UNIVERSITY OF HELSINKI
DEPARTMENT OF FOREST RESOURCE MANAGEMENT
PUBLICATIONS 41

SIMO
–
Adaptable Simulation and Optimization for Forest Management Planning

Edited by Annika Kangas and Jussi Rasinmäki

ISSN 1236-1313
ISBN 978-952-10-4527-1 (pdf)

Preface

This book is the report of SIMO-project (2004-2007) carried out in the Department of Forest Resource Management, University of Helsinki.

TABLE OF CONTENTS

1. Forest planning in Finland

1.1. General

Annika Kangas and Timo Tokola

Practically taken all the forests in Finland are under intensive planning. In private forests, each year about 0.65-1.1 million hectares are planned every year (Oksanen-Peltola 1999). The plans are made for a 10-year period, and in principle new plan is made after that, but currently valid plans cover only 61% of the area of private forests. State gives subsidies for planning in private forests, so that while the price of the plan is about 8 €/ha for a private forest owner, it is around 18 € for the planning organisations, forestry centers. In private forests, planning is based on traditional compartment wise forest inventory (Koivuniemi & Korhonen 2006). It is based partly on measurements and partly on visual assessments of forest growing stock. In addition, systematic sample plots collected in NFI are used for calculating the cutting possibilities throughout Finland (e.g. Nuutinen et al. 2005)

In forest companies planning has also previously been based on field inventory carried out at 10-year intervals. Nowadays, such data is considered to be too old for day-to-day decisions, and companies currently have up-to-date forest information, based on collected data and updated with a forest simulator. When treatments are carried out, the remaining growing stock is measured and the new data is immediately stored in the databases. Thus, companies have largely given up the comprehensive inventories at given intervals.

Traditionally the collected data has been stand-level information concerning basal area, mean diameter and height by tree species and crown layer (Koivuniemi & Korhonen 2006). Then, tree-level data is predicted using a predicted diameter distribution (e.g. Maltamo 1997) and this is used to predict the growth and yield of forests using single-tree growth models (Hynynen et al. 2002).

However, the possibilities for collecting data have changed considerably in recent years. Especially applications of remote sensing are interesting also from the planning point of view. The most promising alternative for traditional field inventories is currently laser scanning (e.g. Næsset 2002, 2004, Maltamo et al. 2004, Suvanto et al. 2005). It produces data with accuracy similar to or even better than traditional field inventories. There are also a lot of other possibilities that could be used as basic data for forest planning including digital aerial photographs (e.g. Korpela 2004). Therefore, a modern forest planning package should be able to utilize data from several different sources. The new data sources also are of different resolution than the old stand wise data, e.g. pixel data, and the planning package should also be able to use data from different resolutions.

In addition to information concerning the current state of the forests, information of previous treatments and their effects on the state of the forests and nature are important in decision making. Therefore, being able to manage the historical information, both with respect to temporal and spatial characteristics, is essential. However, in current planning systems the storage and management of historical information is difficult. When the boundaries between the compartments or the compartment numbers change, the monitoring of the changes in forests is difficult.

1.2. Forest simulators

Annika Kangas and Antti Mäkinen

Predictions of forest development under different treatment options, based on forest growth and yield models, form the foundations for decision making in forest management planning. Various types of forest growth models have been proposed for the task. They can basically be either empirical models (e.g. Hynynen at al. 2002), or process-based models, which are based on ecological theories and describe the eco-physiological processes of individual trees in detail (e.g. Mäkelä et al. 2000).

Growth models can also be categorised by their level of organisation, which is usually that of either a single tree or a stand (Munro 1974). Tree-level models predict the growth of an individual tree, and stand-level model predict the increment in an aggregate variable such as mean diameter or basal area. Tree-level models can then be further categorised as spatial (distance-dependent) or aspatial (distance-independent). Distance-dependent models use information about neighbouring trees and their locations, when predicting the growth of a single tree, whereas distance-independent models do not (e.g. Tomé & Burkhart 1989). Other types of growth models include diameter distribution –based models (e.g. Bailey et al. 1981) and transition matrix models (e.g. Buongiorno & Mitchell 1980).

The history of stand-level growth models in Finland goes back to the yield tables introduced in the 19th century. Stand-level growth information was used in the earliest versions of MELA-system, but no simulators based on stand-level information have been used in recent years. Yet, in comparisons stand-level simulators have been found more robust and even more accurate than tree-level simulators (Gustavsen 1998, Mäkinen et al. 2008). Models suitable for stand-level simulators have been presented by Vuokila & Väliaho (1980) for pine (*Pinus sylvestris*) and spruce (*Picea abies*) and by Oikarinen (1983) and Saramäki (1977) for birches (*Betula pendula*, *Betula pubescens*). Other stand-level growth models for Finnish conditions include the volume growth models presented by Gustavsen (1977) and Nyyssönen & Mielikäinen (1978).

The most commonly used growth models in Finland are empirical and distance-independent, as it has been too expensive to acquire spatial information on the forest structure for practical forest management planning purposes and generated spatial information does not necessarily improve the predictions (Hynynen et al. 2002). These models are used in the MOTTI stand-level simulator (Hynynen et al. 2005, Salminen et al. 2005) and also in MELA forest optimization package (Siitonen 1996). In addition, the models of Nyyssönen and Mielikäinen (1978) are used as growth models in MONSU (Pukkala 2006).

In other countries, stand-level simulators have been used longer, and some of them are still in use. The trend is still towards the use of single-tree models. For instance the AVVIRK and GAYA simulators in Norway is based on the development of an average tree (Eid & Hobbelstad 2000). However, single-tree based simulator T has recently been developed (Gobakken et al. 2008). In Sweden, the earlier versions of HUGIN simulator were based on stand-level models (Hägglund 1981), but the current version, as well as the new HEUREKA system, are based on single-tree models (Lämås & Eriksson 2003). In addition, there is a vast number of other simulators developed to aid decisions in forestry e.g. Monte (Palahi et al. 2004), SADfLOR (Borges et al. 2003), SAGALP (Chen et al. 2002, Seo et al. 2005), CAPSIS (de Coligny et al. 2004), FORRUS-S (Chumachenko et al. 2003), SGIS (Næsset 1997), DRYMOS (Chatziphilippidis et al. 2004), SILVA (Pretzsch et al. 2002), SIBYLA (Fabrika 2002, Fabrika & Ďurský 2004), DSD (Lexer et al. 2005), and CONES (Vacik et al. 2004).

1.3. Spatial aspects in forest planning

Timo Pekkonen

In forest management planning one is searching for an optimal management according to some criteria. A new plan is made every tenth year or so, but the measurements made for previous plans are not very well utilized in a new planning. The planner may use the earlier plans only as supporting information. One problem in utilization was earlier that the plans were only on paper. Nowadays when all data, measurements and maps are numerical and stored in databases, a problem is poor localization of the measurements. The field measurements are generalized quite subjectively to forest compartments which in turn are delineated subjectively on aerial photographs. Although one tries to delineate homogeneous compartments the resulting compartments can be rather heterogeneous. In addition, they can vary over time even if the delineation is carried out by the same planner.

At present the inventory methods are subject to changes. One of the most noticeable changes is the use of GPS devices. Using them a rather accurate position can be measured in the forest. The interpreted laser scanning data may be used to make the location even more accurate. With these techniques the field measurements can be located at points quite accurately. Also the GPS devices in harvesters make it possible to localize the treatments in the forests.

Generally, the spatial data used in a forest management planning can be grouped as follows:

- Topographical data, elevation contours, ditches, rocks etc. usually stored in vector format
- Remote sensed data, satellite images, laser scanners measurements and air photos stored in regular or irregular grids
- The treatments carried out in forests, as accurate compartments localized by harvesters (in future).
- Interpreted information from remote sensing, topographical or other auxiliary data, may be located as automatically segmented small, micro compartments
- Inventory measurements, which still today are mainly located in inventory compartments. In the near future GPS and laser scanning images can be used to get accurate co-ordinates.
- Treatments suggested by the forest planner, in the form of compartments delineated on an aerial photograph and checked in the field.

Remote sensed and interpreted data are special cases because they consist of a huge amount of measurements. In one square meter area there can be several pieces of aerial photograph pixels or laser measurements.

To use historical data sets requirements to the data storage too. One should have storage for a spatio-temporal data accumulating with years. This may not be a straightforward task. Rasinmäki (2003) has presented a conceptual model for storing of spatio-temporal data. His model is general enough to incorporate all types of data listed above. Rasinmäki has also applied his model to a situation where several compartment wise inventories were made in the same area and all data was stored in an object-relational database.

A basic task in forest management planning system is to simulate several development paths for the forest stands. Today the forest stands are handled quite independently of each other. However, an increasing amount of emphasis is put to the spatial relations between the stands, neighbourhood, length of the common borders etc. One possibility to use this kind of data is to calculate it once at the simulation start. However, if we let the computational units to change during the simulation, then the spatial relations too are subject to changes. Then one may need an on-line function which gives the spatial relations required during the simulation. SIMO GIS-component is aimed to be such a tool.

1.4. Selecting the optimal plan

Annika Kangas

Simplest way to make forest planning is to use simulation. It means a system that enables making if-then calculations of different cutting scenarios. Such models were, for instance, HUGIN in Sweden (Hägglund 1981) and AVVIRK in Norway (e.g. Eid and Hobbelstad 2000). With simulation, similar cutting rules are applied to all stands, possibly varying with respect to some stand-level classification.

Typically, however, forest planning problems are described so that each stand in the forest has several different treatment schedules that are possible alternatives for it (Fig. 1). For instance, harvests with two different rotation times produces two different schedules for one stand. Each schedule may include several treatments with a different timing. It may be that the schedule for one stand includes one or two thinnings before the final harvest, and planting after it. The development of the stand is then predicted under each of these schedules, based on forest simulators. With different combinations of standwise treatment schedules, a huge number of different production programs for the whole area could be obtained. Such forest planning problems have typically been solved with linear programming (e.g. Davis et al. 2001, Buongiorno & Gilles 2003). Forest planning packages based on linear programming were developed in many countries, for instance the FORPLAN model in USA (Johnson 1986, Johnson et al. 1986, see also Iverson and Alston 1986), MELA model in Finland (Siitonen 1996), and GAYA-JLP in Norway (Hoen & Solberg 1996).

In many cases the real problems are too complicated for these exact methods. Then, the problem either is simplified so that it can be solved with exact methods, or the solution is searched using heuristic methods (e.g. Davis et al. 2001, Pukkala 2002). These methods can produce a good solution with fairly simple calculations, but they cannot guarantee an optimal solution. The benefit in these methods is that the true decision problems can be described better than with exact methods.

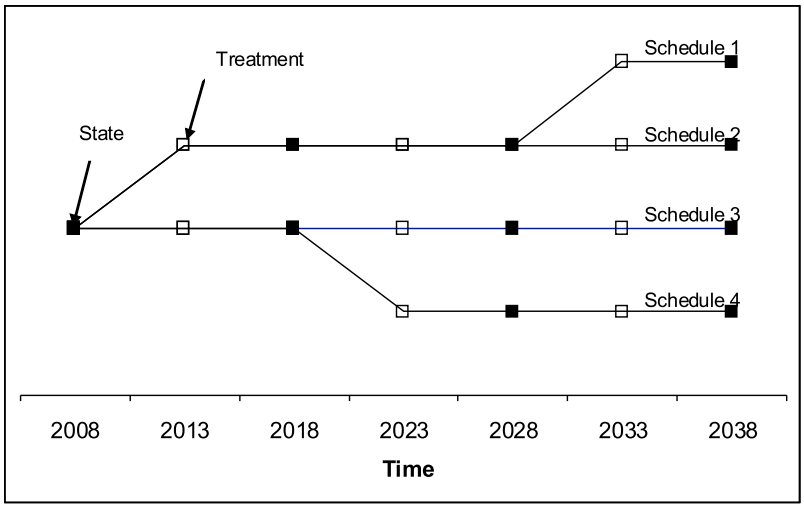

Figure 1. Principle of treatment schedules

1.5. Objectives of the SIMO project

Annika Kangas and Timo Tokola

The development of data collecting methods, growth and yield models and optimization methods, as well as the growing demands of users of forest planning packages has lead to a growing demand of more versatile, effective and customer-oriented planning systems and practices. Furthermore, the forestry databases and forest planning packages form the platform with which new ideas and methods concerning planning, or new models and methods for utilizing measured information as well as new data gathering methods, can be tested. Therefore, a system that can be modified according to the current needs is a prerequisite for research work in these areas. In previous decades in Finland, such a system was the MELA-system. However, as it nowadays is a commercial product maintained by Finnish Forest Research Institute, it no longer serves as a platform for testing new ideas in the same way.

The forestry databases and systems require a long time to develop. The principles are, however, similar everywhere. In the middle of 1990's a general data model for a forestry database was developed in the University of Joensuu (Tokola et al. 1997). The model aimed at improving the possibilities to update and extend the traditional relational databases. This system has served as a basis for natural resources databases in several countries, for example in Indonesia and Sambia. This method was build for data management problems and it is closely related to solutions for inventory problems. This work provided the basis for the SIMO project.

The aim of the SIMO project was to develop a new generation planning system. The basic properties aimed at are the following:

- ability to use different sources of information and in different levels (e.g. tree, pixel, plot, sub-stand, stand)
- utilization of data from different time points
- use of different combinations of models at different levels (e.g. tree, plot, stand) for predicting the development of forests
- localization of the models according to the information available
- search for optimal solutions to problems at different levels (e.g. stand, woodlot, area, nation)
- ability to restrict stand treatments in defined sub-areas (e.g. lekking areas of capercaillie or separate forest estates in larger area)
- spatial optimization problem solving
- possibility for users to adapt the system (e.g. parameters, models, basic data) according to their needs
- easy extendability of the system for future needs

The system was developed in the University of Helsinki and released as open source software. The results of this work can be used as such by researchers, teachers and commercial software companies, and each person willing can also be part of the developing team (see www.simo-project.org/contribute.html).

The project involved from the start all the major forest organizations carrying out forest planning in Finland, namely Tapio and Forestry Centers accounting for planning in private non-industrial forests, Metsähallitus accounting for planning in the public forests and UPM-Forest, Tornator and Metsämannut accounting for major part of forest planning in forests owned by industry.

The application-driven database design procedure is based on a needs assessment derived from interviews, a review of operations and an evaluation of source data. This approach assumes that the users have a clear idea of what they want. A forestry information system, which is based on this procedure, has a better chance of satisfying the users' needs. Furthermore, the implementation of the information system is easier to control, thus minimizing a risk for failure. Therefore, the project started with comprehensive interviews of several persons from each organization, whereby the needs of these organizations were analyzed (see Kangas et al. 2006, Wathén 2007).

The needs assessment was carried out by interviewing representatives responsible for forest planning in each organization one by one. The interviews were organized in three steps:

1. the target organizations were asked to define freely their own ideas about ideal system
2. researcher prepared list of requirements and each party was giving comments after studying prepared material
3. ready made Use Case-draft were available and target organizations gave comments and refined documents during interview

The list of required attributes and functionality were reported in the form of M.Sc. thesis (Wathén 2007). According to the study the stand-based system for managing and treating forests continues in the future. Because of variable data acquisition methods with different accuracy and sources, and development of single tree interpretation, more and more forest data is collected without field work. The benefits of using more specific forest data also calls for use of information units smaller than tree stand.

No major differences in parties' view of the systems requirements were noticed in this study. Rather the interviews completed the full picture from slightly different angles. In organizations the forest management is considered quite inflexible and it only draws the strategic lines. It does not yet have a role in operative activity, although the need and benefits of team level forest planning are admitted. Demands and opportunities of variable forest data, new planning goals and development of information technology are known.

In forest planning tools of GIS-based resource information systems are utilized. The system itself is an IT-system with specific functionality. After the initial step of user needs assessment the next step of this type of IT-project is the system design. The design will serve in building a prototype that can be used for learning, collecting experience and most importantly as a tool for working out a detailed system design of the forest information system. In this document the system that is the result of the research project is described in general level. More detailed documentation can be found from the homepage of SIMO; www.simo-project.org

2. SIMO – SIMulation and Optimization for forest management planning

Jussi Rasinmäki

SIMO is a forest management planning framework that enables the user to build different forest growth and yield simulators, couple those with optimization methods, and apply the combination to diverse planning problems.

2.1. Data model

In SIMO a very flexible data model is used which could be described as:

> "The forest consists of some objects that have some kind of attributes. These objects can have sub objects that again have some attributes. Again, these sub objects can have sub objects of their own, and once again these sub objects have their own attributes, etc.".

Thus, the data model of SIMO has two main features: (i) forest is seen as a hierarchical collection of objects, and (ii) attributes of the objects are not fixed at the data model level. At the implementation level, the hierarchical data model is expressed as a rooted tree graph, i.e. a simple, undirected, connected, acyclic graph with a special root node (Fig. 2). Further, all the nodes in the graph are associated with a data object level, and there can be any number of data levels in any particular simulation; ; e.g., the stand-stratum-tree and the stand-microstand-stratum-tree data level divisions in Fig. 2 a) and b) are two expressions of the same data model.

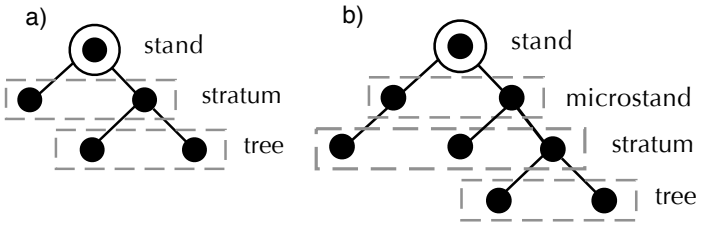

Figure 2. Illustration of the rooted tree graph data model used in SIMO. In both cases a stand is the root node.

Usually in the simulation several alternative development scenarios for a simulation unit like stand are generated. After the simulation, optimization is then used to select the best scenario among the alternatives. The data model accommodates developing scenarios as branches (Fig. 3).

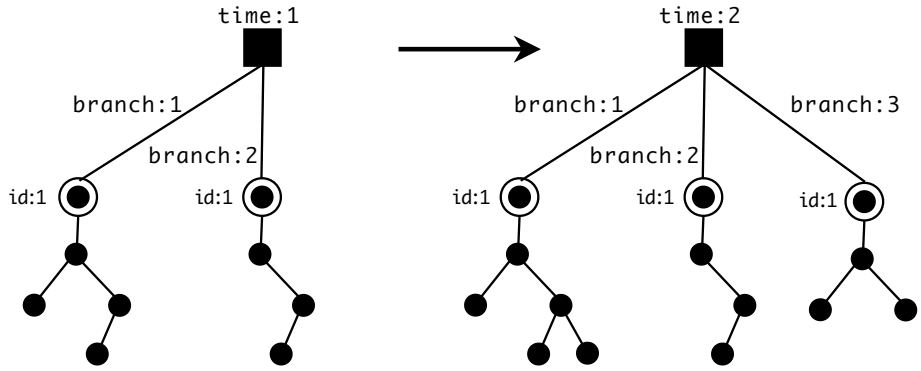

Figure 3. Handling alternative scenarios in the data model. The alternative states for a simulation unit are stored in branches for each time step in the simulation.

As stated above not only can one freely define the kinds of objects in the simulation, also the object properties are not fixed in the data model. This is achieved through expressing the object attributes as value–value interpretation-pairs; i.e., each attribute value is paired with the interpretation of the value (see Tokola et al. 1997 for an earlier implementation).

However, when used for any individual simulation in SIMO, the contents of the data model is fixed for that computation. This is done via a lexicon, which defines the data object hierarchy; i.e., sets the names for different data levels and their relationships, and defines the attributes that the object at each data level can have. In practice this is done with an XML document, which has the same kind of hierarchical structure as the rooted tree graph of the data model (Fig 4.).

```
<rootlevel>
        <name>simulation</name>
        <num_vars>…</num_vars>
        <cat_vars>…</cat_vars>
        <sublevels>
                <sublevel>
                        <name>stand</name>
                        <type>static</type>
                        <num_vars>…</num_vars>
                        <cat_vars>…</cat_vars>
                        <sublevels>
                                <sublevel>
                                        <name>stratum</name>
                                        <type>dynamic</type>
                                        <num_vars>…</num_vars>
                                        <cat_vars>…</cat_vars>
                                        <sublevels>
                                                <sublevel>
                                                        <name>tree</name>
                                                        <type>dynamic</type>
                                                        <num_vars>…</num_vars>
                                                        <cat_vars>…</cat_vars>
                                                </sublevel>
                                        </sublevels>
                                </sublevel>
                        </sublevels>
                </sublevel>
        </sublevels>
</rootlevel>
```

Figure 4. XML-description of the data model for a simulation; data levels are given their names and their positions in the hierarchy. In this case the simulation consists of stands, which consist of strata, which have trees.

The attributes at each data level are divided into two categories: numerical (num_vars-tag) and categorical (cat_vars-tag) attributes (Fig. 5). In the lexicon any number of attributes for each data level can be defined together with optional minimum and maximum values. Should the actual data values in the simulation data lie outside these limits, a warning is generated during the simulation. For categorical attributes, all the possible values are enumerated in the lexicon together with the interpretation of the value, while for the numerical attributes the unit of measurement is given.

```
<num_vars>
  <variable>
    <name>ALT</name>
    <unit>m</unit>
    <min_value>0</min_value>
    <max_value>600</max_value>
    <description>Altitude above see level</description>
  </variable>
  …
</num_vars>
<cat_vars>
  <variable>
    <name>MAIN_GROUP</name>
    <min_value>1</min_value>
    <max_value>8</max_value>
    <description>Land use</description>
    <values>
      <enum>
        <value>1</value>
        <description>Forest land</description>
      </enum>
      <enum>
        <value>2</value>
        <description>Scrub land, low productive land<</description>
      </enum>
      …
    </values>
  </variable>
</cat_vars>
```

Figure 5. XML-description of of the data model for a simulation; numerical and categorical variables for each level are defined. The unit of measurement is set for the numerical attributes, and the categorical attribute values are enumerated.

2.2. What is it made of? – the components of SIMO

SIMO consists of four main modules: data import, simulation, optimization and reporting, which are accessed using a command line interface (Fig. 6)

The components have been programmed with Python programming language, and the source code is available through http://trac.simo-project.org/browser. The main components implement a generic simulation and optimization framework based on the hierarchical data model; i.e., no forestry specific information is hard-coded into the framework components. All components use a key-value-based database (Oracle Berkeley DB) for permanent storage of serialized Python objects.

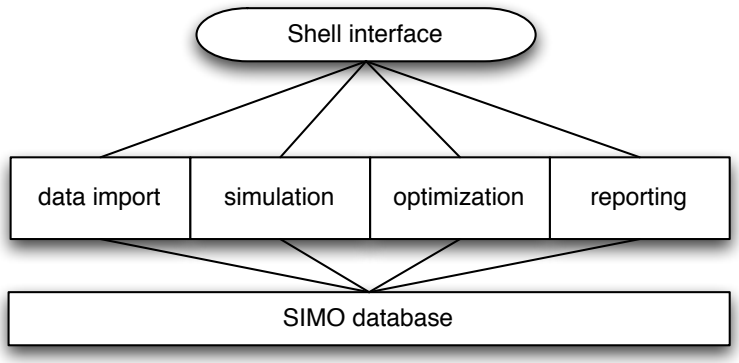

Figure 6. The main components of SIMO

In an individual computation, the framework is modified using XML files that describe the precise content of the data model, and the simulation and optimization tasks for the computation. In addition, the programmatic implementations of the prediction and operation models are needed. These are collected into model libraries, implemented as shared function libraries (dll-files in Windows, so-files in Linux/Unix). Fig. 7 has a schematic presentation of framework usage for a computation utilizing all the main modules. In the subsequent chapters a more detailed explanation is given of the different phases in the computation.

The ini-file seen in Fig. 7 is used to set a host of parameter values and file locations when starting SIMO from the shell interface. A detailed walk through of the ini file and all XML document content is in the SIMO manual available from
http://www.simo-project.org/documentation/SIMOmanual.pdf.

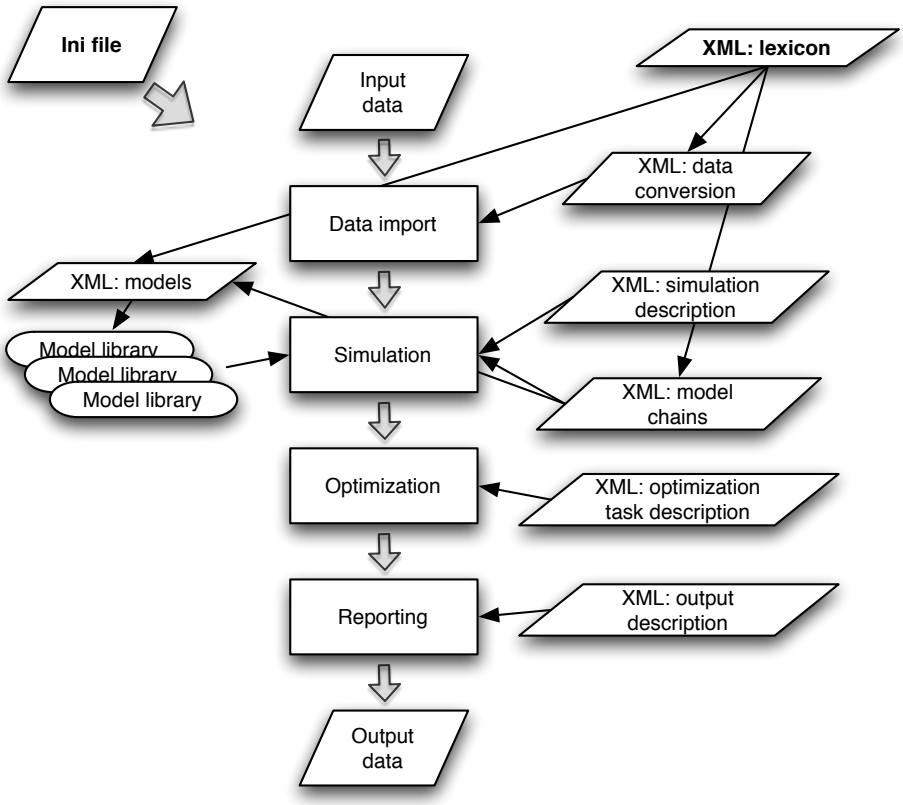

Figure 7. SIMO program flow in a case utilizing all main modules.

3. Simulator

Jussi Rasinmäki, Jouni Kalliovirta and Antti Mäkinen

3.1. Data import

The basis for data import, as well as all the other modules, lies in the lexicon defining data object levels and their attributes. Based on the lexicon a mapping from the incoming data to the internal data model is given (Fig. 8): incoming variable to internal variable mapping, unit conversions for numerical values and incoming value to internal value mappings for categorical attribute values.

Currently SIMO can import data from text files that can be of two different types. In an inlined format the data from all data levels are combined into a single text file and each row has an identifier indicating the data level it contains data for. Object relationships are indicated by relative positions of the rows; e.g., the stand row comes first, below it are all strata rows for the stand. In the "by level" format the data for each data level is in a separate text file and the relationships between objects in different data levels are indicated with object identifiers; i.e., primary and foreign keys in database terminology.

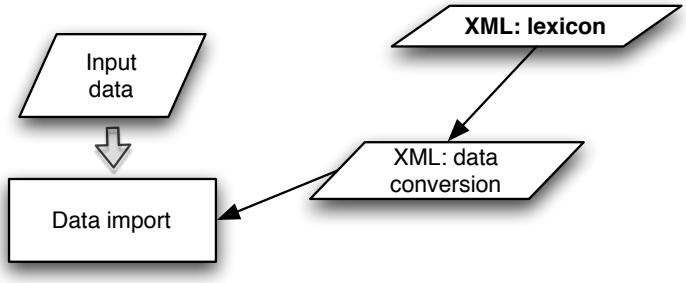

Figure 8. Data import is based on a data conversion mapping that builds the connection between the incoming data format and the lexicon used in the simulation.

3.2. Simulation

SIMO simulations hinge on four main concepts: lexicon, model chains, model libraries, and simulation description (Fig. 9).

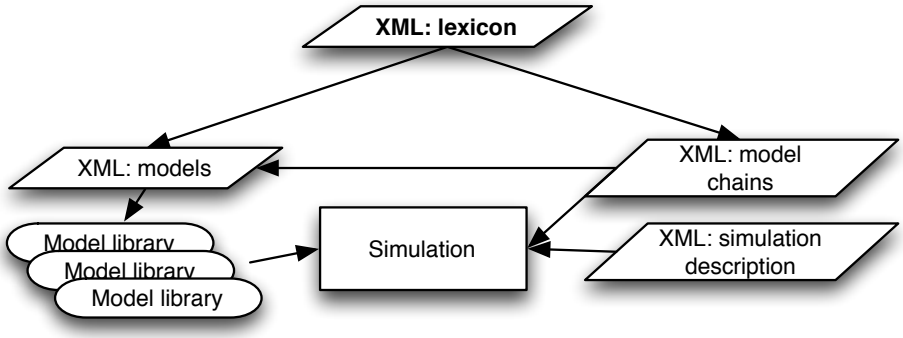

Figure 9. The four main components of SIMO simulations: lexicon, model chains, model libraries, and simulation description.

3.2.1. Model libraries

Model libraries are collections of individual model implementations. Models have two roles within the framework being used in describing the natural processes in the forest, e.g., growth and mortality (prediction models), and the human interventions, i.e., forestry operations like harvesting (operation models). Each implementation consists of two components: the programmatic implementation of the model, and an XML definition that binds the model to the lexicon. The programmatic implementation is based on a standardized function interface that each model implementation must comply with, thus ensuring standardized communication between the simulator core and the models. There are two interfaces for the models, one for the models used solely to predict properties of objects, and one for the models used to modify the existence of data objects in addition to value prediction. Currently the simulation core supports libraries written in Python and C. The C models are implemented as shared libraries that expose the models as functions. Fortran is a candidate for an additional model implementation language. In addition to the models used in prediction, aggregation and conversion models have also been defined

The XML definition binds the programmatic implementation to the framework by listing the model variables using the framework lexicon; the data level the model variable belongs to, and the attribute it corresponds to. If optional lower and upper limits of the variable are declared, they are used to generate a warning during the simulation that the known limitations for the model have been exceeded.

The XML definition also serves as documentation of the model by listing the model name, author information, general description of the model, the original publication of the model, the species it is applicable to, geographical coverage, and other conditions for application, and description of the research material used to define the model. This information is collected from the original research publication at the time the model is implemented in the function library.

3.2.2. Model chains

Once the model implementations exist, a way to actually utilize them in simulations is needed. This is when the concept of model chain enters the picture. A model chain is a way to decompose a simulation computation into parts, and to describe the simulation as a collection of these parts.

The role of the model chain is to describe how the models from the model libraries are applied to the data. The basic unit of a model chain is a task that consists of an optional condition and a model. Conditions are logical expressions which are evaluated in the simulation process using data values. If the condition is satisfied, the model attached to the task element is executed. The simulation is a collection of these conditional model executions. However, to be able to describe complex simulations, the basic task structure can be extended by dividing a task into sub-tasks, each of which may again be conditional. Thus the simulation is described at the top level as a set of tasks that have an execution order. Each of the top level tasks is described by a tree of tasks where the traversal of each branch and the model execution at the leaf at the tip of the branch is conditional.

In conceptual form, a simulation can be described as a graph in with the rectangular nodes represent a condition–model combinations and the edges the sequential flow of the simulation. This graph form translates directly into the model chain definitions of the simulation (Fig. 10).

A model chain has an evaluation level in the data entity hierarchy. In a simulation, the tasks in the model chain are carried out for each of the data objects from the evaluation level in the data set. As each data object knows its parentage and descendants; the variables in the models and task conditions are not restricted to the model chain evaluation level. Each data object in the hierarchy knows its descendants to the finest scale units in the hierarchy and parentage up to the user chosen simulation level, at which the data objects are processed one at a time. Therefore, the data objects in the hierarchy are independent from the simulation level up; e.g. no data can be exchanged between stands if the stand is defined as the simulation data level.

A simulation usually processes data at several levels, e.g., predicts the growth of individual trees and then aggregates the results to stratum and stand level. For this purpose, a simulation in the framework consists of a set of model chains which have an execution order to guarantee execution in the correct sequence. Moreover, there are two distinct sets of model chains: initialization and loop chains. Initialization chains are only executed once at the beginning of the simulation. Their role is to augment the simulation data by adding attributes or data objects that are missing in the original data; e.g., trees need to be generated in a simulation based on individual tree growth models using stand level data. Loop chains, used to describe the actual simulation, are executed once for each time period in the simulation.

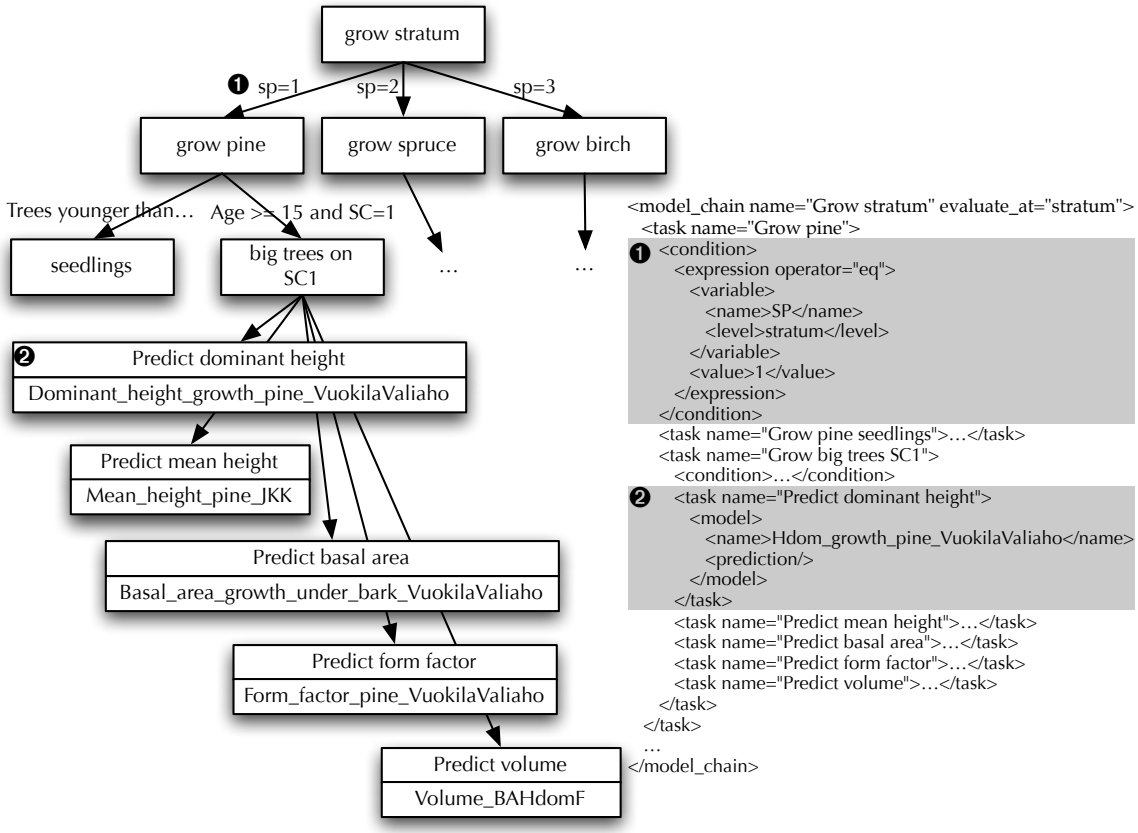

Figure 10. Model chain as XML and a corresponding graph showing the model chain structure of tasks (rectangles), conditions (labels on links between rectangles) and model calls (two-part rectangles).

3.2.3. Simulation description

The simulation XML file gathers together all the components that define a simulation: model chains, simulation span definitions, stopping conditions, simulation parameter and variable values (Fig. 11).

The simulation span and model chain definitions are interrelated. It is possible to define several time spans for a single simulation with different time steps and different sets of model chains to be applied. In addition to the time span definitions, the length of the simulation can be controlled by a stop condition that can force the simulation to stop prior to the last time step.

The simulation span time step definition has relevance to the model definitions as the time period for model predictions varies. The time period is therefore included in the model definition. The simulator core correspondingly has a memory for the predictions given by models. If the same model result is applicable for several subsequent simulation steps, the result is taken from the model memory instead of a new prediction. An example would be a yearly simulation and a growth model that gives the average yearly growth for the next five years. The model result would be computed only every fifth period in the simulation.

```
<simulation>
 <control>
  <growth_season_end_date>-07-15</growth_season_end_date>
  <span>                                                          ❶
   <time>
    <timeStep>1</timeStep>
    <timeUnit>year</timeUnit>
    <steps>20</steps>
   </time>
   <initChains>
    <chain>Chain_create_missing_variables.xml</chain>
   </initChains>
   <simulationChains>
    <chain>Chain_grow_stratum.xml</chain>
    <chain>Chain_generate_trees.xml</chain>
    <chain>Chain_generate_tree_attributes.xml</chain>
   </simulationChains>
   <operationChains>
    <chain>Chain_operations_case_2.xml</chain>
   </operationChains>
  </span>
 </control>
 <stop_logic>                                                     ❷
  <expression operator="eq">
   <variable>
    <name>REGENERABLE</name>
    <level>stand</level>
   </variable>
   <value>1</value>
  </expression>
 </stop_logic>
 <parameters>                                                     ❸
  <parameter>
   <variable>simulationUnit</variable>
   <value>stand</value>
  </parameter>
  …
 </parameters>
 <init_variables>                                                 ❹
  <variable>
   <name>REGENERABLE</name>
   <level>comp_unit</level>
   <value>0</value>
  </variable>
  …
 </init_variables>
</simulation>
```

Figure 11. A simulation description: (1) 20 year simulation with a one year time step, (2) the simulation is stopped either after 20 years or when the stand level variable REGENERABLE has the value 1. Besides the time, model chains and stop logic, also parameter (3) and variable (4) values are set.

3.3. Simulator Testing

So far two different types of forest growth and yield simulators have been built on SIMO platform; a tree-level growth simulator and a stand-level growth simulator. The two simulators use different growth models for predicting the forest development in time, but share a number of common models, such as implementations of different forestry operations. As the two simulators use different models for growth prediction, it is probable that there will be some differences in the growth estimates predicted with the two simulators. Both of the simulators were tested and validated with a number of test simulations during the development process. Forest growth models cannot be expected to provide perfect growth predictions but by testing and analyzing the simulators, different sources of errors can be eliminated and accuracy and reliability of the simulators increased.

The validity of the two simulator implementations and the possible differences in the growth predictions were analyzed by simulating a number of sample plots with measured growths. The growth predictions by the two simulators were also compared to the growth predictions produced by Motti simulator, which is a well tested stand-level forest simulator (Hynynen et al. 2005, Salminen et al. 2005). The sample plot data set for the analysis was the same as used by Välimäki (2006) for validating the behaviour of growth models in overstocked, i.e.. denser than average, forest stands. The data set included 60 tree-level sample plots from 30 stands in central Finland and the growth of the trees in the sample plots was known for the past 20 years. Although the data set was quite small, different site classes and age classes were well represented. The growth for the sample plots was predicted for a 20 year period with both tree-level and stand-level simulators and also with Motti simulator. As the actual growth for the trees in the sample plots was known, the prediction errors in different points in time for all three simulators could be calculated.

The results by Mäkinen et al. (2008) showed that both the tree-level and stand-level simulator implementation on SIMO platform produce at least equally good estimates about forest growth compared to Motti simulator. Thus, although the models used are in principle the same, there are small deviations due to the different implementation. Table 1 shows the relative mean prediction errors at different points in time for the analyzed stand level variables: mean height (H_{mean}), mean diameter (D_{mean}), basal area per hectare (BA) and number of stems per hectare (N). Most of the variables were underestimated by all three simulators, which is largely due to the high density of the stands. The prediction errors and the standard deviations between the three different simulators varied for each of the variables, but the overall scale of the errors was the same for all of the simulators.

Table 1. The test results

Simulation length (years)		Hmean [%]			Dmean [%]			BA [%]			N [%]		
		motti	simo tree	simo stand	motti	simo tree	simo stand	motti	simo tree	simo stand	motti	simo tree	simo stand
5	mean	-3,2	-4,5	-4,0	0,1	-2,3	-0,2	-4,8	-4,6	0,4	-9,0	-6,1	-3,0
	stdev	5,9	6,0	8,9	4,6	4,4	3,4	14,1	11,1	19,4	11,8	7,5	7,1
10	mean	-5,3	-6,2	-4,9	-0,8	-2,3	-0,3	-9,5	-6,3	-3,8	-12,6	-9,6	-3,9
	stdev	8,5	8,6	11,2	5,3	5,2	4,4	15,5	14,0	13,3	13,6	10,4	9,6
15	mean	-6,1	-6,7	-5,2	-0,1	-1,6	0,0	-8,7	-4,3	-3,4	-12,2	-8,7	-0,3
	stdev	10,1	10,3	12,9	6,5	5,4	4,9	17,4	16,4	12,0	15,9	13,7	12,8
20	mean	-6,4	-6,6	-5,2	1,2	-0,6	0,3	-5,3	-0,3	-0,2	-10,3	-6,1	-6,3
	stdev	11,4	11,6	14,3	9,6	5,5	5,2	19,4	17,9	12,8	16,7	14,3	17,8

4. GIS in SIMO

Timo Pekkonen

The SIMO GIS component is a tool which is used by the forest simulator for determining the spatial information and topological relations of the forests. The component is functioning as a link between the application and the data storage i.e it is an interface to the spatio-temporal data.
An implementation of a GIS component depends on how and where it is used i.e.
- efficiency requirements and
- which kind of data source, data base system it is used with.

If the requirements set for effectiveness are great then the component may need its own cache memory for storing the data. If in the data source there are all geographical methods available and very effective operations are not required, then the GIS component can be a simple filter between the simulator and the data source. A useful way to consider a GIS component is to keep it as a function or method to retrieve spatial data.

4.1. Implementation principle

A GIS component can be utilized locally in space and time. This means that every time when the component is used a restricted geographical data, e.g. a couple of stands, is given as input to it. Another alternative is that a whole forest area for planning constitutes the input to the GIS component. During the simulation the geographical operations are made and requests fulfilled. As snapshots the geographical data corresponding a special simulation phase can be retrieved from the component.
Implementing of the SIMO GIS component was based on the following two points:

1. The component is used as an on-line tool, as effective as possible
2. The source of the geographical data is not restricted. For example the source data may or may not contain topological relations.

According to the point 2 the polygons corresponding to forest compartments were selected to be the source data for GIS component (Fig. 12). Later the input data can be extended to contain other types of spatial data too.

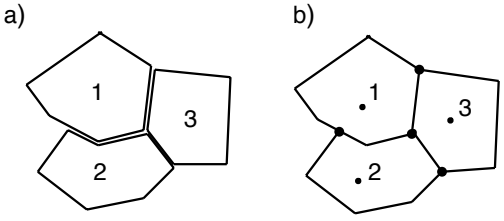

Figure 12. (a) Source data and (b) the corresponding internal data structure, the topological model.

4.2. The topological model

A topological model for a spatial data is a data structure where one can determine the topological relations between the objects in data. The SIMO GIS component uses a rather simple data structure for the topological model. The structure contains line segments and points. The line segments are starting from a junction point, called node, and ending to another node (Fig 13). The segments are linked to each other at the nodes.

In Figure 13 there are four line segments L1, L2, L3 and L4 which are linked to each other at the nodes N1 and N2. The line segments delineate three polygons which are referred by the reference points (black dots) R1, R2 and R3. In parenthesis there are two flags, called level flags or level code, which define the relation between the line segments and the polygons. A line segment is a border for a polygon if it and the corresponding reference points have common flags equalling to 1. It is said that the polygon and the line are on a same level. So the first polygon referenced by point R1 is bounded by the border lines L1 and L2 the polygon reference by R2 is bounded by the lines L1 and L4. The polygon referenced by the point R3 is bounded by the segments L2 and L3.

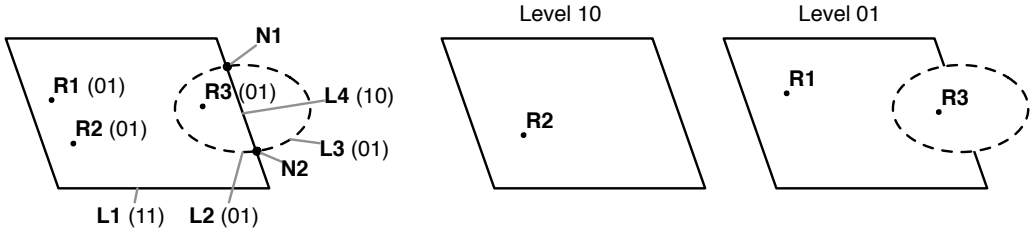

Figure 13. Use of the topological model in the GIS component. A sample map consists of four lines L1, L2, L3 and L4 with junction points in the nodes N1 and N2. The three reference points R1, R2 and R3 defines three polygons according to the level codes (shown in parenthesis). The reference point R1 defines the polygon bounded by the border lines L1 and L2, the point R2 defines the polygon bounded by the lines L1 and L4 and the point R3 the polygon bounded by L2 and L3 (see text). Polygons constructed from the topological model at the two levels illus-

All data structures in the GIS component are linked lists. A topological model is represented by two types of objects: Ordinary objects, like polygons and lines, and the topological objects. The latter objects represent the topology between the ordinary objects. The polygons are represented by reference points which are points inside the polygons.
The objects are presented as object and co-ordinate records (Fig. 14).

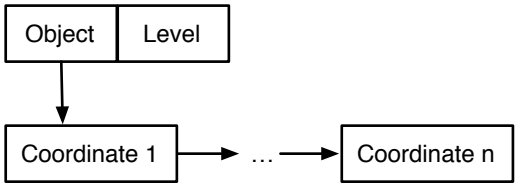

Figure 14. Representation of objects

The object record contains object type, the level flags, identifier etc. Only the level flags, or level codes, are explicitly presented in the Figure 14. The geometry of an object is defined by one or more co-ordinates. They are stored in co-ordinate records which are linked as an ordered list. The object record has a link to the first co-ordinate record.

The topology is presented with three types of records (see Fig. 15): nodes, borders and so called tails. By a tail is meant here either the beginning or the end of a border line segment. As indicated in the Figure 15 the border records are linked with an object record i.e. the corresponding line record with links in both directions. Each border has a start and an end tail. All tails leaving a node are stored in an ordered circular list. The tails are ordered according to the direction they are leaving the node. The node has a link to the first tail in the list and all tails have a link to the node.

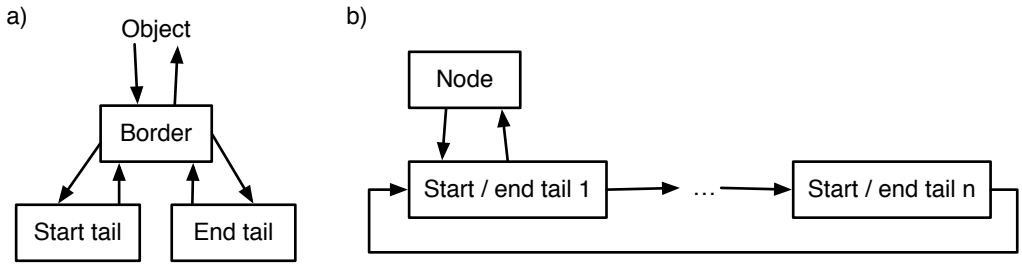

Figure 15. Linking of tail records to (a) the borders and (b) the nodes. The tail records in (a) and (b) refer to the same data structure.

The node records can be considered as key records in the topological data structure. As seen in the polygon search algorithm (4.2.3.2) the tail lists linked to the nodes have a central role when traversing the polygon rings. This kind of node-oriented linking is especially suitable when level codes are used to store coverage of several polygons in the same topological model.

Use of implicit topology described above can also be used in geographical data bases. Storing temporary history of the objects are quite straight forward easier than in the case when a full topology is stored and updated in the database. An attempt to this direction is (Ohsawa et al. 2002) which uses the same kind of implicit topology presented above.

4.3. Basic algorithms

In the following sections some basic algorithms implemented in GIS component are described. The second algorithm (section 4.3.2) demonstrates the use of the level codes to separate the hierarchical polygons from each other.

4.3.1. Creating a topological model from the polygons

When a topological model is created from polygons it is assumed that

1. the polygons are sound i.e. polygon rings do not touch or intersect each other or themselves
2. there are no polygons overlapping each other and if two polygons touch each other the common border segments have exactly the same co-ordinates.

The creation of a topological model is done in two stages. First the reference points corresponding to the source polygons are constructed. A reference point is found out on the horizontal line which is in the middle of the east-north direction. The constructed points are inserted to the model structure.

In the second phase the common border segments are filtered. Here all line segments are processed and one of the duplicate line segments is deleted.

The filtration of the line segments is made by the following steps:

1. Tabulate all simple, two point line segments, i.e. from point to point segments, of all polygon borders.
2. Tabulate the co-ordinates of all line segments. Sort the co-ordinates and filter out the duplicate co-ordinates

Next a linked data structure, a net structure, is used to filter out the duplicate line segments. In the net structure there are edges, which are two point line segments, and vertices, junction points of two or more edges. The edges are linked to the vertex points.

3. Create a net with vertices at the filtered co-ordinates (cf. step 2) and the line segments as edges. Link the edges to the vertices. During linking the duplicate line segments are easily detected and filtered out.

Next node-to-node lines are extracted from the net.

4. Process all edges in the net as follows:

   If the edge is not processed earlier then expand the edge to a line starting and ending to a vertex where more than two edges are leaving the vertex. If succeeded then the start and end point of the line are added as nodes (if not added earlier) to the topological model. If not succeeded, the segment expands to a ring, then an arbitrary ring point is added as a node to the model. Link the expanded line to the nodes in the topological model.

When all polylines are constructed and added to the topological model then the temporary net structure can be released.

4.3.2. Find a pointed polygon

The next algorithm finds out a polygon which is on the given level and has a given point, the search point, inside it:

1. Find the nearest line segment to the west from the search point which is on the given level i.e. there are at least one level flag being 1 both in the given level code and level code of the line.
2. Determine on which side of the line the search point is and according to it determine a border tail, the start tail, so that it would be the start tail of the border when traversing the polygon ring in clockwise manner.
3. Traverse the polygon ring and gather the co-ordinates of border points in the traversal order

   A polygon ring is traversed in a clockwise order, border by border as follows:
   a. Save the start border and tail for the checking the end of traversal.
   b. Store the co-ordinates of the border from the start tail to the other tail
   c. Take the node at the end tail of the border and find out the tail linked to the node which is on the right level and is the next to the right from the end tail.
   d. If the next tail is the original start tail (saved in step 1) then the ring is complete; otherwise change the start tail and the border and return to the step 3.2.

In the GIS component the area of a ring is defined to be positive if the ring points are processed in clockwise order; if processed in counter clockwise order the area is negative.

4. Calculate the area of the ring. If the area is positive then the outer ring of the polygon is found; otherwise the ring belongs to some inner ring, a hole. In that case update the search point to the west most point of the ring and return to the step 1.

Now it remains to find out all wholes, inner rings, inside the polygon and a unique reference point to the polygon

5. Search for all border lines inside the outer ring and construct all inner rings i.e. with a negative area and which are not inside some other inner ring.
6. Find out all reference points inside the outer ring and select the reference point which is not inside any inner ring. The selected reference point corresponds the polygon searched for.

A spatial index can be used in steps 1, 5 an 6 to speed up the search process. Some experiments were made using the Mg R-tree library of Pavlata (Pavlata 2004) constructing a spatial index into cpu-memory. The use of a spatial index was more effective than using a search over minimum rectangles of the objects when the source data consists of more than about thousand polygons.

4.3.3. Determining the polygon neighbours

The determination of neighbouring polygons are made simply by storing the identifiers of the left- and right side-polygon of a border line to the border structure. A temporal data structure, ring, is constructed for the neighbourhood search as helping aid (Fig. 16).

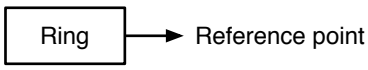

Figure 16. The ring data structure used in neighbourhood search.

The ring structure contains the area and the co-ordinates of the west most point of the ring and a link to the corresponding reference point i.e. to the polygon.

1. Construct all rings in the topological structure as follows:

Process all borders in the topological model and for both sides of the border do the following steps

   a. If the ring of the corresponding side is already created don't do anything; otherwise construct a ring by traversing the borders of the corresponding polygon. Store the area and the west most point to the ring and set the reference point link initially to null.
   b. Traverse once more the borders and store the ring to each border as a left- or right-side ring depending on the traversal order.

Next the outer rings of all polygons are completed i.e. a link to the corresponding reference point is stored in the outer borderlines of the polygons.

2. Process all reference points in the topological model and find out a border segment of the corresponding polygon in the way described in the previous section. Note that the areas of the rings are positive in the case of an outer ring and negative in the case of an inner ring. Save a link to the reference point to the corresponding ring structure.

Finally the reference point links are saved to the inner polygon rings.

3. Process all borders in the topological model. If on either side of the polygon there is ring where the area is negative, indicates an inner ring, and the reference point link is null, then get the west most point of the ring and using it as a search point find out the outer ring of the polygon and the corresponding reference point. Save the reference point link to the inner ring.

In the GIS component there is a function which can be used to get the neighbours of a given polygon. It processes all border lines and gathers the neighbours from the ring links of the borders.

4.4. Future development

The development of the GIS component and inclusion of new operations depends on the future use of the component. If the input sources are polygons like presently and if the geometrical operations needed are not of very special kind then some more standard approach like GEOS[1] (Geometry Engine – Open Source) may be used instead of the current function library.

Use of source data coming from database containing a complete topology is straight forward. In that case all lines are going from a node to another and they can be added directly to the topological model. If in the database there are stored no points, which are inside the polygons and can be used as reference points, then they should be constructed. This can be done either in the database or in the GIS-component. In the latter case the reference points are quite easily constructed using the outer rings of the polygons.

[1] http://trac.osgeo.org/geos/

5. Optimization

Antti Mäkinen & Jussi Rasinmäki

Decision making in forest planning utilizes optimization methods for finding the optimal harvest schedules that maximize the benefit of the forest owner. A common way of quantifying the benefit is by calculating the Net Present Values (NPV) of different operations. However, the actual management problem is not that simple in many cases. A number of restrictions limit the group of possible operations. Besides the economic values, such as NPV, different ecological and sociological values may guide the decision making and make the optimization problems complicated and difficult to solve. In many cases the objectives in the decision problem can be spatial, such as aggregating key habitats or harvest areas. The spatiality in an optimization problem can cause the problem to be non-linear which makes it impossible to solve the problem with linear programming.

As the optimization problems in forestry can be of various types and require different techniques for solving them, a number of optimization methods have been implemented in SIMO. For linear programming, SIMO includes an interface for J (former version JLP) linear programming library (Lappi 1992), which is an efficient tool for solving linear programming problems and which has been developed specifically for forestry applications. For solving non-linear and complex optimization problems, two different heuristic algorithms have been implemented in SIMO: HERO (Pukkala & Kangas 1993) and TabuSearch.

5.1. Optimization problem definition

Defining an optimization problem in SIMO happens through modifying an XML document (Fig. 17).

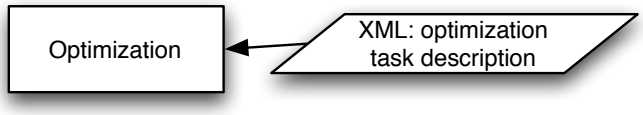

Figure 17. Optimization is controlled with an XML document describing the optimization method used, optimization goal and constraints.

A simple syntax let's the user define the type of the optimization task, i.e. minimize or maximize, the objective function and the constraints. The same optimization problem definition can be used for both linear programming problems and the heuristic problems with some limitations. The objective function definitions are more flexible for the heuristic methods as the utility function does not need to be linear. As there usually is more than one objective in the optimization problem, the objective function or utility function can be divided into sub-utility functions with user-defined weights. The number of constraints is unlimited and constraints are handled as strict constraints that limit the feasible solution space.

5.2. LP optimization

Linear programming optimization in SIMO has been implemented by building an interface to the J-program (http://www.metla.fi/products/J/) developed by Juha Lappi in the Finnish Forest Research Institute. LP optimization is controlled using the same optimization task description document as for other optimization methods. Naturally, only optimization goals and constraints allowed in LP-problems can be used when using J to solve the task.

5.3. Heuristic optimization

Heuristic search and optimization algorithms cover a number of different techniques for solving complex optimization problems which are not necessarily solvable with traditional optimization techniques. The term heuristic can be understood as a technique that seeks good solutions at a reasonable computational cost without being able to guarantee either feasibility or optimality (Reeves & Beasley 1993). Heuristic techniques cannot guarantee optimality as the search process does not always find the global optimum. Heuristic techniques cannot even determine how close to the global optimum a given solution is. The heuristic techniques are well suited for complex problems, such as in spatial optimization, where exact optimization techniques cannot necessarily be utilized. The term metaheuristic means the different smart algorithms that limit the search space and avoid the local optimums where the search algorithm can get stuck in.

A common type of heuristic optimization techniques is the local search, which is commonly used to solve combinatorial optimization problems. In combinatorial optimization, the optimization algorithm tries to find the optimal solution in a discrete solution space. One way of finding the optimal solution is to go through the whole solutions space by calculating the objective function value for every possible solution. This is however a poor strategy as the solution space grows exponentially as the optimization problem gets larger. Heuristic algorithms use different strategies for seeking the optimal solution by searching only a small portion of the total solution space. Local search methods search the neighbourhood of the current solution and try to find a better solution by making moves (i.e. small changes) in the current solution. In many techniques the optimization algorithm accepts only moves that will result in better solution, e.g. moves that will increase the objective function value in maximizing problems. This can cause the algorithm to find only a local optimum, not the global optimum, which is the target of the search process. This can be avoided with different strategies that different metaheuristic algorithm utilize. Common metaheuristic algorithms include simulated annealing, threshold accepting, tabu search and genetic algorithm.

Defining an optimization task for a heuristic algorithm is fairly simple. The objective or utility function can be divided into sub-utilities with independent weights. The sub-utility functions can be of different types, e.g. linear or non-linear. The common utility function types are additive, conjunctive and distance function. The restrictions are applied as strict restrictions. A typical forest planning problem can be formulated as a utility maximization problem as in the example below:

Maximize:

$$U = \sum_{i=1}^{I} a_i u_i(q_i)$$

s.t.

$$q_i = Q_i(x), \; i = 1,\ldots, I$$

$$\sum_{j=i}^{N_n} x_{jn} = 1, \; n = 1,\ldots, N$$

$$x_{jn} = \{0,1\}$$

In the optimization problem above the total utility U is calculated with an additive utility function, where I is the number of management objectives, a_i is the weight for the objective i, and q_i is the value of the objective i, calculated with the operator Q_i with a binary variable vector x. The items in vector x (x_{jn}) indicate if the unit n is treated according to management scenario j. The number of alternative management scenarios for unit n is denoted by N_n. The conjunctive and distance utility functions differ from the additive utility function only in the calculation of the total utility.

6. Reporting

Jussi Rasinmäki

There several report formats available in SIMO (Table 2).

Table 2. The report formats available in SIMO.

Format name	Contents
xml, ascii, table	simulation data from different data levels and all simulation branches
table-csv	simulation data for a single data level from all simulation branches
smt	simulation data for a single data level from the first simulation branch (usually the optimization result) for the last year of the simulation
branching_graph	text file that can be converted into a figure using dot program (http://www.graphviz.org/). One file per simulation unit, describes the branching of the simulation to alternative scenarios for the simulation unit. Lists the operations causing the branching.
operation_result	text file containing the operation result variables; e.g. cut timber assortment volumes, cash flow
chart	chart figures of the development of attributes over time
aggregation	aggregations; e.g. sum, mean; for attributes computed over user defined time periods and possibly over other classifying variables. The report can be obtained both as text file and as image files.

Again, the actual content for the different report formats is modified using XML documents (Fig. 18).

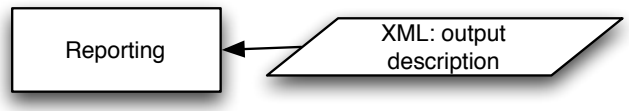

Figure 18. Reporting is controlled with an XML document listing the variables to include in the report from each level in the data hierarchy.

7. Final remarks

Timo Tokola

The SIMO project was completed at the end of 2007. The objectives of the project and requirements set by the users were very challenging to the system design. It was obvious from the beginning that there is no single database structure and functionality for this type of system. The construction of a functional and operative forest inventory system puts an emphasis on a thorough design of the system. The initial and most important phase of the system design is the construction of a data model, which is used to perceive, organize and describe data in a conceptual schema. The basis for that was mostly laid during the Ph.D. study of Dr. Rasinmäki, resulting in a rather flexible design, serving as a framework for system development. The entire system is set of software development tools which can be applied to various types of forest inventories and planning tasks. However, its use can be extended to the management of other natural resources as well. Still, when operational application is planned to be based on SIMO, there is further need to plan technical solution. It is also possible to use totally different database structure and utilize only specific part of system.

At the final stage, main focus was given to traditional management inventory of Finland with estate level simulation and optimization approach. Inventory by stands still has many options in terms of source data and models, many of which are already included into system and pilot versions are in use, and some of them have been tested in the M.Sc. thesis by Anu Hankala (Hankala et al. 2007). During the project we identified a few issues which took special attention. The control of estimation error during simulation was traditionally ignored. Now, the system includes many tools to find reasons for erroneous estimations. For instance, data mining tools have been tested in order to improve the data quality (Mäkinen et al. 2007). Sometimes, the reason can be found from models. One special study was implemented to find out the behavior of tree models in exceptional stands (Välimäki 2007). Another issue was related to estimation of timber assortments. Specific dynamic optimization procedure was used to maintain this field. Also, a tailored study about bucking procedure for forest planning purposes was completed as a M.Sc. thesis (Kuusisto 2007).

The other strategic level and operational level inventory and simulation systems could be also implemented using SIMO-framework tools, but they were not included into first phase of SIMO. The main effort was spend with structure of simulator. The GIS part was started only to experimental level. The optimization methods were included to system as a final step and with minor test period. So, there is need for improvement in the many parts of this Information System. We hope that system is continuing to expand to different applications and open source policy will ensure the public interest to further develop this type of planning tools (www.simo-project.org).

8. References

Bailey, R.L., Abernethy, N.C. & Jones, E.P. 1981. Diameter distribution models for repeatedly thinned slash pine plantations. USDA Forest Service, general Technical Report SO-34. pp. 115-122.

Borges, J. G., A. Falcão, C. Miragaia, P. Marques and M. Marques. 2003. A decision support system for forest resources management in Portugal. In: G. J. Arthaud and T. M. Barrett (Eds.) System Analysis in Forest Resources. Proceedings of the Eighth Symposium, September 2730, 2000, Snowmass Village, Colorado, USA. Kluwer Academic Publishers, Managing Forest Ecosystems Vol. 7: 155-164.

Buongiorno, J. and Gilles, J.K. 2003. Decision methods for forest resource management. Academic Press. 439 p.

Buongiorno, J. & Mitchie, B.R. 1980. A matrix model of uneven-aged forest management. Forest Science 35: 548-556.

Chatziphilippidis, Gr. and Spyroglou, G. 2005. Modelling the Growth of Quercus frainetto in Greece. In: Hubert Hasenauer (Ed.): Sustainable Forest Management. Growth Models in Europe. P369-391. Springer

Chen, B.W. u. Gadow, K. v., 2002: Timber Harvest Planning with Spatial Objectives using the Method of Simulated Annealing. Forstwiss. Centralblatt 121: 25-34.

Chumachenko, S. I., V. N. Korotkov, M. M. Palenka and D. V. Politov, 2003. Simulation modeling of long-term stand dynamics at different scenarios of forest management for coniferous-broad-leaved forests. Ecological Modelling 170: 345-361.

de Coligny F., Ancelin P., Cornu G., Courbaud B., Dreyfus P., Goreaud F.,Gourlet-Fleury S., Meredieu C., Orazio C., Saint-André L., 2004. CAPSIS: Computer-aided projection for strategies in silviculture: open architecturefor a shared forest-modelling platform. In Nepveu G. (Ed.): Connection between Forest Resources and Wood Quality: Modelling Approaches and Simulation Software. Nancy, France. INRA-ENGREF, pp. 371-380.

Davis, L-S., Johnson, K.N., Bettinger, P.S. and Howard, T.E. 2001. Forest management – to sustain ecological, economic and social values. 4th edition. McGraw Hill. 804 p.

Eid, T. and Hobbelstad, K. 2000. AVVIRK-2000 – a large-scale forestry scenario model for long-term investment, income and harvest analyses. Scandinadian Journal of Forest Research 15: 472-482.

Fabrika, M., 2002: Multifunctional optimisation of stand tending by SDSS and growth modelling. Proceedings from International IUFRO Symposium on Management and Modelling Multifunctional Forest Enterprises and Properties. Sopron, Hungary - May 26-28, 2002, 41-53.

Fabrika, M., Ďurský, J, 2004.: Risk management of forest ecosystems by dynamic model SIBYLA. Proceedings from IUFRO conference on Sustainable Harvest Scenarios in Forest Management, WP 4.04.10, Tále, Slovakia, August 25-27, 2004, 123-136.

Gobakken, T. & al. 2008. T – a forest simulator for bio-economic analyses based on models for individual trees. To appear in Scandinavian Journal of Forest Research.

Gustavsen, N.G. 1977. Valtakunnalliset kuutiokasvuyhtälöt. Folia Forestalia 331. (in Finnish)

Gustavsen, H. 1998. Volymtillväxten och övre höjdens utveckling i talldominerade bestånd i Finland - en utvärdering av några modellers validitet i nuvarande skogar. Finnish Forest Research Institute, Research Notes 707. 190 p + app. (in Swedish).

Hägglund, B. 1981. Forecasting growth and yield in established forests – An outline and analysis of the outcome of a subproject within the HUGIN project (Rep 31). Department of Forest Survey, Swedish University of Agricultural Sciences.

Hankala, A., Mäkinen, A., Rasinmäki, J., Kalliovirta, J. & Kangas, A. 2008. The use of micro compartments in forest planning and its effect on forest owner's net income. pp. 64 In: Hahn, A., Knoke, T. & Schneider, T. (Eds.) Linking forest inventory and optimization. Abstracts of the international conference of the IUFRO Units 4.02.00 and 4.04.00. 1st-4th April 2008. Freising, Germany

Hoen, H.F. & Solberg, B. 1996. Forestry scenario modeling for economic analysis – experiences using the GAYA-JLP model. In: Päivinen, R., Roihuvuo, L., Siitonen, M. (Eds.). Large Scale Forestry Scenario Models: experiences and requirements. EFI Proceedings 5:435-440.

Hynynen, J., Ojansuu, R., Hökkä, H., Salminen, H. & Haapala, P. 2002. Models for predicting stand development in MELA system. Metsäntutkimuslaitoksen tiedonantoja 835. 116 s.

Hynynen, J., Ahtikoski, A., Siitonen, J., Sievänen, R. & Liski, J. 2005. Applying the MOTTI simulator to analyse the effects of alternative management schedules on timber and non-timber production. Forest Ecology and Management 207: 5-18.

Iverson, D.C. and Alston, R.M. 1986. The genesis of FORPLAN: a historical and analytical review of Forest Service planning models. Gen. Tech. Rep. INT-214. Ogden, UT: USDA Forest Service, Intermountain Forest and Range Experiment Station. 37 p.

Johnson, K.N. 1986. FORPLAN version 1: An overview. Washington DC: USDA Forest Service, Land Management Planning Section. 85 p.

Johnson, K.N., Stuart, T.W. and Crim, S.A. 1986. FORPLAN version 2: An overview. Washington DC: USDA Forest Service, Land Management Planning Section. 110 p.

Kangas, J. Uuttera, J. Wathén, M. Haapasalo, E., Laamanen, R., Soimasuo, J., Suutarla T. & Ärölä, E. 2006. Käyttäjien näkökulmia uuden sukupolven metsätietojärjestelmien kehittämiseen. Metsätieteen aikakauskirja 1/2006:54-59.

Koivuniemi, J. & Korhonen, K. 2006. Inventory by compartments. Chapter 16, pp. 271-278. In: Kangas, A. & Maltamo, M. (eds.) Forest Inventory, Methods and Applications. Managing Forest Ecosystems Vol. 10. Springer, Dordrect. 362 p.

Korpela, I. 2004. Individual tree measurements by means of digital aerial photogrammetry. Silva fennica Monographs 3.

Kuusisto, L. 2007. Apteeraussimulaattorin käyttö malliketjun jatkeena. Master's thesis, University of Helsinki.

Lappi, J. 1992. JLP – A linear programming package for management planning. Finnish Forest Research Institute, Research Papers 414, 134 p.

Lexer M.J., Vacik H., Palmetzhofer D., Oitzinger G. 2005. Improving forestry extension services for small-scale private landowners in southern Austria with a computer based decision support tool. Computers and Electronics in Agriculture 49:81-102.

Lämås, T. and Eriksson, L.O. 2003. Analysis and planning systems for multi-resource, sustainable forestry - The Heureka research programme at SLU. Canadian Journal of Forest Research, 33(3):500-508.

Mäkelä, A., Landsberg, J., Ek, A.R., Burk, T.E., Ter-Mikaelian, M., Ågren, G., Chadwick, D.O. ja Puttonen, P. 2000. Process-based models of forest ecosystem management: current state-of-art and challenges for practical implementation. Tree Physiology 20:289-298.

Mäkinen, A., Kangas, A., Kalliovirta, J., Rasinmäki, J. & Välimäki, E. 2008. Comparison of treewise and standwise forest simulators by means of quantile regression. Forest Ecology and Management. Manuscript accepted for publication 17th Jan 2008.

Mäkinen, A., Kangas, A. & Tokola, T. 2007. Applying data mining methods for forest planning data validation. Manuscript.

Maltamo, M. 1997. Comparing basal area diameter distributions estimated by tree species and for the entire growing stock in a mixed stand. Silva Fenn. 31: 53-65.

Maltamo, M., Eerikäinen K., Pitkänen, J. Hyyppä, J. & Vehmas, M. 2004. Estimation of timber volume and stem density based on scanning laser altimetry and expected tree size distribution functions. Remote Sensing of Environment 90:319-330.

Munro, D.D. 1974. Forest growth models—a prognosis. pp. 7–21 in Growth models for tree and stand simulation, Fries, J. (ed.). Department of Forest Yield Research, Research Notes Nr 30, Royal College of Forestry, Stockholm.

Næsset, E. 1997. A spatial decision support system for long-term forest management planning by means of linear programming and a geographical information system. Scandinavian Journal of Forest Research 12: 77-88.

Næsset, E. 2002. Predicting forest stand characteristics with airborne scanning laser using a practical two-stage procedure and field data. Remote Sensing Environ. 80: 88–99.

Næsset, E. 2004. Accuracy of forest inventory using airborne laser-scanning: evaluating the first Nordic full-scale operational project. - Scand. J. For. Res. 19: 554-557.

Nuutinen, T., Hirvelä, H. & Salminen, O. 2005. Valtakunnan metsien 9. inventointiin perustuvat hakkuumahdollisuusarviot vuosille 2003-2032 Lapin metsäkeskuksen alueella. Metsätieteen aikakauskirja 2B/2005: 289-305.

Nyyssönen, A. Mielikäinen, K. 1978. Metsikön kasvun arviointi. Acta Forestalia Fennica 163. (in Finnish)

Ohsawa, Y. Guo, W. Sakurai, M. 2002. A Data Structure for Spatio-Temporal Information Management. ISPRS Commission IV, Symposium on Geospatial Theory, Processing and Applications, Ottawa. Online at: http://www.isprs.org/commission4/proceedings02/pdfpapers/208.pdf

Oikarinen, M., 1983. Etelä-Suomen viljeltyjen rauduskoivikoiden kasvumallit. Communicationes Ins. For. Fenniae 113. (in Finnish)

Oksanen-Peltola, L. 1999. Johdanto. In: Heikinheimo, M. (ed.) 1999. Metsäsuunnittelun tietohuolto. Metsäntutkimuslaitoksen tiedonantoja 741. 105 p.

Palahí, M., Pukkala, T., Pascual, L., Trasobares, A. 2004. Examining alternative landscape metrics in ecological forest landscape planning: a case for capercaillie in Catalonia. Investigaciones Agrarias: Sist. Recur. For. 13 (3), 527-538.

Pavlata, O. 2004. Mg R-tree Library: A simple r-tree implementation with c++ source code. Online at: http://www.volny.cz/r-tree/.

Pretzsch, H., Biber, P., and Dursky, J. 2002. The single tree-based stand simulator SILVA: construction, application and evaluation. Forest Ecology and Management 162 (1): 3-21.

Pukkala, T. and Kangas, J. 1993. A heuristic optimization method for forest planning and decision-making. Scandinavian Journal of Forest Research 8: 560-570.

Pukkala T. 2004. Monikäytön suunnitteluohjelma Monsu. Ohjelmiston toiminta ja käyttö. University of Joensuu Faculty of Forestry. 72 pp. In Finnish.

Pukkala, T. (Ed.). 2002. Multi-objective forest planning. Managing Forest Ecosystems Vol 6. Kluwer Academic Publishers. Dordrecht. 207 p.

Rasinmäki, J. 2003. Modelling spatio-temporal environmental data. Environmental Modelling & Software 18:877-886

Reeves, C. R. & Beasley, J. E. 1993. Introduction. In: Reeves, C. R. (editor). Modern heuristic techniques for combinatorial problems. Blackwell Scientific Publications, New Yourm pp. 1-19.

Salminen, H., Lehtonen, M., Hynynen, J. 2005. Reusing legacy FORTRAN in the MOTTI growth and yield simulator. Computers and Electronics in Agriculture, 49: 103-113.

Saramäki, J., 1977. Ojitettujen turvemaiden hieskoivikoiden kehitys Kainuussa ja Pohjanmaalla. Metsäntutkimuslaitoksen julkaisuja 91.2. (in Finnish)

Seo, J.-H., Vilčko, F., Sánchez Orois, S., Kunth, S., Son, Y-M. u. Gadow, K. v., 2005: A case study of forest management planning using a new heuristic algorithm. Tree Physiology 25: 929-938.

Siitonen, M. 1996. Introduction to the MELA System and its use. In: Päivinen, R., Roihuvuo, L. & Siitonen, M. (eds.). Large-Scale Forestry Scenario Models: Experiences and Requirements. Proceedings of the International Seminar and Summer School, 15-22 June 1995, Joensuu, Finland. EFI Proceedings 5: 171-174.

Suvanto, A., Maltamo, M., Packalén, P. and Kangas, J. 2005. Kuviokohtaisten puustotunnusten ennustaminen laserkeilauksella. Metsätieteen aikakauskirja 4/2005: 413-428.

Tokola, T., Turkia, A., Sarkeala, J., & Soimasuo J. 1997. An entity-relationship model for forest inventory. Canadian Journal of Forest Research 27:1586-1594.

Tomé, M. & Burkhart, H.E. 1989. Distance-dependent competition measures for predicting growth of individual trees. Forest Science 35: 816-831.

Vacik, H., Lexer, J.M., Palmetzhofer, D.; Stampfer, K., Limbeck-Lilienau, B. (2004): Application of the Spatial Decision Support System CONES for regeneration planning in mountain forests. In: FERIC: A Joint Conference of IUFRO 3.06 Forest Operations under Mountainous Conditions and the 12th International Mountain Logging Conference "Improving the Bottom Line", June 13 - 16, 2004, Vancouver; CD ROM.

Vuokila, Y. ja Väliaho, H. 1980. Viljeltyjen havumetsiköiden kasvatusmallit. Metsäntutkimuslaitoksen julkaisuja 99.2. 271 s.

Välimäki, E. 2007. Kasvumallien validointi tiheissä metsissä. Master Thesis, University of Helsinki.

Wathén, M. 2007. Suunnittelunäkemys metsäorganisaatioissa ja sen vaikutus tietojärjestelmälle asetettaviin vaatimuksiin. Master Thesis, University of Helsinki. http://urn.fi/URN:NBN:fi:fe20072040