Overview

TRAGIC++ is an interactive forest stand simulation program. In TRAGIC++, the development of a forest is simulated as the collective dynamics of individually growing trees, each of which competes with the others for light, space, and nutrients. Interaction is possible by removing or planting single or groups of trees and by changing the environmental conditions (light and nutrients). The TRAGIC++ program is based on a relatively simple process-based tree growth model that can be parameterised to represent trees of different species.

The trees grow in a three dimensional world that consists of below and aboveground space. Below ground, the trees expand their root systems in search of "wusels", which are generic nutrient packets that are provided as external input to the system on a yearly basis. In addition decomposing litter releases "wusel" to the soil, simulating an internal nutrient cycle. Above ground, trees expand their crowns and supporting biomass according to their wusel input and net photosynthetic productivity. It is assumed that trees maximise their instantaneous (annual) growth rates up to the constraints imposed by photosynthesis and nutrient uptake.

A TRAGIC++ simulation proceeds in yearly time steps. During each simulation cycle, energy and wusels are input to the system, and then the trees grow, reproduce, and die according to the rules of the model. The overall cycle is as follows (this is the CForest::Cycle function):

```
StartCycle();

Replanting();
Shadowing();
CalcSigmaC();              //only used if climate data provided
Photosynthesis();
Dying();
Litterfall();
GeoConstraintZ();
GeoConstraintXY();

RootDeath();
Wuseling();
Regeneration();
Partition();
RootGrowth();

Thinning();
Decomposition();
Ageing();

FinishCycle();
```

The details of each of these processes are described further in the other sections of this document.

Most processes are computed for each individual tree on a species by species basis. For each species in the forest, the trees are updated in order from tallest to shortest. (Note that, in the program code, species are listed according to the order in which they are added to the forest. Since species are always treated in turn, this order, in addition to tree order, may affect simulation results in mixed species stands.)

The operation of the model is dependent upon the values of a number of key parameters that are input before runtime by the program's user (see parameter list). Some of these are forest level parameters that apply to all trees in the forest. Others are species level parameters that apply only to the trees of a given species. During runtime, the user also has the option to halt program execution, inspect all variables and modify the values of some of the parameters. In addition trees may be removed or planted (from the same or a new species) at any time during the run.

Climate

In the standard TRAGIC++ climate is not considered at all. Photosynthesis is estimated for each species by the annual yield of an unshaded unit of leaf dry matter (fixed SigmaC). It becomes modified, however, due to mutual shading of trees. There is an option to include climate data at daily temporal resolution in order to vary the photosynthetic capacity between years. This module is still under testing and not described here.

Ground

Wusels

The belowground and rooting zone in TRAGIC++ is a discrete, three dimensional space. It is regularly divided into cubes having the unit of $pixel^3$ (i.e. Voxel). Pixel length in metres is a forest level parameter of the model. The size of pixel and voxel can be specified in the forest parameter list. Default values used in the example built into TRAGIC++ are 1.0 and 0.2 m respectively. The ground space is bounded in the horizontal plane at the forest edges. Beyond the edges it is extended as a torus to provide the shading and competitive parameter for light and nutrient interception. In the vertical plane the soil surface and the soil depth bound it respectively (the depth of soil is also a forest level parameter). The soil, or ground, surface has a topology, in which the absolute elevation of each cubic pixel is determined by an arbitrary function. However, this function has to be provided prior to compilation, it is not part of the save and restart facilities. Currently a slightly modulated valley is used as the default surface (see parameters SIZE and SPACE in CTree.cxx at line 3066). If a special surface topology is required, contact Michael Hauhs at BITÖK.

Within the ground, there are "voxels", which constitute cubic pixels that may be exclusively occupied by the fine roots of one single tree. Every voxel has a standard fine root mass ("voxel weight", defined as a forest parameter), and this mass is therefore the smallest unit by which a tree can increase its root mass. Only one tree can occupy a voxel at any given time. The volume of the ground which is occupied by tree roots is called the "voxel space". During growth trees are updated in the sequence of their height. A voxel is assigned exclusively to any tree that attempts to grow into it. If this voxel had already been assigned as fine root to another (or the same tree), the voxel weight unit is added to the litter component. Root competition therefore increased the production of litter, but never the density of roots in any voxel. Small trees are favoured as they get the last chance of root growth in each cycle (see section on root growth below).

The random walk by which the Wusels are added to the forest each year is termed "wuseling". During each simulation cycle, a fixed number of nutrient compartments (called "wusels"; the number of wusels added each year is a forest parameter), plus a variable number of wusels regained from decomposition, are added to the ground in the form of a random walk process. Each wusel carries with it a fixed mass of nutrients. (If one wants to impose a strictly constant nutrient supply during a rotation the decomposition rate has to be turned off, e.g. by setting its parameter to tiny values; see below. In this case the mass budget can not be displayed.)

First, each new wusel is created, and is randomly assigned a position (x, y coordinates in pixels) in the ground. In the current version all wusel are started from the soil surface. Next, each wusel's position is increased by a value (i) selected randomly from the range $0<i<5$, where i is an integer. For the purposes of wuseling, the ground is assumed to be toroidal, so that there is wrapping in the x and y directions. However, in the vertical direction, wusels whose z coordinate exceeds the soil depth are irreversibly lost from the forest. After each wusels step of this random diffusion through the soil, each wusel's position is compared to the position of all of the tree-owened-voxels in the ground to see if it has "hit" a fine root. If a root has been hit, this hit its location and direction are recorded by the tree and is later used to fuel new biomass growth und to guide the direction of root growth. In addition, when a wusel hits a root, all or part of its nutrient content is be absorbed by the tree. Hence root uptake is a passive process during any given cycle, while the trees have to actively grow roots in order to increase their uptake chances (For further details, see root growth section.)

The model does not include any considerations of soil hydrology or groundwater flow. The probabilities for the random walk are not accessible through parameters they might be changed prior to compilation only (see in CWusel.cxx at line 163 the method: CWusel::DoWusel).

Decomposition

Detritus is regularly added to the soil due to fine root turnover, litterfall, senesence, and cutting. This annual litter biomass is distributed to three different quality classes depending on the nutrient content (to change the number of litter quality classes requires reprogramming as the budget would no longer hold). The decomposition releases the nutrients from each quality class according to the equations of Agren and Bosatta (1991). The parameter set for the decomposition routine can be specified at the forest level. The array containing the amounts of different litter quality becomes dynamically extended with each annual cycle it will be saved and reloaded along with the other run variables during a restart.

The nutrient content of the litter becomes gradually released during decomposition and is annually added as Wusel to the ground.

The conversion of the litter into quality classes is organised as follows: all woody litter (branches, transport roots, stem wood and heartwood) is transferred into the lowest quality class. For branches and transport roots this is done in the cycle when they die, while "dying" sapwood is first converted to heartwood and is only added when the respective tree dies and its timber is not removed. Leaf and fine root litter is added into the standard quality class in the cycle when they die.

If a tree dies and has still an internal pool of free nutrients these are added to the leaf litter up to its doubled nutrient concentration and then added to the high quality litter class. Extra nutrients remaining after that are taken out of the simulation by adding into a dummy variable

AccumI. This variable constitutes another export of nutrients from the forest. It is introduced because of the passive nutrient uptake by trees. One may add or compare AccumI with the variable LostWusel, describing the export over the lower boundary of the belowground voxelspace resulting from the random walk.

Trees

Major parts of a tree

A tree has above and belowground parts. Above ground, a tree is made up of a vertical stack of *segments*, each of which may have up to 8 lateral *branches* that extend radially from the centre of the segment. On each branch, there are *needles* (also called *foliage*). Below ground, the tree has *transport roots* and *fine roots*.

A tree's mass is divided into three main parts: productive biomass, supporting biomass and heartwood. Productive biomass is the sum of the tree's foliage mass and fine root mass. Supporting biomass is the sum of a tree's branch, segment and transport root masses. Together, the sum of the productive and supporting biomass constitute a tree's total living biomass. Heartwood consists of supporting biomass that is no longer alive (i.e., sapwood that does no longer supply any foliage). Heartwood simply serves as structural material in the tree's stem.

The biomass and bookkeeping in TRAGIC++ focuses at theses masses and the locations of fine roots, stem segments and branches. It does not keep track of the locations of transport roots, needle age classes or of outer shape of the stem. The diameter at 1.3 m height is calculated instantaneously from the form factor, tree height and total stem biomass (sapwood and heartwood). Hence excessive height growth on top of a stagnant sapwood pool may cause the diameter and cross section to decline. This is, of course, to be avoided through parameterisation. In other words TRAGIC++ uses a constant form factor for each species (see species parameter list).

For each tree, masses of excess, or unused, energy (carbon) and nutrients (nitrogen) are stored in the variables *EPool* and *IPool*, respectively. The size of these pools will increase if a tree is not able to meet its potential foliage or root growth. The tree can also draw from these pools as required, but backflow of energy is restricted to 10% of current value of NetPhoto (see line 1625 in Ctree.cxx: *Reusefraction = 0.1F;*). In the current version backflow of nutrients from the pool is entirely prohibited (see line 1610 in Ctree.cxx: *Double ReleaseFactor = 0.0F;* as it would give an error in the nutrient budget otherwise)

Other derived variables used to describe the size and shape of a tree include:

Elevation = ground level elevation of tree's stem (m) due to topology provided
Height = vertical distance between ground elevation and top of tree crown (m)
Real height = elevation + height (m)
Crown height = distance between the top and bottom of a tree's crown (m)

Volume = volume of tree crown, assuming a conical shape (m^3)
Tree bounds = edges of the rectangle that circumscribes a tree, as defined by the radii of it's longest branches

As a tree gains mass, its total number of segments and, thus, its cross-sectional area increase. A tree may also expand radially by growing more branches, or by adding foliage to existing branches. Growth of the crown geometry is provided by two processes: the height growth as a function of the individual height growth strategy (provided by species-specific parameter) and the new foliage from the current year. It is assumed that the leader shoot does not require any additional foliage and hence new foliage is distributed by adding branches to last year's leader and lateral expansion of existing branches. In addition the actual height growth may be less than the potential height growth controlled by the root uptake of nutrients (see below).

Photosynthesis

The way that a tree's photosynthesis is calculated depends upon the program settings. The simplest way is "via fixed SigmaC". With this method, it is assumed that the tree converts radiant energy to biomass at a fixed potential rate per year, equal to the value of the SigmaC tree species parameter.

A segment's potential gross photosynthesis (kg_{DW}) is calculated as a function of foliage weight (W_f), SigmaC, and the photosynthetic light ratio (PLR, kg_{DW}/kg_C). The total potential gross photosynthesis for a tree (P_g, kg_{DW}) is then computed as the sum of all of the segment level values. A given segment's gross photosynthesis is only counted if it exceeds respiration otherwise the branches are shed from this segment for which respiration exceeds photosynthesis.

The equation is as follows:

$$P_g = \sum_{i=1}^{n} (W_{f_i} \cdot \text{SigmaC} \cdot \text{PLR}_i)$$

where, *i* denotes a segment with a positive net photosynthesis value, and:

$$\text{PLR}_i = \frac{1}{1 + e^{\frac{2\log(SFPH_i) - \alpha}{\beta}}}$$

$SFPH_i$ = Shading foliage per hectare for the local segment

α and β are the species specific shading parameters. There is hence some inconsistency as the shading is calculated from any biomass that is within the upward cone from a segment including other species of course. However, the extinction parameters to calculate PLR are used from the species that is shaded by this biomass. Hence it is rather the shading sensitivity that becomes lumped into these parameters.

Once gross photosynthesis and respiration (see below) have been computed, a tree's *net* photosynthesis (P_n, kg_{DW}?) is calculated as follows:

$$P_n = \frac{P_g + \text{MaintResp}}{\text{RespG} + \text{CperDW}}$$

Other photosynthesis calculation methods have not been sufficiently tested and are not included in this description.

Respiration

Maintenance respiration (kgDW) is computed for both above and belowground tree parts as follows:

$$\text{MaintResp}_1 = \text{R}_\text{r} \cdot \text{W}_\text{r} + \text{R}_s \sum_{i=1}^{n} \left(\text{W}_{s_i} + \text{W}_{b_i} \right)$$

where i denotes a tree segment.

If the net photosynthesis calculation method is via fixed SigmaC, maintenance respiration is also considered to include a foliage component, such that:

$$\text{MaintResp}_2 = \text{MaintResp}_1 + \text{R}_\text{f} \cdot \text{W}_\text{f}$$

Partitioning

It is assumed that trees maximise their growth rate by dynamically adjusting the specific activity of roots (calculated from the mass of currently acquired nutrients per dry mass of fine roots: *SigmaI*) with the specific activity of shoots (given in currently acquired Carbon per dry mass of roots: *SigmaC*[1]). Thus if root uptake constraints growth proportionally more roots are grown and if shoot photosynthesis constraints growth more foliage is grown.

Hence a tree's net mass increase due to photosynthesis is partitioned to above and below ground parts of the tree according to the following rules that are intended to maintain a constant ratio between a tree's root uptake (σ_I, $\text{kg}_\text{I}/\text{kg}_\text{DW}$) and foliage photosynthesis (σ_C, $\text{kg}_\text{C}/\text{kg}_\text{DW}$). σ_I and σ_C are calculated as follows:

$$\sigma_I = \frac{\text{HitAmount} \cdot (1 - \text{S}_\text{r})}{\text{W}_\text{r}}$$

where HitAmount is the mass of wusel absorbed by the fine roots in the current cycle (kg_I).

$$\sigma_C = \frac{\text{P} \cdot \text{CperDW}}{\text{W}_\text{f}}$$

A ratio (α_r, $\text{kg}^2_\text{I}/\text{kg}^2_\text{C}$) is then computed to link the two:

$$\alpha_r = \frac{\text{IFract} \cdot \sigma_I}{\text{CperDW} \cdot \sigma_C} \qquad \alpha_r < 10.5$$

[1] This is an species specific parameter provided on input, unless the option with variable climate is used. In the latter case it is dynamically calculated from the meteoric variables.

where IFract is the ratio of nitrogen content in tree to total living biomass (kg_I/kg_{DW}):

$$\text{IFract} = \frac{\text{PIs}_1 \cdot (\text{W}_\text{t} + \text{W}_\text{s} + \text{W}_\text{b}) + \text{PIs}_3 \cdot (\text{W}_\text{f} + \text{W}_\text{r})}{\text{W}_\text{t} + \text{W}_\text{s} + \text{W}_\text{b} + \text{W}_\text{f} + \text{W}_\text{r}}$$

PIs (kg_I/kg_{DW}) is a series specifying the nutrient contents in various biomass components, where:

$$\text{PIs}_0 = 0.003;\ \text{PIs}_1 = 0.003;\ \text{PIs}_\text{n} = 0.003(\text{n}+1)\ \ \text{n}>1$$

[An additional variable is always calculated but seems to be no longer used:

$$\beta_r = \frac{\text{SenFRootWeight} - \alpha_r \cdot \text{SenFoliageWeight} \cdot \sigma_I}{\text{P}_\text{n}} \qquad \beta_r \geq 0.005\ ]$$

The partitioning coefficients ($\lambda_{f,r,b,t,s}$) are then calculated iteratively for a tree (max 16 iterations), until the sum of these is equal to $1 \pm \varepsilon$. This is done according to the following general procedure:

First, select an initial value for λ_f (start with 0.5), calculate the new crown growth based on this value, and then set values for λ_b and λ_s as follows:

$$\lambda_b = \frac{\text{W}_{\text{b}_\text{t}} - \text{W}_{\text{b}_{\text{t}-1}} - \text{W}_{\text{b}_\text{recycled}}}{\text{P}_\text{n}} \qquad \lambda_s = \frac{\text{W}_{\text{s}_\text{t}} - \text{W}_{\text{s}_{\text{t}-1}} - \text{W}_{\text{s}_\text{recycled}}}{\text{P}_\text{n}} \qquad 0 \leq \lambda_{b,s} \leq 1$$

Root partitioning coefficients are estimated as follows:

$$\lambda_r = \text{Max}\left(0.02, \alpha_r \cdot \lambda_f\right) \qquad \lambda_r \leq 0.7$$

$$\lambda_t = \frac{\text{W}_{\text{t}_\text{t}}}{\text{P}_\text{n}} \qquad 0 \leq \lambda_t \leq 0.3$$

If the sum of the five coefficients is not close to one, the value of λ_f is then modified a little and the process is reiterated.

In the above equations, the double-subscript *t* is used to refer to the most recently computed value in the iteration cycle, and *t-1* is the value that was recorded one cycle prior to *t*. Recycled sapwood in the branches and segments of a tree is computed during tree senescence routines (see description below).

If a tree's HitAmount (kg_I absorbed by fine roots) is less than the value of the IIcrit species parameter, or if the tree has no net photosynthesis this cycle, then partitioning is not performed. In such cases, the tree is said to have undergone a dry cycle. If a tree has more than DieMaxDryCycle (species parameter) consecutive dry cycles, it will die. This is the only way a tree can die due to lack of nutrients.

Regeneration

If regeneration is selected (in the species list), then each mature tree (> 25 years) in the forest, at each time step, has the opportunity to regenerate (currently 1 attempt is allowed for each tree/cycle, however the offspring may fail to find a suitable place to grow (see below) and is in that case transferred to the litter cohort).

A new tree is created with the standard initial settings for its species. The new tree will initially obtain the height strategy (a_o, a_1, a_2) and partition-root-to-height growth parameters from the parent tree. If mutation is switched on, these four parameter values will then be mutated according to the deviation levels specified as program parameters.

A random location is chosen at the ground and it is tested if the newly generated tree would fit into this location without aboveground overlapping existing trees. If it fits it is added to the list of trees and the stand statistics are updated accordingly. For a detailed description how this updating of the various classes is organised in TRAGIC++ see Dörwald (1999).

Height growth

A tree's potential height growth over a given year is modelled with the following equation:

$$\Delta H_{pot} = \frac{a_1}{A} + \frac{2a_2 \log(A)}{A} * e^{a_o + a_1 \log(A) + a_2 (\log(A))^2}$$

The value of ΔH_{pot} is then modified according to the current growing conditions of the tree as follows:

$$\Delta H_{real} = \Delta H_{pot} * \eta$$

where:

$$\eta = (1 - \lambda_R)^4 * \text{Root2Height} \quad 0.02 \le \eta \le 1.5$$

If a tree has no geometrical constraints on its growth, then its total height will be incremented by ΔH_{real}. If, however, the branches of another tree cover the top of a tree's crown, its height growth may be limited. In such cases, if the amount of free space above the tree is less than

ΔH_{real}, the tree will only increase in height by the amount specified by the minimum height growth species parameter.

This partitioning variable for root growth (*LambdaR)* is used to adjust the potential height growth to the actual site conditions in terms of nutrient availability. The factor *Root2Height* is provided form the species parameter set, but is treated as a tree specific variable, as it may be changed by mutations during inheritance.

Thus trees that need more roots (i.e. on poor soils) reduce their height growth by this mechanism. The sensitivity of this process may be explored over evolutionary runs during which the strategy parameters are allowed to change during regeneration events (see parameter list: strategy deviation parameters at the species level). Only the strategy deviation parameter for *Root2Height* is also used during a planting event in order to randomise this factor in the individual trees due to a normal distribution where the standard deviation is given in the deviation parameter. When one wants to plant identical trees only, make sure the deviation parameter for initial tree height (*MaxDHeight*) and for the *Root2Height* factor are set to zero.

Crown growth
The second process that controls crown expansion is the distribution of new foliage. The distribution function *DIST* has three species-specific parameter (CrnPar0, CrnPar1, CrnPar2) for a quadratic function (see strategy parameter list and line 687 in CTree.cxx):

```
Double Value =   Owner->Params.Strategy.PartitionCrown[0]*sq(x) +
                 Owner->Params.Strategy.PartitionCrown[1]*x +
                 Owner->Params.Strategy.PartitionCrown[2];
```

This prescribes the vertical distribution within the crown. This distribution may become locally modified by further parameters described below.

Among branches the new foliage is distributed due to horizontal differential shading. The branch with lowest and highest shading biomass (i.e. biomass that casts shade **onto** this branch) receives the highest and lowest amount of new needles, respectively. The species parameter *StemPlasticity* controls how much the difference between the basal coordinates of the stem (base) and the centre of the shading biomass distorts the segment links out of the vertical, i.e. by increasing this parameter trees are allowed to grow towards light gaps.

Once the new foliage for each branch is assigned, it is decided how much each individual branch has to be extended. This is controlled by two parameters (*InOutPart, Kg2Cm*). *InOutPart* says how much of the new branch foliage is used for extending the branch. For this extension *Kg2Cm* is used to convert foliage mass into crown sector volume and the radius for this branch is updated accordingly. In addition a maximum of branch overlap is checked. If a branch extends more into a neighbour crown than allowed by this parameter. In the current version almost no overlap is allowed. This can only be changed prior to compilation (line 49 in CTree.cxx). Excess new foliage that becomes "ungrowable" due to these constraints is transferred to the energy pool variable (*EPool*).

```
#define BRANCH_MAXOVERLAP (0.0005)
```

Underground, a tree increases its root mass as required to access nutrients ("wusels") in the soil. A tree's growth in TRAGIC++ may become limited by geometrical or by biomass constraints (lack of C or nutrients) in its aboveground parts. The underground growth is only

restricted by the total volume provided and the immortal voxel, otherwise no geometrical restrictions apply to the fine root growth, that is supposed to be entirely opportunistic (i.e. into the direction where the roots where most often hit by wusel during the last round of the random walk). No geometric relationship is established between transport roots and fine roots. The transport roots are an abstract compartment for each tree that is calculated from the total age of its roots and a transport root cost factor that is set globally for the underground (*TRootAgeCost* in forest parameter list).

Root growth

A tree's root growth has two parts: fine root growth and transport root growth. Fine root growth is partly a random process (in terms of geometry), whereas its mass is determined by net photosynthesis and nutrient availability (via wusel hits). Transport root growth is a deterministic function of fine root growth.

For each tree, potential fine root growth is first calculated as a function of net photosynthesis and root partitioning, where:

$$\text{NrOfFueledVoxel} = \text{Max}(0, \frac{\lambda_R * P_n}{\text{VoxelWeight}})$$

this value is compared with the number of "growable" voxels, which is a function of wusel availability, and the smallest of the two values is used to compute fine root growth.

$$\text{NrOfGrowableVoxel} = \left(\frac{50 \cdot \text{MaxRootGrowth}}{\text{PixelSize} / \text{VoxelPerPixel}} \right) \cdot N_{hits}$$

where N_{hits} is the number of root voxels that have been hit by a wusel in this cycle.

For a number of times, equal to the number of potential new root voxels (either NrOfFueledVoxel or NrOfGrowableVoxel), the tree is given the opportunity to grow a new voxel. For each potential case, the tree grows new fine root mass into the voxel where a wusel is located that hit one of its roots. The tree then gains fine root mass equal to the mass of one voxel. Next, the wusel for that hit is moved one pixel away, allowing the tree to grow another voxel in that direction in another iteration. The root growth loop iterates cyclically through all of the tree's recorded wusel hits (whose locations move by one pixel each time they are "used"), until the tree has attempted to grow it's allowed number of potential new voxels.

Note that voxels are "owned" by a tree. If one tree grows into another tree's voxel, ownership is transferred, and the initial voxel owner loses fine root mass. Each tree, however, has one "immortal" voxel located directly below its stem. This voxel cannot be lost to another tree. When a voxel is lost by a tree, the equivalent amount of fine root mass is added to the soil humus.

Transport root growth ("fine root cost", FRC, kg_{DW}) is computed as Min(FRC, $\sqrt{\text{FRC}}$) where:

$$FRC = NrOfNewVoxels * \frac{AgeOfVoxel}{NrOfVoxel} * TRootAgeCost$$

NrOfNewVoxels = number of voxels gained by tree in this cycle
AgeOfVoxel = tree variable, the value of which is increased each time a tree gets a new voxel
NrOfVoxel = variable equal to the number of voxels currently owned by the tree

Once the fine and transport root mass increases have been computed, a carbon budget is performed where:

$$Diff_R = \lambda_R \cdot P_n - \Delta W_r \qquad \text{and} \qquad Diff_T = \lambda_T \cdot P_n - \Delta W_t$$

ΔW_r (kg_{DW}) and ΔW_t (kg_{DW}) are the net mass increases due to the growth of new roots. $Diff_R$ (kg_{DW}) and $Diff_T$ (kg_{DW}) are subtracted from the tree's net photosynthate (P_n). If P_n is exceeded, then additional mass is removed from the tree's ePool to compensate.

Next, a tree's realized root growth is compared to its potential root growth. If a tree has not realized its potential growth, the difference is added to the iPool.

Realized root growth (IIRealRootGrowth, kg_I) is calculated as:

$$\Delta W_r \cdot PIs_3 + \Delta W_t \cdot PIs_1$$

where PIs_1 and PIs_3 (kg_I/kg_{DW}) are taken from the series described previously.
Potential root growth (IIPotentialRoots, kg_I) is calculated as:

$$(\lambda_R \cdot PIs_3 + \lambda_T \cdot PIs_1) \cdot P_n$$

Stem growth

Stem diameter (D, m) for a given segment is calculated as a function of total living and non-living stem biomass, and tree height (H, m), using the following formula:

$$D = 2 \cdot \left(\sqrt{\frac{W_h + W_s}{\pi (PHIs)(RHOs) H}} \right)$$

Senescence

As a tree becomes larger, and some parts of its mass are largely shaded, some limbs begin to senesce. On branches that are no longer productive, sapwood and needles will be dropped as litter. Also, a portion of segment sapwood will be turned into non-living heartwood whenever foliage is shed from the tree.

Heartwood production

A young segment's mass is considered to be made up entirely of living biomass (sapwood). As a tree grows, lower segments begin to produce heartwood when the following condition is true:

Segment.UpperHeight < Tree.LowestSeg.Height + (Tree.Height – Tree.LowestSeg.Height) *SEGMNT_HEARTWOOD_PRODFACTOR2

where the height of a segment is the difference between its upper and lower heights, and SEGMNT_HEARTWOOD_PRODFACTOR2 = 1.0.

The above condition is stored in a boolean segment level variable called "heartwood producing death height". As this variable is currently set to one heartwood is built from each segment that reaches max *NeedleAge*.

For each simulation cycle, the "health" of each segment in each tree is assessed. First, any segments that have a P_n value which is less than MaintResp are marked as "dead". Next, for all living segments, the age of the needles on each branch is assessed, and if they are found to be older than the maximum allowable needle age, their mass is added to the tree's senescent foliage litter pool, and the branch is marked as dead.

The amount that becomes heartwood (SenSegSapWeight, kgDW) is calculated as follows:

SenSegSapWeight = SapwoodRecyclingFactor*DeadSegmentSapwood

Note that no recycling is effective when the *SapwoodRecyclingFactor = 1* whereas complete recycling and thus no generation of heartwood implies *SapwoodRecyclingFactor = 0*. *SenSegSapWeight* is added to the tree's heartwood mass, and the remainder is stored as recycled sapwood weight. The recycled sapwood weight is used elsewhere in the model to calculate the sapwood partitioning factor, λ_s .

Senescent branch sapweight is added to the litter pool (PITSB). Of course, dead branches also make up senescent branch sapweight.

A tree also produces litter from other sources as a regular part of growth and functioning. Needles that are older than the maximum needle age (tree species parameter) are dropped, and fine root mass is returned to the soil due to the imposed fine root mortality.

The death of fine roots is given by a fixed annual mortality (species parameter for senescence of roots: Sr). Thus the roots are characterised by a half life time, whereas all other living biomass die by either achieving a maximum age (leaves) or due to local negative energy budgets (e.g. if old branches require a net input of energy). However, each tree is assigned at least one immortal voxel below its stem base (see wuseling section). This root voxel cannot be overgrown by any other tree neither can it die due to the assigned fine root mortality. Hence the tree can only die by running out of foliage and hence energy, but it will always have at least one root voxel.

As the hits of the Wusels guide the direction of root growth a tree with an extending root system may have been hit by only few or no wusel during the last cycle. In this case a procedure termed pseudo-wuseling is used to provide virtual hits. They stem from the same random walk, but do not deliver nutrient along with their hits.

Shading

The "Shadowing" routine is called at the begining of each iteration of the yearly forest cycle. It is used to calculate the mass of foliage that contributes to shading on each branch of a tree. The shading foliage mass will come in part from branches on the same tree as the shaded branch, and also from branches on neighbouring trees. The shading routine does not consider the position and angle of the sun or the species that is casting this shade.

The shading foliage mass is later used in the crown growth routines to calculate the amount of photosynthesis and hence of new biomass that will be added to a branch.

A value for the shading foliage mass for each of the branches in a forest is calculated according to the following procedure:

For each tree (called the "shadower"), the first step in the shadowing routine is to compile a list of all trees whose bounds fall within the tree's shading rectangle. The shading rectangle is defined as:

ShadowerTreeBounds expanded to:
(ShadowerTreeRealHeight-ShadowedTreeElevation)*tan(ShadowAngle)

where:

ShadowAngle =	half angle of shadow cone (forest parameter)
ShadowerTreeBounds =	rectangle whose edges are defined by the distance of the tree's longest branch

Any tree whose bounds lie within the area of the expanded bounds rectangle of the shadower tree is added to the list of candidate trees for shading. The bounds are extended by the size of the plot and then truncated at the stand edges (i.e., there is only one wrapping in each direction around the torus to find shading tree).

Next, for each tree in the list of shaded trees, each branch is checked to see if it falls below any of the branches of the shadower tree. If a branch falls below a branch on the shadower, then a mass of shading foliage is added to that branch. The amount of shading foliage to add is computed as follows:

ShadingFoliageWeightToAdd =
CumFoliageWeightofShadingBranchSegment/BRANCHES

where the cumulative foliage weight of the shading branch segment is the mass of all foliage on that branch's segment, plus the sum of all foliage on the segments above. BRANCHES is a program parameter, the value of which is equal to 8.

If a branch shadows more than one branch on a segment, then the ShadingFoliageWeightToAdd is distributed evenly amongst all branches on the shaded segment.

Geometrical limitations to growth

The above-ground forest space is divided into a 2D array of pixels that describe XY location. The position of any part of a tree is therefore defined by its XY location (in pixels or metres) and its height above the ground. No two parts of a tree can occupy the same position. The

routines, GeoConstraintZ and GeoConstraintXY, check the positions of tree stems, segments and branches relative to others to ensure that the trees do not overlap in space.

The GeoConstraintZ function checks for the presence of another tree's branches or segments above the crown of a tree. If a tree's crown is obstructed, the value of the tree's FreeHeight variable is set to the difference in height between the crown tip and the obstructing branch or segment. The tree's net height growth is then constrained to the value of the MinimumHeightGrowth tree species parameter.

The GeoConstraintXY function checks for restrictions to a tree's side growth. Restrictions to the side will prevent a tree from growing branches of even lengths all around a segment. They may also cause a tree to begin to bend its stem away from the restrictive object. When this occurs, new segments are grown with pixel locations slightly offset from the locations of previous segments. The degree of offset can be set with the StemPlasticity tree species parameter.

Forest

Cutting

When a user specifies that tree should be cut, there are several options that can be selected regarding the export of biomass from the forest: no export, timber export, export all but heartwood, and biomass export. These differ slightly with regards to which parts of the tree's biomass are removed from the forest, and which are left on the forest floor.

If "no export" is selected:

TreeYield = SegmntSapwood + Heartwood

Replanting

In addition to "natural" regeneration, the program enables the user to "plant" trees in the forest. Thus, near the beginning of each simulation cycle, the "replanting" function is called in which new trees with standard initial settings for their species are created and situated in the positions specified by the user.

Thinning

This is the main interactive feature of TRAGIC++. Trees can be visually inspected in 2- and 3d representations. Navigation through the forest stand is possible by using the keyboard keys, but greatly facilitated by using a "spacemouse" (see website). Histograms of all tree variables (e.g. diameter distributions) can be interactively created at run time in order to decide which trees to mark (positive) or remove (negative selection).

A Java front end application has been added to allow for a remote interactive thinning exercise through the Web (by Mark Stricker, see Webpage). The transfer of data between TRAGIC++ and Jtragic uses XML Files, the same format that is also used to save and restart simulation runs.

Fire

If the forest is destroyed by fire (at a predefined probability set at the forest level), 50% of the heartwood is added to the litter fraction while all other biomass is exported from the stand.

Mass budget
The performance of the programm is checked by continuously updating the budgets for Carbon und Nutrients. In both case all input and output over the course of the run and balanced against the current pools sizes within the forest.The two variables (*StatTotalCarbonBudgetError, StatTotalLoadBudgetError*) at the forest level are given in absolut units and should be compared to total inputs, for example. Same inevitable small errors occur when regeneration is allowed. In some cases the budget may become wrong due to a restart. In general, runs should not be accepted, if the relative error compared to total inputs exceeds 0.1 %. If exceptional budgets errors are reproducable contact BITÖK).

Parameter list

- list parameters for forest & species with brief description of each, and link to equation where they are used.
- (see other file)

List of symbols

Abbreviations:

DW dry weight (dry biomass)
C carbon
I soil nutrients (content of a wusel)

A age of tree (years)
$a_{0,1,2}$ height growth parameters

D stem diameter (m)
H tree height (m)

W_t mass of transport root (kg_{DW})
W_s mass of stem (stem sapwood) (kg_{DW})
W_r mass of fine root (kg_{DW})
W_b mass of branch (branch sapwood) (kg_{DW})
W_f mass of foliage (kg_{DW})
W_h mass of heartwood (non-living tree biomass) (kg_{DW})

P_g gross photosynthesis (kg_{DW})
P_n net photosynthesis (kg_{DW})

R root to height partition tree strategy parameter (no units)

References:

Agren and Bosatta (1991)
Dörwald, W. (1999)