Active Learning for Abstract Models of Collectives*

Alexander Schiendorfer†, Christoph Lassner‡, Gerrit Anders†, Wolfgang Reif†, and Rainer Lienhart‡
Institute for Software & Systems Engineering†
Multimedia Computing and Computer Vision Lab‡
University of Augsburg, Germany
Email: {alexander.schiendorfer, christoph.lassner, anders, reif, lienhart}@informatik.uni-augsburg.de

***Abstract*—Organizational structures such as hierarchies provide an effective means to deal with the increasing complexity found in large-scale energy systems. In hierarchical systems, the concrete functions describing the subsystems can be replaced by abstract piecewise linear functions to speed up the optimization process. However, if the data points are weakly informative the resulting abstracted optimization problem introduces severe errors and exhibits bad runtime performance. Furthermore, obtaining additional point labels amounts to solving computationally hard optimization problems. Therefore, we propose to apply methods from active learning to search for informative inputs. We present first results experimenting with Decision Forests and Gaussian Processes that motivate further research. Using points selected by Decision Forests, we could reduce the average mean squared error of the abstract piecewise linear function by one third.**

I. Hierarchical Distributed Energy Management

Future energy systems move from systems of relatively few centrally organized units providing most of the power demanded by consumers to many highly distributed units, calling for manageable control mechanisms [1]. To deal with the resulting complexity in scheduling power plants in the face of uncertainties introduced by nature and technical deficiencies, hierarchical organizations based on virtual power plants that form autonomously can be employed [2], [3]. In our vision of future energy management systems [2], inner nodes of the hierarchy are called *autonomous* virtual power plants (AVPP) and act as intermediaries on behalf of their subordinates–thus, single nodes representing a *collective*. Power plants are thus structured into systems of systems represented by AVPPs, which can themselves be part of other AVPPs, as shown in Fig. 1. We model power plants and AVPPs as agents. To achieve a reduction of complexity in the optimization problem to be solved by the overall system when creating schedules, techniques are borrowed from model abstraction [4]. In particular, intermediaries approximate functional dependencies over a combinatorial input domain stemming from the aggregate of their underlying agents by repeatedly sampling input-output pairs and substituting the actual functions by piecewise linear functions [5].

In general, the problem to be solved constitutes a hierarchical resource allocation problem [6]. The resource to be allocated to a set of agents maps to their scheduled contributions in order to meet a predicted demand over a scheduling window $\mathcal{W}$. This window consists of finitely many time steps with a fixed resolution of typically 15 minutes. Agents have to act proactively, i.e., create schedules, since they are subject to inertia and cannot be assumed to react fast enough in case of rapidly increasing (or decreasing) demand. With regard to the case study, we derive the minimal set of constraints from the physical requirements that power plants impose (see [7] for a discussion of the literature): i) a minimal and maximal power boundary, ii) discontinuity given the ability to be switched off, and iii) functions limiting the possible change in production over a certain period of time. The latter might not only depend on the type of an agent but also on its current contribution.

From these physical constraints, we abstract minimal and maximal contributions and switching on and off to a sorted list of *feasible intervals* L_a. A power plant a that is capable of being switched off or running between some boundaries $P_{\min}$ and $P_{\max}$ would then, for instance, be represented by $L_a = \langle [0,0], [P_{\min}, P_{\max}] \rangle$. To allow planning for inertia in a, we introduce functions $\overrightarrow{A}_a^{\min}$ and $\overrightarrow{A}_a^{\max}$ that return the minimum and maximum contribution in a following time step given the current contribution c. In the simplest case, we consider a constant maximal change ΔP:

$$\overrightarrow{A}_a^{\min}(c) \stackrel{\text{def}}{=} \max\{P_{\min}, c - \Delta P\} \quad (1)$$

$$\overrightarrow{A}_a^{\max}(c) \stackrel{\text{def}}{=} \min\{P_{\max}, c + \Delta P\} \quad (2)$$

But of course, these functions can model richer systems than that, e.g., consider a hot or cold start-up [7], or depend on the current contribution as well as rates of change that map combinatorially to the underlying agents [5] in case a itself is an intermediary. In addition to that, cost functions κ_a return the minimal costs incurred for a certain contribution.

We revisit the scheduling problem presented in [8] for some inner node—called intermediary λ—since the problem is solved top-down, as shown in Fig. 1. Each intermediary in turn redistributes its assigned fraction of the overall demand $S_\lambda[t]$ at a given time step t to its subordinate agents $\mathcal{A}_\lambda$ until schedules are assigned to all leaf agents, e.g., physical power plants. The allocation aims to minimize deviations from the demand (Δ) as well as the incurred costs (Γ). Relative importance is specified by the weights α_Δ and α_Γ. Note that the root node Λ is assigned the actual total demand of the environment, i.e.,

*This research is partly sponsored by the research unit *OC-Trust* (FOR 1085) of the German Research Foundation.

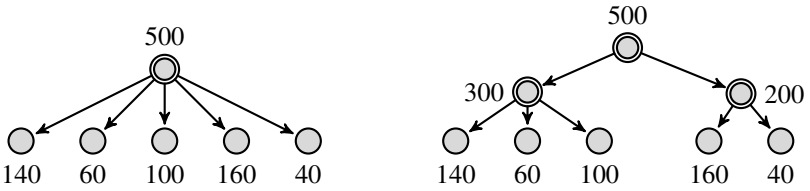


Fig. 1: A central and hierarchical solution example to a resource allocation problems. Inner nodes representing intermediaries are marked by double circles and redistribute their assigned share of an overall demand, e.g., power.

$S_\Lambda[t] = A_{env}[t]$.

$$\begin{aligned} \underset{S_a[t]}{\text{minimize}} \quad & \alpha_\Delta \cdot \Delta + \alpha_\Gamma \cdot \Gamma \qquad (3) \\ \text{subject to} \quad & \forall a \in \mathcal{A}_\lambda, \forall t \in \mathcal{W} : \\ & \exists [x,y] \in L_a : x \leq S_a[t] \leq y, \\ & \overrightarrow{A}_a^{\min}(S_a[t-1]) \leq S_a[t] \leq \overrightarrow{A}_a^{\max}(S_a[t-1]) \\ & \text{with } \Delta = \textstyle\sum_{t \in \mathcal{W}} \left| S_{\mathcal{A}_\lambda}[t] - S_\lambda[t] \right|, \\ & \text{and } \Gamma = \textstyle\sum_{t \in \mathcal{W}, a \in \mathcal{A}_\lambda} \kappa_a(S_a[t]) \end{aligned}$$

We propose two approaches based on self-organization for problem decomposition to solve this problem:

- A so-called "regio-central" approach in which agents transfer models to their intermediary which, at meso-level, centrally optimizes the allocation [5], [7].
- An auction-based decentralized approach [2] where agents need not submit their model but only bid on a given demand based on their private capabilities.

Obtaining a good abstraction of an intermediary's behavior as a compact representation of the underlying subordinate agents' combined behavior is desirable for both algorithms. In the regio-central case, one wants to simplify the resulting optimization problems by reducing the number of decision variables and constraints.

In the auction-based algorithm, an intermediary could, in principle, have all agents bid simultaneously to a single auctioneer, i.e., use a super-flat hierarchy. Clearly, this auctioneer imposes a bottleneck with a rising number of agents. In a truly hierarchical setting, an intermediary ought to be aware of the physical boundaries of its subordinate agents *before* submitting bids in order to avoid inconsistencies that need to be (monetarily) punished by the organization [2]. As a simple illustration, consider that an intermediary better not bid for a contribution greater than 200 if it is comprised of two underlying agents with a maximal contribution of 100 each. Even if the summation of maximal boundaries constitutes no computational effort, additional constraints from limited rates of change and disconnectability make the problem harder. In fact, an intermediary needs to solve an optimization problem quite similar to Eq. (3) in order to calculate bids for a given demand. Similarly to the "regio-central" approach, abstraction has further to be applied to obtain a simplified model of the intermediary that its own superior may use to compute bids.

II. Obtaining Abstracted Models

Having motivated the need for abstraction techniques, we briefly revisit our existing approach [5] to discuss improvements using active learning. An essential abstraction consists of finding the possible contributions of an intermediary by combining their lists of feasible intervals. Assume agent a_1 can contribute in $[x_1, y_1]$ and agent a_2 in $[x_2, y_2]$. Their combined contribution must be in $[x_1, y_1] \oplus [x_2, y_2] = [x_1 + x_2, y_1 + y_2]$. The operator $\oplus$ naturally extends to lists of intervals by combining in a Cartesian fashion and merging overlaps.

But now consider questions such as "What is the minimal cost for an intermediary to contribute x?" or "What is the maximal next contribution given the momentary state y"? Both are optimization problems as generally there are several configurations of subordinates to achieve joint contribution x at different costs. We acquire an abstraction of functional relationships by *sampling* the concrete function.

We illustrate this algorithm detailed in [5] with an exemplary intermediary v consisting of three agents $\mathcal{A}_v = \{p, q, r\}$:

$$L_p = \langle [0,0], [50,100] \rangle, \qquad \kappa^p = 13 \qquad (4)$$
$$L_q = \langle [0,0], [15,35] \rangle, \qquad \kappa^q = 70 \qquad (5)$$
$$L_r = \langle [0,0], [200,400] \rangle, \qquad \kappa^r = 5 \qquad (6)$$

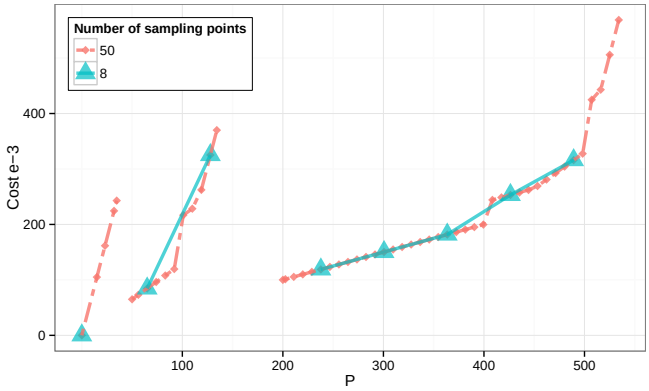
where κ^a is the price per production unit such that the agents' cost functions are defined by: $\kappa_a(x) = \kappa^a \cdot x$. We derive the feasible contribution ranges for v and get $L_v = \langle [0,0], [15,35], [50,135], [200,535] \rangle$. Sampling points are selected from this contribution range equidistantly leading to a sequence of optimization problems to be solved in order to find sampling values $\overline{\kappa}_v(P)$ storing the minimal costs to produce a given contribution P of the intermediary v:

$$\overline{\kappa}_v(P) = \min_{S_p, S_q, S_r} \sum_{a \in \mathcal{A}_v} \kappa_a(S_a) \text{ subject to } \sum_{a \in \mathcal{A}_v} S_a = P \quad (7)$$

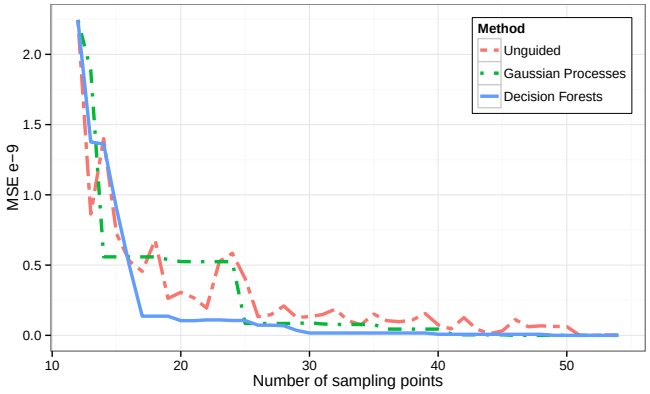
Concretely, we might consider a sequence $\langle \overline{\kappa}_v(15), \overline{\kappa}_v(25), \overline{\kappa}_v(35), \overline{\kappa}_v(50), \overline{\kappa}_v(75), \ldots, \overline{\kappa}_v(400) \rangle$ to collect sampling points. More sampling points typically correlate with higher accuracy, as Fig. 2a shows.

However, if these input-output pairs are selected in a weakly informative way, the resulting abstracted optimization problem introduces severe errors in quality as well as bad runtime performance as Fig. 2b (Unguided) illustrates where 25 equidistantly selected sampling points lead to a worse mean squared error (MSE) (with respect to a validation set consisting of 50 actually sampled points) than about 17 sampling points at more informative positions.

The quality issue is understandable due to unfortunate interpolations. The runtime problem is less obvious but stems from the way MIP-solvers resolve piecewise linear functions which might result in many integral decision variables or additional specially ordered sets that slow down the optimization. In addition and as shown in Eq. (7), determining each sampling point comes at the expense of solving an optimization problem



(a) Accuracy affected by the number of sampling points selected.



(b) MSE values for different sampling methods and point counts.

Fig. 2: Selecting the "right sampling points" is crucial for good accuracy. The cost function κ_v is only defined over the (discontinuous) domain $L_v = \langle [0,0], [15,35], [50,135], [200,535] \rangle$.

itself. Thus, it is desirable to avoid asking for uninformative points such as those between 200 and 400 in Fig. 2a. Even if the sampling points are already obtained, removing redundant points from the piecewise linear functions that enter the overall optimization, i.e., scheduling power plants as described in Eq. (3) is beneficial.

III. Improving the Sampling Point Selection by Active Learning

However, obtaining a sampling point is very costly in terms of time. The overall optimization problem must be solved within 15 minutes, which imposes a strict time-limit. Active learning can be used to tackle this issue. The active learning meta-algorithm 'uncertainty sampling' creates a model with few sampling points and iteratively refines it with additional points in 'interesting' areas (areas with high uncertainty). This technique can be applied with many regressors (e.g., Gaussian Processes (GPs) [9], [10], [11] or Decision Forests (DFs) [12]). A survey on active learning is presented in [13].

Of these options, DFs with linear models as leaf models [14] provide the best inductive bias for the given task: they quickly develop high confidence for areas of linear continuities and low confidence at other points. This closely corresponds to the aim of building a piecewise linear function from the sampled data points.

Each linear model at a tree leaf estimates $p(y \mid x)$ as

$$p(y \mid x) = \mathcal{N}\left(x^T b,\ s^2 x^T \left(X^T X\right)^{-1} x\right), \tag{8}$$

where b are the linear model parameters estimated by the samples reaching the leaf, s is their expected measurement error and X is their design matrix. The expected variance of this normal distribution is therefore sensitive to the distance to the nearest training data point as well as to the sample coherence at the leaf. Hence, it can provide good confidence estimates at points of discontinuity (c.t. [14, p. 58]). However, we noticed that instead of combining the estimated variances of the leafs of each tree to a single variance of a multimodal Gaussian distribution, averaging them produced better results in our experiments.

A comparison of confidence estimates of GPs and DFs is given in Fig. 3b. The confidence estimation of the GPs is only dependent on the closeness to the next sampling point, whereas the DFs include the coherence of the observed training data as well. This is reflected by the high uncertainty in the range of about 15 to 100 in the DF model which contrasts the GP's estimation in this unsteady region.

IV. Evaluation

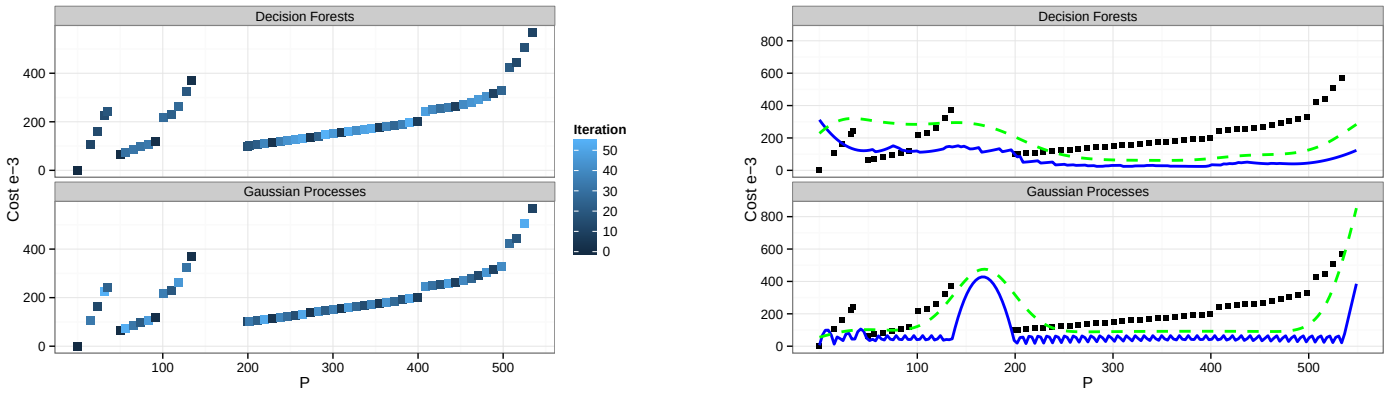
To demonstrate the effectiveness of the approach, we applied it in the scenario described in Section II. We experimented with the GPy[1] implementation of GPs and the Fertilized Forest library of DFs [15]. The full code can be found online. [2] Each method started with 12 equally distributed sampling points to train an initial regressor. Then we incrementally added the point for which the learners' uncertainty estimate was highest and retrained the regressor.

As uncertainty estimate for determining the next point, we did not use the σ^2 estimate of the learner directly, but first applied a convolution with a Gaussian mask (see Fig. 3b). The variance estimate of the DF learner has sharp borders at leaf borders of single trees. The convolution smoothes the borders and combines information of the surrounding area to find more relevant sampling points. Even though the sharp borders are not present in the estimate of the GP learner, the smoothing produced better results for that learner, too.

As evaluation measure for the regression with a fixed amount of data points, we used the standard MSE. However, the performance of the system over its entire trace (all amounts of data points from 12 up to 54) must somehow be integrated. We experimented with a weighted mean with decreasing weights for higher amounts of data points, but found that this resulted in unfair evaluations due to too low weights at the end of the trace. This is why we used the mean MSE (MMSE) to compare overall system performance. This allowed us to adjust each learners' parameters in a grid search, including the kernel width of the Gaussian convolution.

In Fig. 2b, the resulting error traces of an unguided system (using equally distributed points) compared to the results of the guided systems can be found. With only 20 sampling

[1] Homepage: http://sheffieldml.github.io/GPy/.
[2] Experiment repository: http://git.io/buNW



(a) Point selection order: The lighter the point color, the later it was selected by the algorithm.

(b) Smoothed (green, dashed) and smoothened (blue, solid) variance estimations by the two methods given 50 observed points (squares).

Fig. 3: Query point selection by the two evaluated methods.

points, the system guided by DFs reaches the performance of the unguided system with 34 sampling points. The system guided by GPs reaches the same performance with 25 points, but at more than double the runtime of the DF-guided system (approx. a factor of 2.24). The mean MSE (scaled with 10^{-9}) for the entire trace is 0.28 (unguided), 0.26 (GP-guided) and 0.18 (DF-guided).

Figure 3a visualizes the order in which points have been added to the training set by the two learners. The lighter the point color, the later it was added to the training set. The ideal selector selects points at discontinuities to the training set early and points in areas of linear change last. The DF-guided selection shows this behavior stronger than the GP-guided selection as it refrains from selecting points in the linear area around 200 to 400.

V. Conclusion

We motivated the necessity for and the requirements of creating abstract models of collectives in the context of energy management. By analyzing the constraints of the optimization problem, we propose active learning as a natural extension to our approach. In our first experiment, we analyzed the possibilities to use Gaussian Processes and Decision Forests in this context and introduced an error measure to be able to evaluate both methods. Results indicate that both methods can be used to achieve significant improvements. The properties of Decision Forests, however, make them the most promising candidate for this task. Future work consists of validating our results for different compositions of intermediaries, integrating our method into the optimization of the overall system, and evaluating the effectiveness of the resulting abstractions in terms of quality and runtime.

References

[1] S. D. Ramchurn, P. Vytelingum, A. Rogers, and N. R. Jennings, "Putting the 'Smarts' Into the Smart Grid: A Grand Challenge for Artificial Intelligence," *Commun. ACM*, vol. 55, no. 4, Apr. 2012.

[2] G. Anders, A. Schiendorfer, F. Siefert, J.-P. Steghöfer, and W. Reif, "Cooperative Resource Allocation in Open Systems of Systems," *ACM Trans. Auton. Adapt. Syst.*, 2015.

[3] A. Nieße, S. Beer, J. Bremer, C. Hinrichs, O. Lunsdorf, and M. Sonnenschein, "Conjoint Dynamic Aggregation and Scheduling Methods for Dynamic Virtual Power Plants," in *Proc. 3rd Int. Wsh. Smart Energy Networks & Multi-Agent Systems (SEN-MAS'14)*, 2014, pp. 1505–1514.

[4] F. Frantz, "A Taxonomy of Model Abstraction Techniques," in *Simulation Conference Proceedings, 1995. Winter*, 1995, pp. 1413–1420.

[5] A. Schiendorfer, J.-P. Steghöfer, and W. Reif, "Synthesis and Abstraction of Constraint Models for Hierarchical Resource Allocation Problems," in *Proc. 6th Int. Conf. Agents and Artificial Intelligence (ICAART'14), Vol. 2*. SciTePress, 2014, pp. 15–27.

[6] T. Van Zandt, "Hierarchical Computation of the Resource Allocation Problem," *European Economic Review*, vol. 39, no. 3-4, pp. 700–708, April 1995.

[7] A. Schiendorfer, J.-P. Steghöfer, and W. Reif, "Synthesised Constraint Models for Distributed Energy Management," in *Proc. 3rd Int. Wsh. Smart Energy Networks & Multi-Agent Systems (SEN-MAS'14)*, 2014, pp. 1529 – 1538.

[8] G. Anders, A. Schiendorfer, J.-P. Steghöfer, and W. Reif, "Robust Scheduling in a Self-Organizing Hierarchy of Autonomous Virtual Power Plants," in *Proc. 2nd Int. Wsh. Self-optimisation in Organic and Autonomic Computing Systems (SAOS'14)*, W. Stechele and T. Wild, Eds., 2014, pp. 1–8.

[9] S. Seo, M. Wallat, T. Graepel, and K. Obermayer, "Gaussian Process Regression: Active Data Selection and Test Point Rejection," in *Proc. Int. Jnt. Conf. Neural Networks (IJCNN'00)*, vol. 3, 2000, pp. 241–246.

[10] A. Krause and C. Guestrin, "Nonmyopic Active Learning of Gaussian Processes: An Exploration-Exploitation Approach," in *Proc. 24th Int. Conf. Machine Learning (ICML'07)*. ACM, 2007, pp. 449–456.

[11] M. Park, G. Horwitz, and J. W. Pillow, "Active learning of neural response functions with Gaussian processes," in *Advances in Neural Information Processing Systems (NIPS)*, J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K. Weinberger, Eds. Curran Associates, Inc., 2011, pp. 2043–2051.

[12] J. E. Iglesias, E. Konukoglu, A. Montillo, Z. Tu, and A. Criminisi, "Combining Generative and Discriminative Models for Semantic Segmentation of CT Scans via Active Learning," in *Proc. 22nd Inf. Conf. Information Processing in Medical Imaging (IPMI)*, 2011.

[13] B. Settles, "Active learning literature survey," University of Wisconsin–Madison, Computer Sciences Technical Report 1648, 2010. [Online]. Available: http://burrsettles.com/pub/settles.activelearning.pdf

[14] A. Criminisi, J. Shotton, and E. Konukoglu, "Decision Forests for Classification, Regression, Density Estimation, Manifold Learning and Semi-Supervised Learning," Microsoft Research, Tech. Rep. MSR-TR-2011-114, Oct 2011. [Online]. Available: http://research.microsoft.com/apps/pubs/default.aspx?id=155552

[15] C. Lassner and R. Lienhart, "Norm-induced entropies for decision forests," in *Proc. IEEE Winter Conference on Applications of Computer Vision 2015 (WACV'15)*, 2015, to appear. [Online]. Available: http://www.fertilized-forests.org