

# Maintaining large software distributions: new challenges from the FOSS era

Roberto Di Cosmo<sup>1</sup> Berke Durak<sup>2</sup> **Xavier Leroy**<sup>2</sup>  
Fabio Mancinelli<sup>1</sup> Jérôme Vouillon<sup>1</sup>

<sup>1</sup>PPS, University of Paris 7

<sup>2</sup>INRIA Rocquencourt

FRCSS'06, 2006-04-01

Maintaining  
distributions

Di Cosmo et al

Distributions  
and EDOS

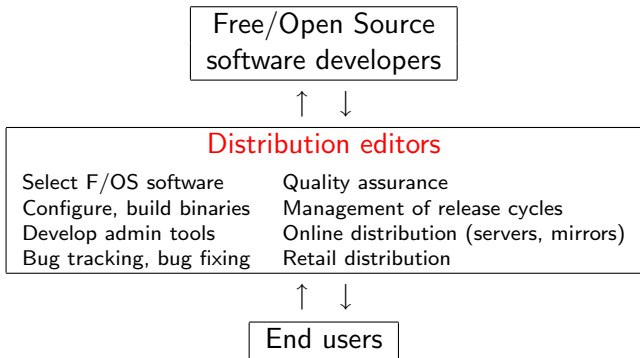
Packages and  
dependencies

Installability

Monotonicity

Conclusions

- 1 Distribution editors and the EDOS project
- 2 Software packages and their dependencies
- 3 The installability problem
- 4 Work in progress: the monotonicity problem
- 5 Conclusions



(Examples: RedHat, Novell, Mandriva, Ubuntu, Debian, Gentoo, FreeBSD, ...)

Objective: develop technology and tools to support and improve the process of editing a F/OSS distribution.

- Dependency management among large, heterogeneous collections of software packages.
- Testing and QA at the distribution scale.
- Efficient distribution of large software systems, using peer-to-peer and distributed database technology.

(FP6-IST STREP: Caixa Mágica, CSP Torino, Edge-IT, INRIA, Mandriva, Nexedi, Nuxeo, Tel Aviv U., U. Geneva, U. Paris 7, Zurich U.)

## Maintaining distributions

Di Cosmo et al

Distributions and EDOS

Packages and dependencies

Installability

Monotonicity

Conclusions

- 1 Distribution editors and the EDOS project
- 2 Software packages and their dependencies**
- 3 The installability problem
- 4 Work in progress: the monotonicity problem
- 5 Conclusions

Distributions are composed of many **packages** (10000).

Examples of packages: executable programs, shared libraries, network services, plug-ins, APIs for programmers, documentation, localization data, source code, . . .

Issue: installing an application can require dozens of additional packages.

→ automatic installers

→ exploiting **dependencies** stated in package metadata.

```
Package: binutils
Priority: standard
Section: devel
Installed-Size: 6004
Maintainer: James Troup <james@nocrew.org>
Architecture: i386
Version: 2.16.1-2
Provides: elf-binutils
Depends: libc6 (>= 2.3.2.ds1-21)
Suggests: binutils-doc (= 2.16.1-2)
Conflicts: gas, modutils (<< 2.4.19-1)
Filename: pool/main/b/binutils/binutils_2.16.1-2_i386.deb
Size: 2377880
MD5sum: 37a46d934443c096e217aec8b2a2e303
Description: The GNU assembler, linker and binary utilities
 The programs in this package are used to assemble, link and
 manipulate binary and object files. They may be used in conjunction
 with a compiler and various libraries to build programs.
Build-Essential: yes
Tag: devel::machinecode, interface::commandline, role::sw:shlib
```

Many package formats in existence, but have in common the following three kinds of dependencies:

- **Requires:**  
Other packages and features required by this package.  
Includes ranges of version numbers and alternatives.

## Example

```
netbase 4.24:
  ifupdown (>= 0.6.4), tcpd, iputils-ping | ping
```

- **Conflicts:**  
Packages that cannot co-exist with this package.
- **Provides:**  
Features provided by this package.

Many package formats in existence, but have in common the following three kinds of dependencies:

- **Requires:**  
Other packages and features required by this package.  
Includes ranges of version numbers and alternatives.
- **Conflicts:**  
Packages that cannot co-exist with this package.

## Example

```
proftpd 1.2.10-27:
    wu-ftp, ftp-server, proftpd-mysql
```

- **Provides:**  
Features provided by this package.

Many package formats in existence, but have in common the following three kinds of dependencies:

- **Requires:**  
Other packages and features required by this package.  
Includes ranges of version numbers and alternatives.
- **Conflicts:**  
Packages that cannot co-exist with this package.
- **Provides:**  
Features provided by this package.

## Example

```
apache 1.3.34-2:
  httpd
```



Package	$P ::= (name, version)$
Repository	$R ::= (\{P_1 \dots P_n\}, D, C)$
Dependencies	$D ::= P \rightarrow (P \vee \dots \vee P) \wedge \dots \wedge (P \vee \dots \vee P)$
Conflicts	$C ::= \{P \# P, \dots\}$

Features, as well as constraints on version numbers, are expanded as disjunctions.

An **installation** of a repository  $(\{P_1 \dots P_n\}, D, C)$  is a set  $I \subseteq \{P_1 \dots P_n\}$  of packages where all dependencies are satisfied and no conflict occurs.

## Maintaining distributions

Di Cosmo et al

Distributions and EDOS

Packages and dependencies

Installability

Monotonicity

Conclusions

- 1 Distribution editors and the EDOS project
- 2 Software packages and their dependencies
- 3 The installability problem**
- 4 Work in progress: the monotonicity problem
- 5 Conclusions

Maintaining  
distributions

Di Cosmo et al

Distributions  
and EDOS

Packages and  
dependencies

Installability

Monotonicity

Conclusions

A package  $P$  is **installable** if there exists an installation  $I$  with  $P \in I$ .

A set of packages  $P$  is **co-installable** if there exists an installation  $I$  with  $P \subseteq I$ .

A distribution is **trimmed** if all packages it contains are installable.

installability  $\longleftrightarrow$  SAT

The reduction  $\leftarrow$  shows that installability is an NP-complete problem.

The reduction  $\rightarrow$  gives an efficient algorithm to decide installability:

- 1 Translate the repository to a boolean formula  $F$  (one boolean variable per package).
- 2 Find solutions to  $P \wedge F$  using any existing SAT-solver.
- 3 Solutions correspond 1-to-1 with installations of  $P$ .

installability  $\begin{matrix} \longrightarrow \\ \longleftarrow \end{matrix}$  SAT

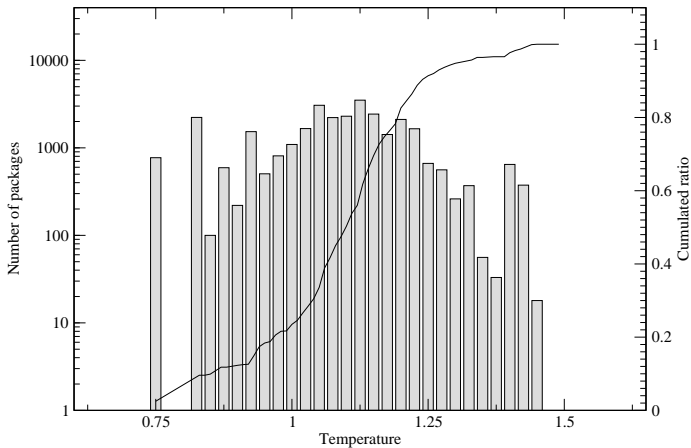
The reduction  $\leftarrow$  shows that installability is an NP-complete problem.

The reduction  $\rightarrow$  gives an efficient algorithm to decide installability:

- 1 Translate the repository to a boolean formula  $F$  (one boolean variable per package).
- 2 Find solutions to  $P \wedge F$  using any existing SAT-solver.
- 3 Solutions correspond 1-to-1 with installations of  $P$ .

# Why is this algorithm efficient in practice?

The generated SAT problems are easy.  
(Hard SAT problems have temperature close to 4.2.)



**rpmcheck** and **debcheck**: check that repositories are trimmed;  
report broken (non-installable) packages.

Repository	# packages	# broken	checking time
Mandriva 2006	4211	42	8 s
Debian snapshot	34701	123	43 s

## Maintaining distributions

Di Cosmo et al

Distributions and EDOS

Packages and dependencies

Installability

Monotonicity

Conclusions

- 1 Distribution editors and the EDOS project
- 2 Software packages and their dependencies
- 3 The installability problem
- 4 Work in progress: the monotonicity problem**
- 5 Conclusions

Repositories evolve over time:

- new versions of packages are added;
- old versions are deprecated;
- wrong dependencies are fixed.

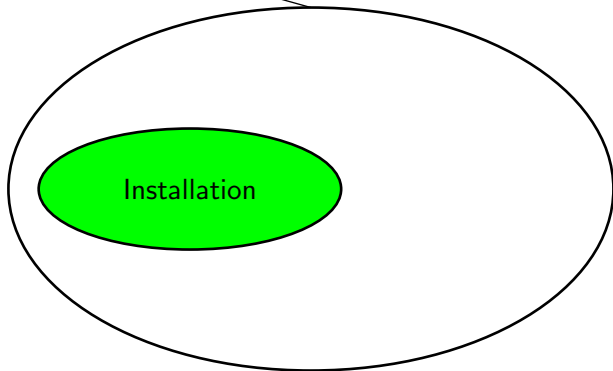
Objective: make sure that these evolutions do not harm users.

## Example

The **monotonicity** problem.

For all installations  $I_1$  of the old repository, there exists an installation  $I_2$  of the new repository that provide all the functionalities of  $I_1$ .

Repository at  $T = 0$



Maintaining  
distributions

Di Cosmo et al

Distributions  
and EDOS

Packages and  
dependencies

Installability

**Monotonicity**

Conclusions

# Monotonicity, graphically

Maintaining distributions

Di Cosmo et al

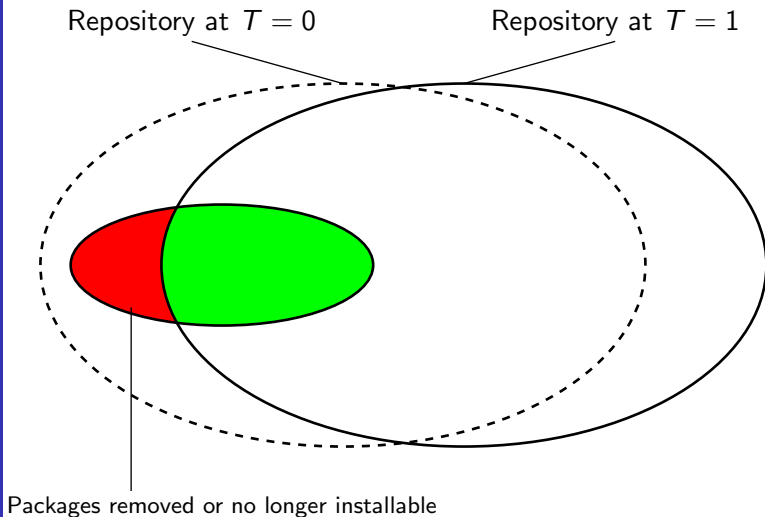
Distributions and EDOS

Packages and dependencies

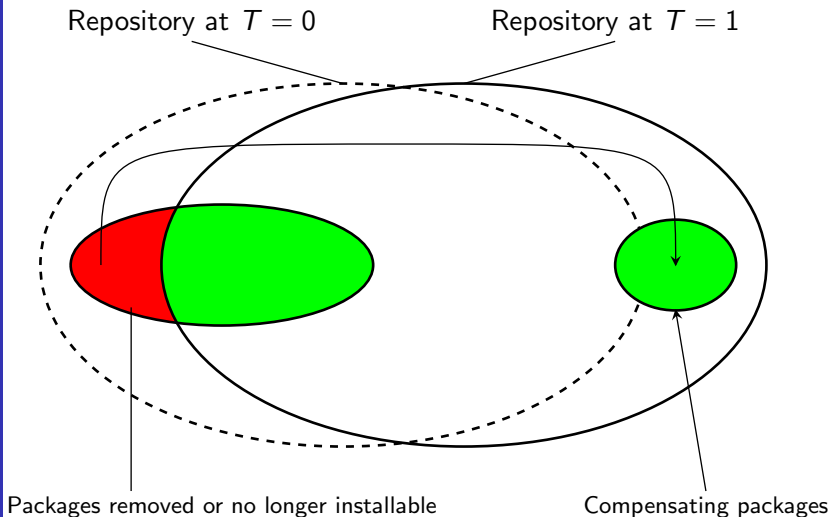
Installability

Monotonicity

Conclusions



# Monotonicity, graphically



## Maintaining distributions

Di Cosmo et al

Distributions and EDOS

Packages and dependencies

Installability

Monotonicity

Conclusions

- 1 Distribution editors and the EDOS project
- 2 Software packages and their dependencies
- 3 The installability problem
- 4 Work in progress: the monotonicity problem
- 5 Conclusions

F/OSS distributions raise new problems of the “software infrastructure” kind.

Distribution-wide analysis of package dependencies is one such problem, surprisingly hard from an algorithmic standpoint.

As often, a formal approach helps.

For more information: <http://www.edoss-project.org>