Automated Decomposition of Model-based Learning Problems

Brian C. Williams and Bill Millar
Recom Technologies, Caelum Research
NASA Ames Research Center, MS 269-2
Moffett Field, CA 94305 USA
E-mail: williams, millar@ptolemy.arc.nasa.gov

Abstract
A new generation of sensor rich, massively distributed autonomous systems is being developed that has the potential for unprecedented performance, such as smart buildings, reconfigurable factories, adaptive traffic systems and remote earth ecosystem monitoring. To achieve high performance these massive systems will need to accurately model themselves and their environment from sensor information. Accomplishing this on a grand scale requires automating the art of large-scale modeling. This paper presents a formalization of decompositional, model-based learning (DML), a method developed by observing a modeler’s expertise at decomposing large scale model estimation tasks. The method exploits a striking analogy between learning and consistency-based diagnosis. Moriarty, an implementation of DML, has been applied to the mal modeling of a smart building, demonstrating a significant improvement in learning rate.

Introduction
Through artful application, adaptive methods, such as nonlinear regression and neural nets, have been demonstrated as powerful modeling and learning techniques, for a broad range of tasks including environmental modeling, diagnosis, control and vision. These technologies are crucial to tackling grand challenge problems, such as earth ecosystem modeling, which require an army of modeling experts. In addition, hardware advances in cheap sensing, actuation, computation and networking have enabled a new category of autonomous system that is sensor rich, massively distributed, and largely immobile. These “immobile robots” are rapidly being deployed in the form of networked building energy systems, chemical plant control networks, reconfigurable factories and earth observing satellite networks. To achieve high performance these massive systems will need to accurately model themselves and their environment from sensor information. However, the labor and skill involved makes these adaptive methods economically infeasible for most large scale modeling, learning and control problems. Our goal is to automate the expertise embodied by a skilled community of modelers at decomposing and coordinating large scale model estimation or learning tasks, and to develop these methods both in the context of data analysis and hybrid control problems. The approach we call decompositional, model-based learning (DML), and is embodied in a system called Moriarty. DML is a key element of a larger program to develop model-based autonomous systems (MBAs). MBAs achieve unprecedented performance through capabilities for self-modeling (e.g., DML) and self-configuration. A complement to Moriarty, called Livingstone, performs discrete modeling and self-configuration (Williams & Nayak 1996), and will fly a deep space probe in 1988.

Our work on DML was developed in the context of synthesizing an optimal heating and cooling control system for a smart, self-modeling building. To study this synthesis process we built a testbed for fine grained sensing and control of a building, called the responsive environment (et al. 1993b, 1993a), and used this testbed to study the manual art of our control engineers at decomposing the model estimation and optimal control components of the overall control problem (Zhang, Williams, & Elrod 1993).

A key insight offered by our study is that the process of decomposing a large model estimation problem is analogous to that used in model-based diagnosis to solve large scale multiple fault diagnosis problems. The decomposition of a diagnostic problem is based on the concept of a conflict – a minimal subset of a model (typically in propositional or first order logic) that is inconsistent with the set of observations (de Kleer & Williams 1987; Reiter 1987). Decompositional learning is based on the analogous concept of a dissent – a minimal subset of an algebraic model that is overdetermined given a set of sensed variables.

The model decomposition task begins with a system of equations, including a set of sensed variables, and a set of parameters to be estimated from sensor data. DML generates the dissents of the equations and uses these dissents to generate a set of estimators that together cover all parameters. It then coordinates the individual estimations, and combines the shared results. This paper focuses on the task of generating a
set of dissents and a corresponding estimator for each dissent. A number of strategies are possible for combining the generated estimators, and will be analyzed elsewhere.

The next two sections summarize model-estimation and the informal process of model decomposition. Section introduces the concept of a dissent used to decompose a model into simple fragments. Section develops a local propagation algorithm used to generate a restricted form of dissent. Section describes an algorithm for turning the set of dissents into a sequence of estimators that cover the model parameters. Section presents experimental results. The paper closes with a discussion of related work and future directions.

**Model Estimation**

Statistical modeling involves estimating the parameters of a system from sensor data; more precisely:

**Definition 1** A system is a pair \( (e(c;v),s) \), where \( e(c;v) \) is a vector of rational expressions over the vector of variables \( v \) and constants \( c \), \( e(c;v) = 0 \) is a vector of independent equations, and the sensed variables \( s \subset v \) are exogenous.\(^1\)\(^2\) An estimation problem is a pair \( (M,p) \) where model \( M \) is a system \( (e(c;v),s) \), and the unknown parameters \( p \), is a vector such that \( p \subset c \).

For example, an office's energy and mass flow (heat, air and water) is modeled by a vector \( e(c;v) \) of 14 equations involving seventeen state variables \( v \):\(^3\)

\[
\begin{align*}
F_{ext} &= F_{sply} & (1) \\
F_{sply} &= F_{ttrn} & (2) \\
Q_{ext} &= C_q F_{ext} T_{ext} & (3) \\
Q_{sply} &= C_q F_{sply} T_{sply} & (4) \\
Q_{ttrn} &= C_q F_{ttrn} T_{ttrn} & (5) \\
Q_{rhtcap} &= Q_{ext} - Q_{sply} + Q_{rhtt} & (6) \\
Q_{rhtcap} &= C_{rht} dT_{sply} / dt & (7) \\
Q_{rht} &= \left( \frac{Q_{rhtmax}}{X_{rhtmax}} \right) X_{rht} & (8) \\
F_{ext} &= F_{tkg} + F_{dmpr} & (9) \\
F_{tkg} &= \left( \frac{\mu_{tkg}}{R_{dk}} \right) \sqrt{P_{dk}} & (10) \\
F_{dmpr} &= \left( \frac{\mu_{dmpr} (X_{dmpr})}{R_{dk}} \right) \sqrt{P_{dk}} & (11) \\
Q_{rht} &= Q_{sply} + Q_{exp} + Q_{str(t)} & (12)
\end{align*}
\]

\(^1\) Variables in bold, such as \( e \) denote vectors. \( v^T \) transposes a row vector to a column vector. We apply set relations and operations to vectors with the obvious interpretation of vectors as sets.

\(^2\) The exogenous variables are those whose values are determined independent of the equations.

\(^3\) \( X, F, T, q \) and \( P \) denote position, air flow, temperature, heat flow and pressure, respectively.

\[
\begin{align*}
Q_{rht} &= C_{rht} dT_{rht} / dt & (13) \\
Q_{wall} &= \sigma_{wall} (T_{rht} - T_{ext}) & (14)
\end{align*}
\]

Nine of the state variables are sensed \( s \), and the constants \( c \) consist of seven unknown parameters \( p \), and four known constants \( c' \):

\[
s = \langle T_{ext}, T_{sply}, T_{rht}, dT_{sply}/dt, dT_{rht}/dt, X_{rht}, X_{dmpr}, F_{sply}, P_{dk} \rangle^T
\]

\[
p = \langle \mu_{tkg}, \mu_{dmpr} (X_{dmpr}), C_q, X_{rhtmax} \rangle^T
\]

Estimation involves adjusting the set of model parameters to maximize the agreement between a specified model and the sensor data using, for example, a Bayesian or a least-squares criteria with a Gaussian noise model. Using least-squares would involve selecting one of the sensed variables \( y \) from \( s \), and manipulating equations \( e(c;v) \) to construct an estimator \( y = f(x;p';c') \) that predicts \( y \) from parameters \( p' \subset p \), other sensed variables \( x \subset s \) and constants \( c' \subset c \). The optimal estimate is then the vector, \( p^* \), of parameter values that minimizes the mean-square error between the measured and predicted \( y \):

\[
p^* = \arg \min_{p} \sum_{(y_i,x_i) \in D} (y_i - f(x_i;p';c'))^2
\]

where \( y_i \) and the \( x_i \) are in the \( i \)th sampling of sensor values \( D \) for \( s \).

The modelers first attempted to estimate all parameters of the thermal problem at once, which required solving a 7-dimensional, nonlinear optimization problem involving a multi-modal objective space. Using arbitrary initial values, a Levenberg-Marquardt algorithm was applied repeatedly to this problem, but consistently became lost in local minima and did not converge after several hours.

**The Art of Model Decomposition**

It is typically infeasible to estimate the parameters of a large model using a single estimator that covers all parameters. However, there is often a large set of possible estimators to choose from, and the number of parameters contained in each estimator varies widely. The art of modeling for data analysis (and DML) involves decomposing a task into a set of "simplest" estimators that minimize the dimensionality of

\(^4\) The least-squares estimate is effective and pervasive in practice. It is the Maximum Likelihood Estimate under appropriate assumptions, but, in general, it is not probabilistically justifiable. Our method is independent of the optimality criteria, but we illustrate using least-squares here for clarity.

\(^5\) We illustrate here for a single response variable \( y \). A vector of response variables \( y \) and estimators \( f \) pertains to the coordination of generated estimators, to be developed elsewhere.
the search space and the number of local minima, hence improving learning rate and accuracy. Each estimator together with the appropriate subset of sensor data forms an estimation subproblem that can be solved separately, either sequentially or in parallel.

For example, our modelers estimated the seven parameters of the thermal problem far more simply by manually decomposing the model into three small subsets, used to generate three estimators. The first estimator, \( f_1 \), is:

\[
F_{\text{ext}} = (p_{ikg} + \rho_{dmp}_{r}(X_{dmp})) \sqrt{P_{\text{det}}/R_{\text{det}}}
\]

where \( y_1 = F_{\text{ext}}, \ x_1 = (P_{\text{det}},X_{dmp})^T, c'_1 = (p_{ikg}, \rho_{dmp}_{r}(X_{dmp}))^T \) and \( p_1 = (R_{\text{det}})^T \). Estimating parameter \( R_{\text{det}} \) using \( f_1 \) and sensor data involves just searching along one dimension. The second estimator, \( f_2 \), is:

\[
dT_{\text{sply}}/dt = C_0 F_{\text{sply}}(T_{\text{ext}} - T_{\text{sply}}) + (X_{\text{rhtmax}}X_{\text{rht}})
\]

where \( y_2 = dT_{\text{sply}}/dt, \ x_2 = (T_{\text{sply}},F_{\text{sply}},T_{\text{ext}}X_{\text{rht}})^T, c'_2 = (X_{\text{rhtmax}},C_0^T)^T \) and \( p_2 = (C_{\text{rht}},Q_{\text{rhtmax}})^T \). This results in a 2-dimensional search, again a simple space to explore. Finally, \( f_3 \) is:

\[
dT_{\text{rht}}/dt = [C_0 F_{\text{sply}}(T_{\text{sply}} - T_{\text{rht}}) + Q_{\text{sply}} + Q_{\text{str}}(t) + \sigma_{\text{walt}}(T_{\text{ext}} - T_{\text{rht}})]C_{\text{rht}}^{-1}
\]

where \( y_3 = dT_{\text{rht}}/dt, \ x_3 = (T_{\text{rht}},F_{\text{sply}},T_{\text{sply}},T_{\text{ext}})^T, c'_3 = (C_0)^T \) and \( p_3 = (C_{\text{rht}},Q_{\text{sply}},Q_{\text{str}}(t))C_{\text{rht}}^{-1} \). This involves exploring a 4-dimensional space, a task that is not always trivial, but is far simpler than the original 7D problem. Using the derived estimators, the estimation of the seven parameters converged within a couple of minutes, in sharp contrast to the estimation of all parameters using a single estimator.

The remainder of this paper concentrates on automatically decomposing a model into a set of these estimators. The equally important problem of combining estimators with shared parameters is touched on only briefly.

**Decomposition Using Dissents**

To automate the process of constructing a decomposition we note that the central idea behind estimation is to select those parameters \( p \) that minimize the error between a model \( e'(p;v) = 0 \) and a set of data points \( D \equiv (s') \) for sensed variables \( s' \). What is important is that the existence of this error results from the model being over-determined by the sensed variables.

\[ e' \text{ and } s' \text{ need not be the complete system of equations and sensed variables } (e,s). \]

Any subsystem \( (e',s') \) that is over-determined may be used to perform an estimation. Of course, not all subsystems are equal. Roughly speaking, the coverage rate is best reduced by minimizing the number of parameters mentioned in \( e' \), and the accuracy of the estimation is improved by minimizing the number of sensed variables \( s' \) per estimator. The key consequence is that the most useful over-determined subsystems are those that are minimal (i.e., it has no proper subsystem that is over-determined). At the core of DML is the generation of minimally over-determined subsystems, called *dissents*.

**Definition 2** The dissent of a system \((e,s)\) is a subsystem \((e_d,s_d)\) of \((e,s)\), such that \((e_d,c;v) = 0\) is over-determined given \( s_d \). \((e_d,s_d)\) is a minimal dissent if no proper subsystem \((e',s')\) of \((e_d,s_d)\) exists such that \((e',s')\) is a dissent of \((e,s)\).

For example, the thermal estimation problem has eight minimal dissents, a small fraction of the complete set of over-determined subsystems, which is on the order of tens of thousands. The dissent with the fewest equations and sensed variables is:

\[
((E_9 - 11)^T(F_{\text{ext}},P_{\text{det}},X_{dmp})^T)
\]

This dissent involves only one unknown parameter, \( R_{\text{det}} \), hence a one dimensional search space. The error in the parameter estimation is influenced by noise in only three sensors. Solving for \( F_{\text{ext}} \) results in the first estimator of the preceding section. In contrast, the largest dissent is:

\[
((E_2 - 14)^T,(dT_{\text{rht}}/dt,dT_{\text{sply}}/dt,F_{\text{sply}},P_{\text{det}},T_{\text{rht}},T_{\text{sply}},X_{\text{rht}},X_{dmp})^T)
\]

This dissent contains all 7 parameters and hence involves a 7-dimensional search. The accuracy of the estimation is influenced by noise from eight sensors. Solving for \( dT_{\text{rht}}/dt \) results in the estimator that was derived manually by our modelers to estimate all 7 parameters at once.

Note that there is a close analogy between the algebraic concept of dissent and the logical concept of a *conflict* used in model-based diagnosis (de Kleer & Williams 1987; Hamscher, Console, & de Kleer 1992). A conflict summarizes a logical inconsistency between a model and a set of observations, while a dissent identifies a potential for error between a model and a set of sensor data. Both are a measure of disagreement between a model and observations. For conflict-based diagnosis this is a logical disagreement; a conflict is a minimal set of *mode literals*, denoting component modes (e.g., \{ok(driver1),open(valve1)\}), whose conjunction is inconsistent with a theory. For DML the disagreement is a continuous error (on a euclidean metric), and a dissent is a minimally over-determined subsystem. There is an important distinction between the

---

*In an additional stage, not presented here (see (Zhang, Williams, & Elrod 1993)), these three estimators were in turn simplified using dominance arguments (in the spirit of (Williams & Raiman 1994)) to a set of six estimators, one requiring two unknown parameters to be estimated, and the remaining involving only one unknown.*

*A (proper) subsystem of \((e,s)\) is a pair \((e',s')\) such that \( e' \cup s' \) is a (proper) subset of \( e \cup s \).*
two concepts. The inconsistency indicated by a conflict is unequivocal, while a dissent merely indicates the potential for error, hence, our use of a more mild term—"dissent"—in naming this form of disagreement.

We exploit this analogy to develop our dissent generation algorithm. It also suggests a much more rich space of connections between model-based learning and model-based diagnosis. In essence conflict-based diagnosis is a discrete, somewhat degenerate, form of learning involving the identification of the discrete modes of a system from data (de Kleer & Williams 1989; Williams & Nayak 1996).

Dissent Generation Algorithm DG1

To generate the set of dissents we identify subsystems, called support, which uniquely determine the values of particular variables:

Definition 3 Given system $S = (e(c; v), s)$, a support of variable $v_i \in v$ is a subsystem $(e', s')$ of $S$, such that $(e', s')$ determines $v_i$, and no proper subsystem of $(e', s')$ determines $v_i$.

A pair of support for variable $v_i$ provide two means of determining $v_i$. Hence the union of the pair over determine $v_i$, and if minimal constitutes a dissent $(e_1 \cup e_2, s_1 \cup s_2)$.

The concept of an environment in conflict-based diagnosis (or more generally a prime implicat (de Kleer, Mackworth, & Reiter 1992)) parallels that of support. An environment is a minimal set of mode literals (e.g., stick-off (valve1)) that entail a value for some variable (e.g., $v = 6$), given a propositional model. If two predictions are inconsistent (e.g., $v = 6$ and $v = 5$), then the union of their two environments form a conflict. Thus while an environment entails a prediction for $x$, a support determines the value of $x$, given sensor data.

For dissents, by further presuming that the equations of the system are invertible, it follows trivially that all dissents can be generated just from the support of the sensed variables $s$.

Proposition 1 $S$ is the complete set of dissents for system $(e, s)$, where:

$$S = \{(e', s' \cup \{s_i\}) | s_i \in s & (e', s') \text{ supports } s_i\}.$$ 

Note, however, that the analogue does not hold for propositional systems, and hence not for conflicts.

To generate supporters and dissents we need a condition for identifying when a subsystem is uniquely determined or minimally over determined, respectively.

A standard presumption, made by causal ordering research (see (Nayak 1992; Iwasaki & Simon 1986)), and frequently for analyzing models of nonlinear physical systems, is that $n$ independent model equations and exogenous variables uniquely determine $n$ unknowns.

Assumption 1 Given estimation problem $(l(e(c; v), s), p)$, let $(e'(c'; v'), s')$ be any subsystem of $(e(c; v), s)$, let $n = |e'(c'; v')|$, $m = |s'|$ and $l = |v'|$.

We assume that $(e'(c'; v'), s')$ is (a) overdetermined if $n + m > l$, (b) dissenting if $n + m = l + 1$, (c) uniquely determined if $n + m = l$, and (d) underdetermined if $n + m < l$.

Note that this condition holds universally for linear systems. It is typically true of physical systems that are nonlinear, although not universally true. For example, $x^2 = 4$ is underdetermined, while $x^2 = -4$ is overdetermined. It is not true, for example, for systems of equations in boolean algebra.

The power of this condition is that it doesn't require knowledge of the form of an equation, just the variables each equation interacts with. Hence using this condition DML is able to decompose a graph of interactions (Williams 1990) into estimation subproblems, without restriction on, or further knowledge about, the form of the underlying equations. This is in the spirit of graphical learning methods, such as (Buntine 1994; Shachter, Anderson, & Poh 1990). An additional feature of the condition is that it is trivial to evaluate, much easier, for example, than determining whether or not an inconsistent propositional clause is minimal during conflict-based diagnosis.

DML generates candidate support by exploiting the analogy to environments, and by using the above condition to quickly test the candidate. Environments can be generated by propagating them locally through a network of clauses, starting at the mode literals, combining and then propagating a new set of environments after each traversal of a clause (de Kleer 1986; de Kleer & Williams 1987). Analogously, a support can be generated by propagating support locally through the network of equations, starting at the sensed variables, combining and then propagating a new set of support after each traversal of an equation. This algorithm is given in Figure 1.

The function CreateDecomposition kicks off the propagation, while AddSupport recursively propagates through successive local equations, and turns the support of sensed variables $s$ into dissents. The core of the algorithm is the function Propagate, which passes a new support through an equation $e$. It uses the function WeaveSupport to combine a newly added support of a variable with a support for each of the other variables of $e$, save one ($v$), producing a composite subsystem $c$ (we call $v$ an effect and $x - \{v\}$ the causes). It then adds $e$ to $c$ to produce a candidate support $s$ for $v$. The function CausalOrientations is used by Propagate to select all possible pairs of causes and effect for each local equation.

With respect to soundness, note that a candidate $s$ does not necessarily constitute a support. First, if any of the support $s_i$ contains $e$, then $s$ will not be a minimal subsystem determining $v$. Otherwise, if the equations in $c$ include $v$, then the addition of $e$ will over determine $v$, and hence $s$. Finally, if two support selected for causes share a variable, and the variable is determined by different sets of equations in the two support,
function CreateDecomposition(sT)
  /*system sT */
  Initialize dissent of sT to empty
  for s_i ∈ s do
    AddSupport(s_i,{},s_i),sT)
  endfor
  return dissent of sT
end CreateDecomposition

function Propagate(v,(e,s),(y,e,x), sT)
  /*equation e, y its effect, x its causes,
  v ∈ x, w support (e,s) & system sT */
  if e \(\not\in\) e then
    S_w = WeaveSupport(v,{(e,s)},e,x)
    for (e,,s,)
     if y \not\in variables of e_w then
        AddSupport(y,(e w U {e},s_w),sT)
      end Propagate
    end
  return
end Propagate

function WeaveSupport(v,S,e,x)
  /*equation e, its causes x, v ∈ x,
  & its supporters S */
  if x is empty, then
    return S
  else
    h = a variable in x
    R = x - {h}
    if h = v, then
      return WeaveSupport(φ, S, e, R)
    else
      S = Support(h), e \not\in e}
      S_e = {(e U e_2, s U s_2)}
      (e,s) ∈ S, (e_2,s_2) ∈ S_2} S_e = {s| s ∈ S_e, ~Overdetermined?(s)}
      S_r = {s| s ∈ S_r, ~Overdetermined?(s)}
      Return WeaveSupport(v,S_e,e,R)
    end WeaveSupport
  end

function AddSupport(v,(e,s),sT)
  /*variable v, support (e,s) & system sT */
  if v \not\in s of sT then
    Add (e,s U {v}) to dissent of sT
    Add v to dissenting vars. of (e,s U {v})
  endif
  Add (e,s) to the support of v
  for e ∈ equations of v do
    if e \not\in e then
      C_v = CausalOrientations(e,v)
      for c ∈ C_v
        do Propagate(v,(e,s),c,sT)
      end
    AddSupport
  end

function CausalOrientations(e,v)
  /*equation e, with v selected as a cause */
  V = variables of e
  return \{(y,e,X)|y ∈ V, X = V - \{v\}, v ∈ X\}
end CausalOrientations

function ConstructEstimator(y, (e,s) T)
  /*dissenting variable y & its dissent (e,s) */
  x = s - \{y\}
  d = variables of e not in x
  estimator f = 'y'
  repeat until all d are eliminated from f do
    find d in d and e ∈ e such that
    d occurs in f & e and
    d doesn’t occur in e - \{e\}
    endif
    e = e - \{e\}
    Solve for d in e, producing 'd = g'
    Substitute g for all occurrences of d in f
    Simplify f
endrepeat
  return (y,f,x,parameters of f)
end ConstructEstimator

Figure 1: Decomposition generation algorithm DG1 for DML
then that variable will be overdetermined, hence c and s will be overdetermined. Otherwise s constitutes a support of v. Each of these cases is detected and eliminated by testing for the membership of e withinc or overdeterminacy as c and s are constructed. These tests ensure the soundness of the algorithm.

Next consider completeness. For model-based diagnosis, the set of environments generated by local propagation is incomplete for general causal theories. However, it is complete for horn clause theories. Analogously, if a support is a system of simultaneous equations, then it will not be identified through the above local propagation algorithm, since the simultaneity represents a codependence between variables. The algorithm does, however, generate all support that are simultaneity-free.

More precisely, we build the concept of simultaneity-free inductively from the concept of a causally-oriented equation.

Definition 4 A causally-oriented equation of system \( S = (e(c; v), s) \) is a pair \( (e, (c; x_c), v_e) \), where \( e(c; x_c) \in e(c; v) \) is the equation, \( v_e \in x_c \) is the effect, and \( x_c = x_c - \{v_e\} \) are the causes. Then for each \( v_i \in x_c \) there exists a support \( (e', s') \) of \( v_i \) such that \( (e', s') \) does not determine \( v_e \).

Definition 5 A support \( (e, s) \) of variable \( v_i \) is simultaneity free, if \( v_i \in s \), or there exists a causal equation \( (e_i, v_i)^T \) of \( (e, s) \) such that each variable in \( (e_i, v_i)^T \)'s causes has a support in \( (e, s) \) that does not contain \( e_i \) and is simultaneity free. Otherwise \( (e, s) \) is simultaneous. A dissent \( (e, s) \) is simultaneity free if there exists a \( v_i \in s \) such that \( (e(c; v), s - \{v_i\})^T \) is a simultaneity free support.

Using these definitions, soundness and completeness is summarized as:

**Proposition 2** Given system \( (e(c; v), s) \), \( \text{CreateDecomposition}((e(c; v), s)) \) identifies all and only those subsystems \( (e'(c'; v'), s') \) of \( (e(c; v), s) \) that are simultaneity-free support for some variable \( v_i \in v \).

**Proof:** This proposition follows as a straightforward inductive proof on the size of the support. We use the earlier case analysis for function \( \text{propagate} \) to prove soundness. In addition it is straightforward to show that, every simultaneity-free support is a candidate \( s \) for \( v_i \) generated by \( \text{Propagate} \).

Returning to the thermal problem, the support generated by DG1 for the sensed variables \( s \) result in eight dissents, listed in Figure 2 in the form \( \text{dissent} \Rightarrow \text{parameters} \).

**Generating Estimators from Dissents**

A dissent, \( D_i \), is converted into an estimator \( y_i = f_i(x_i; p; c_i) \) by selecting one of the sensed variables as \( y_i \), and solving for \( y_i \) in terms of the other sensed variables. A dissent denotes a system of nonlinear equations, which can be extremely difficult to solve in general; however, the fact that the system is simultaneity-free makes this process straightforward. This is implemented by function \( \text{ConstructEstimator} \) which takes as input a dissent and a dissenting variable, a sensed variable \( s_i \in s \), where the dissent was identified by \( \text{Propagate} \). Using \( F_{ext}, \frac{dT_{pix}}{dt} \) and \( \frac{dT_{rm}}{dt} \) as dissenting variables for dissents \( D_2, D_1, \) and \( D_6 \), respectively results in three estimators, given in Section.

A variety of strategies are possible for selecting and coordinating the set of estimators to be used for data analysis. The appropriate strategy depends on the use of the estimators. For example, either the strategy...
must insure sufficient accuracy, given that it may only be using a subset of the sensed variables and hence sensed data, or it can be used to generate a good approximation of parameter values, which are used as initial values for a second, more accurate estimation stage on the complete data set. The appropriate strategy requires careful analysis, and as mentioned earlier is beyond the scope of this paper. However, we touch on a simple approach in the next section, for completeness.

**Implementation and Experiments**

We have developed a system, called *Moriarty*, that implements decompositional, model-based learning. Given an estimation problem \( \langle e(c;v), s, p \rangle \), *Moriarty* automatically generates code to perform the required parameter estimation. A symbolic algebra package, Mathematica, is currently used as the source language and also handles much of the algebraic manipulation required to generate estimators from dissents. *Moriarty*’s target language is presently S-PLUS, a data analysis package. The DG1 algorithm is implemented in Allegro Common Lisp.

*Moriarty* generates a “simplest” sequence of estimators using a greedy algorithm. This algorithm selects an estimator at each step in the sequence so that at least one new parameter is being estimated that wasn’t estimated at an earlier step in the sequence, yet minimizes the number of new parameters being estimated at each step. If two estimators have equal numbers of new parameters, then the estimator is selected that contains the fewest number of sensed variables. The first condition drives the convergence time downwards by reducing the dimensionality of the search space, while the second condition improves accuracy, by reducing the combined sensor error introduced. Applied to the example this algorithm selects dissents D2, D1 and D6, in that order, which contain one, two and four parameters, respectively. In this sequence no parameters are shared. This sequence corresponds to the three estimators of Section 3.

We now quantify the benefits of automated decomposition in the context of the thermal example. First, note that *Moriarty* generates the parameter estimation code required to solve the thermal problem in about 10 seconds on a Sparc 2. Now refer to Figure 3. To obtain a feel for the improvement in learning rate that decomposition provides in this example, the estimators corresponding to the 8 dissents returned by DG1 were run against data sets ranging in size from 10 to 200. The plot labeled F7 is for the original 7-dimensional estimator, while plots F2, F1, and F6 are for the simplest estimators that *Moriarty* found to cover all parameters. The estimators were provided with ball-park initial parameter estimates to allow convergence in the higher dimensional cases, and decomposition lead to significant speed-up even when good initial estimates were available. Note the improvement in learning rate provided by the decomposition over all trial sizes: focusing on trial size 200, for instance, the original estimator requires 166 seconds, while the total time to estimate all parameters using F2, F1, and F6 is under 9 seconds. The estimations produced by the decomposition also demonstrated an improvement in estimation accuracy. For example, the 95F2, F1 and F6 were 75 related.

**Related Work and Discussion**

In addition to model-based diagnosis, there is a variety of related work in the areas of machine learning, qualitative reasoning, and problem decomposition. Of particular note in the area of model-based learning is recent work on Bayesian learning which uses a graphical model to strongly bias the estimation process, substantially improving convergence (see, for example, (Buntine 1994; Heckerman 1995; Russell et al. 1995; Spiegelhalter & Lauritzen 1990; Shachter, Anderson, & Poh 1990; Shachter, Anderson, & Szolovits 1994)). Work on inferring independence of decision variables provides a means of decomposing the Bayesian learning task. The focus there is on graphical models representing probabilistic influence, where the graphical information implicitly used by DML represents deterministic influences. More recent work on Bayesian learning incorporates connectives representing deterministic relations. Recent work in the qualitative reasoning community has also focused on automated modeling for estimation. (Kay & Ungar 1993) uses a deterministic monotone influence diagram to bias estimation, (Bradley & Stolle 1997).
presents a series of symbolic methods for generating model-based estimation codes. Finally, a variety of methods are being developed for decomposing control problems, including work on decomposing phase space (Zhao 1994; Yip & Zhao 1995), and on decomposing optimal Markov decision problems (Dean & Lin 1995; Boutilier; Dean, & Hanks 1995).

To summarize, formalizing the art of large-scale modeling will be crucial to fully exploiting a new generation of sensor rich, massively distributed autonomous systems, and to solving many grand challenges, such as earth ecosystem monitoring. With a focus on modeling deterministic physical systems described by nonlinear equations, this paper developed the decomposition phase of our decompositional, model-based learning approach. The insights underlying the decomposition process were gained by observing the manual process of modelers on a thermal modeling task of a smart building. This suggests a striking analogy between a model decomposition, which we name a dissent, and a minimal conflict, used extensively in the model-based diagnosis community to tackle diagnosis problems on the orders of thousands of components. A dissent is a minimal subsystem of equations that is overdetermined. Identifying the minimal subsystem improves convergence time by minimizing the number of parameters involved, and improves accuracy by minimizing the number of sensed variables, hence reducing error. Following this analogy we developed a dissent generation algorithm DG1 that parallels the conflict recognition phase of model-based diagnosis. This algorithm is sound and complete with respect to generating a restricted form of dissent, which is simultaneity free. Preliminary experimental results on the smart building thermal model demonstrates an order of magnitude improvement in convergence rate achieved by using the smallest dissent to cover the parameter space. DML is currently being considered in a variety of exciting applications, including NASA’s next generation of fully autonomous space probes, and a “biosphere-like” habitat, called a closed loop ecological life support system. Finally the strong analogy here between model-based diagnosis and learning highlights an opportunity for a rich interaction between the subdisciplines and the potential for a more unified theory.

Acknowledgements The modeling team of the Xerox Responsive Environment project - Zhang Ying, Joseph O’ Sullivan and Scott Elrod - provided invaluable insights into the process and payoffs of applying nonlinear estimation and control methods. The responsive environment software/hardware system was developed by Rick Costanza, Scott Clearwater, Jim des Rivieres, Mike Dixon and Scott Elrod. Pandu Nayak provided invaluable feedback and support.

References


Zhang, Y.; Williams, B. C.; and Elrod, S. 1993. Model estimation and energy-efficient control for building management systems. Technical report, Xerox PARC.