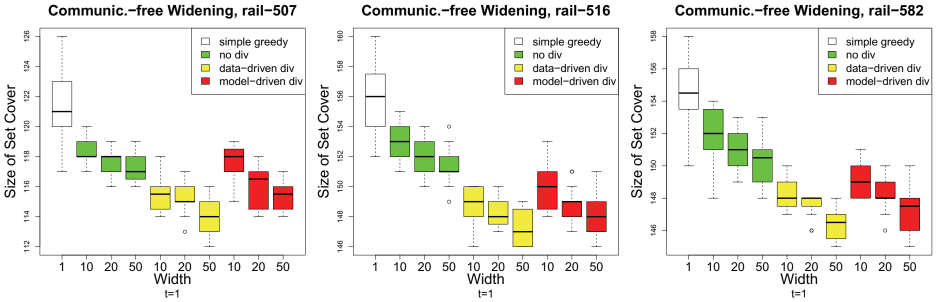

Fig. 6. Results from communication-free Widening without diversity, with data-driven and model-driven diversity, for parameter $t = 1$

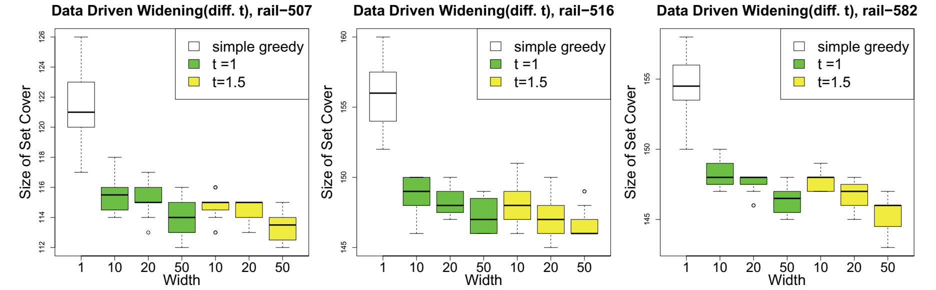

Fig. 7. Results from communication-free Widening using data-driven diversity for different values of parameter t

and with data- resp. model-driven diversity enforcement using a fixed trade-off parameter $t = 1$, while Figure 7 demonstrates the impact of trade-off parameter t for data-driven diversity.

From the above results two main trends become clear. As expected, a larger width of the search does improve the quality of the solution. Enforcing diversity improves the results even further. For communication-free Widening, the first set of tests simply enhances the greedy algorithm by exploring different options when breaking ties in-between equally good intermediate solutions. By increasing parameter t to $t = 1.5$ the widened algorithm is allowed to also explore paths of non-locally optimal choices, which further improves the results. The optimal value for parameter t depends heavily on the dataset, and if fine-tuning is applied, more improvement can be expected. Obviously, if t is too large, this will turn the algorithm into an almost data-independent, random search process, deteriorating solution quality again.

7 Conclusions and Future Work

We continued earlier work on the impact of Widening on data mining algorithms. A number of practical techniques to implement Widening focusing on reduction of communication and enhancing the exploration of the solution space were presented. The latter shows promise to further increase the accuracy of Widening as we have demonstrated using the base greedy set covering algorithm. Focusing on better ways to enforce diversity without the need for extensive communications is an area of future work, as is the application of the presented techniques to other data mining algorithms.

Acknowledgements. We thank Arno Siebes and Oliver Sampson for many interesting discussions and constructive feedback. V. Ivanova is supported by the DFG under grant GRK 1042 (Research Training Group "Explorative Analysis and Visualization of Large Information Spaces").

References

1. Akbar, Z., Ivanova, V.N., Berthold, M.R.: Parallel data mining revisited. Better, not faster. In: Hollmén, J., Klawonn, F., Tucker, A. (eds.) IDA 2012. LNCS, vol. 7619, pp. 23–34. Springer, Heidelberg (2012)
2. Akl, S.G.: Parallel real-time computation: Sometimes quantity means quality. Computing and Informatics 21, 455–487 (2002)
3. Kumar, V.: Special Issue on High-performance Data Mining. Academic Press (2001)
4. Kargupta, H., Chan, P.: Advances in Distributed and Parallel Knowledge Discovery. AAAI/MIT Press (2000)
5. Zaki, M.J., Ho, C.-T. (eds.): KDD 1999. LNCS (LNAI), vol. 1759. Springer, Heidelberg (2000)
6. Zaki, M.J., Pan, Y.: Introduction: Recent developments in parallel and distributed data mining. DPD 11(2), 123–127 (2002)
7. Shafer, J., Agrawal, R., Mehta, M.: Sprint: A scalable parallel classifier for data mining. In: VLDB, pp. 544–555 (1996)
8. Zaki, M.J., Ho, C.-T., Agrawal, R.: Parallel classification for data mining on shared-memory multiprocessors. In: ICDE, pp. 198–205 (1999)
9. Darlington, J., Guo, Y.-K., Sutiwaraphun, J., To, H.W.: Parallel induction algorithms for data mining. In: Liu, X., Cohen, P., Berthold, M. (eds.) IDA 1997. LNCS, vol. 1280, pp. 437–445. Springer, Heidelberg (1997)
10. Srivastava, A., Han, E.-H., Kumar, V., Singh, V.: Parallel formulations of decision-tree classification algorithms. DMKD 3(3), 237–261 (1999)
11. Kufrin, R.: Decision trees on parallel processors. In: PPAI, pp. 279–306 (1995)
12. Zaki, M.J.: Parallel and distributed association mining: a survey. IEEE Concurrency 7(4), 14–25 (1999)
13. Judd, D., McKinley, P.K., Jain, A.K.: Large-scale parallel data clustering. TPAMI 20(8), 871–876 (1998)
14. Dhillon, I., Modha, D.: A data-clustering algorithm on distributed memory multiprocessors. In: Large-scale Parallel KDD Systems Workshop, ACM SIGKDD, pp. 245–260 (2000)
15. Olson, C.F.: Parallel algorithms for hierarchical clustering. JPC 21, 1313–1325 (1995)
16. Garg, A., Mangla, A., Gupta, N., Bhatnagar, V.: PBIRCH: A scalable parallel clustering algorithm for incremental data. In: IDEAS, pp. 315–316 (2006)

17. Dean, J., Ghemawat, S.: MapReduce: simplified data processing on large clusters. Commun. ACM 51(1), 107–113 (2008)
18. Chu, C.-T., Kim, S.K., Lin, Y.-A., Yu, Y.Y., Bradski, G.R., Ng, A.Y., Olukotun, K.: Map-reduce for machine learning on multicore. In: NIPS, pp. 281–288 (2006)
19. Ma, Z., Gu, L.: The limitation of MapReduce: A probing case and a lightweight solution. In: Intl. Conf. on Cloud Computing, GRIDs, and Virtualization, pp. 68–73 (2010)
20. Breiman, L.: Bagging predictors. JML 24(2), 123–140 (1996)
21. Schapire, R.E.: The strength of weak learnability. JML 5, 28–33 (1990)
22. Breiman, L.: Random forests. JML 45(1), 5–32 (2001)
23. Talia, D.: Parallelism in knowledge discovery techniques. In: Fagerholm, J., Haataja, J., Järvinen, J., Lyly, M., Råback, P., Savolainen, V. (eds.) PARA 2002. LNCS, vol. 2367, pp. 127–136. Springer, Heidelberg (2002)
24. Shell, P., Rubio, J.A.H., Barro, G.Q.: Improving search through diversity. In: AAAI (1994)
25. Harvey, W.D., Ginsberg, M.L.: Limited discrepancy search. IJCAI, 607–615 (1995)
26. Felner, A., Kraus, S., Korf, R.E.: KBFS: K-best-first search. AMAI 39(1-2), 19–39 (2003)
27. Berger, B., Rompel, J., Shor, P.W.: Efficient nc algorithms for set cover with applications to learning and geometry. JCSS 49(3), 454–477 (1994)
28. Blelloch, G.E., Peng, R., Tangwongsan, K.: Linear-work greedy parallel approximate set cover and variants. In: SPAA, pp. 23–32 (2011)
29. Johnson, D.S.: Approximation algorithms for combinatorial problems. In: STOC, pp. 38–49 (1973)
30. Beasley, J.E.: Or-library: Distributing test problems by electronic mail. The Journal of the Operational Research Society 41(11), 1069–1072 (1990)