Software Dependency Management with Boolean Modelling: some Use Cases

Daniel Le Berre

Joint work with Emmanuel Lonca, Anne Parrain, Pascal Rapicault

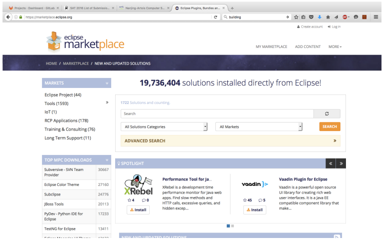
CNRS - Université d'Artois

Nanjing-Artois Computer Science Seminar, June 6, 2016

Outline

Question to the audience

What makes your favorite Linux package repository or Eclipse marketplace different from your favorite smartphone “app store” ?



Projects · Dashboard · GitLab
SAT 2016 List of Submissio...
Nanjing-Artois Computer S...
Eclipse Plugins, Bundles an...
https://marketplace.eclipse.org
building
Create account
Log in
eclipse marketplace
MY MARKETPLACE
ADD CONTENT
MORE
HOME / MARKETPLACE / NEW AND UPDATED SOLUTIONS

19,736,404 solutions installed directly from Eclipse!

MARKETS

- Eclipse Project (44)
- Tools (1593)
- IoT (1)
- RCP Applications (178)
- Training & Consulting (76)
- Long Term Support (11)

1722 Solutions and counting.

Search
All Solutions Categories
All Markets
SEARCH
ADVANCED SEARCH

TOP MPC DOWNLOADS

Subversive - SVN Team Provider	30667
Eclipse Color Theme	27160
Subclipse	24776
JBoss Tools	20113
PyDev - Python IDE for Eclipse	17233
TestNG for Eclipse	13411

SPOTLIGHT

Performance Tool for Ja...
XRebel is a development time performance monitor for Java web apps. Find slow methods and HTTP calls, excessive queries, and hidden excep...
☆ 4 💬 0 Install

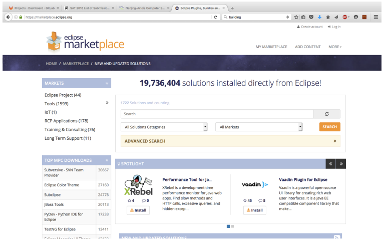
Vaadin Plugin for Eclipse
Vaadin is a powerful open source UI library for creating rich web user interfaces. It is a Java EE compatible component library that make...
☆ 45 💬 5 Install

ANDROID APP ON Google play

Available on the App Store

Question to the audience

What makes your favorite Linux package repository or Eclipse marketplace different from your favorite smartphone “app store” ?



Projects · Dashboard · GitLab | SAT 2016 List of Submissio... | Nanjing-Artois Computer S... | Eclipse Plugins, Bundles an...

https://marketplace.eclipse.org

eclipse marketplace

MY MARKETPLACE | ADD CONTENT | MORE

HOME / MARKETPLACE / NEW AND UPDATED SOLUTIONS

19,736,404 solutions installed directly from Eclipse!

MARKETS
- Eclipse Project (44)
- Tools (1593)
- IoT (1)
- RCP Applications (178)
- Training & Consulting (76)
- Long Term Support (11)

1722 Solutions and counting.

Search

All Solutions Categories | All Markets | SEARCH

ADVANCED SEARCH

TOP MPC DOWNLOADS	
Subversive - SVN Team Provider	30667
Eclipse Color Theme	27160
Subclipse	24776
JBoss Tools	20113
PyDev - Python IDE for Eclipse	17233
TestNG for Eclipse	13411

SPOTLIGHT

Performance Tool for Ja...
XRebel is a development time performance monitor for Java web apps. Find slow methods and HTTP calls, excessive queries, and hidden excep...
☆ 4 0 Install

Vaadin Plugin for Eclipse
Vaadin is a powerful open source UI library for creating rich web user interfaces. It is a Java EE compatible component library that make...
☆ 45 5 Install

The former manage their component dependencies, not the latter !

An example of Ubuntu metadata

```
Package: firefox-3.0
Priority: optional
Section: web
Installed-Size: 3456
Maintainer: Alexander Sack <asac@ubuntu.com>
Architecture: i386
Version: 3.0.16+nobinonly-0ubuntu0.9.04.1
Replaces: firefox (<< 3), firefox-granparadiso, firefox-libthai, firefox-trunk
Provides: firefox-libthai, www-browser
Depends: fontconfig, psmisc, debianutils (>= 1.16), xulrunner-1.9 (>= 1.9.0.1),
     libatk1.0-0 (>= 1.20.0), libc6 (>= 2.4), libcairo2 (>= 1.2.4),
     libfontconfig1 (>= 2.4.0), libfreetype6 (>= 2.2.1), libgcc1 (>= 1:4.1.1),
     libglib2.0-0 (>= 2.16.0), libgtk2.0-0 (>= 2.16.0),
     libnspr4-0d (>= 4.7.3-0ubuntu1~), libpango1.0-0 (>= 1.14.0),
     libstdc++6 (>= 4.1.1),
     firefox-3.0-branding (>= 3.0.3+nobinonly-0ubuntu1~) |
          abrowser-3.0-branding (>= 3.0.3+nobinonly-0ubuntu1~)
Suggests: ubufox,
     firefox-3.0-gnome-support (= 3.0.16+nobinonly-0ubuntu0.9.04.1),
     latex-xft-fonts, libthai0
Conflicts: firefox (<< 3), firefox-granparadiso (<< 3.0~alpha8-0),
     firefox-libthai, firefox-trunk (<< 3.0~a8~cvs20070914t1713-0)
Size: 887822
Description: safe and easy web browser from Mozilla
 Firefox delivers safe, easy web browsing. A familiar user interface,
 enhanced security features including protection from online identity theft,
 and integrated search let you get the most out of the web.
 .
 Install this firefox package too, if you want to be automatically upgraded to
 new major firefox versions in the future.
```

Dependency Management Problem : formal definition

P a set of packages

$P = \{mpm_1, p2cudf_1, p2cudf_2, aspcud_1, aspcud_2, rpm_1, debian_1\}$

depends $P \to 2^{2^P}$ requirement constraints

$P = \{mpm_1 \to \{\{p2cudf_1, p2cudf_2, aspcud_1, aspcud_2\}, \{rpm_1, debian_1\}\}\}$

conflicts $P \to 2^P$impossible configurations

$P = \{p2cudf_1 \to \{p2cudf_2, aspcud_1, aspcud_2\},$
$p2cudf_2 \to \{p2cudf_1, aspcud_1, aspcud_2\}\}$

Definition (consistency of a set of packages)

$Q \subseteq P$ is consistent with $(P, depends, conflicts)$ iff
$\forall q \in Q, (\forall dep \in depends(q), dep \cap Q \neq \emptyset) \wedge (conflicts(q) \cap Q = \emptyset)$.

$Q_1 = \{mpm_1, p2cudf_2, debian_1\}$
$Q_2 = \{mpm_1, aspcud_1, aspcud_2, rpm_1\}$

Dependency Management Problem : formal definition

P a set of packages

$$P = \{mpm_1, p2cudf_1, p2cudf_2, aspcud_1, aspcud_2, rpm_1, debian_1\}$$

depends $P \to 2^{2^P}$ requirement constraints

$$P = \{mpm_1 \to \{\{p2cudf_1, p2cudf_2, aspcud_1, aspcud_2\}, \{rpm_1, debian_1\}\}\}$$

conflicts $P \to 2^P$impossible configurations

$P = \{p2cudf_1 \to \{p2cudf_2, aspcud_1, aspcud_2\},$
$p2cudf_2 \to \{p2cudf_1, aspcud_1, aspcud_2\}\}$

Definition (consistency of a set of packages)

$Q \subseteq P$ is consistent with $(P, depends, conflicts)$ iff
$\forall q \in Q, (\forall dep \in depends(q), dep \cap Q \neq \emptyset) \wedge (conflicts(q) \cap Q = \emptyset)$.

What is the complexity of finding if a Q containing a specific package exists ?

Just as hard as SAT : NP-complete !

See how to decide satisfiability of $(\neg a \vee b \vee c) \wedge (\neg a \vee \neg b \vee c) \wedge a \wedge \neg c$

```
package: a
version: 1
conflicts: a = 2

package: a
version: 2
conflicts: a = 1

package: b
version: 1
conflicts: b = 2

package: b
version: 2
conflicts: b = 1

package: c
version: 1
conflicts: c = 2

package: c
version: 2
conflicts: c = 1
```

```
package: clause
version: 1
depends: a = 2 | b = 1 | c = 1

package: clause
version: 2
depends: a = 2 | b = 2 | c = 1

package: clause
version: 3
depends: a = 1

package: clause
version: 4
depends: c = 2

package: formula
version: 1
depends: clause = 1, clause = 2,
         clause = 3, clause = 4

request: satisfiability
install: formula
```

Dependencies expressed by clauses

- Dependencies can easily be translated into clauses :

```
package:  a
version:  1
depends:  b = 2 | b = 1, c = 1
```

$$a_1 \rightarrow (b_2 \vee b_1) \wedge c_1$$

$$\neg a_1 \vee b_2 \vee b_1, \neg a_1 \vee c_1$$

- Conflict can easily be translated into binary clauses :

```
package:  a
version:  1
conflicts:  b = 2, d = 1
```

$$\neg a_1 \vee \neg b_2, \neg a_1 \vee \neg d_1$$

From decision to optimization

- NP-complete, so we can/should use a SAT solver to solve it
- Finding a solution is usually not sufficient !
  - Minimizing the number of installed packages
  - Minimizing the size of installed packages
  - Keeping up to date versions of packages
  - Preferring most recent packages to older ones
  - ...
- In practice an aggregation of various criteria
- Need a more expressive representation language than plain CNF !

From decision to optimization

- NP-complete, so we can/should use a SAT solver to solve it
- Finding a solution is usually not sufficient !
  - Minimizing the number of installed packages
  - Minimizing the size of installed packages
  - Keeping up to date versions of packages
  - Preferring most recent packages to older ones
  - ...
- In practice an aggregation of various criteria
- Need a more expressive representation language than plain CNF !
- Use of MAXSAT or Pseudo-Boolean optimization solvers

UNIVERSITÉ D'ARTOIS

How did we get interested in such problem ?

- 1999 First study in the ASP community for Debian package management, with proof of NP-completeness, and ASP-based package manager [Syrjänen, 1999]
- 2005 Generic result for Linux package managers (Debian, RPM) during EDOS European Project [Team, 2005]
- 2006 Contacted by Chris Tucker about Sat4j while he was working on Opium dependency manager for Linspire Linux distribution [Tucker et al., 2007] : Sat4j not efficient enough.
- 2007 Contacted by Pascal Rapicault from IBM Rationale while he was working on p2 dependency manager for Eclipse.
- 2008 Mancoosi European Project [Team, 2008] : follow up to the EDOS project
  competition of package managers with user defined multi-objective (lexicographic) optimization : p2cudf.

Outline

The specific case of the Eclipse platform

- Open platform developed by the Eclipse foundation
- Designed for extensibility : the basic platform is enriched with plugins
- Widely adopted (more than 13M downloads for Eclipse 3.4 and 3.5)
- Many vendors ship products on top of Eclipse
- Plugins are usually coming from various, uncontrolled sources (main difference compared to Linux case).

Example of metadata

```
id=org.eclipse.swt, version=3.5.0, singleton=true
Capabilities:
    {namespace=org.eclipse.equinox.p2.iu,
     name=org.eclipse.swt, version=3.5.0}
    {namespace=org.eclipse.equinox.p2.eclipse.type
     name=bundle version=1.0.0}
    {namespace=java.package,
     name=org.eclipse.swt.graphics, version=1.0.0}
    {namespace=java.package,
     name=org.eclipse.swt.layout, version=1.2.0}
Requirements:
    {namespace=java.package,
     name=org.eclipse.swt.accessibility2,
     range=[1.0.0,2.0.0), optional=true, filter=(&(os=linux))}
    {namespace=java.package, name=org.mozilla.xpcom,
     range=[1.0.0, 1.1.0), optional=true, greed=false}
Updates:
    {namespace=org.eclipse.equinox.p2.iu, name=org.eclipse.swt,
                  range=[0.0.0, 3.5.0)}
```

Some specific notions in Eclipse

optional dependencies	must be satisfied as much as possible. Used for the drop in folder.
greedy dependencies	Non greedy dependencies do not force the installation of plugins. Used for platform specific dependencies for instance.
patches	allow to change the dependencies during provisioning. Specifically used when shipping a product based on Eclipse.

Example of installable unit patch for the Platform group

```
id=org.eclipse.ant.critical.fix,version=1.0.0
Capabilities:
    {namespace=org.eclipse.equinox.p2.iu,
     name=org.eclipse.ant.core.critical.fix, version=3.5.0.v2009}
Requirement Changes:
    { from={namespace=org.eclipse.equinox.p2.iu,
            name=org.eclipse.ant.core, range=[3.1.0, 3.4.0)},
      to={namespace=org.eclipse.equinox.p2.iu,
          name=org.eclipse.ant.core, range=[3.4.3]}}
    { from={namespace=org.eclipse.equinox.p2.iu,
            name=org.apache.ant, range=[1.7.1.v2009]},
      to={namespace=org.eclipse.equinox.p2.iu,
          name=org.apache.ant, range=[1.7.2.v2009]}}
Applicability Scope:
    {namespace=org.eclipse.equinox.p2.iu,
     name=org.eclipse.platform.feature.group, range=[3.5.0.v2009]}
Lifecycle:
    {namespace=org.eclipse.equinox.p2.iu, name=my.product,
     range=[1.0.0], greed=false}
Updates:
    {namespace=org.eclipse.equinox.p2.iu,
     name=org.eclipse.platform.feature.group, range=[0.0.0, 3.5.0.v2009)}
```

UNIVERSITÉ D'ARTOIS

Encoding optional dependencies

An optional requirement is just a soft clause in MAXSAT terminology !

Using selector variables and a linear optimization function :

$$f(IU_i^k) = \bigwedge_{cap_j \in optReq(IU_i^k),} (IU_i^k \rightarrow Noop_z \vee \bigvee_{IU_x^v \in alt(cap_j)} IU_x^v) \quad (1)$$

$$minimize \sum Noop_z \quad (2)$$

From decision to optimization

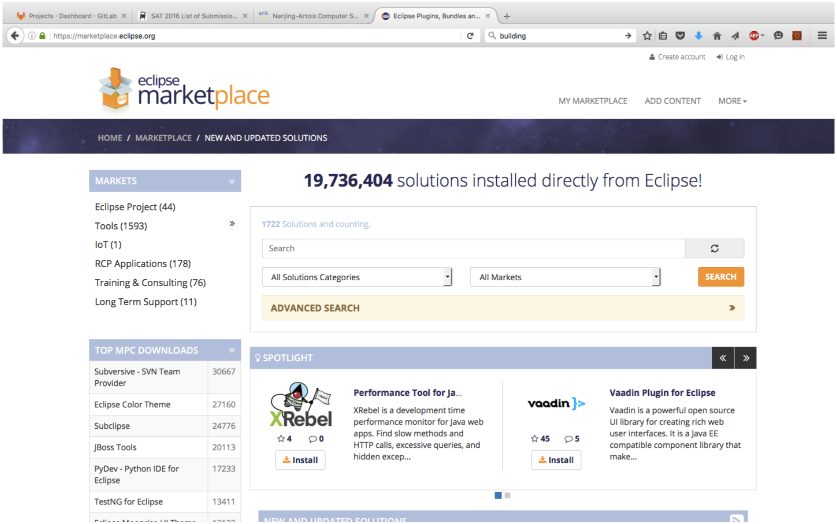
Eclipse dependency problem is under-constrained : it admits lots of solutions. But they are not of equal quality :

1. An IU should not be installed if there is no dependency to it.
2. If several versions of the same bundle exist, the latest one should preferably be used.
3. When optional requirements exist, the optional requirements should be satisfied as much as possible.
4. User installed patches should be applied independently of the consequences of its application (i.e., the version of the IUs forced, the number of installable optional dependencies, etc.).
5. Updating an existing installation should not change packages unrelated from the request being made.

Some thoughts about that experience

- Got lucky to have the ability to work on that problem, because Eclipse needed :
  - An open source solver
  - In Java
  - With the right license
- No issues in modeling constraints (but optionality)
- Main issues met have been :
  - Real integration with Sat4j needed (explanation support, 3.5)
  - Building a better objective function (patches, stability of installations 3.5-3.6)
  - Understanding subtleties like patches and non greedy use cases.
- Main reward is that it works routinely every day since June 2008 !

Sat4j/p2 allowed the creation of the marketplace in 2011



Projects · Dashboard · GitLab | SAT 2016 List of Submissio... | Nanjing-Artois Computer S... | Eclipse Plugins, Bundles an...

https://marketplace.eclipse.org

Create account Log in

eclipse marketplace

MY MARKETPLACE ADD CONTENT MORE

HOME / MARKETPLACE / NEW AND UPDATED SOLUTIONS

MARKETS

- Eclipse Project (44)
- Tools (1593)
- IoT (1)
- RCP Applications (178)
- Training & Consulting (76)
- Long Term Support (11)

19,736,404 solutions installed directly from Eclipse!

1722 Solutions and counting.

Search

All Solutions Categories | All Markets | SEARCH

ADVANCED SEARCH

TOP MPC DOWNLOADS

Subversive - SVN Team Provider	30667
Eclipse Color Theme	27160
Subclipse	24776
JBoss Tools	20113
PyDev - Python IDE for Eclipse	17233
TestNG for Eclipse	13411

SPOTLIGHT

Performance Tool for Ja...

XRebel is a development time performance monitor for Java web apps. Find slow methods and HTTP calls, excessive queries, and hidden excep...

☆ 4 0 Install

Vaadin Plugin for Eclipse

Vaadin is a powerful open source UI library for creating rich web user interfaces. It is a Java EE compatible component library that make...

☆ 45 5 Install

Outline

Motivation for p2cudf

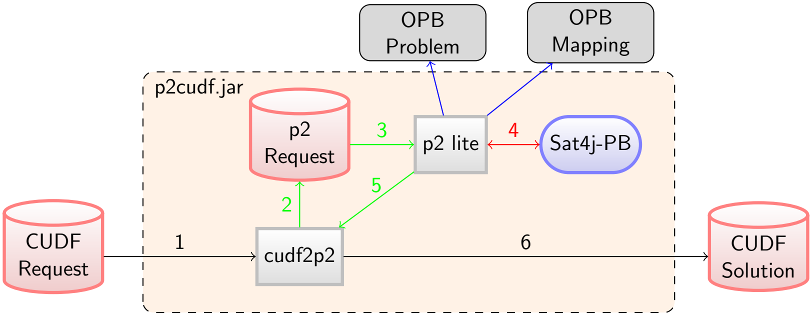
- Reuse existing knowledge about dependency management gathered from Eclipse problems to solve Linux problems
- Use Linux problems to see if p2 approach scales well
- Participate to Mancoosi internal/International Solver Competitions

Main differences with Eclipse :

- Simpler metadata (no patches, groups, etc) $\Rightarrow$ simplified Eclipse 3.5 p2 code base to avoid unnecessary abstractions
- Specific optimization function : lexicographic order of basic, counting-based, criterion to be either maximized or minimized

Initial p2cudf architecture (MISC 2010)

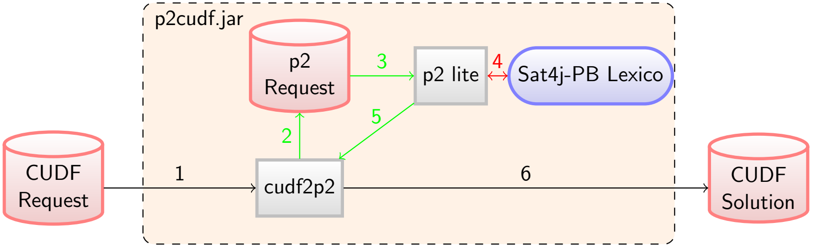
Translate the initial problem into an OPB problem (lexico-optimization translated into a single optimization function) through p2 software.



Can easily swap the underlying boolean optimization solver (e.g. wbo or msuncore).

Current p2cudf architecture (MISC 2011)

Since MISC Live 3, Sat4j contains a specific lexicographic optimization scheme



Would require a lexico optimization OPB format to allow testing others OPB engine (created late 2014 for LION 9 competition).

Why changing the architecture ?

- Using a single objective function allows to easily try any OPB solver
- Such objective function can contain a huge number of literals and huge coefficients : p2cudf had sometimes problems to converge to the optimal solution
- Using a specific lexicographic procedure limit the size of the objective function
- a specific procedure must be implemented
- Solvers using a specific lexicographic procedure performed generally well in MISC 2010
- Sat4j PB has sometimes hard time to prove unsatisfiability, requiring proving each criterion optimal might be an issue

UNIVERSITÉ D'ARTOIS

Lessons learned from p2cudf

- CUDF input format is great to present software dependency problems use cases
- CUDF semantic has been very hard to implement in p2cudf
- Work done in making p2 work in Eclipse has been successfully reused (e.g. aggressive slicing stage)
- All improvements on the optimization function in p2 could not be reused in p2cudf
- MISC provides many use cases, and allowed us to spot limitations in p2
- p2cudf allows us to try ideas because it is a research tool

See p2cudf results at `http://www.mancoosi.org/misc/`

Outline

Conclusion

- Dependency management is NP-complete, you need a SAT solver in your package manager !
- Since 2008, Eclipse includes a SAT (Pseudo-Boolean) solver : Sat4j
- In 2016, Linux distributions still do not use SAT solvers for dependency management, despite the availability of mpm and aspcud
- Why ?

Conclusion

- Dependency management is NP-complete, you need a SAT solver in your package manager !
- Since 2008, Eclipse includes a SAT (Pseudo-Boolean) solver : Sat4j
- In 2016, Linux distributions still do not use SAT solvers for dependency management, despite the availability of mpm and aspcud
- Why ? developers vs researchers initiative !

Conclusion

- Dependency management is NP-complete, you need a SAT solver in your package manager !
- Since 2008, Eclipse includes a SAT (Pseudo-Boolean) solver : Sat4j
- In 2016, Linux distributions still do not use SAT solvers for dependency management, despite the availability of mpm and aspcud
- Why ? developers vs researchers initiative !
- Importance of “production ready” prototypes !

Thanks for your attention

Questions ?

Morgado, A., Heras, F., Liffiton, M. H., Planes, J., and Marques-Silva, J. (2013).
Iterative and core-guided maxsat solving : A survey and assessment.
Constraints, 18(4) :478–534.

Syrjänen, T. (1999).
A rule-based formal model for software configuration.
Master's thesis, Helsinki University of Technology.

Team, M. (2008).
http ://www.mancoosi.org/.

Team, W. (2005).
Report on formal management of software dependencies.
Technical report, Environment for the Development and Distribution of Open Source Software (EDOS) FP6-IST-004312.

Tucker, C., Shuffelton, D., Jhala, R., and Lerner, S. (2007).
Opium : Optimal package install/uninstall manager.
In *ICSE*, pages 178–188. IEEE Computer Society.

UNIVERSITÉ D'ARTOIS