Ecological Model Repositories Revisited: Casting Ecological Model Composition Problems as Dynamic Constraint Satisfaction Problems

Jeroen Keppens and Qiang Shen
School of Informatics, The University of Edinburgh
Email: {jeroen,qiangs}@inf.ed.ac.uk

Abstract

Compositional modelling, the predominant knowledge-based approach to automated model construction, takes a scenario describing the constituent interacting components of a domain system and translates it into a useful mathematical model. This paper extends this approach to enable its application to systems that can not be easily described in terms of interacting functional components. In particular, ecological systems are employed to illustrate the ideas. The previously presented informal approach, which casts the compositional modelling problem as a dynamic constraint satisfaction problem, is herein formalised by a set of algorithms. The major purpose of this work is to allow the recent advances in constraints research to benefit automated model construction and hence model-based reasoning at large.

Introduction

The significance of compositional modelling in model-based reasoning in general (Falkenhainer, B. & Forbus, K. 1991; Keppens, J. & Shen, Q. 2001) and its role in ecological problem solving in particular (Heller, U. & Struss, P. 1998; 2001; Salles, P. & Bredeweg, B 2002) is well understood. This work concerns the task of building model repositories of ecological systems, which poses two important new challenges to compositional modelling.

Firstly, the processes and components of an ecological system are dependent on one another and on the ways they are described when modelled. In population dynamics for example, models describing the predation or competition phenomena between two populations rely on the existence of a population growth model for each of the populations involved in the phenomenon. This inhibits the conventional approach of searching for a consistent and adequate combination of partial models, one for each component in the scenario. This approach provides an adequate solution for physical systems because these are comprised of components implementing a particular functionality that can be described by one or multiple partial models. Although the seminal work on compositional modelling (Falkenhainer, B. & Forbus, K. 1991) recognised the existence of complex interdependencies in model construction, it only provided a partial solution for it: all the conditions under which certain modelling choices could be relevant had to be specified manually in the knowledge base.

Secondly, the domain of ecology lacks a complete theory of what constitutes an adequate model. Most existing compositional modellers are based on a predefined concept of model adequacy and they employ inference mechanisms that are guaranteed to find a model that meets this criterion of adequacy. The criteria for adequacy of ecological models varies between ecological domains and even between the ecologists that require the model in a common area. Therefore, compositional ecological modelling requires a flexible facility to define the properties that the generated ecological models should satisfy.

This paper tackles these issues by providing a formal method to translate a compositional ecological modelling problem into an activity-based dynamic constraint satisfaction problem (aDCSP) (Miguel, I. & Shen, Q. 1999; 2000), based on an earlier approach as presented in (Keppens, J. & Shen, Q. 2000). The work is demonstrated with a more detailed discussion of a realistic example. The primary advantage of this work is that it enables compositional modelling problems to be solved by means of efficient aDCSP techniques. As such, compositional modellers can benefit from recent, and future, advances in dynamic constraint satisfaction.

Compositional Ecological Modelling

This section describes the compositional ecological modelling problem (CEMP), introduces the underlying concepts and illustrates them by means of a population dynamics application.

Preliminary concepts

Participants refer to the objects of interest, which are involved in the scenario or its model. These participants may be real-world objects or conceptual objects, such as variables that express features of real-world objects in a mathematical model. For instance, a population of a species is a typical example of a real-world object, and a variable that expresses the number of individuals of this species forms an example of a conceptual object.

Relations describe how the participants are related to one another. As with participants, some relations represent a real-world relationship, such as: predation(frog, insect). Other relations may be conceptual in nature, such as equation (1), which describes the important textbook model of logistic population growth:

$$\text{change} = \text{parameter} \times \text{size} \times (1 - \frac{\text{size}}{\text{capacity}}) \quad (1)$$

To be consistent with other compositional modelling approaches, this paper employs a LISP-style notation for relations. As such, the above two sample relations become:

```
(predation frog insect)
(== change (d/dt (* change-rate size
                    (- 1 (/ size capacity)))))
```

Assumptions form a special type of relation. They are hypotheses or presumptions that can be made in the construction of a scenario model. As a scenario to be modelled does not provide a consistent and appropriate set of assumptions upon which to base the resulting scenario model, it is up to the compositional modeller to find such an assumption set.

The version of the implemented modeller discussed herein employs two types of assumption: relevance and model assumptions (Keppens, J. & Shen, Q. 2000). *Relevance assumptions* state what phenomena are to be included in or excluded from the scenario model. The general format of a relevance assumption is shown in (2). The phenomenon that is incorporated in the scenario model when describing a relevance assumption is identified by ⟨*name*⟩ and is specific to the subsequent participants or relations. For example, relevance assumption (3) states that the growth of participant `?population` is included in the model.

$$\texttt{(relevant} \quad \langle name \rangle \quad [\{\langle participant \rangle\} \mid \langle relation \rangle]\texttt{)} \quad (2)$$

$$\texttt{(relevant growth ?population)} \quad (3)$$

Model assumptions specify which type of model is utilised to describe the behaviour of a certain participant or relation. The formal specification of a model assumption is given in (4). Often the ⟨*name*⟩ in (4) corresponds to the name of a known (partial) model of the phenomenon or process being described. The example in (5) states that the population `?population` is being modelled using the logistic approach.

$$\texttt{(model} \quad [\langle participant \rangle \mid \langle relation \rangle] \quad \langle name \rangle\texttt{)} \quad (4)$$

$$\texttt{(model ?population logistic)} \quad (5)$$

The knowledge base

The knowledge base employed by the compositional ecological modeller consists of two main constructs: property definitions and model fragments.

Property definitions describe features of interest to the application requiring a scenario model. A typical example of a feature of interest is the requirement that a certain variable in the model is endogenous or exogenous.

To be more specific, the property definitions below describe when a variable `?v` is *endogenous* and *exogenous* respectively.

```
(defproperty endogenous
  :source-participants ((?v :type variable))
  :structural-condition ((or (== ?v *) (d/d ?v *)))
  :property (endogenous ?v))
```

```
(defproperty exogenous
  :source-participants ((?v :type variable))
  :structural-condition ((not (endogenous ?v)))
  :property (exogenous ?v))
```

The first property definition states that whenever either `?v` $=$ `*` or $\frac{d}{dt}$`?v` $=$ `*` is true (where `*` matches any constant or formula), `?v` is deemed to be endogenous. According to the second property definition, a variable is said to be *exogenous* if such an object exists and it is not endogenous.

By describing such features formally in the knowledge base, property definitions enable them to be imposed as criteria on the selection of scenario models. In this way, the variable describing the size of a particular population in an eco-system, for instance, can be forced to be endogenous. Formally,

Definition 1 *A* property definition Π *is a tuple* $\langle P^s, \Phi, \pi \rangle$ *where* $P^s = \{p_1^s, \ldots p_m^s\}$ *is a set of* source-participants, *a predicate calculus sentence* Φ *whose free variables are elements of* P^s*, and a relation* π*, whose free variables are also elements of* P^s*, such that*

$$\forall p_1^s, \ldots, \forall p_m^s \Phi \rightarrow \pi$$

Required properties can be specified in two different ways: either *globally* as goals for the scenario model construction or locally as a *required purpose* of a certain model fragment. The latter use of model properties will be illustrated later.

Model fragments are the building blocks with which scenario models are constructed. For example, the model fragment below states that a population `?p` can be described by two variables `?p-size` (describing the size of `?p`) and `?p-change` (describing the rate of change in population size) and a differential equation

$$\frac{d}{dt}\texttt{?p-size} = \texttt{?p-change}$$

The usage of this partial scenario model is subject to two conditions: (1) the growth phenomenon is relevant with regard to `?p`, and (2) the variable `?p-change` is endogenous in the eventual scenario model. The former requirement is indicated by the relevance assumption and the latter by the purpose-required property:

```
(defModelFragment population-growth
  :source-participants ((?p :type population))
  :assumptions ((relevant growth ?p))
  :target-participants ((?p-size :type variable)
                        (?p-change :type variable))
  :postconditions ((size-of ?p-size ?p)
                   (change-of ?p-change ?p)
                   (d/dt ?p-size ?p-change))
  :purpose-required ((endogenous ?p-change)))
```

The purpose-required property is usually satisfied by additional model fragments, such as the one below:

```
(defModelFragment logistic-population-growth
  :source-participants ((?p :type population)
                        (?p-size :type variable)
                        (?p-change :type variable))
  :structural-conditions ((size-of ?p-size ?p)
                          (change-of ?p-births ?p))
  :assumptions ((model ?p-size logistic))
  :target-participants ((?r :type parameter)
                        (?k :type variable)
                        (?d :type variable))
  :postconditions ((capacity-of ?k ?p)
                   (density-of ?d ?p-size)
                   (== ?d (C+ (/ ?p-size ?k)))
    (== ?p-change (- (* ?r ?p-size (- 1 ?d))))))
```

The latter model fragment implements the logistic population growth model. It is instantiated if variables exist that describe the size and change in a population and it is applied if the logistic model for population size has been selected.

Generally speaking, model fragments are rules of inference that describe how new knowledge can be derived from existing knowledge by committing the emerging model to certain assumptions. They are *instantiated* by matching source-participants to existing participants in the scenario or emerging model and by matching the structural conditions to corresponding relations. If *applied*, the target-participants and postconditions of the model fragment are instantiated and added to the resulting model. Following (6), an instantiated model fragment is applied if its assumptions are deemed true. Formally,

Definition 2 *A model fragment* μ *is a tuple* $\langle P^s, P^t, \Phi^s, \Phi^t, A, \Pi\rangle$ *where* $P^s = \{p^s_1, \ldots p^s_m\}$ *is a set of variables called* source-participants*,* $P^t = \{p^t_1, \ldots, p^t_n\}$ *is a set of variables called* target-participants*,* $\Phi^s = \{\phi^s_1, \ldots, \phi^s_v\}$ *is a set of relations, called* structural conditions*, whose free variables are elements of* P^s*,* $\Phi^t = \{\phi^t_1, \ldots, \phi^t_s\}$ *is a set of relations, called* postconditions*, whose free variables are elements of* $P^s \cup P^t$*,* $A = \{a_1, \ldots, a_t\}$ *is a set of relations, called* assumptions*, and* Π *is a set of relations, called* purpose-required properties*, such that for* $i = 1, \ldots, s$*:*

$$\forall p^s_1, \ldots, \forall p^s_m, \exists p^t_1, \ldots, \exists p^t_n\ \phi^s_1 \wedge \ldots \wedge \phi^s_v \rightarrow (a_1 \wedge \ldots \wedge a_t \rightarrow \phi^t_i) \tag{6}$$

$$\forall \pi \in \Pi, \forall p^s_1, \ldots, \forall p^s_m, \forall p^t_1, \ldots, \forall p^t_n, \phi^s_1 \wedge \ldots \wedge \phi^s_v \wedge a_1 \wedge \ldots \wedge a_t \wedge \neg\pi \rightarrow \bot \tag{7}$$

Note that, in this work, each property definition $\langle P^s, \Phi, \pi\rangle$ is equivalent to a model fragment $\langle P^s, \{\}, \Phi, \{\pi\}, \{\}, \{\}\rangle$.

In most compositional modellers, such as (Levy, A.Y., Iwasaki, Y., & Fikes, R. 1997; Nayak, P.P. & Joskowicz, L. 1996; Rickel, J. & Porter, B. 1997; Heller, U. & Struss, P. 1998; 2001), model fragments represent direct translations of components of physical systems into influences between variables. Because the compositional modeller presented herein aims to serve as an ecological model repository, the contents of the model fragments employed in this work differs from that of conventional compositional modellers in two important regards:

Firstly, model fragments contain partial models describing certain phenomena instead of influences. These partial models normally correspond to those developed in ecological modelling research. Typical examples include the logistic population growth model (Verhulst 1838) and the Holling predation model (Holling 1959) devised in the population dynamics literature.

Secondly, the partial models contained in the model fragments often need to be composed incrementally. For example, the aforementioned sample model fragment `logistic-population-growth` requires an emerging scenario model, which may be generated by the other sample model fragment `population-growth`. Thus, one model fragment, e.g. `logistic-population-growth`, can expand on the partial model contained in another, e.g. `population-growth`. Because one model fragment may expand on another, it is presumed that no model fragment μ generates new relations that are preconditions of model fragments that μ expands on. This would make little sense in the context of this compositional modeller as it would imply a recursive extension of an emerging scenario model with the same set of variables and equations.

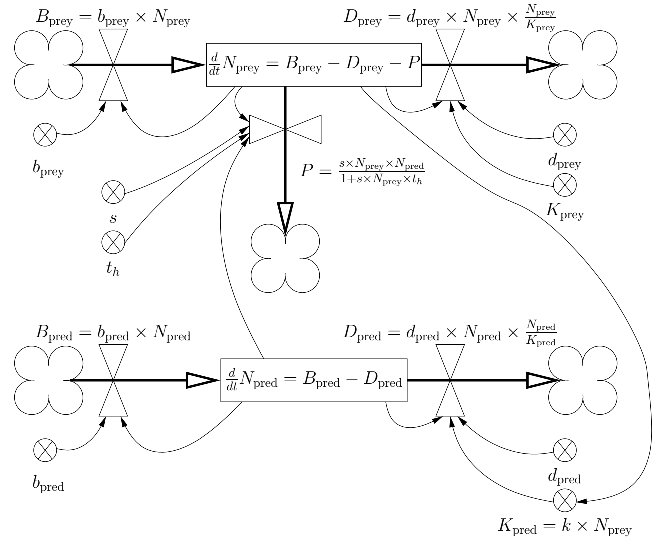

Figure 1: Behavioural model of the two species predation scenario

The compositional modelling problem

The objective of the compositional modeller presented herein is to translate a scenario describing an ecological system into a scenario model of the behaviour of that ecological system. A typical example of a scenario studied in ecology textbooks is

```
(defScenario pred-prey-prey-scenario
  :entities ((predator :type population)
             (prey :type population))
  :relations ((predation predator prey)))
```

Here, an ecological system is depicted which consists of a `predator` population that feeds on a `prey` population. The population dynamics community has developed a number of mathematical models to describe the phenomena that take place in a system like this, such as the logistic model of population growth (Verhulst 1838) and the Holling model of predation between species (Holling 1959). Figure 1 shows a mathematical model of the aforementioned scenario that combines the logistic and Holling models and that employs the system dynamics stock-flow formalism (Ford, A. 1999).

Generally speaking, the compositional ecological modelling problem (CEMP) addressed in this paper can be formalised as follows:

Definition 3 *Given a scenario* $\langle P_s, R_s\rangle$*, a set of model fragments* F*, a set of property definitions* D*, and a set of required property instances* Π*, a CEMP is the problem of finding a scenario model* $\langle P_m, R_m\rangle$*, which is not inconsistent*

($\langle P_s, R_s\rangle, F, F \nvdash \bot$), such that it logically follows from the scenario ($\langle P_s, R_s\rangle, F \vdash \langle P_m, R_m\rangle$), and that it logically entails the required property instances ($\langle P_m, R_m\rangle, D \vdash \Pi$).

Casting a CEMP as an aDCSP

The process of casting a CEMP as an aDCSP involves two stages. First, a space of plausible models is generated. This is achieved by first creating a hypergraph, or the *model space*, containing all plausible models and the assumptions they depend on. Then, this model space is translated into an aDCSP.

Scenario + Knowledge Base = Model Space

The model space is constructed as follows. First, it is initialised with the participants and relations of the given scenario. By applying model fragments, new participants and relations are inferred from ones that are already part of the scenario space. The inferences described by the applied model fragments are stored explicitly as ATMS justifications. Finally, the model space is extended with the causes of inconsistencies, including unsatisfied properties and relations that can not be combined with regard to the domain knowledge or representational framework.

More formally, a model space is an ATMS (de Kleer, J. 1986) containing all the participants, relations and assumptions that can be instantiated from a given scenario. In this work, the generalised version of the ATMS, introduced in (de Kleer, J. 1988) is employed as it allows the use of negations of nodes in the justifications. The algorithm GENERATEMODELSPACE($\langle P_s, R_s\rangle$) describes how such a model space can be created from a scenario $\langle P_s, R_s\rangle$. It first initialises the model space θ with the participant instances (P_s) and the relation instances (R_s) from the scenario. Then, for each model fragment whose source-participants and structural conditions match participants and relations already in θ, new instances of its target-participants, assumptions and postconditions are added to θ. Because each property definition $\langle P^s, \Phi, \pi\rangle$ is equivalent to a model fragment $\langle P^s, \{\}, \Phi, \{\pi\}, \{\}, \{\}\rangle$, this procedure applies to property definitions as well as model fragments. Matching the source-participants and structural conditions of a model fragment μ to the emerging model space is performed by the function match(μ, θ, σ), where μ is the model fragment being matched, and σ is a substitution from the source-participants of μ to participant instances. It is specified as given below:

$$\text{match}(\mu,\theta,\sigma) = \begin{cases} \text{true} & \text{if} \quad \sigma = \{p^s_1/o_1, \ldots, p^s_m/o_m\} \wedge \\ & \quad\; P^s = \{p^s_1, \ldots, p^s_m\} \wedge \\ & \quad\; o_1 \in \theta \wedge \ldots \wedge o_m \in \theta \wedge \\ & \quad\; \forall \phi \in \Phi^s, \sigma\phi \in \theta \\ \text{false} & \text{otherwise} \end{cases}$$

with $\mu = \langle P^s, P^t, \Phi^s, \Phi^t, A, \Pi\rangle$.

Algorithm 1: GENERATEMODELSPACE($\langle P_s, R_s\rangle$)

$\theta \leftarrow$ new ATMS;
for each $o \in P_s$, add-node(θ, o);
for each $r \in R_s$, add-node(θ, r);
for each $\mu, \sigma, (\mu = \langle P^s, P^t, \Phi^s, \Phi^t, A, \Pi\rangle) \wedge$ match(μ, θ, σ)
do
- justification $\leftarrow \emptyset$;
- **for each** $a \in A$
  do newnode $\leftarrow$ add-node($\theta, (\sigma a)$); justification $\leftarrow$ justification $\cup$ {newnode};
- **for each** $p \in P^s$
  do justification $\leftarrow$ justification $\cup$ {find-node($\theta, (\sigma p)$)};
- **for each** $\phi \in \Phi^s$
  do justification $\leftarrow$ justification $\cup$ {find-node($\theta, (\sigma\phi)$)};
- add-node($\theta, n_{(\sigma,\mu)}$);
- add-justification($\theta, n_{(\sigma,\mu)}, \wedge_{n \in \text{justification}} n$);
- **for each** $p \in P^t$
  do $\sigma \leftarrow \sigma \cup \{p/\text{gensym}()\}$; $o \leftarrow$ add-node($\theta, (\sigma p)$); add-justification($\theta, o, n_{(\sigma,\mu)}$);
- **for each** $\phi \in \Phi^t$
  do $o \leftarrow$ add-node($\theta, (\sigma\phi)$); add-justification($\theta, o, n_{(\sigma,\mu)}$);

for each $n_1, \ldots, n_m$, inconsistent($\{n_1, \ldots, n_m\}$)
do add-justification($\theta, n_\bot, n_1 \wedge \ldots \wedge n_m$);

Each match, specified by a model fragment $\mu = \langle P^s, P^t, \Phi^s, \Phi^t, A, \Pi\rangle$ and a substitution σ, is processed as follows:

- For each assumption $a \in A$, a new node, denoting the assumption instance σa, is created and added to θ.
- Then, a new node $n_{(\sigma,\mu)}$, denoting the instantiation of μ via substitution σ, is created, added to θ and justified by the implication:

$$(\wedge_{a \in A} \sigma a) \wedge (\wedge_{p \in P^s} \sigma p) \wedge (\wedge_{\phi \in \Phi^s} \sigma\phi) \rightarrow n_{(\sigma,\mu)}$$

- Finally, a new instance for each target-participant $p \in P^t$ and for each postcondition $\phi \in \Phi^t$ is created. For the target-participants, this involves creating a new symbol for each new participant instance with the function gensym() and extending σ with the substitution $\{p/\text{gensym}()\}$. A new node n is created and added to θ for each new participant instance σp and for each new instantiated relation $\sigma\phi$. Each of these nodes is justified by the implication $n_{(\sigma,\mu)} \rightarrow n$.

Once all possible applications of model fragments have been exhausted, the inconsistencies in the model space are identified and recorded in the ATMS. In the algorithm, nogoods are generated for each set $\{n_1, \ldots, n_m\}$ of inconsistent nodes, denoted inconsistent($\{n_1, \ldots, n_m\}$). There are three sources of inconsistencies that are each reported to the ATMS in a different way:

- *Global properties*: Let π be an instance of a global property that any scenario model must satisfy. Then, any combination of assumptions and negations of assumptions that prevents π from being satisfied is inconsistent. Therefore, inconsistent($\{\neg\pi\}$) must be reported for any required global property π.
- *Purpose-required properties*: Any application of a model fragment $\mu = \langle P^s, P^t, \Phi^s, \Phi^t, A, \Pi\rangle$ without satisfying its purpose-required properties Π yields an inconsistency (see (7). Hence, for each node $n_{(\sigma,\mu)}$ denoting the instantiation

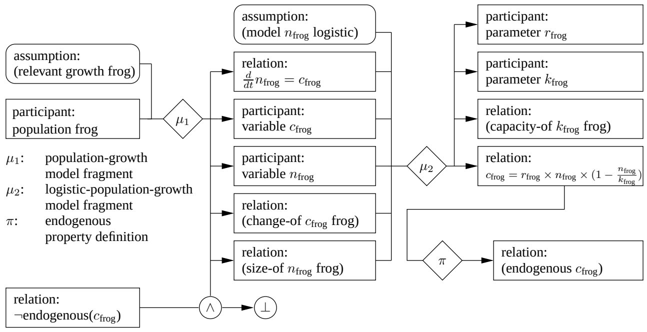

Figure 2: Partial model space

of μ via substitution σ, and for each node $n_{\sigma\pi}$ describing the appropriate instance of a purpose-required property $\pi \in \Pi$, inconsistent$(\{n_{(\sigma,\mu)}, \neg n_{\sigma\pi}\})$ is reported.

- *Non-composable relations*: In any mathematical formalism designed to describe simulation models of dynamic systems, certain combinations of relations may over-constrain the model, and hence, be unsuitable for generating the behaviour of a system of interest. Combinations of such non-composable relations do not yield an adequate model either and must be reported as an inconsistency as well. Although a detailed discussion of this issue is beyond the scope of this paper, the notion of non-composable relations can be illustrated by means of the system dynamics formalism employed to represent the ecological models. Here mathematical relations are specified as influences from one or more variables to another variable. These include assignment relations containing the composable operators defined as part of the compositional modelling language CML (Bobrow, D. *et al.* 1996). Relations describing assignments, terms and factors influencing the same variable are mutually non-composable. However, different terms or different factors influencing the same variable can be composed with one another.

To illustrate the model space construction algorithm, figure 2 presents a small sample model space. It results from the application of the `population-growth` and `logistic-population-growth` model fragments and the `endogenous` property definition, which were described earlier, for a single population "frog". If a larger scenario involving multiple populations and relations between these populations were specified, a similar partial model space would be generated for each individual population.

From model space to aDCSP

Once the model space has been constructed, it can be translated into an aDCSP. The translation procedure, summarised as algorithm CREATEADCSP(), consists of three steps as described below:

Algorithm 2: CREATEADCSP()

comment: σ is the set of substitutions
$\sigma \leftarrow \{\}$;
comment: Generate attributes and domains
for each A, assumption-class(A)
do {
  $x \leftarrow$ create-attribute();
  $D(x) \leftarrow \{\}$;
  $\sigma \leftarrow \sigma \cup \{A/x\}$;
  for each $a \in A$
  do {
    $v \leftarrow$ create-value();
    $D(x) \leftarrow D(x) \cup \{v\}$;
    $\sigma \leftarrow \sigma \cup \{a/x : v\}$;
comment: Generate activity constraints
for each A, assumption-class(A)
do {
  $s \leftarrow$ subject(A);
  for each $\{a_{1\top}, \ldots, a_{p\top}, \neg a_{1\bot}, \ldots, \neg a_{q\bot}\} \in \mathcal{L}(s)$
    do add$(\sigma a_{1\top} \wedge \ldots \wedge \sigma a_{p\top} \wedge \sigma\neg a_{1\bot} \wedge \ldots \wedge \sigma\neg a_{q\bot} \rightarrow$ active$(\sigma A))$;
comment: Generate compatibility constraints
for each $\{a_{1\top}, \ldots, a_{p\top}, \neg a_{1\bot}, \ldots, \neg a_{q\bot}\} \in \mathcal{L}(n_\bot)$
  do add$(\sigma a_{1\top} \wedge \ldots \wedge \sigma a_{p\top} \wedge \sigma\neg a_{1\bot} \wedge \ldots \wedge \sigma\neg a_{q\bot} \rightarrow \bot$;

1. *Generate the attributes and domain values from the assumptions*. The aDCSP attributes correspond to the underlying assumption classes (i.e. groups of assumptions indicating alternative choices with regards to the same model construction decision). A relevance assumption and its negation jointly form an assumption class. For example, A_1 $=\{$`(relevant growth frog)`, $\neg$`(relevant growth frog)`$\}$ specifies such an assumption class. The set of model assumptions involving the same participants or relations, but with different model names (i.e. $\langle$name$\rangle$ in the formal specification (4)), also form an assumption class. For instance, A_2 $=\{$`(model n`$_{\text{frog}}$ `exponential)`, `(model n`$_{\text{frog}}$ `logistic)`, `(model n`$_{\text{frog}}$ `other)`$\}$, where $\mathtt{n}_{\text{frog}}$ is a variable denoting the size of a population, specifies such an assumption class. Running this step of the algorithm, an attribute is created for each assumption class, with the domain of such an attribute consisting of all assumption instances in the assumption class.
2. *Create activity constraints*. The attributes and domain values created in the previous step are only meaningful in situations where the participant and/or relation instances contained in the arguments of the corresponding assumptions exist. For example, the assumption `(model n`$_{\text{frog}}$ `logistic)` is only relevant if the participant instance $\mathtt{n}_{\text{frog}}$ exists. It is clear that all assumptions within one assumption class have the same participant and/or relation instances as their arguments. Because each assumption class corresponds to one attribute, the attribute can be activated if and only if the participant and/or relation instances associated with the related assumption class is active. Therefore, this step creates activity constraints that activate an attribute based on the conjunction of the environments contained within the labels of the participants/relations of the assumption class. For instance, as can be deduced from figure 2, $\mathtt{n}_{\text{frog}}$ is activated when `(relevant growth frog)` is committed. Thus, the attribute corresponding to assumption class A_2, defined in step 1, is activated under the attribute value assignment associated with the `(relevant growth frog)` assumption.
3. *Create compatibility constraints*. In the ATMS (or model

space), all sources of inconsistencies are contained in the label of the nogood node. Therefore, the compatibility constraints are created directly by translating the environments in the label $\mathcal{L}(\perp)$ into the corresponding conjunctions of attribute-value assignments.

Towards the analysis of complexity

The translation method presented above enables the use of efficient CSP solution algorithms. This section briefly discusses the complexity issues involved.

The algorithm complexity arises from four sources: 1) model space construction, 2) label propagation in the ATMS, 3) model space to aDCSP translation, and 4) aDCSP solution. GENERATEMODELSPACE$(\langle P_s, R_s \rangle)$ essentially performs a fixed sequence of instructions and produces a small set of nodes and inferences for each match of a model fragment. Therefore, its time and space complexity is linear with respect to the number of possible matches of model fragments. CREATEADCSP() extracts certain information from the model space and rewrites it in a different formalism without further manipulations. Therefore, its time and space complexity is linear with respect to the size of the model space.

The label propagation algorithm of an ATMS is known to have an exponential time complexity. However, because the model space is built up incrementally (by GENERATEMODELSPACE$(\langle P_s, R_s \rangle)$) from the root nodes of the ATMS network (i.e. the ones that correspond to facts and have no antecedents) to the leaf nodes (i.e. the ones that have have no consequents, other than the nogood node) and because the inconsistencies are added at the end, this complexity only increases exponentially with the depth of the network and the number of participants and relations in individual model fragments, rather than with the size of the model space. This limits the complexity impact of label propagation. Firstly, the depth of the ATMS network is restricted by the domain. In many conventional compositional modellers, where model fragments are direct translations from scenario components to scenario model equations, this depth would be only one. Empirically, constructing the model space for sophisticated eco-systems, the depth of a model space never exceeded 8. Secondly, the size of the individual model fragments does not change significantly with the size of knowledge base.

As for the fourth and final source of complexity, it is driven by the nature that the aDCSP solution algorithm must determine a consistent combination of assumptions in the model space. The space of attribute value assignments, in which a solution must be found, increases exponentially with the size of the number of assumptions, and hence, with the model space. Therefore, the overall complexity of the present approach is largely dominated by the aDCSP solution algorithm employed. Fortunately, with the advances of CSP research, a number of efficient methods are available for such use. This helps minimise the overhead incurred for compositional modelling. Note that most compositional modellers, with the exception of certain approaches (e.g. (Nayak, P.P. & Joskowicz, L. 1996)) that solve a problem of reduced complexity, perform a similar search but with purpose-built algorithms. This work hopes to improve the efficiency of compositional modellers by enabling the use efficient aDCSP solution algorithms.

A Population Dynamics Example

The examples used throughout the previous sections were taken from a more extensive application study of the present work. The application was aimed to construct a repository of basic population dynamic models, describing the phenomena of growth, predation and competition. This section presents an overview of how the proposed approach is employed in this application.

Knowledge base

In essence, the knowledge base of a compositional modeller is a collection of the component models, each extended with a specification of the requirements for composition with other component models. The first model fragment, shown earlier, describes the most basic recurring part of population growth models (i.e. the differential equation `?p-size` $= \frac{d}{dt}$`?p-change`). It contains several conditions for inclusion in a model: 1) a population `?p` must exist in the scenario, 2) the growth phenomenon must be relevant to `?p`, and 3) the new variable `?p-change`, describing the change in population size, must be an endogenous variable.

```
(defModelFragment population-growth
  :source-participants ((?p :type population))
  :assumptions ((relevant growth ?p))
  :target-participants ((?p-size :type variable)
                        (?p-change :type variable))
  :postconditions ((size-of ?p-size ?p)
                   (change-of ?p-change ?p)
                   (d/dt ?p-size ?p-change))
  :purpose-required ((endogenous ?p-change)))
```

The variable `?p-change` becomes endogenous if the model contains an equation describing change in population size. These equations differ between population growth models. The following model fragments describe two such component models: exponential growth (Malthus 1798) and logistic growth (Verhulst 1838).

```
(defModelFragment exponential-population-growth
  :source-participants ((?p type population)
                        (?p-size :type variable)
                        (?p-change :type variable))
  :structural-conditions ((size-of ?p-size ?p)
                          (change-of ?p-change ?p))
  :assumptions ((model ?size exponential))
  :target-participants ((?r :type parameter))
  :postconditions ((== ?p-change (* ?r ?p-size))))

(defModelFragment logistic-population-growth
  :source-participants ((?p :type population)
                        (?p-size :type variable)
                        (?p-change :type variable))
  :structural-conditions ((size-of ?p-size ?p)
                          (change-of ?p-births ?p))
  :assumptions ((model ?p-size logistic))
  :target-participants ((?r :type parameter)
                        (?k :type variable)
                        (?d :type variable))
  :postconditions ((capacity-of ?k ?p)
                   (density-of ?d ?p-size)
                   (== ?d (C+ (/ ?p-size ?k)))
    (== ?p-change (* ?r ?p-size (- 1 ?d)))))
```

There is one twist to compositional modelling of population growth. Sometimes, the actual growth model is implicitly contained in another type of model. In such cases, the growth phenomenon and the corresponding differential equation is still relevant, but none of the dedicated growth models can be employed. For example, as to be shown later, the Lotka-Volterra predation model comes with its own equations describing growth.

The model fragment `other-growth` allows for an empty growth model, named `other`, to be selected. However, due to the purpose-required property stating that any instance of `?p-change` must be endogenous, this empty model can only be selected if a growth model is implicitly included elsewhere.

```
(defModelFragment other-growth
  :source-participants ((?p :type population)
                        (?p-size :type variable)
                        (?p-change :type variable))
  :structural-conditions ((size-of ?p-size ?p)
                          (change-of ?p-births ?p))
  :assumptions ((model ?p other)))
```

In addition to population growth, two other phenomena are included in the knowledge base: predation and competition. Predation and competition relations between species are represented by predicates over the populations: e.g. `(predation grizzly-bear salmon)` and `(competition grizzly-bear brown-bear)`. However the existence of a phenomenon does not necessarily mean that it must be contained within the model. It would make little sense to model predation and competition without modelling the size of the populations, because models of these phenomena relate population sizes to one another. Therefore, the incorporation of the predation phenomenon is made dependent upon the existence of variables representing population size. Also, human expert modellers may prefer to leave a phenomenon out of the resulting model. To keep this choice open, the following two model fragments construct a participant representing the phenomena of predation and competition and make it dependent upon a relevance assumption:

```
(defModelFragment predation-phenomenon
  :source-participants ((?pred :type population)
                        (?prey :type population)
                        (?pred-size :type variable)
                        (?prey-size :type variable))
  :structural-conditions ((predation ?pred ?prey)
                          (size-of ?pred-size ?pred)
                          (size-of ?prey-size ?prey))
  :assumptions ((relevant predation ?pred ?prey))
  :target-participant ((?pred-phen :type phenomenon))
  :postconditions ((pred-phen ?pred-phen ?pred ?prey))
  :purpose-required ((has-model ?pred-phen)))
```

```
(defModelFragment competition-phenomenon
  :source-participants ((?p1 :type population)
                        (?p2 :type population)
                        (?p1-size :type variable)
                        (?p2-size :type variable))
  :structural-conditions ((competition ?p1 ?p2)
                          (size-of ?p1-size ?p1)
                          (size-of ?p2-size ?p2))
  :assumptions ((relevant competition ?p1 ?p2))
  :target-participant ((?comp-phen :type phenomenon))
  :postconditions ((comp-phen ?comp-phen ?p1 ?p2))
  :purpose-required ((has-model ?comp-phen)))
```

Both model fragments have a purpose-required property of the form `(has-model ?phen)`. This property expresses the condition that a model must exist with respect to a phenomenon:

```
(defproperty has-model
  :source-participants ((?p :type phenomenon))
  :structural-conditions ((is-model-of ?p *))
  :property (has-model ?p))
```

The next two model fragments implement such models (thereby satisfying the above has-model purpose-required property) for the predation phenomenon between two populations. They describe two well-known predation models: the Lotka-Volterra model (Lotka 1925; Volterra 1926) and the Holling model (Holling 1959).

```
(defModelFragment Lotka-Volterra
  :source-participants ((?pred-phen :type phenomenon)
                        (?pred :type population)
                        (?pred-size :type variable)
                        (?pred-change :type variable)
                        (?prey :type population)
                        (?prey-size :type variable)
                        (?prey-change :type variable))
  :structural-conditions
    ((predation-phenomenon ?pred-phen ?pred ?prey)
     (size-of ?pred-size ?pred)
     (change-of ?pred-change ?pred)
     (size-of ?prey-size ?prey)
     (change-of ?prey-change ?prey))
  :assumptions ((model ?pred-phen lotka-volterra))
  :target-participants ((?prey-rate :type variable)
                        (?pred-factor :type variable)
                        (?prey-factor :type variable)
                        (?pred--rate :type variable))
  :postconditions
    ((== ?prey-change (- (* ?prey-rate ?prey-size)
         (* ?prey-factor ?prey-size ?pred-size)))
     (== ?pred-change (- (* ?pred-factor ?prey-size ?pred-size)
                         (* ?pred-rate ?pred-size)))
     (is-model-of ?pred-phen lotka-volterra)))
```

As mentioned earlier, the Lotka-Volterra model introduces its own growth model for the prey and predator populations by assigning specific equations to the variables, describing changes in the sizes of the predator and prey populations, `?pred-change` and `?prey-change` respectively. Thus, it satisfies the purpose-required property in the application of the `population-growth` model fragment for the `?prey` and `?pred` populations.

```
(defModelFragment Holling
  :source-participants ((?pred-phen :type phenomenon)
                        (?pred :type population)
                        (?pred-size :type variable)
                        (?capacity :type variable)
                        (?prey :type population)
                        (?prey-size :type variable))
  :structural-conditions
    ((predation-phenomenon ?pred-phen ?pred ?prey)
     (size-of ?pred-size ?pred)
     (size-of ?prey-size ?prey)
     (capacity-of ?capacity ?pred))
  :assumptions
    ((model ?pred-phen holling))
  :target-participants
    ((?search-rate :type variable)
     (?handling-time :type variable)
     (?prey-req :type variable)
     (?predation :type variable))
  :postconditions
    ((d/dt ?pred-size (C- ?predation))
     (== ?predation (/ (* ?search-rate ?prey-size ?pred-size)
         (+ 1 (* ?search-rate ?prey-size ?handling-time))))
     (== ?capacity (C+ (* ?prey-req ?prey)))
     (is-model-of ?pred-phen holling)))
```

The Holling model employs a variable denoting the capacity of a population. Such a variable may be introduced by a logistic growth model. In practice, logistic growth models and Holling predation models are often used in conjunction. The compositional modeller need not be aware of such combinations of models, however. All it needs to know is the prerequisites of the individual component models contained within each model fragment.

The final model fragment in the knowledge base implements a model of competition between two species. As this model fragment contains the only population competition model in the knowledge base, it does not contain a model assumption to describe the model.

```
(defModelFragment competition
  :source-participants
    ((?comp-phen :type phenomenon)
     (?p1 :type population)
     (?p1-size :type variable)
     (?p1-density :type variable)
     (?p1-capacity :type variable)
     (?p2 :type population)
     (?p2-size :type variable)
     (?p2-density :type variable)
     (?p2-capacity :type variable))
  :structural-conditions
    ((comp-phen ?comp-phen ?p1 ?p2)
     (density-of ?p1-density ?p1-size)
     (capacity-of ?p1-capacity ?p1-size)
     (density-of ?p2-density ?p2-size)
     (capacity-of ?p2-capacity ?p2-size))
  :target-participants
    ((?weight-12 :type variable)
     (?weight-21 :type variable))
  :postconditions
    ((== ?p1-density
         (C+ (/ (* ?weight-12 ?p2-size) ?p1-capacity)))
     (== ?p2-density
         (C+ (/ (* ?weight-21 ?p1-size) ?p2-capacity)))
     (is-model-of ?comp-phen default)))
```

Model space

A model space is constructed when the knowledge base is instantiated with respect to a given scenario. Consider for example the following scenario

```
(defScenario pred-prey-prey-scenario
  :entities ((predator :type population)
             (prey1 :type population)
             (prey2 :type population))
  :relations ((predation predator prey1)
              (predation predator prey2)
              (competition prey1 prey2)))
```

This scenario describes a `predator` population that preys on two other populations, `prey1` and `prey2`, whilst the two prey populations compete with one another.

The full specification of the model space is too unwieldy to present here but an abstract graphical representation of the model space for this scenario is shown in figure 3 instead. This model space contains the following knowledge:

- From each of the three populations in the scenario, a set of three population growth models (i.e. `exponential`, `logistic` and `other`) is derived. This inference is dependent upon a relevance assumption of the population growth phenomenon, and a model assumption that corresponds to one of the three population growth models.
- From both predation relations (i.e. `(predation predator prey1)` and `(predation predator prey2)`), and the populations related by them, a set of two predation models (i.e. `Lotka-Volterra` and `Holling`) is derived. This inference is dependent upon a relevance assumption of the predation phenomenon and a model assumption that corresponds to one of the two predation models.
- From the competition relation `(competition prey1 prey2)`, and the populations related by it, a competition model is derived. Because there is only one competition model, the inference of the competition model is only dependent upon a relevance assumption that corresponds to the competition phenomenon.

In addition to the hypergraph of figure 3, the model space also contains a number of constraints on the conjunctions of assumptions that are consistent. As explained earlier, these stem from two sources: 1) non-composable relations and 2) purpose-required properties. An example will be given of each type.

Let `prey1-size` be the variable representing the size of the `prey1` population, and let `predation-phenomenon-1` denote the predation phenomenon between `predator` and `prey1`. In this example, the model fragments `exponential-population-growth` and `Lotka-Volterra` will each generate a different equation for computing the value of a variable representing the change in `prey1-size`. Because both equations can not be composed, the following inconsistency is generated:

(relevant growth prey1) $\wedge$ (model prey1-size exponential)$\wedge$
(relevant growth predator) $\wedge$ (relevant predation predator prey1)$\wedge$
(model predation-phenomenon-1 lotka-volterra) $\rightarrow \bot$

Other inconsistencies arise from purpose-required properties. For example, if the model fragment `predation-phenomenon` is applicable and the predation relation is deemed relevant, then the purpose-required property `(has-model ?pred-phen)` will become a condition for consistency. Under certain combinations of assumptions, this property may not be satisfied. Say, when the Holling predation and exponential growth models are both selected, the Holling model is not generated because there is no `?capacity` for which `(capacity ?capacity ?pred)` is true. No predation model is created in this case (because the `Holling` model fragment can not be instantiated), even though the predation phenomenon is deemed relevant under this set of assumptions. This is inconsistent with the `has-model` purpose-required property in the `predation-phenomenon` model fragment, and the responsible combination of assumptions is therefore marked as nogood.

(relevant growth predator) $\wedge$ (model predator-size exponential)$\wedge$
(relevant growth prey1) $\wedge$ (model prey1-size exponential)$\wedge$
(relevant predation predator prey1)$\wedge$
(model predation-phenomenon-1 holling) $\rightarrow \bot$

aDCSP and solution

The model space is translated into an aDCSP to enable the selection of a consistent set of assumptions, using advanced CSP solution techniques. The aDCSP derived from the above model space is depicted in figure 4.

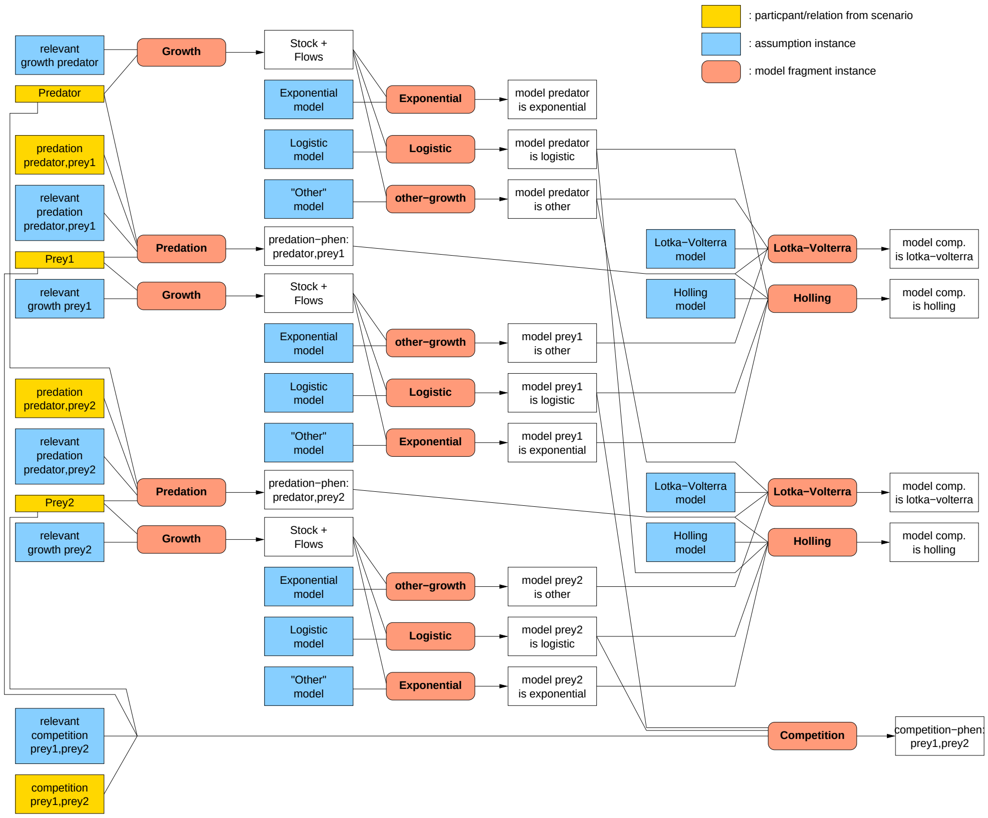

Figure 3: Model space for the 1 predator and 2 competing prey scenario

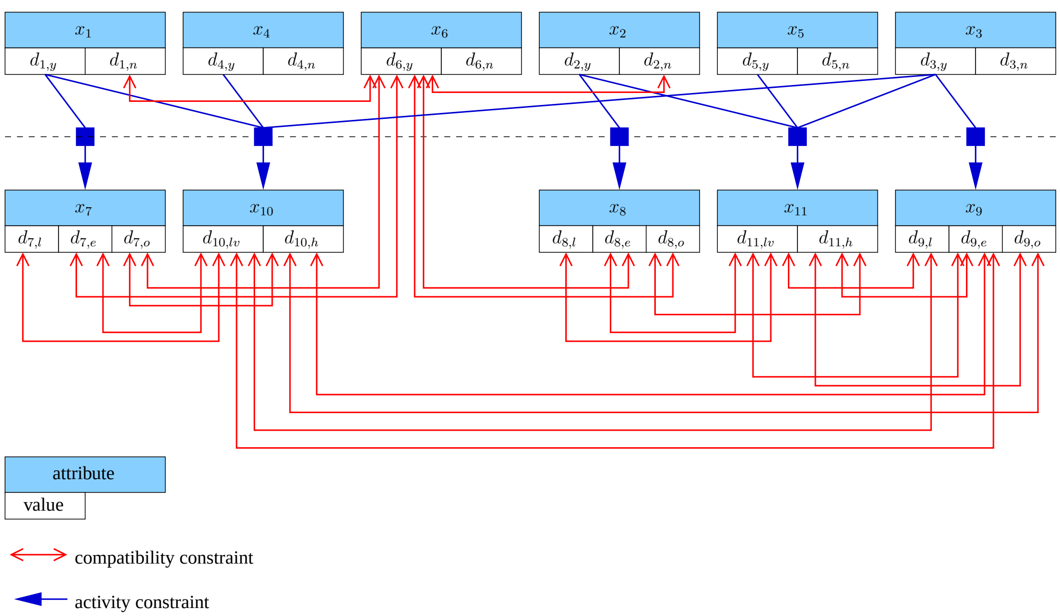

Figure 4: aDCSP derived from the models space reflecting the 1 predator and 2 competing prey scenario

Attribute	Meaning
x_1	(relevant growth prey1)
x_2	(relevant growth prey2)
x_3	(relevant growth predator)
x_4	(relevant predation predator prey1)
x_5	(relevant predation predator prey2)
x_6	(relevant competition prey1 prey2)
x_7	(model size-1 *)
x_8	(model size-2 *)
x_9	(model size-3 *)
x_{10}	(model predation-phenomenon-1 *)
x_{11}	(model predation-phenomenon-2 *)

Table 1: Attribute list

This aDCSP contains 11 attributes. They are listed with the corresponding assumption classes in table 1. The first 6 attributes correspond to the notion of relevance phenomenon: 3 population growth phenomena, 2 predation phenomena and 1 competition phenomenon to be precise. The other 5 attributes correspond to 5 sets of models: 3 sets of population growth models and 2 sets of predation models.

Conventional aDCSP solution algorithms, such the ones presented in (Mittal, S. & Falkenhainer, B. 1990; Miguel, I. & Shen, Q. 2000), can find a number of solutions to the aDCSP depicted in figure 4. For instance, in one such solution, all phenomena are deemed relevant, *logistic* models are selected to describe the population growth phenomena and *Holling* models are chosen for the predation phenomena. This model corresponds to the following set of equations:

$$\begin{aligned}
\frac{d}{dt}\text{p-size}_{\text{predator}} &= \text{p-change}_{\text{predator}} \\
\text{d}_{\text{predator}} &= \frac{\text{p-size}_{\text{predator}}}{\text{k}_{\text{predator}}} \\
\text{p-change}_{\text{predator}} &= \text{r}_{\text{predator}} \times \text{p-size}_{\text{predator}} \times \text{d}_{\text{predator}} \\
\text{k}_{\text{predator}} &= \text{prey-req}_{\text{prey1}} \times \text{p-size}_{\text{prey1}} + \text{prey-req}_{\text{prey2}} \times \text{p-size}_{\text{prey2}} \\
\frac{d}{dt}\text{p-size}_{\text{prey1}} &= \text{p-change}_{\text{prey1}} - \text{predation}_{\text{prey1}} \\
\text{d}_{\text{prey1}} &= \frac{\text{p-size}_{\text{prey1}} + \text{weight-12} \times \text{p-size}_{\text{prey2}}}{\text{k}_{\text{prey1}}} \\
\text{p-change}_{\text{prey1}} &= \text{r}_{\text{prey1}} \times \text{p-size}_{\text{prey1}} \times \text{d}_{\text{prey1}} \\
\text{predation}_{\text{prey1}} &= \frac{\text{search-rate}_{\text{prey1}} \times \text{p-size}_{\text{prey1}} \times \text{p-size}_{\text{predator}}}{1 + \text{search-rate}_{\text{prey1}} \times \text{p-size}_{\text{prey1}} \times \text{handling-time}_{\text{prey1}}} \\
\frac{d}{dt}\text{p-size}_{\text{prey2}} &= \text{p-change}_{\text{prey2}} - \text{predation}_{\text{prey2}} \\
\text{d}_{\text{prey2}} &= \frac{\text{p-size}_{\text{prey2}} + \text{weight-21} \times \text{p-size}_{\text{prey1}}}{\text{k}_{\text{prey2}}} \\
\text{p-change}_{\text{prey2}} &= \text{r}_{\text{prey2}} \times \text{p-size}_{\text{prey2}} \times \text{d}_{\text{prey2}} \\
\text{predation}_{\text{prey2}} &= \frac{\text{search-rate}_{\text{prey2}} \times \text{p-size}_{\text{prey2}} \times \text{p-size}_{\text{predator}}}{1 + \text{search-rate}_{\text{prey2}} \times \text{p-size}_{\text{prey2}} \times \text{handling-time}_{\text{prey2}}}
\end{aligned}$$

where the variable names correspond to those used in the model fragments and the indices refer to the populations that a variable relates to.

The domain also allows for alternative models in which certain phenomena are ignored, and these models correspond to alternative solutions to the same aDCSP. There are two ways of discriminating between alternative solutions to the aDCSP. The first involves imposing additional properties on the model. The second involves assigning preferences to assumptions and/or properties. These are, however, beyond the scope of this paper.

Conclusion and Future Work

This paper has presented a novel approach to compositional modelling that enables the construction of models of ecological systems. This work differs from existing approaches in that it automatically translates the compositional modelling problem into an aDCSP. This allows formal criteria to be defined that must be satisfied by adequate scenario models. More importantly, it also enables efficient, existing and future, aDCSP solution techniques to be effectively applied to compositional modelling.

Most recently, the approach presented herein has been extended to allow assumptions to be assigned preference valuations. These preferences describe the suitability of the model design decisions represented by the assumptions they are assigned to. This variation on the CEMP can be translated into a constraint satisfaction optimisation problem (Tsang, E. 1993) with activity constraints. In its extended form, the compositional ecological modeller has been applied to automated model construction of large and complex ecosystems such as the MODMED model of Mediterranean vegetation (Legg, C.J., Muetzelfeldt, R.I., & Heathfield, D.N. 1995).

The analysis of the complexity of the present approach is rather informal. Much remains to be done in this regard, especially when comparing complexity to that of existing compositional modellers. For this, additional work will be required to adapt the current translation procedure to suit existing compositional modelling problems. Most compositional modellers are of exponential complexity, however. As they employ problem-specific solution algorithms, little is known about opportunities for improving their efficiency. This work hopes to be a first step toward further understanding this important issue.

References

Bobrow, D.; Falkenhainer, B.; Farquhar, A.; Fikes, R.; Forbus, K.D.; Gruber, T.R.; Iwasaki, Y.; and Kuipers, B.J. 1996. A compositional modeling language. In *Proceedings of the 10th International Workshop on Qualitative Reasoning about Physical Systems*, 12–21.

de Kleer, J. 1986. An assumption-based TMS. *Artificial Intelligence* 28:127–162.

de Kleer, J. 1988. A general labeling algorithm for assumption-based truth maintenance. In *Proceedings of the 7th National Conference on Artificial Intelligence*, 188–192.

Falkenhainer, B., and Forbus, K. 1991. Compositional modeling: finding the right model for the job. *Artificial Intelligence* 51:95–143.

Ford, A. 1999. *Modeling the Environment - An Introduction to System Dynamics Modeling of Environmental Systems*. Island Press.

Heller, U., and Struss, P. 1998. Diagnosis and therapy recognition for ecosystems - usage of model-based diagnosis techniques. In *Proceedings of the 12th International Symposium "Computer Science for Environment Protection"*.

Heller, U., and Struss, P. 2001. Transformation of qualitative dynamic models - application in hydro-ecology. In Hotz, L.; Struss, P.; and Guckenbienl, T., eds., *Intelligent Diagnosis in Industrial Applications*. Shaker Verlag. 95–106.

Holling, C. 1959. Some characteristics of simple types of predation and parasitism. *Canadian Entomologist* 91:385–398.

Keppens, J., and Shen, Q. 2000. Towards compositional modelling of ecological systems via dynamic flexible constraint satisfaction. In *Proceedings of the 14th International Workshop on Qualitative Reasoning about Physical Systems*, 74–82.

Keppens, J., and Shen, Q. 2001. On compositional modelling. *Knowledge Engineering Review* 16(2):157–200.

Legg, C.J.; Muetzelfeldt, R.I.; and Heathfield, D.N. 1995. Modelling vegetation dynamics in mediterranean ecosystems: Issues of scale. In *Proceedings of the 39th Symposium of the International Association for Vegetation Science*.

Levy, A.Y.; Iwasaki, Y.; and Fikes, R. 1997. Automated model selection for simulation based on relevance reasoning. *Artificial Intelligence* 96:351–394.

Lotka, A. 1925. *Elements of physical biology*. Baltimore: Williams & Wilkins Co.

Malthus, T. 1798. An essay on the principle of population. printed for J. Johnson in St. Paul's Church Yard, London, England.

Miguel, I., and Shen, Q. 1999. Hard, flexible and dynamic constraint satisfaction. *Knowledge Engineering Review* 14(3):199–220.

Miguel, I., and Shen, Q. 2000. Solution techniques for constraint satisfaction problems: Advanced approaches. *Artificial Intelligence Review*.

Mittal, S., and Falkenhainer, B. 1990. Dynamic constraint satisfaction problems. In *Proceedings of the 8th National Conference on Artificial Intelligence*, 25–32.

Nayak, P.P., and Joskowicz, L. 1996. Efficient compositional modeling for generating causal explanations. *Artificial Intelligence* 83:193–227.

Rickel, J., and Porter, B. 1997. Automated modeling of complex systems to answer prediction questions. *Artificial Intelligence* 93:201–260.

Salles, P., and Bredeweg, B. 2002. A case study of collaborative modelling: building qualitative models in ecology. In Bredeweg, B., ed., *Proceedings of the International workshop on Model-based Systems and Qualitative Reasoning for Intelligent Tutoring Systems*, 75–84.

Tsang, E. 1993. *Foundations of Constraint Satisfaction*. London and San Diego: Academic Press.

Verhulst, P. 1838. Recherches mathématiques sur la loi d'accroissement de la population. *Nouveaux mémoires de l'académie royale des sciences et belles-lettres de Bruxelles* 18:1–38.

Volterra, V. 1926. Fluctuations in the abundance of a species considered mathematically. *Nature* 118:558–560.