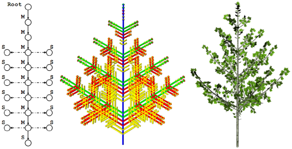
Structural factorization

Katarína Streit (Smoleňová)

ksmolen@gwdg.de
Departement Ecoinformatics, Biometry and Forest Growth
Georg-August-University Göttingen

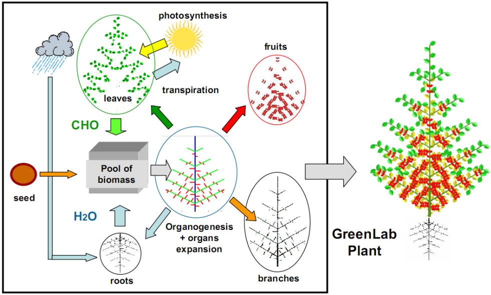
September 17, 2013 / Prague

“Modelling of Ecosystems by Tools from Computer Science”



GreenLab

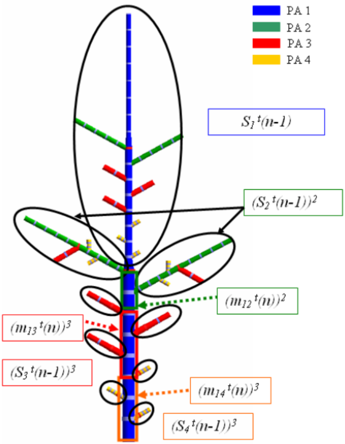
- Functional-structural plant model
  (more in my talk tomorrow!)
- Fundamental result - **structural factorization** (SF)



(Cournède *et al.*, 2006)

Basic idea

Based on the concept of physiological age, plant structure is factorized into smaller parts (substructures) that may repeat themselves a large number of times (Cournède, 2009)



(Letort, 2008)

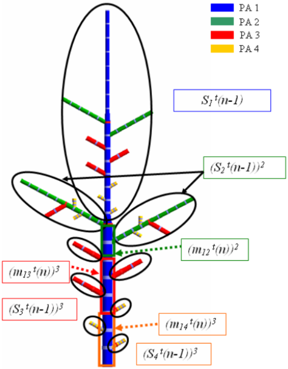
Description of plant structure

- Substructure - complete plant structure, generated after one or several cycles by a bud
- Substructure $S_p^t(n)$ characterized by
  physiological age (PA) p
  chronological age (CA) n
  growth cycle t

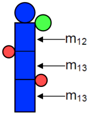
Substructure decomposition

At each t, substructure is decomposed into:

- Base growth unit (GU) (the oldest GU) $\prod_{p\leq q\leq P}\left(\left(m_{pq}^{t}(n)\right)\right)^{u_{pq}(t+1-n)}$
- Lateral substructures borne by the base GU (one cycle younger) $\prod_{p\leq q\leq P}\left(S_{q}^{t}(n-1)\right)^{b_{pq}(t+1-n)}$
- Substructure grown from the apical bud of the base GU (one cycle younger) $S_{p}^{t}(n-1)$



(Letort, 2008)



$m_{pq}^{t}(n)$ - metamer at cycle t of PA p and CA n, bearing buds of PA q

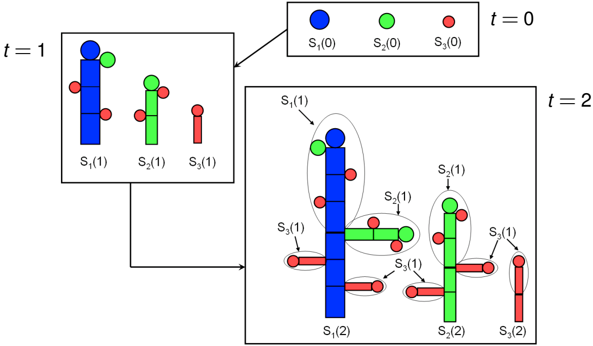
u_{pq} - number of metamers m_{pq} in a growth unit of PA p

b_{pq} - number of axillary buds of PA q in a growth unit of PA p

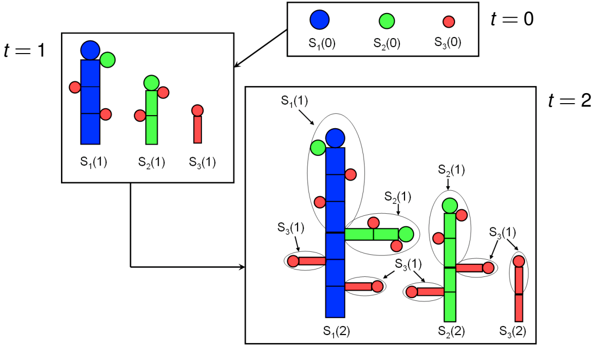
Inductive construction of plant structure

- Substructures are built by induction:
    - Substructures of CA 0 are buds
    - If they built all the substructures of CA $n-1$, we can deduce the substructures of CA n

Substructure organisation



Substructure organisation



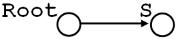
$$S_p^t(n) = \left[\prod_{p \le q \le P} \left(m_{pq}^t(n)\right)^{u_{pq}(t+1-n)} \left(S_q^t(n-1)\right)^{b_{pq}(t+1-n)}\right] S_p^t(n-1)$$

Recursive version of SF in XL

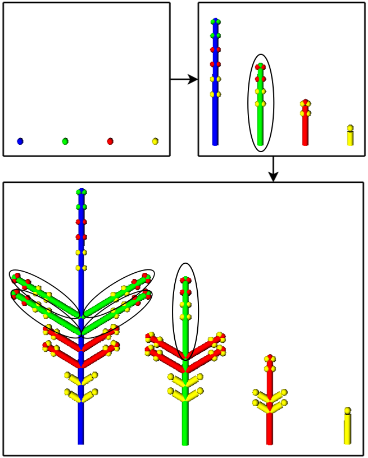
```
module Bud(int p) extends Sphere
{ ... /* set radius, shader */ }
module Metamer(int p, int q) extends Cylinder
{ ... /* set radius, length, shader */ }
module Substructure(int p, int n) ==>
   if (n > 0) (
      for (int j = 5; j >= p-1; j--) (
         for (int i = 0; i < u[p-1][j]; i++) (
            RH(ang_ph[p-1]) Metamer(p, j+1)
            for (int k = 0; k < b[p-1][j]; k++) ( [
               if (k > 0) ( RH(360.0 / b[p-1][j] * k) )
               RU(ang_br[p-1][j]) Substructure(j+1, n-1)
            ] )
         )
      )
      Substructure(p, n-1)
   ) else (
      Bud(p)
   );
public void run() [
   Axiom ==> Substructure(1, 0);
   Substructure(p, n) ==> Substructure(p, n+1);
]
```

Recursive version of SF in XL

```
module Bud(int p) extends Sphere
{ ... /* set radius, shader */ }
module Metamer(int p, int q) extends Cylinder
{ ... /* set radius, length, shader */ }
module Substructure(int p, int n) ==>
   if (n > 0) (
      for (int j = 5; j >= p-1; j--) (
         for (int i = 0; i < u[p-1][j]; i++) (
            RH(ang_ph[p-1]) Metamer(p, j+1)
            for (int k = 0; k < b[p-1][j]; k++) ( [
               if (k > 0) ( RH(360.0 / b[p-1][j] * k) )
               RU(ang_br[p-1][j]) Substructure(j+1, n-1)
            ] )
         )
      )
      Substructure(p, n-1)
   ) else (
      Bud(p)
   );
public void run() [
   Axiom ==> Substructure(1, 0);
   Substructure(p, n) ==> Substructure(p, n+1);
]
```

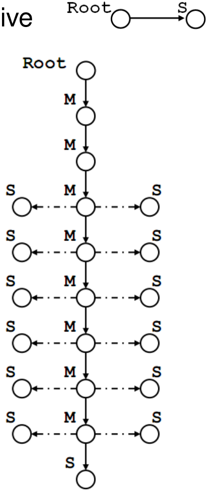


Demonstration example



3 versions:

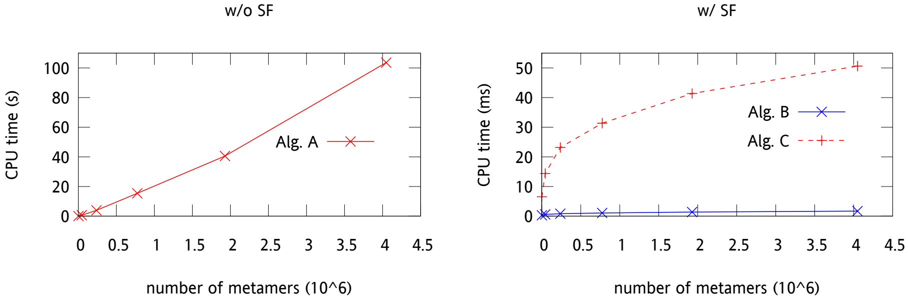
- no instantiation rules
- recursive
- lists



Simulation Performance

- Measured in GroIMP
- System: Intel Core i7 CPU 950, 3.07 GHz, 12 GB RAM
- Tree structure generation using XL rules
  - A - no instantiation rules, basic
  - B - instantiation rules, recursion
  - C - instantiation rules, lists
- Instantiation rules applied in the end of measured period (B, C)
  - Not included in the measurement

Simulation Performance



Alg. A - w/o inst. rules, Alg. B - w inst. rules (recursive), Alg. C - w inst. rules (lists)

Thank you for your attention!