Active Learning for Model Abstraction*

Alexander Schiendorfer*, Christoph Lassner†, 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, reif}@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 that results from uncertainties in nature as well as computational efforts in scheduling. Abstraction-based methods provide a way to calculate a simpler behavior model to be used in optimization in lieu of a combination of a set of behavior models. In particular, functional dependencies over the combinatorial domain are approximated by repeatedly sampling input-output pairs and substituting the actual function by piecewise linear functions. However, if the selected input-output pairs are weakly informative, the resulting abstracted optimization problem introduces severe errors in quality as well as bad runtime performance. This problem is reminiscent of the task of selecting the next most informative input for supervised learning algorithms in case labeled input is rare. We therefore propose to apply methods from active learning based on decision trees for regression to search for informative input candidates to sample and present preliminary results that motivate further research.**

Keywords—Active Learning, Abstraction, Optimization, Holarchies, Hierarchical System, Decision Trees

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 and controlling 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]. Inner nodes of the hierarchy are called *autonomous* virtual power plants (AVPP) and act as intermediaries on behalf of their subordinate agents. Prosumers are thus structured into systems of systems represented by AVPPs, which can themselves can be part of other AVPPs, as shown in Fig. 1. To achieve a reduction of complexity in the optimization problem to be solved by the overall system, techniques are borrowed from model abstraction [4]. In particular, functional dependencies over a combinatorial input domain stemming from the aggregate of underlying agents are approximated 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], where 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 $\mathcal{W}$ consisting of finitely many time steps with a fixed resolution of 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. We derive the minimal set of constraints from the physical requirements that power plants impose (see [7] for a discussion of the literature):

- a minimal and maximal power boundary
- discontinuity given the ability to be switched off
- functions limiting the possible change in production over a certain period of time.

The latter function might depend on the type of an agent as well as the 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^t. A power plant a that is capable of being switched off or run between some boundaries $P_{\min}$ and $P_{\max}$ would then for instance be represented by $L_a^t = \langle [0,0], [P_{\min}, P_{\max}] \rangle$. 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. In the simplest case—we consider a constant maximal change ΔP—these functions are defined as:

$$\overrightarrow{A}_a^{\min}(x) \overset{\text{def}}{=} \max\{P_{\min}, x - \Delta P\}$$
$$\overrightarrow{A}_a^{\max}(x) \overset{\text{def}}{=} \min\{P_{\max}, x + \Delta P\}$$

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

We present the scheduling problem 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 $S_\lambda[t]$ to its subordinate agents $\mathcal{A}_\lambda$ until all leaf agents, i.e., physical power plants, are assigned schedules. Note that the root node Λ is assigned the

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

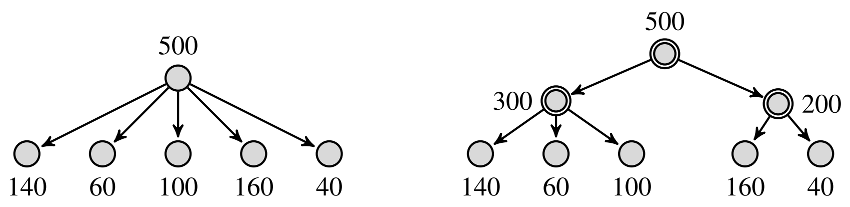


Fig. 1: Resource allocation problems can be solved using a hierarchical decomposition structure. Inner nodes representing intermediaries are marked by double circles.

actual total demand of the environment, i.e., $S_\Lambda[t] = A_{env}[t]$.

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

We propose to solve this problem using two approaches based on self-organizing hierarchies for problem decomposition:

- A so-called "regio-central" approach: agents transfer models to their local supervisor who, 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) individual 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 regional-central case, one wants to simplify the resulting optimization problems, in particular by reducing the decision variables and constraints to be considered. 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. When installing a hierarchical organization for scalability, 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 the fact that an intermediary better not bid for a contribution greater than 200 if it is comprised of two underlying physical agents with a maximal contribution of 100 each. Even if the mere summation of maximal boundaries constitutes no computational effort, the possible trajectories resulting 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. 1 in order to calculate bids for a given demand. Thus, abstraction has to be applied to obtain a simplified model of the intermediary that its own superior may use to compute bids.

II. Obtaining Abstracted Models

As we have now established the need for effective abstraction techniques, we briefly revisit our existing approach to discuss improvements using active learning.

A. Interval Calculations and Sampling Abstraction

A rather straightforward and first essential abstraction in this problem is obtained by considering 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 2 in $[x_2, y_2]$. Clearly, 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 the intervals in a Cartesian fashion and merging the overlapping intervals. That way, one will find that the combined production of two agents a_1, a_2 with $L^{a_1} = \langle [0,0], [1,4] \rangle$ and $L^{a_2} = \langle [0,0], [7,10] \rangle$ will be either of $\langle [0,0], [1,4], [7,14] \rangle$.

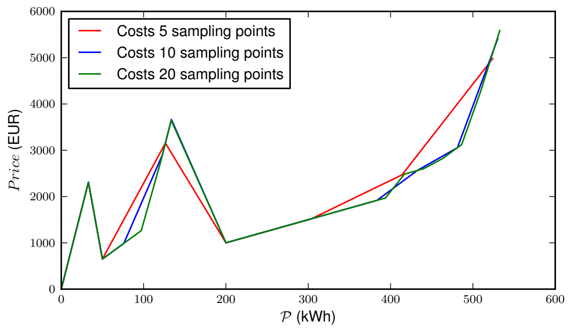
But now consider questions such as "What is the minimal cost for an aggregate to contribute x?" or "What is the maximally possible next contribution given the momentary state is y"? Both involve solving an optimization problem since there can naturally be many configurations of subordinate agents leading to a contribution of x with different costs. To address these issues, we acquire an abstract representation of functional relationships by *sampling*, i.e., solving several optimization problems and collecting input and output values (e.g., contribution-cost pairs). Underlying constraints C and decision variables (X, D) of the sampling problems are kept identical of the optimization problem in Eq. 1.

Algorithm 1 Sampling Abstraction for cost function κ_λ

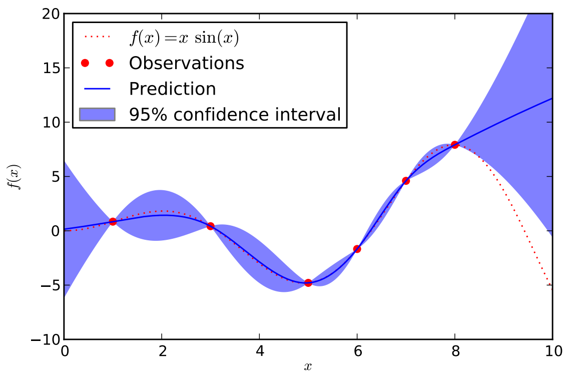
```
Require: $(X, D, C)$ are the constraints for an intermediary $\lambda$
Ensure: $\gamma$ are contribution-cost pairs
1: $I \leftarrow s$ sampling points $\in L^\lambda$          ▷ domain of $\kappa_\lambda$
2: procedure SAMPLING-ABSTRACTION($\lambda, I$)
3:     for all $\{i \in I\}$ do
4:         $C' \leftarrow C \cup \{(S_\lambda[0] = i)\}$
5:         $o \leftarrow$ solve $\langle X, D, C' \rangle$ : minimize $\Gamma[0]$
6:         $\gamma \leftarrow \gamma \cup \{(i, o)\}$
7:     return $\kappa_\lambda \leftarrow$ pwLinear($\gamma$)
```

This is just copied unpublished material from my thesis, I'll rewrite that.

Therefore, Concretely, these problems consist of the constraints in the SAM and introduce an additional constraint that fixes the input variable to some particular value and tries to minimize or maximize the output variable. For example, $S_v[0]$ (with $S_v[0]$ still being the sum of all $S_a[0]$) is bound to be



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



(b) A probabilistic regression model allows to quantify uncertainty at given points in the domain of a learned function.

400 and the objective is to maximize $S_v[1]$. Then, these input-output pairs can be represented by a suitable approximation method instead. As of now, we employ *piecewise linear functions* that are readily supported by MIP or Constraint Solvers such as CPLEX and have been applied in model abstraction in simulation engineering [4]. After finding the feasible production ranges of a VA by general abstraction, we can perform sampling abstraction by using a number of sampling points distributed across the production range and collect the respective outputs. Currently these sampling points are selected equidistantly across the full range. We sketch the approach in Alg. 2 for the positive production change speed. The "sampled" piecewise linear function can then be used for additional constraints in the AAM .

Algorithm 2 Sampling Abstraction for change speeds

```
Require: $(X, D, C)$ is the SAM of $v$
Ensure: $\kappa$ are pairs of the positive change speed
1: $\mathcal{I} \leftarrow s$ sampling points $\in L^a$
2: procedure SAMPLING-ABSTRACTION($v, s$)
3:     for all $\{i \in \mathcal{I}\}$ do
4:         $C' \leftarrow C \cup \{(S_v[0] = i)\}$
5:         $o \leftarrow$ solve $\langle X, D, C' \rangle$ : minimize $\Gamma[1]$
6:         $\kappa \leftarrow \kappa \cup \{(i, o)\}$
7:     return pwLinear($\kappa$)
```

B. *Modifying existing SCSOPs for sampling Input-Output pairs*

Given that the agent models are already available in a constraint language as needed for the load distribution problem, the main idea in sampling consists of reusing these constraints and formulating new problems. Compared to the distribution model, no residual load or initial states are included in these "sampling" models. A special role is given to the decision variables for the agents' production at time step 0. Usually constraints link the current state to these variables, e.g. $S_a[0] = 10.0$. But in this case these constraints are dropped and an additional constraint only ensures that a suitable initial state is found which is bound to an aggregate value: $\sum_{a \in \mathcal{A}_\lambda} S_a[0] = 500.0$. Then a suitable valuation for the productions at 0 are sought by the solver and a target expression is maximized or minimized.

C. *Piecewise Linear Functions for Sampled Functions*

D. *Sampling Change Functions for an Abstracted Agent Model*

The following graphs illustrate what piecewise linear approximations of functions of decision expressions in synthesized regional models look like for the AVPP example presented in Sect. **??**. We revisit the virtual agent w consisting of three concrete agents:

a:	$L^a = \langle [0,0], [50,100] \rangle$, $\gamma^a = 13$
b:	$L^b = \langle [0,0], [15,35] \rangle$, $\gamma^b = 70$
c:	$L^c = \langle [0,0], [200,400] \rangle$, $\gamma^c = 5$

where γ is the price per production unit such that the total cost for one time step is given by $\sum_{a' \in \{a,b,c\}} \gamma^{a'} S_{a'}[t]$. General abstraction gave the feasible production space of this virtual agent in Sect. **??**: $L^v = \langle [0.00.0], [15.035.0], [50.0135.0], [200.0535.0] \rangle$. Different numbers of sampling points are selected equidistantly over the whole production space in addition to the boundary points of the intervals. Therefore, the more sampling points are used the higher the accuracy of the obtained piecewise linear function.

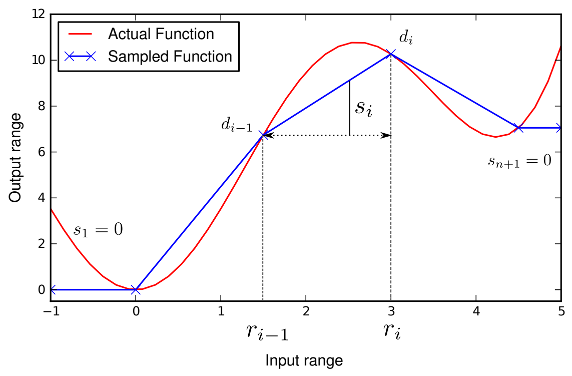
The objective for the function in Fig. 3b was set to minimize the cost given a combined production. As in this example low production rates are set to be quite expensive, also the combined production of the virtual agent is more expensive at low rates.

E. *Issues with Sampling Abstraction*

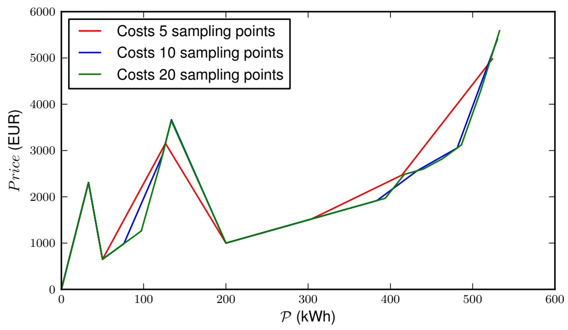
However, if the selected input-output pairs are selected in a weakly informative way, the resulting abstracted optimization problem introduces severe errors in quality as well as bad runtime performance.

III. Improving Sampling Point Selection by Active Learning

Just some very preliminary pointers to literature



(a) Example function $f(x) = x^2 + 3x \sin(x)$ sampled at $\{0, 1.5, 3.0, 4.5\}$.



(b) Cost function sampled at different accuracies

Active learning using Gaussian process has been applied to several problems [8], [9], [10].

IV. Evaluation

We investigate the effects of selecting a particular set of sampling points for one group that could have emerged as part of a self-organization process.

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] 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.

[9] A. Krause, A. Singh, and C. Guestrin, "Near-Optimal Sensor Placements in Gaussian Processes: Theory, Efficient Algorithms and Empirical Studies," *J. Mach. Learn. Res.*, vol. 9, pp. 235–284, Jun. 2008.

[10] 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.