# Models for hierarchical inheritance structures in object-oriented programming languages

Mariacristina Romano

Università degli studi di Milano
Facoltà di Scienze e Tecnologie
*Corso di laurea magistrale in Fisica*

15 aprile 2015

# Introduction

Complex Systems and Computer science

# Introduction

Complex Systems and Computer science

Object-oriented paradigm

# Introduction

Complex Systems and Computer science

Object-oriented paradigm

Inheritance

# Introduction

Complex Systems and Computer science

Object-oriented paradigm

Inheritance

Code reuse

# Introduction

Complex Systems and Computer science

Object-oriented paradigm

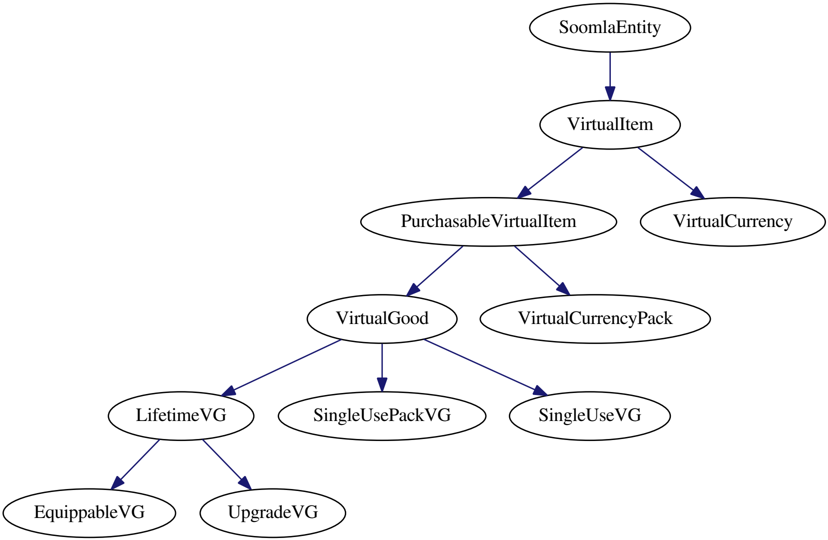
Inheritance

Code reuse

Hierarchies

# Example of inheritance hierarchy

## Data Analysis - Project Soomla Cocos2dx



# Noisy complex system dataset

## Data Analysis - dataset

Packages have been downloaded from **GitHub**, the actual largest code host on the web.

To give a **complete overview** of inheritance hierarchies, three different programming languages have been analyzed.

The **dataset** contains:

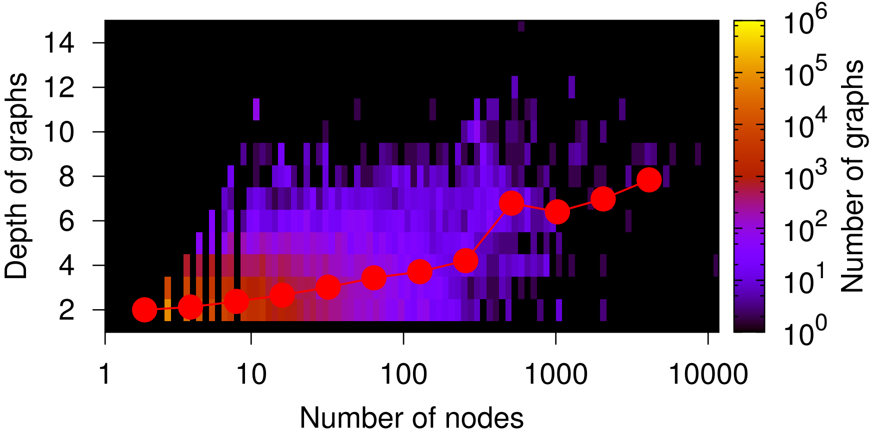
- 17333 C++ projects (3233447 hierarchies)
- 25318 Java projects (3504681 hierarchies)
- 20010 Python projects (2491603 hierarchies)

Almost 10 millions of inheritance hierarchies!

# Depth VS Size is logarithmic

## Data Analysis

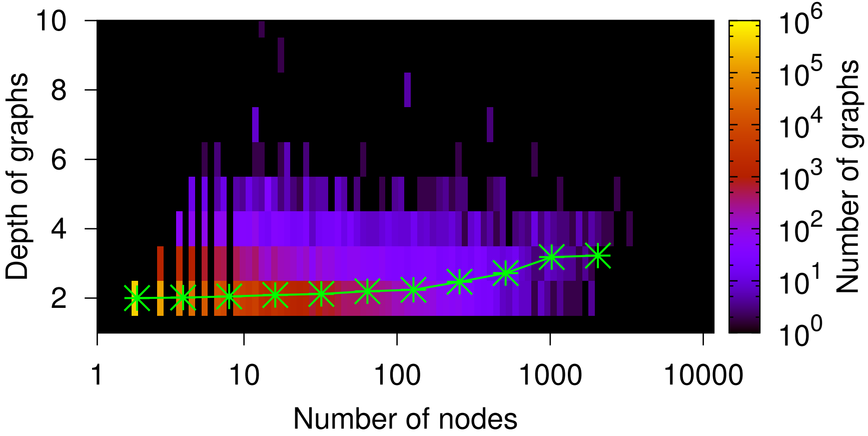
C++



# Depth VS Size is logarithmic

## Data Analysis

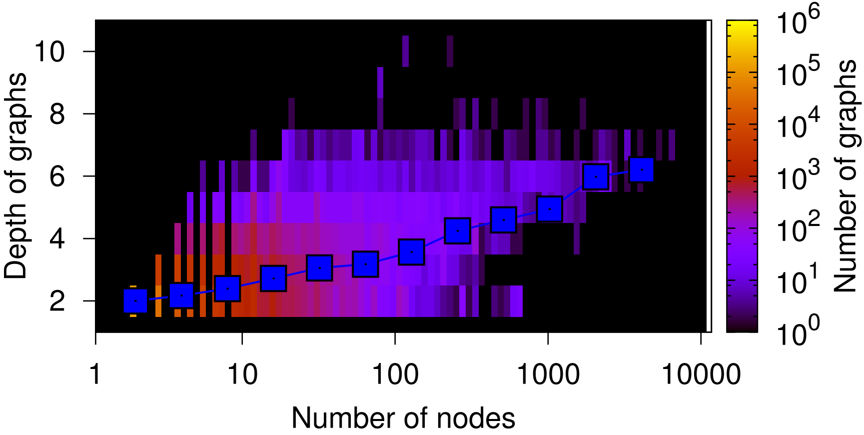
Java



# Depth VS Size is logarithmic

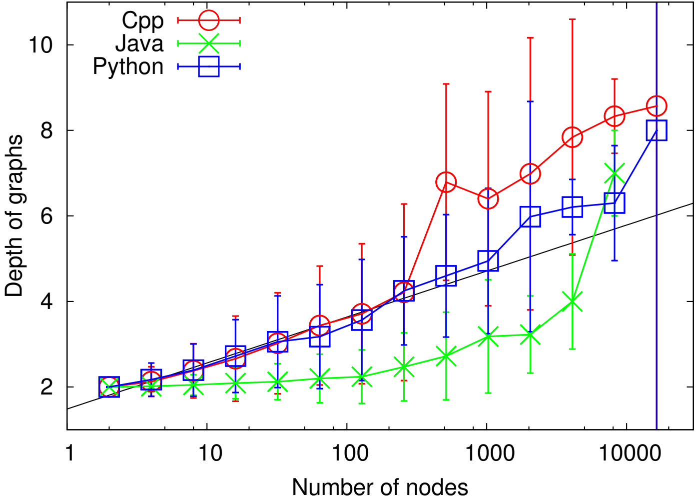
## Data Analysis

Python



# Depth VS Size is logarithmic

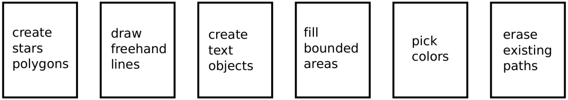
## Data Analysis - Comparison among languages



# Sharing Tree model

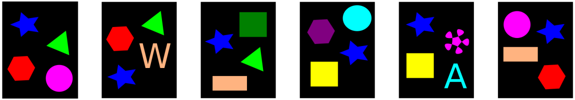
**(A microscopic model)**

# How to build the structure
## Sharing Tree model (microscopic model)



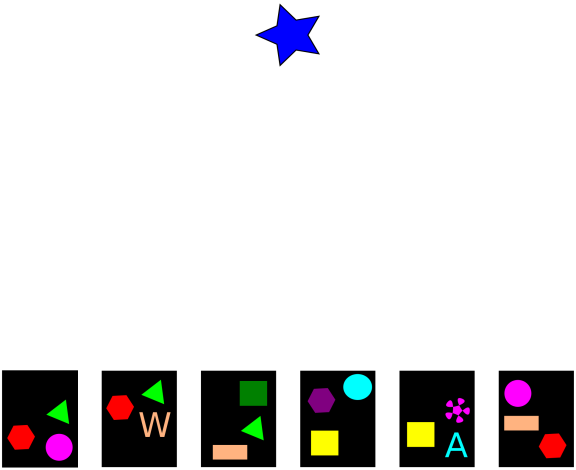
# How to build the structure

## Sharing Tree model (microscopic model)



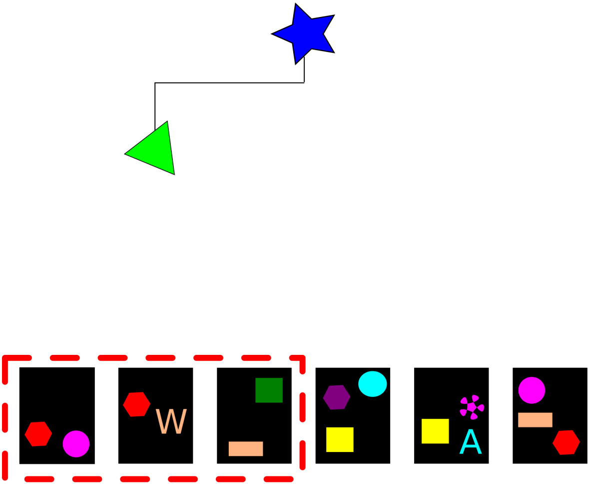
# How to build the structure

## Sharing Tree model (microscopic model)



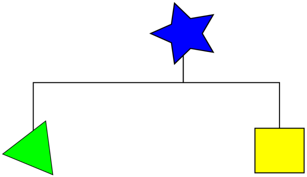
# How to build the structure

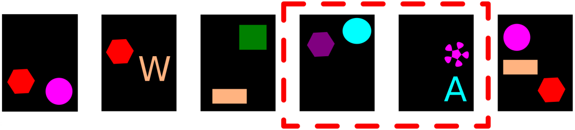
## Sharing Tree model (microscopic model)



# How to build the structure

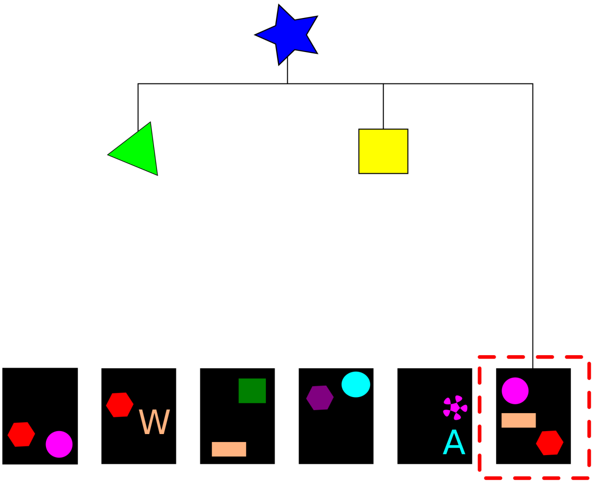
## Sharing Tree model (microscopic model)





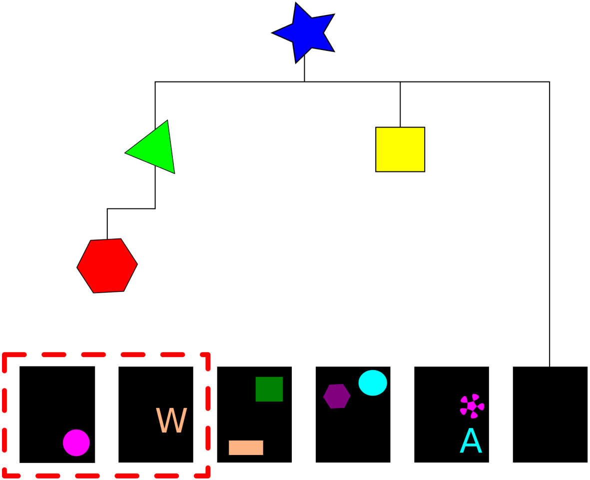
# How to build the structure

## Sharing Tree model (microscopic model)



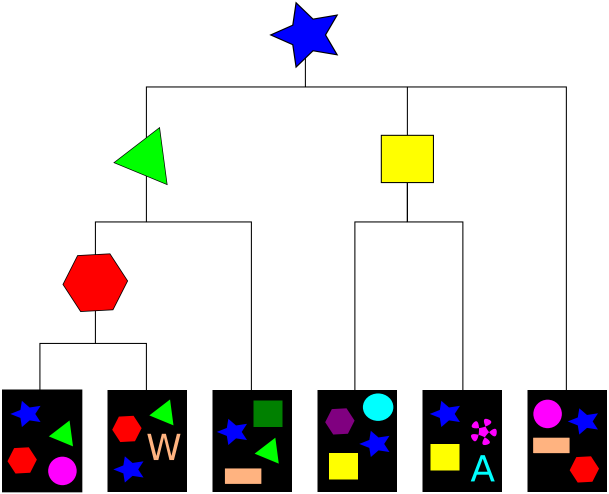
# How to build the structure

## Sharing Tree model (microscopic model)



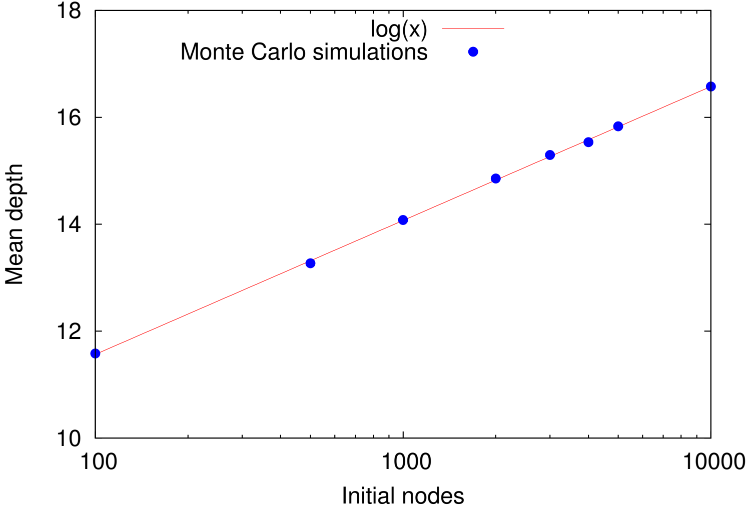
# How to build the structure

## Sharing Tree model (microscopic model)



# Depth VS Size is logarithmic

## Sharing Tree model (microscopic model)



# Minimal Effort model

**(A mean field model)**

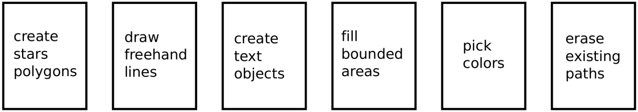
# The Effort to build a hierarchy

## Minimal Effort model (mean field model)

$$\mathtt{E} = \sum_{\sigma}^{\mathcal{N}} \mathsf{cost}(\sigma)$$

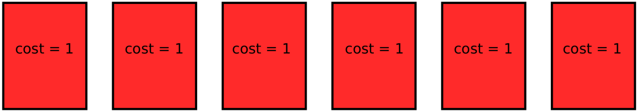
# You need $\mathfrak{n}$ classes to perform a task

## Minimal Effort model (mean field model)



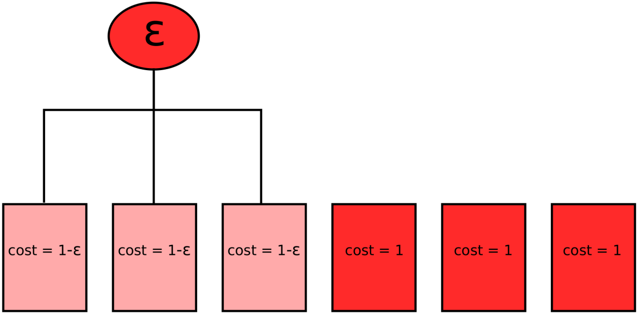
# The cost of each class

## Minimal Effort model (mean field model)



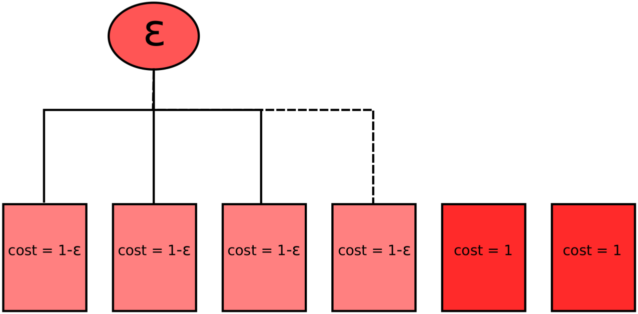
# Reuse

## Minimal Effort model (mean field model)



# Competition

## Minimal Effort model (mean field model)



# Competition

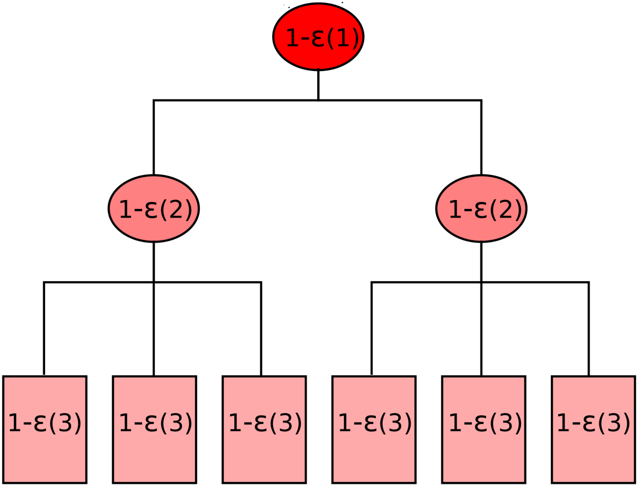
## Minimal Effort model (mean field model)

$$\mathtt{E} = \sum_{\sigma}^{\mathcal{N}} \mathsf{cost}(\sigma)$$

$$\mathtt{E} = \sum_{\sigma}^{\mathcal{N}} [1 - \varepsilon(\mathfrak{m})]$$

# The effort of "writing" a tree

## Minimal Effort model (mean field model)



# How much of the code is shareable?

## Minimal Effort model (mean field model)

The probability to find a selected symbol in a sequence of k extractions is

$$p = 1 - \left(1 - \frac{1}{\mathcal{S}}\right)^{\mathsf{k}}$$

# How much of the code is shareable?

## Minimal Effort model (mean field model)

The probability to find a selected symbol in a sequence of k extractions is

$$p = 1 - \left(1 - \frac{1}{\mathcal{S}}\right)^{\mathsf{k}}$$

$$\mathcal{S} \to \infty \qquad \mathsf{k} \to \infty \qquad \beta \equiv \frac{\mathsf{k}}{\mathcal{S}} \qquad e^{-\beta} = \lim_{\mathcal{S} \to +\infty} \left(1 - \frac{1}{\mathcal{S}}\right)^{\beta \mathcal{S}}$$

# How much of the code is shareable?
## Minimal Effort model (mean field model)

The probability to find a selected symbol in a sequence of k extractions is

$$p = 1 - \left(1 - \frac{1}{\mathcal{S}}\right)^{\mathsf{k}}$$

The probability to find the symbol in $\mathfrak{m}$ independent sets

$$p = 1 - e^{-\beta} \qquad \rightarrow \qquad \rho = \left(1 - e^{-\beta}\right)^{\mathfrak{m}}$$

# How much of the code is shareable?

## Minimal Effort model (mean field model)

The probability to find a selected symbol in a sequence of k extractions is

$$p = 1 - \left(1 - \frac{1}{\mathcal{S}}\right)^{\mathsf{k}}$$

The probability to find the symbol in $\mathfrak{m}$ independent sets

$$p = 1 - e^{-\beta} \qquad \rightarrow \qquad \rho = \left(1 - e^{-\beta}\right)^{\mathfrak{m}}$$

The **shareable code** is

$$\varepsilon(\mathfrak{m}) = \frac{\mathcal{S}}{\mathsf{k}}\left(1 - e^{-\beta}\right)^{\mathfrak{m}} = \frac{1}{\beta}\left(1 - e^{-\beta}\right)^{\mathfrak{m}} \equiv e^{-\alpha\mathfrak{m}}$$

# The shared code

## Minimal Effort model (mean field model)

$$\mathtt{E} = \sum_{\sigma}^{\mathcal{N}} \mathsf{cost}(\sigma)$$

$$\mathtt{E} = \sum_{\sigma}^{\mathcal{N}} [1 - \varepsilon(\mathfrak{m})]$$

$$\boxed{\mathtt{E} = \sum_{\sigma}^{\mathcal{N}} \left[1 - \mathsf{e}^{-\alpha\mathfrak{m}}\right]}$$

# Mean field approach

## Minimal Effort model (mean field model)

The **number of nodes** at each level

$$\{\mathfrak{n}(l)\}_{l=1}^{\mathtt{L}} = \{\mathfrak{n}(1), \mathfrak{n}(2), \ldots, \mathfrak{n}(\mathtt{L}) \equiv 1\}$$

The **mean number of brothers** is

$$\mathfrak{m}(l) = \frac{\mathfrak{n}(l)}{\mathfrak{n}(l+1)}$$

The effort as a sum over levels

$$\mathtt{E}[\mathtt{L}, \{\mathfrak{n}(l)\}] = \sum_{l=1}^{\mathtt{L}-1} \left[1 - \varepsilon\left(\frac{\mathfrak{n}(l)}{\mathfrak{n}(l+1)}\right)\right] \mathfrak{n}(l)$$

# $\mathtt{E}$ as a function of the structure
## Minimal Effort model (mean field model)

$$\mathtt{E} = \sum_{\sigma}^{\mathcal{N}} \mathsf{cost}(\sigma)$$

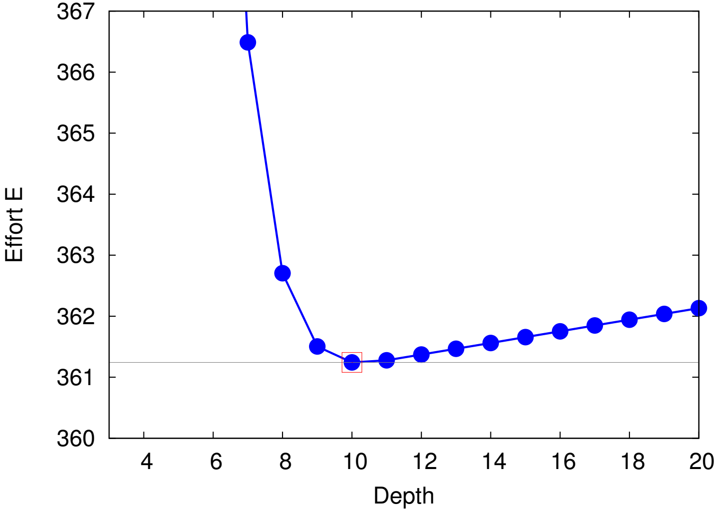
$$\mathtt{E} = \sum_{\sigma}^{\mathcal{N}} [1 - \varepsilon(\mathfrak{m})]$$

$$\mathtt{E} = \sum_{\sigma}^{\mathcal{N}} \left[1 - e^{-\alpha\mathfrak{m}}\right]$$

$$\mathtt{E}[\mathtt{L}, \{\mathfrak{n}(l)\}] = \sum_{l=1}^{\mathtt{L}-1} \left(1 - e^{-\alpha \frac{\mathfrak{n}(l)}{\mathfrak{n}(l+1)}}\right) \mathfrak{n}(l)$$

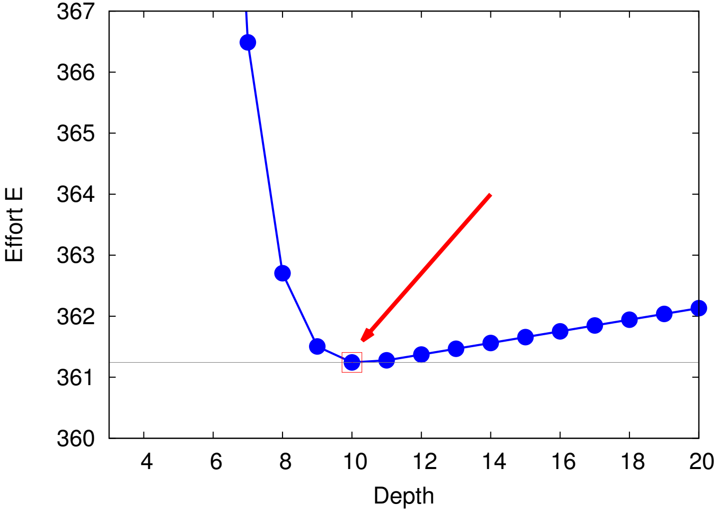
# The functional $\mathtt{E}$

## Minimal Effort model (mean field model)



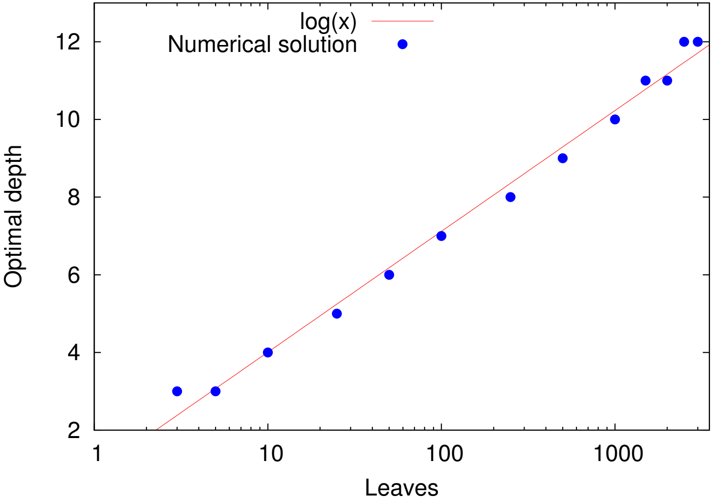
# The functional E

## Minimal Effort model (mean field model)



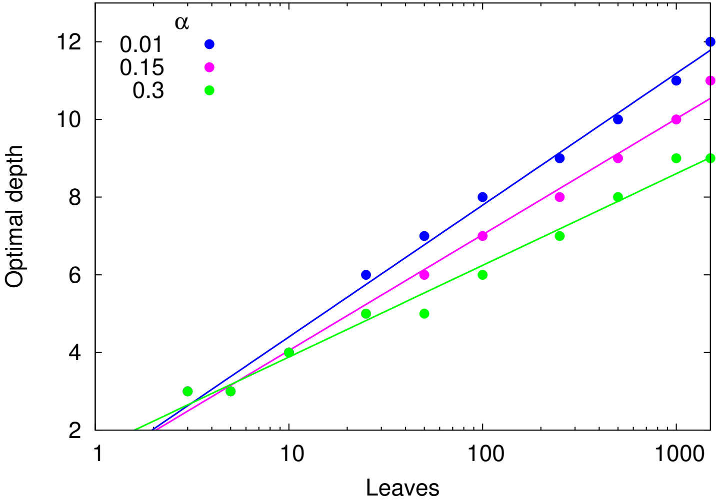
# Depth VS Size is logarithmic

## Minimal Effort model (mean field model)



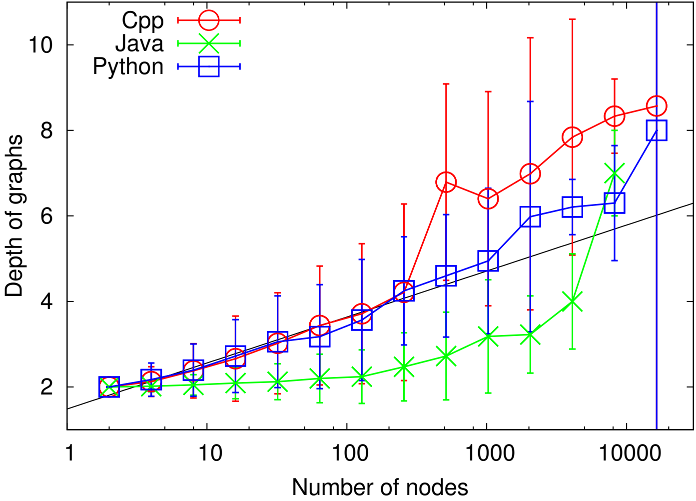
# Shareability of the code

## Minimal Effort model (mean field model)



# Depth VS Size is logarithmic

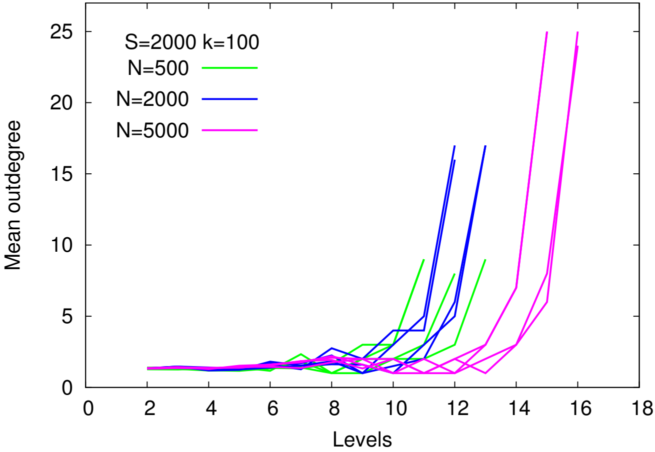
## Data Analysis - Comparison among languages



# Hierarchies structures

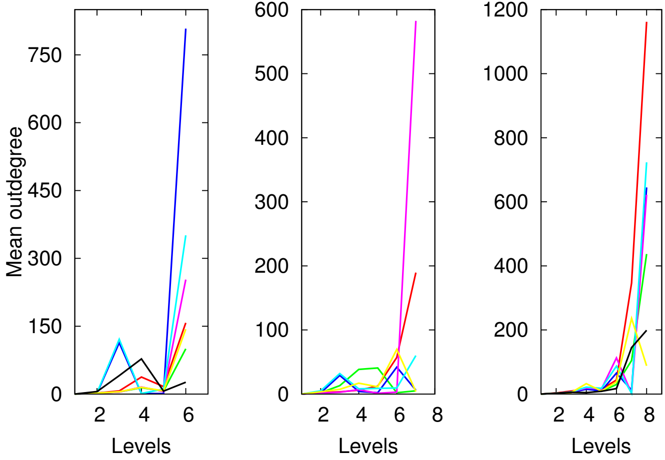
# Mean outdegree grows close to the root

Sharing Tree model



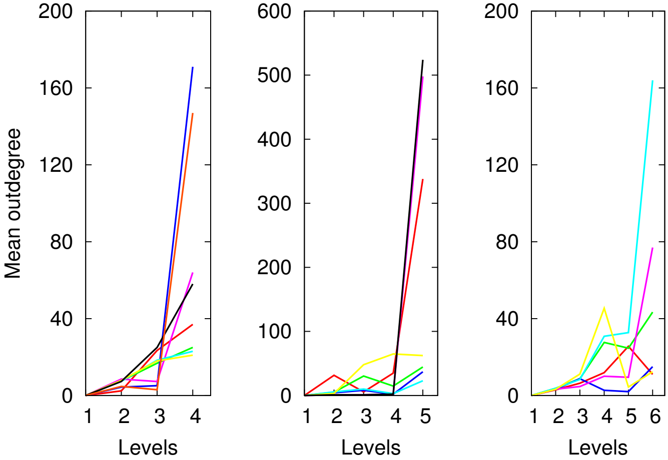
# Mean outdegree grows close to the root

C++



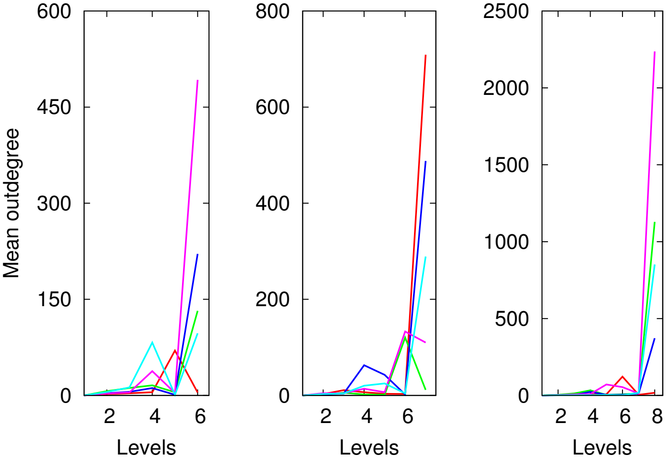
# Mean outdegree grows close to the root

Java



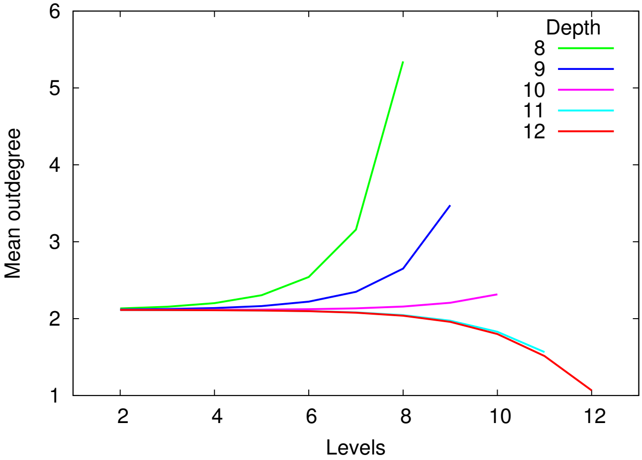
# Mean outdegree grows close to the root

Python



# Is shallow better?

Minimal Effort model



# Conclusions

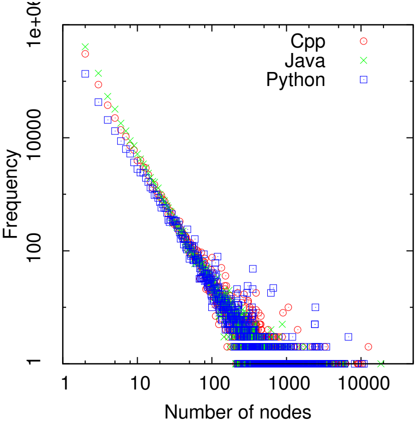
# Conclusions

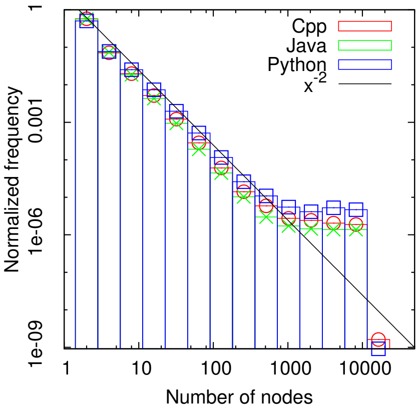
- Different OO programming languages show **common behaviors** (in sizes distribution, outdegree distribution, depth VS size, ...)
- The two different **models** (microscopic and mean field) are compatible
- Hierarchies arise from a mechanism of **competition** between the sake of the reuse and the difficulty of the abstraction
- We have an interpretation about the **shallow hierarchies** in Java
- Both models predict the growth of the **mean outdegree** close to the root
- We argue that depths are **sub-optimal**

# Extras

# Sizes distribution

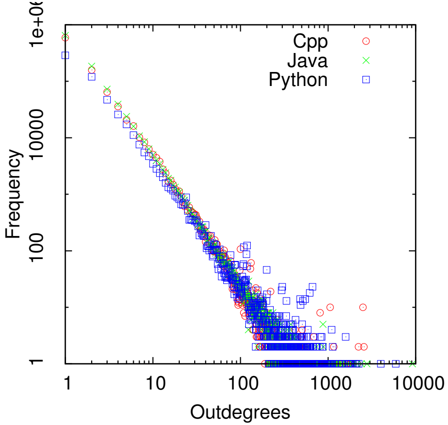
## Extra

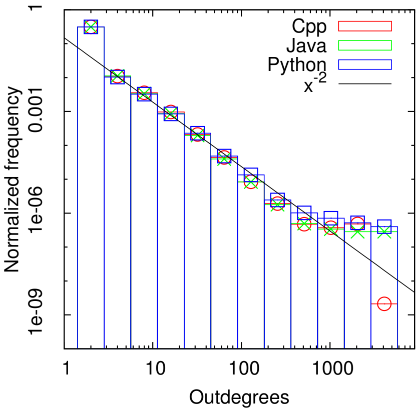


# Outdegrees distribution

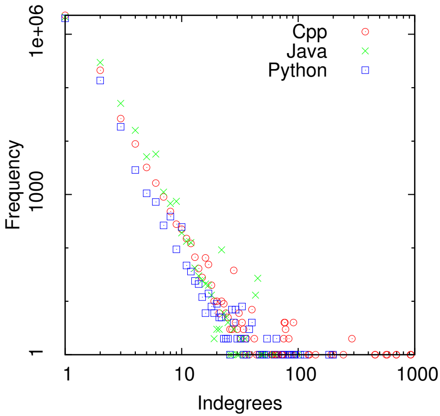
## Extra

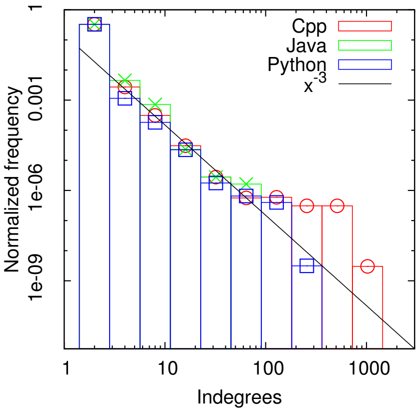


# Indegrees distribution

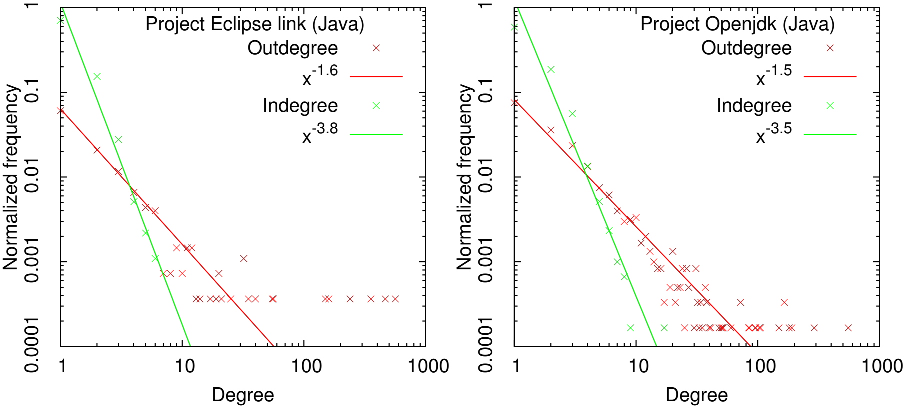
## Extra



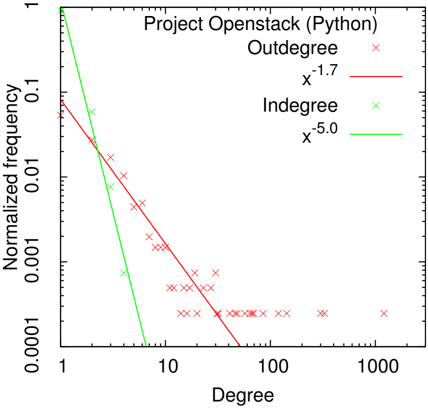
# Tree Approximation - Java

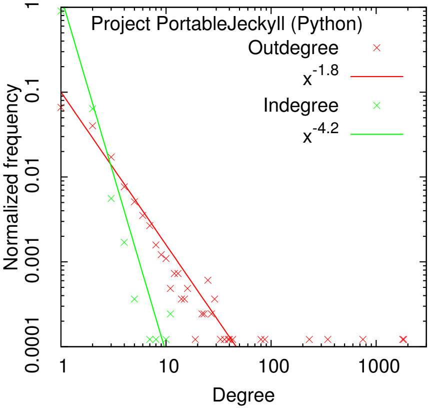
## Extra

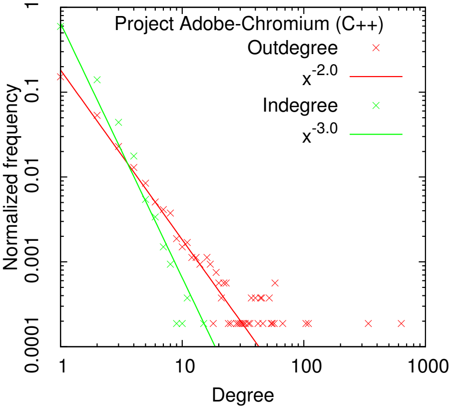


# Tree Approximation - Python

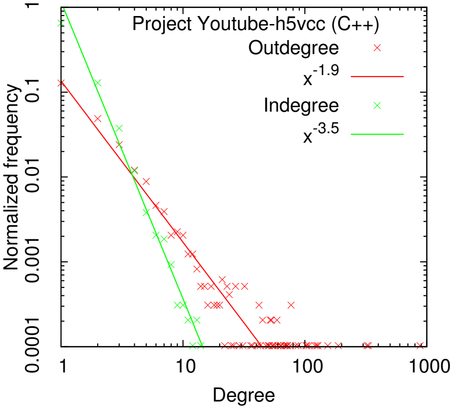
## Extra





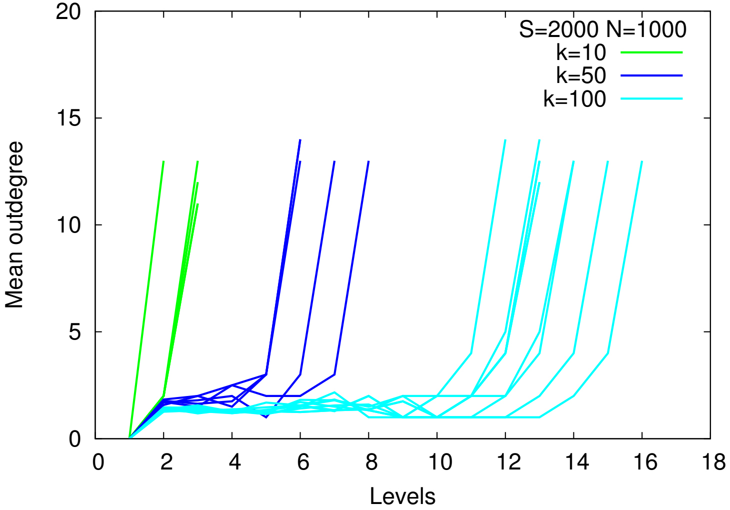
Project Adobe-Chromium (C++)
Outdegree
$x^{-2.0}$
Indegree
$x^{-3.0}$
Normalized frequency
0.0001
0.001
0.01
0.1
1
10
100
1000
Degree

Project Youtube-h5vcc (C++)
Outdegree
$x^{-1.9}$
Indegree
$x^{-3.5}$
Normalized frequency
0.0001
0.001
0.01
0.1
1
10
100
1000
Degree

# Abstractability in Sharing Tree model

## Extra



# Most common symbol in Sharing Tree model 1/3

## Extra

Probability to find a selected symbol in one sequence is

$$p = \frac{\binom{\mathcal{S}-1}{\mathsf{k}-1}}{\binom{\mathcal{S}}{\mathsf{k}}} = \frac{\mathsf{k}!(\mathcal{S}-1)!}{\mathcal{S}!(\mathsf{k}-1)!}$$

Probability that $w$ classes contain a selected symbol with

$$Pr(w) = \binom{\mathcal{N}}{w} p^{w} (1-p)^{\mathcal{N}-w}$$

Each symbol $s \in \mathcal{S}$ appears in $w_s$ classes. The set $\{w_s\}_{s=1}^{\mathcal{S}}$ contains $\mathcal{S}$ IIDRV. The most common symbol is the one that appears $\omega$ times

$$\omega = \max\{w_1, \ldots, w_{\mathcal{S}}\}$$

# Most common symbol in Sharing Tree model 2/3

**Extra**

Consider the cumulative distribution function of $\omega$

$$F_\omega(y) = Pr(\omega \leq y)$$

Since $\omega$ is the maximal occurrence and $w_s$ are independent

$$\begin{aligned} Pr(\omega \leq y) &= Pr(w_1 \leq y, w_2 \leq y, \ldots, w_\mathcal{S} \leq y) \\ &= Pr(w_1 \leq y) Pr(w_2 \leq y) \ldots Pr(w_\mathcal{S} \leq y) \end{aligned}$$

and since all $w_s$ have the same cumulative mass function

$$F_\omega(y) = F_w^\mathcal{S}(y)$$

The probability distribution of $\omega$

$$\begin{aligned} Pr(y = \omega) &= Pr(\omega \leq y) - Pr(\omega \leq y - 1) \\ &= F_w^\mathcal{S}(y) - F_w^\mathcal{S}(y - 1) \end{aligned}$$

# Most common symbol in Sharing Tree model 3/3

**Extra**

Remembering that $w_s$ are binomial random variables, the occurrence $\omega$ of the most common symbol is therefore distributed as

$$\Psi(\omega) = \left(\sum_{i=0}^{\omega} \mathrm{Bin}(\mathcal{N}, i)\right)^{\mathcal{S}} - \left(\sum_{i=0}^{\omega-1} \mathrm{Bin}(\mathcal{N}, i)\right)^{\mathcal{S}}$$

Making explicit the formula of the Binomial distribution

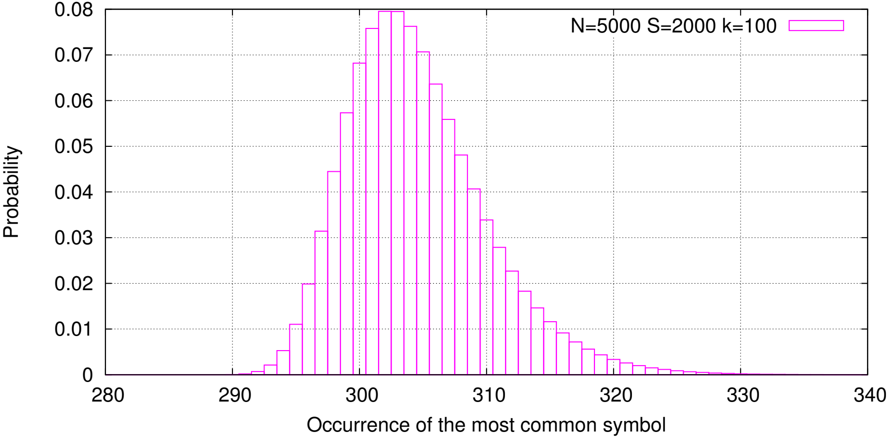
$$\Psi(\omega) = \left(\sum_{i=0}^{\omega} \binom{\mathcal{N}}{i} p^i (1-p)^{\mathcal{N}-i}\right)^{\mathcal{S}} - \left(\sum_{i=0}^{\omega-1} \binom{\mathcal{N}}{i} p^i (1-p)^{\mathcal{N}-i}\right)^{\mathcal{S}}$$

The mean of the distribution is

$$\langle \omega \rangle = \mathcal{N} - \sum_{j=0}^{\mathcal{N}-1} \left(\sum_{k=0}^{j} \binom{N}{k} p^k (1-p)^{\mathcal{N}-k}\right)^{\mathcal{S}}$$

# Most common symbol distribution

## Extra



# Sharing Tree process
## Extra

The mean number of elements of a group at each step $t$ is given by

$$f(x_t, t) = x_t - \sum_{j=0}^{x_t-1} \left( \sum_{i=0}^{j} \binom{x_t}{i} \Pi_t^i (1 - \Pi_t)^{x_t - i} \right)^{\mathcal{S}-t}$$

where $\Pi_t$ is obtained with the hypergeometric distribution and considering that if a symbol has been used as *the most common* then it cannot be reused, and so at each step $\mathcal{S} \to \mathcal{S} - 1$ and $\mathsf{k} \to \mathsf{k} - 1$.

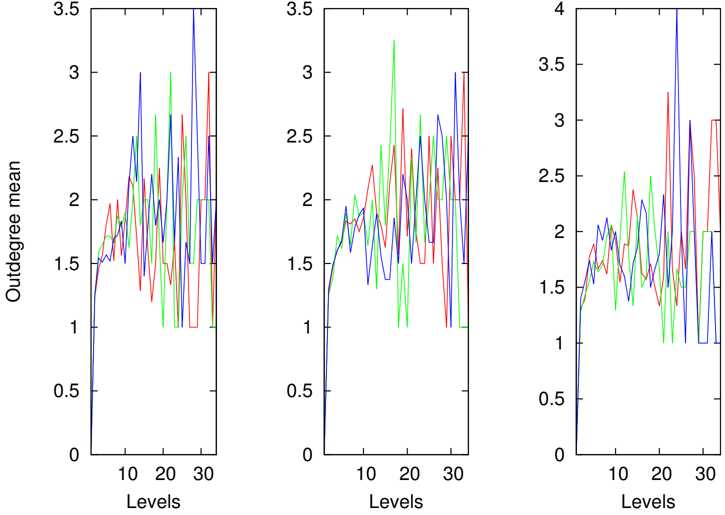
$$\Pi_t = \frac{\binom{\mathcal{S}-1-t}{k-1-t}}{\binom{\mathcal{S}-t}{\mathsf{k}-t}} = \frac{\Gamma(\mathcal{S} - t)\Gamma(\mathsf{k} - t + 1)}{\Gamma(\mathsf{k} - t)\Gamma(\mathcal{S} - t + 1)}$$

The process for the mean number of elements can so be defined as

$$x_{t+1} = f(x_t, t)$$

where $x_0 = x(0)$ and is equal to $\mathcal{N}$ for the main process.

# Null model 1
## Extra



# Null model 2

## Extra