Automation of Data Acquisition Strategies in Model Calibration for System Models: Sensor Placement

Tianyi Li*
System Dynamics Group, Sloan School of Management, MIT

Munther Dahleh†
Institute for Data, Systems and Society, MIT

.

Abstract

Model calibration is a challenging process for large-scale system models with a non-trivial number of variables. Although domain heuristics are often called for to guide the model calibration process, in particular, the scheduling of data acquisition on model variables, there lacks a general automatic framework for this strategic task. This problem points to the well-known sensor placement problem in physical dynamic systems, and existing approaches are ready to be translated to system models in social and economic sciences. The sensor placement problem on system models addresses the following question: with the model at hand and a pre-existing data availability, what are the (next) k model variables that would bring the largest utility to model calibration, once their datasets are acquired? In this study, we first translate two established solution approaches of this optimization problem, the information entropy approach and the miss probability approach, from physical dynamic systems to social science system models. Next, based on the idea of Data Availability Partition and drawing on the insights of the two existing solutions, we propose a new objective function for this optimization problem, constructed from the individual utilities of model variables, which could be understood from the entropy perspective. The new solution could essentially be embedded in the broader theoretical framework of the evaluation of side information. On the basis of graph theory, analytical results of the optimal placement solution under the new objective function are derived for binary and multi-ary trees; for a general tree structure with n nodes, an algorithm to determine the optimal placement is devised, with complexity upper bounded by $O(n\log_2(n))$. For arbitrary model structures, approximate solution schemes for this combinatorial problem are pinned down. Our new solution scheme is compared with the two translated approaches on a sample model structure, and results suggest the advantages of our solution. In this study, in-depth discussions on the design of data acquisition strategies for system models are carried out, which may bring important insights for system modeling practice in social and economic sciences.

Keywords— sensor placement, data acquisition, model calibration, system models, combinatorial problem

1 Introduction

One important issue in system modeling in social and economic sciences is model calibration, namely, the process of using as many available data on model variables as possible to constrain model behaviors and determine model parameters, so that the calibrated model could be used to generate reliable dynamics. In actual practice, for large-scale system models with a non-trivial number of variables (e.g., $\sim$100), a few serious problems arise with this task. First, it is almost always the case that not all variables in a simulation model could have empirical data available; the modeler is often able to collect datasets for only a subset of model variables. Second, even if datasets could be

*tianyil@mit.edu
†dahleh@mit.edu

acquired, their quality may vary a lot. The data might come from different sources, either reliable or unreliable, and sometimes multiple datasets may even be contradictory to each other. They might be of different resolutions in the time (or spatial) scale, and the length of these time series is almost always shorter than the model's simulation time range. Third, and more importantly, with datasets of the best possible quality at hand, the modeler nevertheless often has little idea about how to efficiently plan the calibration process for the large system, i.e., which part of the model is to be calibrated first, and which part next, such that the acquired datasets on model variables could be made best use of. System models are not "flat", in which the interdependencies between model variables are complex and often composed of highly nonlinear formulations; therefore, to properly guide the calibration process in system modeling, specific calibration strategies need to be designed in the first place, which is undoubtedly a non-trivial strategic problem for large-scale system models that contain multiple compartments or sub-modules. As one might observe, in social and economic sciences, these integrated system models are now becoming more and more popular in the age of computation.

Across disciplines, many empirical heuristics are called for to determine such calibration strategies for system models, and most of them rely on domain knowledge. Discussion on the design of calibration strategies for large-scale system models has been seen in crop yield models (*Angulo et al.*, 2013; *Grassini et. al*, 2015; *He et. al*, 2017), coastal erosion models (*Simmons et. al*, 2019), agro-ecosystem models (*Hansen et. al*, 2012; *Kersebaum et. al*, 2015), hydrologic and water quality models (*Daggupati et. al*, 2015), aggregated agricultural ecosystem models (*Tian et. al*, 2012; *Negm et. al*, 2014) and at a general level, system dynamics models (*Oliva*, 2003). Many calibration strategies have been proposed and investigated, asking questions from different perspectives, such as: which datasets (on what variable, during which period) should be used for calibration and for validation, in order to obtain the least calibration error in crop models (*He et. al*, 2017); for distributed models (e.g., sub-basins within the entire basin), whether to use lumped parameters or instead calibrate each segment separately, will the best fitting performance be yielded (*Khakbaz et al.*, 2012); whether to use Global Sensitivity Analysis to sort out insensitive variables from the entire system (*Saltelli et al.*, 2008); how to determine a minimum number of locations that are required to achieve robust estimates of yield gaps at larger spatial scales (*van Bussel et al.*, 2015). On top of these domain-specific topics, recently, a few research questions begin to concern the calibration process in a more general problem setting, such as how many datasets are required to optimize a model (*Simmons et. al*, 2019), and what is the minimum variable subset that is required to calibrate models (*Grassini et. al*, 2015).

Among these discussions, it is important to note that people mean different things when talking about model calibration strategies. One stream of studies focus on strategies that would yield good inversion results, i.e., strategies that improve the model's fit to the data. As a result, measures of a model's data-fitting performance (e.g., calibration errors) are often used as the objective function for the strategy design in these studies. In this setting, people regard heuristics in parameter determination as calibration strategies and may focus on the specific design of the objective function (e.g., which variable to include and which to exclude). Essentially, these studies assume the data acquisition process to be efficient; that is, datasets are not difficult to collect and are often of good quality, and the problem points to how to use the always available datasets in an optimal way so as to calibrate a good model.

A second type of studies in calibration strategy design approaches the problem from a different angle. The assumption that datasets are easily available is dismissed, and the idea is that in practice, data acquisition is often expensive and time-consuming, especially in fine-scale field models (e.g. *van Bussel et al.*, 2015). Therefore, with the model at hand, experimentalists need guidelines for experiment design such that they could decide on the most effective measurements to improve the usefulness of their available data (*Kersebaum et. al*, 2015). In other words, they need to decide on which variables the data are to be acquired, so that the model will be calibrated to the best extent. These ideas regarding the prioritization of information collection point to important model-based decision-making. Studies suggest that such tasks could be supported by expert judgements, a well-established methodology in management sciences grounded with solid theoretical analysis (*Cooke*, 1991; *Cooke and Goossens*, 2004; *Kraan and Bedford*, 2005). A good example is in *Kersebaum et. al* (2015), where variables are assigned with different "relevance" to the model according to expert interviews, and each variable is associated with a "weight" calculated from this relevance factor, together with the number of observations etc.; in the end, based on this quantitative attempt, the classification of datasets on different variables is made on a level basis. As one would imagine, such a quantification effort produces sound utilities in practice, although the process involves much human intervention. It then brings the idea that, modelers from various application domains could well benefit, if the determination of calibration strategies on system models could be made fully automatic by an efficient computer algorithm that could apply to different model structures. By this means, human factors as potential pitfalls during the process may be largely avoided.

One would expect that, such an automation will help answer the following question regarding the data-acquisition process: with a system model at hand and the current data-availability condition, what are the next k variables whose datasets, once acquired, would be the most useful for model calibration? And in a further sense, how to arrange the

model calibration process in *stages* so as to bring out the maximum utilities of the available datasets, i.e., which k_1 variables should be calibrated first and which k_2 variables are calibrated next? From a different perspective, this algorithmic effort would also be useful in the problem of model selection: given a determined availability of data and multiple candidates of system models, which candidate model among them is the best one to explain these data? Such a problem targets a very important aspect in model estimation, as the commesuration of the availability of datasets on certain model variables with the resolution of model components constructed to explain these data is always the first-order concern in successful parameter estimation (*Li et al.*, 2020). Overall, as one could see, these questions do not focus on models' fitting performance, but instead concern the efficiency of utilizing available data in model calibration. Moreover, in this study we are also not much concerned with the Bayesian point of view on model calibration, which may regard the noise structure in datasets (e.g., types of system errors and measurement errors), the choice of estimators (e.g., different optimizers, different likelihood functions, and the potential use of Kalman Filtering), or the sensitivity analysis of estimated parameters (e.g., their confidence intervals) as well as model outputs (e.g., overall model sensitivities). For the purpose of the current study, we simplify these discussions and regard the (varied) qualities of datasets with computable scores that are compatible with our algorithmic establishments; yet the overlooked are important aspects in model estimation which are to be further addressed in future attempts.

Essentially, as one could see, questions in the above are concerned with the *staging design* of model calibration procedures. This is an important issue in applied system models, which are often integrated, process-based, and contain multiple sub-modules. *Daggupati et. al* (2015) categorized the staging design of model calibration in agricultural and ecosystem models, which may nevertheless apply to all system models. A model calibration process could be *single-stage*, i.e., calibrate all variables at the same time; or *stepwise single-pass*, i.e., based on a ranking of variables, calibrate the most useful variable at each step, not necessarily producing an optimal sequence (e.g., SWAT model (*Santhi et. al*, 2001; *Arnold et. al*, 2012), DAISY model (*Hansen et. al*, 2012)); or most ideally, *stepwise multi-pass*, i.e., calibrate several model variables together at one stage (e.g., DRAINMOD-FOREST (*Tian et. al*, 2012), CREAMS/GLEAMS (*Knisel and Douglas-Mankin*, 2012), DRAINMOD-DSSAT (*Negm et. al*, 2014)). Although properly defined, however, in most applied system models of specific domains, the currently used staging heuristics mostly rely on "yes/no" decisions, and are not flexible under different data availability conditions; an automatic staging algorithm that could deal with all possible data availability conditions has not been realized.

Theoretically, the problem of concern is closely attached to the *sensor placement* problem, a well-known topic in studies of physical dynamic systems. Given a system with a large panel of variables and a limited budget for sensors, people are interested in the optimal placement of these sensors on candidate nodes of the system graph, such that a certain objective function (i.e., a specific utility defined on the system) is maximized. Translating into the current setting, now we are interested in the selection of the optimal subset of model variables on which measurements are to be made, such that the model could be calibrated to the best extent. Among sensor placement studies, many consider the coverage of nodes in the (physical) system (*Dhillon and Chakrabarty*, 2003; *Lin and Chiu*, 2005) as the objective of an optimal placement; in other cases, people aim to minimize the information entropy of the system (*Papadimitriou*, 2004; *Papadimitriou and Lombaert*, 2012) through an efficient placement configuration, which is significant for the parametric identification of structural systems, a topic closely related to the current problem setting. Among various studies, sensor placement is generally formulated as a combinatorial problem, and approximation solution schemes such as simulated annealing (*Lin and Chiu*, 2005) and sequential optimal algorithms (*Dhillon and Chakrabarty*, 2003; *Papadimitriou and Lombaert*, 2012) are designed to generate feasible solutions.

Intrinsically, the placed sensors could be viewed as delivering *side information* about the system (i.e, complementary information regarding measurement of the system), hence the utility of these sensors could be studied under the theoretical framework for the evaluation of side information on graphs (*Rinehart and Dahleh*, 2010, 2011, 2012). In these studies, the objective function of the optimization problem on graphs concerns the shortest path (*Rinehart and Dahleh*, 2011) or the network flow (*Rinehart and Dahleh*, 2012); essentially, in this study we are proposing a new objective function for the evaluation of side information, focusing on the structural properties of the system graph and its topological features. In a further sense, the problem in concern may also be linked to the literature on expected value of sample information (EVSI, e.g., *Dakins et al.*, 1996; *Ades et al.*, 2004)). In practice, since the ground-true values of model variables are almost always impossible to be completely known without noise interference, the measurement on variables (i.e., the placement of sensors) might thus be viewed as sampling attempts and then yield certain expected values.

Conceptually, this issue of strategic model calibration echoes with the problem of variable subset selection, a classic topic in data processing and machine learning that has wide applications in business and finance (e.g., *Tian et al.*, 2015; *Amendola et al.*, 2017). The critical idea in variable subset selection is that the single-pass optimal sequence of length k, possibly obtained from some greedy algorithms, may not be the optimal k-pass sequence; a variable that is completely useless by itself can provide a significant performance improvement when taken with others, and multiple

variables that are useless by themselves can be useful together (*Guyon and Elisseeff*, 2003). These ideas are identical to ours in the above discussion. Normally, the subset selection is based on a ranking of variables, with each variable associated with a score that indicates its significance to the model based on a certain utility metric. To realize the automation of data acquisition strategies in a similar manner, we would need to construct a measure to represent variables' utilities in the model calibration process. In machine learning literature, the score of a variable is primarily represented by its contribution to the underlying model's fitting performance, e.g., to which extent the variable is useful in accounting for the fitting errors. In this case, variables are essentially independently considered and their utilities are often not conditional on each other. By contrast, the utility of a specific variable in our model calibration process is determined by the structure of the model and also by the availability of data on other variables; hence in our problem setting, variables' utilities are supposed to be conceptualized in a conditional manner, which is a critical idea in this study.

Organization of the paper

This paper is organized as follows. In Section 2, we set up the theoretical framework and pin down the sensor placement problem of this study. Two objective functions for the defined optimization problem, both of which were originally proposed for sensor placement on physical dynamic systems (*Papadimitriou and Lombaert*, 2012; *Dhillon and Chakrabarty*, 2003), are translated to the current problem setting for system models in social and economic sciences. In Section 3, based on the idea of data availability partition (DAP, *Oliva*, 2004), we propose a new objective function for this combinatorial problem, constructed from the individual utilities of model variables, which could be understood from the entropy perspective. The empirical issue of the varied quality of real datasets is addressed, with corresponding terms built into the objective function. In Section 4, the sensor placement problem is embedded in the broader theoretical framework for the evaluation of side information (*Rinehart and Dahleh*, 2011). In Section 5, we discuss the comparison between using the proposed metrics as the utility term for model variables with using well-established graph centrality measures as the utilities. Analytical results of the optimal placement solutions under our new objective function are derived in Section 6. These results apply to system models having tree structures: for binary and multi-ary trees, exact solutions are obtained; for general tree structures, we devised an algorithm to determine the optimal placement sequence, whose computational complexity is proved to be $O(n\log_2(n))$ for a model with n variables. For models with arbitrary topologies (e.g., with feedback loops), the optimization is NP-hard and approximate solution schemes (sequential optimal and simulated annealing) are discussed in Section 7. The three investigated objective functions for the problem are compared on a sample model structure, and results are discussed in Section 8. The paper is concluded in Section 9.

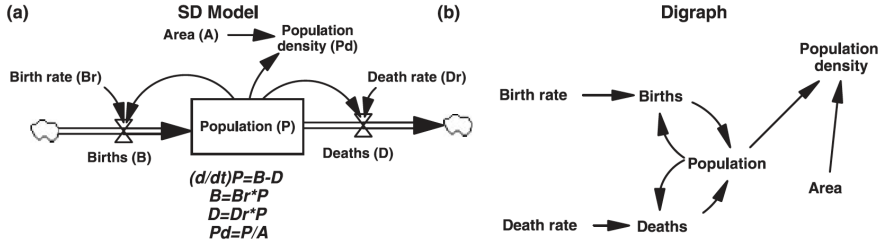
2 Problem Statement and Existing Solutions

System models are commonly seen in social and economic sciences. For a specific problem, modelers formulate differential equations to capture the dynamic relationships between system variables (as opposed to analysis based on regression), and the resulting model diagram therefore could be abstracted as a graph (or a network) where variables are represented by nodes and their relationships are represented by edges, on which graph-theoretical analysis could be conducted. Unlike physical systems where often only stock variables (denoting differential relations $\dot{a} = b + c$) are the matter of concern, in many cases dynamic system models in social sciences also highlight auxiliary variables which are not stocks (denoting plain additive relations $a = b + c$; e.g., "population growth rate = birth rate - death rate", the rates are auxiliary variables). Dependencies between variables (i.e., edges on the graph), which are always directed, are either linear or non-linear (i.e., multiplicative); this high nonlinearity adds to the difficulty of the theoretical analysis on those model graphs, and disallows the application of mathematical tools that work exclusively on linear relationships. A sample system model structure is shown in Figure 1 (adapted from *Oliva* (2004)), where the illustrative view of the model (Figure 1a) is abstracted into the graphical view (Figure 1b), with edges indicating linear or nonlinear dependencies between model variables (nodes) consisting of both stock and non-stock variables. The graph could then be further represented with matrix forms (Figure 1c,d).

System models are used in simulations to produce time series for certain output quantities, with model variables able to be tuned such that the model generates desired dynamics. These models are extremely useful in policy analysis, where modelers are able to simulate and then evaluate the effects of proposed policy measures, in which case a certain policy measure corresponds to determining values for a selected set of model variables that serve as policy handles, i.e., decision variables. These decision variables could be either parameters or not, as parameters who bear the same value along the time series are essentially degenerated from variables having non-trivial time series. For

those variables that are not policy handles, the modeler needs to determine their time series at proper values such that the system is not letting "everything goes", (the system could simulate many different dynamics for the output variables in concern), which is completely undesirable in policy analysis.

Indeed, such risks point to a core weakness of this modeling methodology: the model behavior needs to be maximally constrained so as to maintain the persuasiveness of simulation results. A system model has little merit when it is unconstrained and its variables are subject to free tuning; only when some parts of the model are determined, so are the dynamics of corresponding system components, could the model be made useful in analysis. Take the sample model in Figure 1. When there is no information about the proper values of any model variable, the model itself is not interesting. If, for example, the time series of *Population (P)* and *Birth rate (Br)* are known (e.g., annual or seasonal data series), then, according to model equations in Figure 1a, we are able to determine the time series for *Births (B)*, and further calculate *Deaths (D)* and *Death rate (Dr)*. Elsewise, if only the time series of *Population (P)* is known, we could now use *Birth rate (Br)* and *Death rate (Dr)* as decision variables and tune them towards the best-fit values, where the model-simulated series of *Population (P)* match the data series to the best extent. And then, at that point, the time series for *Births (B)* and *Deaths (D)* are determined accordingly. One could see that, in both cases, a large part of the model (except *Population density (Pd)* and *Area (A)*) is pinned down and now the dynamics of these system variables are not "flying around". In this example, the model itself might still have certain values since it is small and modular, even when its dynamics are not determined; for more complex system models, things are worse and the shear model structure is hardly of any use in policy analysis before some components of the system could be constrained.



(c) Adjacency matrix

	Br	*B*	*P*	*Dr*	*D*	*Pd*	*A*
Br	0	1	0	0	0	0	0
B	0	0	1	0	0	0	0
P	0	1	0	0	1	1	0
Dr	0	0	0	0	1	0	0
D	0	0	1	0	0	0	0
Pd	0	0	0	0	0	0	0
A	0	0	0	0	0	1	0

(d) Reachability matrix

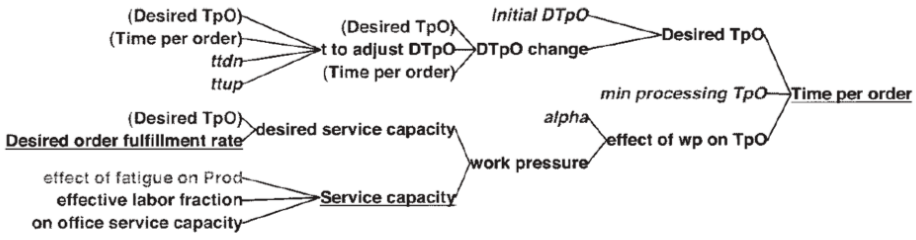
	Br	*B*	*P*	*Dr*	*D*	*Pd*	*A*
Br	1	1	1	0	1	1	0
B	0	1	1	0	1	1	0
P	0	1	1	0	1	1	0
Dr	0	1	1	1	1	1	0
D	0	1	1	0	1	1	0
Pd	0	0	0	0	0	1	0
A	0	0	0	0	0	1	1

Figure 1: A sample model structure (adapted from *Oliva* (2004)), showing different views (illustrative, graphical, matrix forms) of the system. The model is used in our experiments for testing the algorithms (Section 8).

More generally speaking, the procedure of this post-modeling task thus described is the model calibration process, in which datasets on some model variables are collected (i.e., these variables are "measured"), and based on these acquired data, values of certain model variables are pinned down, at places where they make the simulated time series best match the data on those measured variables. Another (yet more complicated) sample calibration scenario of system models is shown in Figure 2 (adapted from *Oliva* (2004)). This structure (i.e., partial model) is cut out from a complete model, consisting of a list of variables whose interdependencies are represented by the tree structure, with arrows indicating that the variable on the right end is dependent on the left-end variable. Among variables on the tree, there are data available on the rightmost variable **Time per order** (as well as on **Service capacity** and **Desired order fulfillment rate**; all three are underlined). Therefore, by matching the simulated time series and the real data on **Time per order**, one is able to determine the best-fit values of parameters *alpha, ttup, ttdn, Initial DTpO* of this partial model, and then, based upon these values, the dynamics of the other unmeasured variables on this tree (*Desired TpO, DTpO change, t to adjust DTpO, effect of wp on TpO, work pressure, desired service capacity*; see rows below "***Subject to***" in Figure 2) could be simulated.

Per the above example, conceptually, the availability of data on a model variable (e.g., **Time per order** in this

case) has a certain utility in model calibration (i.e., having this variable measured, we are able to simulate the model component that it associates with (Figure 2), spanned by a number of unmeasured variables), which could be defined in a specific quantitive manner. In a broader sense, we are more concerned with the aggregated utility of the entire data-availability condition regarding all model variables. Once such a utility function is successfully defined, we are then able to formulate the following strategic problem in model calibration: given a limited budget on the acquisition of datasets, determine the optimal set of model variables (i.e., sensors) such that the availability of data on these variables has the largest utility regarding the model calibration process.



Minimize

Σ(Time per order(t) – **Time per order**(t))^2 for $\{t \mid$ **Time per Order**(t) = *value*$\}$

Over:

Initial DTpO > 0; alpha < 0; ttup > 0; ttdn > 0

Subject to:

Time per order = max(Desired TpO * effect of wp on TpO, min processing TpO)
Desired TpO = INTEG (DTpO change, Initial DTpO)
DTpO change = (Time per order-Desired TpO)/t to adjust DTpO
t to adjust DTpO = IF THEN ELSE(Time per order>Desired TpO, ttup, ttdn)
effect of wp on TpO = EXP(alpha * work pressure)
work pressure = (desired service capacity – **Service capacity**)/desired service capacity
desired service capacity = **Desired order fulfillment rate** * Desired TpO
min processing TpO = 0.6

Figure 2: Calibration of a (partial) system model, showing the DAP structure (adapted from *Oliva* (2004)).

Now we set up rigorous mathematical formulations for this sensor placement problem. Mathematically, a system model is represented by a directed graph G which consists of n nodes, each representing a variable $x_i \in \boldsymbol{X}$ in the model. The graph could either bear a tree structure, in which case the original system model possesses no feedback loop (e.g., Figure 2), or an arbitrary non-tree structure containing loops that represent certain feedback mechanisms in the model (e.g., Figure 1b). The (asymmetric) connectivity matrix $\boldsymbol{A} = \{0,1\}^{n\times n}$ of the graph indicates the interdependencies between variables (e.g., Figure 1c); when there is dependence of variable x_j on variable x_i, edge $(i,j) \in E$ exists and $a_{ij} = 1$. These dependent relationships could be either linear or non-linear (e.g., multiplicative, exponential, polynomial etc., or even table functions, which are not uncommon for real-world system models); we do not consider their specific formulations, but focus on the existence of these dependencies, i.e., the connectivities of the abstracted system graph. Note that although complex mathematical formulations of model variables degenerate into 0/1-valued entries in the connectivity matrix $\boldsymbol{A}$, this is not equal to assuming linearity of the system; in this study we do not put any weight on linear dependencies and essentially view all relationships as equal.

Building upon the graph-theoretical setup, the problem now is to determine the optimal subset of k model variables $\boldsymbol{X^d} \in \boldsymbol{X}$ where $|\boldsymbol{X^d}| = k$, such that the datasets of these measured variables, once acquired, produce the maximum aggregated utility for model calibration, under a certain objective function U. The data-availability condition could be represented by a vector $\boldsymbol{d} = \{0,1\}^{1\times n}$; $d_i = 1$ or 0 indicates that the model has or doesn't have data on variable x_i. By this means, the problem could be formulated as an optimal *sensor placement* problem [1]:

$$\begin{aligned} \boldsymbol{d}_{best} &= \underset{\boldsymbol{d}}{\operatorname{argmax}}\ U(\boldsymbol{d}, \boldsymbol{A}, \boldsymbol{\theta}), \\ s.t.\ \ d_i &\in \{0,1\};\ ||\boldsymbol{d}||_1 = k, \end{aligned} \tag{1}$$

[1] We use the term "sensor placement" throughout the paper in order that the discussions are aligned with existing studies and techniques in physical sciences. In the setting of social and management sciences, the placement of sensors upon receival of signals is better described as the acquisition of information from measurements on selected model variables, which is an active rather than passive process.

with $\boldsymbol{\theta}$ denoting parameters in the specific formulation of U.

Most existing studies on the sensor placement problem concern physical dynamic systems; among these studies, *Papadimitriou and Lombaert* (2012) and *Dhillon and Chakrabarty* (2003) provided two candidates for the objective function U that could be readily translated to our setting of social and economic system models. After the translation of these existing approaches, we propose a new form of U combining the insights of the two existing ideas (Section 3). Before proceeding with these discussions, to better understand the translation of existing methods and our new formulation, we first introduce the idea of *data availability partition*, proposed by *Oliva* (2004).

For a model variable $x_i \in \boldsymbol{X}$ (a node in graph $G = (\boldsymbol{X}, E)$)			
Symbol	**Description**	**Location**	**Set**
d_i	availability of dataset (0/1-valued)	Section 2	$\boldsymbol{d}$
r_i	reliability of dataset	Section 3.3	$\boldsymbol{r}$
l_i	length of dataset	Section 3.3	$\boldsymbol{l}$
$\boldsymbol{x}_i^{\boldsymbol{d}}$	measured (data-available) variables in the DAP of x_i	Section 2.1	✓
$\boldsymbol{\alpha}_i$	parameters in the DAP of x_i	Section 2.1	✓
$\boldsymbol{\beta}_i$	unmeasured (intermediate) variables in the DAP of x_i	Section 2.1	✓
σ_i	variances for the measurement of x_i	Section 2.2	$\boldsymbol{\sigma}$
$\hat{m}_i$	miss probability of x_i	Section 2.3	$\hat{\boldsymbol{m}}$
$v(i\|\boldsymbol{d})$	(existing) data-available utility of x_i	Section 3.1	-
$v^{\dagger}(i\|\boldsymbol{d})$	counterfactual/add-on data-available utility of x_i	Section 3.1	-
v_i^{in}	in-degree centrality of x_i	Section 5	-
v_i^{out}	out-degree centrality of x_i	Section 5	-
v_i^{Katz}	Katz centrality of x_i	Section 5	-

Table 1: List of notations regarding a model variable $x_i \in \boldsymbol{X}$ and their descriptions. Checkmarks indicate where a symbol represents a set. Symbols appearing in the paper but not in the list are not concerned with a specific node (e.g., node i corresponding to x_i) in the model graph. To note (not in the list), s represents a specific sequence of DAPs of the model among all possible sequences $\mathcal{S}$. Also note that letters used in subscripts or superscripts do not bear the same meaning as when they represent symbols in the table, e.g., the meaning of $\boldsymbol{d}$ and $\boldsymbol{l}$ (data availability vector and data length vector) do not apply to $\boldsymbol{X}^{\boldsymbol{d}}$ and $\boldsymbol{X}^{\boldsymbol{l}}$ (the set of measured nodes and the set of leaf nodes among all system variables $\boldsymbol{X}$).

2.1 Data availability partition (DAP)

In the above example (Figure 2), the calibration scenario is matching the data and simulated series of the rightmost variable (measured). One could then imagine that, for each measured variable $x_i \in \boldsymbol{X}^{\boldsymbol{d}}$, a similar sub-graph could be cut out from the entire system graph G, with the variable being the topmost node of the tree (e.g., see Figure 3; (b) or (c) shows the four sub-graphs associated with the four measured variables i, j, k, l in the illustrative model (a)). This is exactly the case; such a sub-structure is termed as the *data availability partition* (DAP), and *Oliva* (2004) proposed an algorithm to automate this partition based on a breadth-first search backtracking the incidents of the measured variable in discussion. Further, the output of the DAP algorithm not only returns the nodes on the DAP tree, but also categorizes these nodes into three sets according to their roles in model calibration: some nodes in the tree are also measured, besides the topmost node; some are parameters that are to be tuned in calibration, and are always leaf nodes of graph G; the rest are unmeasured variables that are intermediate nodes of the tree, whose dynamics are determined through simulation. [2]

[2]In practice, if a leaf node is not a parameter (i.e., not constant over time), one almost always has data for this leaf variable. It is very rare in system models that a leaf node is neither a tunable parameter nor a measured variable; this is often considered

Therefore, given a measured (data-available) variable x_i, the DAP algorithm returns the corresponding tree structure separated from the model that calibrates on x_i, paying attention to the distinction of variables' roles:

$$\boldsymbol{x}_i^d, \boldsymbol{\alpha}_i, \boldsymbol{\beta}_i \longleftarrow DAP(\boldsymbol{A}, \boldsymbol{d}, x_i), \tag{2}$$

where $\boldsymbol{x}_i^d$ denote variables in the separated structure that are also measured; $\boldsymbol{\alpha}_i$ denote system parameters; $\boldsymbol{\beta}_i$ denote unmeasured variables. If the complete model has a strict tree structure, then the backtracking DAP graph for any measured node stops either at a leaf node or another measured variable; in the case of cyclic graphs where feedback loops exist in the original system, which is often the case among real-world system models, additionally, the DAP will possibly stop at variables in the feedback vertex set (see Section 3.4). Also note that the determination of DAP graphs of a model is not unique (Figure 3), and the sequence of their determination with respect to the measured nodes plays a non-trivial role (see Section 3.2).

In the following discussion, based on the idea of DAP, different formulations of the objective function U could be constructed, including the two translated approaches (*Papadimitriou and Lombaert*, 2012; *Dhillon and Chakrabarty*, 2003) and our new formulation (Section 3).

2.2 The information entropy approach (*Papadimitriou and Lombaert*, 2012)

One candidate objective function U is constructed from the Fisher information matrix $\boldsymbol{Q}$, and the optimal $\boldsymbol{X^d}$ minimizes the information entropy of the system (*Papadimitriou and Lombaert*, 2012), which originates from the uncertainty in measuring variables (i.e., the measurement error). Assuming linearity (which might not be a good assumption for real-world system models; see below), measured variables (sensors) $\boldsymbol{X^d}$ are regressed on model variables $\boldsymbol{X}$, represented by the selection matrix $\boldsymbol{L}$ (which has a one-to-one correspondence to the data-availability condition $\boldsymbol{d}$, as one could imagine), and $\boldsymbol{X}$ are linear combinations of leaf nodes $\boldsymbol{X^l}$, represented with the backtracking reachability matrix $\boldsymbol{\Phi}$ (i.e., the transitive closure with respect to $\boldsymbol{X}$). Each variable x is assumed to be associated with a measurement error, captured through the variance σ_x^2. That is:

$$\boldsymbol{X^d} = \boldsymbol{L}\boldsymbol{X}; \boldsymbol{X} = \boldsymbol{\Phi}\boldsymbol{X^l}; \boldsymbol{\Sigma} = diag(\sigma_{\boldsymbol{x}}^2)\boldsymbol{I}, \tag{3}$$

in which the sensors $\boldsymbol{X^d}$ are essentially projected to leaf nodes $\boldsymbol{X^l}$ in the linear space $\boldsymbol{L\Phi}$. The information entropy $-\ln[det(\boldsymbol{Q})]$ is calculated from the Fisher information matrix $\boldsymbol{Q} = |\boldsymbol{X^l}| \times |\boldsymbol{X^l}|$, which is constructed from $\boldsymbol{L}$, $\boldsymbol{\Phi}$ and $\boldsymbol{\Sigma}$:

$$\boldsymbol{Q} = (\boldsymbol{L\Phi})^T(\boldsymbol{L\Sigma L}^T)^{-1}(\boldsymbol{L\Phi}). \tag{4}$$

In the end, the objective function is $U = -\ln[det(\boldsymbol{Q})]$, and the problem is formulated as to minimize the information entropy:

$$\boldsymbol{d}_{best} = \underset{\boldsymbol{d}}{\operatorname{argmin}} \ -\ln[det(\boldsymbol{Q})]. \tag{5}$$

Conceptually, a certain placement of sensors corresponds to a specific dependency matrix $\boldsymbol{L}$; a placement $\boldsymbol{L}$ samples the intrinsic measurement errors Σ on model variables ($\boldsymbol{L\Sigma L}^T$), and projects them onto leaf nodes through linearity ($\boldsymbol{L\Phi}$). Hence $\boldsymbol{Q}$ is the covariance matrix that embeds the externalized uncertainty of a certain measurement (placement), and $-\ln[det(\boldsymbol{Q})]$ represents the information entropy. Note that as mentioned in *Papadimitriou and Lombaert* (2012), $\boldsymbol{Q}$ will be singular when the number of sensors to be placed is smaller than the degree of freedom of the model, i.e., $|\boldsymbol{X^d}| = k < |\boldsymbol{X^l}|$. In these cases, one maximizes the product of the k non-zero eigenvalues of $\boldsymbol{Q}$, instead of maximizing the product of all eigenvalues of $\boldsymbol{Q}$.

One could see that this formulation could be paraphrased to our current problem setting. Given the connectivity $\boldsymbol{A}$ of the model graph G, we translate our data-availability vector $\boldsymbol{d}$ to $\boldsymbol{L}$. For the variance $\sigma_{\boldsymbol{x}}^2$, we could either assume a uniform $\sigma = 1$ for all variables, or use a method to calculate the specific uncertainty for each different variable (see Section 3.3). In the original formulation, all variables are tracked back linearly to leaf nodes $\boldsymbol{X^l}$ (3); in our current setting, per the discussion of DAP, the dependency of each measured variable is not always tracked all the way back to leafs and may end half-way at other sensors $\boldsymbol{X_d}$. This suggests an amplification of the basis besides leaf nodes: $\boldsymbol{X^l} \to \boldsymbol{X}_+^{\boldsymbol{l}} = [\boldsymbol{X^l}, \boldsymbol{X^d}]$, and $\boldsymbol{\Phi}$ could always be determined from the graph connectivity $\boldsymbol{A}$. With these efforts, $\boldsymbol{Q}$ could be computed through (4), and the optimal placement of sensors aims to minimize the information entropy of the model. However, as mentioned earlier, a major defect of this formulation is that here all nonlinear relationships

as bad practice in system modeling and is avoided since in this case this variable cannot be calibrated at all. If this happens, one could always assume that this variable has a constant value over the simulation range, and this variable could then be viewed as a parameter.

in the model are viewed exclusively as linear dependencies, which might induce large system errors since in practice nonlinearity constitutes a nontrivial part of most complex systems in social sciences.

2.3 The miss probability approach (*Dhillon and Chakrabarty*, 2003)

Another idea for the optimal sensor placement is to maximize the coverage of the system (i.e., in terms of the number of nodes being reached) obtained from these sensors (*Dhillon and Chakrabarty*, 2003; *Lin and Chiu*, 2005). Here we adapt the setup in *Dhillon and Chakrabarty* (2003), which computes the miss probabilities of nodes in the system. The core idea is that there is a certain detectability of each sensor located at x_i, with respect to each variable x_j in the system, indicated by a probability $p_{i,j}$; the detection matrix is thus $\boldsymbol{P} = [p_{i,j}]_{n\times n}$, and the miss probability matrix $\boldsymbol{M}$ is then:

$$\boldsymbol{M} = [m_{i,j}]_{n\times n} = \mathbb{1}_{n\times n} - \boldsymbol{P}, \tag{6}$$

relying on which we identify the miss probability vector $\hat{\boldsymbol{m}}$ for each variable x_i:

$$\hat{\boldsymbol{m}} = [\hat{m}_i]_{1\times n} = [\prod_{j\in \boldsymbol{X^d}} m_{j,i}]_{1\times n}, \tag{7}$$

whose ith element is the product of the miss probabilities on variable x_i from all sensors. In *Dhillon and Chakrabarty* (2003), authors focus on designing sequential placement algorithms and did not formulate an explicit combinatorial problem. Effectively, such a problem could be formulated as (with the objective function $U = -||\hat{\boldsymbol{m}}||_1$):

$$\begin{gathered} \boldsymbol{d}_{best} = \underset{\boldsymbol{d}}{\operatorname{argmin}}\ ||\hat{\boldsymbol{m}}||_1, \\ \text{(optional) } s.t.\ card(\hat{m}_i = 1) = 0, \quad \forall \hat{m}_i \in \hat{\boldsymbol{m}}, \end{gathered} \tag{8}$$

where the objective is to minimize the average miss probability of all system variables, with the optional extra constraint that no variable is going to be collectively missed by all sensors.

The original setting in (*Dhillon and Chakrabarty*, 2003) dealt with the physical location of sensors and system nodes, where sensors' detectability $p_{i,j}$ is determined by the geological distance and blocked by potential obstacles lying in the field, which is considering a physical grid with regular topolopy. The formulation of $\hat{m}_i$ from $p_{i,j}$ implicitly assumes that each sensor has independent detectability, and the coverage on each node is determined collectively by these independent sensors. This assumption may hold true for the coverage on the geological field, but are not the case for the coverage of variables in system models as in our current setting. For system models with arbitrary graph topologies, the coverage of model variables is determined by the entire data-availability condition and the detectability of sensors is inter-dependent on each other. Hence for the current problem setting we construct a new form of $\hat{m}_i$ directly, instead of through $p_{i,j}$, due to the invalidation of the assumption of independent sensors.

Recall the DAP structure (Figure 2b), which is also valid if the target (topmost) variable x_i is not a data-available variable but an unmeasured variable. A natural idea to adapt the above approach is that, the collective detectability on x_i from all sensors could be represented by the proportion of sensors $\boldsymbol{x}_i^d$ among leaf nodes $[\boldsymbol{x}_i^d, \boldsymbol{\alpha}_i]$ in the DAP of x_i. If all the leaf nodes of its DAP tree are sensors, then x_i will be fully detected; if no leaf is a sensor as it is traced back, the target node x_i will not be detected at all. This suggests that the miss probability $\hat{m}_i$ is given by:

$$\hat{m}_i = 1 - \frac{|\boldsymbol{x}_i^d|}{|\boldsymbol{x}_i^d| + |\boldsymbol{\alpha}_i|} = \frac{|\boldsymbol{\alpha}_i|}{|\boldsymbol{x}_i^d| + |\boldsymbol{\alpha}_i|};\ \boldsymbol{x}_i^d, \boldsymbol{\alpha}_i \longleftarrow DAP(\mathbf{A}, \mathbf{d}, x_i), \tag{9}$$

upon which one calculates the miss probabilities $\boldsymbol{M}$ for all model variables, and the optimization problem (8) is pinned down.

3 Constructing the New Objective Function

Recall the calibration scenario of system models (Figure 2b). Each measured variable (sensor) is associated with a DAP tree, which is backtracking its incidents in the model; unmeasured variables in this tree will have calibrated simulation time series (as opposed to "'everything goes' time series generated with arbitrary parameter values) thanks to the calibration on the measured variable, regardless of their specific linear or non-linear formulations. Hence, conceptually, the sensors will be of greater utility to the model if with their help the system could be calibrated to the maximum extent, i.e., having as many system variables that have calibrated time series as possible.

This is similar to the coverage idea of the miss probability approach, and one immediately sees that the coverage of sensors on model variables could be calculated in a straightforward way with the idea of DAP. Such a view of the problem then allows us to construct the objective function U in a molecular way. Indeed, in this case, a utility is defined on each DAP, which may be viewed as the utility $v(i)$ of each individual sensor $x_i \in \boldsymbol{X^d}$ that leads to the DAP; thus $v(i)$ could be formulated as some functional form of the DAP results $(\boldsymbol{x}_i^d, \boldsymbol{\alpha}_i, \boldsymbol{\beta}_i)$:

$$v(i) = f(\boldsymbol{x}_i^d, \boldsymbol{\alpha}_i, \boldsymbol{\beta}_i), \tag{10}$$

With this, now we are able to calculate U through aggregating the individual utility of each sensor: $U \propto \sum_i v(i)$. One advantage of this way of constructing U is that now we can compare different sensors through their individual utility $v(i)$, which is not possible with the existing two approaches; and more importantly, we are going to see that pinning down the utility $v(i)$ for each sensor also allows us space to easily address additional concerns of the problem and incorporate them in utility terms (see Section 3.3).

It is significant to note that, since the backtracking DAP tree may stop at other sensors, the individual utility $v(i)$ of a sensor is *dependent* on the entire data-availability condition $\boldsymbol{d}$, i.e., $v(i) \to v(i|\boldsymbol{d})$. Thus for two different data-availability conditions $\boldsymbol{d_1}$, $\boldsymbol{d_2}$, there is $v(i|\boldsymbol{d_1}) \neq v(i|\boldsymbol{d_2})$; in particular, for a certain $\boldsymbol{d}$, there is $v(i|\boldsymbol{d}) \neq v(i|\boldsymbol{0})$, i.e., the utility of datasets on a specific variable is not guaranteed to be the same when there is data available on other variables as when there is no other data. Conceptually, the dependent nature of nodes' utilities underlines the difference between our problem setting and the sensor placement problem setting in physical dynamic systems; the latter often assumes independent utilities of sensors.

3.1 Add-on utility vs. existing utility

As mentioned above, once $v(i|\boldsymbol{d})$ is determined, the objective function $U(\boldsymbol{d}, \boldsymbol{A}, \boldsymbol{\theta})$ could then be constructed from $v(i|\boldsymbol{d})$. For brevity, from now on we write $U(\boldsymbol{d}, \boldsymbol{A}, \boldsymbol{\theta})$ as $U(\boldsymbol{X^d})$ or $U(\boldsymbol{d})$. Next, since $U(\boldsymbol{X^d})$ is formulated, the counterfactual/add-on utility of a new sensor x_j (a variable that currently does not have data available) should be given by:

$$v^\dagger(j|\boldsymbol{d}) = U([\boldsymbol{X^d}, x_j]) - U(\boldsymbol{X^d}). \tag{11}$$

$U(\boldsymbol{X^d})$ is essentially defining a scale for $v^\dagger(i|\boldsymbol{d})$, and represents a *state*. [3]

The calculation of $v^\dagger$ allows us to decide the optimal placement of a new sensor given the existing configuration (e.g., on the variable with the largest $v^\dagger$), which facilitates the sequential optimal placement of sensors as an approximate solution scheme of this combinatorial problem (see Section 7). Nevertheless, it is important to note that the counterfactual/add-on utility of a new sensor is defined on a different basis from the utility of an existing sensor, and both utilities are dependent on the entire data availability condition. Formally, for a node x_i, there is:

$$v^\dagger(i|\boldsymbol{d}) \neq v(i|\boldsymbol{d}). \tag{13}$$

This implies that we can only compare the existing utilities and add-on utilities of sensors within the same category, *at each stage* of the calibration process. That means, once a sensor has been placed (the dataset of a system variable are successfully acquired) and its entry in the vector $\boldsymbol{d}$ is switched from 0 to 1, its counterfactual utility $v^\dagger$ in the previous stage is invalidated, and its utility is calculated with and represented by v thereafter.

3.2 Utility of individual sensors: coverage & information entropy

Now we determine the specific form of $v(i|\boldsymbol{d})$ based on DAP. Per the above discussion, the straightforward idea is to regard the coverage of its DAP as the utility of a sensor x_i, i.e., the number of variables in the model whose time

[3] The definition of a scale imposes mathematical constraints for the construction of $U(\boldsymbol{X^d})$: first, it should satisfy that $U(\boldsymbol{0}) = 0$, i.e. an empty data availability vector (sensor set) should have zero utility. Next, U should be monotonously increasing with respect to the elements in $\boldsymbol{X^d}$, i.e. gathering more datasets on system variables will not decrease the aggregated value of the data-availability vector. This guarantees that the utility of a new sensor $v^\dagger(j|\boldsymbol{d})$ is always non-negative. With these two constraints, a relative scale defined over U could be established, and we could compare $v^\dagger(j_1|\boldsymbol{d})$ and $v^\dagger(j_2|\boldsymbol{d})$ for arbitrary j_1 and j_2, and such relationships should be transitive:

$$v^\dagger(j_1|\boldsymbol{d}) > v^\dagger(j_2|\boldsymbol{d}), v^\dagger(j_2|\boldsymbol{d}) > v^\dagger(j_3|\boldsymbol{d}) \Longrightarrow v^\dagger(j_1|\boldsymbol{d}) > v^\dagger(j_3|\boldsymbol{d}). \tag{12}$$

However, this comparison is only made on the relative basis. In order that the absolute value of $v^\dagger(j|\boldsymbol{d})$ is meaningful, U must be scalable, which is not satisfied on our final construction of $U(\boldsymbol{X^d})$ (see Section 3.3); hence we are not able to use $v^\dagger$ to answer such questions as, how many times the utility of sensor x_i is that of x_j.

series could be simulated after x_i has been calibrated. This number is given by $|\boldsymbol{\beta}_i|+1$, including all the unmeasured variables in its DAP tree plus variable x_i itself:

$$v(i|\boldsymbol{d}) = |\boldsymbol{\beta}_i| + 1. \tag{14}$$

Seemingly simplistic, the idea of coverage can nevertheless be understood by way of information entropy, and our formulation essentially assimilates the insights of the two existing solutions discussed. This is because for each unmeasured variable, after it is calibrated within a certain DAP, the uncertainty of its time series (which are indeterministic and could take arbitrary values if the variable is not calibrated) is reduced to zero, i.e., resulting in a reduction of information entropy. Due to the nonlinearity of the system and the indefinite value range for different model variables, it is then feasible if we make the simplified assumption that the reduction of information entropy is a uniform ΔH_0 on each calibrated variable. Hence maximizing the reduced information entropy H_{cal} through model calibration is therefore equivalent to maximizing the coverage of nodes, i.e,:

$$H_{cal} = (|\boldsymbol{\beta}_i| + 1)\Delta H_0. \tag{15}$$

This entropy perspective of our formulation of $v(i|\boldsymbol{d})$ is conceptually consolidating and sets up the base for further discussion of the extended form of v (see following).

With sensor's individual utility pinned down, the aggregated utility of a specific data-availability condition $\boldsymbol{d}$ (i.e., a specific placement configuration of sensors) is then the maximum number of variables in the system that could be covered:

$$U(\boldsymbol{d}) = \underset{s\in\mathcal{S}}{\operatorname{argmax}} \sum_i v(i|\boldsymbol{d}), \quad \forall i \ s.t. \ d_i = 1 \tag{16}$$

Note here the sequence $s \in \mathcal{S}$ of taking the sum of $v(i|\boldsymbol{d})$, i.e. the sequence of conducting DAP partitions for sensors plays a role in determining U, and U equals the largest value of the sum of individual DAP utilities $\sum_i v(i|\boldsymbol{d})$ across different sequences. In cases where an upstream variable is linked to multiple downstream variables, one variable may appear in multiple DAP trees; yet it is covered at the first DAP graph in which it appears, and thus does not belong to the $\boldsymbol{\beta}$ in the other DAPs [4] (Figure 3). As we do not double-count the coverage of a sensor by other sensors ($\boldsymbol{x}_i^d$ and $\boldsymbol{\beta}_i$ are distinct), the number of DAPs to be conducted does not change with different sequences; thus the set of possible DAP sequences $\mathcal{S}$ is the permutation of $\boldsymbol{X}^{\boldsymbol{d}}$. Searching over this permutation, $U(\boldsymbol{d})$ equals the maximum aggregated utility $\sum_i v(i|\boldsymbol{d})$ produced by the optimal sequence of DAPs. [5]

Notably, the variance of $\sum_i v(i|\boldsymbol{d})$ across different sequences $\boldsymbol{s}$ (which is not always computable though) reflects the topological feature of the model graph. An observation follows immediately:

Proposition If $\sum_i v(i|\boldsymbol{d})$ remains invariant for all sequences in $\mathcal{S}$ (permutation of $\boldsymbol{x_d}$), then the system graph has a quasi-tree structure, in which the out-degree of unmeasured nodes corresponding to the 0-entries in $\boldsymbol{d}$ is always 1, except for the topmost node whose has 0 out-degree. Conversely and more importantly, if the system graph is a tree, $\sum_i v(i|\boldsymbol{d})$ always remains invariant across all sequences in $\mathcal{S}$.

Importantly, this proposition implies that for systems having tree structures, since the sequence of DAPs does not matter in these cases, the calculation of $U(\boldsymbol{d})$ could always be conducted in a sequential manner, and then the sequentially optimal solution is now global optimal (see Section 7). For the same reason, we are able to derive some analytical results during the computation of $U(\boldsymbol{d})$ for tree structures (see Section 6). On the other hand, contrary to the situation in the proposition, for graphs that have a large average out-degree of nodes, it is expected that $\sum_i v(i|\boldsymbol{d})$ may have a large variance across different sequences. In these cases, nodes have large out-degrees are often key nodes in the model that appear in the formulation of multiple downstream variables; it is then welcomed that the datasets on these variables are acquired in model calibration. This notion implies an alternative idea to characterize variable's individual utilities $v(i)$ based on nodes' centrality measures (see Section 5).

One advantage of using the simplistic form (14) for sensors' individual utility is that it is scalable, and the U constructed from this form of $v(i|\boldsymbol{d})$ is thus also scalable. [6] However, using shear coverage as the utility term neglects much information from the DAP. Consider two DAP partitions that have the same number of $\boldsymbol{\beta}$, while one has $|\boldsymbol{\alpha}| = 1$ and the other has $|\boldsymbol{\alpha}| = 2$. To calibrate this DAP, in the first case one only tunes one parameter, while in the second case one has access to two parameters. Therefore, arguably, the calibration process would be more

[4] Algorithmically, this requires an iterative reduction of the model graph after each round of DAP identification, in which the unmeasured nodes on the newly obtained DAP are eliminated from the graph to prevent double-counting.

[5] The sequence does not matter if we are concerning the shear coverage of system variables (as in (14)), since the aggregated coverage should be the same across all sequences; hence in this case the argmax could be dropped in (16). However, this extra search space is non-trivial for more contemplated forms of $v(i|\boldsymbol{d})$ (see (20)-(22)).

[6] As discussed in the previous footnote, here one is able to compare $v^\dagger(i|\boldsymbol{d})$ and $v^\dagger(j|\boldsymbol{d})$ by their absolute values.

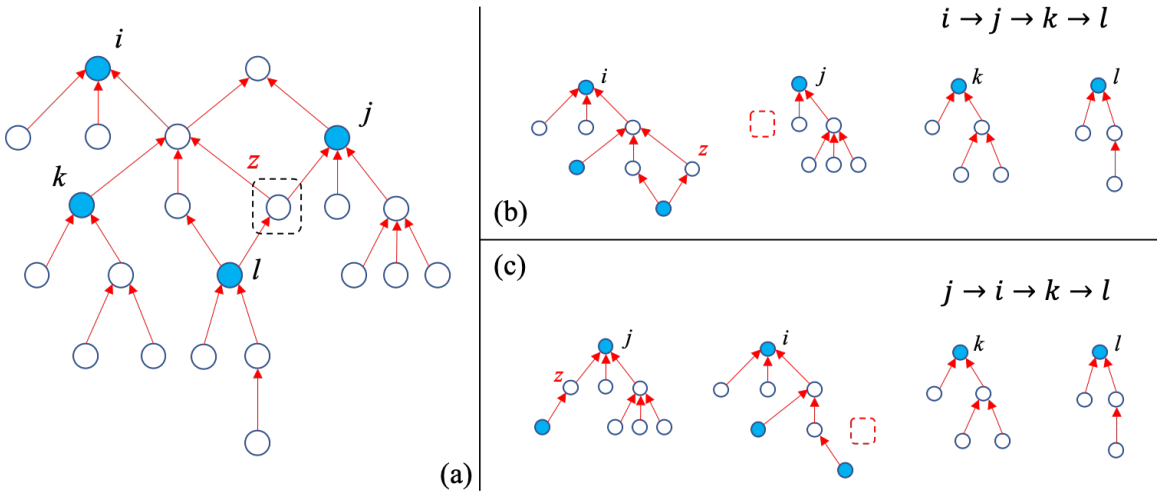


Figure 3: The sequence of DAPs conducted plays a role in the evaluation of the objective function U. (a) Sample system graph with four sensors {i,j,k,l} (filled nodes) which leads to four corresponding DAP trees in a sequence. (b) and (c): Two candidate sequences of DAP. Node z may belong to the DAP of either i or j; yet z can only appear in the first DAP that contains it and is missing in the second DAP (dashed box). Therefore, the two sequences $i \to j \to k \to l$ and $j \to i \to k \to l$ represent two distinct strategies s_1 and s_2 in the strategy space $\mathcal{S}$ (see Section 4).

successful in the first scenario than in the second scenario, since in the single-parameter scenario (one degree of freedom) the calibration result is deterministic: minimizing the difference between the simulation trajectory and the data series for the target variable leads to a single optimal value for the parameter and determined dynamics for unmeasured variables. This is not the case for the two-parameter scenario; indeed, the more parameters at hand, the less deterministic and reliable the calibration results would be, since with many parameters one is very likely to end up in a local optimum during model calibration, as it is often the case that the number of local minima in the parameter space grows with the size of the space. Note it is true that the more parameters are available, the more probable that the data series could be well fit; however, the goodness of fit does not project into the reliability of the calibration. In fact, as mentioned before, one repeated critique for simulation models in social sciences is the excessive number of parameters in the model that renders all trajectories reachable; modelers are always rewarded for using a small set of parameters that could largely constrain model dynamics.

These discussions suggest an improved form for $v(i|\boldsymbol{d})$, taking into consideration the number of parameters (degrees of freedom) used in the calibration of DAPs, which could be well understood from the entropy perspective. In model calibration, assume the number of local minima N_{lm} grows exponentially with the size of the parameter space (i.e., DOF of DAPs):

$$N_{lm} \propto e^{c \times DOF}. \tag{17}$$

Assume equal likelihood $p = 1/N_{lm}$ of each local minimum, then the entropy around local minima is:

$$H_{lm} = -\sum_{N_{lm}} p \log p = \log(N_{lm}) = c \times DOF. \tag{18}$$

Recall the entropy reduction through calibration H_{cal} in (15). To increase the coverage of, while reduce the uncertainty in, the model calibration process, we want to maximize H_{cal} and minimize H_{lm}. Since the two entropy terms are defined on different basis and thus not additive, we construct the utility function of a DAP from these two concerns as:

$$v(i|\boldsymbol{d}) \sim \frac{H_{cal}}{H_{lm}} = \frac{(|\boldsymbol{\beta}_i| + 1)\Delta H_0}{c \times |\boldsymbol{\alpha}_i|} \sim \frac{(|\boldsymbol{\beta}_i| + 1)}{|\boldsymbol{\alpha}_i|} \tag{19}$$

since the DOF of a DAP is given by the $|\boldsymbol{\alpha}_i|$ in its output. Adding a parameter c controlling the relative importance of the two terms, we arrive at:

$$v(i|\boldsymbol{d}) = \frac{(|\boldsymbol{\beta}_i| + 1)^c}{|\boldsymbol{\alpha}_i|}. \tag{20}$$

A few exceptions take place under this formulation. If the measured variable x_i itself is a leaf node without incidents, then in its DAP, $|\boldsymbol{\alpha}_i| = 0$ in the denominator. In this case, we allow that its utility is a small constant: $v(i|\boldsymbol{d}) = \epsilon \sim 0$. On other occasions, a DAP tree may contain no parameter while the measured variable i is not a leaf node; this corresponds to the situation where the target variable could not be calibrated at all, since no parameter is available to be tuned. In this case, we decide that $v(i|\boldsymbol{d}) = 0$, i.e., the measurement on this variable has zero utility for model calibration. [7]

3.3 Varied quality of real-world datasets

So far, we have assumed implicitly that the acquired datasets on different variables are of uniform quality. As mentioned earlier, this assumption is almost always not valid in practice and the quality of real-world datasets varies a lot. To ensure that our solution is operational in real application, the quality of datasets needs to be considered in their utility.

Generally speaking, data quality in social sciences could be decomposed into quantity (i.e. the length of time series) and reliability (e.g., of data sources). To characterize these two aspects in our formulation, we associate each variable x_i with two additional metrics, the length l_i, and the reliability r_i of its (potential) datasets. The data-availability vector $\boldsymbol{d}$ is thus accompanied by two other vectors of the same size, the data length vector $\boldsymbol{l}$ and the data reliability vector $\boldsymbol{r}$. Now, when we collect a piece of time series d_i for model variable x_i, we indicate the quality of this piece of data by two scalars l_i and r_i that both fall within [0,1]. First, l_i is the length of this real-world dataset normalized by the simulation time range of the model (thus falling into $[0, 1]$). It is almost always the case empirically that the acquired data are not of sufficient length compared with the model horizon (e.g., the simulation dates back too far into history, or projects too far into the future), or that the data are not of sufficient resolution (e.g., sampled from a larger time interval compared with the model simulation step); in either case, l_i is strictly below 1. Second, as in most social science studies the modeler is able to (and often is asked to) determine the reliability of the datasets he has collected in a straightforward way, we achieve this by assigning a score r_i for each piece of data, which indicates its probability of being reliable; for example, $r_i = 0.8$ indicates that the dataset on x_i is reliable with 80% probability. In practice, r_i could be labelled manually by the modeler as in expert judgments (e.g., *Cooke*, 1991), a straightforward and operational treatment in social science studies. The value of r_i could be related to the level of noisiness in the collected time series (e.g., datasets having larger variance are less reliable), or it could reflect the reliability of the source of information (e.g., data from government announcements are more reliable than from private sources). One might also adopt the method of probabilistic inversion (*Kraan and Bedford*, 2005) in determining r_i from expert judgments. This labeling effort for $\boldsymbol{r}$ is similar to the method used in *Kersebaum et. al* (2015), where variables are assigned with different "relevance" to the model according to expert judgments; however, arguably, here in our study the evaluation of datasets' reliability is more objective than the evaluation of data's "relevance", and in many cases the modeler could indeed have external quantitative information to decide the values for $\boldsymbol{r}$. Last to note, as mentioned previously, the use of two scores l_i and r_i to indicate the quality of datasets does not dig into the noise structure (e.g., system or measurement errors) of the data, or the sensitivity analysis in parameter estimation; using the scores essentially simplifies these discussions and draw our considerations of the varied quality of real datasets to be compatible with the established algorithmic framework.

The quality of data on a specific sensor x_j could now be captured with $r_j l_j$. To incorporate data quality as an additional term in a sensor's utility $v(i|\boldsymbol{d})$, an idea is that the utility is proportional to the minimum quality of all measured variables that appear in the DAP graph for x_i, which include variables in the set $\boldsymbol{x}_i^d$ and the target variable x_i itself:

$$v(i|\boldsymbol{d}) = \frac{(|\boldsymbol{\beta}_i| + 1)^c}{|\boldsymbol{\alpha}_i|} \min_{j \in [\boldsymbol{x}_i^d, x_i]} (r_j l_j). \tag{21}$$

Since at least x_i is in the set, the new term will always exist. However, this straightforward formulation neglects the structural feature of the underlying system model. As one could imagine, depending on the number of upstream nodes in the model graph that it leads to, for a specific measured variable, its dataset can be of different utilities under different model topologies. Even when its data is of poor quality, the variable will not do much harm to model calibration if it only appears in the formulations of few other variables.

[7] Also note that when $|\boldsymbol{\alpha}_i|$ is added to the formulation of $v(i|\boldsymbol{d})$, the non-negativity constraint for $v^\dagger(i|\boldsymbol{d})$ might be compromised in some cases. This will happen whenever we try to measure a parameter (i.e., i in $v^\dagger(i|\boldsymbol{d})$ is a parameter), which might make some DAP trees relying on this parameter become untunable and thus of zero utility, and therefore the aggregated U will instead be reduced when adding in this new measurement (see (11)). Hard constraints are imposed in these cases to ensure $v^\dagger(i|\boldsymbol{d}) \geq 0$.

With this notion in mind, an improved idea is to assign reliability scores for all model variables, not only those measured variables $\boldsymbol{x_d}$, and then the data-reliability of a DAP could now be indicated by the average reliability score of all nodes in the DAP $\bar{r}_i = \text{ave}_{j\in[\boldsymbol{x}_i^d,\boldsymbol{\beta}_i,x_i]}(r_j)$, instead of the minimum score. That is:

$$v(i|\boldsymbol{d}) = \frac{(|\boldsymbol{\beta}_i|+1)^c}{|\boldsymbol{\alpha}_i|}\text{min}_{j\in[\boldsymbol{x}_i^d,x_i]}(l_j)\text{ave}_{j\in[\boldsymbol{x}_i^d,\boldsymbol{\beta}_i,x_i]}(r_j). \tag{22}$$

Expectedly, the above formulation will be smoother than using the minimum r_j to represent data-reliability of the DAP graph. This treatment requires a pre-processing step that is able to successfully assign r_i to all model variables. If the underlying graph has a tree structure (otherwise, see below), the following algorithm could accomplish this task: first, the reliability scores for measured (i.e., data-available) variables are manually labelled by the modeler; then, for variables whose data are not available, its reliability score is 1 if it is a leaf node, otherwise, score r_i is the product of the scores of its incident variables, $r_i = \prod_{x_j\in[a_{ji}=1]} r_j$. This algorithm will guarantee the assignment of a non-zero r_i to all model variables (assuming that the labelled r_i are not zero).

Notably, the resulting $\boldsymbol{r}$ vector is also useful for the two translated methods. For the information entropy approach, the variance of each variable $\sigma^2_{\boldsymbol{x}}$ could readily be determined by the inverse of $\boldsymbol{r}$ if one does not want to use a flat variance: $\sigma^2_{\boldsymbol{x}} = 1/\boldsymbol{r}$, which completes the construction of $\boldsymbol{Q}$. For the miss probability approach, the reliability of x_i could be built into $\hat{m}_i$ in a way similar to our formulation, such that the miss probability $\hat{m}_i$ will increase when the reliability of sensors detecting x_i is low. That is:

$$\hat{m}_i = (1-r_i)\frac{|\boldsymbol{\alpha}_i|}{|\boldsymbol{x}_i^d|+|\boldsymbol{\alpha}_i|}. \tag{23}$$

Moreover, echoing with earlier statements, the advantage of constructing the objective U via a first definition of the utility term v for each individual sensor could be demonstrated in the flexibility of incorporating additional terms in the utility function, as one could see here in the case of data quality.

3.4 Dealing with feedback loops

In the above discussions we have assumed that the model graph (and thus the DAP graphs) contains no loops. However, it is very common that system models in social and economic sciences may contain feedback loops, although many domain-specific models do have acyclic structures such as the crops models. If loops contain measured nodes, the boundaries of the backtracking DAP graphs will still be exclusively either leaf nodes or measured nodes, and the above algorithm for $\boldsymbol{r}$ would work with no further treatment needed; in the case when there exists loops in the model graph that contain no measured node at all, a treatment is implemented in the algorithm. In these situations, to prevent loopy calculations in determining r_i as well as loopy backtracking in the determination of DAPs [8], one node in each such loop (i.e., without measured nodes) needs to be identified and be manually assigned a reliability score, as if it is a measured variable. [9] To accomplish this, we identify a feedback vertex set (FVS) (*Festa et al.*, 1999) in the graph, with which each loop has a node associated, and let the nodes in the FVS which do not belong to $\boldsymbol{x}_i^d$ be viewed as the pseudo measured nodes that we are looking for. Since the identification of a minimum FVS is NP-complete (*Karp*, 1972), we thus aim for an effective, instead of minimum FVS, possibly through the following algorithm: starting from the longest cycle in the graph and proceeding with cycles in a descending order of length, if no vertex of the cycle in discussion belongs to the temporary FVS, we add the vertex of the cycle that has the largest degree to the existing FVS and proceed. In the end, with this additional treatment based on FVS, the successful assignment of reliability scores $\boldsymbol{r}$ to all model variables and the finite and deterministic backtracking of DAPs are guaranteed for arbitrary graph topologies, either with or without feedback loops, and our algorithm applies to system models of both sorts.

4 Sensor Placement and the Value of Side Information

From a high level, our solution to the sensor placement problem in this study could be understood in the theoretical framework of the value of side information (e.g. *Rinehart and Dahleh*, 2011). Basically, the placement $\boldsymbol{d}$ of sensors can be viewed as the side information Y for the system model, whose value (utility) $U(Y) = U(\boldsymbol{d})$ is determined

[8] Loopy calculations arise as the backtracking in determining DAP or r_i never stops in a loop: a node traces back to itself after a number of steps, and the calculation proceeds.

[9] This pseudo measured variable will also be added to the denominator of (23) in determining $\hat{m}_i$.

by a specific objective function h, based on the graph's inherent information W and a decision x. In (*Rinehart and Dahleh*, 2011) the optimization problem is formulated as:

$$U(Y) = \min_{x \in \mathcal{X}} h(x, W|Y). \tag{24}$$

Under this formulation, the add-on utility of new sensors in our problem setting is essentially:

$$v^{\dagger}(i|\boldsymbol{d}) = v^{\dagger}(\Delta Y|Y) = U(Y + \Delta Y) - U(Y). \tag{25}$$

The original setting in *Rinehart and Dahleh* (2011) needs to be slightly rephrased to the problem setting of our study. First, in *Rinehart and Dahleh* (2011), the information W is the realization of the graph's random edge weights, whereas in the current study we do not focus on edge weights but rather the structural features of the system, i.e., the connectivity $\boldsymbol{A}$; edges are unweighted in our problem. Moreover, in the original problem, the objective function is formulated by taking expectation over W; this is overlooked in our formulation of U as in the current scheme we do not consider stochasticity on edge weights or on graph connectivities. Second, the original formulation (24) directly applies to our new objective function (16), but needs to be simplified when applying to the two translated objective functions. This is because in these two approaches, once a placement $\boldsymbol{d}$ is considered, there is no further decision x to be made; by contrast, in our approach, the objective function U needs to optimize over the sequence $\mathcal{S}$ on $\boldsymbol{d}$, with s playing the role of x. If assuming that the strategy space is not explicit and is averaged out in the utility of side information (as in (2) of *Rinehart and Dahleh* (2011)), then the two translated approaches could also be cast in this framework. Essentially, in *Rinehart and Dahleh* (2011), information optimization is equivalent to the strategic *design* of the covariance matrix that contains the side information, while in our problem, the covariance matrix is fixed, whereas the side information $\boldsymbol{d}$ as the placement of sensors decides the constrained *selection* on the covariance matrix that contributes to the entropy of the system.

Overall, adopting the notations of the current study, (24) could be directly paraphrased to:

$$U(\boldsymbol{d}) = \min_{s \in \mathcal{S}} h(s, \boldsymbol{A}, \boldsymbol{\theta}|\boldsymbol{d}), \tag{26}$$

with $h = -\sum_i v(i|\boldsymbol{d}), \quad \forall i \ s.t. \ d_i = 1$. Moreover, the capacity constraint C in our case is the number of sensors k to be placed. The constraint set $\Gamma(k)$ (as in *Rinehart and Dahleh* (2011)) is thus ($C = k$):

$$\Gamma(C) = \Gamma(k) = \{\boldsymbol{d}| \ ||\boldsymbol{d}||_1 \leq k\}, \tag{27}$$

and we optimize over the constraint set to search for the best placement $\boldsymbol{d}$, i.e., the side information of the largest value:

$$U(\boldsymbol{d}_{best}) = \max_{\boldsymbol{d} \in \Gamma(k)} U(\boldsymbol{d}, \boldsymbol{A}, \boldsymbol{\theta}). \tag{28}$$

One notes that equation (1) is recovered, and our problem setting is successfully embedded in the broader theoretical framework of *Rinehart and Dahleh* (2011).

5 DAP Complement Graph and Node Centralities

In this study, we formulate a new objective function $U(\boldsymbol{X}^{\boldsymbol{d}})$ for the defined sensor placement problem constructed from variables' individual utilities $v(i|\boldsymbol{d})$, relying on the idea of DAP of system models. A DAP graph backtracks the predecessors of a target variable, the topmost node of the DAP tree; the probe ends where the visited variable is measured (or pseudo-measured), or it is a leaf node of the graph. Intrinsically, this method focuses on the *in-degree* of nodes in the directed model graph. By intuition, one might raise a different idea when first gets exposed to this problem: the utility of a target node could also be related to its *out-degree*, i.e., the utility of a measured variable could be determined by how many unmeasured variables in the system it could lead to, including both its immediate out-neighbors and more distant nodes it could further reach. To some extent, this forward-tracking idea may seem more natural than the backtracking approach, since the measurement on incident nodes will surely contribute to the determination of the unmeasured variables these incidents target to; clearly, if all incidents of an unmeasured node are measured, this node could then be fully determined. Technically, these two perspectives of the problem are complementary to each other, and the forward-tracking graph could be viewed as the complementary graph G'_d to the DAP graph G_d: the backtracking G_d counts the number of unmeasured variables that could be covered through the calibration of the measured x_i, and the forward-tracking G'_d counts the number of contributions that x_i is able to make for its unmeasured successors.

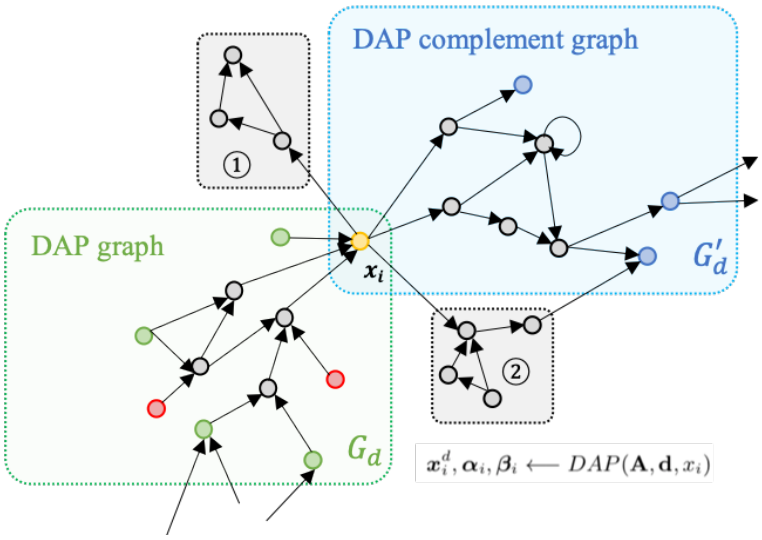


Figure 4: DAP graph G_d and its complement graph G'_d. Small boxes showing two situations where an outgoing link from the target variable x_i does not belong to the forward-tracking graph G'_d.

Nevertheless, although the two points of view are complementary to each other, there are a few caveats that favor our choice of the backtracking idea in the algorithmic design. It is important to recognize that, unlike in G_d where all the incident nodes of the target node x_i are included in the graph, some outgoing links of x_i do not belong to the forward-tracking G'_d (Figure 4). This may happen in two situations: either the outgoing link leads to a sub-structure that does not have measured nodes as endings (case 1 in Figure 4), or the sub-structure does end at measured nodes, but having side components along the way (case 2 in Figure 4). In both cases, there are some variables in the structure extending from the outgoing links of x_i that are not going to have deterministic time series out of calibration. Only those pathways from x_i that end at other measured nodes without side components along the way should be counted as x_i's contributions and thus belong to the complementary graph G'_d. As a result, determining G'_d from the complete system graph requires both forward and backward tracing, which is prohibitive in algorithmic design; by contrast, determining G_d only requires backward searching (e.g., breadth-first search in (*Oliva*, 2004)) and thus is more feasible in computation. Moreover, in the forward-tracking scheme, it is not straightforward to incorporate the number of parameters $|\boldsymbol{\alpha}_i|$ in the utility $v(i|\boldsymbol{d})$, since different parts of the graph G'_d may be calibrated by different numbers of parameters; only in the backtracking tree G_d could $|\boldsymbol{\alpha}_i|$ be representative of the degree of freedom. Needless to say, due to its irregular structure, it is even more difficult on G'_d to consider data quality factors in utility terms.

The idea of tracking the target node's in- or out-neighbors to represent its utility points to nodes' centrality measures on graphs. By intuition, the data on a certain variable will be very useful if this variable has many neighbors in the model graph, i.e., it is a key node. However, we have shown so far that such an intuition might be incorrect, since the out-degree does not always count (Figure 4), and even when only considering the in-degree, it is possible that one node has few 1-step incidents but have many remote neighbors, and the calibration on this variable may still be very useful for the model. Therefore, degree-based centrality measures, such as in- or out-degree centrality or Katz centrality (we denote with v_i^{in}, v_i^{out} and v_i^{Katz}), are not good candidates for $v(i|\boldsymbol{d})$, although they may seem to be more straightforward solutions and are much easier to compute. Intrinsically, if pure node centralities are adopted, they are unconditional utility measures for the target variable, i.e., $v(i)$ instead of $v(i|\boldsymbol{d})$, whose calculation relies exclusively on the model's adjacency matrix $\boldsymbol{A}$, but not the pre-existing data-availability condition $\boldsymbol{d}$. Comparison of $v(i|\boldsymbol{d})$ with degree-based $v(i)$ is demonstrated on a sample model structure, and results confirm the advantage of our construction of $v(i|\boldsymbol{d})$ (see Section 8).

6 Analytical Results for Tree Structures

Given that for tree graphs the sequential optimal solution is always global optimal under our new objective function U, as derived from the Proposition, we are able to derive some analytical results for the optimal sensor placement configuration on tree structures, assuming no pre-existing sensors in the system. We consider the formulation of

$v(i|\boldsymbol{d})$ at (20) in our derivations, ignoring the discussion on the quality of datasets. We consider $c = 1$ and $\epsilon = 0$, assuming that all leaf nodes are parameters and sensors placed on leaf nodes have zero utility on their own; however, the placement on leafs will increase the utility of other sensors by decreasing the degree of freedom on other DAP trees.

We start with binary trees, then extend the results to multi-ary trees, and finally discuss general tree structures. It is almost impossible for real system models to have strict binary or multi-ary tree structures; however, the analytical discussion on regular trees provides theoretical bounds for general tree structures, and sets the ground for further derivations. Throughout the discussion, the idea of DAP trees is frequently visited.

6.1 Binary trees

Suppose a binary tree with L layers (starting from the top; $l = 1$) and thus $2^L - 1$ nodes, and k sensors are to be placed. For an L-layer binary tree, there are 2^{L-1} leaf nodes (parameters): $\sum_i |\boldsymbol{\alpha}_i| = 2^{L-1}$, and the number of unmeasured variables (no pre-existing measurement) is $\sum_i (|\boldsymbol{\beta}_i| + 1) = 2^{L-1} - 1$. Therefore, the problem is to find the optimal placement configuration $\boldsymbol{d}$ of the k sensors so as to maximize

$$U(k) = \underset{\boldsymbol{d}}{\operatorname{argmax}} \sum_i \frac{|\boldsymbol{\beta}_i| + 1}{|\boldsymbol{\alpha}_i|}, \quad s.t.\ ||\boldsymbol{d}||_1 = k, \tag{29}$$

by effectively partitioning the 2^{L-1} leaf nodes and $2^{L-1} - 1$ non-leaf nodes among the k DAP trees.

As mentioned, we study the optimal placement sequentially. We denote by $\hat{U}_k^l$, the utility of a placement of the kth sensor at layer l (after the sequentially optimal placement of the past $k-1$ sensors). Therefore, for each value of k, we have:

$$U(k) = \max_l \hat{U}_k^l, \quad l = 1, 2, ..., L, \tag{30}$$

i.e., $U(k)$ is optimizing the placement of the kth sensor over different layers. For a specific k, $\hat{U}_k^l$ consists of k terms in the summation of $\sum_i \frac{|\boldsymbol{\beta}_i|+1}{|\boldsymbol{\alpha}_i|}$, calculated on the k DAP trees. We further denote the numerator and the denominator of each term, and have:

$$\hat{U}_k^l = \frac{\hat{N}_k^1}{\hat{D}_k^1} + \frac{\hat{N}_k^2}{\hat{D}_k^2} + ... + \frac{\hat{N}_k^k}{\hat{D}_k^k}, \tag{31}$$

where each $\hat{N}$ and $\hat{D}$ is a function of l. Moreover, in a consistent manner, for the optimal $\hat{U}_k^l$ as in $U(k)$, we note $U(k) = \frac{N_k^1}{D_k^1} + \frac{N_k^2}{D_k^2} + ... + \frac{N_k^k}{D_k^k}$.

When $k = 1$, the single sensor could be placed on any of the L layers. We have

$$\hat{U}_1^l = \frac{\hat{N}_1^1}{\hat{D}_1^1} = \frac{2^{L-l} - 1}{2^{L-l}}. \tag{32}$$

Thus

$$U(1) = \max_l \hat{U}_1^l = \frac{N_1^1}{D_1^1} = \frac{2^{L-1} - 1}{2^{L-1}}, \tag{33}$$

i.e., the best placement is to place it at the topmost node ($l = 1$).

Next, when $k = 2$, we are placing the second sensor at one of the layers $l = 2, ..., L$, with the first sensor placed at the optimal location in the previous round. The effect of the second sensor, placed at layer l, is to subtract 2^{L-l} from the denominator of $U(1)$ (i.e., D_1^1), and subtract $2^{L-l} - 1$ from the numerator (i.e., N_1^1), and the two subtractions make a new term:

$$\hat{U}_2^l = \frac{\hat{N}_2^1}{\hat{D}_2^1} + \frac{\hat{N}_2^2}{\hat{D}_2^2} = \frac{N_1^1 - (2^{L-l} - 1)}{D_1^1 - 2^{L-l}} + \frac{2^{L-l} - 1}{2^{L-l}} = \frac{2^{L-1} - 1 - (2^{L-l} - 1)}{2^{L-1} - 2^{L-l}} + \frac{2^{L-l} - 1}{2^{L-l}} = 1 + \frac{2^{L-l} - 1}{2^{L-l}}, \tag{34}$$

with $l = 2, ..., L$. Therefore we have

$$U(2) = \max_l \hat{U}_2^l = \frac{N_2^1}{D_2^1} + \frac{N_2^2}{D_2^2} = \frac{2^{L-1} - 1 - (2^{L-2} - 1)}{2^{L-1} - 2^{L-2}} + \frac{2^{L-2} - 1}{2^{L-2}} = 1 + \frac{2^{L-2} - 1}{2^{L-2}}. \tag{35}$$

When $k = 3$, the effect of the third sensor is to make similar subtractions: 2^{L-l} from the denominator, and $2^{L-l} - 1$ from the numerator, from one of the two terms in $U(2)$. This corresponds to placing the third sensor in one of the

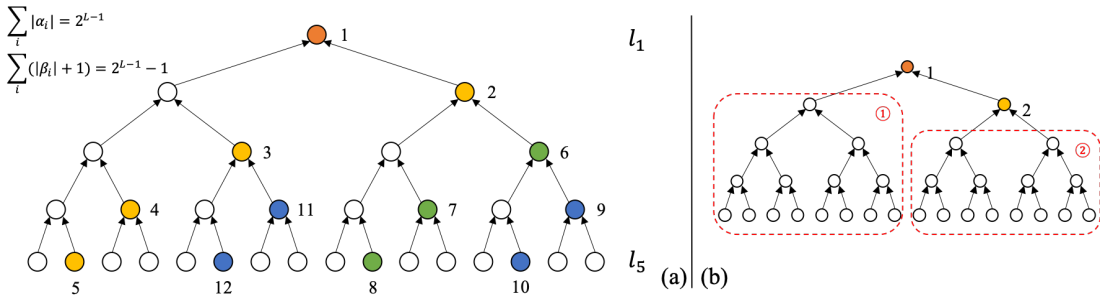


Figure 5: Sensor placement on a $L = 5$ binary tree. (a) Optimal placement for $k = 12$, with numbers labeling the sequence, and colors denoting periods of placement. (b) Two placement options at $k = 3$, on either of the two existing DAP trees.

two existing DAP trees led by the two placed sensors (dashed boxes in Figure 5b). Since D_2^1 is smaller than D_2^2, the better subtraction is on $\frac{N_2^1}{D_2^1}$, i.e., placing the third sensor in the first DAP tree. We have:

$$\hat{U}_3^l = \frac{\hat{N}_3^1}{\hat{D}_3^1} + \frac{\hat{N}_3^2}{\hat{D}_3^2} + \frac{\hat{N}_3^3}{\hat{D}_3^3} = \frac{N_2^1 - (2^{L-l} - 1)}{D_2^1 - 2^{L-l}} + \frac{N_2^2}{D_2^2} + \frac{2^{L-l} - 1}{2^{L-l}} = \frac{N_2^1 - (2^{L-l} - 1)}{D_2^1 - 2^{L-l}} + \frac{2^{L-2} - 1}{2^{L-2}} + \frac{2^{L-l} - 1}{2^{L-l}}, \tag{36}$$

and

$$U(3) = \max_l \hat{U}_3^l = \sum_{i=1}^{3} \frac{N_3^i}{D_3^i} = \frac{2^{L-1} - 1 - (2^{L-2} - 1) - (2^{L-3} - 1)}{2^{L-1} - 2^{L-2} - 2^{L-3}} + \frac{2^{L-2} - 1}{2^{L-2}} + \frac{2^{L-3} - 1}{2^{L-3}}. \tag{37}$$

One is able to see that, in a similar way, the effect of following sensors, placed in one of the existing DAP trees, is to split the selected tree into two sub-trees, and computationally, make subtractions on the numerator and denominator of the corresponding term, and add a new term at the end of $U(k-1)$. Before k exceeds L, the best strategy is always to split the first DAP tree, i.e., the deepest tree, because its denominator is getting smaller and smaller after rounds of subtractions. After $k = L$, however, this denominator is reduced to 1, and the deepest tree is saturated with sensors; so the next $L - 2$ sensors are to be placed in the second deepest tree, and after that the third deepest tree (twice, see below), and so on, hence an optimal sequence is established. Such an optimal placement sequence on a $L = 5$ binary tree is shown in Figure 5a.

Analytically, for $1 < k \leq L$, the increments of $U(k)$ are given by:

$$\Delta U(k) = 1 - \frac{1}{2^{L-(k-1)-1}} + \frac{(k-1)-1}{2^{L-(k-1)-1}} - \frac{(k-1)-2}{2^{L-(k-1)}} = 1 + \frac{(k-1)-2}{2^{L-(k-1)}}, \quad 1 < k \leq L. \tag{38}$$

Therefore,

$$U(k) = U(1) + \sum_{i=2}^{k} \Delta U(i) = \sum_{i=2}^{k} (1 - \frac{1}{2^{L-i}}) + \frac{(2^{L-1} - 1) - (2^{L-2} - 1) - \ldots - (2^{L-k} - 1)}{2^{L-1} - 2^{L-2} - \ldots - 2^{L-k}} = k + \frac{k-2}{2^{L-k}} - \frac{2^{k-1} - 1}{2^{L-2}} \tag{39}$$

For the next $L - 2$ sensors, the increments are determined in a similar way, and we have:

$$\Delta U(k) = 1 + \frac{(k-L)-2}{2^{L-(k-L)-1}}, \quad L < k \leq 2L - 2. \tag{40}$$

For the next $L - 3$ sensors, there are:

$$\Delta U(k) = 1 + \frac{(k-2L-2)-2}{2^{L-(k-2L-2)-2}}, \quad 2L - 2 < k \leq 3L - 5, \tag{41}$$

which repeats for another round while k keeps increasing since two optimal series of $L - 3$ sensors are available after the first $2L - 2$ sensors have been placed ({9,10} and {11,12} in Figure 5b), and one turns to the second series upon

finishing the first one. Similarly, 4 optional series are available for the next $L-4$ sensors, and 2^{m-2} options are available when $L-m$ sensors are to be placed in a subsequence. As a result, the $U(k)$ curve with a sufficient range of k is periodical, with discontinuities at $k = L, 2L-2, 3L-5, 4L-8....$ Around these discontinuities the average utility of sensors is a local maximum, and the envelope of these local maxima gradually reaches a limit that falls below 2 (Figure 6a-b). As L increases, the average utility of sensors increases, but is always upper bounded by 2 (Figure 6c). This bound 2 could be computed in the following way: during the ith period of sensors, $(L-i)$ sensors are placed, adding to the sequence; this batch of $L-i$ sensors all together produces an extra utility upper bounded by $L-i-1+L-i < 2(L-i)$. This leads to the upper bound $U(k)/k < 2$ of sensors' average utility when the average is calculated at the end of periods. Since the efficiency of placement is the highest at period ends, the upper bound of the average utility holds true at any time.

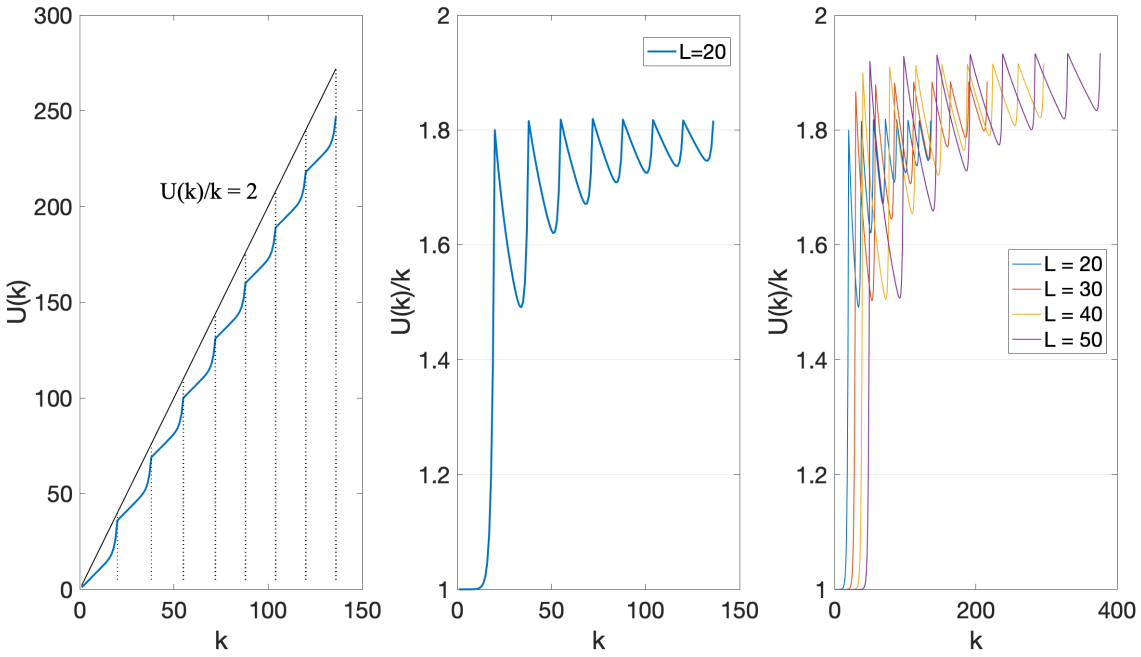


Figure 6: Analytical results of the optimal sensor placement on binary trees. (a) and (b): $L = 20$. $U(k)$ is below the reference curve $U(k) = 2k$. The curvature is periodical at discontinuities $k = L, 2L-2, 3L-5, 4L-8...$, where the average utility of sensors reaches local maxima. (c) Average utility of sensors under different values of L. As L increases, the envelope of local maxima is elevated and gradually reaches a limit below 2.

6.2 Multi-ary trees

The above results for binary trees can be generalized to w-ary trees where each non-leaf node has w incidents. Following similar derivations, when $k = 1$, (33) extends to:

$$U_w(1) = \frac{1}{w-1}\frac{w^{L-1}-1}{w^{L-1}}. \tag{42}$$

For $k > 1$, the computation of increments is slightly different. This is because for $w > 2$ trees, $w-1 > 1$ sensors are being placed at each layer in the optimal sequence. This is indeed following the abovementioned heuristic of finding the optimal location of the next sensor, which could be summarized as the following algorithmic rule: at each round, to determine the location of the next sensor in an optimal placement sequence, one always chooses the longest multiple-leaf DAP tree among all DAPs already exist in the graph, and then places the sensor at its earliest available branch extending from the root (top node) of the tree.

This rule determines the optimal sequence on binary trees (Figure 5a), and is further demonstrated in a $w = 3$ tree (Figure 7b). Going back to the computation of U, to generalize (39), we calculate the utility after the placement of

the $(w-1)$ sensors on a certain layer is finished, and thus obtain the utility sequence at $w-$intervals of k. We have:

$$
\begin{aligned}
&(w-1)U_w(1) = \frac{w^{L-1}-1}{w^{L-1}},\\
&(w-1)U_w[1+(w-1)] = (w-1)(1-\frac{1}{w^{L-2}}) + \frac{w^{L-2}-1+(w-1)}{w^{L-2}},\\
&(w-1)U_w[1+2(w-1)] = (w-1)(1-\frac{1}{w^{L-2}}) + (w-1)(1-\frac{1}{w^{L-3}}) + \frac{w^{L-3}-1+2(w-1)}{w^{L-3}},\\
&\ldots\\
&(w-1)U_w[1+\mu(w-1)] = (w-1)\sum_{i=1}^{\mu}(1-\frac{1}{w^{L-(i+1)}}) + \frac{w^{L-(\mu+1)}-1+\mu(w-1)}{w^{L-(\mu+1)}}.
\end{aligned}
\tag{43}
$$

Then

$$
U_w[k=1+\mu(w-1)] = \mu - \frac{1}{w-1}[(w+1)w^{\mu+1-L} - w^{2-L} - 1] + \frac{\mu}{w^{L-(\mu+1)}},
\tag{44}
$$

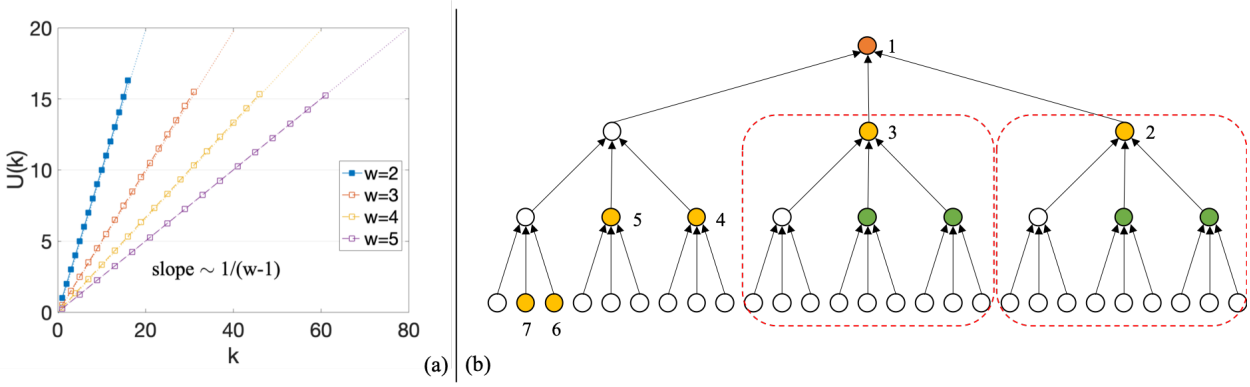


Figure 7: Results for w-ary trees. (a) In the $k < L$ scheme before the rapid increase near $k = L$, the slope of $U(k)$ as a function of k is approximately $1/(w-1)$. (b) Optimal placement sequence for a $w = 3$, $L = 4$ tree, notations similar to Figure 5. After the first $7 = (L-1)(w-1)+1$ sensors have been placed, one chooses either of the two existing longest multiple-leaf-trees (dashed boxes) and place the next two sensors (green).

which recovers (39) when $w = 2$. One could see that (Figure 7a), when w increases, the average utility of sensors decreases: in the $k < L$ scheme before the rapid increase near $k = L$, the slope of $U(k)$ as a function of k is approximately $1/(w-1)$. For the whole range of k, extending the discussion on the upper bound of sensors' average utility on binary trees, we have the following result:

Theorem 1 For a w-ary tree ($w \geq 2$), the average utility of sensors in an optimal placement configuration is upper bounded by $\frac{2}{w-1}$, i.e.,

$$
\frac{U(k)}{k} < \frac{2}{w-1}.
\tag{45}
$$

□

Proof Similar to the binary case, one may derive the upper bound for the average utility $U(k)/k$ for $w > 2$ cases. During the ith period of sensors, $(w-1)(L-i)$ more sensors are added to the sequence (yellow nodes in Figure 7b); these sensors, bringing in $(L-i)$ new DAP trees, all together produce extra utilities upper bounded by $(L-i)(w-1)/(w-1)+L-i < 2(L-i)$. Since the overall efficiency of placement is the highest at the end of every period, a universal upper bound $2/(w-1)$ for sensors' average utility can be concluded. ■

A series of nodes all of which only have one in-neighbor is sometimes called a *stem* (e.g., *Lin*, 1974). Thus for a tree graph that has no stem longer than 2, we have:

Corollary For a general tree structure which has no stem longer than 2, the average utility of all sensors in an optimal placement $U(k)/k$ is upper bounded by 2. □

This result points to the highest efficiency of binary structures in our sensor placement problem: if you have a fixed number of sensors to be placed in a system with arbitrary structures, the best topology to maximize the overall utility

of these sensors is (almost always) to formulate the system into a binary tree. Moreover, as an important heuristic in data acquisition strategies, one may conclude that, for models having w-ary tree structures with n variables in total, the most efficient number of sensors to be placed is around $m\log_w(n)$ with m being an integer (i.e., k is around the periods of $\log_w(n)$). Finally, the optimal placement sequence as described by the algorithmic rule, applies whenever $c \leq 1$ and $\epsilon \sim 0$, although the analytical results for $U(k)$ do not generalize to these extensions. Theorem 1 may hold true for non-zero ϵ, as long as it is sufficiently small.

6.3 General trees

Now we discuss results on general tree structures where a node could have multiple in-neighbors but only one out-neighbor, which is quite common among real system models. It is difficult to derive analytical results for $U(k)$ on a general tree graph, as one would expect; nevertheless, we are able to devise an effective algorithm (Algorithm 1) to determine the optimal placement sequence, whose idea is similar to the greedy tree-breaking algorithm in graph dismantling (*Braunstein et al.*, 2016). Suppose the tree G has n nodes and $m < n$ leaves. Since the utility of a DAP tree is non-zero only when at least one parameter exists in the tree, the model could possibly accommodate $k \leq m$ sensors. The model is determined as a directed acyclic graph (DAG) and one identifies its topological order (e.g., using Kahn's algorithm (*Kahn*, 1962)). To compute the utilities, each node i is associated with a two-entry vector $[N_i, D_i]$. In the initiation period, all leaf nodes are initiated with $[0, 1]$; then, along the topological order of the DAG, the vectors of upstream nodes are updated by aggregating the values of their incident nodes:

$$N_i = 1 + \sum_{j \ s.t.(i,j)\in E} N_j; \ D_i = \sum_{j \ s.t.(i,j)\in E} D_j, \tag{46}$$

up to the topmost node (Figure 8). After the initiation, at each of the k rounds, one identifies the node i that has the largest utility N_i^c/D_i, puts it in the sensor set as a successful placement. After the elimination of i from the original graph, for all the upstream nodes remaining in the DAG that node i could reach, values in node i's vector $[N_i, D_i]$ are subtracted from their vectors. The program iterates until all k sensors have been placed.

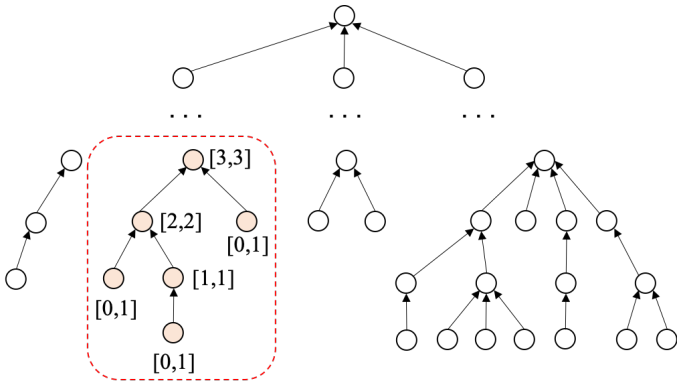


Figure 8: Illustration of the initialization procedure (46) of the optimal placement algorithm for general tree structures (Algorithm 1).

Algorithm 1 Optimal Sensor Placement on Tree Graphs

Graph $G = (\boldsymbol{X}, E)$. Number of sensors to be placed k. Set of placed sensors $\boldsymbol{X^d} = \{\}$.
Initialization
    Identify a topological order of graph G.
    Initiate the $[N_i, D_i]$ vectors of all leaf nodes with [0,1].
    Along the topological order, for node i: $N_i = 1 + \sum_{j \ s.t.(i,j)\in E} N_j$, $D_i = \sum_{j \ s.t.(i,j)\in E} D_j$
Sequential Placement
    while $|\boldsymbol{X^d}| < k$ **do**
        Find the node i of the highest utility N_i^c/D_i from $\boldsymbol{X}\backslash\boldsymbol{X^d}$. Put $i \to \boldsymbol{X^d}$.
        For all $j \in \boldsymbol{X}\backslash\boldsymbol{X^d}$ reachable from i: $N_j \leftarrow N_j - N_i$, $D_j \leftarrow D_j - D_i$.

Computationally, identifying the topological order of graph G is $O(n+|E|)$ [10], and the initiation along the DAG is $O(n)$. For the iterative update during placement, we have an upper bound for complexity (see Appendix for proof):

Theorem 2 The computational complexity of the Sequential Placement procedure in Algorithm 1 is upper bounded by $O(n\log_2(n))$. □

Therefore the entire placement algorithm is $O(n\log_2(n))$. The upper bound of complexity is reached on binary trees (see Appendix), and all other tree structures have a less computational cost. Taking together Theorem 1 and 2, a *balance* of maximizing sensors' utilities and minimizing the computational cost in determining the desired optimal placement for these sensors could be discovered (e.g., on binary trees), which is the inherent trade-off in this sensor placement problem.

7 Approximate Solution Schemes

For systems with arbitrary topologies where nodes may have multiple out-neighbors and there are possible loops in the graph, it is not guaranteed that the sequentially optimal solution is the global optimal solution. As an approximate solution scheme, similar to *Papadimitriou and Lombaert* (2012) and *Dhillon and Chakrabarty* (2003), we formulate the sequential optimal placement workflow under our new objective function (Algorithm 2) in companion with the combinatorial problem. The subroutine takes in a variable x_i that is currently unmeasured, and outputs its add-on utility $v^\dagger(i|\boldsymbol{d})$ based on the existing data availability $\boldsymbol{d}$. A sequentially optimal solution scheme could then be assembled based on this I/O workflow by sequentially adding the variable that has the largest add-on utility $v^\dagger$ among candidate variables to the existing sensor set $\boldsymbol{X^d}$.

Algorithm 2 Automatic Data Acquisition Strategy for System Models - Sequentially Optimal Subroutine

Initialization

Model adjacency matrix $\boldsymbol{A}$.

Data availability, reliability, length vector $\boldsymbol{d}, \boldsymbol{r}, \boldsymbol{l}$.

Determination of Reliability Scores

Identify an FVS of the system graph.

Calculate data reliability scores $\boldsymbol{r}$ for each variable in the system.

Individual Utility of Data-available Variables

for x_i *s.t.* $d_i = 1$ **do**

Conduct DAP on x_i: $\boldsymbol{x}_i^d, \boldsymbol{\alpha}_i, \boldsymbol{\beta}_i \longleftarrow DAP(\boldsymbol{A}, \boldsymbol{d}, x_i)$

Calculate utility $v(i|\boldsymbol{d})$.

Aggregate Utility of Data-availability Condition

Decide the set of DAP sequences $\mathcal{S}$ under data-availability $\boldsymbol{d}$.

Determine the maximum aggregate utility $U(\boldsymbol{X^d})$.

Input unmeasured variable x_j

Decide the set of DAP sequences with the new data-availability condition $[\boldsymbol{X^d}, x_j]$.

Calculate the updated utility $U([\boldsymbol{X^d}, x_j])$.

Output counterfactual/add-on utility of x_j: $v^\dagger(j|\boldsymbol{d}) = U([\boldsymbol{X^d}, x_j]) - U(\boldsymbol{X^d})$

As for global optima, brute force searching is costly even for models with a mid-range number (e.g., ~ 20) of variables; instead, a simulated annealing scheme could be implemented to solve the combinatorial problem. At each round t, one variable in $\boldsymbol{X^d}$ is randomly selected and replaced with a random new variable that is not in the *a priori* sensor set $\boldsymbol{X_0^d}$. This defines $\tilde{\boldsymbol{X}}^{\boldsymbol{d}}$, the neighborhood of $\boldsymbol{X^d}$. The acceptance is thus conditioned on:

$$\exp[-\frac{U(\boldsymbol{X^d}) - U(\tilde{\boldsymbol{X}}^{\boldsymbol{d}})}{h_s t}] > \text{random}[0,1] \tag{47}$$

with h_s controlling the cooling speed.

[10] Note the relation $|E| = n - 1$ for trees.

8 Example

We apply the sensor placement solution framework developed in this study to the sample model structure (Figure 1). The structure is simple and straightforward, consisting of 7 nodes (4 variables, 3 parameters) and 2 loops. A key reason to use this small model as the example is that the structure bears good symmetry, where variables B_r and D_r, B and D are non-distinguishable in the abstract system graph; therefore, placement solutions will be invalidated if these symmetries are not recovered in the results.

We compare the placement results of the three solution schemes discussed (PL2012, DC2003, our new method). For our method, (22) is used as the utility function v as we are considering the quality of data. Three scenarios of *a priori* data-availability conditions $\boldsymbol{X}_0^{\boldsymbol{d}}$ before the placement of sensors are explored: (1) when the model is currently sensor-free $\boldsymbol{X}_0^{\boldsymbol{d}} = \{0\}$; (2) when $\boldsymbol{X}_0^{\boldsymbol{d}} = \{B_r\}$; (3) when $\boldsymbol{X}_0^{\boldsymbol{d}} = \{B_r, P\}$. Under each scenario, we identify the $k = 1, 2, 3$ variables whose datasets will generate the largest utility for model calibration once they are acquired, in the sequential optimal sense as well as in the global optimal sense. The quality of datasets is assumed to be $r = 0.8$ and $l = 1$ for both pre-existing sensors as well as possible new sensors; the results are shown for two cases where $c = 1$ and $\epsilon = 0.1$ or 1 (Figure 9). Tests show that results are robust for various c, ϵ, r and l. Simulated annealing results ($h = 0.001$) agree with brute force search results (not shown). At the scenario $\boldsymbol{X}_0^{\boldsymbol{d}} = \{0\}$, multiple solutions may exist as global optima, reflecting the symmetries (B_r and D_r, B and D) of the model structure which are correctly revealed by the three algorithms; the symmetry is broken under the other two scenarios, as expected. For sequential optima, one optimal sequence is obtained for each case.

Results of the three approaches do not copy each other. For the information entropy approach, there is an exclusive highlight on variable P_d, which is selected into the optimal sensor set in almost all scenarios. This phenomenon is consistent with the theory since P_d is the single destination node of the directed graph (Figure 1), and so the calibration on P_d will reduce the entropy of the system to the maximum extent. For the miss probability approach, instead, there is a highlight on D_r (and potentially B_r as well), which are the two leaf nodes of the system. This indicates that sensors placed on these leaf nodes will reduce the average miss probability of the system by the largest margin, which agrees with the propagatory construction of nodes' miss probabilities. Overall, results from the two translated schemes echo with their theoretical build-up: the entropy approach is top-down and thus highlights the root node in the directed graph, whereas the probability approach is bottom-up and hence highlights the leaf nodes.

When the utility of leaf nodes is high ($\epsilon = 1$), the result of our new approach falls in between the above two results and highlights both ends (P_d and D_r). When leaf nodes' utility is small ($\epsilon = 0.1$), our placement highlights P_d to a moderate extent, and the result is the most deterministic (which is desirable in real modeling practice) among results of the three approaches without breaking the symmetry of the model structure. Moreover, unsurprisingly, our result is also the one that reflects the conditional nature of the problem to the greatest extent, as a critical feature built in its construction; for both the information entropy and the miss probability approaches, the output strategies are very much invariant across the three scenarios of *a priori* data-availability condition.

We further compare variables' individual utilities obtained in our solutions with their graph centrality measures (discussed in Section 5) under the scenario $\boldsymbol{X}_0^{\boldsymbol{d}} = \{B_r, P\}$ (Figure 10). Three degree-based centralities are considered: (1) in-degree centrality v^{in}, (2) out-degree centrality v^{out}, and (3) Katz centrality v^{Katz}. Results show that, expectedly, as for the utility $v(i|\boldsymbol{d})$ of existing sensors, P is more useful than the leaf B_r, and as for the add-on utility $v^{\dagger}(i|\boldsymbol{d})$ of new sensors, a placement on P_d is the most useful for model calibration. The results shown are at $\epsilon = 0.1$, but the relative strength of sensors' utilities remains invariant up to $\epsilon = 1$, thus the argument holds for all cases. One could see that graph centrality measures are not able to highlight existing and add-on sensor utilities at the same time, although they might be indicative to certain extent; results from our solution are more informative than these straightforward metrics.

9 Concluding Remarks

In this study, we embedded the determination of data-acquisition strategies for system models' calibration into the well-established theoretical framework of sensor placement. The problem in concern is formulated as a combinatorial optimization problem that searches for the optimal data-availability condition (i.e., placement configuration) regarding all model variables (i.e., sensors). Different objective functions are investigated for the optimization task. We first translate two existing approaches to the current setting, the information entropy (*Papadimitriou and Lombaert*, 2012) and the miss probability (*Dhillon and Chakrabarty*, 2003) approaches, both originated from sensor placement studies on physical dynamic systems. Relying on the concept of Data Availability Partition (*Oliva*, 2004), we then propose

Global optimal

Scenario	k	PL2012	DC2003	$\epsilon = 0.1$	$\epsilon = 1$
$x_d^0 = \{\}$	1	P_d	B_r; D_r	B; D	B; D
	2	$[B, P_d]$; $[D, P_d]$	$[B_r, D_r]$	$[B, D]$	$[B_r, B]$; $[B_r, D]$; $[D_r, B]$; $[D_r, D]$
	3	P_d + $[B_r, B]$; $[B_r, D]$; $[D_r, B]$; $[D_r, D]$; $[A, B]$; $[A, D]$	$[B_r, D_r, A]$	$[B, D, P_d]$	P_d + $[B_r, D_r]$; $[B_r, A]$; $[A, D_r]$
$x_d^0 = \{B_r\}$	1	P_d	D_r	B; D	B; D
	2	$[B, P_d]$; $[D, P_d]$	$[D_r, A]$	$[D, P_d]$	$[D_r, P_d]$; $[A, P_d]$
	3	P_d + $[D_r, B]$; $[D_r, D]$; $[A, B]$; $[A, D]$	$[D_r, A]$ + B; P; D; P_d	P_d + $[D, B]$; $[A, D]$	P_d + $[D_r, B]$; $[D_r, D]$; $[A, B]$; $[A, D_r]$
$x_d^0 = \{B_r, P\}$	1	B; D; P_d	D_r	P_d	P_d
	2	$[B, P_d]$; $[D, P_d]$; $[B, D]$	$[D_r, A]$	$[D, P_d]$	$[B, P_d]$; $[D_r, P_d]$
	3	$[B, D, P_d]$	$[D_r, A]$ + B; D; P_d	P_d + $[D, B]$; $[D_r, B]$; $[D_r, D]$	P_d + $[D, B]$; $[D_r, B]$; $[D_r, D]$

Sequential Optimal

Scenario	k	PL2012	DC2003	$\epsilon = 0.1$	$\epsilon = 1$
$x_d^0 = \{\}$	-	$P_d \to B \to B_r$	$B_r \to D_r \to A$	$B \to D \to P_d$	$B \to B_r \to P_d$
$x_d^0 = \{B_r\}$	-	$P_d \to B \to D_r$	$D_r \to A \to B$	$B \to P_d \to D$	$B \to P_d \to D_r$
$x_d^0 = \{B_r, P\}$	-	$B \to D \to P_d$	$D_r \to A \to B$	$P_d \to - \to -$	$P_d \to B \to D_r$

Figure 9: Placement results on the sample model structure (Figure 1). Top: global optimal solutions. Bottom: sequential optimal solutions. Results from three solution schemes are compared under three scenarios of *a priori* data-availability condition: (1) $\boldsymbol{X_0^d} = \{0\}$; (2) $\boldsymbol{X_0^d} = \{B_r\}$; (3) $\boldsymbol{X_0^d} = \{B_r, P\}$. For our new objective function, results under two values of ϵ (0.1 and 1) are shown.

a new objective function for the optimization problem, constructed from variables' individual utilities conditioning on existing data availability, which could essentially be understood in the theoretical framework for the evaluation of side information (*Rinehart and Dahleh*, 2011). Under the novel objective function, analytical results for the optimal placement on binary and multi-ary trees are derived with graph-theoretical analysis; for a general tree structure with n nodes, an optimal placement algorithm is devised, with the time complexity upper bounded by $O(n\log_2(n))$. It is learned from the results that given a fixed budget on the number of sensors to be placed, binary trees represent the topology where sensors provide the greatest aggregated utility under the optimal placement, whereas the computational cost associated with this optimal placement configuration is also the largest on binary trees. Almost all other tree structures lead to a smaller average utility of sensors compared with binary structures, but also a less computational cost in realizing the optimal placement. Finally, for arbitrary model structures that are not necessarily trees, approximation solution schemes are pinned down in the study.

The three objective functions are tested on a sample model structure. Compared with the two translated solutions, our new solution highlights the conditional nature of the placement problem, with its output strategies believed to be more adaptive to specific problem settings. However, apart from its celebrated self-consistency in both analytical and numerical results, the computational disadvantage could be identified with our new approach. For an arbitrary model structure (not trees), since we search for the best sequence $\mathcal{S}$ from the permutation of an input set of nodes, an extra computational cost is incurred on top of the final combinatorial search. The permutation search space could grow exponentially with the size of the input set; for large-scale system models, the cost might be prohibitive and thus an additional approximation solution scheme is expected to be implemented. Intrinsically, this extra computational cost might be viewed as trading-off with the more adaptive nature of our new formulation.

Automation of data acquisition strategies, applicable to all sorts of system models in social, economic and organizational fields, is successfully realized in our study, which is the primary contribution of this work. Having three modes for the objective function, this scalable strategy toolkit provides system modelers with useful recommendations for the staging of data acquisition and model calibration procedures, which are often non-trivial parts of a modeling

$\boldsymbol{x_d^0} = \{B_r, P\}$

	B_r	B	$\boldsymbol{P}$	D_r	D	$\boldsymbol{P_d}$	A
$v(i\|\boldsymbol{d})$	0.08		**2.88**				
$v^{\dagger}(i\|\boldsymbol{d})$		0		0	0	**1.6**	0.08
v^{in}	0	2	**2**	0	2	**2**	0
v^{out}	1	1	**3**	1	1	0	1
v^{Katz}	0	4	4	0	4	**6**	0

Figure 10: Comparing variables' individual utilities $v(i|\boldsymbol{d})$ and $v^{\dagger}(i|\boldsymbol{d})$ with their graph centrality measures (in-degree centrality v^{in}, out-degree centrality v^{out}, Katz centrality v^{Katz}) under the scenario $\boldsymbol{X}_0^{\boldsymbol{d}} = \{B_r, P\}$. Graph centrality measures are not able to highlight existing and add-on sensor utilities at the same time.

practice. When domain heuristics are not available or misguiding, this toolkit may be particularly useful.

Upon the theoretical and applicational establishments of this study, it is expected that more contemplated objective functions for this problem would be proposed and studied in future work. For example, in the current work we do not consider the specific cost in the acquisition of datasets, which might indeed play a non-trivial role in real applications. Such a cost factor could be included in the formulation of sensor's individual utility $v(i|\boldsymbol{d})$. Meanwhile, although in the proposed formulation the varied quality of datasets in the real world has been successfully addressed, the current treatment might still be considered ad-hoc before their mathematical establishments are pinned down; moreover, formulations that better address the noise structure of real data as well as the sensitivity of model parameter estimation are to be invented and investigated, which might help bring the merits of the current algorithmic attempt to a more significant level. Generally speaking, the current study is primarily focusing on the mathematical structure of system models from a graph-theoretical perspective; in future studies, more practical concerns in social and economic modeling are expected to be addressed through the established solution framework.

Appendix

Proof of Theorem 2

The Sequential Placement procedure in Algorithm 1 consists of two steps. Locating a minimum value on a tree requires $O(\log(n))$ per call (*Cormen*, 2011). For a binary tree of length L and $n = 2^L - 1$ nodes, there are 2^{L-1} leaf nodes and thus at most $k = 2^{L-1}$ sensors. So the total complexity of the first step of all k calls is upper bounded by $O(n\log(n))$. Next we discuss the complexity of the second step, i.e., the update of vectors.

The number of nodes in the graph whose vectors need to be updated after the placement of a sensor i, is essentially the depth l of node i (minus 1), with $l = 1$ for the top node. Along the optimal sequence (e.g., Figure 5), the depth of the placed sensors is $1, 2, 3, ..., L, 2, 3, 4, ..., L, ...$, and therefore the overall complexity of the placement task on a binary tree is upper bounded by $kL = 2^{L-1}L = n\log_2(n)$. To finish the proof, we show that all other tree structures require less computation cost than binary trees. Conceptually, this is because that binary trees represent the optimal topology of trees in the current problem setting that best balance the depth and abundance of nodes.

A general tree structure could be seen as originated from a binary tree, after pruning and appending some nodes. If we are able to prove that any structural change on a binary tree will only reduce the complexity, the proof could be finished. First, for w-ary trees, through a similar computation, the complexity is given by $n\log_w(n) \leq n\log_2(n)$. Thus it can be concluded that *adding* branches on a binary tree while keeping the overall number of nodes at n, will always result in a less complexity.

Next, suppose one removes some branches from the binary tree. Keeping all n nodes in the tree, in order to try to maximally increase the computational cost, these pruned structures ought to be appended to a leaf node at the bottom level of the tree such that the largest gain in depth is obtained. Suppose one prunes a branch at depth l_{m_1} (Figure 11a). By appending the pruned tree T_{m_1} of length $L - l_{m_1}$ to a leaf node of the original tree, each of the $2^{L-l_{m_1}}$ nodes on this sub-tree would have a surplus at most $L - l_{m_1}$ in its depth. However, after one of the two branches has been pruned, the upper node at depth $l_{m_1} - 1$ will be cut out from the whole tree at the first step of the optimal sequence, since the top node of this sub-tree now has a greater utility than the topmost node of the background tree. Therefore, the $2^{L-l_{m_1}}$ nodes in the remaining branch T'_{m_1} (the brother branch of T_{m_1}) will each lose a depth $L - l_{m_1}$ since they are cut off from the main tree. Overall, the gains of depth will not exceed the loss of depth, and therefore this pruning-and-maximally-appending procedure will not increase the total computational cost.

To conclude that any pruning will never increase the overall depth, one needs to look at the case where two branches are pruned, and then generalizes to all cases. Suppose two branches are pruned on depth l_{m_1} and l_{m_2} (Figure 11b), and the pruned structures T_{m_1} and T_{m_2} are appended sequentially under one original leaf node (Figure 11c). The gains of depth are $L - l_{m_1}$ for $2^{L-l_{m_1}}$ nodes in T_{m_1}, and $(L - l_{m_1}) + (L - l_{m_2})$ for $2^{L-l_{m_2}}$ nodes in T_{m_2}. Similar to the case in the single-pruning scenario, two corresponding trees T'_{m_1} and T'_{m_2} are cut off, and the loss of depth are $(L - l_{m_1})$ for $2^{L-l_{m_1}}$ nodes, and $(L - l_{m_2})$ for $2^{L-l_{m_2}}$ nodes. However, one more tree is also cut off from the original tree, which is the brother branch T' of the minimum common tree T of the original T_{m_1} and T_{m_2}, since the leftover of T makes the upper node capping T and T' also a preferred sensor location than the topmost node of the background tree. The minimum common tree T of T_{m_1} and T_{m_2} has depth $m_c \leq min(l_{m_1}, l_{m_2})$. Therefore, cutting off T' will result in an additional loss of depths $(L - l_{m_c})$ for $2^{L-l_{m_c}}$ nodes. Overall, one could see that the loss of depths are no less than the gains in the double-pruning scenario, same as in single-pruning. Therefore, by induction, one concludes that for any pruning on binary trees, the computational cost will never increase, which finishes the proof. In all, for general tree structures, the complexity of determining the optimal sensor placement is upper bounded by $O(n\log_2(n))$, which is obtained on binary trees.

As mentioned, this algorithm has the similar idea to the greedy tree-breaking algorithm in *Braunstein et al.* (2016). In both algorithms one node is selected at each round (for removal in *Braunstein et al.* (2016) and for placement in our algorithm), after a series of message passing and update on the graph. In *Braunstein et al.* (2016), at each round the complexity is $O(\log(n)+T)$ where T is the diameter of the tree, and the total complexity is then $O(n(\log(n)+T))$, compared with $O(n\log(n))$ of our algorithm. This is due to the fact that in greedy tree-breaking, the underlying tree is undirected thus an extra depth of search (diameter) is needed to determine the root, whereas in our algorithm the tree is directed and the root is determined *a priori*. ■

References

Ades, A. E., Lu, G., & Claxton, K. (2004), Expected value of sample information calculations in medical decision modeling, *Medical decision making*, 24(2), 207-227.

Amendola, A., Giordano, F., Parrella, M. L., & Restaino, M. (2017), Variable selection in high?dimensional regression: a nonparametric procedure for business failure prediction, *Applied Stochastic Models in Business and Industry*, 33(4), 355-368.

Angulo, C., Rötter, R., Lock, R., Enders, A., Fronzek, S., & Ewert, F. (2013), Implication of crop model calibration strategies for assessing regional impacts of climate change in Europe, *Agricultural and Forest Meteorology*, 170, 32-46.

Arnold, J. G., Moriasi, D. N., Gassman, P. W., Abbaspour, K. C., White, M. J., Srinivasan, R., ... & Kannan, N. (2012), SWAT: Model use, calibration, and validation, *Transactions of the ASABE*, 55(4), 1491-1508.

Braunstein, A., DallAsta, L., Semerjian, G., and Zdeborov, L. (2016) Network dismantling, *Proceedings of the National Academy of Sciences*.

Cooke, R. (1991), Experts in uncertainty: opinion and subjective probability in science, *Oxford University Press*.

Cooke, R. M., & Goossens, L. H. (2004), Expert judgement elicitation for risk assessments of critical infrastructures, *Journal of risk research*, 7(6), 643-656.

Cormen, T. H. (2011), Algorithms unlocked, *MIT press*.

Daggupati, P., Pai, N., Ale, S., Douglas-Mankin, K. R., Zeckoski, R. W., Jeong, J., ... & Youssef, M. A. (2015), A recommended calibration and validation strategy for hydrologic and water quality models, *Transactions of the ASABE*, 58(6), 1705-1719.

Dakins, M. E., Toll, J. E., Small, M. J., & Brand, K. P. (1996), Risk-based environmental remediation: Bayesian Monte Carlo analysis and the expected value of sample information, *Risk Analysis*, 16(1), 67-79.

Dhillon, S. S., & Chakrabarty, K. (2003, March), Sensor placement for effective coverage and surveillance in distributed sensor networks, In *2003 IEEE Wireless Communications and Networking*, 2003. WCNC 2003. (Vol. 3, pp. 1609-1614). IEEE.

Festa, P., Pardalos, P. M., & Resende, M. G. (1999), Feedback set problems, In *Handbook of combinatorial optimization* (pp. 209-258), Springer, Boston, MA.

Grassini, P., van Bussel, L. G., Van Wart, J., Wolf, J., Claessens, L., Yang, H., ... & Cassman, K. G. (2015), How good is good enough? Data requirements for reliable crop yield simulations and yield-gap analysis, *Field Crops Research*, 177, 49-63.

Guyon, I., & Elisseeff, A. (2003), An introduction to variable and feature selection, *Journal of machine learning research*, 3(Mar), 1157-1182.

Hansen, S., Abrahamsen, P., Petersen, C. T., & Styczen, M. (2012), Daisy: Model use, calibration, and validation, *Transactions of the ASABE*, 55(4), 1317-1333.

He, D., Wang, E., Wang, J., & Robertson, M. J. (2017), Data requirement for effective calibration of process-based crop models, *Agricultural and forest meteorology*, 234, 136-148.

Kraan, B., & Bedford, T. (2005), Probabilistic inversion of expert judgments in the quantification of model uncertainty, *Management science*, 51(6), 995-1006.

Kahn, A. B. (1962), Topological sorting of large networks, *Communications of the ACM*, 5(11), 558-562.

Karp, R. M. (1972), Reducibility among combinatorial problems, In *Complexity of computer computations* (pp. 85-103), Springer, Boston, MA.

Kersebaum, K. C., Boote, K. J., Jorgenson, J. S., Nendel, C., Bindi, M., Frühauf, C., ... & Rötter, R. P. (2015), Analysis and classification of data sets for calibration and validation of agro-ecosystem models, *Environmental Modelling & Software*, 72, 402-417.

Khakbaz, B., Imam, B., Hsu, K., & Sorooshian, S. (2012), From lumped to distributed via semi-distributed: Calibration strategies for semi-distributed hydrologic models, *Journal of Hydrology*, 418, 61-77.

Knisel, W. G., & Douglas-Mankin, K. R. (2012), CREAMS/GLEAMS: Model use, calibration, and validation, *Transactions of the ASABE*, 55(4), 1291-1302.

Li, T., Rahmandad, H., & Sterman, J. (2020), Improving Parameter Estimation of Epidemic Models: Likelihood Functions and Kalman Filtering, *in preparation*.

Lin, C. T. . (1974), Structural controllability, *IEEE Transactions on Automatic Control*, 19(3), 201-208.

Lin, F. Y., & Chiu, P. L. (2005), A near-optimal sensor placement algorithm to achieve complete coverage-discrimination in sensor networks, *IEEE Communications Letters*, 9(1), 43-45.

Negm, L. M., Youssef, M. A., Skaggs, R. W., Chescheir, G. M., & Jones, J. (2014), DRAINMOD?DSSAT model for simulating hydrology, soil carbon and nitrogen dynamics, and crop growth for drained crop land, *Agricultural water management*, 137, 30-45.

Oliva, R. (2003), Model calibration as a testing strategy for system dynamics models, *European Journal of Operational Research*, 151(3), 552-568.

Oliva, R. (2004), Model structure analysis through graph theory: partition heuristics and feedback structure decomposition, *System Dynamics Review: The Journal of the System Dynamics Society*, 20(4), 313-336.

Papadimitriou, C. (2004), Optimal sensor placement methodology for parametric identification of structural systems, *Journal of sound and vibration*, 278(4-5), 923-947.

Papadimitriou, C., & Lombaert, G. (2012), The effect of prediction error correlation on optimal sensor placement in structural dynamics, *Mechanical Systems and Signal Processing*, 28, 105-127.

Rinehart, M. , & Dahleh, M. A. . (2010b), The Value of Sequential Information in Shortest Path Optimization, *The American Control Conference*, Baltimore, MD June, 30-July 2.

Rinehart, M. , & Dahleh, M. A. . (2011), The value of side information in shortest path optimization, *IEEE Transactions on automatic control*, 56(9).

Rinehart, M. , & Dahleh, M. A. . (2012), The value of side information in network flow optimization, *Systems & Control Letters*, 61(1), 79-85.

Saltelli, A., Ratto, M., Andres, T., Campolongo, F., Cariboni, J., Gatelli, D., ... & Tarantola, S. (2008), Global sensitivity analysis: the primer, *John Wiley & Sons*.

Santhi, C., Arnold, J. G., Williams, J. R., Dugas, W. A., Srinivasan, R., & Hauck, L. M. (2001), Validation of the swat model on a large river basin with point and nonpoint sources 1, *JAWRA Journal of the American Water Resources Association*, 37(5), 1169-1188.

Sarrazin, F., Pianosi, F., & Wagener, T. (2016), Global Sensitivity Analysis of environmental models: Convergence and validation, *Environmental Modelling & Software*, 79, 135-152.

Simmons, J. A., Splinter, K. D., Harley, M. D., & Turner, I. L. (2019), Calibration data requirements for modelling subaerial beach storm erosion, *Coastal Engineering*, 152, 103507.

Tian, S., Youssef, M. A., Skaggs, R. W., Amatya, D. M., & Chescheir, G. M. (2012), DRAINMOD-FOREST: Integrated modeling of hydrology, soil carbon and nitrogen dynamics, and plant growth for drained forests, *Journal of Environmental Quality*, 41(3), 764-782.

Tian, S., Yu, Y., & Guo, H. (2015), Variable selection and corporate bankruptcy forecasts, *Journal of Banking & Finance*, 52, 89-100.

van Bussel, L. G., Grassini, P., Van Wart, J., Wolf, J., Claessens, L., Yang, H., ... & van Ittersum, M. K. (2015), From field to atlas: Upscaling of location-specific yield gap estimates, *Field Crops Research*, 177, 98-108.

Yun, Y. H., Wang, W. T., Tan, M. L., Liang, Y. Z., Li, H. D., Cao, D. S., ... & Xu, Q. S. (2014), A strategy that iteratively retains informative variables for selecting optimal variable subset in multivariate calibration, *Analytica chimica acta*, 807, 36-43.

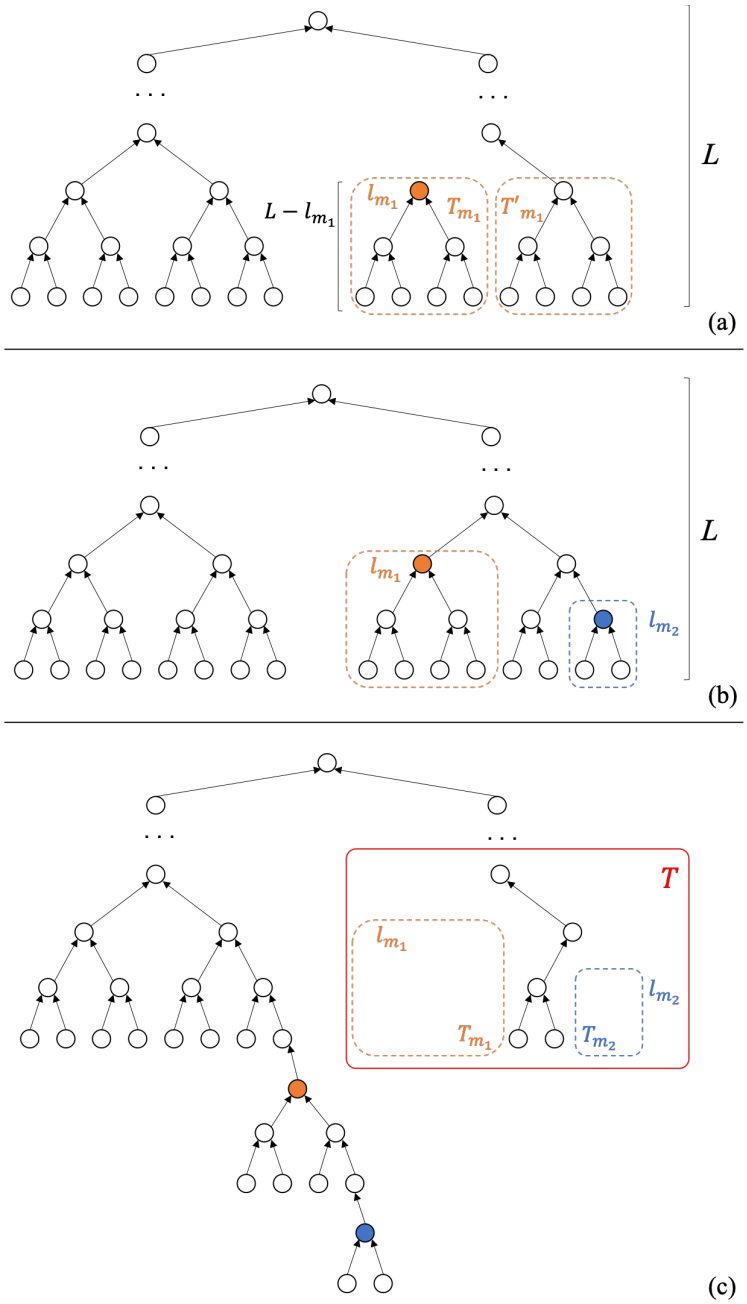


Figure 11: Illustrations for the Proof of Theorem 2. (a) Single-pruning. The pruned tree T_{m_1} and its brother branch T'_{m_1}. (b) Double-pruning. T_{m_1} and T_{m_2} are pruned from the original binary tree, at depth l_{m_1} and l_{m_2}, respectively. (c) The two pruned structures are appended sequentially to a leaf node of the background tree such that the largest gain in depth is obtained. Besides the brother trees of T_{m_1} and T_{m_2}, the brother tree T' of the minimum common tree T for T_{m_1} and T_{m_2}, is also cut off from the background tree.